Altova MapForce 2021 Enterprise Edition

ユーザーマニュアル

Altova MapForce 2021 Enterprise Edition
ユーザーマニュアル

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

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

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

公開日: 2015-2021

(C) 2015-2021 Altova GmbH

目次

1 はじめに 21

1.1 新規機能 22
1.2 サポートメモ 31
1.3 MapForce とは ? 36
1.4 基本概念 41
1.5 ユーザーインターフェイスの概要 43
1.6 規約 52

2 チュートリアル 53

2.1 XML を新しいスキーマに変換 54
2.2 複数のソースから1つのターゲットにマップ 64
2.3 複数のターゲットスキーマとの作業 70
2.4 ファイルを動的に処理と生成する 78

3 一般的なタスク 87

3.1 マッピングとの作業 88
3.1.1 マッピングにコンポーネントを追加する 88
3.1.2 URL からコンポーネントを追加する 89
3.1.3 データストリーミングについて 92
3.1.4 変換言語の選択 93
3.1.5 マッピングの検証 94
3.1.6 マッピング出力の検証 95
3.1.7 出力のプレビュー 96
3.1.8 テキストビューの機能 97
3.1.9 テキストビュー内の検索 101
3.1.10 XSLT コードのプレビュー 105
3.1.11 XSLT コードの生成 105
3.1.12 XQuery コードのプレビュー 106

3.1.13 複数のマッピングウィンドウとの作業 106
3.1.14 マッピング設定の変更 107
3.2 コンポーネントとの作業 111
3.2.1 コンポーネント内の検索 112
3.2.2 コンポーネントの整列 113
3.2.3 コンポーネント設定を変更する 114
3.2.4 入力の複製 115
3.3 接続との作業 117
3.3.1 必須の入力について 118
3.3.2 優先する接続線の表示の変更 120
3.3.3 接続の注釈 120
3.3.4 接続設定 121
3.3.5 一致する子要素の接続 123
3.3.6 コンテキストメニューの接続 126
3.3.7 見つからない親接続の通知 127
3.3.8 接続と子接続の移動 128
3.3.9 コンポーネント削除後の接続の保持 131
3.3.10 見つからないアイテムの扱い方 133
3.4 マッピングプロジェクトとの作業 137
3.4.1 プロジェクトを開く、検索する、閉じる 138
3.4.2 新規プロジェクトの作成 138
3.4.3 コード生成設定の設定 140
3.4.4 プロジェクトフォルダーの管理 141

4 マッピングのデザイン 143
4.1 相対と絶対パスの使用 145
4.1.1 コンポーネント上で相対パスを使用する 145
4.1.2 ファイルベースのデータベースへパスを設定する 147
4.1.3 破損したパスの参照を修正する 149
4.1.4 多種の実行環境内のパス 150
4.1.5 コピーと貼り付けと相対パス 151
4.2 マッピング接続の種類 152
4.2.1 ターゲット優先マッピング 152
4.2.2 ソース優先マッピング 152

4.2.3 全てコピー接続 160
4.3 チェーンマッピング 162
4.3.1 例: パススルーが有効な場合 163
4.3.2 例: パススルーが無効な場合 167
4.4 複数の入力または出力ファイルを動的に処理 170
4.4.1 複数の入力ファイルを単一の出力ファイルにマップする 172
4.4.2 複数の入力ファイルを複数の出力ファイルにマッピングする 174
4.4.3 ファイル名をマッピングパラメーターとして提供する 174
4.4.4 複数の出力ファイルをプレビューする 175
4.4.5 例: 1つの XML ファイルを複数のファイルに分割 175
4.4.6 例: データベーステーブルに対して複数の XML を生成 177
4.5 マッピングにパラメーターを与える 179
4.5.1 単純型入力コンポーネントの追加 180
4.5.2 単純型入力コンポーネント設定 181
4.5.3 デフォルトの入力値を作成する 183
4.5.4 例: ファイル名をマッピングパラメーターとして使用する 184
4.6 マッピングから文字列の値を返す 188
4.6.1 単純型出力コンポーネントの追加 189
4.6.2 例: 関数出力のプレビュー 190
4.7 変数の使用 192
4.7.1 変数の追加 193
4.7.2 変数のコンテキストとスコープの変更 197
4.7.3 例: データベースのテーブルの行の計算 199
4.7.4 例: ノードのフィルターと番号付け 200
4.7.5 例:記録のグループ化、および、サブグループ化 201
4.8 データの並べ替え 203
4.8.1 複数のキーを使用した並べ替え 205
4.8.2 変数を使用して並べ替える 206
4.9 データのグループ分け 208
4.9.1 例: キー別にレコードをグループ分けする方法 211
4.10 フィルターと条件 214
4.10.1 例: ノードのフィルター 215
4.10.2 例: 条件付で値を返す 217
4.11 データのジョイン 220
4.11.1 ジョインの条件の追加 222

4.11.2 3つ以上の構造をジョインする ... 225
4.11.3 例: XML 構造のジョイン ... 226
4.12 Value-Map の使用 ... 232
4.12.1 例: 曜日の置換 ... 235
4.12.2 例: 職位の置換 ... 237
4.13 例外の追加 ... 241
4.13.1 例: "より大きい" 条件の例外 ... 242
4.13.2 例: ノードが存在しない場合の例外 ... 242
4.14 文字列の解析とシリアル化 ... 245
4.14.1 コンポーネントの解析/シリアル化について ... 245
4.14.2 例: 文字列の解析 (固定長テキストから Excel へ) ... 247
4.14.3 例: 文字列にシリアル化 (XML からデータベースへ) ... 252
4.15 ノード名のマッピング ... 257
4.15.1 ノード名へのアクセスを取得する ... 258
4.15.2 特定の型のノードへのアクセス ... 265
4.15.3 例: 要素名を属性値にマップする ... 269
4.15.4 例: 名前別にノードをグループ化またはフィルターする ... 273
4.16 認証を使用してマッピングを実行する方法 ... 278
4.16.1 資格情報 ... 278
4.16.2 動的な認証 ... 288
4.16.3 保管されたユーザー名とパスワード ... 290
4.17 StyleVision を使用してマッピングの出力のスタイルを指定する方法 ... 292
4.17.1 StyleVision スタイルシートを使用したマッピングの例 ... 293
4.18 マッピングドキュメンテーションの生成 ... 296
4.18.1 定義済みの StyleVision パワースタイルシート ... 297
4.18.2 カスタムスタイルシート ... 302
4.19 マッピングのルールと戦略 ... 304
4.19.1 シーケンス ... 305
4.19.2 マッピングコンテキスト ... 306
4.19.3 優先コンテキスト ... 316
4.19.4 複数のターゲットコンポーネント ... 321

5 マッピングのデバッグ 324

5.1 デバッガーの準備 ... 327

5.2 デバッガーコマンド ... 328
5.3 デバッグモードについて ... 329
5.4 ブレークポイントの追加と削除 ... 332
5.5 値ウィンドウの使用 ... 335
5.6 コンテキストウィンドウの使用 ... 337
5.7 ブレークポイントウィンドウの使用 ... 339
5.8 部分的に生成された出力のプレビュー ... 341
5.9 コネクタの現在の値を表示する ... 342
5.10 最近の過去にステップバックする ... 343
5.11 コネクタにより処理された値の履歴を確認する ... 344
5.12 コンテキストを値に設定する ... 345
5.13 デバッガー設定 ... 346

6 データソースとターゲット 347
6.1 XML と XML スキーマ ... 348
6.1.1 XML スキーマの生成 ... 348
6.1.2 XML コンポーネント設定 ... 349
6.1.3 "スキーマ" コンポーネントとしてを DTD 使用する ... 353
6.1.4 派生した XML スキーマの型 ... 353
6.1.5 QNames ... 356
6.1.6 Nil の値 / Nillable ... 357
6.1.7 コメントと処理命令 ... 359
6.1.8 CDATA セクション ... 360
6.1.9 ワイルドカード – xs:any / xs:anyAttribute ... 362
6.1.10 複数のスキーマからのデータのマージ ... 367
6.1.11 カスタム名前空間の宣言 ... 368
6.1.12 デジタル署名 ... 371
6.2 MapForce とデータベース ... 378
6.2.1 データソースへの接続 ... 380
6.2.2 データベースマッピングの概要 ... 464
6.2.3 データをデータベースにマップする ... 495
6.2.4 データベースのデータのジョイン ... 545
6.2.5 データベースデータのフィルターと並べ替え (SQL WHERE/ORDER) ... 561
6.2.6 仮想テーブルとしての SQL SELECT ステートメント ... 567

6.2.7 XML データから / データベースフィールドへのマッピング 575
6.2.8 データベースの参照とクエリ 586
6.2.9 ストアドプロシージャ 603
6.2.10 トランザクションロールバックの処理 628
6.2.11 データベースのトレースとエラーのログ 638
6.3 CSV とテキストファイル 647
6.3.1 例: CSV ファイルから XML へのマッピング 647
6.3.2 例: アイテムの反復 650
6.3.3 例: CSV と FLF ファイルから階層を作成する 652
6.3.4 CSV オプションの設定 655
6.3.5 例: 固定長テキストファイルを(データベースへ)マッピング 659
6.3.6 FLF オプションの設定 666
6.4 MapForce FlexText 672
6.4.1 概要 672
6.4.2 FlexText チュートリアル 675
6.4.3 FlexText コンポーネント設定 687
6.4.4 ターゲットコンポーネントとして FlexText を使用する 689
6.4.5 FlexText レファレンス 690
6.4.6 FlexText と正規表現 721
6.5 EDI 728
6.5.1 EDI 用語解説 730
6.5.2 EDI コンポーネントをマッピングに追加する 731
6.5.3 EDI マッピングの実行 732
6.5.4 EDI コンポーネント設定 733
6.5.5 EDI コンポーネント検証 736
6.5.6 EDI から XML を生成する 740
6.5.7 ASC X12 743
6.5.8 HIPAA X12 747
6.5.9 HL7 Version 2 751
6.5.10 IATA PADIS 753
6.5.11 NCPDP SCRIPT 757
6.5.12 SAP IDoc 759
6.5.13 SWIFT MT 762
6.5.14 SWIFT MX / ISO 20022 769
6.5.15 TRADACOMS 769

6.5.16 UN/EDIFACT ... 782
6.5.17 EDI 構造のカスタム化 ... 793
6.5.18 EDI 検証のカスタマイズ ... 821
6.6 JSON ... 831
6.6.1 JSON ファイルをマッピングコンポーネントとして追加する ... 832
6.6.2 JSON コンポーネント設定 ... 833
6.6.3 JSON5 へのサポート ... 836
6.6.4 JSON Lines ... 838
6.6.5 例: JSON を CSV に変換する ... 840
6.6.6 例: Excel を JSON に変換する ... 844
6.7 MS OOXML Excel 2007 以降のファイル ... 849
6.7.1 Excel 2007+ コンポーネントについて ... 851
6.7.2 Excel 2007+ ファイルをマッピングコンポーネントとして追加する ... 853
6.7.3 ワークシートの追加と削除 ... 854
6.7.4 行の範囲の追加と移動 ... 856
6.7.5 セルの範囲の選択 ... 857
6.7.6 既存の行の間に行を挿入する ... 861
6.7.7 Excel 2007+ コンポーネント設定 ... 863
6.7.8 例:Excel 2007+ を XML にマッピングする ... 866
6.7.9 例: Excel 行を XML ファイルに変換する ... 868
6.7.10 例: データベースデータを Excel 2007+ にマッピング ... 872
6.7.11 例: 既存の Excel ワークシートの更新 ... 874
6.8 XBRL ... 878
6.8.1 XBRL ファイルをマッピングコンポーネントとして追加する ... 879
6.8.2 インライン XBRL からデータを読み込む ... 880
6.8.3 XBRL コンポーネントアイテムについて ... 882
6.8.4 構造ビューの選択 ... 883
6.8.5 XBRL タクソノミマネジャー ... 885
6.8.6 カスタム XBRL タクソノミ パッケージ ... 901
6.8.7 XBRL 設定と優先順位 ... 903
6.8.8 XBRL デフォルトと作業する ... 910
6.8.9 XBRL ハイパーキューブとの作業 ... 914
6.8.10 XBRL テーブルと作業する ... 921
6.8.11 XBRL マッピングのサンプル ... 927
6.9 HL7 Version 3 ... 932

6.10 プロトコルバッファー ... 934
6.10.1 バイナリファイルをマッピングに追加する ... 935
6.10.2 プロトコルバッファーコンポーネントの設定 ... 938
6.10.3 例: プロトコルバッファーからデータを読み取る ... 939
6.10.4 例: プロトコルバッファーにデータを書き込む ... 944
6.11 バイナリファイル ... 950
6.11.1 例: バイナリファイルの読み取り ... 951
6.11.2 例: バイナリファイルの書き込み ... 953

7 関数 955
7.1 使用方法 ... 956
7.1.1 マッピングにビルトイン関数を追加する ... 956
7.1.2 マッピングに定数を追加する ... 958
7.1.3 関数の検索 ... 959
7.1.4 関数の型と詳細を確認する ... 960
7.1.5 関数引数の追加、または、削除 ... 961
7.2 デフォルトとノード関数 ... 962
7.2.1 デフォルトとノード関数の作成方法 ... 962
7.2.2 既存のルールの編集と削除 ... 966
7.2.3 デフォルトとノード関数のしくみ ... 967
7.2.4 ノード関数とデフォルトを条件付きで適用する ... 973
7.2.5 ノードメタデータをノード関数に提供する ... 978
7.2.6 例: 空の CSV フィールドの置換 ... 981
7.3 ユーザー定義関数 ... 986
7.3.1 ユーザー定義関数の作成 ... 988
7.3.2 ユーザー定義関数内のパラメーター ... 991
7.3.3 インラインと正規ユーザー定義関数 ... 996
7.3.4 ユーザー定義関数のナビゲート ... 998
7.3.5 ユーザー定義関数の編集 ... 998
7.3.6 ユーザー定義関数の削除 ... 999
7.3.7 ユーザー定義関数の呼び出しとインポート ... 1000
7.3.8 マッピング間で UDF をコピーし張り付ける用法 ... 1002
7.3.9 例: ルックアップと連結 ... 1002
7.3.10 例:再帰的な検索 ... 1006

7.4 ...カスタム XSLT 1.0 または 2.0 関数のインポート ... 1010
7.4.1 例: カスタム XSLT 関数の追加 ... 1011
7.4.2 例: ノードの値の集計 ... 1014
7.5 ...カスタム XQuery 1.0 関数のインポート ... 1017
7.5.1 例: カスタム XQuery 関数のインポート ... 1018
7.6 ...カスタム Java と .NET ライブラリのインポート ... 1022
7.6.1 例: カスタム Java クラスのインポート ... 1025
7.6.2 例: カスタム .NET DLL アセンブリのインポート ... 1027
7.7 ...Java、C# および C++ ライブラリを手動で参照する ... 1029
7.7.1 .mff ファイルの構成 ... 1029
7.7.2 .mff ファイルを MapForce にインポートする ... 1034
7.7.3 データ型のマッピング ... 1035
7.7.4 例: C# を .mff に適用する方法 ... 1037
7.7.5 例: C++ を .mff に適用する方法 ... 1038
7.7.6 例: Java を .mff に適用する方法 ... 1040
7.8 ...関数ライブラリの管理 ... 1043
7.8.1 ローカルとグローバルライブラリ ... 1045
7.8.2 相対的なライブラリパス ... 1046
7.9 ...正規表現 ... 1048
7.10 ...関数ライブラリレファレンス ... 1052
7.10.1 core | aggregate functions (集計関数) ... 1069
7.10.2 core | conversion functions (変換関数) ... 1075
7.10.3 core | file path functions (ファイルパス関数) ... 1090
7.10.4 core | generator functions (ジェネレーター関数) ... 1094
7.10.5 core | logical functions (論理関数) ... 1096
7.10.6 core | math functions (数学関数) ... 1102
7.10.7 core | node functions (ノード関数) ... 1107
7.10.8 core | QName functions (QName 関数) ... 1112
7.10.9 core | sequence functions (シーケンス関数) ... 1115
7.10.10 core | string functions (文字列関数) ... 1139
7.10.11 db ... 1152
7.10.12 edifact ... 1155
7.10.13 lang | datetime functions (日付時刻関数) ... 1159
7.10.14 lang | file functions (ファイル関数) ... 1179
7.10.15 lang | generator functions (ジェネレーター関数) ... 1180

7.10.16 lang | logical functions (論理関数) ... 1181
7.10.17 lang | math functions (数学関数) ... 1183
7.10.18 lang | QName functions (QName 関数) ... 1191
7.10.19 lang | string functions (文字列関数) ... 1192
7.10.20 mime ... 1206
7.10.21 xbrl ... 1210
7.10.22 xlsx ... 1212
7.10.23 xpath2 | accessors ... 1216
7.10.24 xpath2 | anyURI functions (anyURI 関数) ... 1218
7.10.25 xpath2 | boolean functions (ブール値関数) ... 1218
7.10.26 xpath2 | constructors ... 1219
7.10.27 xpath2 | context functions (コンテキスト関数) ... 1220
7.10.28 xpath2 | durations, date and time functions (期間、日付、および時刻関数) ... 1223
7.10.29 xpath2 | node functions (ノード関数) ... 1238
7.10.30 xpath2 | numeric functions (数値関数) ... 1244
7.10.31 xpath2 | string functions (文字列関数) ... 1246
7.10.32 xpath3 | external information functions ... 1256
7.10.33 xpath3 | formatting functions ... 1258
7.10.34 xpath3 | math functions ... 1262
7.10.35 xpath3 | URI functions ... 1268
7.10.36 xslt | xpath functions (xpath 関数) ... 1270
7.10.37 xslt | xslt functions (xslt 関数) ... 1273

8 SOAP Web サービスの実装 1277
8.1 ...SOAP/WSDL サポート情報 ... 1280
8.2 ...WSDL ファイルから SOAP Webサービスプロジェクトを作成 ... 1282
8.3 ...SOAP Web サービス フォールトの定義 ... 1291
8.4 ...Java SOAP Web サービスの指定 ... 1293
8.5 ...例: SOAP Web サービスの生成 (C#) ... 1295
8.6 ...例: SOAP Web サービスの生成 (Java) ... 1308

9 Web サービスの呼び出し 1315
9.1 ...Web サービス呼び出しの追加 (REST-スタイル) ... 1317

9.1.1 Web サービスパラメーターの定義 1321
9.1.2 Web サービスパラメーターを URL からインポートする 1326
9.1.3 Web サービス情報を WADL からインポートする 1327
9.1.4 リクエスト構造の定義 1328
9.1.5 レスポンス構造の定義 1330
9.1.6 HTTP レスポンスを条件付きで処理する方法 1331
9.2 ...Web サービス呼び出しの追加 (SOAP) 1334
9.3 ...Web サービス呼び出し設定 1335
9.4 ...HTTP セキュリティの設定 1339
9.5 ...WS-Security の設定 1341
9.6 ...例: REST-スタイル Web サービスの呼び出し 1344
9.7 ...例: RSS フィードからのデータのマッピング 1349
9.8 ...例: SOAP Web サービスの呼び出し 1354
9.9 ...例: OAuth 2.0 認証 1358
9.10 ...デジタル証明書の管理 1370
9.10.1 Linux 上でサーバー証明書を信頼する方法 1372
9.10.2 macOS 上でサーバー証明書を信頼する方法 1373
9.10.3 Windows 上でサーバー証明書を信頼する方法 1374
9.10.4 Windows 上で証明書ストアにアクセスする方法 1375
9.10.5 Windows から証明書をエクスポートする方法 1378
9.10.6 Linux 上のクライアントの証明書 1384
9.10.7 macOS 上のクライアントの証明書 1385
9.10.8 Windows 上のクライアント証明書 1386

10 マッピングの自動化と MapForce 1388
10.1 ...RaptorXML Server を使用して自動化する 1389
10.2 ...MapForce Server を使用して自動化する 1390
10.3 ...サーバーでの実行のためにマッピングを準備する 1391
10.4 ...MapForce Server 実行可能ファイルにマッピングをコンパイルする 1396
10.5 ...マッピングを FlowForce Server にデプロイする 1399
10.6 ...AS2 統合 1403
10.7 ...MapForce コマンドラインインターフェイス 1408

11 Altova グローバルリソース 1413
11.1 ...グローバルリソースの作成 1414
11.2 ...グローバルリソースとしてのデータベース 1415
11.3 ...グローバルリソースとしての MapForce と StyleVision 変換結果 1419
11.4 ...グローバルリソース XML ファイル 1420
11.5 ...多種の実行環境内でのグローバルリソース 1421
11.5.1 SLT、XSLT2、XQuery 内のグローバルリソース 1421
11.5.2 C++、C#、Java 内のグローバルリソース 1421
11.5.3 MapForce Server 内のグローバルリソース 1421
11.5.4 FlowForce Server 内のグローバルリソース 1422
11.5.5 FlowForce Server にリソースをデプロイする 1423
11.6 ...例: 変数入力ファイルを持つマッピングを実行する 1425
11.7 ...例: 出力を変数フォルダに出力する 1427
11.8 ...例: データベースの切り替え 1429
11.9 ...例: アプリケーションワークフローを作成する 1433

12 MapForce のカスタマイズ 1438
12.1 ...MapForce オプションの変更 1438
12.1.1 Java 設定 1439
12.1.2 ネットワークプロキシ設定 1440
12.2 ...キーボードのショートカット 1443
12.2.1 ショートカットのカスタム化 1444
12.3 ...メニューをカスタム化する方法 1446
12.4 ...カタログファイル 1448

13 Visual Studio のための MapForce プラグイン 1452
13.1 ...プラグインの有効化 1453
13.2 ...マッピングとプロジェクトと作業 1455
13.3 ...共通メニューと関数へのアクセス 1456

14 Eclipse のための MapForce プラグイン 1458

14.1 ...Eclipse のための MapForce プラグインのインストール ... 1459
14.2 ...MapForce パースペクティブ ... 1463
14.3 ...共通メニューと関数へのアクセス ... 1466
14.4 ...マッピングおよびプロジェクトと作業 ... 1469
14.4.1 MapForce/Eclipse プロジェクトの作成 ... 1469
14.4.2 新しいマッピングの作成 ... 1471
14.4.3 Eclipse プロジェクトに既存のマッピングをインポートする ... 1473
14.4.4 自動ビルドの構成と MapForce コードの生成 ... 1476
14.5 ...Eclipse のための MapForce プラグインの有効化 ... 1479

15 メニューレファレンス 1480
15.1 ...ファイル ... 1481
15.2 ...編集 ... 1484
15.3 ...挿入 ... 1485
15.4 ...プロジェクト ... 1488
15.5 ...コンポーネント ... 1490
15.6 ...接続 ... 1492
15.7 ...関数 ... 1493
15.8 ...出力 ... 1494
15.9 ...デバッグ ... 1496
15.10 ..表示 ... 1497
15.11 ..ツール ... 1499
15.12 ..ウィンドウ ... 1500
15.13 ..ヘルプメニュー ... 1501

16 Code Generator 1506
16.1 ...What's new 1508
16.2 ...Generating C++ code ... 1510
16.2.1 Generating code from a mapping ... 1511
16.2.2 Generating code from a mapping project ... 1512
16.2.3 Building the project ... 1512
16.2.4 Running the application ... 1513
16.3 ...Generating C# code ... 1515

16.3.1 Selecting the target platform ... 1516
16.3.2 Generating code from a mapping ... 1517
16.3.3 Generating code from a mapping project ... 1518
16.3.4 Building .NET Framework projects ... 1519
16.3.5 Building .NET 5 and .NET Core projects ... 1520
16.4 ...Generating Java code ... 1522
16.4.1 Generating code from a mapping ... 1523
16.4.2 Generating code from a mapping project ... 1524
16.4.3 Building the project with Ant ... 1524
16.4.4 Handling JDBC references ... 1526
16.4.5 Example: Generate and run Java code ... 1527
16.5 ...Integrating MapForce-Generated Code ... 1536
16.5.1 Java example ... 1537
16.5.2 C# example ... 1539
16.5.3 C++ example ... 1541
16.5.4 Changing Input and Output Programmatically ... 1542
16.6 ...Generating Code from XML Schemas or DTDs ... 1548
16.6.1 About Schema Wrapper Libraries (C++) ... 1550
16.6.2 About Schema Wrapper Libraries (C#) ... 1553
16.6.3 About Schema Wrapper Libraries (Java) ... 1555
16.6.4 Integrating Schema Wrapper Libraries ... 1557
16.6.5 Example: Book Library ... 1560
16.6.6 Example: Purchase Order ... 1582
16.7 ...Reference to Generated Classes (C++) ... 1589
16.7.1 altova::DateTime ... 1589
16.7.2 altova::Duration ... 1592
16.7.3 altova::DayTimeDuration ... 1594
16.7.4 altova::YearMonthDuration ... 1595
16.7.5 altova::meta::Attribute ... 1595
16.7.6 altova::meta::ComplexType ... 1596
16.7.7 altova::meta::Element ... 1597
16.7.8 altova::meta::SimpleType ... 1597
16.7.9 [YourSchema]::[CDoc] ... 1598
16.7.10 [YourSchema]::[ElementType] ... 1601
16.7.11 [YourSchema]::MemberAttribute ... 1602

16.7.12 [YourSchema]::MemberElement ... 1602
16.8 Reference to Generated Classes (C#) ... 1604
16.8.1 Altova.Types.DateTime ... 1604
16.8.2 Altova.Types.DateTimeFormat ... 1607
16.8.3 Altova.Types.Duration ... 1609
16.8.4 Altova.Xml.Meta.Attribute ... 1611
16.8.5 Altova.Xml.Meta.ComplexType ... 1612
16.8.6 Altova.Xml.Meta.Element ... 1612
16.8.7 Altova.Xml.Meta.SimpleType ... 1613
16.8.8 [YourSchema].[Doc] ... 1614
16.8.9 [YourSchema].[ElementType] ... 1616
16.8.10 [YourSchemaType].MemberAttribute ... 1617
16.8.11 [YourSchemaType].MemberElement ... 1617
16.9 Reference to Generated Classes (Java) ... 1619
16.9.1 com.altova.types.DateTime ... 1619
16.9.2 com.altova.types.Duration ... 1623
16.9.3 com.altova.xml.meta.Attribute ... 1626
16.9.4 com.altova.xml.meta.ComplexType ... 1627
16.9.5 com.altova.xml.meta.Element ... 1627
16.9.6 com.altova.xml.meta.SimpleType ... 1628
16.9.7 com.[YourSchema].[Doc] ... 1629
16.9.8 com.[YourSchema].[ElementType] ... 1630
16.9.9 com.[YourSchema].[YourSchemaType].MemberAttribute ... 1632
16.9.10 com.[YourSchema].[YourSchemaType].MemberElement ... 1632
16.10 Code Generation Tips ... 1634
16.11 Code Generator Options ... 1635
16.12 SPL Reference ... 1637
16.12.1 Basic SPL structure ... 1637
16.12.2 Declarations ... 1638
16.12.3 Variables ... 1640
16.12.4 Predefined variables ... 1641
16.12.5 Creating output files ... 1642
16.12.6 Operators ... 1644
16.12.7 Conditions ... 1644
16.12.8 Collections and foreach ... 1645

16.12.9 Subroutines ... 1646
16.12.10 Built in Types ... 1649

17 The MapForce API 1654
17.1 ...Accessing the API ... 1655
17.2 ...The Object Model ... 1658
17.3 ...Error Handling ... 1659
17.4 ...Example C# Project ... 1661
17.5 ...Example Java Project ... 1665
17.6 ...JScript Examples ... 1668
17.6.1 Start Application ... 1668
17.6.2 Simple Document Access ... 1669
17.6.3 Generate Code ... 1670
17.6.4 Generate Code (Alternative) ... 1672
17.6.5 Run a Mapping ... 1673
17.6.6 Project Tasks ... 1677
17.7 ...Object Reference ... 1681
17.7.1 Interfaces ... 1681
17.7.2 Enumerations ... 1821

18 ActiveX Integration 1828
18.1 ...必要条件 ... 1829
18.2 ...ActiveX コントロールをツールボックスに追加する ... 1831
18.3 ...アプリケーションレベルの統合 ... 1833
18.4 ...ドキュメントレベルの統合 ... 1836
18.5 ...ActiveX 統合のサンプル ... 1839
18.5.1 C# ... 1839
18.5.2 HTML ... 1845
18.5.3 Java ... 1852
18.5.4 VB.NET ... 1861
18.6 ...コマンド レファレンス ... 1864
18.6.1 「ファイル」メニュー ... 1864
18.6.2 「編集」メニュー ... 1865

18.6.3 「挿入」メニュー ... 1866
18.6.4 「プロジェクト」メニュー ... 1866
18.6.5 「コンポーネント」メニュー ... 1867
18.6.6 「接続」メニュー ... 1868
18.6.7 「関数」メニュー ... 1869
18.6.8 「出力」メニュー ... 1869
18.6.9 「デバッグ」メニュー ... 1870
18.6.10 「表示」メニュー ... 1870
18.6.11 「ツール」メニュー ... 1871
18.6.12 「ヘルプ」メニュー ... 1872
18.6.13 「ヘルプ」メニュー ... 1872
18.7 ...オブジェクトレファレンス ... 1873
18.7.1 MapForceCommand ... 1873
18.7.2 MapForceCommands ... 1875
18.7.3 MapForceControl ... 1876
18.7.4 MapForceControlDocument ... 1883
18.7.5 MapForceControlPlaceHolder ... 1890
18.7.6 列挙 ... 1892

19 付録 1894

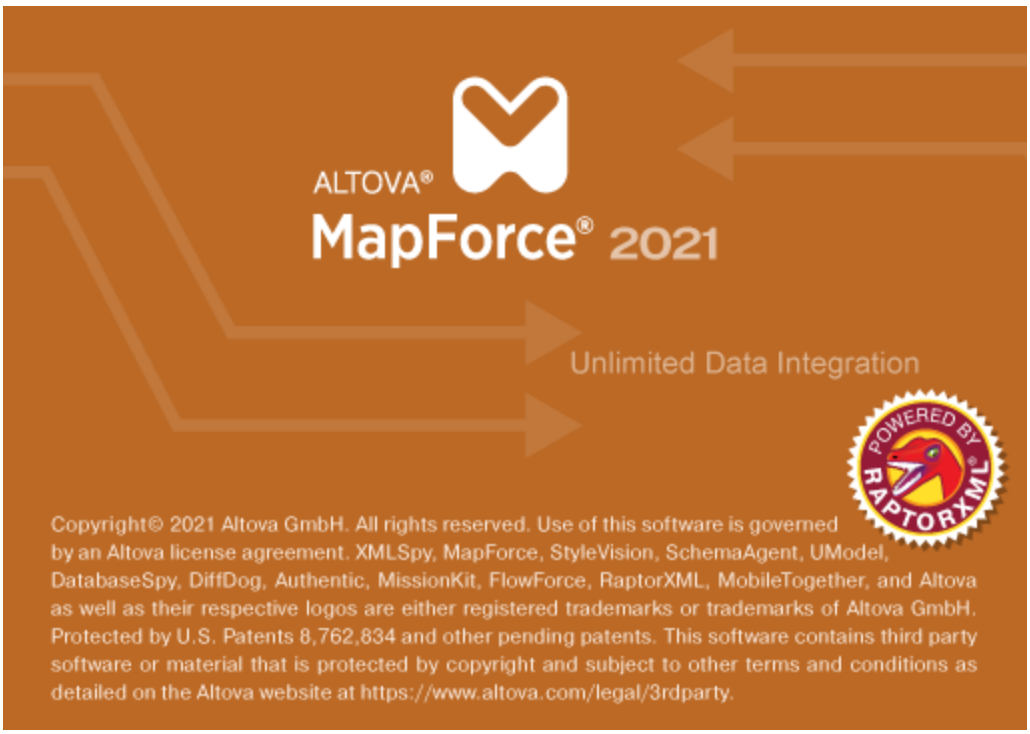
19.1 ...エンジン情報 ... 1895
19.1.1 XSLT および XQuery エンジンに関する情報 ... 1895
19.1.2 XSLT と XPath/XQuery 関数 ... 1899
19.2 ...技術データ ... 2001
19.2.1 OS とメモリ要件 ... 2001
19.2.2 Altova XML バリデーター ... 2001
19.2.3 Altova XSLT と XQuery エンジン ... 2001
19.2.4 Unicode のサポート ... 2001
19.2.5 インターネットの使用 ... 2002
19.3 ...ライセンス情報 ... 2003
19.3.1 電子的なソフトウェアの配布 ... 2003
19.3.2 ソフトウェアのアクティベーションとライセンスの計測 ... 2003
19.3.3 エンドユーザー使用許諾契約書 ... 2004

20　用語　2005

インデックス　2008

1 はじめに

MapForce® 2021 Enterprise Edition は高度なデータ統合プロジェクトにて決定的な役割を果たす視覚的なデータマッピングツールです。MapForce® は、プラットフォーム更新済みのWindows 7 SP1、Windows 8、Windows 10 とプラットフォーム更新済みのWindows Server 2008 R2 SP1 またはそれ以降 上で作動する32/64-bit Windows アプリケーションです。64 ビットサポートは、Enterprise とProfessional エディションでご利用いただくことができます。MapForce はVisual Studio とEclipse、および Microsoft Office 製品と統合することもできます。サポートメモを参照してください。



最終更新日: 2021 年 02 月 24 日

1.1 新規機能

MapForce 2021 リリース2 の新規機能:

- C# コードを生成する際 .NET Core 3.1 と.NET 5.0 をコード生成オプション からターゲットフレームワークとして選択することができます(.NET Framework プロジェクトに既存のサポートを追加します)。詳細に関してはC# コードの生成を参照してください。
- XSLT 3.0 はマッピング言語としてサポートされるようになりました。XSLT コードの生成 を参照してください。また、MapForce にはマッピング言語がXSLT 3.0 である場合サポートされる新規のビルトイン関数が搭載されています。関数ライブラリルファレンスを参照してください。
- 内部のアップデートと最適化。

MapForce 2021 の新規機能:

- 簡素化されたXBRL タクソノミマネジャーおよびリリースから独立したタクソノミアップデートはMapForce を含むAltova のすべてのXBRL-有効化済みアプリケーションのために役に立っています。現在インストールされているXBRL タクソノミをビュー、管理、および更新することのできる新規のXBRL タクソノミマネジャーをこのために使用することができるようになりました。
- REST W eb サービスを呼び出す際、W eb サービスのURL をマッピングのパラメーターとして提供できます。このオプションはURL に対して完全なコントロールが必要でマッピングのランタイムに指定する必要がある場合追加の柔軟性を提供します。Web サービスの呼び出しの追加(REST-スタイル)を参照してください。
- SWIFT MT フォーマットからのマッピング、またはフォーマットへのマッピングへのサポート。
- カスタムEDI コレクションを希望するディレクトリまたはユーザーのプロファイルディレクトリに保管することができます。カスタムEDI コレクションの作成を参照してください。
- マッピングがターゲット EDI コンポーネントにデータを書き込む際、EDI 出力の代わりにXML を生成するようにコンポーネントを切り替えることができます。これを行うことにより、生成されたデータをプレビューする代替の方法が与えられます。ボタンを素早くクリックしてEDI ファイルをXML フォーマットに変換することができます。これはXML スキーマまたはマッピングの接続を必要としません。詳細に関しては EDI をXML に変換する を参照してください。
- MapForce マッピングはBLOB (バイナリラージオブジェクト) データをバイナリファイルから読み取り、バイナリファイルをディスクに書き込むことができます。これは、例えば、BLOB フィールドをデータベースから読み取り、ディスク上でイメージファイルとして保存すること、または、PDF などのバイナリファイルをディスクから読み取りXML ファイル内の `xs:base64Binary` フィールドとして保存することが可能になります。詳細に関してはバイナリファイルを参照してください。
- 新規のデータベースバージョンがサポートされています: MariaDB 10.4、10.5
- 新規のEclipse バージョンがサポートされています: 2019.09, 2019.12, 2020.03, 2020.06
- マッピング内でSQL ジョインコンポーネントを使用して複数のデータベーステーブルをジョインまたはビューする場合、LEFT OUTER JOIN またはINNER JOIN としてジョインモードを設定できます。ジョインモードの変更を参照してください。
- 内部のアップデートと最適化

2020 Release 2 の新規機能:

- 新規のライブラリの管理 ウィンドウによりドキュメントまたはプログラムレベルでインポートされたすべての関数ライブラリを表示管理できるようになりました(これにはMapForce ユーザー定義関数と他のライブラリが含まれています)。例えば、1つのマッピングから他のマッピングに簡単にユーザー定義関数をコピーして貼り付けることができます。マッピング間でUDF をコピーし張り付ける用法を参照してください。
- マッピングファイルがライブラリをインポートする際に、インポートされたライブラリファイルのパスはデフォルトで相対パスです。相対的なライブラリパスを参照してください。以前のリリース同様アプリケーションレベルでマッピングファイルをインポートすることは可能ですが、この場合ライブラリのパスは絶対パスです。
- マッピングファイルがXSLT またはXQuery ライブラリをインポートする場合、相対的なパスを使用してインポートされたライブラリを参照するXSLT またはXQuery コードを生成することができます。新規のオプションはマッピング設定 ダイアログボックス内で使用することができます。
- MapForce API はインポートされたライブラリをプログラミング的に管理できる機能により拡張されました(例えば、ライブラリの追加または削除機能など)。

- XML スキーマラッパーライブラリのために生成されたコードは要素名前空間とプレフィックスを管理する機能を提供します。例 Purchase Order を参照してください。
- 新規のデータベースバージョンがサポートされます: PostgreSQL 12.1 とInformix 14.10.
- (変換言語がBUILT-IN の場合)マッピングからのExcel ファイルの生成に加え MapForce は既存のExcel ファイルの更新をサポートします。例: 既存のExcel シートの更新 を参照してください。
- UN/EDIFACT D.19A とD19B Directories のためのサポート。EDI を参照してください。
- OAuth 2.0 資格情報を作成する際、POST リクエストボディ内で認証の詳細を構成することができます。これは既にサポートされるPOST リクエストヘッダー内で認証の詳細を提供する標準のメソッドに加えられたオプショナルのメソッドです。OAuth 2.0 資格情報を参照してください。
- 内部のアップデートと最適化。

MapForce リリース2020 の新規機能:

- マッピングにより OAuth 2.0 承認 を必要とするサービスを呼び出すことが可能になりました。
- 資格情報オブジェクトを使用してHTTP またはOAuth 認証の詳細を他の実行環境に移行することが可能になりました。資格情報を参照してください。
- JSON ライン テキストフォーマットを使用してJSON ファイルから、またはJSON ファイルへのデータのマッピングをサポートします。
- Visual Studio のためのMap Force プラグイン とコード生成内のVisual Studio 2019 へのサポート。
- Eclipse 4.9 – 4.12 のためのサポート。Eclipse のためのMapForce プラグインを参照してください
- Oracle パッケージにパブリックストアドプロシージャ、または関数が含まれる場合、これらはマッピングでも使用することができます。マッピングへストアドプロシージャを追加する方法 を参照してください。
- データベースオブジェクト名が特定のスキーマではなくデフォルトのスキーマに対して相対的に扱われるようにデータベースコンポーネントを構成することができます。この機能により異なるデータベース間を切り替える時間を節約することができます。データベースとスキーマ間の切り替えを参照してください。
- MapForce 内で作成されたグローバルリソースをFlowForce Server にデプロイすることができます。FlowForce Server にリソースをデプロイを参照してください。
- ルックアップテーブルを使用して値を置き換える場合、テーブルデータ(キー値ペア) をCSV またはExcel などの外部ソースからマッピングに張り付けることができます。また、代替の使用を必要としない値や不足する関数など定義済みのルックアップテーブル内で値が見つからない場合、このような値の処理が `substitute-missing` 関数の使用を必要としないため扱いが簡単になります。値マップの使用を参照してください。
- 内部のアップデートと最適化

MapForce 2019 リリース3 の新規機能:

- データベーストレースとエラーのログ のためのサポート
- 生成されたプログラムコード内およびサーバー実行内でのデータベーストランザクションの処理のためのサポート。マッピングがデータベースを更新する際にエラーが発生すると、操作の一部が失敗しても、マッピングの実行を継続することができます。 データベース例外の扱いを参照してください。
- Web サービスの拡張機能は以下を含みます:
  - Web サービスにより返されるHTTP ステータスコードをマップする能力 (RESTful Web サービスに適用することができます)
  - マッピングからWeb サービスのユーザー名、パスワード、およびURL を提供する能力 (WSDL-スタイルとRESTful Web サービスに適用されます)
  - マッピングからWeb サービスの生ボディ(MIME エンティティ) を作成する機能、または新規のmime 関数を使用して、Web サービスから返される生ボディをマップする機能。RESTful Web サービスのために適用することができます。
  - エラーが発生した場合、(RESTful Web サービスのために) Web サービスにより返されるレスポンスをマップする処理能力。

  詳細に関しては、Web サービスの呼び出しを参照してください。
- 新規のcharset-encode とcharset-decode 関数を使用して、マッピングはBase64 データをエンコード、またはデコードすることができます。

- UN/EDIFACT D.18B Directory へのサポート。EDI を参照してください。
- 新規の出エータベースバージョンがサポートされます: PostgreSQL 11、Db2 for i 7.2 および7.3
- Eclipse 4.7 – 4.10 へのサポート。Eclipse のためのMapForce プラグイン を参照してください。
- MapForce からJava 仮想マシンを明示的に設定するためのサポート。Java 設定 を参照してください。
- 内部のアップデートと最適化

MapForce リリース2019 の新規機能:

- MapForce (とMapForce Server) はデータをプロトコルバッファーでエンコードされたバイナリファイルから、またはバイナリインスタンスにデータをマップすることができるようになりました。プロトコルバッファーを参照してください。
- 新規のデータベースがサポートされるようになりました Firebird 3.0、Informix 12.10、MariaDB 10.2、MySQL 8.0、PostgreSQL 10、SQL Server 2017、Linux のためのSQL Server (前のバージョンにサポートが追加されました)。
- カスタム定義の条件を満たす固有のノードにノード関数またはデフォルトを選択して適用できます。例えば、正規表現に一致する名前を持つ特定の型のノードのみにノード関数を適用することができます。を参照してください。ノード関数とデフォルトを条件付きで適用する。
- ノード関数を作成する際、ノード名、またはノード注釈などをベースにしてデータを条件付きで処理することができます。ノードメタデータをノード関数に提供を参照してください。
- グループ化関数を含むマッピングは生成されたC#、C++、またはJava コード内で更に素早く実行することができます。
- 内部の更新と最適化。

MapForce 2018 リリース2 の新規機能:

- 関数をコピーし貼り付けることなく、デフォルトの値と関数を単一のアイテムだけではなく、複数の子孫アイテムに構造的に適用できるようになりました。この機能によりマッピングを簡素化し、マッピングの必要条件を解決する多種の方法が使用できるようになりました。次を参照してください: デフォルトとノード関数。
- JSON コンポーネントを含むマッピングは、BUILT-IN 変換エンジンだけではなく、生成されたコード (C# とJava)により実行することができるようになりました。
- マッピングの空のエリアをダブルクリックしてビルトイン関数、ユーザー定義関数、および、制約を簡単にマッピングに追加できるようになりました(次を参照してください: マッピングにビルトイン関数を追加する とマッピングに定数を追加する)。
- 次のデータベースがサポートされるようになりました MariaDB 10、Teradata 16
- ビルトイン関数 と定数 をマッピングの空のエリアをクリックして便利に追加できるようになりました。
- 内部のアップデートと最適化

MapForce リリース2018 の新規機能:

- FlowForce Server Advanced Edition がMapForce とMapForce Server と共に実行されている場合、AS2 プロセスを自動化するデータワークフローを作成することができます。AS2 統合を参照してください
- NCPDP (National Council for Prescription Drug Programs) SCRIPT 標準へのサポート、次を参照してください: NCPDP SCRIPT
- Visual Studio 2013、2015、と2017 のためのプログラムコード生成をサポートします。次を参照してください: コードジェネレーター
- 以下のデータベースバージョンをサポートします: Sybase ASE 16, PostgreSQL 9.6, MySQL 5.7
- Microsoft Excel にデータをマッピングすると、マッピングから直接、既存の列の間に新規のExcel 列を既存の接続を再度行うことなく、挿入することができます。次を参照してください: 既存の行の間に行を挿入する 。
- 内部のアップデートと最適化

MapForce 2017 リリース3 の新規機能:

(C) 2015-2021 Altova GmbH

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

MapForce リリース2017 の新規機能:

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

MapForce 2016 R2 の新規機能:

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

MapForce リリース2016 の新規機能:

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

MapForce バージョン 2015 R4 の新規機能:

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

MapForce バージョン 2015 R3 の新規機能:

- XML 出力内の`<?xml ... ?>`宣言を抑制するオプション
- W eb サービスを呼び出す際、プリエンプティブ HTTP 認証、HTTPS、と基本的 W S-Security を使用することができます (HTTP セキュリティの設定 とW S-Security の設定を参照してください)。
- Strict Open XML Excel 2013 ファイルの読み取りのサポート
- FlexText は正規表現を使用して分割および切り替えを行います。(FlexText と正規表現を参照してください)。
- (EDI CSV、固定値フィールド、JSON、XML を含む)テキストベースのコンポーネントは、プレーンファイルに加え、文字列を解析およびシリアル化することができます。
- SQLite データベース へのサポート(SQLite 接続のセットアップを参照してください)。
- 新規文字列パディング関数: pad-string-left とpad-string-right
- 新しいコンポーネント型: 単純型出力
- 内部的な更新と最適化

MapForce リリース2015 の新規機能:

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

MapForce バージョン 2014 R2 の新規機能:

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

MapForce リリース 2014 の新規機能:

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

MapForce バージョン 2013 R2 SP1 の新規機能:

- 高速変換エンジン RaptorXML Server

MapForce バージョン 2013 R2 の新規機能:

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

MapForce リリース 2013 の新規機能:

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

MapForce バージョン 2012 R2 の新規機能:

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

MapForce リリース2012 の新規機能:

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

MapForce バージョン 2011R3 の新規機能:

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

MapForce バージョン 2011R2 の新規機能:

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

MapForce リリース2011 の新規機能:

- マッピングチェーン にて、中間コンポーネントのプレビュー(パススループレビュー)

(C) 2015-2021 Altova GmbH

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

MapForce バージョン 2010 リリース 3 の新規機能:

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

MapForce バージョン 2010 リリース 2 の新規機能:

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

MapForce リリース 2010 の新規機能:

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

MapForce バージョン 2009 SP1 の新規機能:

- ユーザー定義関数内のパラメーターの順序はユーザーにより定義されることができます（ ユーザー定義関数内のパラメーターを参照してください）
- XML スキーマに対して妥当ではない XML ファイルの処理
- 全般的（標準的）なユーザー定義関数における複雑な階層構造の引数サポート（ インラインと正規ユーザー定義関数を参照してください）
- C++ コードの生成時における Apache Xerces 3.x のサポート

MapForce バージョン 2009 の新規機能:

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

MapForce バージョン2008 リリース2 の新規機能:

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

MapForce バージョン2008 の新規機能:

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

MapForce バージョン2007 リリース3 の新規機能:

- データベースフィールドへ/からXML データマッピング(XML データを データベースフィールドへ / からマッピング を参照してください)。
- データベースへの直接のクエリ
- SQL-WHERE フィルターならびにSQL ステートメントウィザード
- コード ジェネレータ の最適化ならびにドキュメンテーションの改善
- EDI X12 リリースの3040 から5030 を全てサポート
- 93A から06B のUN/EDIFACT メッセージを完全にサポート

(C) 2015-2021 Altova GmbH

1.2 サポートメモ

MapForce® は32/64 ビット W indows アプリケーションで次のオペレーティングシステムで作動します:

- プラットフォーム更新済みのW indows 7 SP1、W indows 8、W indows 10
- プラットフォーム更新済みのW indows Server 2008 R2 SP1 または以降

64 ビットのサポートはEnterprise とProfessional エディションでご利用いただくことが出来ます。

MapForce はオプションで次の統合された開発環境にプラグインとしてご利用いただくことが出来ます:

- Visual Studio 2010/2012/2013/2015/2017/2019、Visual Studio のためのMapForce プラグインを参照してください。
- Eclipse 2020-12, 2020-09, 2020-06, 2020-03、Eclipse のためのMapForce プラグインを参照してください。

MapForce はMicrosoft Office 製品を以下のように統合します:

- Access データベースへデータを、またはAccess データベースへデータからデータをマップすることができます。サポートされるバージョンに関しては、データベースとMapForce を参照してください。
- W ord 2000 または以降のフォーマットでマッピングドキュメントを生成することができます。マッピングドキュメントの生成とカスタマイズを参照してください。
- Excel 2007 または以降のスプレッドシートからのデータを、またはスプレッドシートへデータをマップすることができます。Microsoft OOXML Excel 2007+を参照してください

プログラムコード生成に適用することのできるサポート情報に関しては、コード生成についての紹介を参照してください。

サポートされるデータソースとターゲット

MapForce マッピングの変換言語を変更する場合、その特定の言語のために一部の機能がサポートされていない場合があります。次のテーブルは、サポートされるデータソースとターゲット書式を言語ごとに、**MapForce Enterprise Edition** 別にリストしています。

脚注:

- "Built-in" はマッピングを MapForce 内の「出力」タブをクリックすることによりマッピングを実行する、またはMapForce Server 内でマッピングを実行することを意味します。

マッピング形式		XSLT 1.0	XSLT 2.0	XSLT 3.0	XQuery	C++	C#	Java	
XML[1]		●	●	●	●	●	●	●	
CSV とテキスト						●	●	●	
MapForce FlexText を介したテキスト						●	●	●	
JSON							●	●	
XBRL[2]						●	●	●	
Excel 2007 お	データソースとして		●	●			●	●	●

マッピング形式		XSLT 1.0	XSLT 2.0	XSLT 3.0	XQuery	C++	C#	Java	
よび以降[3]	データターゲットとして（新規のファイルを生成）						●	●	●
	データターゲットとして（既存のファイルの更新）								●
EDI	ASC X12					●	●	●	●
	HIPAA X12					●	●	●	●
	HL7 version 2					●	●	●	●
	HL7 version 3	●	●	●	●	●	●	●	●
	IATA PADIS					●	●	●	●
	NCPDP SCRIPT EDIFACT format					●	●	●	●
	NCPDP SCRIPT XML format	●	●	●	●	●	●	●	●
	SAP IDoc					●	●	●	●
	SWIFT MT								●
	SWIFT MX / ISO 20022	●	●	●	●	●	●	●	●
	TRADACOMS					●	●	●	●
	UN/EDIFACT					●	●	●	●
データベース[4]	ADO					●	●		●
	ADO.NET						●		●
	JDBC							●	●

(C) 2015-2021 Altova GmbH

マッピング形式		XSLT 1.0	XSLT 2.0	XSLT 3.0	XQuery	C++	C#	Java	
	Native SQLite								●
	Native PostgreSQL								●
	ODBC					●	●	●	●
Web サービス[5]	REST-Style								●
	SOAP						●	●	●
プロトコルバッファー									●
生バイナリファイル									●

脚注:

1. 自動署名処理を使用するXMLはBUILT-IN内でのみサポートされています。
2. 自動署名処理を使用するXBRLはBUILT-IN内でのみサポートされています。
3. *Office Open XML* 書式（.xlsx）のExcel 2007または以降を指しています。*Strict Open XML Spreadsheet* 書式は.xlsx拡張子を使用しますが、（データターゲットとしてではなく）データソースとしてのみサポートされています。詳細に関しては、Microsoft OOXML Excel 2007+ を参照してください。
4. データベースの種類とターゲットの環境により制限は適用されます。詳細に関しては、異なる実行環境内のデータベースマッピングを参照してください。
5. （マッピングから生成されていない）Web サービスの呼び出しを指しています。SOAP Web サービスのために、クライアントの証明書、先行認証、およびWS-Security は生成されたC# とJava コード内でサポートされていません。詳細に関しては、Web サービスの呼び出しを参照してください。

生成されたコード内のサポートされる機能

MapForce Enterprise Edition 内の各言語内のコード生成とサポートに関連する機能を次のテーブルはリストしています。

機能	XSLT 1.0	XSLT 2.0	XSLT 3.0	XQuery	C++	C#	Java	BUILT-IN
マッピングにパラメータを提供する	●	●	●	●	●	●	●	●
マッピングから動的に入力ファイル名を提供する	●	●	●	●	●	●	●	●
マッピング入力としてワイルドカードファイル名を提供する[1]		●	●	●	●	●	●	●
マッピングから動的に出力ファイル名を生成する		●	●		●	●	●	●

機能	XSLT 1.0	XSLT 2.0	XSLT 3.0	XQuery	C++	C#	Java	BUILT-IN
マッピングから文字列の値を返す	●	●	●		●	●	●	●
変数		●	●	●	●	●	●	●
コンポーネントの並べ替え		●	●	●				●
関数のグループ別け		●	●		●	●	●	●
フィルター	●	●	●	●	●	●	●	●
コンポーネントのジョイン								●
Value-Map コンポーネント	●	●	●	●	●	●	●	●
デフォルトとノード関数								●
マッピングの例外		●	●	●	●	●	●	●
文字列の解析とシリアル化[2]								●
動的なノード名		●	●	●	●	●	●	●
XML とXBRL デジタル署名								●
データベースバルク入力								●
入力パラメーターを持たないデータベースSQL SELECT					●	●	●	●
入力パラメーターを持つデータベースSQL SELECT								●
データベースストアドプロシージャ								●
データベース例外の扱い[3]					●	●	●	●
データベーストレースとエラーのログ								●
EDI からXML へのクイック(非標準) 変換								●
SOAP W eb サービスの生成						●	●	

(C) 2015-2021 Altova GmbH

機能	XSLT 1.0	XSLT 2.0	XSLT 3.0	XQuery	C++	C#	Java	BUILT-IN
MapForce Server 実行ファイルの生成								●
FlowForce Server へのマッピングのデプロイ								●
バイナリファイルからデータを読み取る								●
バイナリファイルにデータを書き込む								●

脚注:

1. XSLT 2.0、XSLT 3.0 とXQuery は`fn:collection`関数を使用します。Altova XSLT 2.0、XSLT 3.0 とXQuery エンジン内の実装はワイルドカードを解決します。他のエンジンは異なる振る舞いをする可能性があります。
2. JSON に関しては、解析とシリアル化 Java とC# はこれをサポートされています。
3. 前のテーブルで示されているとおり、データベース例外の扱いはマッピング言語が現在接続されているデータベースドライバーによりサポートされている場合に可能になります。

1.3 MapForce とは ?

Altova W eb サイト: データマッピングルール

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

内蔵の変換言語のためにデザインされているデータマッピングはMapForce Server を使用した自動化に適しています。MapForce Server を使用すると、コマンドラインからのみではなく、API 呼び出し (.NET、Java、COM) から、または予定された、または再帰的なジョブとしてFlowForce Server の管理下でW indows、Linux、およびmacOS 上で実行することができます。詳細に関しては MapForce Server を使用した自動化を参照してください。

MapForce を使用してデザインされたマッピングによりデータを便利に、ファイルベース、またはは他のフォーマットに変換することができます。使用するテクノロジーに関わらず MapForce は通常自動的にデータの構造を決定、またはデータのために スキーマを提供するオプションを与えます。MapForce はサンプルインスタンスファイルから自動的に生成することもできます。例えば、XML インスタンスファイルが存在し、スキーマ定義が存在しない場合、MapForce はスキーマ定義を生成し、他のファイル、またはフォーマットのためにXML ファイル内のデータを使用できるようにします。

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

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

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

(C) 2015-2021 Altova GmbH

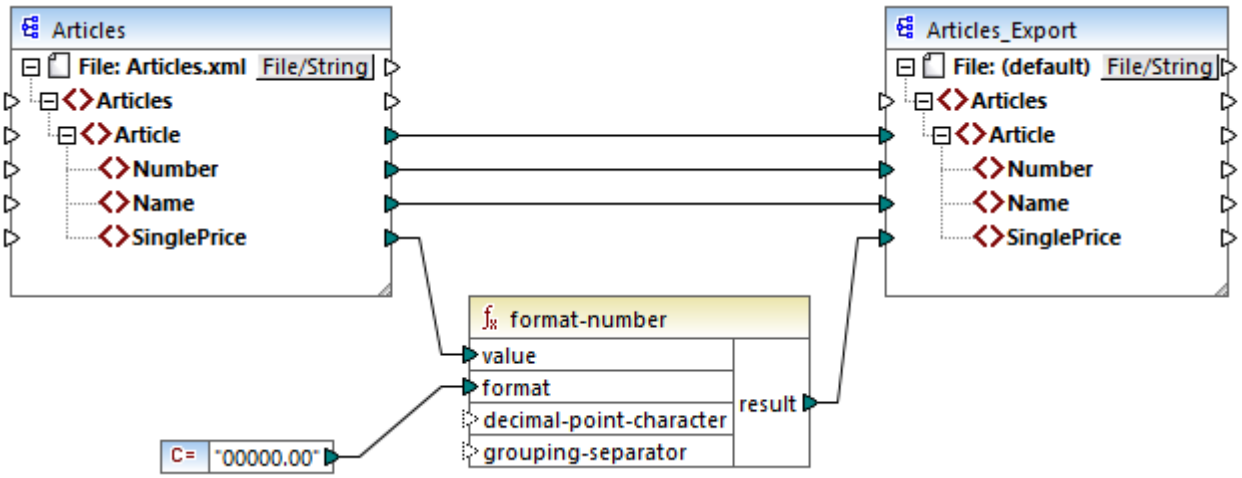
MapForce Basic Edition	MapForce Professional Edition	MapForce Enterprise Edition
• XSLT 1.0 • XSLT 2.0 • XSLT 3.0	• MapForce 内蔵の変換言語 • XSLT 1.0 • XSLT 2.0 • XSLT 3.0 • XQuery • Java • C# • C++	• MapForce 内蔵の変換言語 • XSLT 1.0 • XSLT 2.0 • XSLT 3.0 • XQuery • Java • C# • C++

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

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

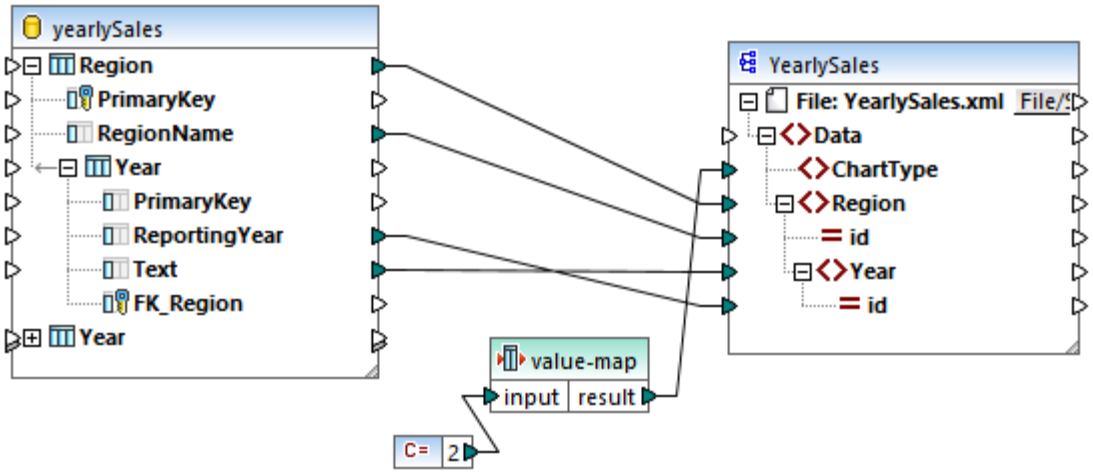
MapForce マッピングの変換言語を変更すると、その特定の言語のために特定の機能がサポートされていない場合があります。サポートメモを参照してください。

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



2 つの XML ファイル間のサンプルデータ変換

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



XML ファイルとデータベース間のサンプルデータ変換

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

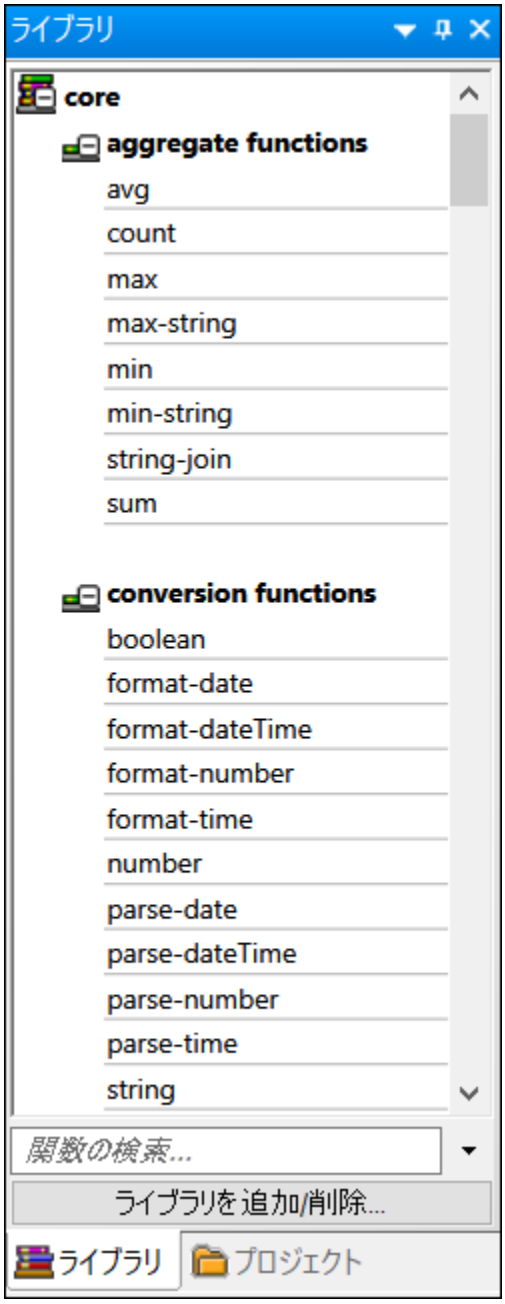
MapForce 内で作成することのできるデータマッピングは、単一かつ定義済みファイルだけではありません。同じ変換内で、ディレクトリから複数の入力ファイルを動的に処理することができ、複数の出力ファイルを生成することができます。ですから、[複数の X ファイルからデータを読み込み、単一の Y ファイルに書き込む] または [ファイル X を読み込み複数のファイル Y を生成する] というシナリオが可能になります。

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

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

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

(C) 2015-2021 Altova GmbH



ライブラリウィンドウ

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

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

MapForce Enterprise Edition では、W eb サービス言語定義 (W SDL) ファイルをベースにしたSOAP 1.0 とSOAP 2.0 ウェイブサービスを視覚的にデザインすることができます。詳細はSOAP W eb サービスの実装 を参照してください。W SDL-スタイル、またはREST-スタイルW eb サービスをマッピング内から呼び出すこともできます。詳細に関しては、W eb サービスの呼び出しを参照してください。

MapForce Professional とEnterprise Editions では、マッピングデザイン ファイルの詳細なドキュメントをHTML、およびRTF フォーマット で生成することができます。ドキュメンテーションデザインをカスタム化することができます(例: ドキュメントから特定のコンポーネントを包含、および、削除することが選択可能です)。

Altova MissionKit 製品と共にMapForce を使用している場合、MapForce はこれらの製品とAltova サーバーベース製品を以下のテーブルで示されている通り統合します。

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

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

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

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

MapForce の全てのエディションは32-ビット アプリケーションで動作します。更に、MapForce Professional とEnterprise エディションは64-ビット アプリケーションで使用することができます。

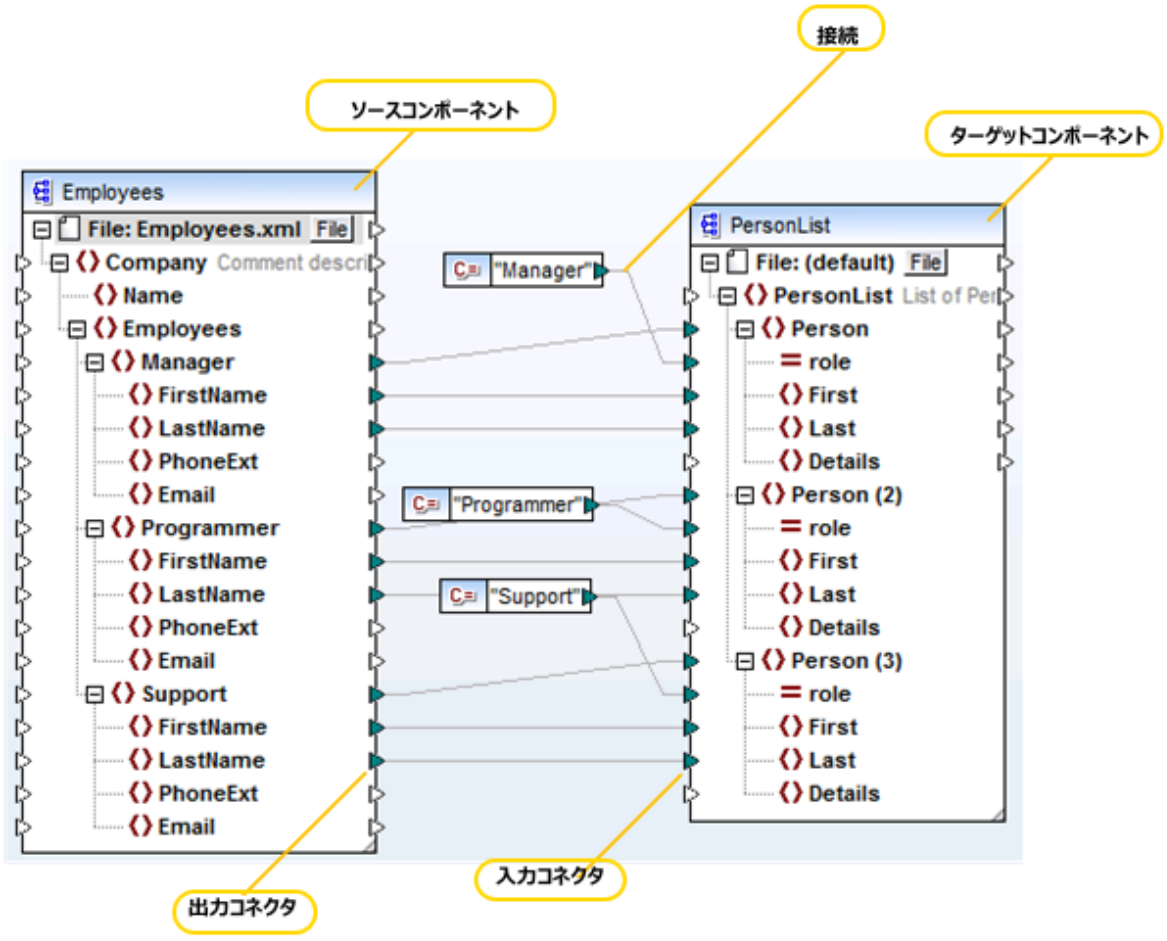
(C) 2015-2021 Altova GmbH

1.4 基本概念

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

マッピング

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



MapForce マッピングの基本的構造

コンポーネント

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

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

コネクタ

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

接続

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

ソースコンポーネント

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

ターゲットコンポーネント

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

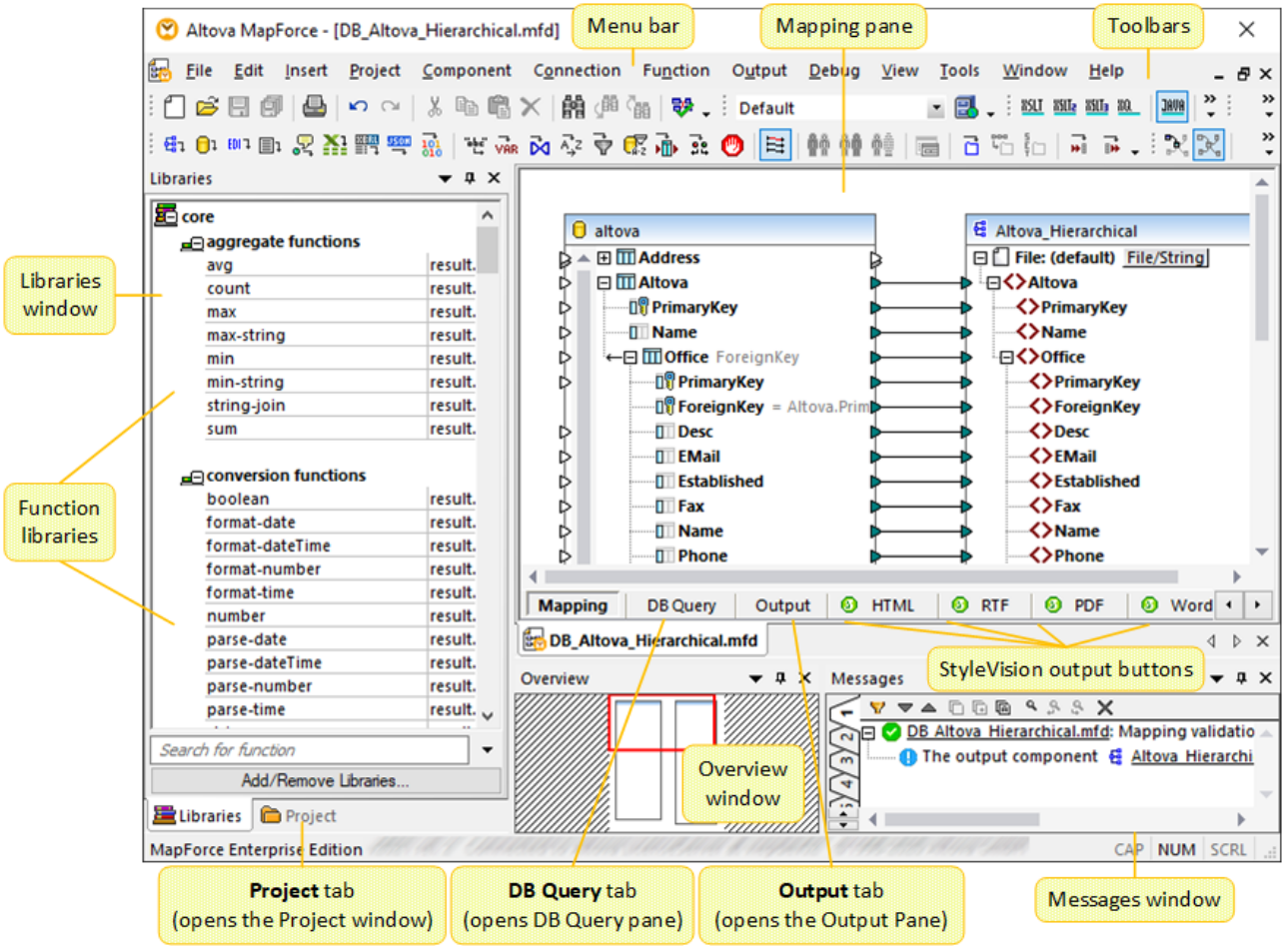
(C) 2015-2021 Altova GmbH

1.5 ユーザーインターフェイスの概要

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

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

下のイメージにはMapForce グラフィカルなユーザーインターフェイス(MapForce Enterprise Edition) の主要なパーツが表示されています。

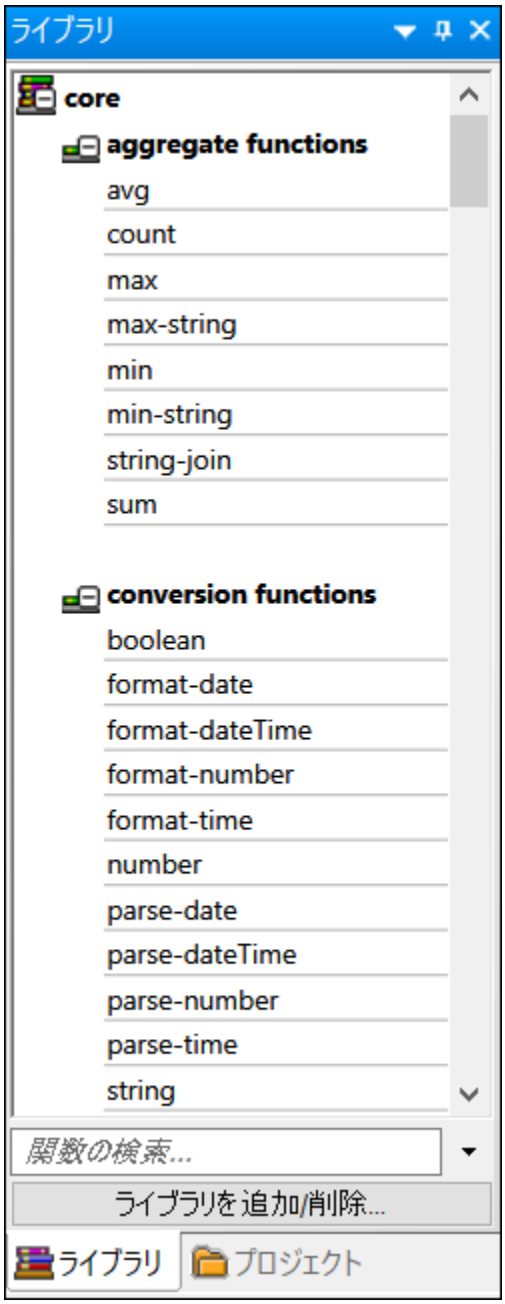


メニューバーとツールバー

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

ライブラリウィンドウ

ライブラリウィンドウは、ライブラリごとに整理されたMapForce 内蔵の関数をリストします。使用することのできる関数は「出力」メニューまたは言語選択ツールバーから選択される変換言語により異なります。変換言語の選択も参照してください。ユーザー定義関数を作成した場合、またはインポートした場合、これらの関数もライブラリウィンドウに表示されます。



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

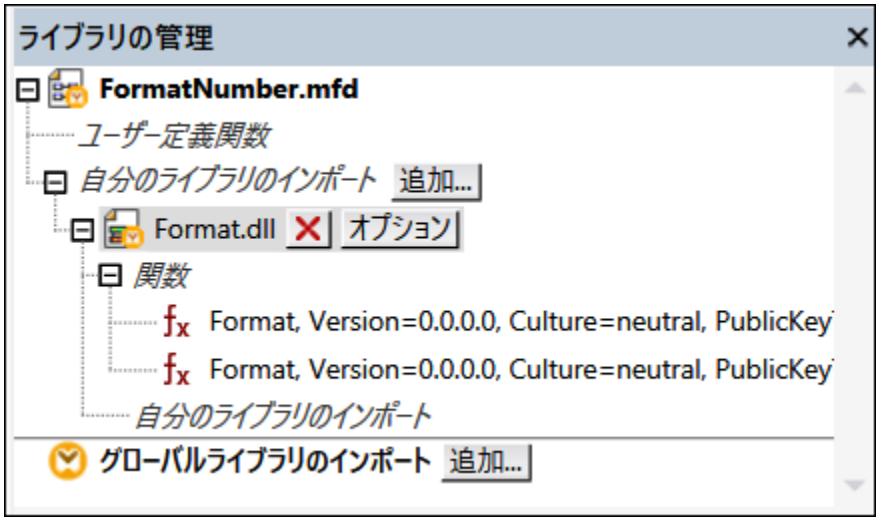
(C) 2015-2021 Altova GmbH

ライブラリウィンドウの管理

このウィンドウから現在開かれているマッピングにより(コンパイル済みの Java クラスファイル、.NET DLL アセンブリファイルを含む) 使用されているすべてのユーザー定義関数 (UDF) とインポートされたカスタムライブラリを表示し管理することができます。

デフォルトではライブラリの管理ウィンドウは表示されていません。表示するには以下の一つを行うことができます:

- 「**表示**」メニューから「**ライブラリの管理**」をクリックします。
- ライブラリウィンドウのベースの **ライブラリの追加/削除** をクリックします。



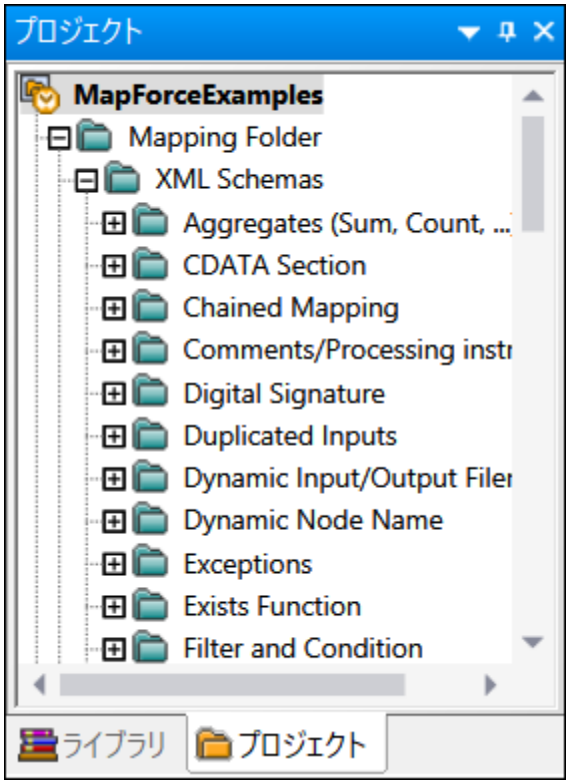
UDF とライブラリを現在アクティブなマッピングのためにのみ、またはすべての開かれているマッピングのために表示することを選択できます。現在開かれているすべてのマッピングのためにインポートされた関数とライブラリを表示するには、ウィンドウ内を右クリックし、コンテキストメニューから「**開かれているドキュメントの表示**」選択します。

名前の代わりに開かれているマッピングのパスを表示するには、コンテキストメニューから「**ファイルパスの表示**」を選択します。

詳細に関しては関数ライブラリの管理を参照してください。

プロジェクトウィンドウ

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



マッピング ペイン

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

XSLT ペイン

XSLT ペインは、「XSLT」ボタンをクリックすると、マッピングから生成されたXSLT 変換コードを表示します。このペインを切り替えるにはXSLT、XSLT 2 またはXSLT3 を変換言語として選択し、同じ名前のタブをクリックします。

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

(C) 2015-2021 Altova GmbH

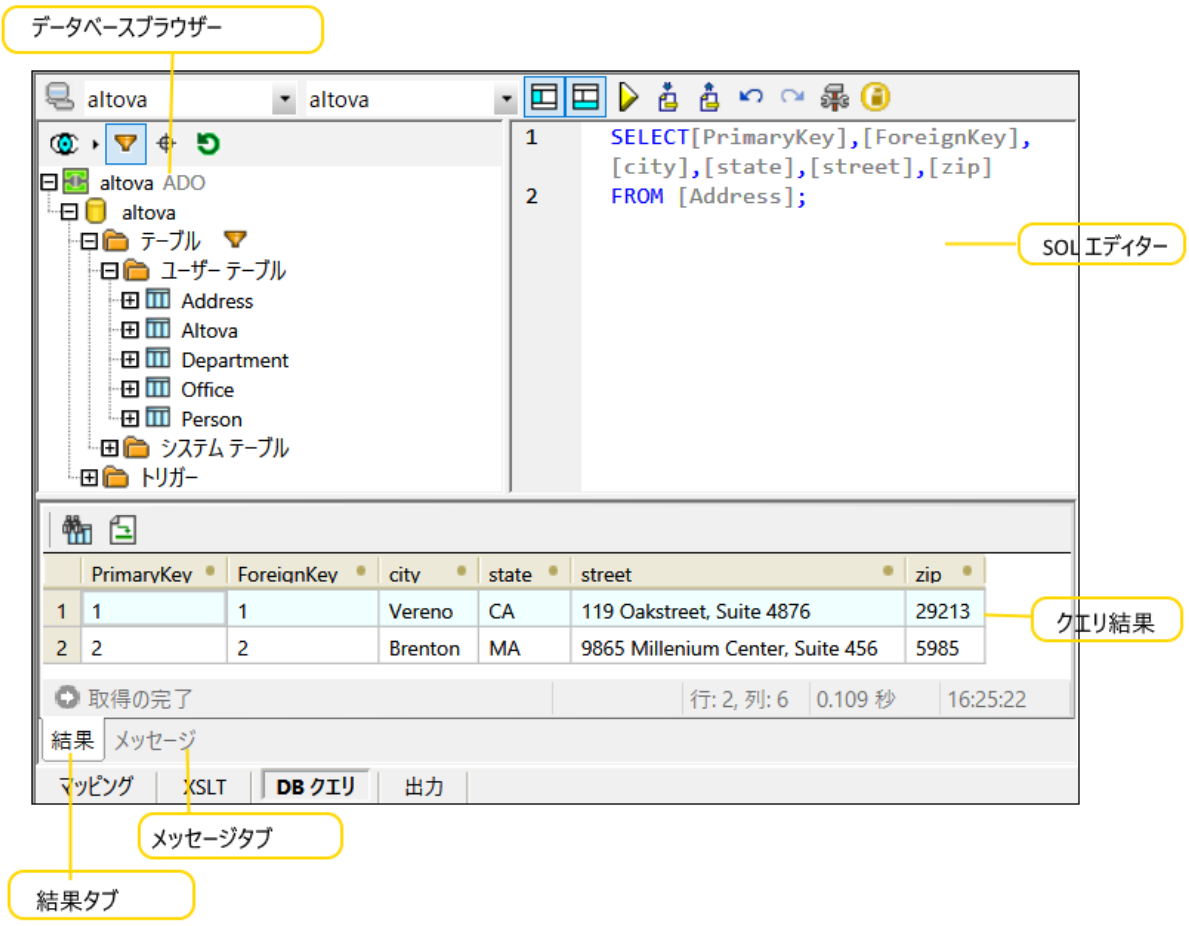
（インデント、行末マーカーを含む）表示設定を構成するには、ペインを右クリックして、コンテキストメニューから「**テキストビュー設定**」を選択します。または、「**テキストビュー設定**」（ ）ツールバーボタンをクリックします。

XQuery ペイン

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

DB クエリペイン

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



更に詳しい情報に関しては、データベースの参照とクエリを参照してください。

出力ペイン

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

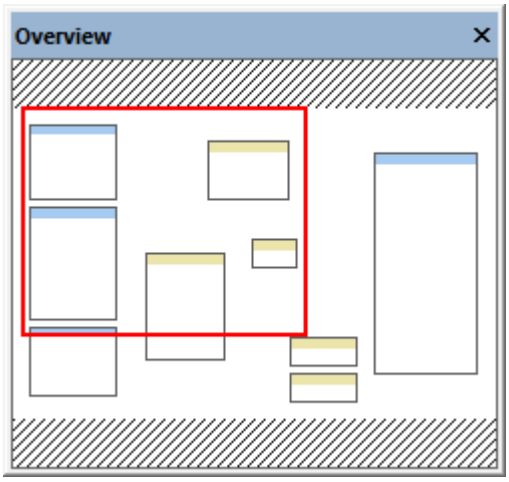
XSLT ペインと同様の仕組みを持つラインの付番とコードの折りたたみ機能を提供します（上を参照）。

StyleVision 出力ボタン

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

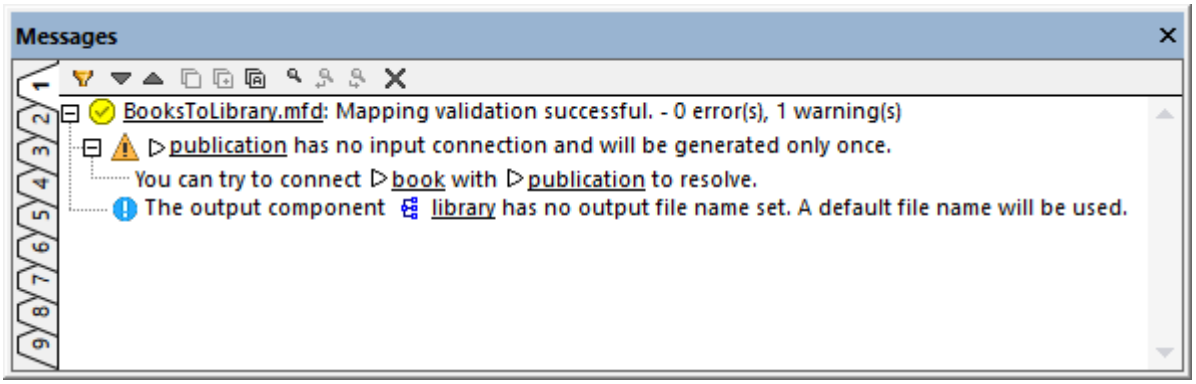
概要ウィンドウ

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



メッセージウィンドウ

メッセージウィンドウにはマッピングのプレビュー または 検証中 の警告やエラーメッセージが表示されます。



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

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

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

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

(C) 2015-2021 Altova GmbH

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

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

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

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

アプリケーションステータスバー

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

1.6 規約

サンプルファイル

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

- C:\Users\<username>\Documents\Altova\MapForce2021\MapForceExamples
- C:\Users\<username>\Documents\Altova\MapForce2021\MapForceExamples\Tutorials

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

グラフィカルユーザーインターフェイス

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

2 チュートリアル

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

XML を新しいスキーマに変換

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

複数のソースから1 つのターゲットにマップ

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

複数のターゲットスキーマとの作業

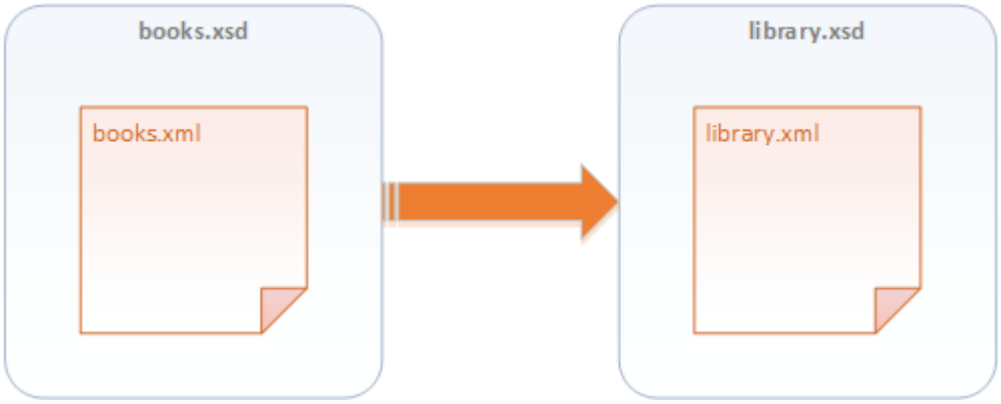
このチュートリアルは、2 つ以上のターゲット出力を作成する更に複雑なマッピングとの作業方法について説明します。具体的な方法は、同じマッピング内でブックレコードのリストを保管するファイル、最初のXML ファイル内のブックのサブセットのみを含む他のXML ファイルを生成し、発行年によりフィルターします。データのフィルタリングをサポートするには、「フィルター」コンポーネント、関数、数値定数を使用します。

ファイルを動的に処理と生成する

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

2.1 XML を新しいスキーマに変換

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



データ変換の抽象的なモデル

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

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

books.xml

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

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


(C) 2015-2021 Altova GmbH

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

library.xml

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

必要とされるデータ変換を行うには、以下のステップを行います。

ステップ 1: XSLT2 を変換言語として選択する

これを以下の方法で行うことができます:

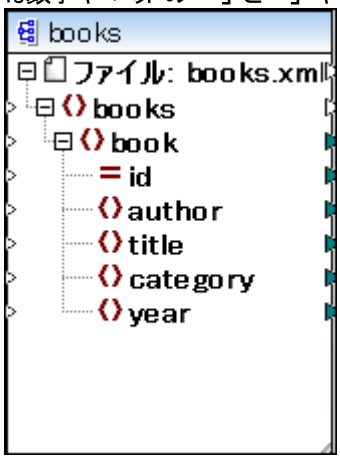
- 「XSLT2」 ツールバーボタンをクリックします。
- 「出力」 メニューから「XSLT 2.0」をクリックします。

ステップ 2: ソース XML ファイルをマッピングに追加する

このマッピングのためのソース XML ファイルは、は、以下のパスで見つけることができます: **<マイドキュメント>\Altova\MapForce2021\MapForceExamples\Tutorial**books.xml。ファイルを以下の方法でマッピングに追加することができます:

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

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



マッピングコンポーネント

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

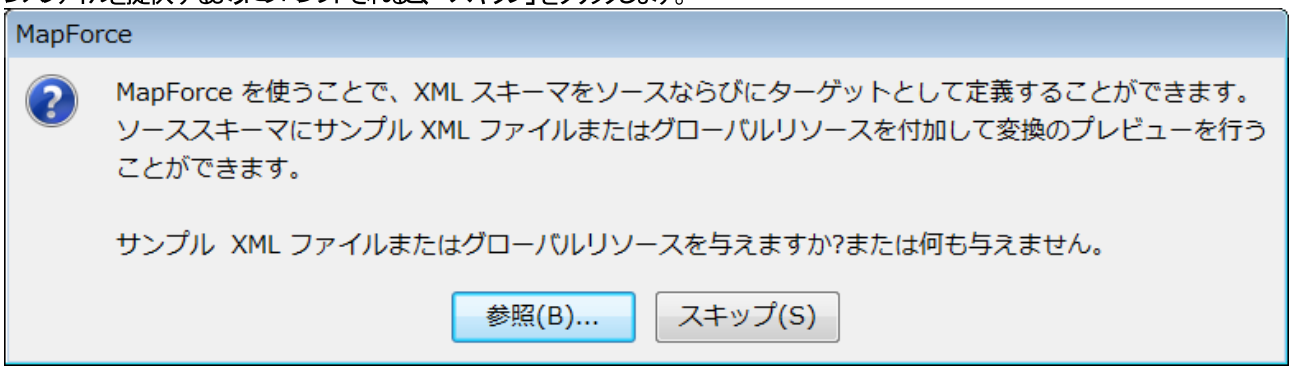
トップレベルノード は、ファイル名を表します。このサンプルの場合、は、XML インスタンスファイルの名前を表示します。構成内の、XML 要素は アイコンにより表示されており、XML 属性は アイコンにより表示されています。

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

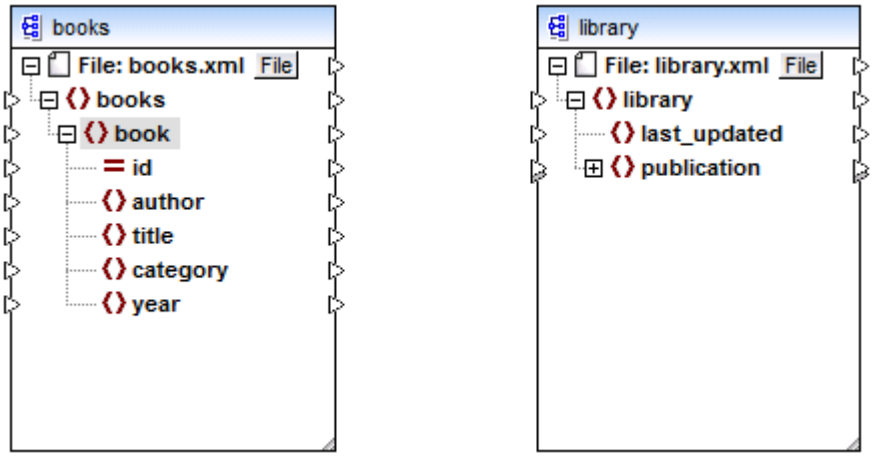
ステップ 3: ターゲット XML スキーマをマッピングに追加する

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

このサンプルでは、以下で見つけることのできる既存のスキーマファイルを使用します: **<マイドキュメント >\Altova\MapForce2021\MapForceExamples\Tutorial\library.xsd**。マッピングに追加するには、ソースXML ファイルと同様のステップを踏んでください。(「XML スキーマ / ファイルの挿入」( ) ツールバーボタンをクリックします)。MapForce にインスタンスファイルを提供するようにプロンプトされると、「スキップ」をクリックします。



この時点では、マッピングデザインは、以下のようになります:

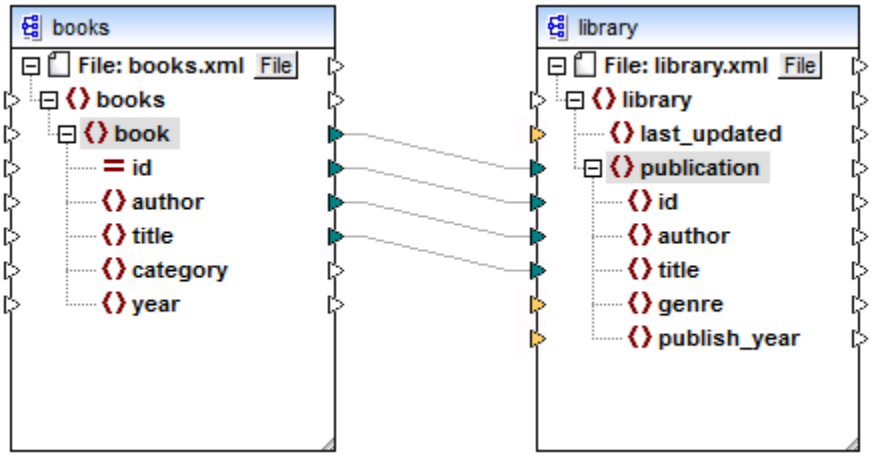


(C) 2015-2021 Altova GmbH

ステップ 4: 接続の作成

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

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



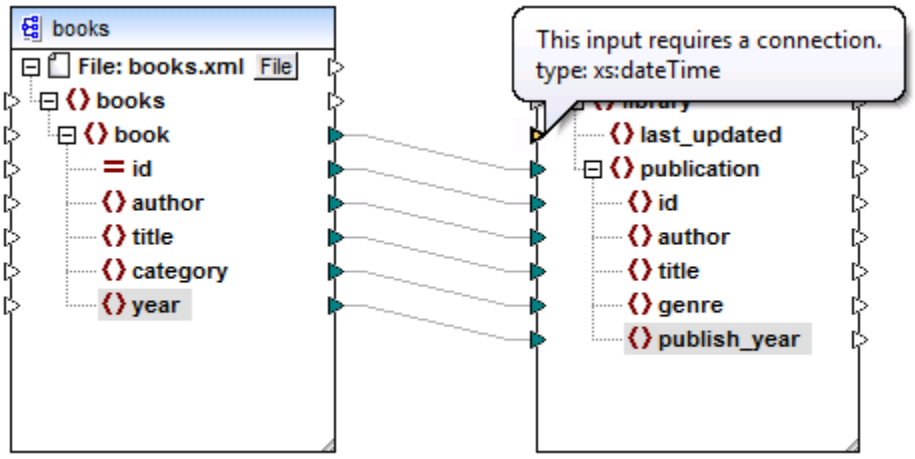
「マッチする子を自動的に接続する」の振る舞いを以下の方法で有効化または無効化することができます:

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

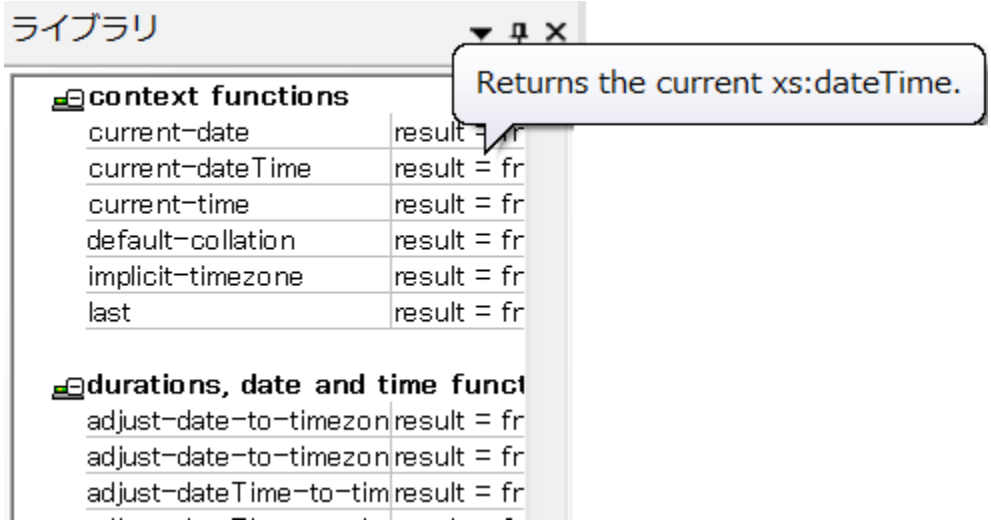
ターゲットコンポーネントの入力コネクタの一部が、MapForce により、必須アイテムを表示する、オレンジ色にハイライトされていることに注意してください。XML ファイルの妥当性を確認するために、必須アイテムの値を以下のように提供してください:

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

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

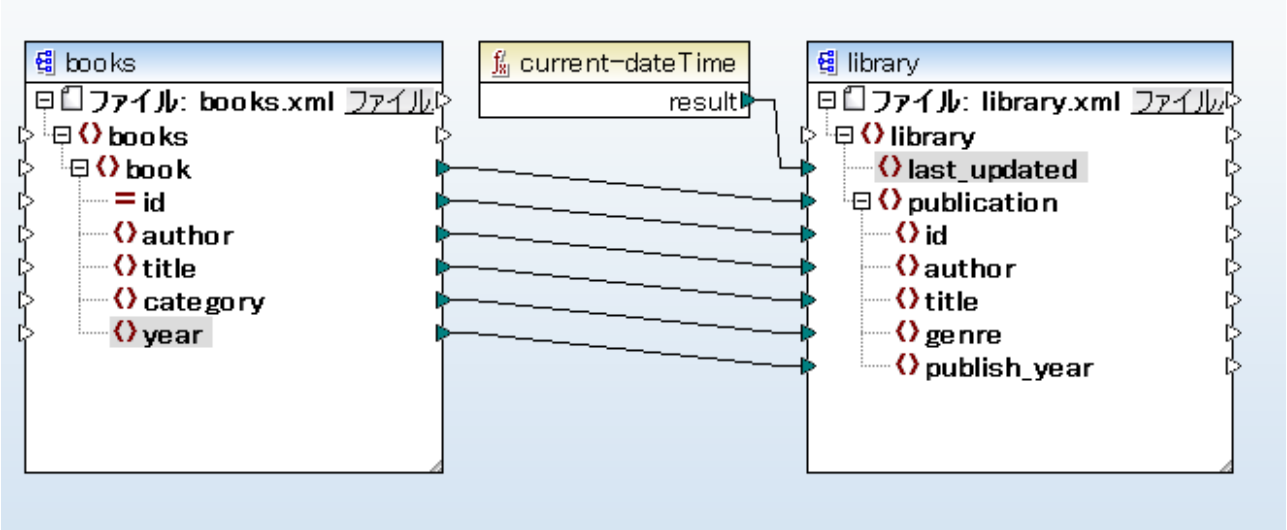


「**データ型の表示**」 ツールバーボタンをクリックすることにより、常に各アイテムのデータ型を表示することができます。
現在の日時（すなわち、`xs:dateTime` の値）を関 XSLT 数の日付と時刻で取得することができます。マッピングに対する XSLT 関数を検索するには、ライブラリウィンドウの下の部分にあるテキストボックスに「date」を入力します。または、マッピングの空のエリアをダブルクリックし、「current-date」を入力します。



上に表示されるように、マウスを関数の結果の部分にかざすと、詳細が表示されます。ヒントを表示するには、「**ヒントの表示**」 ツールバーボタンが有効化されていることを確認してください。
マッピングに関数を追加するには、関数をマッピングペインにドラッグし、`<last_updated>` 要素の出力を入力に接続します。

(C) 2015-2021 Altova GmbH



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

ソースのためのコンポーネント設定

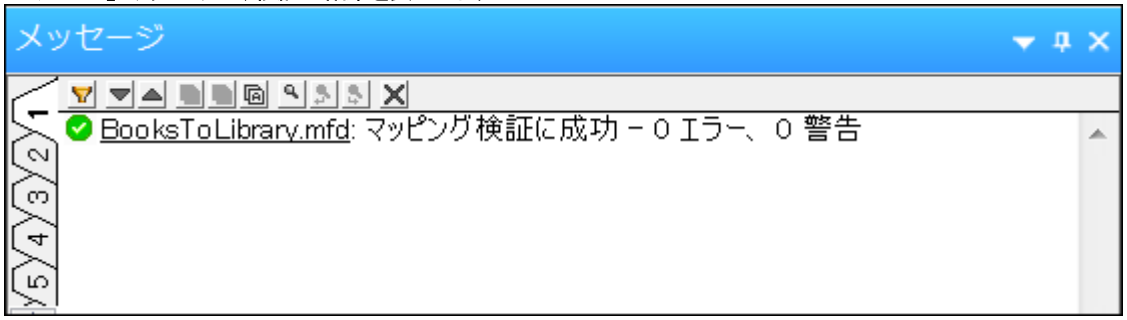
ターゲットのコンポーネント設定

ステップ 5: マッピングを検証し保存する

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

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

「メッセージ」ウィンドウは、検証の結果を表示します:



メッセージウィンドウ

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

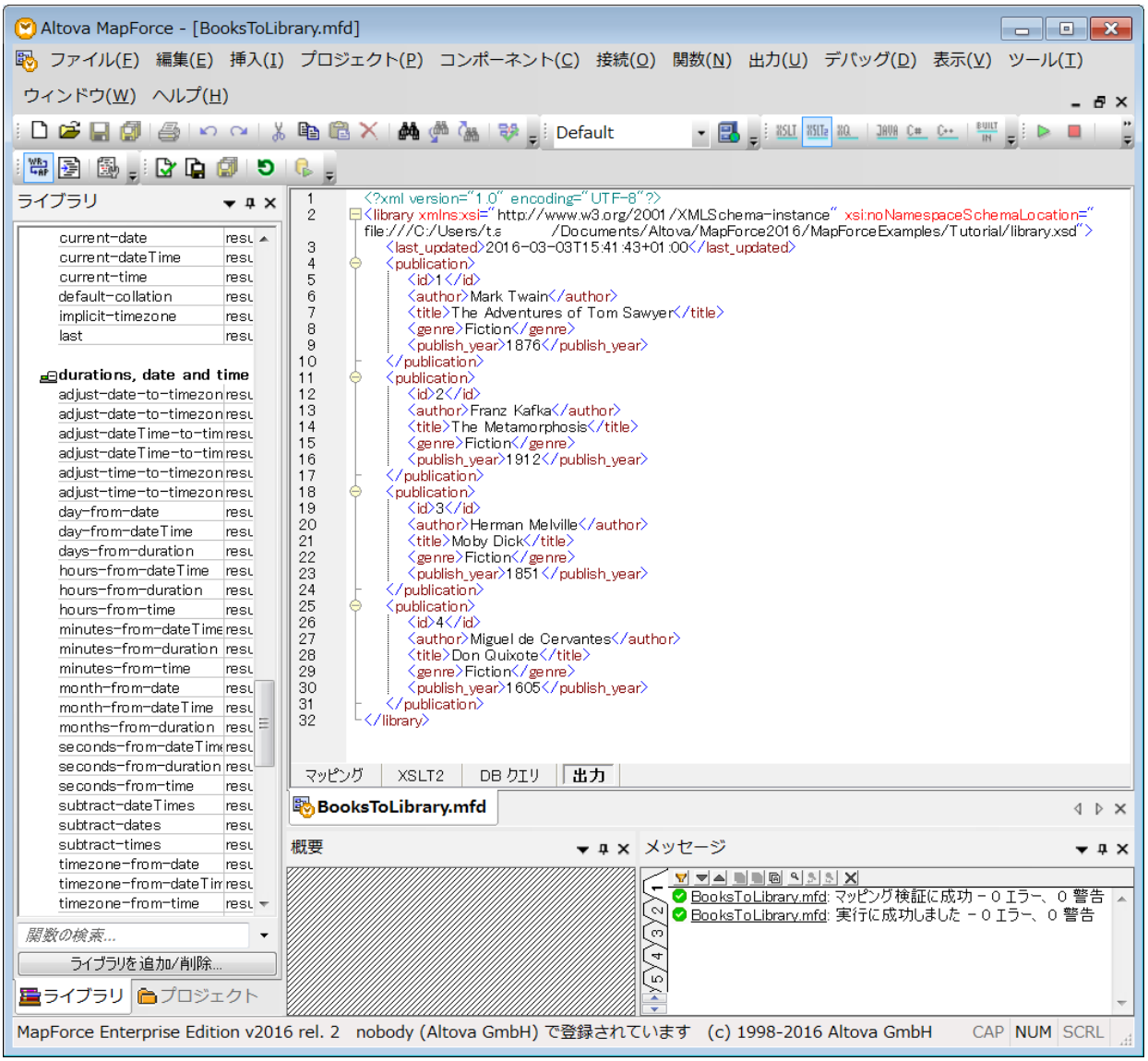
- 「ファイル」メニューから「保存」をクリックします。
- 「保存」 ツールバーボタンをクリックします。

このチュートリアルで作成されたマッピングは以下で検索することができます: **<マイドキュメント>\Altova\MapForce2021\MapForceExamples\Tutorial\BooksToLibrary.mfd**。ですから、これから先、自分で作成したマッピングを継続して使用するか、または **BooksToLibrary.mfd** ファイルを使用することができます。

(C) 2015-2021 Altova GmbH

ステップ 6: マッピングの結果のプレビュー

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



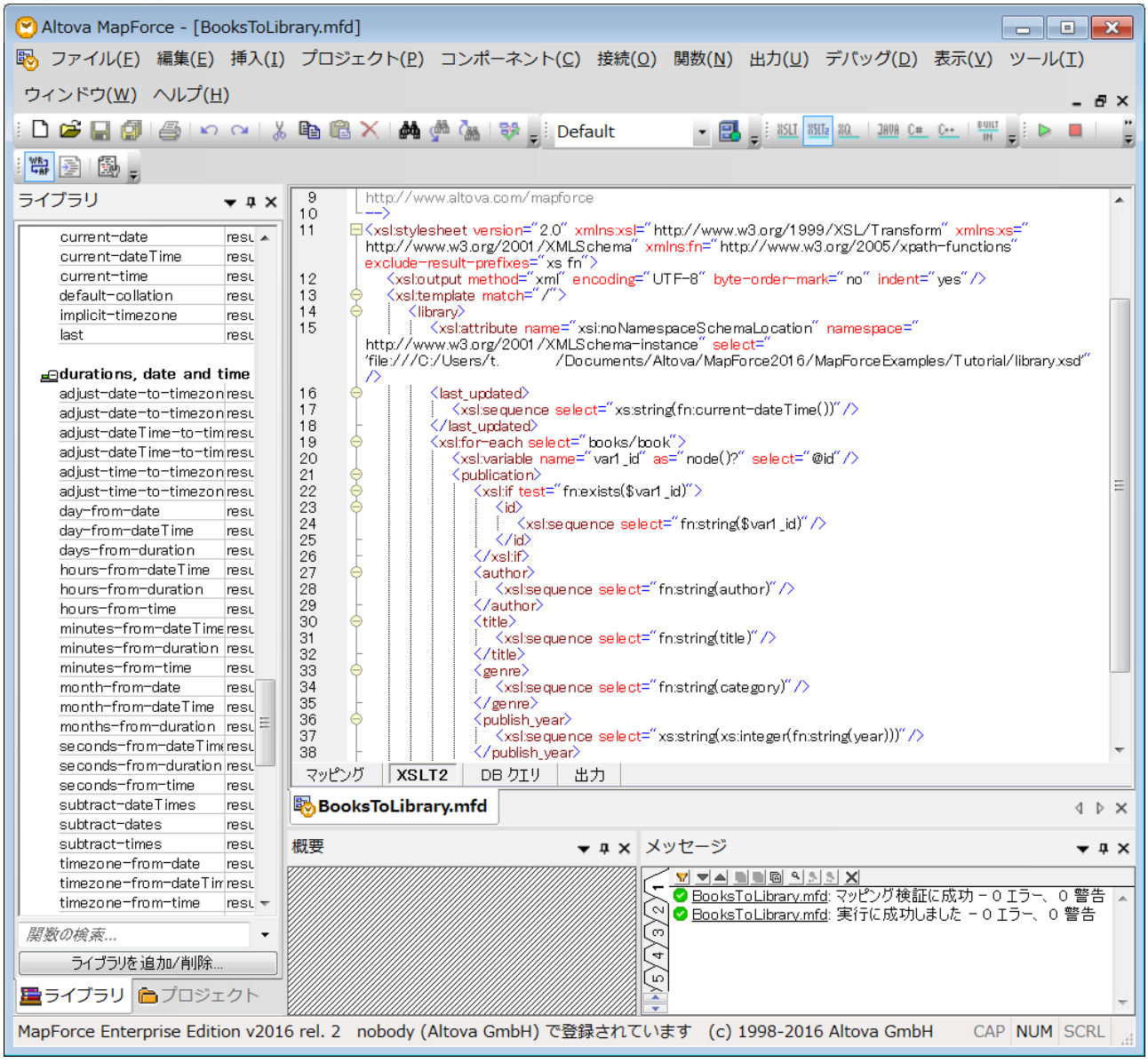
出力ペイン

MapForce 内で変換の結果を確認することができます。

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

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

変換を行う 生成された XSLT コードもプレビューすることができます。コードをプレビューするには、マッピングペインの下にある、「XSLT2」 ボタンをクリックします。



XSLT2 ペイン

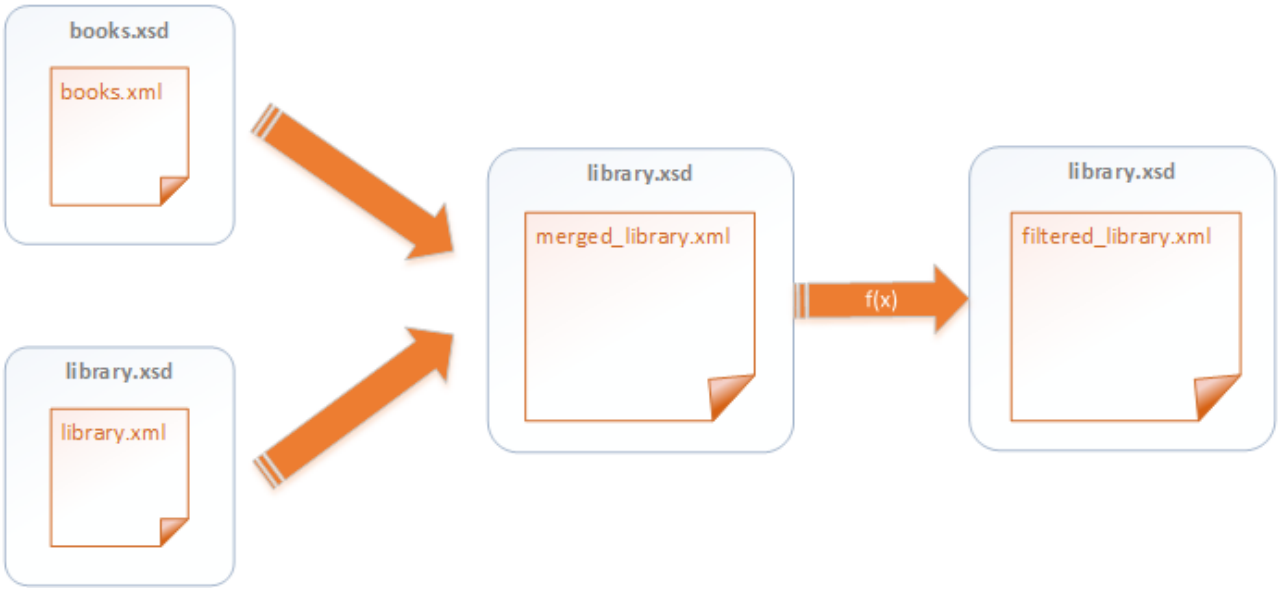
XSLT2 コードを生成しファイルに保存するには、メニューアイテム「**ファイル | コード生成 | XSLT 2.0**」を選択します。プロンプトされると、生成されたコードが保存されるフォルダーを選択してください。コード生成が完了すると、デスティネーションファイルは、以下の2つのファイルを含みます:

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

2.2 複数のソースから1つのターゲットにマップ

前のチュートリアルでは、ソースファイル（**books.xml**）からデータをターゲットファイル（**library.xml**）に変換しました。ターゲットファイル（**library.xml**）はマッピングの実行前は存在しておらず、マッピングの変換により生成されました。 **library.xml** ファイル内にすでにデータが存在すると仮定し、このデータを **books.xml** から変換されたデータとマージするとします。**library.xml** 空のデータを **books.xml** からのデータとマージすると仮定します。

このチュートリアルの目的は、**merged_library.xml** という名前のファイルを生成するマッピングをデザインすることです。生成されたファイルは、以下の2つのソースからのデータを含みます: **books.xml** と **library.xml** 。ソースとして使用される両方のファイルは異なるスキーマを持つことに注意してください。ファイルを動的に処理と生成するで説明されているとおり、ソースファイルと同じスキーマを持つ場合、異なるアプローチを使用してデータをマージすることができます



データ変換の抽象的なモデル

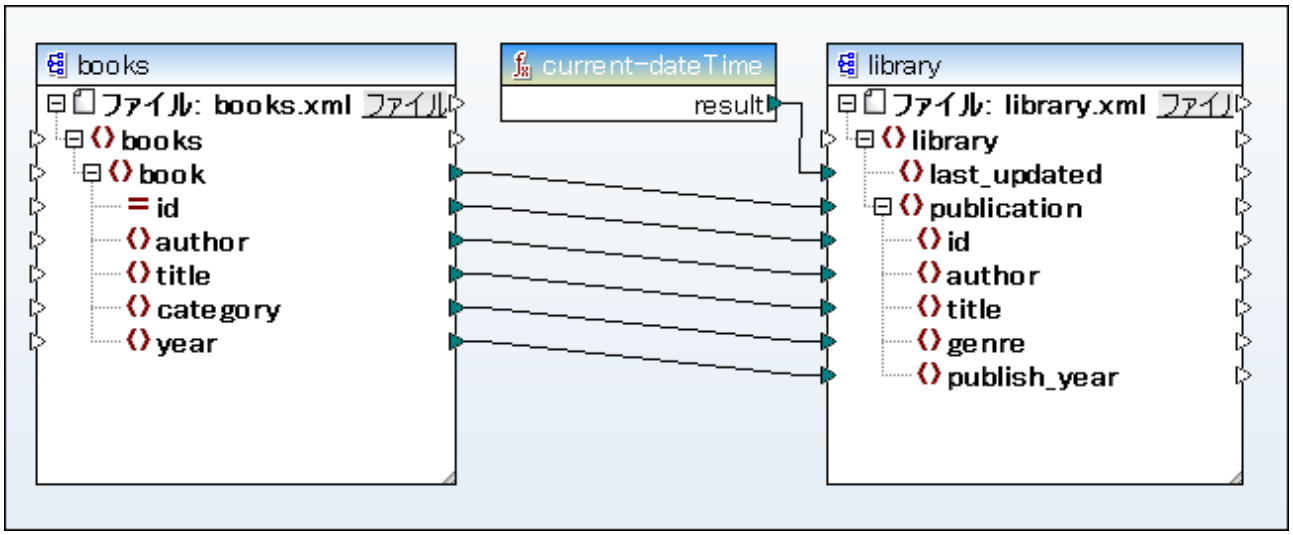
チュートリアルの目的を達成するためには、次のステップを踏みます。

ステップ 1: マッピングデザインファイルの準備

このチュートリアルは、＜**マイドキュメント**＞¥Altova¥MapForce2021¥MapForceExamples¥Tutorial¥フォルダーからの **BooksToLibrary.mfd** マッピングを開始点として使用します。このマッピングはXML を新しいスキーマに変換するチュートリアル内で作成済みです。チュートリアルを開始するには、MapForce で **BooksToLibrary.mfd** ファイルを開き、新しい名前で保存します。

> 複数のファイルを参照するため、＜**マイドキュメント**＞¥Altova¥MapForce2021¥MapForceExamples¥Tutorial¥フォルダー内に新しいマッピングが保存されることを確認してください。

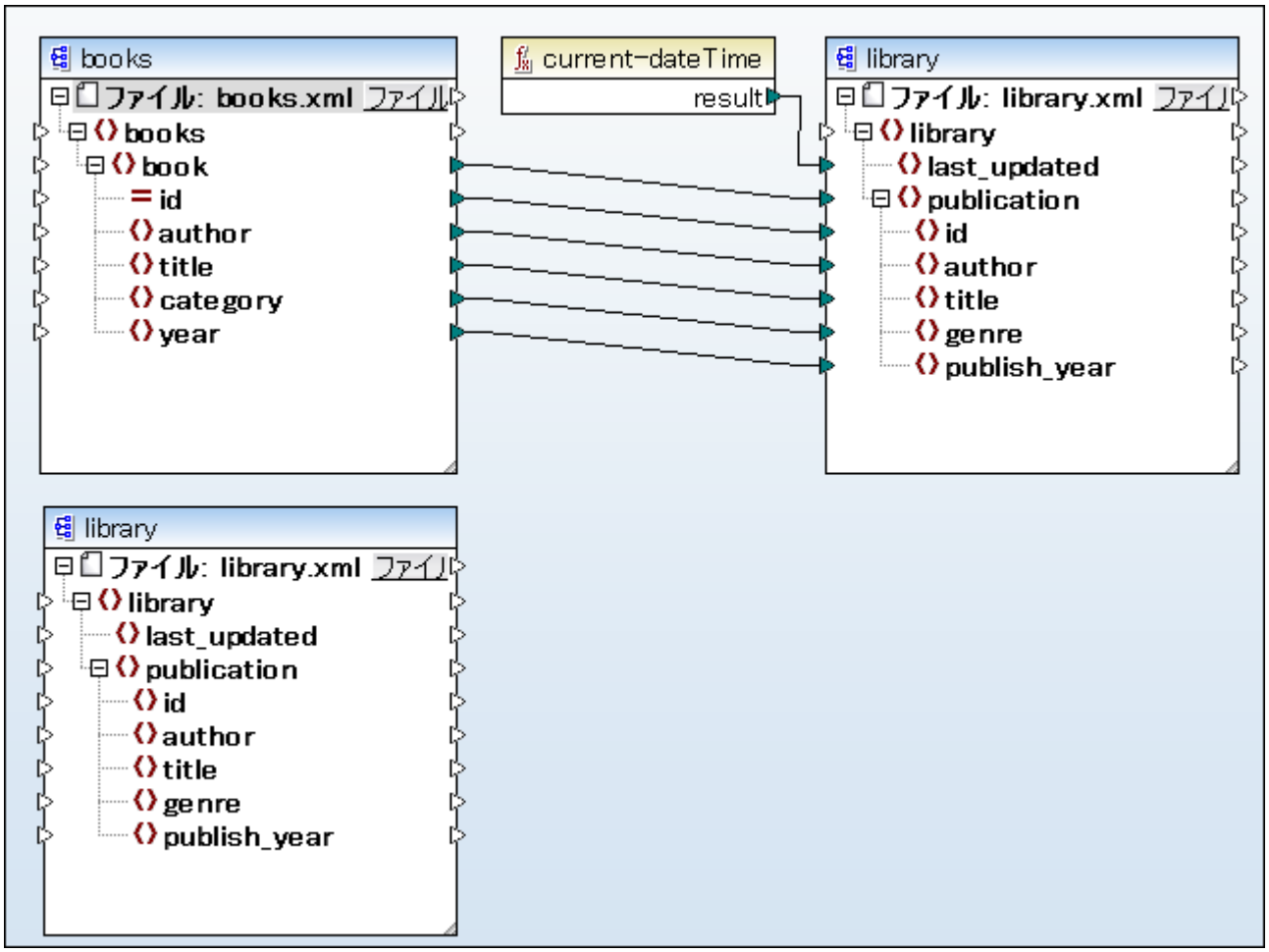
(C) 2015-2021 Altova GmbH



BooksToLibrary.mfd (MapForce Basic Edition)

ステップ 2: 2番目のソースコンポーネントの作成

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



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

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

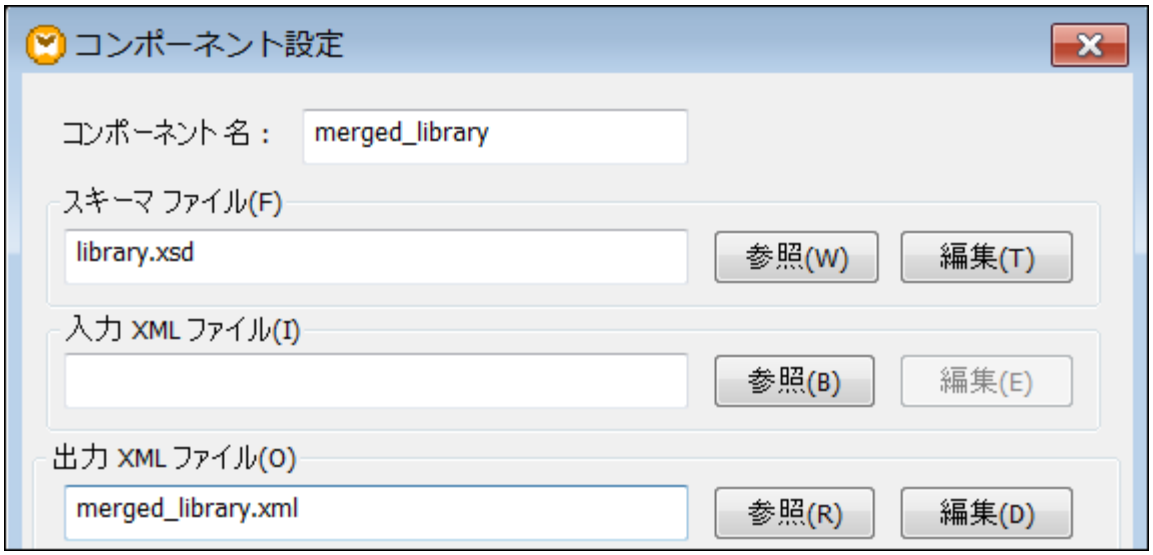
ステップ 3: 入力/出力ファイルの検証とセット

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

(C) 2015-2021 Altova GmbH

*最初のソースのコンポーネント設定 (**books**)*

*2番目のソースのコンポーネント設定 (**library**)*

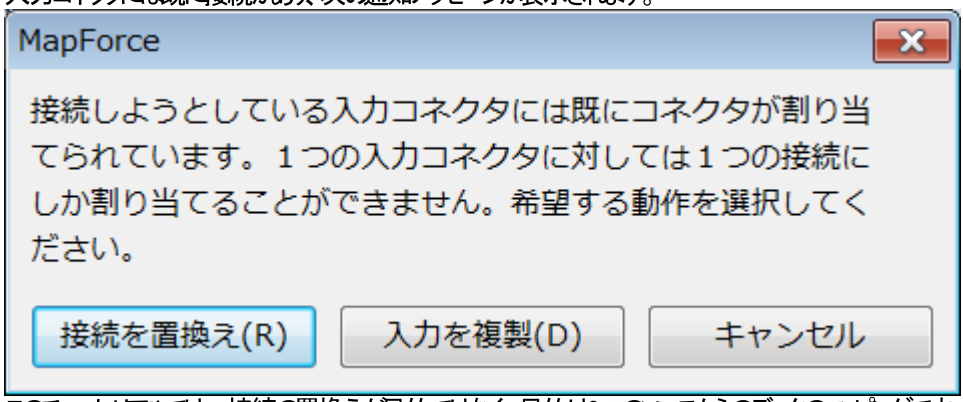


*ターゲットのコンポーネント設定 (**merged_library**)*

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

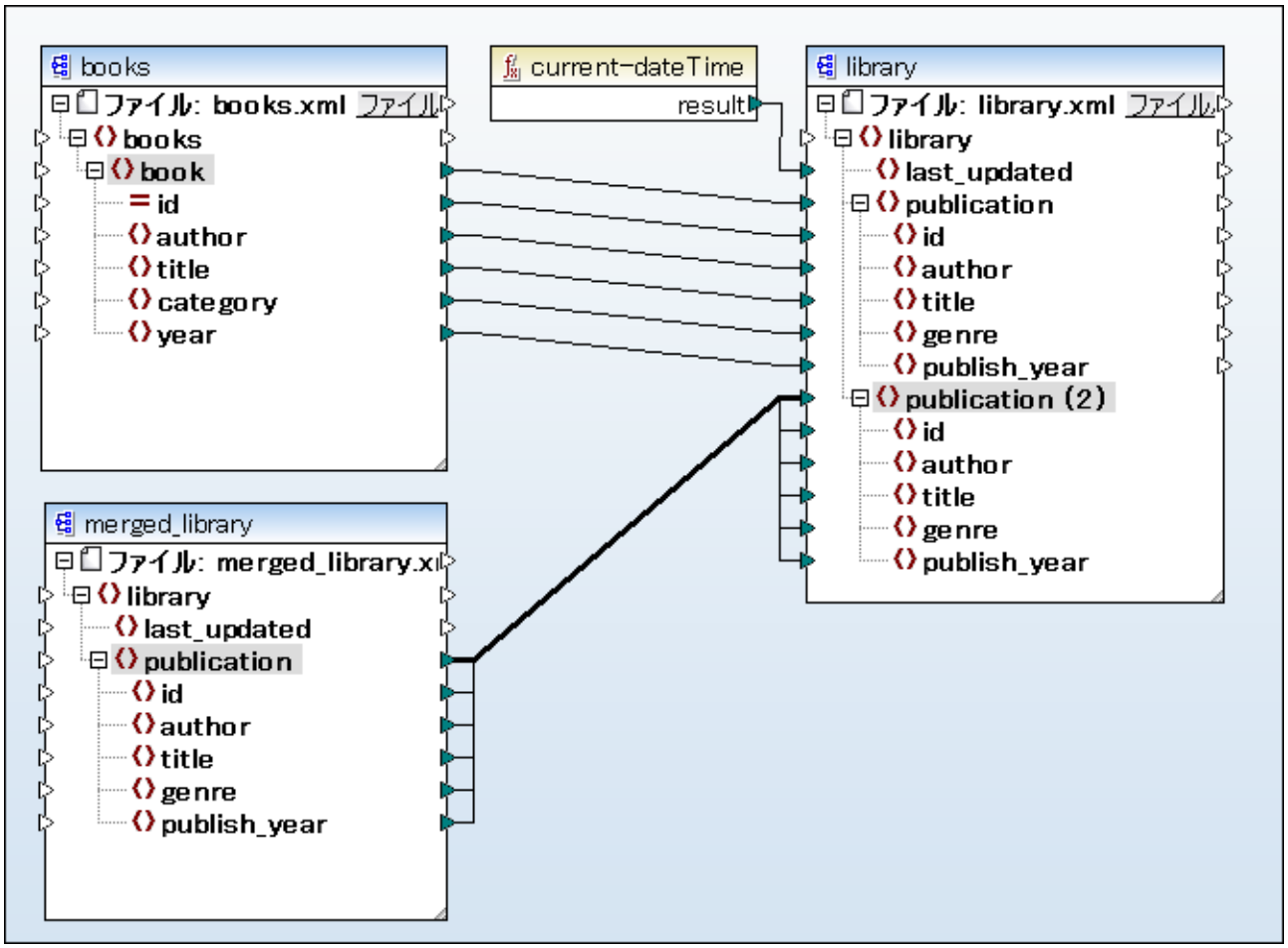
ステップ 4: 接続の作成

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



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

(C) 2015-2021 Altova GmbH



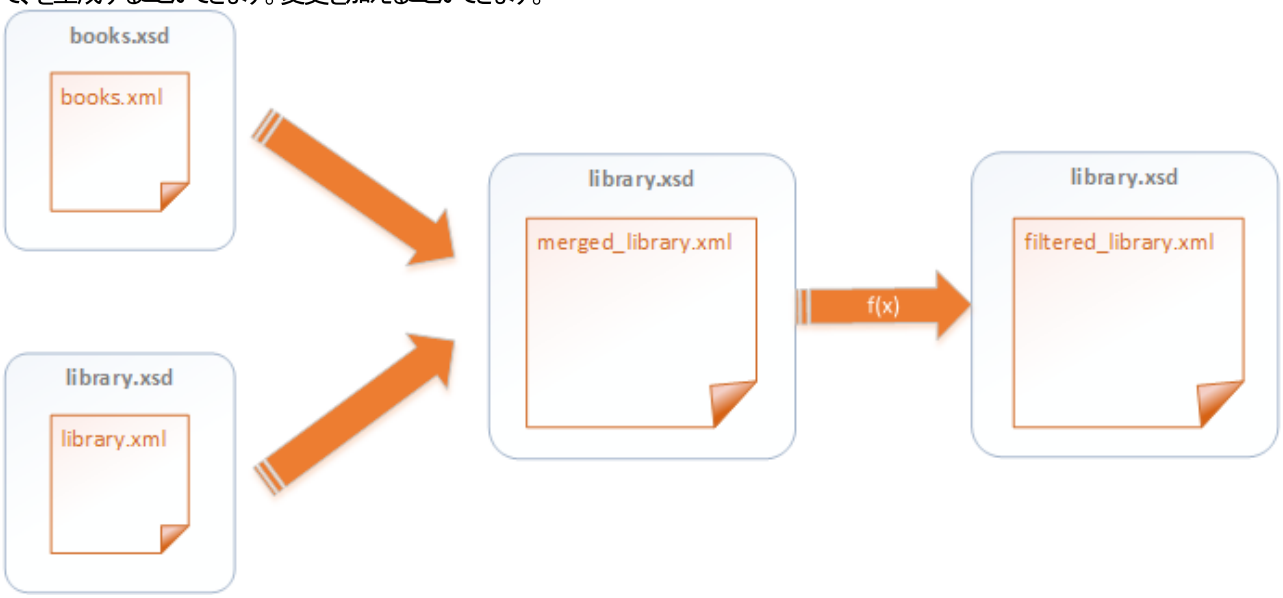
ターゲットコンポーネント内の`publication`アイテムが複製されました。新規`publication(2)`ノードはソース**library**コンポーネントからのデータを受け入れます。更に重要な点は、マッピング内でこのノードは`publication(2)`と表示されますが、結果XMLファイル内での名前は、想定された目的である`publication`となります。

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

2.3 複数のターゲットスキーマとの作業

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

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



データ変換の抽象的なモデル

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

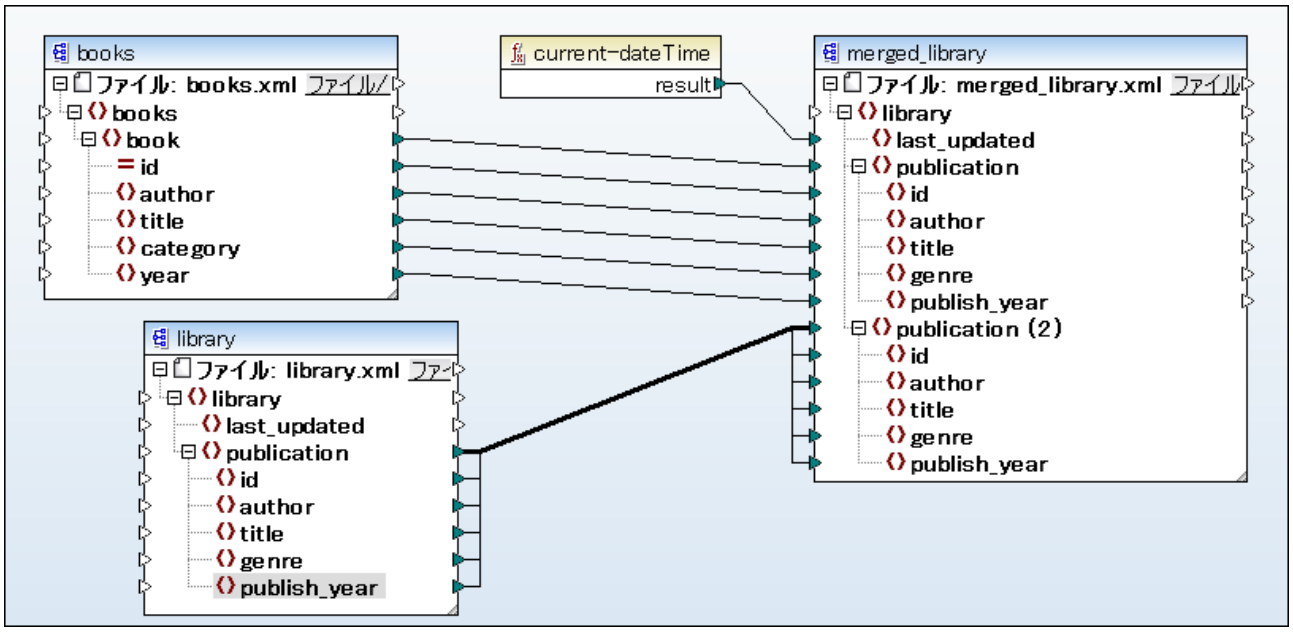
このチュートリアルのゴールは、**merged_library.xml** と**filtered_library.xml** を必要に応じて生成することです。チュートリアルの目的を達成するためには、次のステップを踏みます。

ステップ 1: マッピングデザインファイルの準備

このチュートリアルは、＜マイドキュメント＞¥Altova¥MapForce2021¥MapForceExamples¥Tutorial¥フォルダーからの **MultipleSourcesToOneTarget.mfd** マッピングを開始点として使用します。このマッピングは複数のソースから1つのターゲットにマップ チュートリアル内で作成済みです。チュートリアルを開始するには、MapForce で**MultipleSourcesToOneTarget.mfd** ファイルを開き、新しい名前で保存します。

> 複数のファイルを参照するため、＜マイドキュメント＞¥Altova¥MapForce2021¥MapForceExamples¥Tutorial¥フォルダー内に新しいマッピングが保存されることを確認してください。

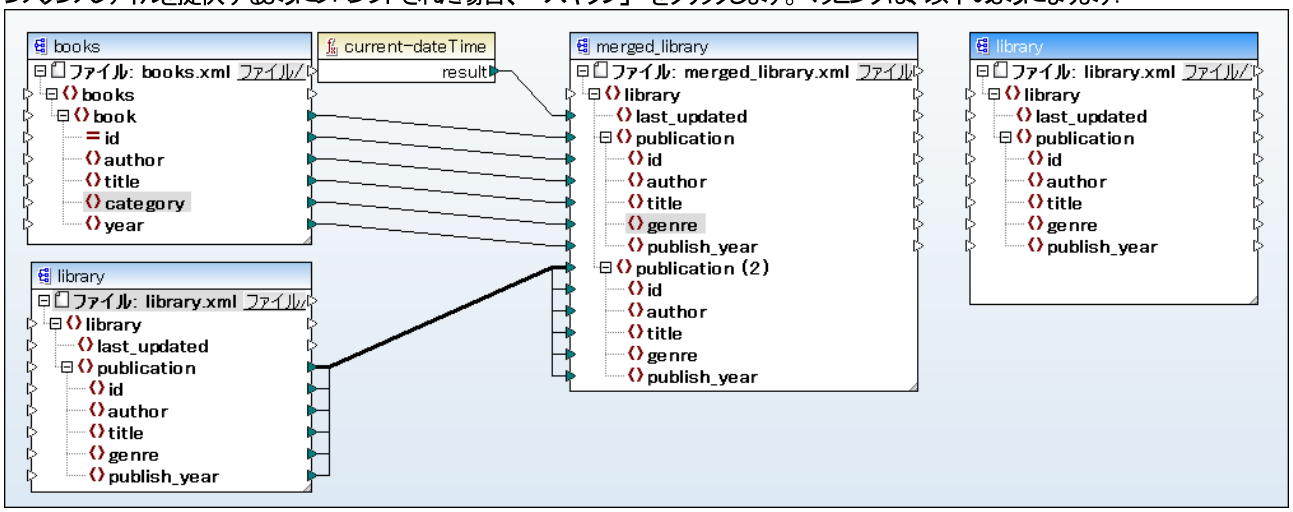
(C) 2015-2021 Altova GmbH



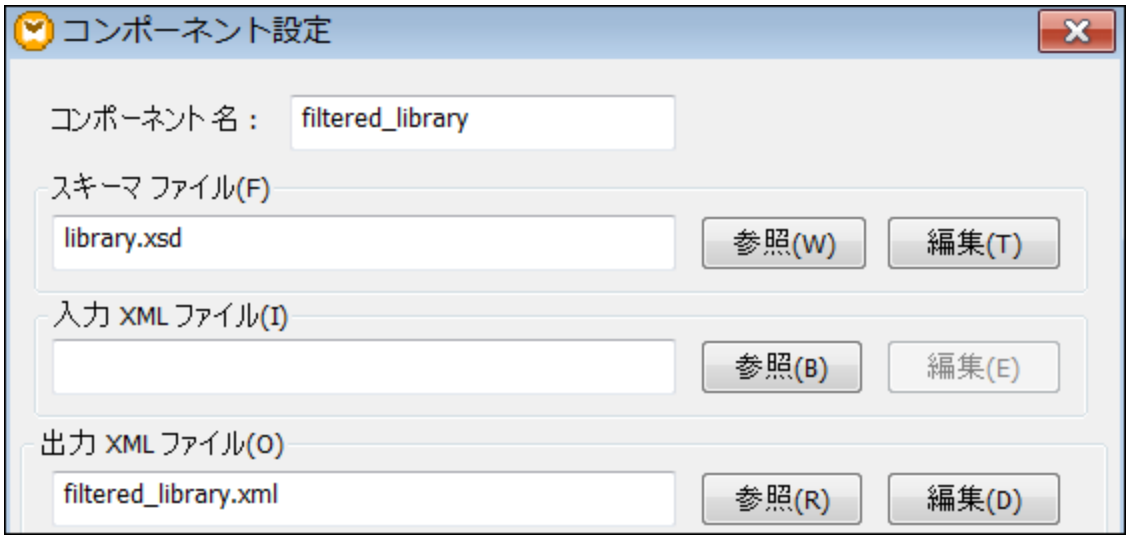
MultipleSourcesToOneTarget.mfd (MapForce Basic Edition)

ステップ 2: 2番目のターゲットコンポーネントの追加と構成

2番目のターゲットコンポーネントを追加するには、「XML スキーマ / ファイルの挿入」( ) ツールバーボタンをクリックし、<マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥Tutorial¥フォルダーにある **library.xsd** ファイルを開きます。サンプルインスタンスファイルを提供するようにプロンプトされた場合、「スキップ」 をクリックします。マッピングは、以下のようになります:



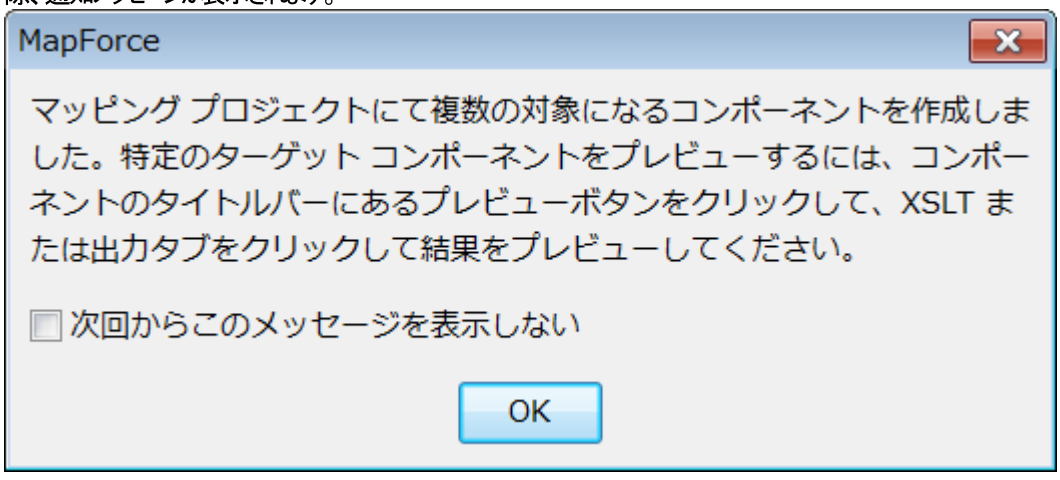
上に表示されるように、マッピングには2つのソースコンポーネントと **books** と**library**、と2つのターゲットコンポーネントがあります。ターゲットコンポーネントを区別するために、2番目のターゲットコンポーネントを**filtered_library** に名前を変更し、生成される XML ファイルの名前も変更します。これを行うには、右端にあるコンポーネントをダブルクリックして、コンポーネントの設定を次のように編集します:



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

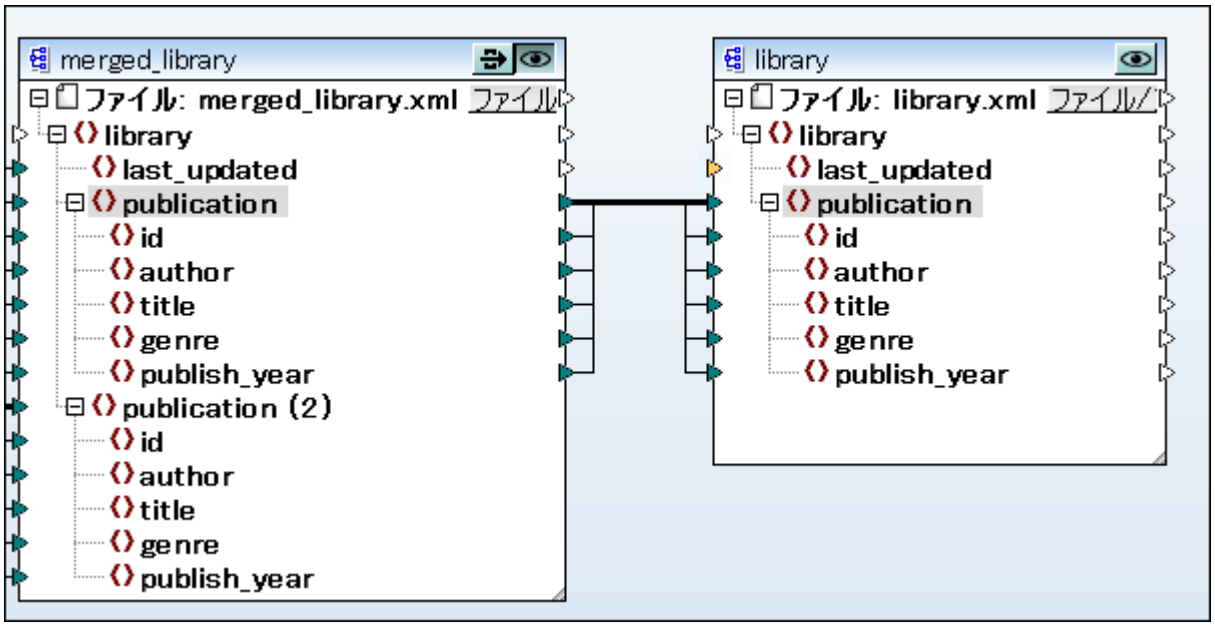
ステップ 3: 接続の作成

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



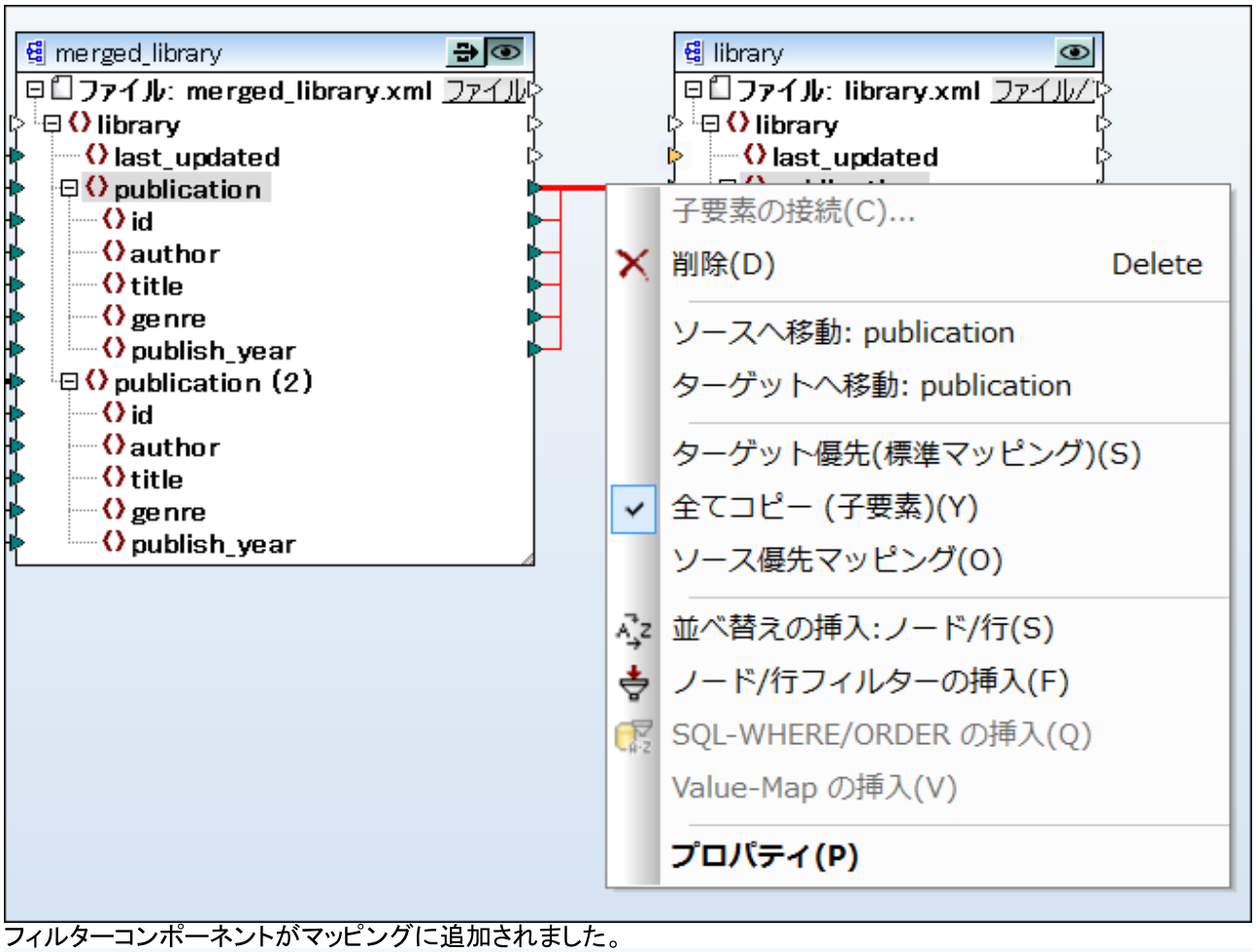
「OK」をクリックします。次の新規のボタンがターゲットコンポーネントの右上に表示されます:「**プレビュー**」( ) と「**パススルー**」( )。次のステップでこれらのボタンは使用され、説明されます。

(C) 2015-2021 Altova GmbH

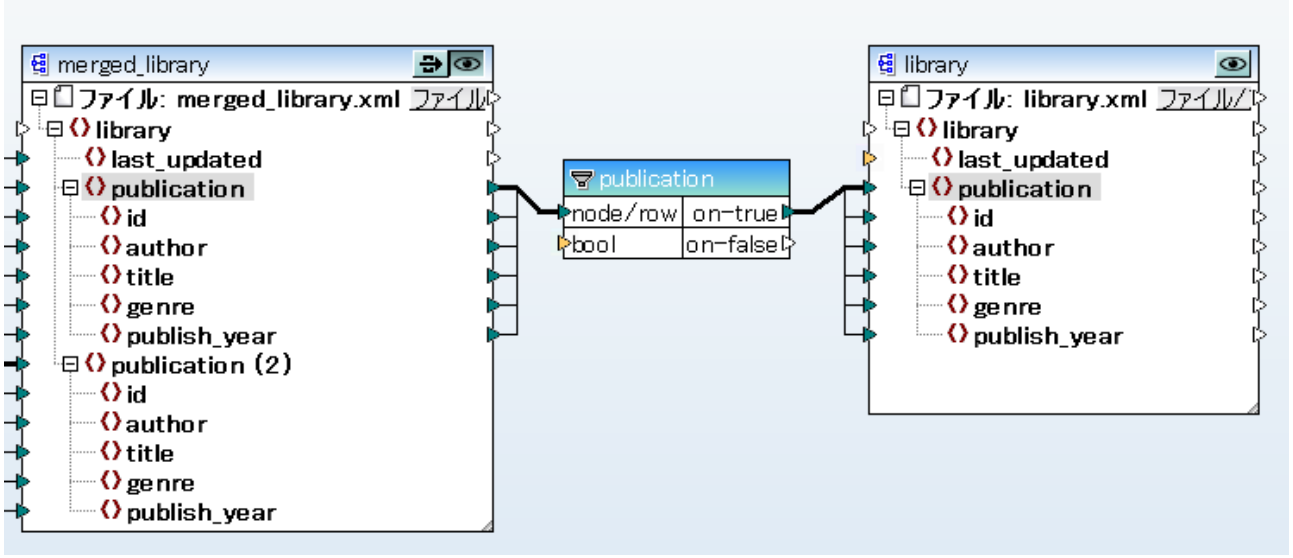


ステップ 4: データのフィルター

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

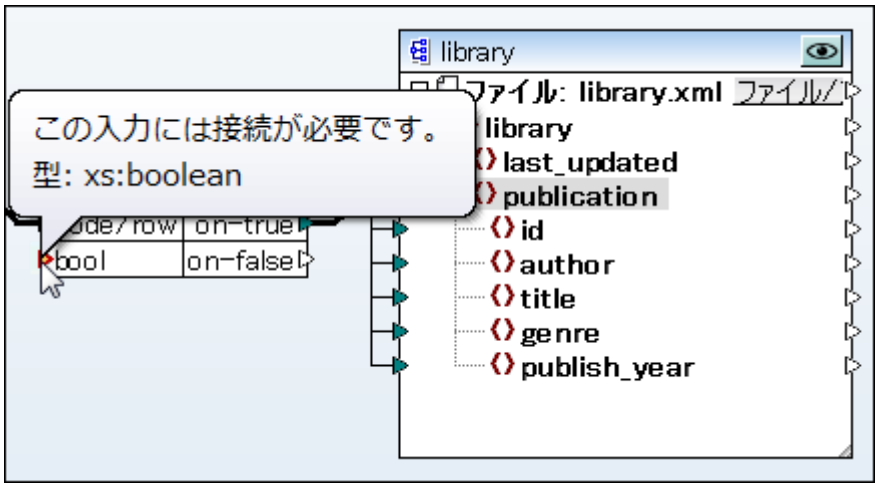


フィルターコンポーネントがマッピングに追加されました。



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

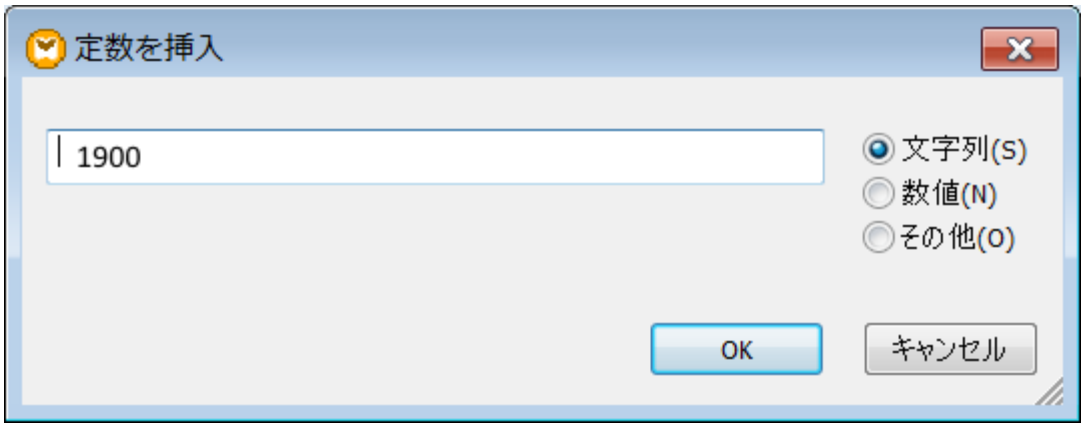
(C) 2015-2021 Altova GmbH



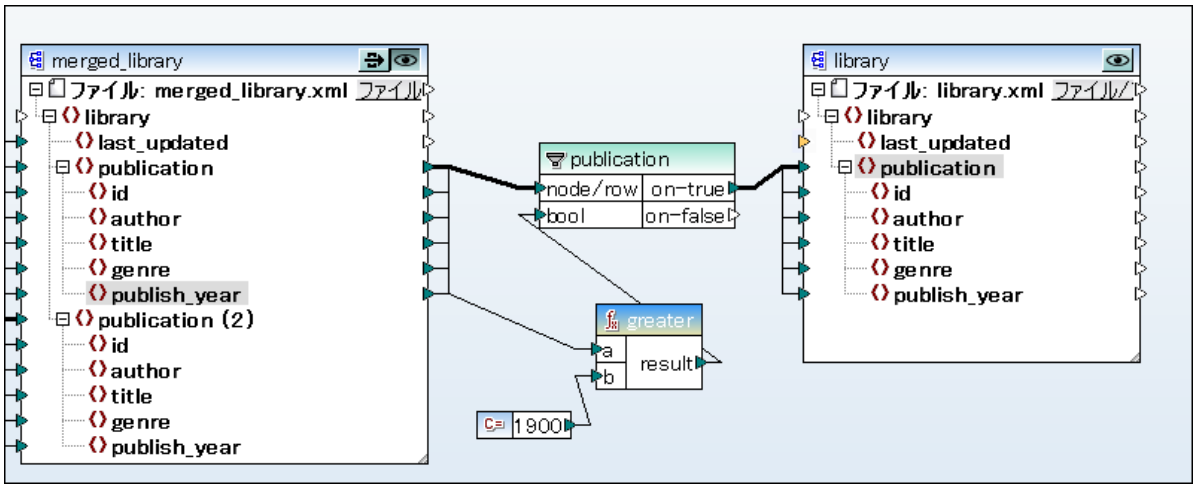
フィルターコンポーネントは`true` または`false` を返す条件を必要とします。ブール型の条件は、現在の**publication** シーケンスのデータがターゲットにコピーされると`true` を返します。条件が、`false` を返すと、データはコピーされません。

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

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



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



ステップ 5: 各ターゲットコンポーネントの出力をプレビューし保存する

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

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

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

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

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

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

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

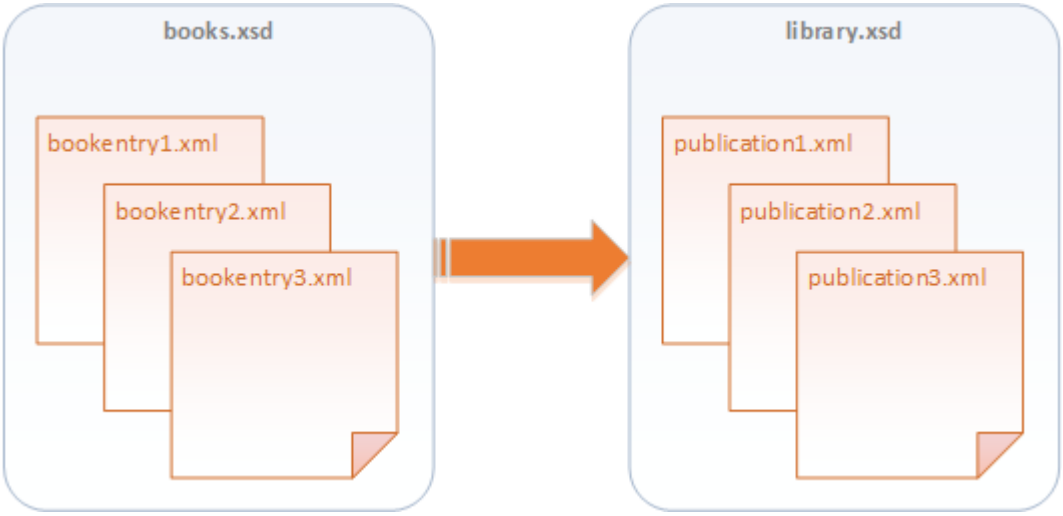
(C) 2015-2021 Altova GmbH

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

2.4 ファイルを動的に処理と生成する

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

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



データ変換の抽象的なモデル

3つのソースXMLファイルをサンプルとして使用します。ファイルの場所は**マイドキュメント >\Altova\MapForce2021\MapForceExamples\Tutorial** フォルダーで確認することができ**bookentry1.xml**、**bookentry2.xml**, と**bookentry3.xml** という名前が付けられています。これらのファイルはそれぞれ単一のブックを保管します。

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

bookentry1.xml

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

bookentry2.xml

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

bookentry3.xml

ソースXML ファイルは次のフォルダーで使用することのできる **books.xsd** スキーマを使用します: **<マイドキュメント>\Altova\MapForce2021\MapForceExamples\Tutorial**。ソースファイルを新規 XML スキーマに変換するには、(同じフォルダーで見つけることのできる) **library.xsd** スキーマを使用します。変換の後、マッピングは3つのファイルを新しいスキーマに従い生成します (下のコードリストを参照)。生成されたファイルが以下のようになるよう、マッピングを構成します: **publication1.xml**、**publication2.xml**、と**publication3.xml**。XML 宣言と名前空間が削除されなければならない点に注意してください。

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

publication1.xml

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

publication2.xml

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

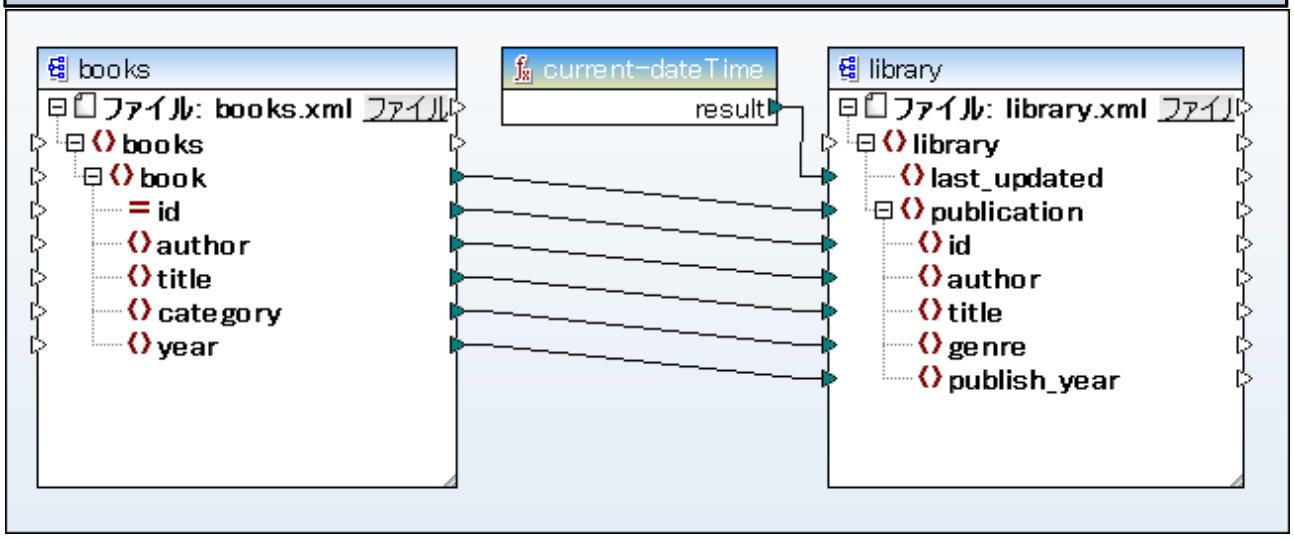
publication3.xml

チュートリアルの目的を達成するために、次のステップを踏みます.

ステップ 1: マッピングデザインファイルの準備

このチュートリアルは、開始のポイントとして<マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥Tutorial¥フォルダーから**BooksToLibrary.mfd** マッピングを使用しています。すでにXML を新しいスキーマに変換するチュートリアル内でこのマッピングをデザインしています。開始するには、MapForce で**BooksToLibrary.mfd** ファイルを開き、新しい名前で同じフォルダーに保存します。

複数のファイルを参照するため、<マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥Tutorial¥フォルダー内に新しいマッピングが保存されることを確認してください。



BooksToLibrary.mfd (MapForce Basic Edition)

ステップ 2: 入力を構成する

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

コンポーネント設定 ダイアログボックス

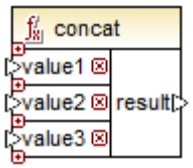
ファイル名内のアスタリスク（＊）ワイルドカード文字は、マッピング入力に**bookentry-** プレフィックスを伴う全てのファイルを使用するように命令します。パスは相対的であるため、MapForce はマッピングファイルと同じディレクトリ内の全ての**bookentry-** ファイルを検索します。マッピングファイル ＊ ワイルドカード文字を使用しているため、必要であれば、絶対パスを入力することもできます。

ステップ 3: 出力を構成する

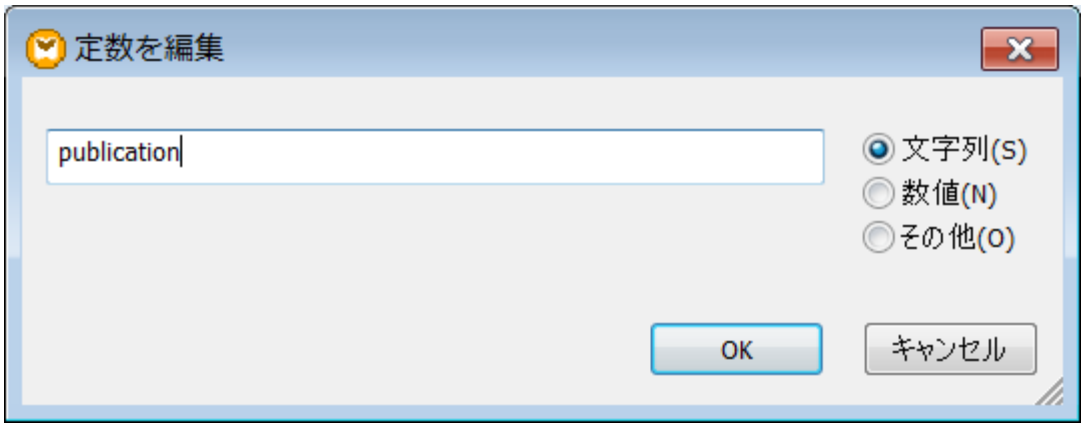
各出力ファイルのファイル名を作成するには、`concat` 関数 を使用します。この関数は引数に与えられた全ての値を連結します。

`concat` 関数を使用してファイル名を作成する

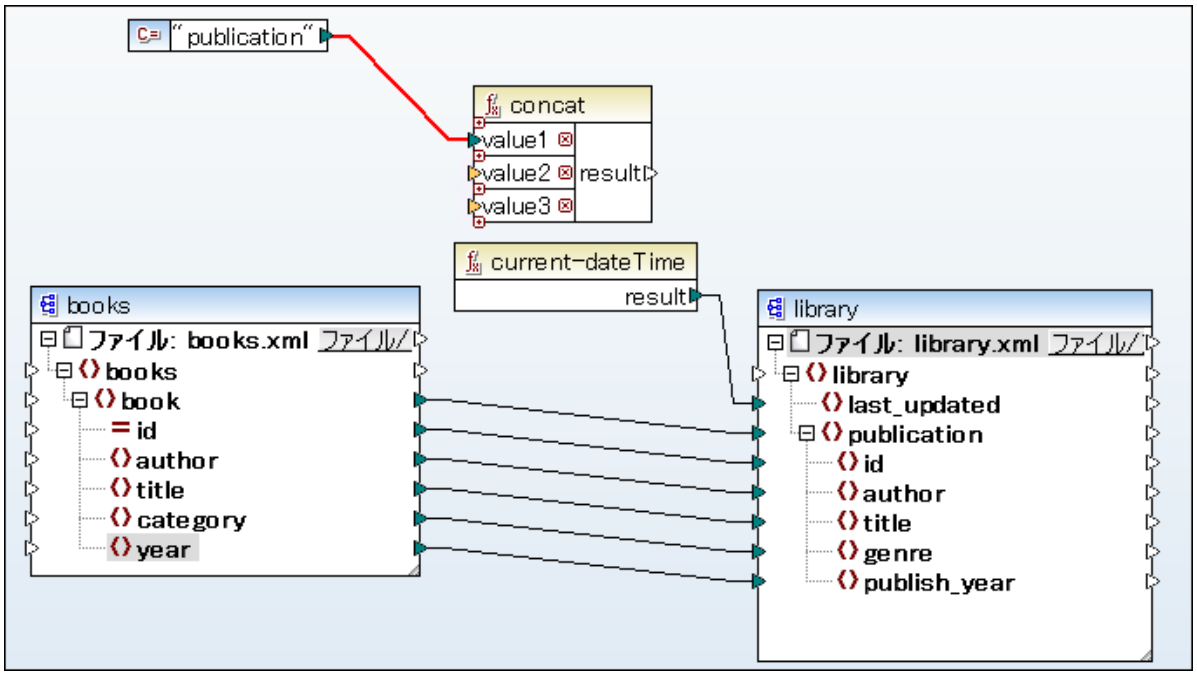
1. `concat` 関数 をライブラリウィンドウで検索し、マッピングエリアにドラッグします。デフォルトでは、この関数は 2 パラメータが 2 つあるマッピングに追加されます。しかし、必要であれば新しい パラメーターを追加することができます。関数コンポーネント内の「**パラメーターの追加**」（⊡）シンボルをクリックして、3番目の パラメーターを追加します。「**パラメーターの削除**」（⊠）シンボルをクリックすると パラメーターは削除されることに注意してください。



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

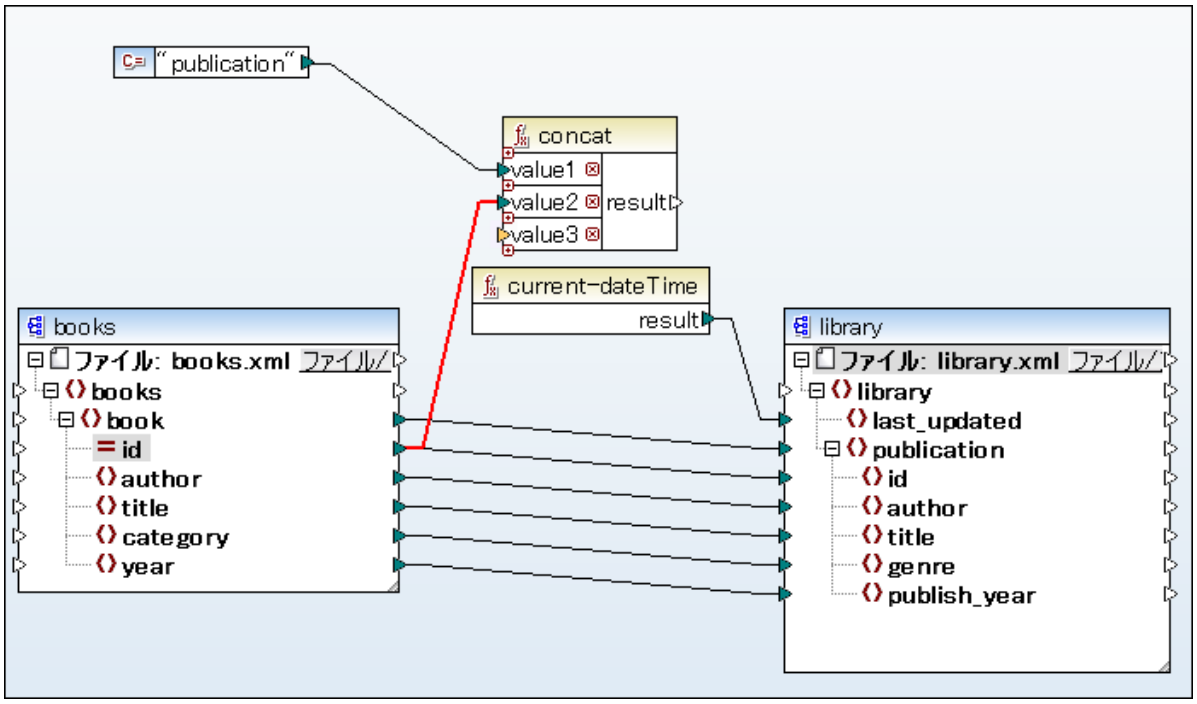


3. `concat` 関数の**value1** と定数をと接続します。

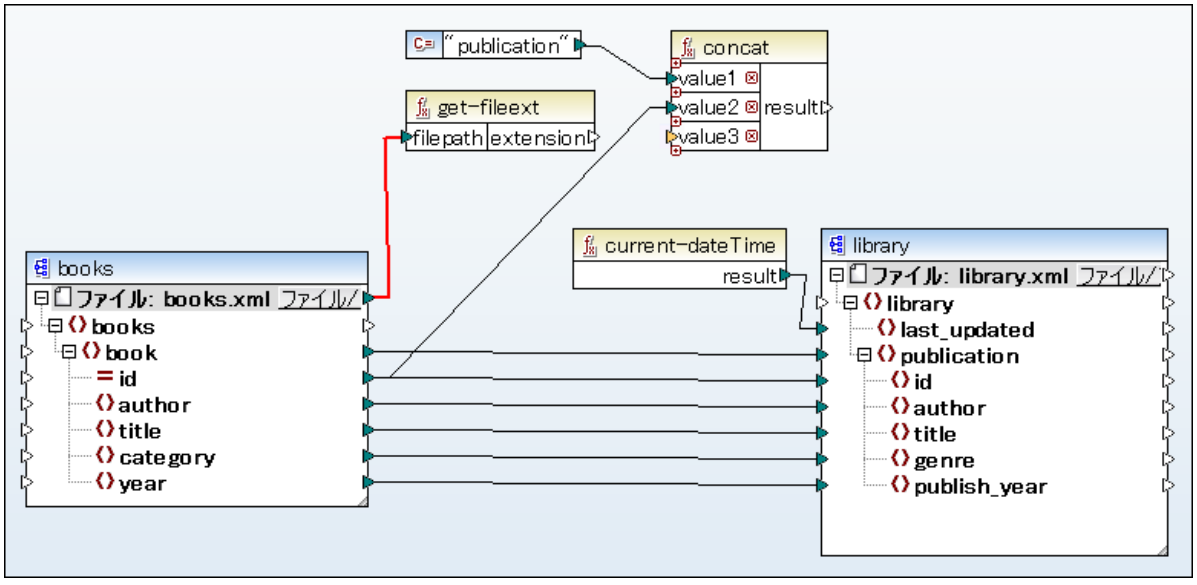


4. ソースコンポーネントの `id` 属性を `concat` 関数の**value2** と接続します。

(C) 2015-2021 Altova GmbH



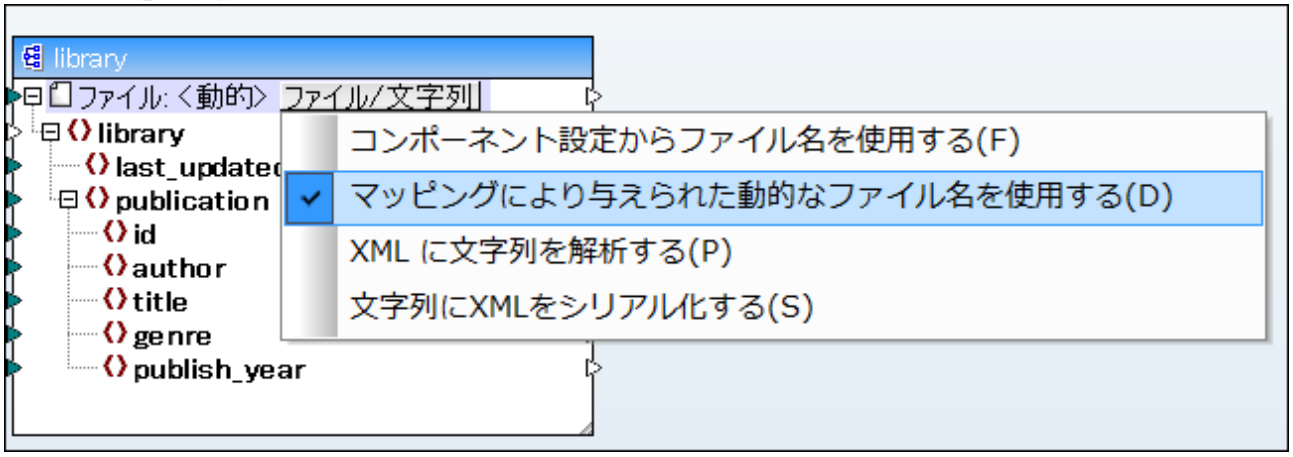
5. ライブラリウィンドウ内で `get-fileext` 関数を検索し、マッピングエリアにドラッグします。ソースコンポーネント（**File: books.xml**）のトップノードからこの関数の**filepath** パラメータへ接続を作成します。`get-fileext` 関数の結果から `concat` 関数の**value3** への接続を作成します。これを行うことにより、ソースファイル名から拡張子の部分（この場合、.xml）を抽出することができます。



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

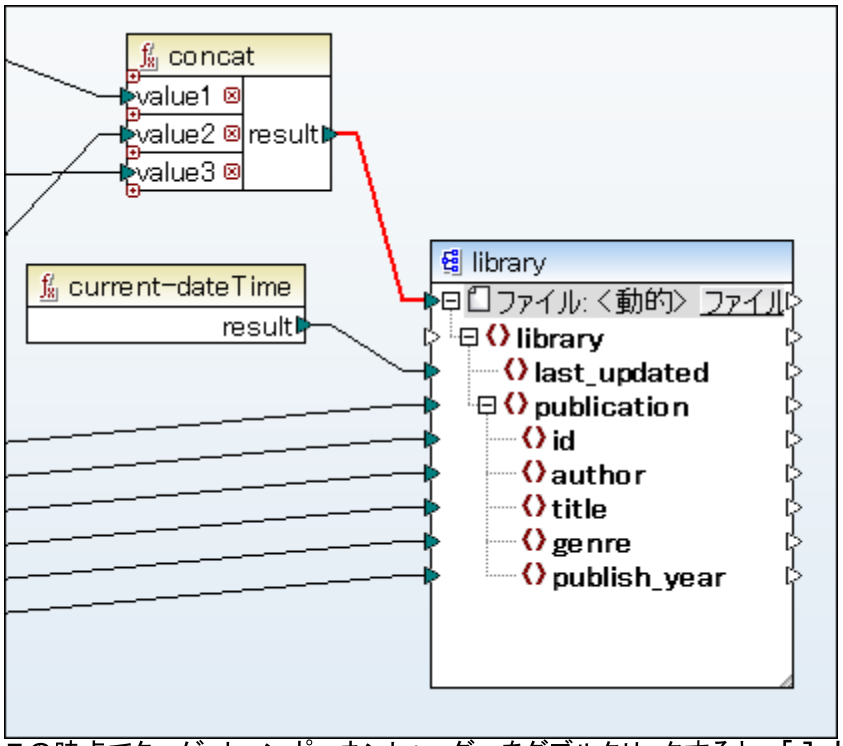
パート	サンプル
定数「publication」は定数文字列の値「publication」を与えます。	publication
ソースXMLファイルの属性 `id` は、各ファイルのための一意の識別子の値を与えます。これは全てのファイルが同じ名前で生成されないようにするためです。	1
`get-fileext` 関数は、生成されるファイルの拡張子を返します。	.xml

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

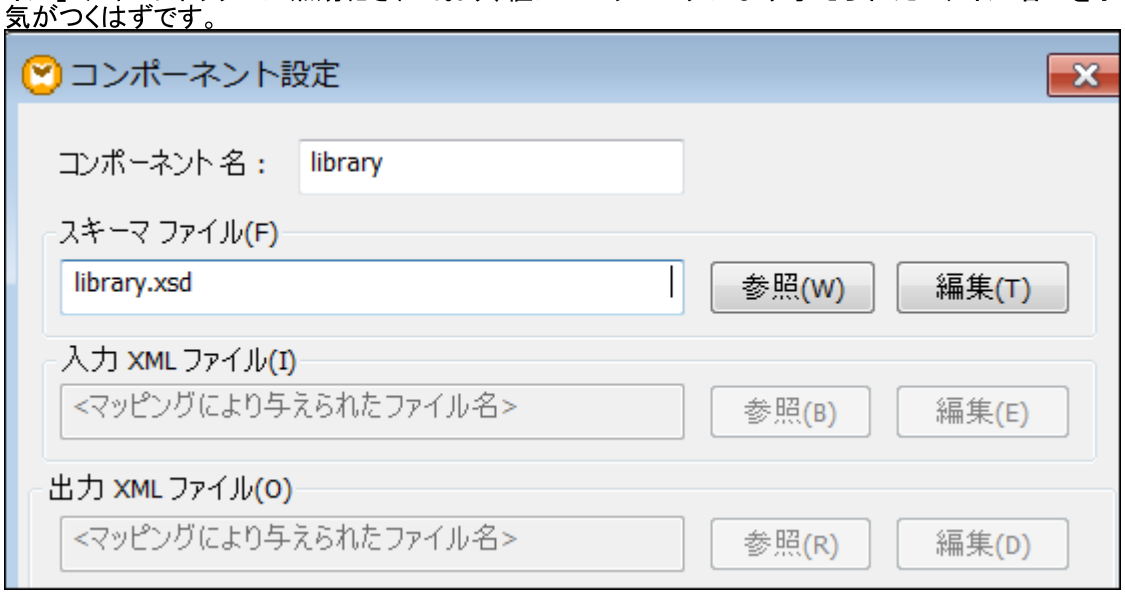


マッピングにより与えられた名前を伴うMapForceにインスタンスファイルを動的に生成するように命令しました。この特別な例では、`concat` 関数により名前が作成されました。ですから、`concat` 関数の結果とターゲットコンポーネントのファイル: <動的> ノードを接続します。

(C) 2015-2021 Altova GmbH

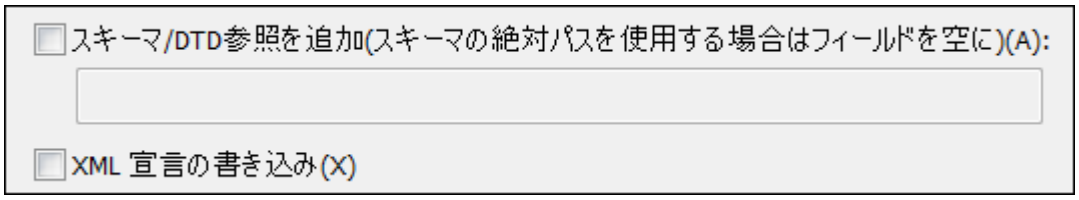


この時点でターゲットコンポーネントヘッダーをダブルクリックすると、「入力 XML ファイル」と「出力 XML ファイル」テキストボックスが無効化されており、値が <マッピングにより与えられたファイル名> を示していることに気がつくはずです。

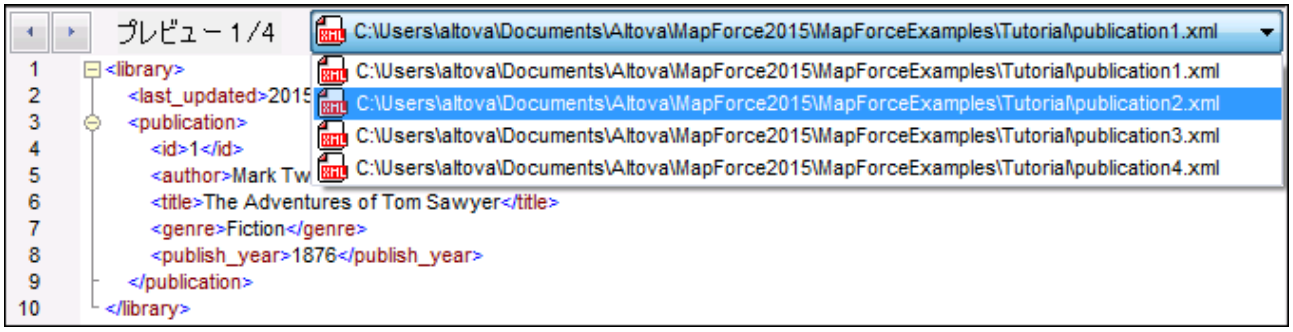


これは、インスタンスファイル名を動的にマッピングから提供したことを示し、コンポーネント設定内で定義することは関連性がありません。

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



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



(C) 2015-2021 Altova GmbH

3 一般的なタスク

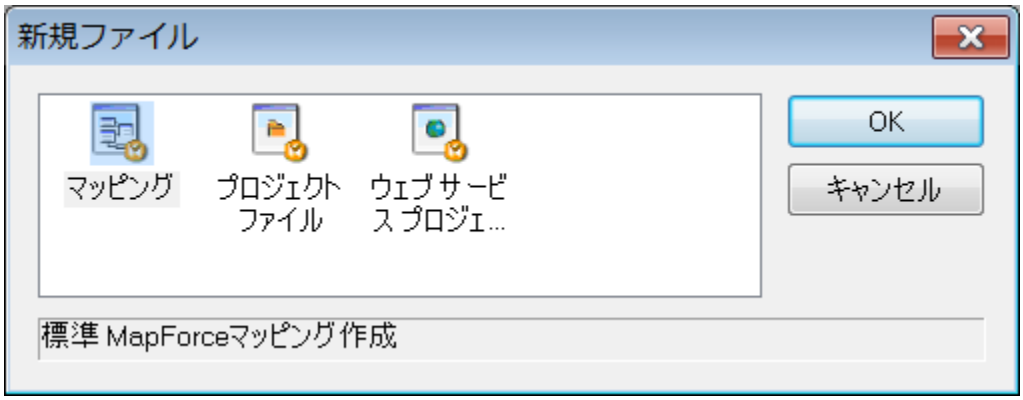
このセクションは、マッピングとの作業、コンポーネント、接続、マッピングプロジェクトなどのMapForce の全般的なタスクとコンセプトを説明します。

3.1 マッピングとの作業

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

新規マッピングの作成方法:

1. 以下の内の1つを行ってください：
   - 「ファイル」メニューから「新規作成」をクリックします。
   - 「新規作成」（ ）ツールバーボタンをクリックします。
2. 「マッピング」をクリックして、, and then click「OK」をクリックします。



マッピングが作成されました、しかしながら、空のため何も起こりません。最低でも妥当なマッピングは2つの接続されたコンポーネント を必要とし次のステップはマッピングにコンポーネントを追加 し 接続線を描きます。

3.1.1 マッピングにコンポーネントを追加する

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

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

(C) 2015-2021 Altova GmbH

- 変数
- XBRL ドキュメント
- スキーマおよびDTD

マッピングにコンポーネントを追加するには以下を行います:

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

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

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

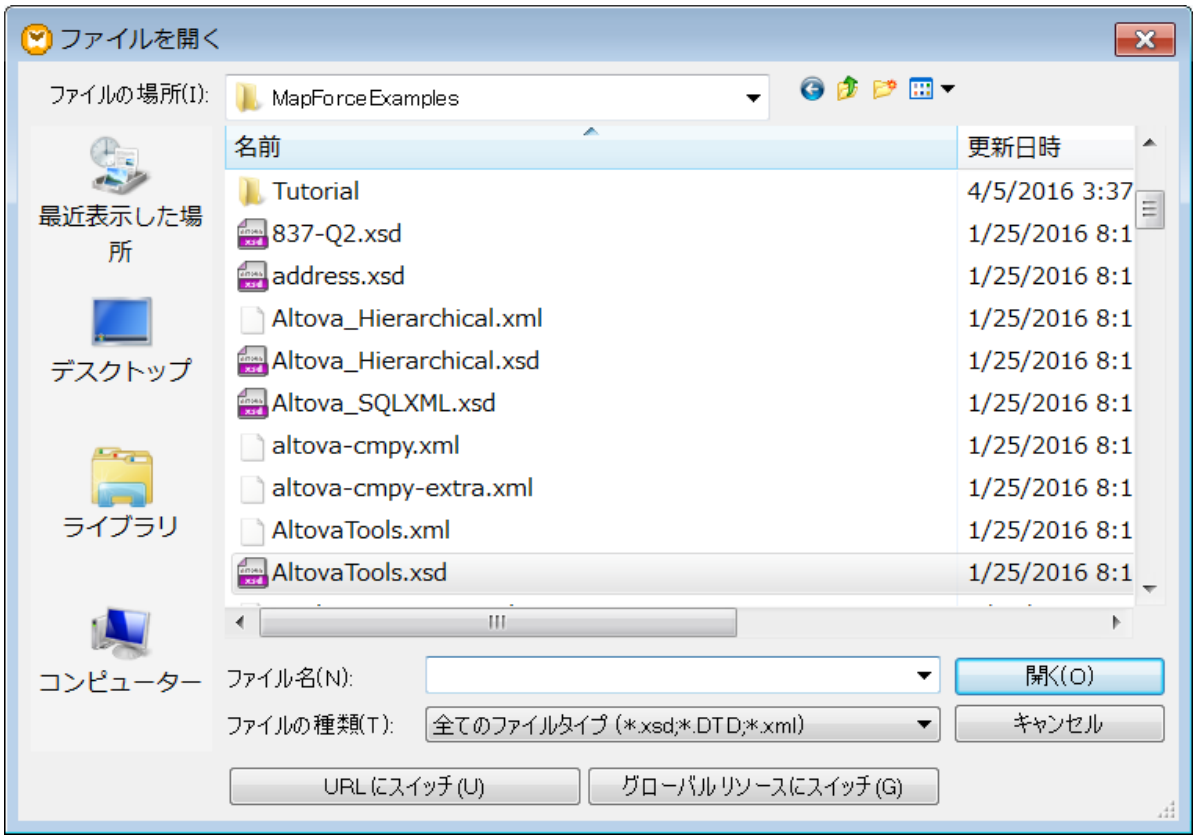
コンポーネントについての説明は、 コンポーネントとの作業を参照してください。マッピングソースまたはターゲットとしてサポートされる個々の技術に関しては、データソースとターゲットを参照してください。データを一時的に保管または変換する MapForce ビルドインコンポーネントに関しては、または(フィルタリングまたは並べ替えなどの) マッピングのデザインを参照してください。

3.1.2 URL からコンポーネントを追加する

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

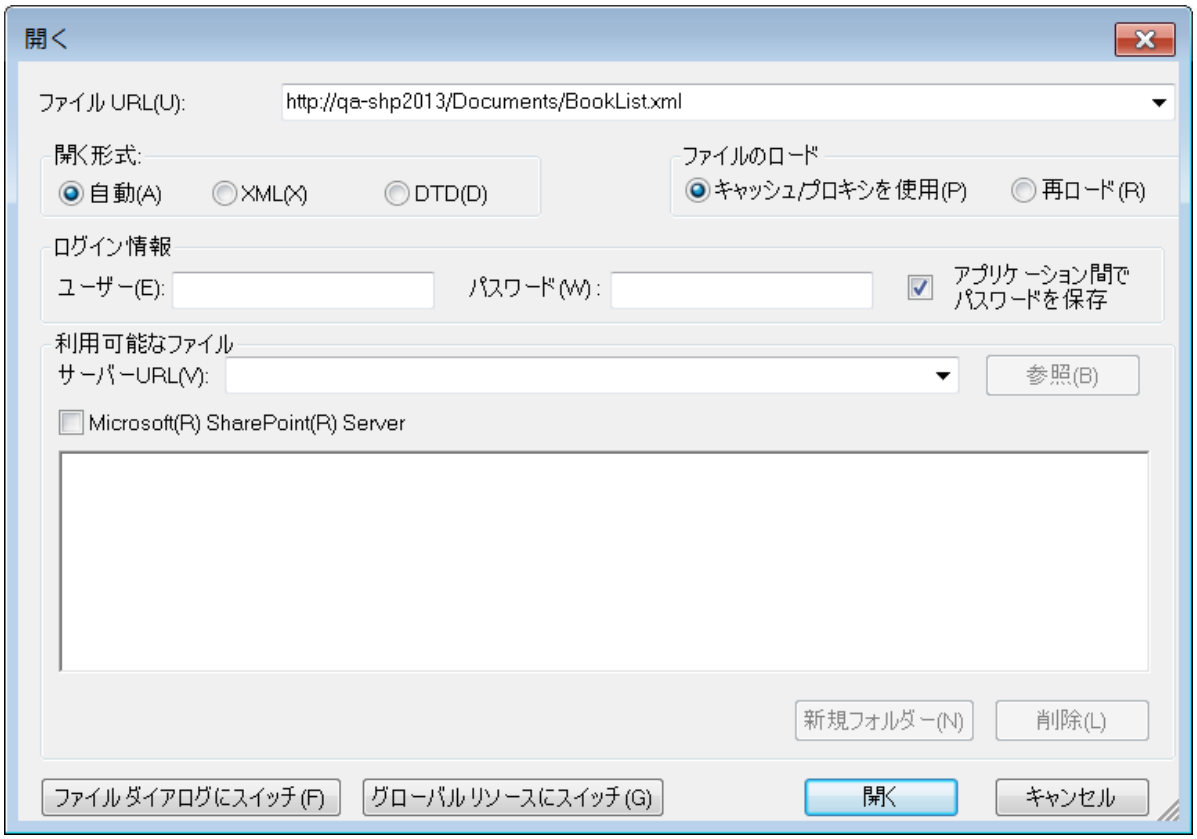
URL からコンポーネントを追加する:

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



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

(C) 2015-2021 Altova GmbH



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

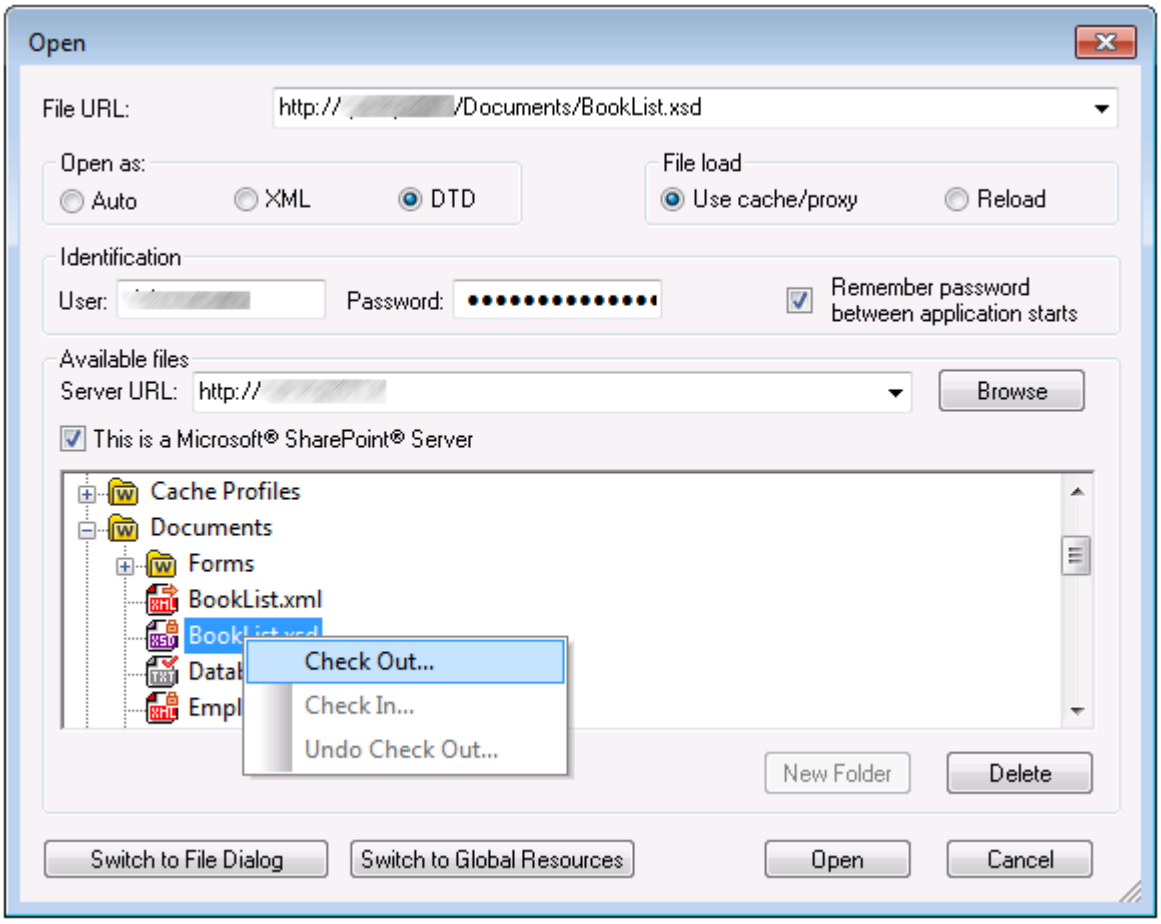
サーバーがパスワード認証を必要とする場合、ユーザー名とパスワードを入力するようにプロンプトされます。次回 MapForce を起動する際に使用できるように、ユーザー名とパスワードを保存するには、「開く」ダイアログボックス内の**「アプリケーション間でパスワードを保存」**チェックボックスを選択します。

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

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

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

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



開くダイアログボックス（URL に切り替える モード）

3.1.3 データストリーミングについて

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

データストリーミングは次のデータソースに適用することができます:

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

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

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

メモリ使用の注意点

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

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

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

3.1.4 変換言語の選択

データ変換言語として次を選択することができます:

- XSLT 1.0
- XSLT 2.0
- XSLT 3.0
- XQuery
- C++
- C#
- Java
- BUILT-IN (MapForce のデフォルトのネイティブ変換エンジン)

変換言語を選択するには、以下を行います:

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

マッピングの変換言語を変更する場合、MapForce 機能の一部はその言語のためにサポートされない場合があります。BUILT-IN は他の言語と比較して、MapForce 機能の多くをサポートします。詳細に関しては、サポートメモを参照してください。

BUILT-IN 変換言語

マッピングの変換言語として BUILT-IN

を選択すると、MapForce はネイティブの変換言語を使用してデータマッピングを実行します。選択された変換言語が Java、C#、または C++ の場合、マッピングの出力をプレビューすると MapForce はこのオプションを明示的に使用します。

BUILT-IN エンジンは外部プロセッサーを必要とすることなくマッピングを実行し、これはメモリの使用が考慮される場合に良いオプションです。特定の言語でプログラムコードを生成する必要がない場合、サポートメモで説明される通り、他の言語と比較して MapForce の多数の機能をサポートするため BUILT-IN をデフォルトのオプションとして使用します。更に、BUILT-IN を変換言語として選択すると、MapForce Server を使用した実行のためにマッピングを使用することができます。更に、マッピングと MapForce の自動化を参照してください。

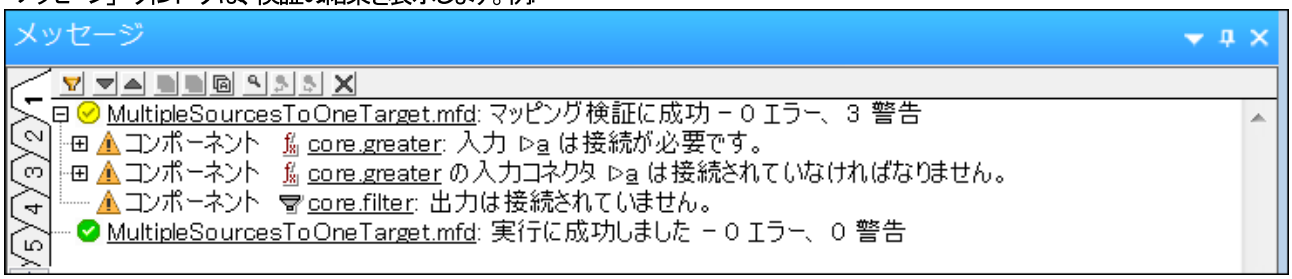
3.1.5 マッピングの検証

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

マッピングを正確に検証するには、以下を行います:

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

「メッセージ」ウィンドウは、検証の結果を表示します。例:



メッセージウィンドウ

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

アイコン	意味
	変換は成功しました。
	変換は警告と伴い完了しました。
	変換は失敗しました。

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

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

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

(C) 2015-2021 Altova GmbH

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

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

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

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

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

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

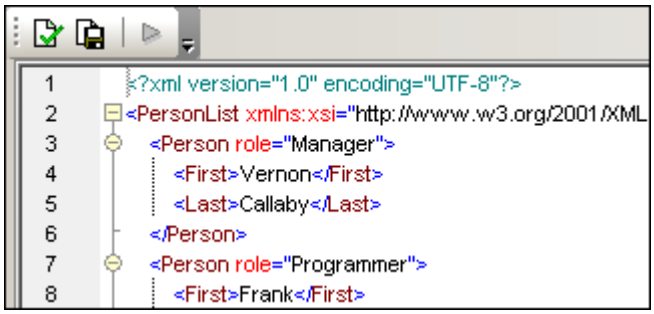
3.1.6　マッピング出力の検証

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

XML ファイルに関しては、コンポーネント設定ダイアログボックスの「**スキーマ/DTD レファレンスの追加**」フィールド内 のインスタンスファイルに関連するスキーマを指定することができます（XML コンポーネント設定 を参照）。生成されたXML 出力により参照されるスキーマファイルをパスは指定します。これにより、マッピングの実行時、出力インスタンスが検証されることができます。このフィールドに`http://`、絶対または相対パスを入力することができます。「**スキーマ/DTD レファレンスの追加**」フィールドを選択しない場合、スキーマに対する、出力ファイルの検証は不可能です。このチェックボックスを選択し、空白のままにしておくと、検証が終わると、コンポーネント設定ダイアログボックスのスキーマファイル名が出力に生成されます。

コンポーネント設定を開くには、以下を行います:

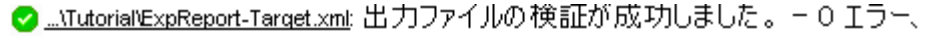
- 「**出力の検証**」 ツールバーボタンをクリックします。



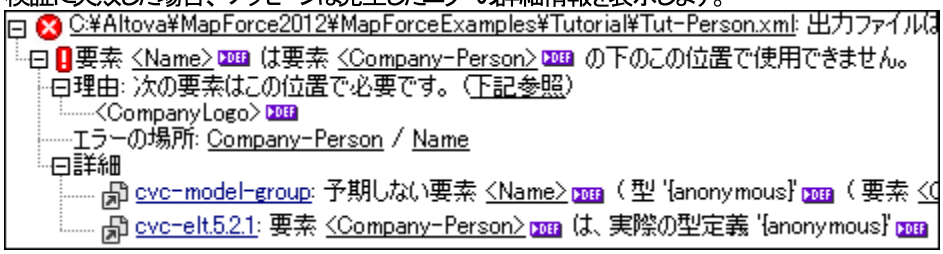
- 「**出力**」メニューから「**出力ファイルの検証**」をクリックします。

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

検証の結果はメッセージウィンドウ内に表示されます。例:



検証に失敗した場合、メッセージは発生したエラーの詳細情報を表示します。



検証メッセージは更に詳しい詳細情報を含む多くのハイパーリンクを含みます。:

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

3.1.7 出力のプレビュー

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

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

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

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

変換の出力をプレビューする方法:

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

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

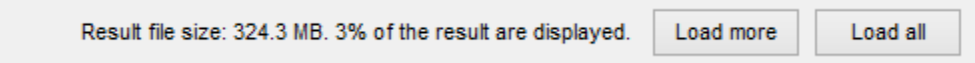
変換の出力を保存するには、以下を行います:

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

(C) 2015-2021 Altova GmbH

部分的な出力のプレビュー

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



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

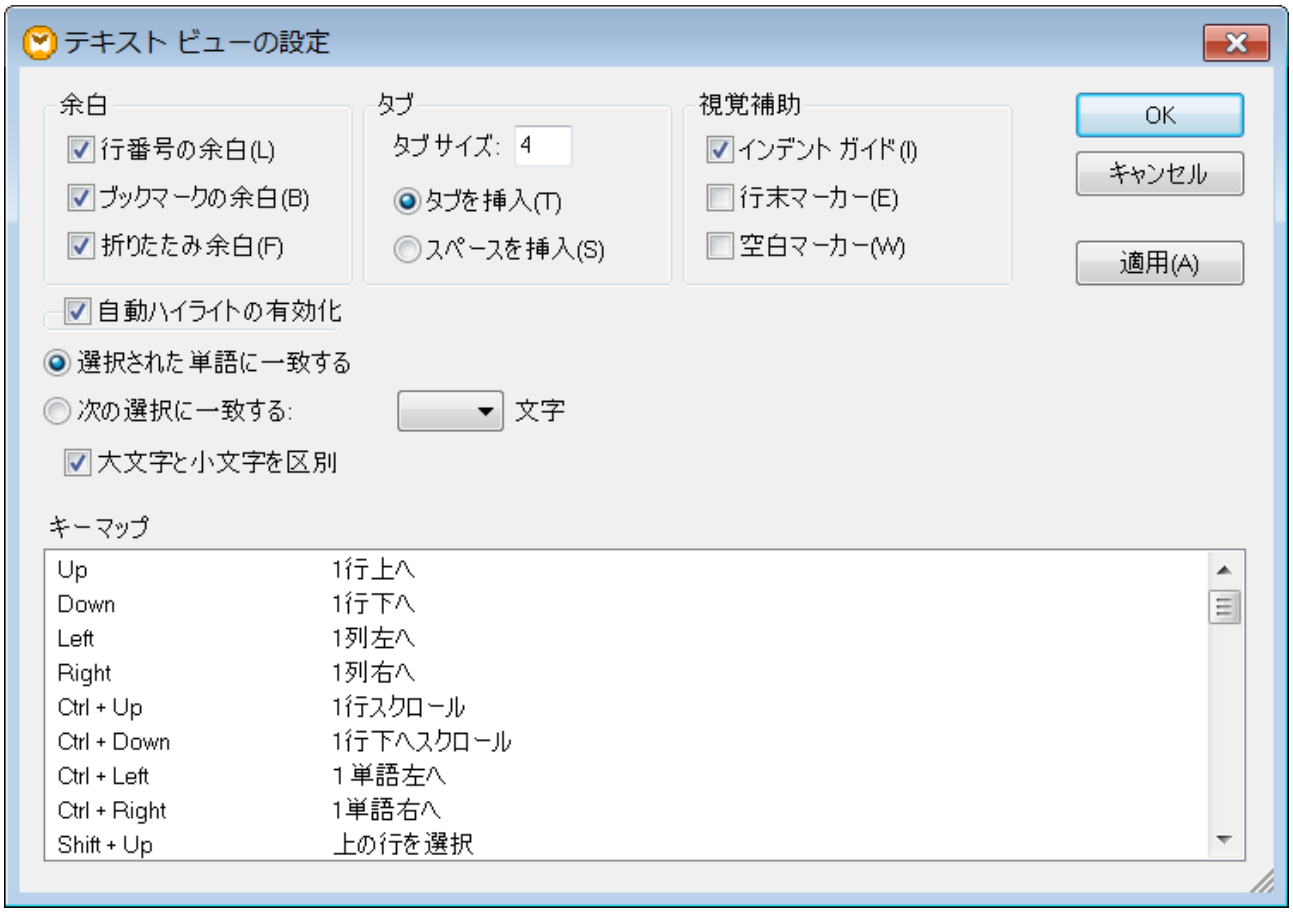
オプションダイアログボックスの**全般**タブからプレビューの設定を構成することができます（MapForce オプションの変更を参照）。

3.1.8　テキストビューの機能

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

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

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



テキストビュー設定 ダイアログボックス

「テキストビュー設定」ダイアログボックスを開くには、以下を行います:

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

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

テキストビューツールバー

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

行番号

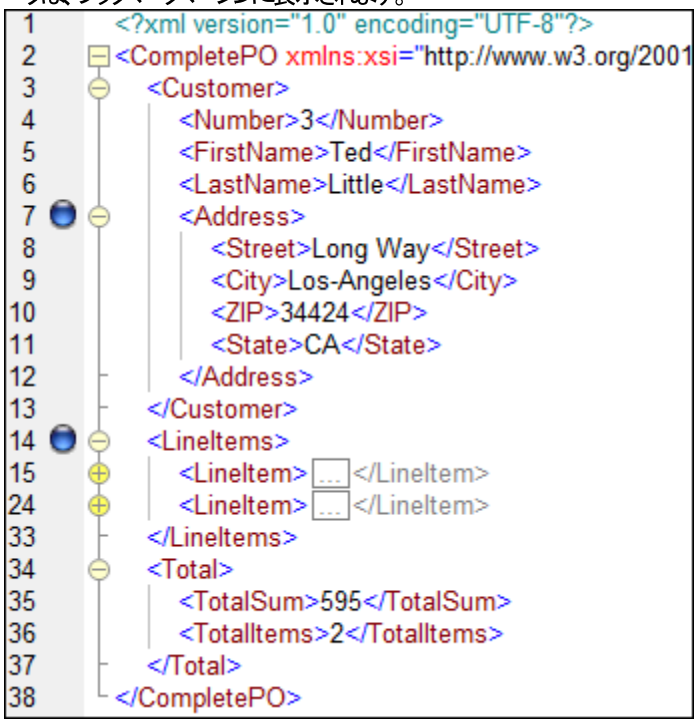
行番号は、**「テキストビュー設定」**ダイアログボックス内で切り替えることのできる行番号 マージンで表示されます。テキストのセクションが折りたたまれると、折りたたまれたテキストの行番号は表示されません。

構文の色

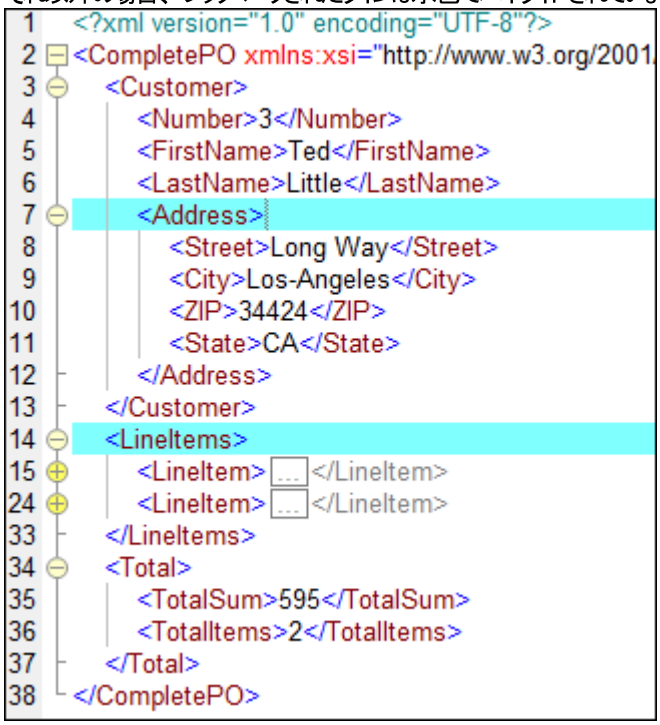
構文の色分けはテキストの構文の値に従い適用されます。例えば、XML ドキュメント内では、XML ノードが要素、属性、コンテンツ、CDATA セクション、コメント、処理命令、ノード名(そして一部の場合ではノードのコンテンツ)により異なる色分けが使用されています。

ブックマーク

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



それ以外の場合、ブックマークされたラインは水色でハイライトされています。



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

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

ブックマークの挿入/削除 (Ctrl + F2)

次のブックマークへ移動 (F2)

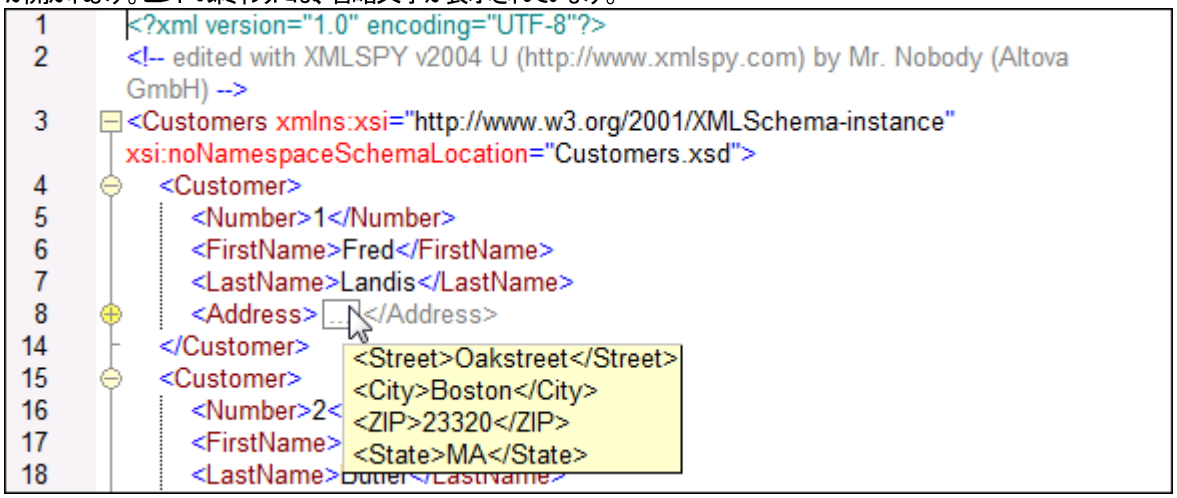
前のブックマークへ移動 (Shift + F2)

全てのブックマークを削除 (Ctrl + Shift + F2)

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

ソースの折りたたみ

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



インデントのガイド

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

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

行末のマーカーと空白文字のマーカー

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

(C) 2015-2021 Altova GmbH

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

ズームインとズームアウト

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

整形出力

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

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

ワード ラップ

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

テキストのハイライト

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

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

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

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

3.1.9　テキストビュー内の検索

「出力」ペイン、「XQuery」ペイン、および「XSLT」ペイン 内のテキストは、広範囲のオプションの組み合わせと視覚的な補助を用いて、検索をすることができます。

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

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

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

以下の点に注意してください:

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

(C) 2015-2021 Altova GmbH

- 検索ダイアログが閉じられると、現在の検索は **F3** を押すことにより順方向検索し、**Shift+F3** を押すことにより逆方向検索します。検索ダイアログが表示されます。

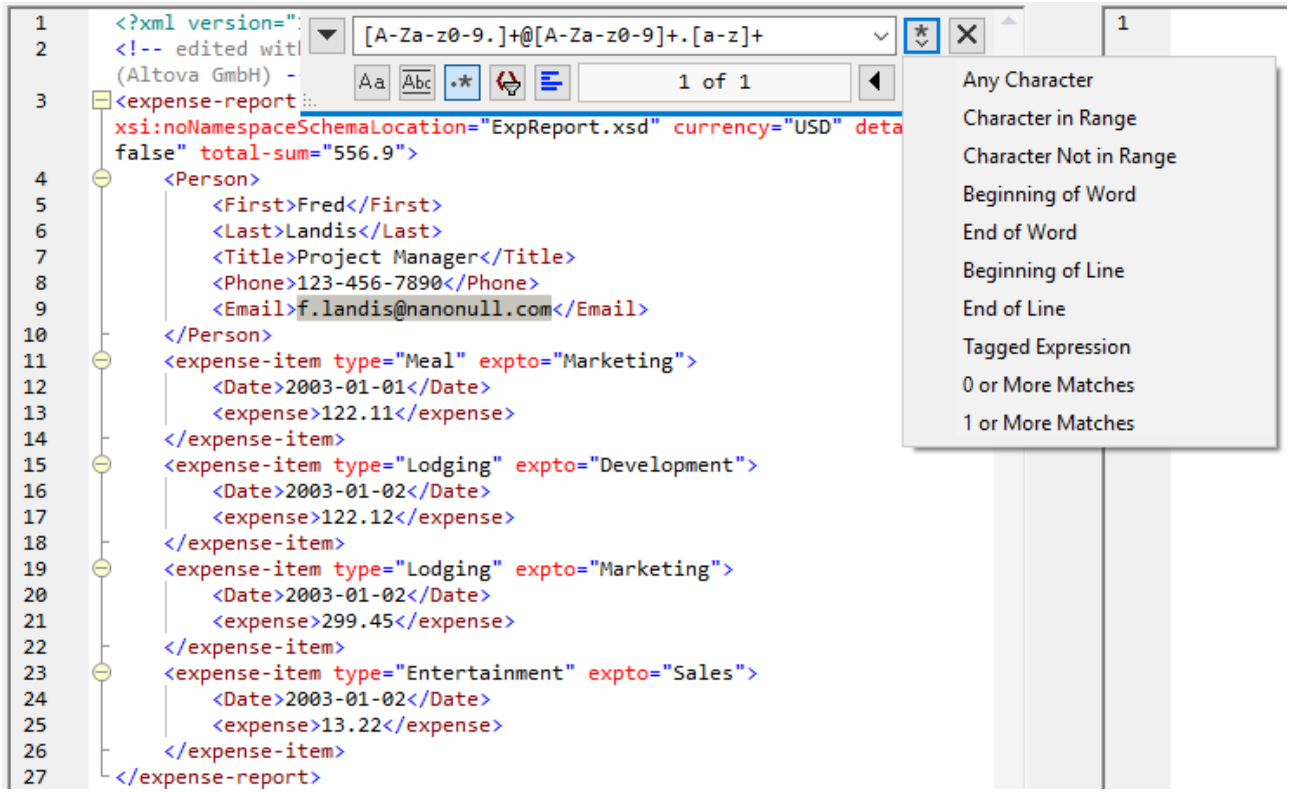
検索オプション

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

オプション	アイコン	説明
大文字と小文字を区別する		切り替えられると、大文字と小文字を区別する検索が行われます（「Address」は「address」とは異なります）。
単語単位で検索		テキスト内の文字のみが一致されます。例えば、入力文字列 *fit* に対して、**単語単位を一致する** がオンにされていると、単語 *fit* のみが検索文字列に一致します。*fitness* 内の *fit* は一致しません。
正規表現		オンに切り替えられると、検索用語は、正規表現として読み取られます。「正規表現の使用」を参照してください。
アンカーの検索		検索用語が入力されると、ドキュメント内の一致がハイライトされ、一致の内の1つが現在の選択としてマークされます。**アンカーの検索** の切り替えは、最初の選択がカーソルの位置に対して相対的かを決定します。**アンカーの検索** がオンに切り替えられると、現在選択されている一致が選択され、現在のカーソルの場所の次の一致に一致します。**アンカーの検索** がオフに切り替えられていると、現在選択されている一致はドキュメントの最初から数えて最初の一致が一致します。
選択範囲内の検索		オンに切り替えられると、現在のテキストの選択範囲をロックし検索を選択されている範囲に制限します。それ以外の場合、ドキュメント全体が検索されます。テキストの新しい範囲を選択する前に、選択範囲内の検索 オプションをオフに切り替えて現在の選択範囲のロックを解除します。

正規表現の使用

正規表現（regex）を使用して、テキスト文字列を検索することができます。これを行うには、最初に *正規表現* オプションをオンに切り替えます。これは検索用語フィールド内のテキストが正規表現として評価されるように指定します。次に正規表現と検索フィールドに入力します。正規表現の作成をヘルプするために、検索用語フィールドの右にある **正規表現 ビルダー** ボタンをクリックします。これにより、検索用語フィールド内のテキストが正規表現として評価されます。下のスクリーンショットは、電子メールアドレスを検索するための簡単な正規表現を表示しています。



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

.	任意の文字を一致する。これは単一の文字のプレースホルダーです。
(abc)	(and) メタ文字は、タグ付けされた式の開始と終了をマークします。一致する箇所を、後で参照する(バックレファレンス)目的のためにタグ(「記録」) する場合、役に立つ可能性があります。9個までのサブ式をタグ付け(そして後から参照)することができます 例えば、 **(the) \1** は、文字列 the the に一致します。この式は、以下のように説明することができます: 以前に一致したタグ付けされた箇所が後に続き、スペース文字が後に続く、文字列「the」を一致、(および、タグされた箇所として記録)します。
\n	n が 1 から 9 の箇所では、n は、最初から9番目のタグ付けされた箇所を指します(上を参照してください)。
\<	単語の先頭に一致。
\>	単語の末尾に一致。
\	バックスラッシュが後に置かれている文字をエスケープします。すなわち、式 ¥x は、文字 x を文字通り使用することができます。例えば、 \[は、文字のセットの開始としてではなく、 [として解釈されます。
[...]	このセット内の文字に一致します。例えば、 **[abc]** は、a, b または c の文字に一致します。範囲を使用することができます: 例えば、小文字のために **[a-z]** を使用します。
[^...]	このセット内では内文字に一致します。例えば、 **[^A-Za-z]** は、アルファベット文字以外の文字に一致します。
^	(上記のとおり)セット内で使用されていない限り)行頭 に一致します。

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

特別文字の検索

3.1.10 XSLT コードのプレビュー

XSLT 1.0、XSLT 2.0、またはXSLT 3.0 をデータ変換言語変換言語として選択している場合、MapForce により生成されたXSLT コードをプレビューすることができます。

XSLT コードをプレビューするには、マッピングウィンドウの下の**XSLT** タブ（または**XSLT2** または**XSLT3**）タブを必要に応じてクリックします。

メモ　XSLT 1.0、XSLT 2.0, またはXSLT 3.0 変換言語として選択すると使用することができます。

3.1.11 XSLT コードの生成

マッピングは対応する言語のためにデザインされていると仮定して、マッピングからXSLT 1.0、XSLT 2.0、またはXSLT 3.0 コードを生成することができます。変換言語の選択も参照してください。

XSLT コードを生成する:

1. メニューアイテム「ファイル | コードの生成 | XSLT 1.0（または**XSLT 2.0**、**XSLT 3.0**）を選択します。
2. 生成されたXSLT ファイルを保存するフォルダーを選択し、「OK」をクリックします。 MapForce は、コードを生成し、メッセージウィンドウ内にオペレーションの結果を表示します。

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

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

生成されたコード内のライブラリパスも参照してください。

RaptorXML Server を使用して自動化する

XSLT コードの生成後、.xslt ファイルを持つ同じディレクトリ内に **DoTransform.bat** という名前のバッチファイルが生成されます。RaptorXML Server を使用して **DoTransform.bat** を実行することができます。RaptorXML Server を使用した自動化も参照してください。

RaptorXML Server を使用して変換を行う:

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

RaptorXML がインストールされた場所をオペレーティングシステムの **path** 環境変数に追加する必要がある場合があります。RaptorXML ドキュメントを確認することができます（https://www.altova.com/ja/documentation）。

3.1.12 XQuery コードのプレビュー

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

生成された XQuery コードをプレビューする方法:

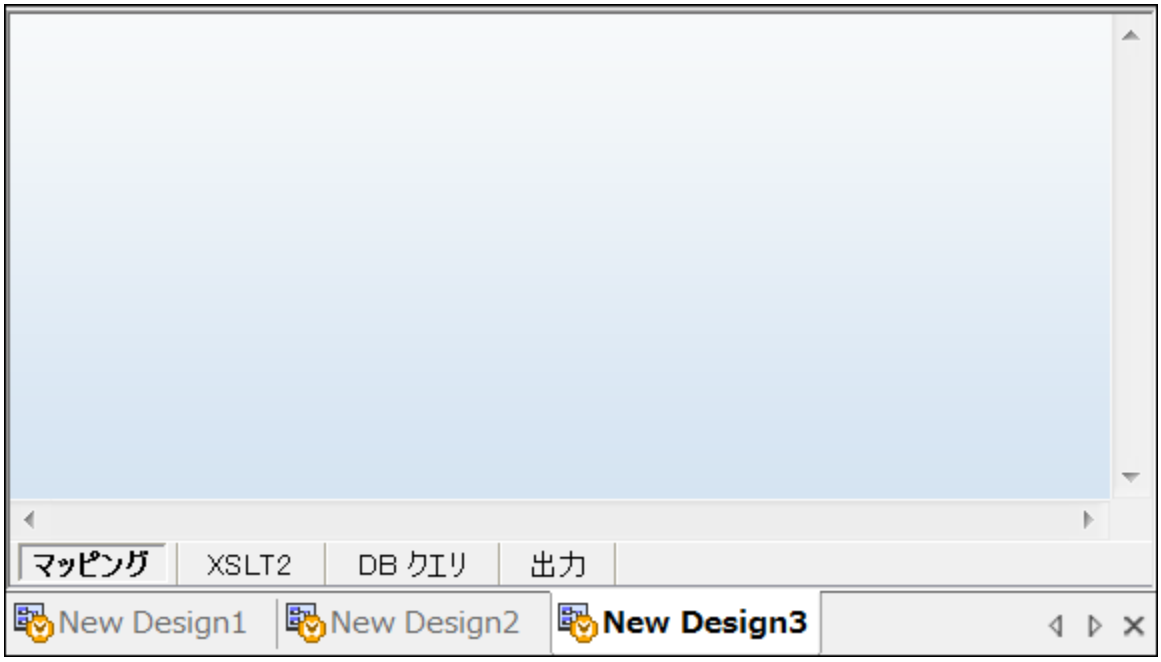
- マッピングウィンドウの下にある「XQuery」タブをクリックします。

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

3.1.13 複数のマッピングウィンドウとの作業

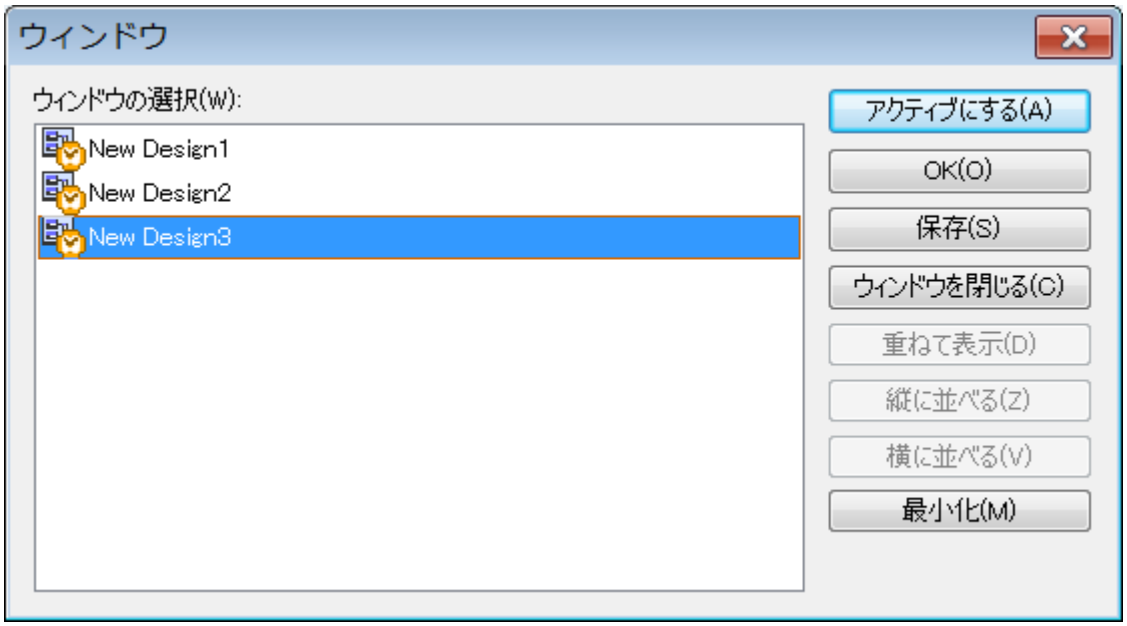
MapForce は、複数のドキュメントインターフェイス（Multiple Document Interface）（MDI） を使用します。MapForce で開く各マッピングファイルには個別のウィンドウがあります。これにより、複数のマッピングウィンドウと作業、または並べ替えやサイズ調整を MapForce メイン（親） ウィンドウで行うことができます。で行うことができます。全ての開かれているウィンドウを標準ウィンドウのレイアウトを使用して以下のように並べ替えることができます: 上下に並べる、左右に並べる、重ねて表示。

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



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

(C) 2015-2021 Altova GmbH



ウィンドウダイアログボックス

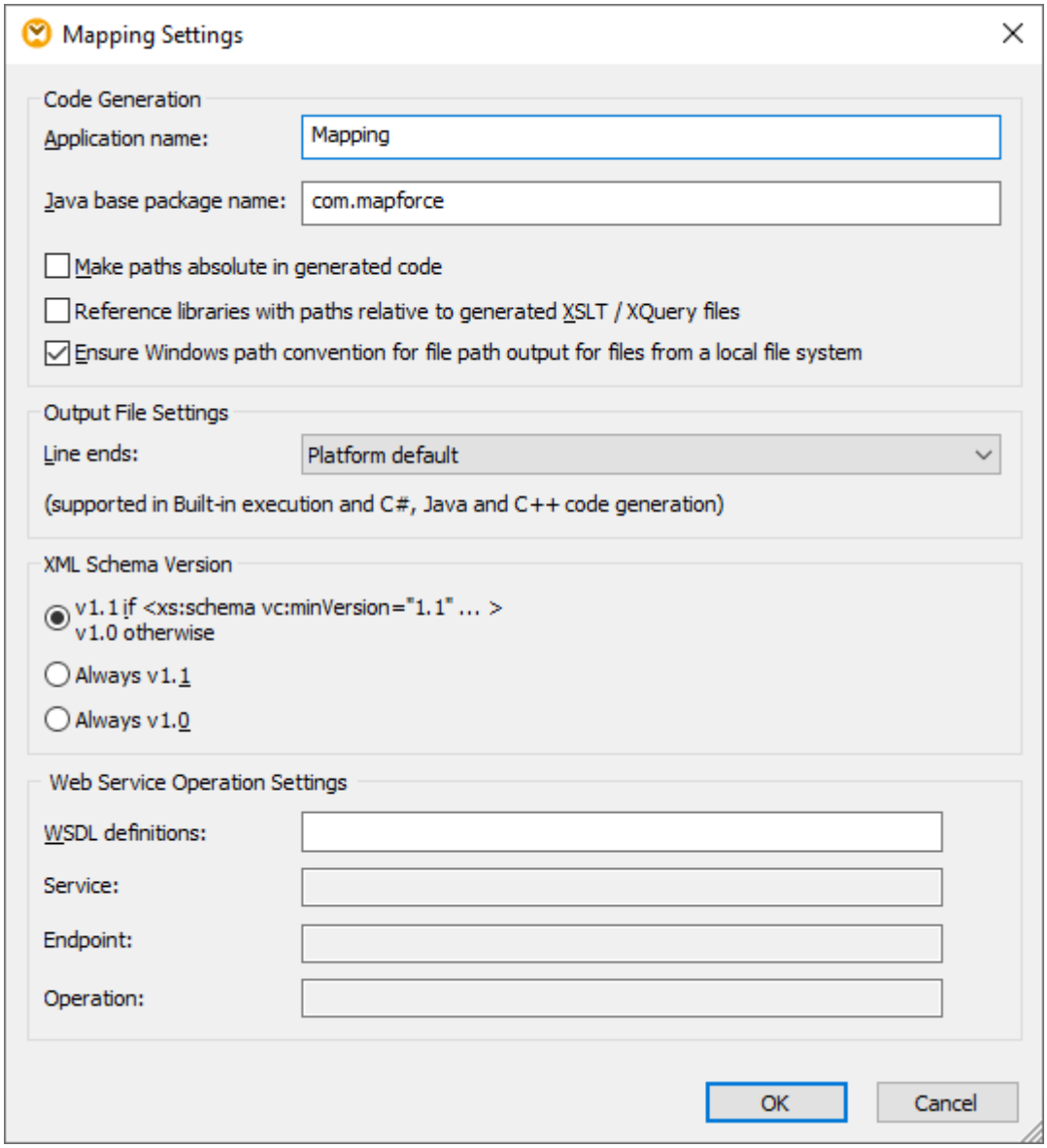
「**ウィンドウ**| **ウィンドウ**」を使用して、ウィンドウダイアログボックスを開くことができます。ウィンドウダイアログボックスから複数のウィンドウを選択するには、必要なエントリを「Ctrl」キーを押しながらクリックしてください。

3.1.14　マッピング設定の変更

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

マッピング設定ダイアログボックスの開きかた

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



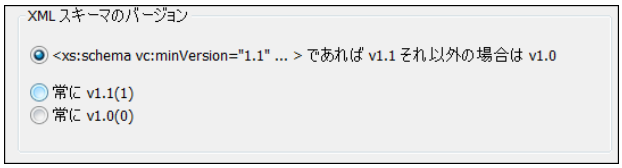
マッピング設定ダイアログボックス

使用することのできる設定は以下の通りです。

アプリケーション名	生成されたXSLT ファイルまたは生成されたJava、C#、またはC++ アプリケーションの名前を定義します。
ベース パッケージ名	このオプションはJava が変換言語として選択されている場合に適用することができます。Java 出力のためのベース パッケージ名を定義します。
生成されたコード内でパスを絶対パスにする	このチェックボックスは(XSLT ライブラリなどの)外部ライブラリファイルへのパスを除き、マッピングコンポーネント内のすべてのパスに影響を与えます。

(C) 2015-2021 Altova GmbH

	チェックボックスは生成されたプログラムコード内でおよびMapForce Server Execution ファイル(mfx) 内で、およびFlowForce Server にデプロイされたマッピング内で ファイル パスが相対パス化または絶対パスかを定義します。多種の実行環境内でのパスも参照してください。以下も参照多種の実行環境内のパス
生成された XSLT／XQuery ファイルに相対的なパスを持つレファレンスライブラリ	マッピング言語がXSLT またはXQuery* の場合このチェックボックスを適用することができます。 XSLT またはXQuery ライブラリをマッピング参照する場合、およびマッピングからXSLT またはXQuery ファイルを生成する場合このオプションは通常役に立ちます。ライブラリ パスを生成されたXSLT またはXQuery コードのディレクトリに対して相対的にする場合、チェックボックスを選択します。 チェックボックスが選択されていない場合、ライブラリ パスは生成されたXSLT またはXQuery コード内で絶対パスになります。生成されたコード内のライブラリ パスも参照してください。
W indows パスがファイル パスの規則であることを確認	マッピング言語がXSLT 2.0、XSLT 3.0、またはXQuery* 2.0の場合このチェックボックスは適用可能です。 チェックボックスはW indows のパス規則が使用されていることを確認します。XSLT 2.0、XSLT 3.0、またはXQuery を生成する場合、現在処理されているファイル名は、ローカルファイルのためにフォーム`file://URI` 内のパスを返す `document-uri` 関数を使用して取得されます。 このチェックボックスがアクティブな場合、`file://URI` パス仕様は更なる処理のために自動的に完全なW indows ファイル パス(例、"C:¥..") に変換されます。
行末	このコンボ ボックスにより、出力ファイルの行末を指定することができます。「プラットフォーム デフォルト」によりターゲットのオペレーティング システムで使われているデフォルトの設定(W indows には(CR+LF)、macOS には(LF)、Linux には(LF)) が使用されます。行末は手動でも選択することができます。MapForce Server 実行可能ファイル(.mfx) にマッピングをコンパイルする、または、他のオペレーティングシステム上で作動するFlowForce Server にマッピングをデプロイする際に重要です。
XML スキーマ バージョン	マッピングファイルで使用されたXML スキーマ バージョンを定義することができます。バージョン1.0 または1.1に準拠するスキーマをロードするかを定義します。すべてのバージョン1.1 特有の機能は現在サポートされています。 *xs:schema vc:minVersion="1.1"* 宣言が存在する場合、バージョン1.1 が使用されます。それ以外の場合、バージョン1.0 が使用されます。

	 XSD ドキュメントが`vc:minVersion` 属性 または`1.0` または`1.1` 以外の`vc:minVersion` 属性の値を持たない場合、XSD 1.0 がデフォルトのモードです。 **メモ** **vc:minVersion** 属性と**xsd:version** 属性を混同しないでください。前者がXSD バージョン番号を持つ場合、後者はドキュメントのバージョン番号を持ちます。 既存のマッピング内のこの設定を変更することは、選択されたスキーマバージョンの全てのXML スキーマを再ロードすることを意味し、妥当性を変更する可能性があります。
Web サービスオペレーション設定	マッピングドキュメントがW eb サービス実装の一部である場合、W SDL 定義、サービス、エンドポイント、オペレーションフィールドが自動的に入力されます。

＊MapForce Professional または Enterprise Edition を必要とします。

(C) 2015-2021 Altova GmbH

3.2 コンポーネントとの作業

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

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

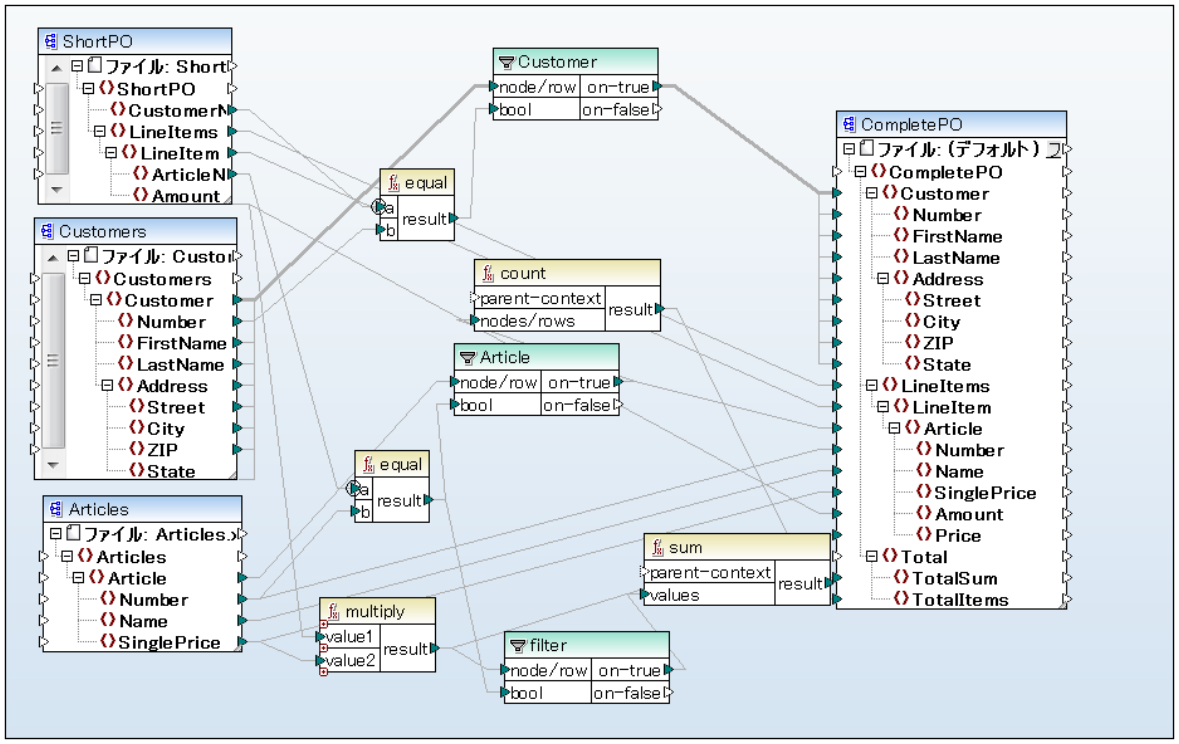
構造コンポーネントは、データの抽象的な構造とスキーマを表します。例: (メニューコマンド「**挿入 | XML スキーマ/ファイル**」を使用して) マッピングエリアにXML ファイルを追加すると、マッピングコンポーネントになります。構造コンポーネントとその特化についての詳細は、*データソースとターゲット*を参照してください。一部の例外を除いて、構造コンポーネントは、*アイテム*と*シーケンス*から構成されます。アイテムはマッピングユニットの最も低いレベルです (例: XML ファイル内の単一の属性、または単純型の要素)。シーケンスは*アイテム*のコレクションです。

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

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

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

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



CompletePO.mfd

このマッピングサンプルは以下のパスで見つけることができます: **<マイドキュメント>\Altova\MapForce2021\MapForceExamples\CompletePO.mfd**。

3.2.1 コンポーネント内の検索

特定のノード/アイテムをコンポーネント内で検索する方法:

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

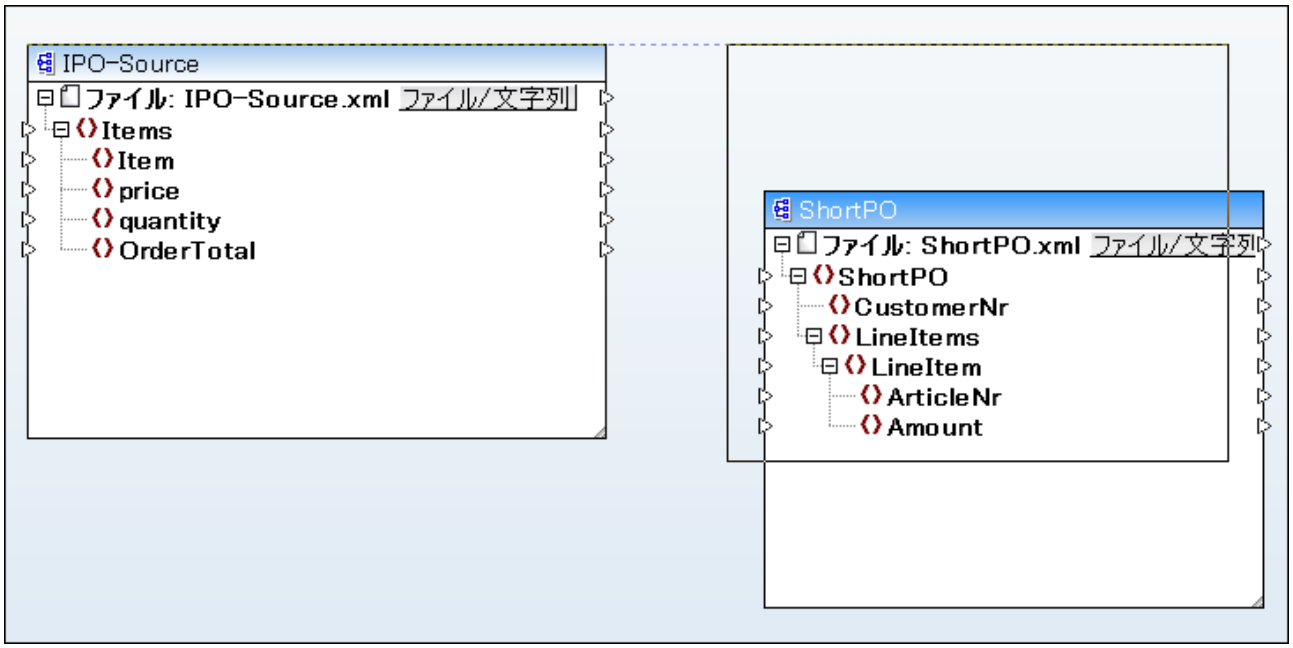
(C) 2015-2021 Altova GmbH

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

3.2.2　コンポーネントの整列

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

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



コンポーネント 自動配置 ガイドライン

このオプションを有効化または無効化する:

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

3.2.3 コンポーネント設定を変更する

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

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

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

- XML コンポーネント設定
- データベースコンポーネント設定
- CSV コンポーネント設定
- 固定長フィールド コンポーネント設定
- FlexText コンポーネント設定
- JSON コンポーネント設定
- Excel 2007+ コンポーネント設定
- XBRL コンポーネント設定
- EDI コンポーネント設定
- プロトコル バッファー バイナリコンポーネントの設定

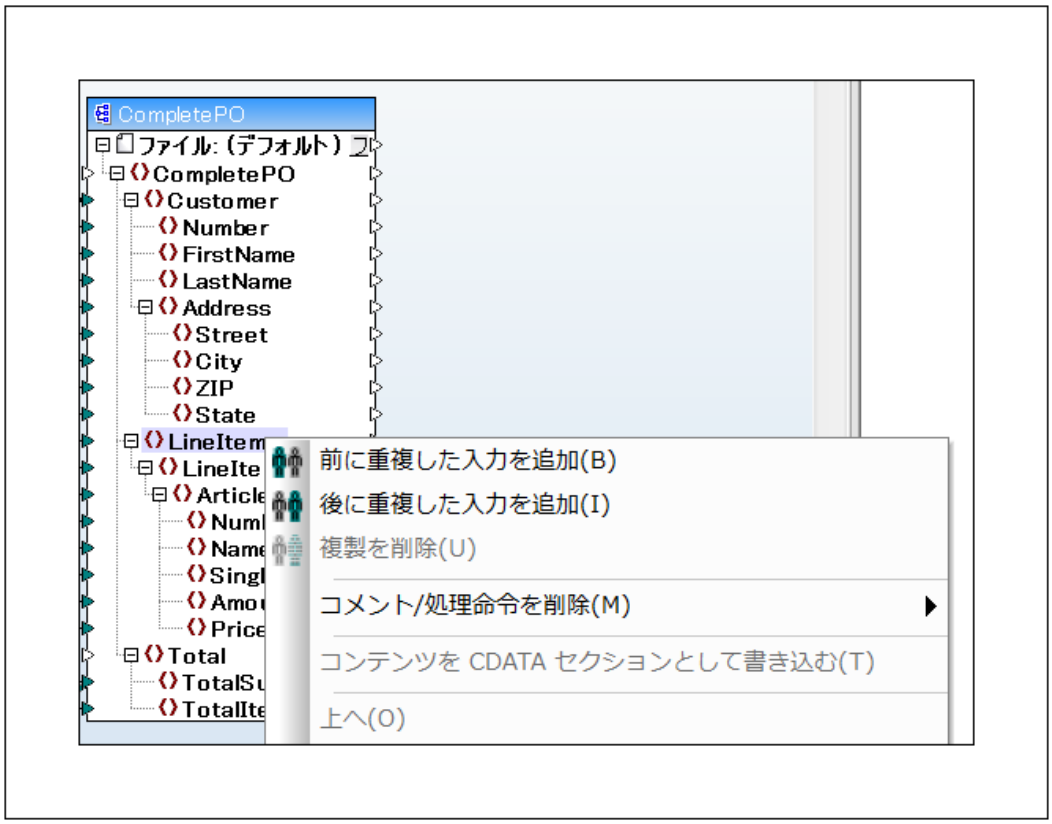
(C) 2015-2021 Altova GmbH

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

3.2.4　入力の複製

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

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



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

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

メモ　XML インスタンスを無効にするため、XML 属性の複製は許可されていません。スキーマ内の要素の `maxOccurs` 属性の値に関わらず、XML 要素の場合、入力の複製は許可されています。この振る舞いは、スキーマが後に変更することが可能なため、ま

た、ソースデータが任意のため、意図的です。 例えば、マッピングは、マッピング上で入力が複製されても、単一のXML 要素を生成することができます。

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

(C) 2015-2021 Altova GmbH

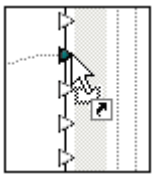
3.3 接続との作業

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

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

2つのアイテム間に接続線を引く:

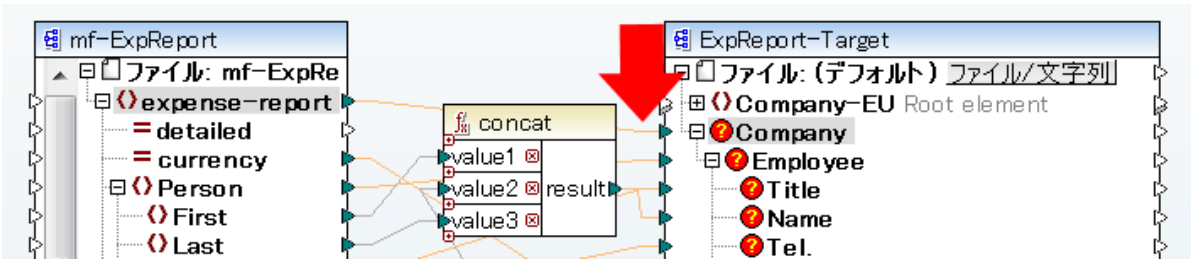
- ソースアイテムの出力コネクタをクリックし、目的アイテムまでドラッグします。ドロップアクションが許可されると、ヒントのリンクがカーソルの横に表示されます。



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

異なったアイテムに接続を移動する:

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



異なったアイテムへの接続をコピーする:

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

接続の対極にあるアイテムを表示する:

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

接続設定 を変更するには以下を行います:

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

接続設定を参照してください。

接続を削除するには、以下を行います:

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

3.3.1 必須の入力について

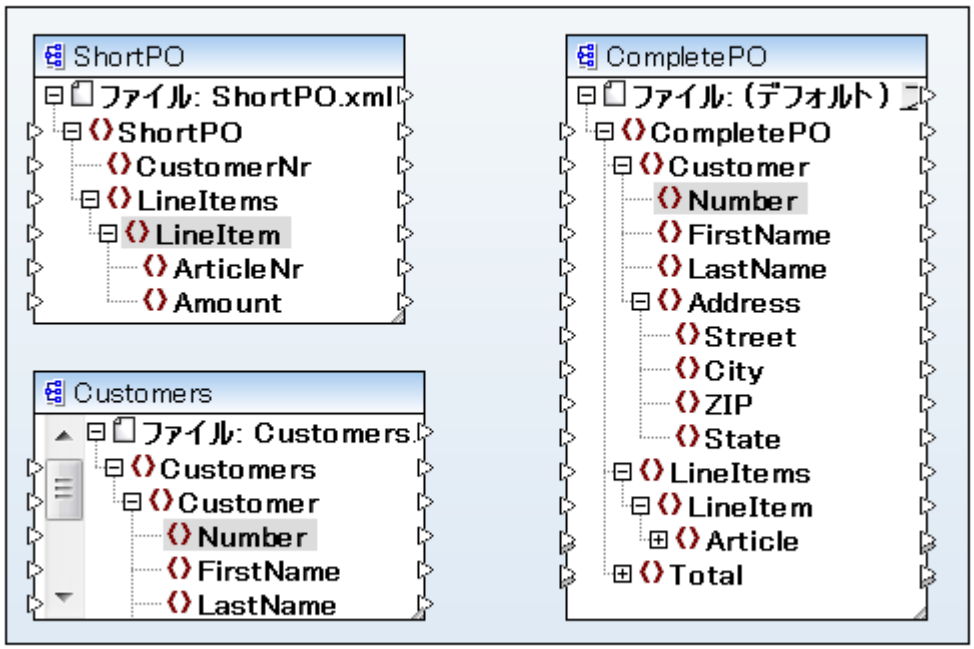
マッピング処理内での手助けをするために、MapForce は、ターゲットコンポーネント内の必須の入力をオレンジ色でハイライトします:

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

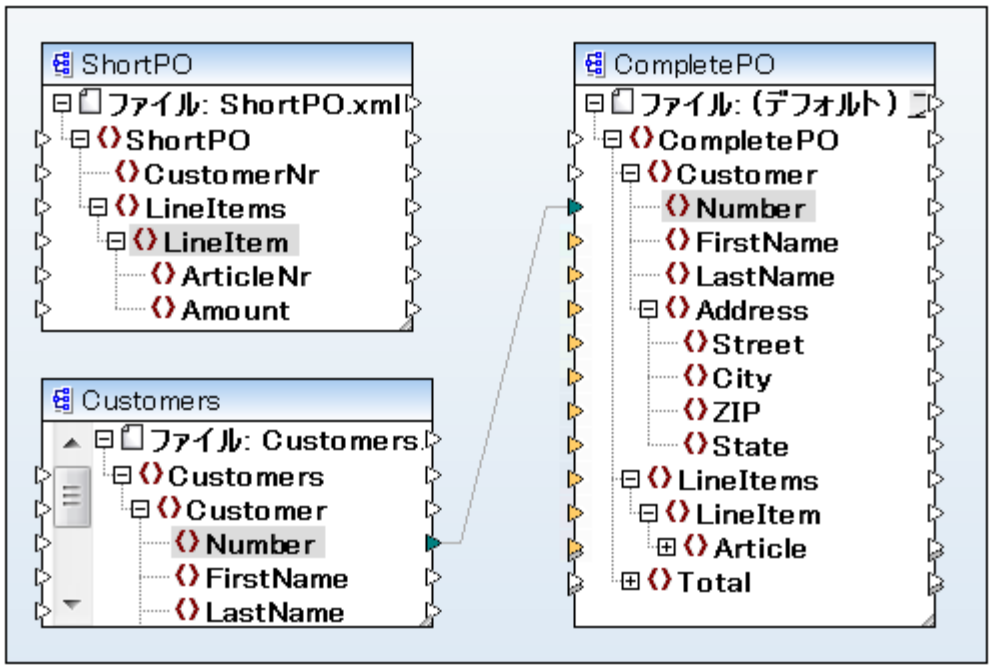
例

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

(C) 2015-2021 Altova GmbH



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



3.3.2 優先する接続線の表示の変更

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

選択コンポーネントの接続線の表示 により以下のように表示が切り替わります:

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

ソースからターゲットへの接続線の表示により以下の表示が切り替わります:

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

3.3.3 接続の注釈

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

接続に注釈をつける:

1. 接続を右クリックして、コンテキストメニューからを「**プロパティ**」選択します 、
2. 現在選択されている接続の名前 を「**詳細**」フィールドに入力します。これにより注釈設定グループ内の全てのオプションが有効化されます。

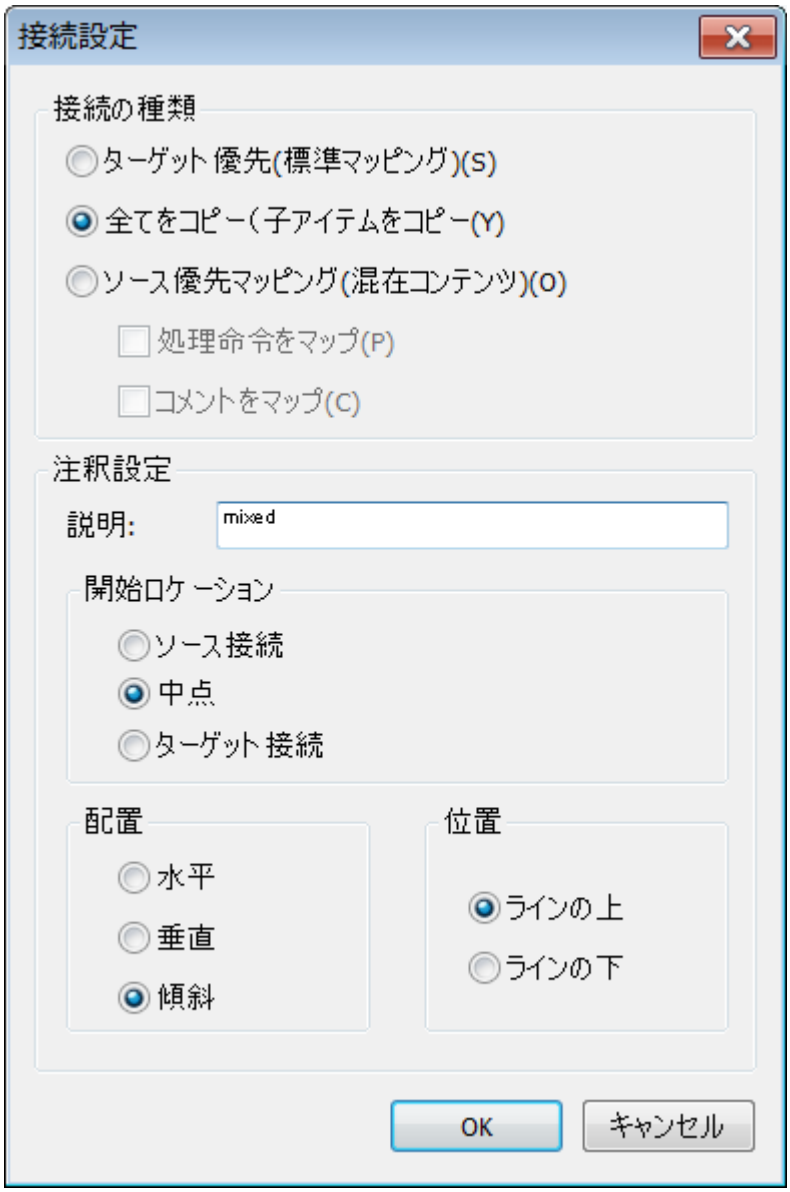
(C) 2015-2021 Altova GmbH

3. 残りのグループを使用して、ラベルの「開始の箇所」、「配置」、「位置」を定義します。
4. 注釈テキストを確認するには、表示オプションツールバー内の「注釈の表示」 アイコンを有効化します。

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

3.3.4 接続設定

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

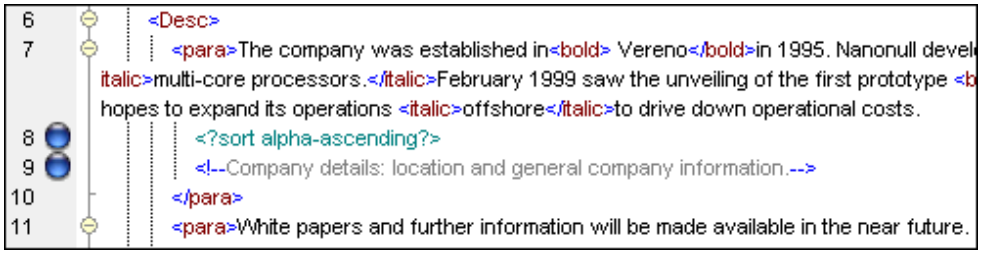


接続設定ダイアログボックス

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

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

(C) 2015-2021 Altova GmbH



メモ CDATA セクションはテキストとして扱われます。

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

3.3.5　一致する子要素の接続

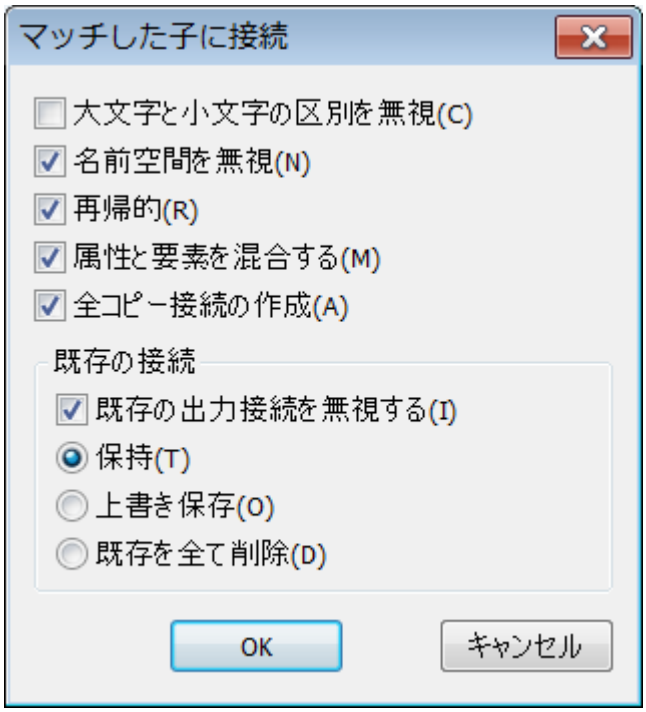
ソースとターゲットコンポーネントの両方内で同じ名前のアイテム間で複数の接続を作成することができます。「全てコピー」接続は、デフォルトで作成されます（ 全てコピー接続を参照）。

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

- 「一致する子要素の自動接続」（ ）ツールバーボタンをクリックします。
- 「接続」メニューから「一致する子要素の自動接続」をクリックします。

「マッチする子を接続する」のための設定を変更する方法:

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



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

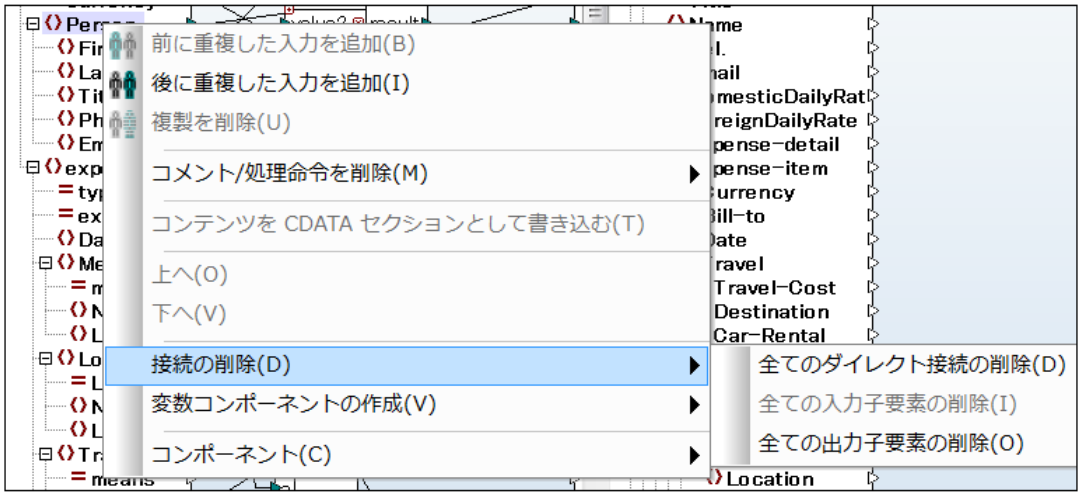
メモ 「子の自動接続の切り替え」（ ）ツールバーボタンがアクティブな場合、ここで定義する設定が2つのアイテムを接続する際に適用されます。

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

(C) 2015-2021 Altova GmbH

接続の削除

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



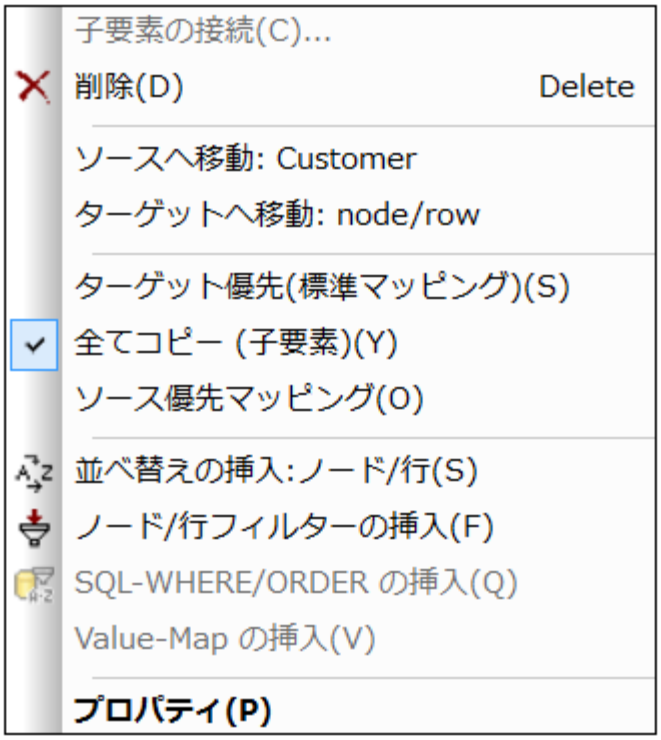
接続の削除:

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

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

3.3.6 コンテキストメニューの接続

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



マッチする子を接続	「マッチする子を接続する」ダイアログボックスを開きます。(一致する子要素の接続 を参照)。接続がマッチする子アイテムを持つことができる場合、このコマンドを有効化することができます。
削除	選択された接続を削除します。
ソースへ移動: <item name>	現在の接続のソースコネクタを選択します。
ターゲットに移動: <item name>	現在の接続のターゲットコネクタを選択します。
ターゲット優先 (標準)	接続の型をターゲット優先に変更します(ターゲット優先接続 を参照)。
全てコピー (子アイテムのコピー)	接続の種類を「全てコピー」に変更し、ソースとターゲットコンポーネント内の全ての等しいアイテムと自動的に接続します(全てコピー接続 を参照)。 ソースアイテムとターゲットアイテムの両方が子アイテムを持つ場合、このコマンドは有効化されます。
ソース優先 (複合型コンテンツ)	接続の種類を「ソース優先」に変更します(ソース優先マッピング を参照)。

(C) 2015-2021 Altova GmbH

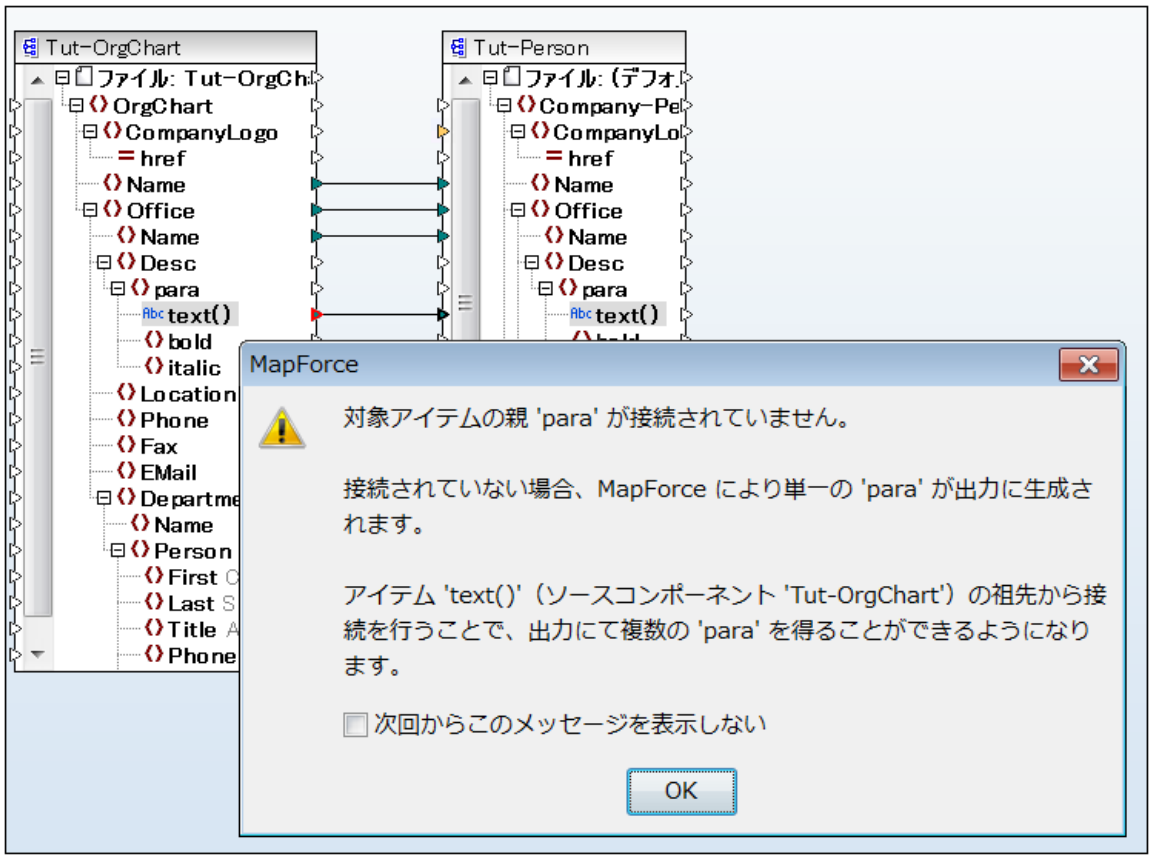
	ソースアイテムとターゲットアイテムの両方が子アイテムを持つ場合、このコマンドは有効化されます。
並べ替えの挿入: Nodes/Rows	ソースとターゲットアイテム間のソートコンポーネントを追加します。(データの置換えを参照)。
フィルターの挿入: Nodes/Rows	ソースとターゲットアイテム間のフィルターコンポーネントを追加します。(フィルターと条件を参照)。
SQL-W here 条件の挿入	ソースとターゲットアイテム間のSQL-W here コンポーネントを追加します。(SQL W HERE /ORDER コンポーネントを参照)。.
Value-Map の挿入	ソースとターゲットアイテ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

3.3.7 見つからない親接続の通知

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

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

<マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥Tutorial¥ フォルダーに収められている **Tut-OrgChart.mfd** マッピングを以下に示します。ソースコンポーネントにある text() アイテムがターゲットコンポーネントにある text() アイテムへ接続されると、メッセージボックスが表示され、親アイテムのpara が接続されておらず、出力では一度だけ生成される旨が通知されます。複数のpara アイテムをターゲットにて表示するには、ソースならびにターゲットアイテムのpara を接続する必要があります。



Tut-OrgChart.mfd (MapForce Basic Edition)

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

上に表示されるようなメッセージを非表示にするには 以下を行います:

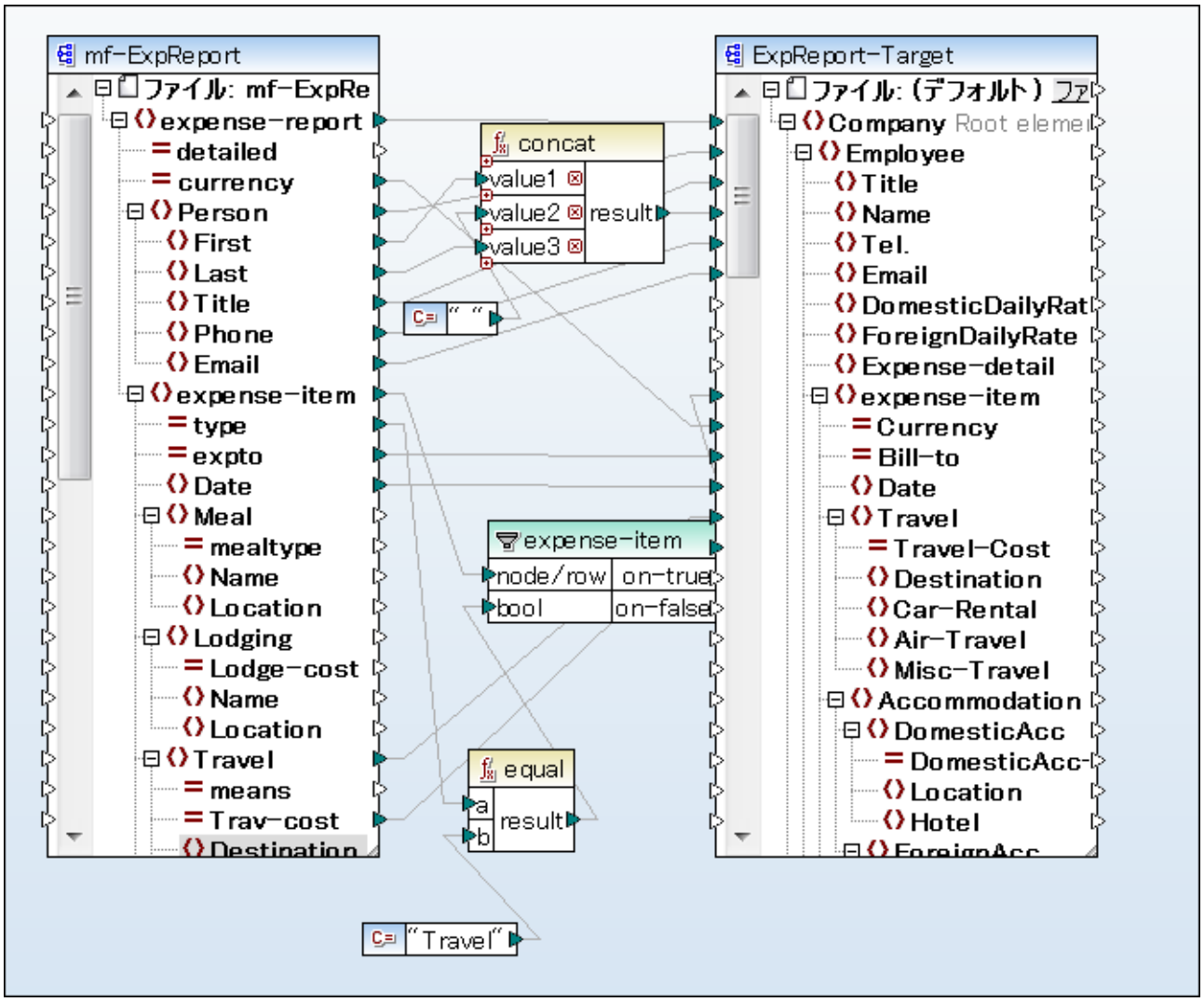
1. 「ツール」 メニューから「オプション」をクリックします。
2. 「メッセージ」 グループをクリックします。
3. 接続を作成した際、祖先アイテムへの接続を提案アイテム チェックボックスをクリックします。

3.3.8 接続と子接続の移動

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

<マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥Tutorial¥ フォルダーで使用することのできる **Tut-ExpReport.mfd** ファイルは、この例で修正され、別名で保存されます。

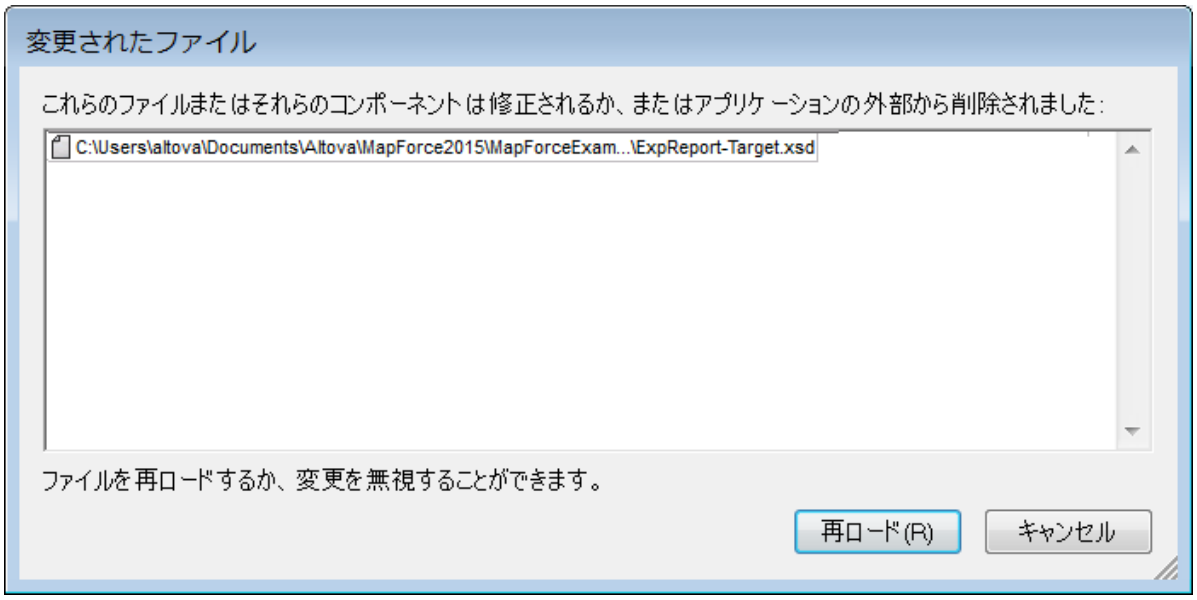
(C) 2015-2021 Altova GmbH



Tut-ExpReport.mfd (MapForce Basic Edition)

動作を理解するためには、以下を行います:

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



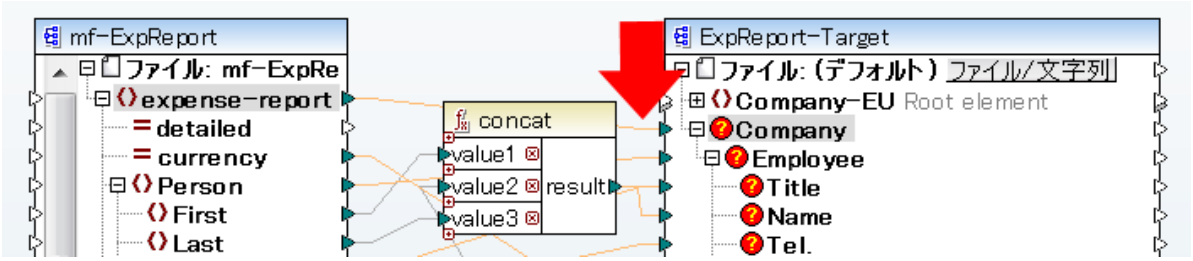
4. 更新されたスキーマを再ロードするために、「**再ロード**」ボタンをクリックします。ルート要素が削除されたため、コンポーネントは複数の見つからないノードを表示します。

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

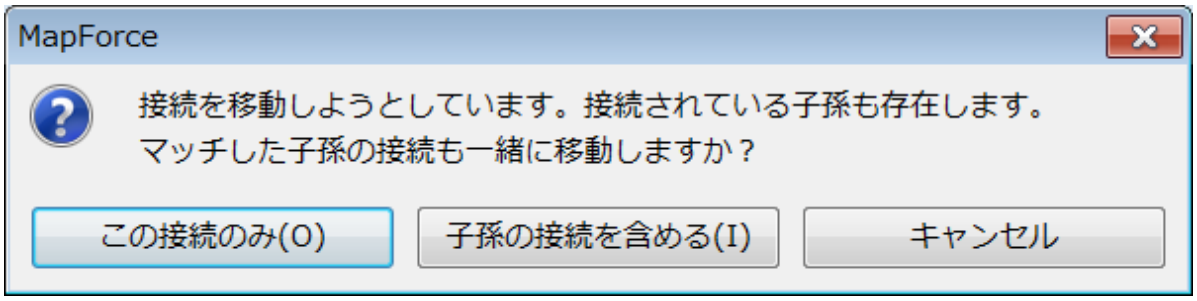
6. `Company-EU` を新しいルート要素として選択し、「**OK**」をクリックして確認します。`Company-EU` ルート要素がコンポーネントの上に表示されます。

(C) 2015-2021 Altova GmbH

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



通知ダイアログボックスが現れます。



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

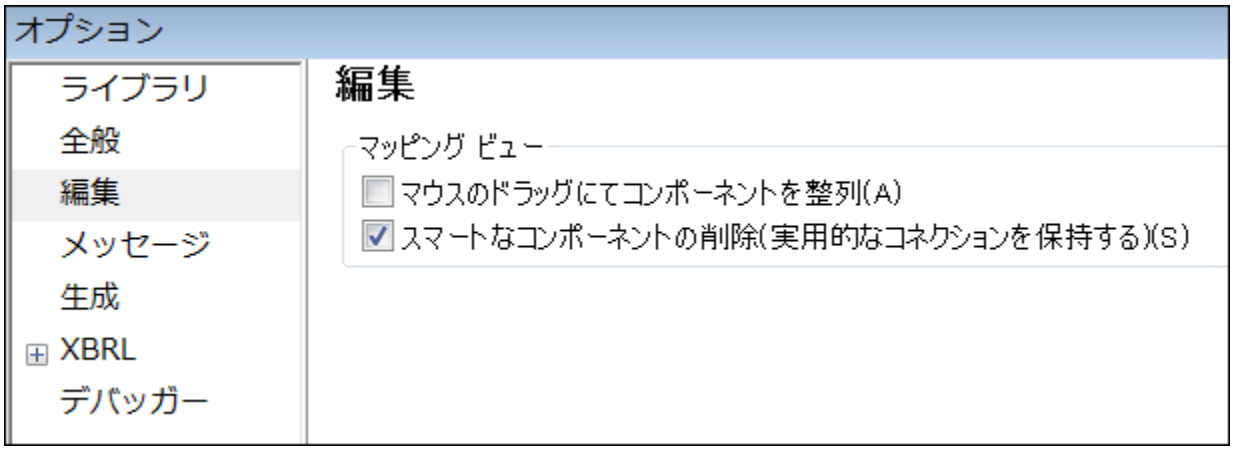
メモ マッピングするノードが、ソースノードと同じ名前を持ち、異なる名前空間にある場合、通知ダイアログボックスに、以下を表示する追加ボタンが表示されます:「子孫とマップ名前空間を含む」。このボタンをクリックすると、同じ名前空間の子接続が同じ子ノードのソース親ノードとして移動されます。このボタンをクリックすると、異なる名前空間ノードの下の同じ子ノードのソース親ノードと同じように同じ名前空間の子接続を移動します。

3.3.9 コンポーネント削除後の接続の保持

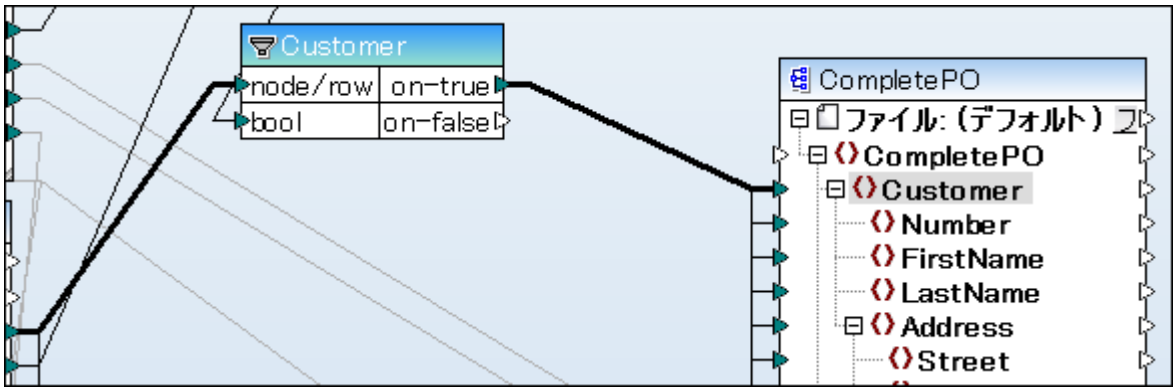
他のコンポーネントに複数の(子) 接続を持つ、コンポーネントを削除すると、何が起こるかを決定することができます。例: フィルターまたはソートコンポーネント。全ての子接続を保持し、個別に復元する必要がないためこの機能は役に立ちます。

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

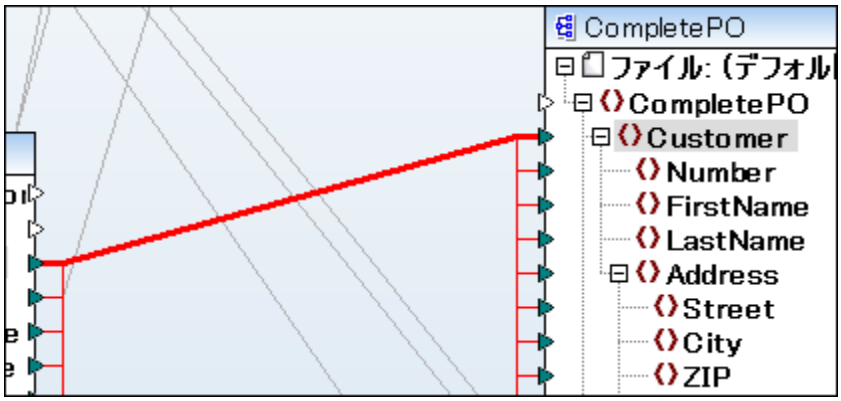
現在の設定を確認するために**ツール | オプション | 編集**」(タブ) を選択します。チェックボックスのデフォルト設定は、**非アクティブ**です。すなわち「スマートなコンポーネント削除(役に立つ接続を保持)」が無効化されます。



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



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



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

マッピングエリアをクリックすると、コネクタの選択が解除されます。

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

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

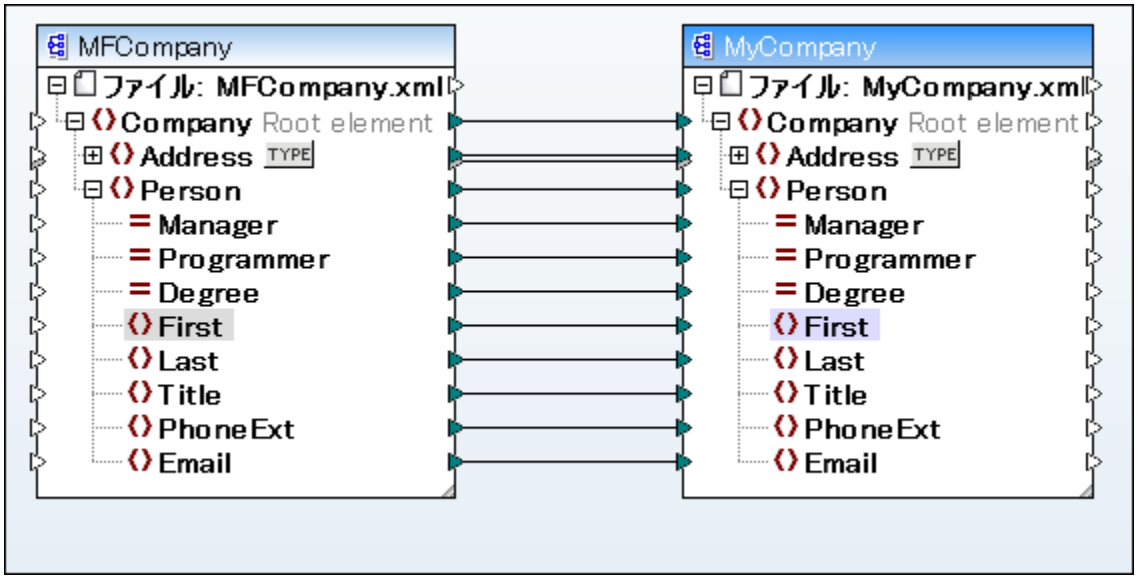
(C) 2015-2021 Altova GmbH

3.3.10 見つからないアイテムの扱い方

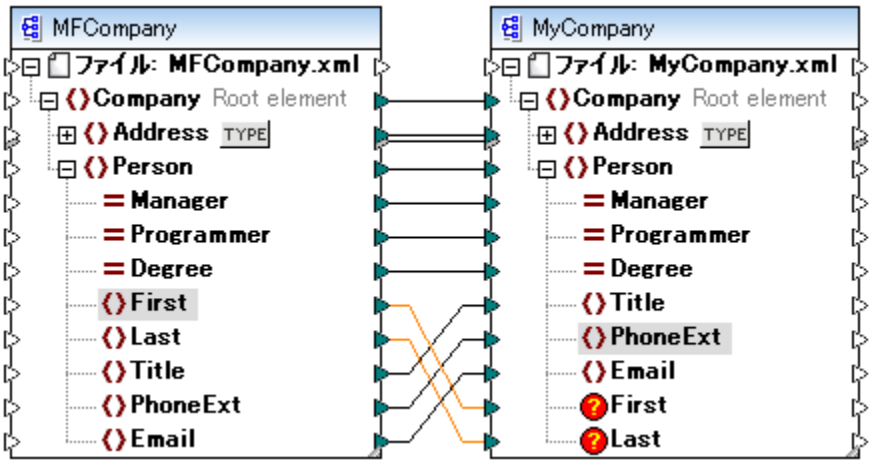
時が経つに従って、XML スキーマにて定義された要素や属性が追加/削除され、マッピング内にあるコンポーネントの構造が変更されるということがあります。MapForce ではプレースホルダーアイテムを使用することで、全ての接続を保持し、アイテムが削除された際にコンポーネント間で関連する接続データの保持を行います。

サンプル

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



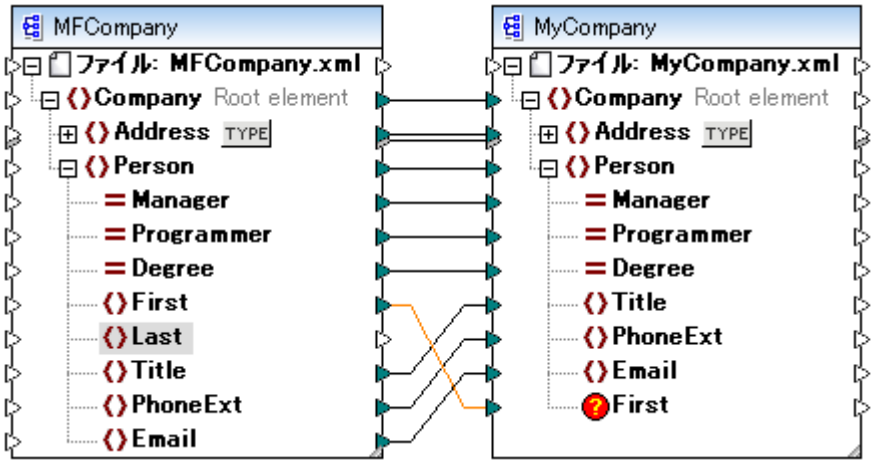
XMLSpy にて MyCompany.xsd を編集し、スキーマ内にある First ならびに Last アイテムを削除します。MapForce へ再度切り替えると、スキーマを再ロードするよう促されます。再ロードすることで、MapForce 内にあるコンポーネントが更新されます。



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

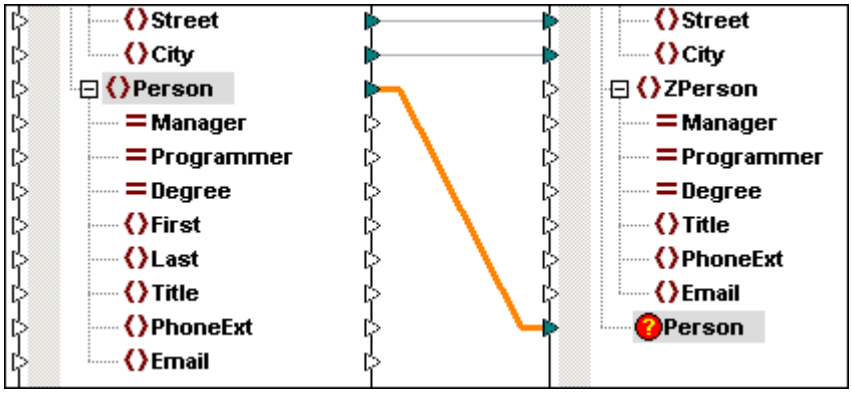
この状態でマッピング(または生成されたコード)のプレビューを行うこともできますが、警告がメッセージウィンドウに表示されます。プレビューならびにコード生成において、上のアイコンで示されるような、見つからないアイテムは無視されます。

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



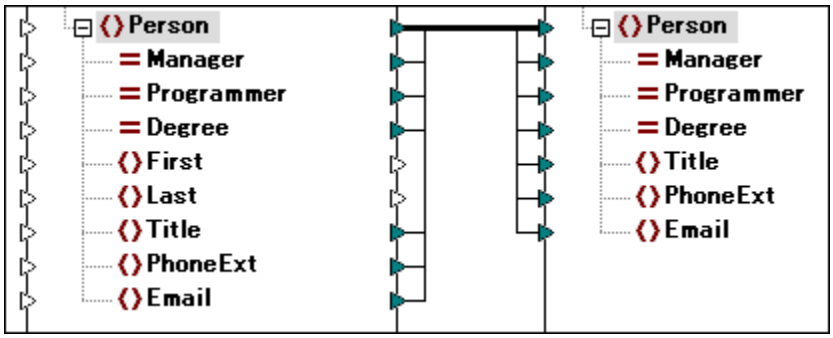
名前の変更されたアイテム

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



「全てコピー」接続と見つからないアイテム

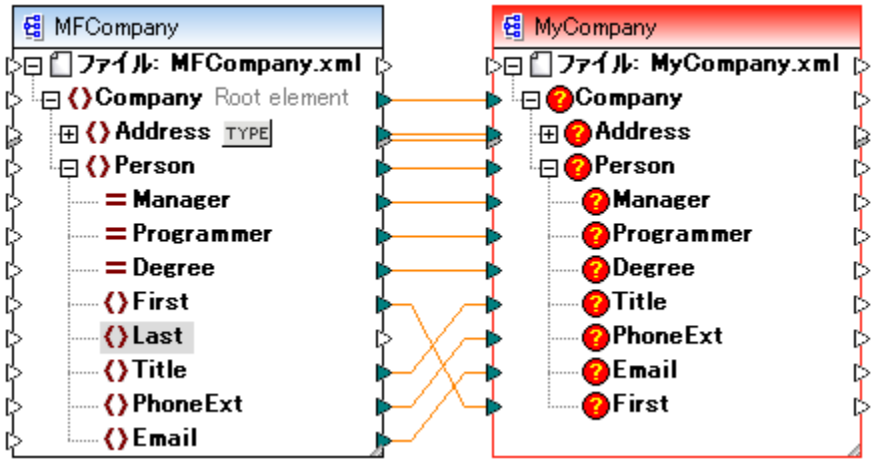
全てコピー接続は通常の接続と同じように扱われますが、見つからないアイテムへの接続は保持も表示もされません。



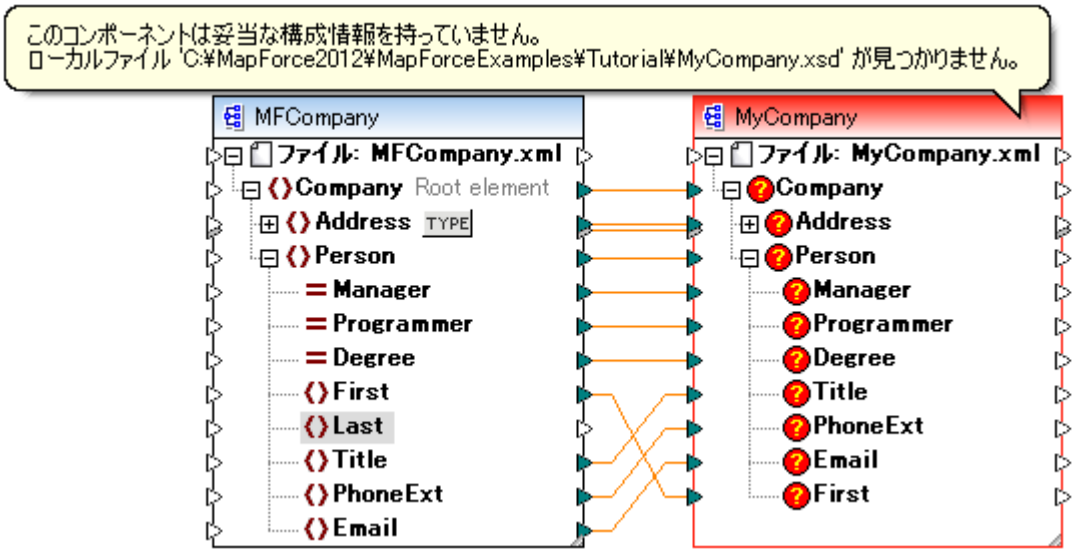
(C) 2015-2021 Altova GmbH

名前変更または削除されたコンポーネントソース

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

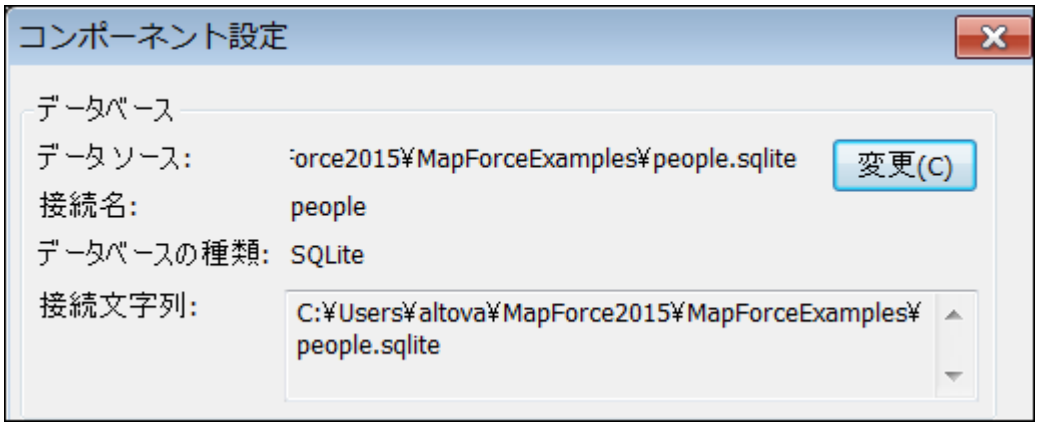


マウスカーソルをハイライトされているコンポーネントへマウスオーバーすることで、ポップアップが表示されます。



ハイライトされたコンポーネントのタイトルバーをダブルクリックすることで、コンポーネント設定ダイアログボックスが表示、異なるスキーマを選択することができます。コンポーネント設定の変更を参照してください。

コンポーネントがデータベースの場合、ダイアログボックス内の変更ボタンをクリックすることで、別のデータベースやテーブルを選択することができるようになります。同名のテーブルへの接続は保持されます。

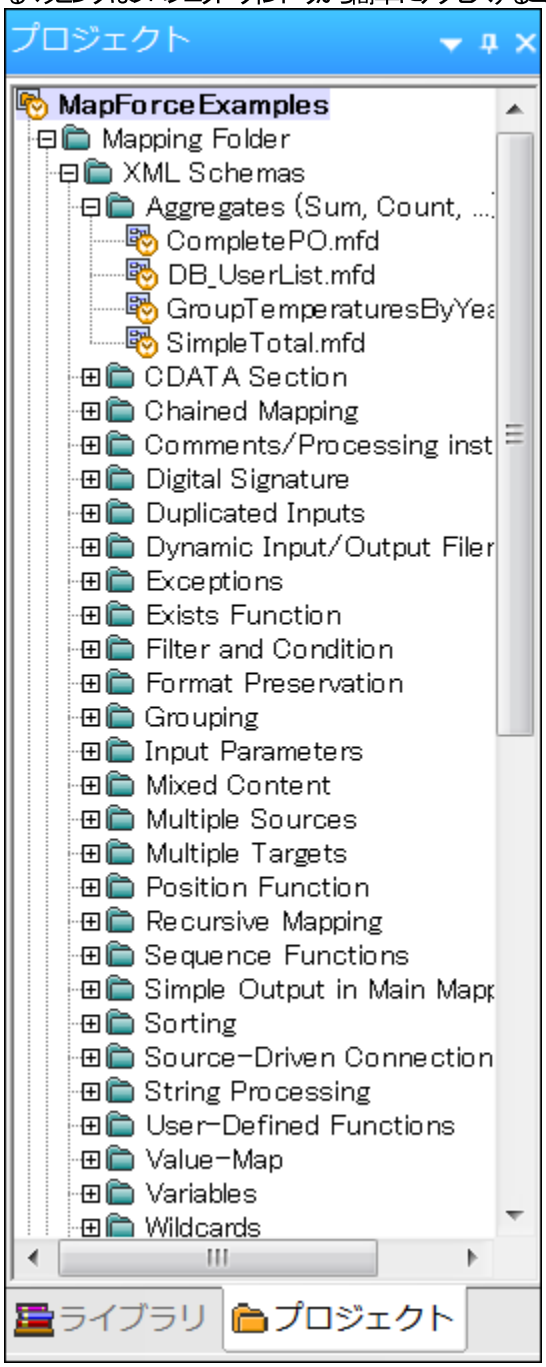


全ての妥当な(正確な)接続 同様の構造を持つスキーマまたはデータベースを選択した場合全ての妥当な(正確な)接続が保持されます。

(C) 2015-2021 Altova GmbH

3.4 マッピングプロジェクトとの作業

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



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

プロジェクトの主な利点は、特別なプロジェクトに含まれる全てのマッピングのための(ターゲット言語と出力ディレクトリなど) 共通のコード生成設定を定義が可能な点です。またプロジェクト内にフォルダーを作成することができ、プロジェクト内の個別のフォルダーのためのカスタムコード生

成（C++、C#、とJava ）の設定を指定することも可能です。 MapForceにより生成されたプログラムコードに関する詳細は、 コードジェネレータを参照。

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

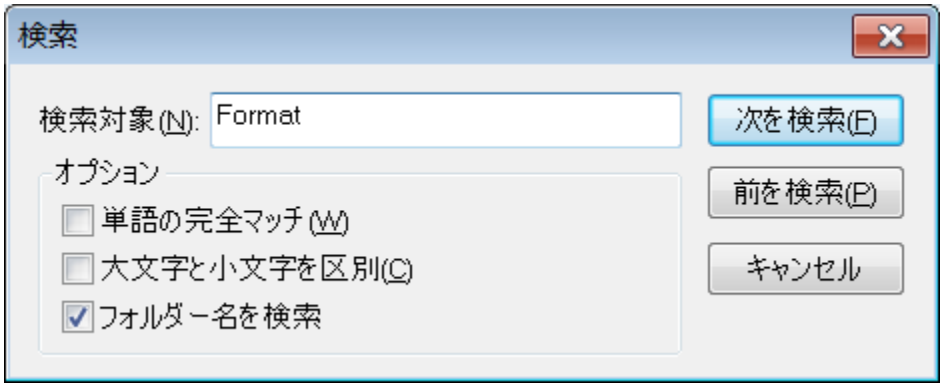
3.4.1 プロジェクトを開く、検索する、閉じる

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

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

プロジェクト内でファイルを検索する方法:

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



プロジェクトを閉じる方法:

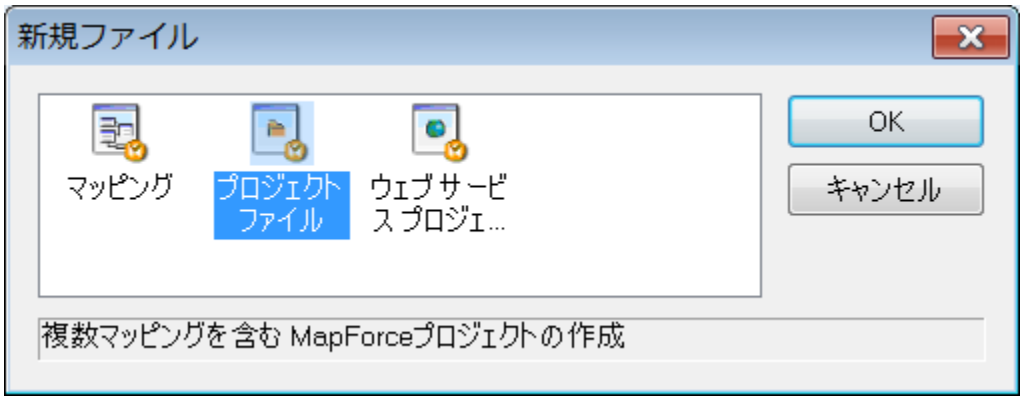
- 「プロジェクト」メニューから「プロジェクトを閉じる」をクリックします。

3.4.2 新規プロジェクトの作成

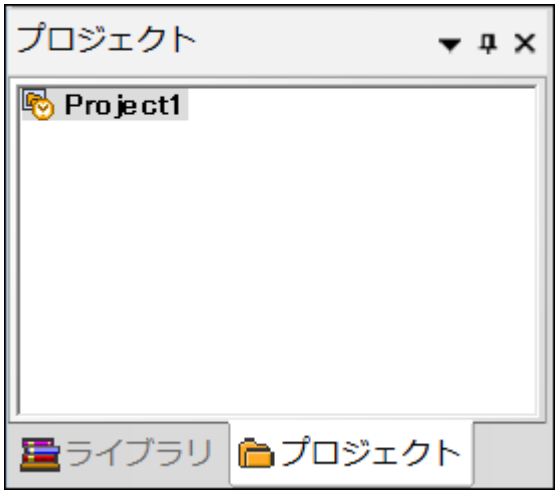
新規プロジェクトを作成する

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

(C) 2015-2021 Altova GmbH



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



プロジェクトにマッピングを追加します。

現在アクティブなマッピングをプロジェクトに追加するには、以下を行います:

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

プロジェクトに既存のマッピングファイルを追加するには、以下を行います:

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

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

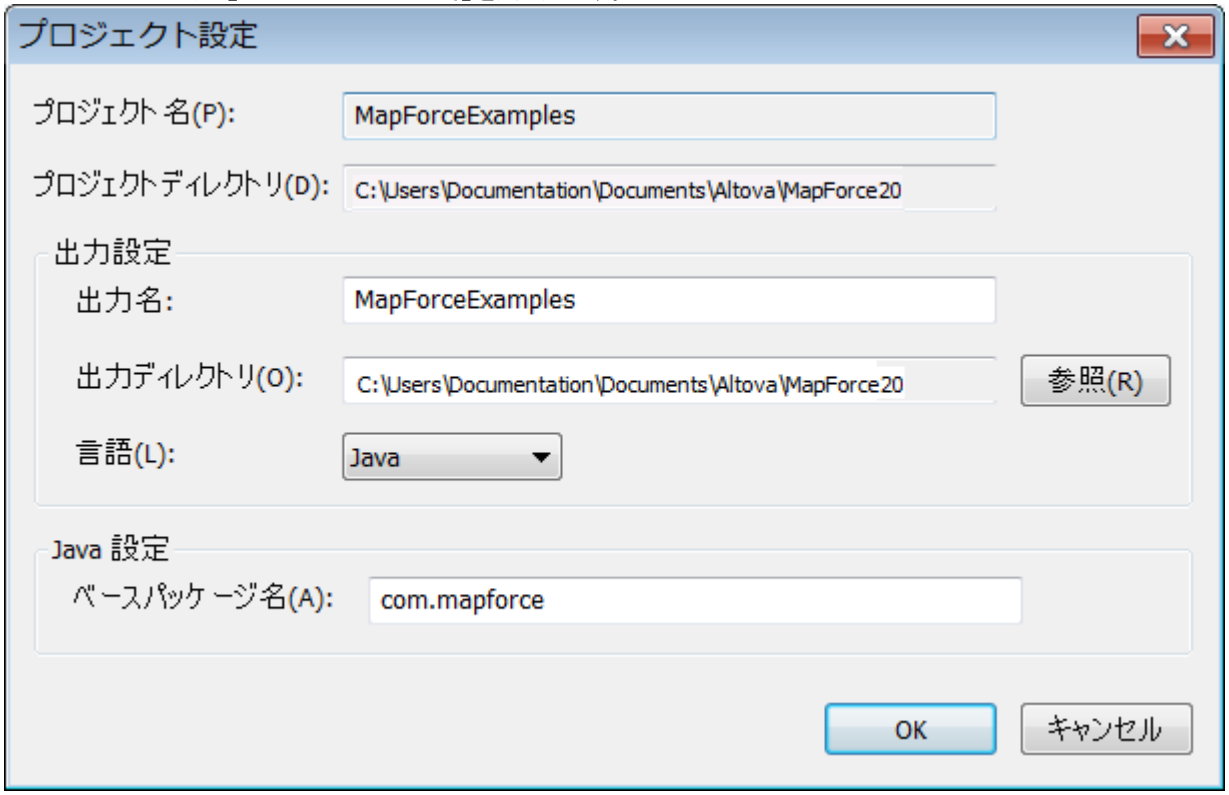
プロジェクトからファイルまたはフォルダーを削除するには、以下を行います:

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

3.4.3 コード生成設定の設定

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

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



プロジェクト設定ダイアログボックス

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

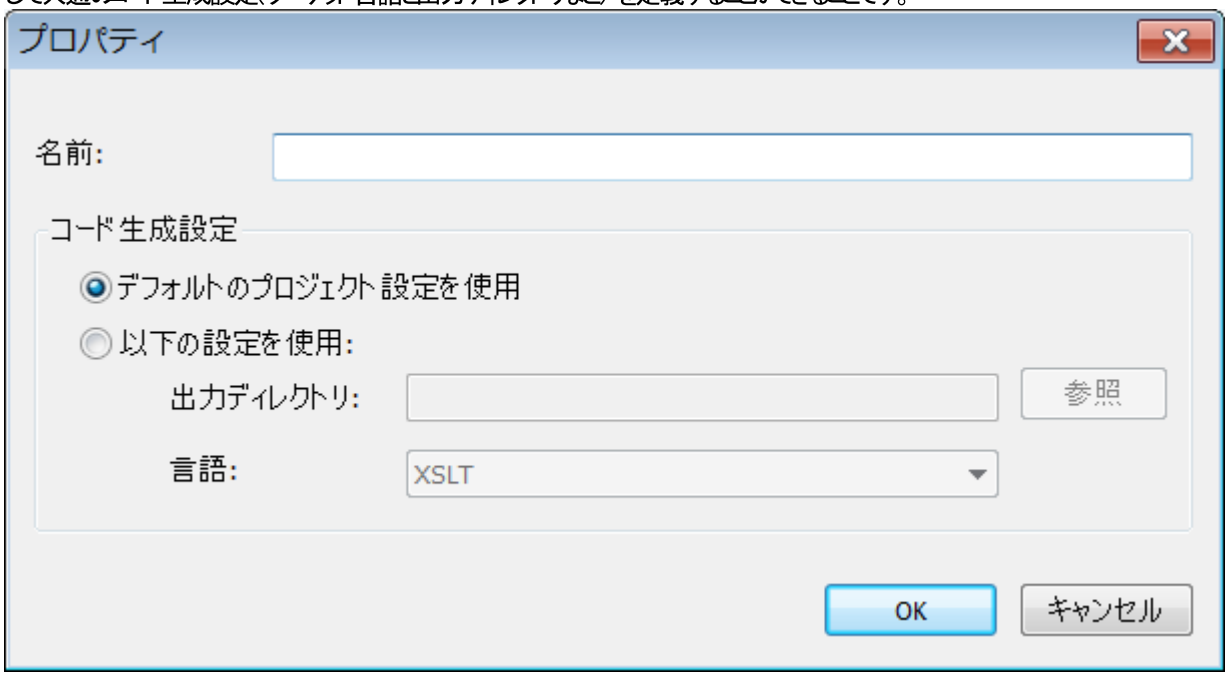
出力名	入力された値は生成されるプロジェクトまたはソリューションの名前、また生成されたコード内のオブジェクトの名前を決定します。
出力ディレクトリ	(このプロジェクト内の全てのマッピングから) 生成されたコードが保存されるWindows フォルダーを決定します。デフォルトでは、出力はプロジェクトディレクトリ内の出力ディレクトリに保存されます。

(C) 2015-2021 Altova GmbH

言語	このプロジェクト内のすべてのマッピングファイルのためのコード生成言語を定義します。
ベースパッケージ名	この設定は、Java を変換言語として選択する場合適用することができます。Java プロジェクト内で生成されたベースパッケージ名を定義します。

3.4.4 プロジェクトフォルダーの管理

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



フォルダープロパティダイアログボックス

MapForce プロジェクト内にフォルダを作成する

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

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

名前	フォルダーの名前

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

(C) 2015-2021 Altova GmbH

4 マッピングのデザイン

Altova Web サイト: データ統合ツール

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

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

...を実行するには	以下を参照してください...
Word 2007+, HTML、またはPDF などの読みやすい書式にマッピングにより生成された結果を変換します。	StyleVision を使用してマッピングの出力のスタイルを指定する方法
マッピングからドキュメンテーションを生成します。	マッピングドキュメンテーションの生成

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

(C) 2015-2021 Altova GmbH

4.1 相対と絶対パスの使用

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

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

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

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

4.1.1 コンポーネント上で相対パスを使用する

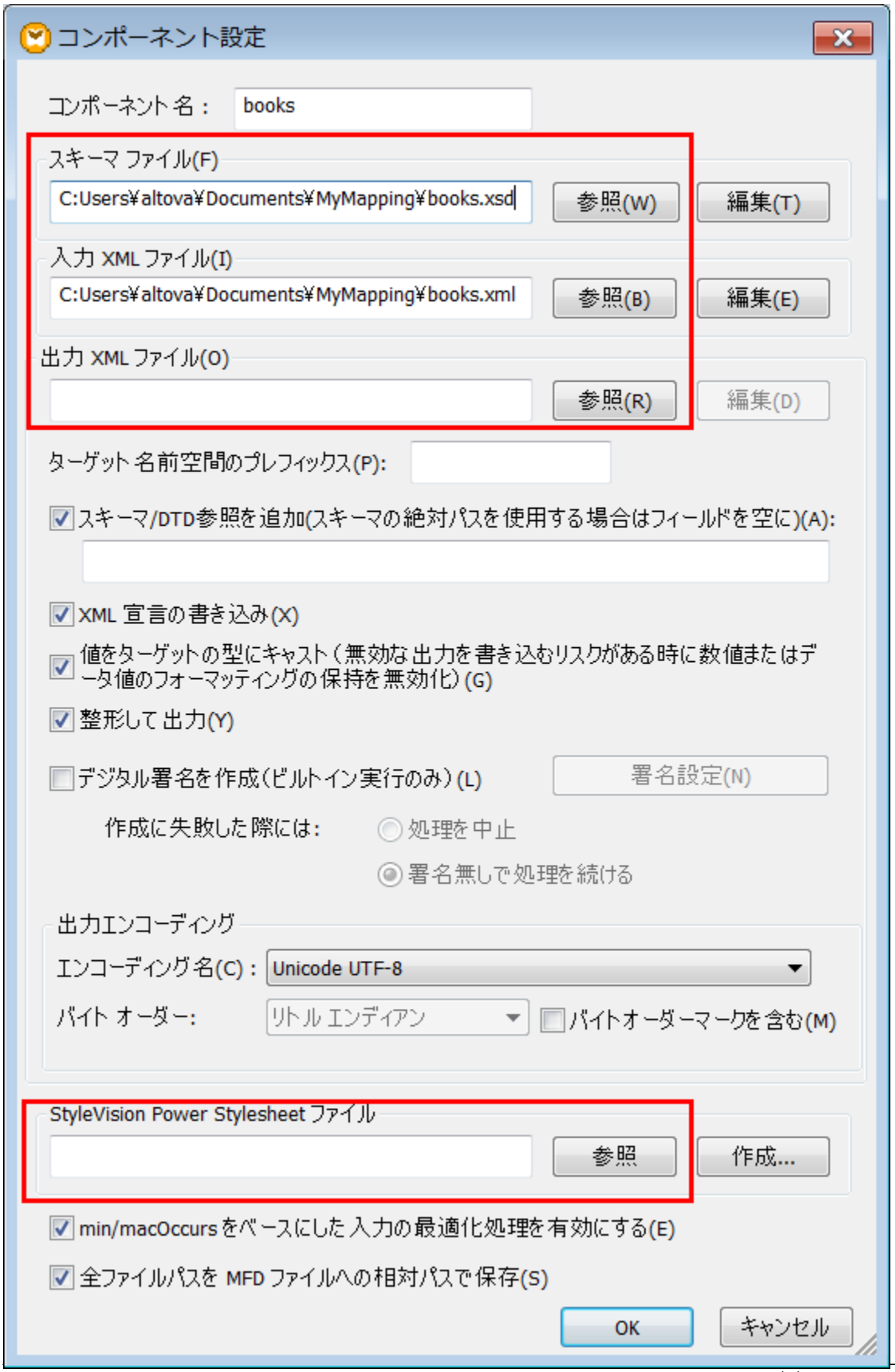
コンポーネント設定ダイアログボックス(XML コンポーネントのために下に表示されています) は、コンポーネントにより参照される可能性のある複数のファイルのための絶対または相対パスを指定するオプションを提供します:

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

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

> 相対ファイルパスを入力する前に、マッピングファイル(.mfd) を最初に保存してください。それ以外の場合、全ての相対パスはの W indows のパーソナルアプリケーションフォルダーに対して解決されます(ドキュメント¥Altova¥MapForce2021)これは、予期しない振る舞いの場合があります。

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



(C) 2015-2021 Altova GmbH

コンポーネント設定ダイアログボックス

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

- 複雑なユーザー定義関数の入力または出力パラメーターと複雑な型の変数により使用される構造ファイル
- 入力または出力 フラットファイル*
- XML フィールドをサポートするデータベースコンポーネントにより参照されるスキーマファイル*
- 入力または出力 XBRL、FlexText、EDI、Excel 2007+、JSON ファイル**

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

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

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

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

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

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

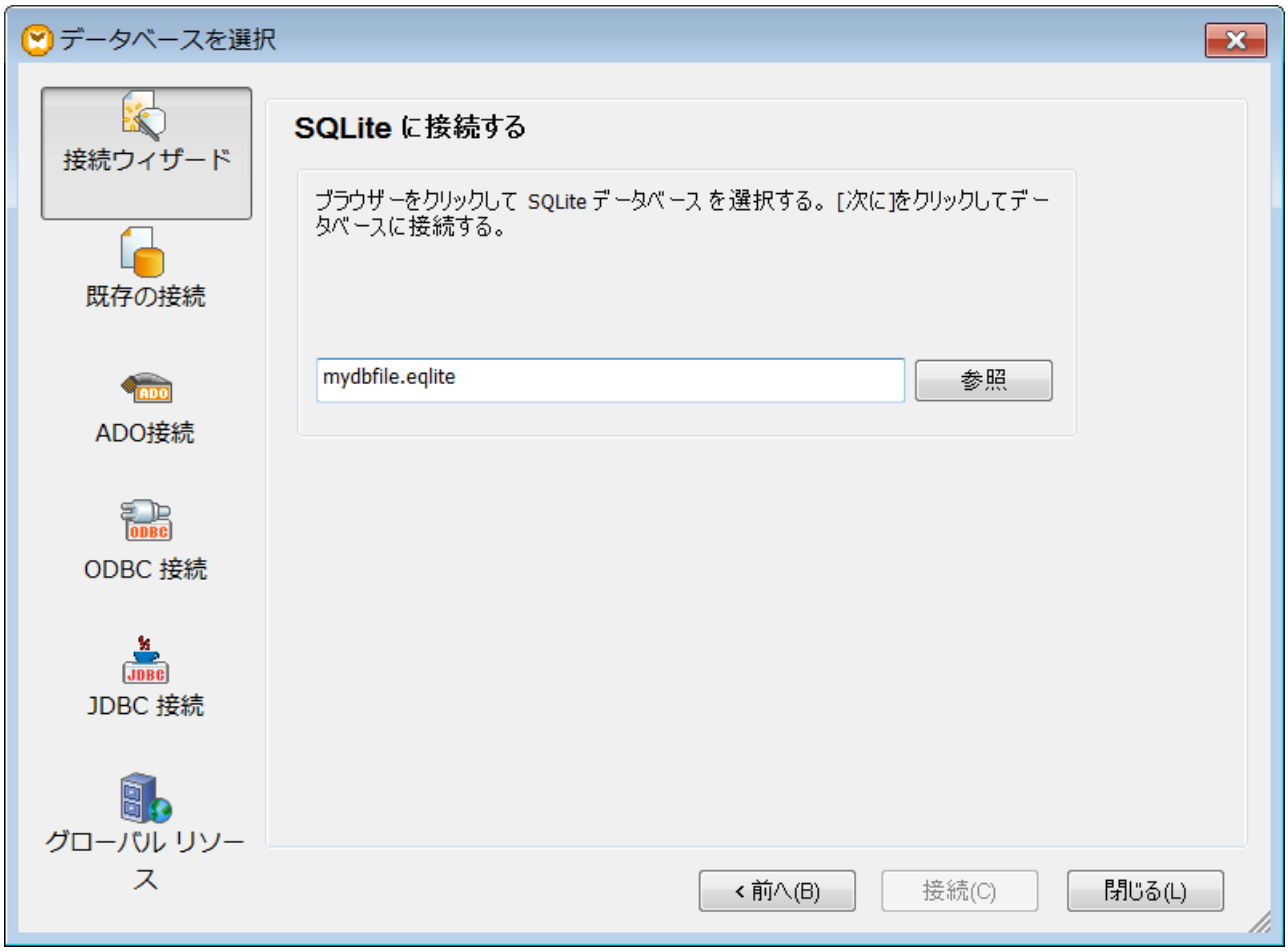
- オペレーティングシステム上のマッピングのロケーションが将来変更される可能性のある場合。
- （SVN などの）バージョンコントロールシステムを使用して）マッピングがソースコントロール化にあるディレクトリに保管されている場合。
- 異なったマシンに、または異なったオペレーティングシステムにマッピングを実行のためにデプロイする場合。

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

4.1.2 ファイルベースのデータベースへパスを設定する

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

相対ファイルパスを入力する前に、マッピングファイル（.mfd）を最初に保存してください。それ以外の場合、全ての相対パスはの Windows のパーソナルアプリケーションフォルダーに対して解決されます（ドキュメント¥Altova¥MapForce2021）これは、予期しない振る舞いの場合があります。



データベース接続ウィザード

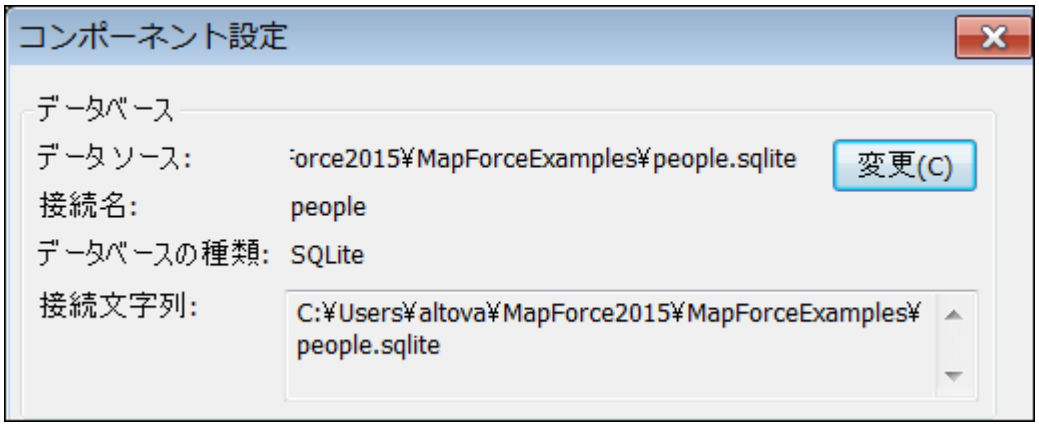
データベースがSQLite データベースの場合、以下の条件がtrue の場合、「**接続**」ボタンが有効化されます:

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

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

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

(C) 2015-2021 Altova GmbH



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

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

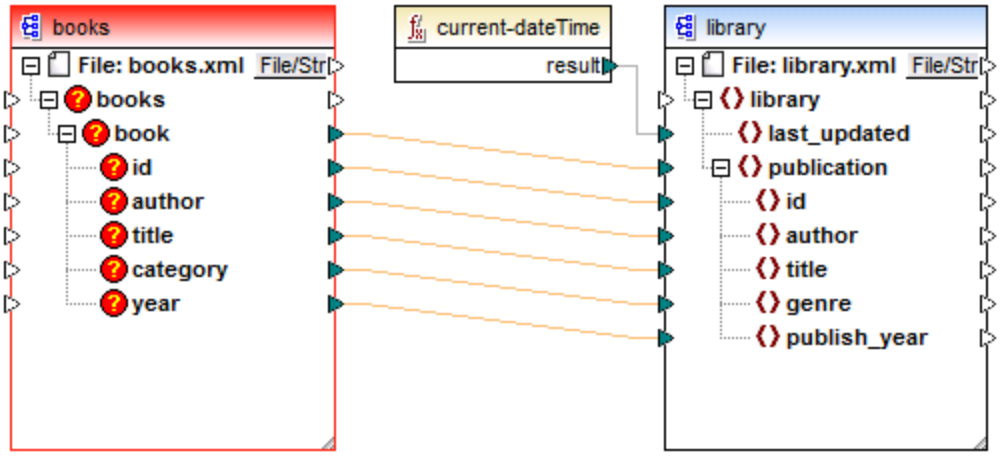
メモ プログラムコードを生成する際に、MapForce Server 実行ファイル(.mfx)をコンパイル、またはFlowForce Server にマッピングをデプロイする際に、マッピング設定から「**生成されるコード内でパスを絶対パスにする**」が選択されている場合、相対パスが絶対パスに変換されます(生成されたコード内のパスについてを参照)

4.1.3 破損したパスの参照を修正する

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

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

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



破損したパスの参照

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

4.1.4 多種の実行環境内のパス

コードをマッピングから生成し、MapForce Server 実行可能ファイル（.mfx）に対してマッピングをコンパイルする場合 生成されたファイルは、MapForce で実行することはできません。選択されたターゲット環境で実行できるようになります。（例: RaptorXML Server、MapForce Server、またはC# アプリケーション）。マッピングの実行に成功するには、全ての相対的なパスはマッピングが実行される個所で意味をなす必要があります。

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

ターゲット言語	ベースパス
XSLT、XSLT2、XSLT3	XSLT ファイルのパス
XQuery*	XQuery ファイルのパス
C++、C#、Java*	生成されたアプリケーションの作業ディレクトリ
BUILT-IN*（MapForce でマッピングをプレビューする場合）	マッピング（.mfd）ファイルのパス
BUILT-IN*（MapForce Server でマッピングを実行する場合）	現在の作業ディレクトリ
BUILT-IN*（ FlowForce Server の管理下で、MapForce Server でマッピングを実行する場合 ）	ジョブの作業ディレクトリまたはFlowForce Server の作業ディレクトリ

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

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

生成されたコード内で全てのパスを絶対パスに変換する方法:

1. マッピングの空のエリアを右クリックします。マッピングの設定 ダイアログボックスが表示されます。
2. 「生成されたコードでは絶対パスを使用する」チェックボックスを選択します。

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

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

チェックボックスが選択されていない場合、ファイルパスは、コンポーネント設定内で定義されたとおり保存されます。

生成されたコード内のライブラリパス

マッピングファイルにはオプションで多種のライブラリへのパス参照が含まれている場合があります。例えば、マッピングファイルは他のマッピングまたはカスタムXSLT、XQuery*、C#*、またはJava* ライブラリまたは.mff*（MapForce 関数）ファイルからの関数からのユーザー定義関数をインポートする場合があります。関数ライブラリの管理も参照してください。

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

(C) 2015-2021 Altova GmbH

オプション「生成されたコード内でパスを絶対パスにする」はマッピングコンポーネントのみに適用され、外部ライブラリへのパスに影響を与えません。代わりにXSLT とXQuery 以外のすべてのライブラリのためにライブラリパスは解決され、生成されたコード内で絶対パスに変換されます。例えば、マッピングファイルに.NET .dll または Java .class ファイルが含まれ、生成されたコードを他の環境（例えば他のコンピューターで）で実行する場合、参照されるライブラリはターゲット環境内の同じパスに存在する必要があります。

XSLT またはXquery ファイルをマッピングから生成する場合、*生成されたXSLT またはXquery ファイルに相対的な*ライブラリパスを以下のように作成することができます:

1. マッピングの空のエリアを右クリックします。マッピングの設定 ダイアログボックスが表示されます。
2. チェックボックス「XSLT / Xquery ファイルに相対的なパスを持つレファレンスライブラリ」を選択します。

上のチェックボックスを選択するとXSLT またはXquery ライブラリファイルが実際にそのパスに存在することを確認してください。

XSLT またはXquery ライブラリファイルへのパスを生成されたコード内で絶対パスにするにはこのチェックボックスをクリアします。

4.1.5 コピーと貼り付けと相対パス

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

4.2 マッピング接続の種類

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

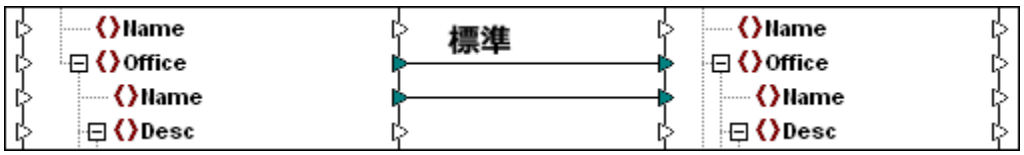
- ターゲット優先(標準)
- ソース優先(混合コンテンツ)
- 全てコピー(子アイテムのコピー)

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

4.2.1 ターゲット優先マッピング

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

標準マッピングは実線で表示されます。



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

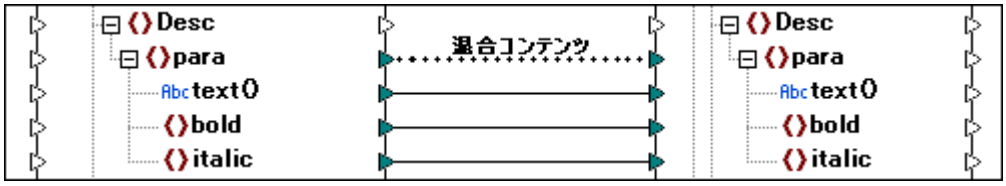
```
<p>This is our <i>best-selling</i> product.</p>
```

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

4.2.2 ソース優先マッピング

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

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



混合コンテンツのマッピングは点線により表示されます。

(C) 2015-2021 Altova GmbH

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

ソース優先 / 混合コンテンツマッピングでは以下がサポートされます:

マッピング

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

メモ CDATA セクションはテキストとして扱われます。

4.2.2.1 混合コンテンツのマッピング

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

ソース XML インスタンス

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

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

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

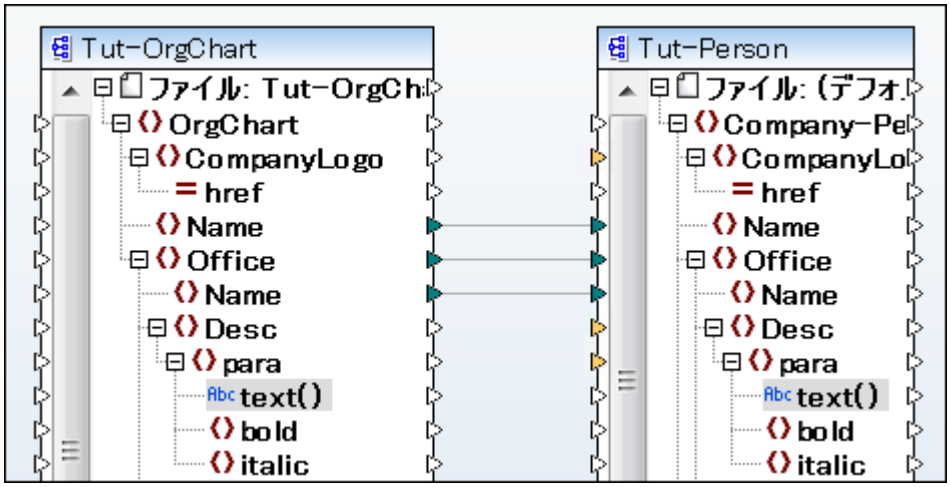
XML インスタンスファイルにあるテキストのシーケンスならびに bold/italic ノードは以下のような構成(順序)となっています:

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

    <bold>Nano-grid.</bold>The company ...
```

```
        <italic>offshore...</italic>to drive...
    </para>
```

初期のマッピング

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



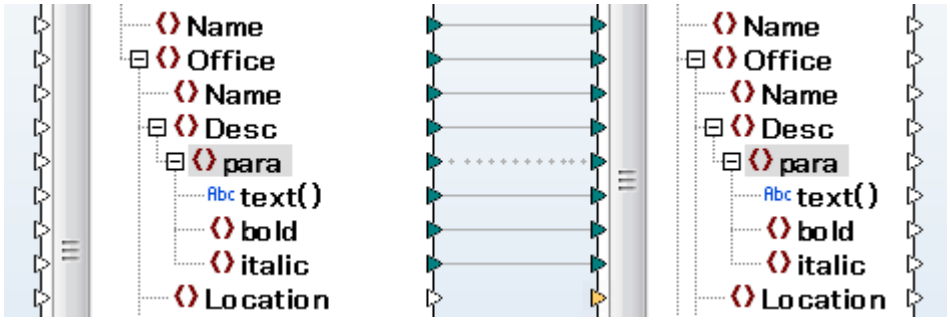
上のマッピングを出力

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

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

para 要素のマッピング

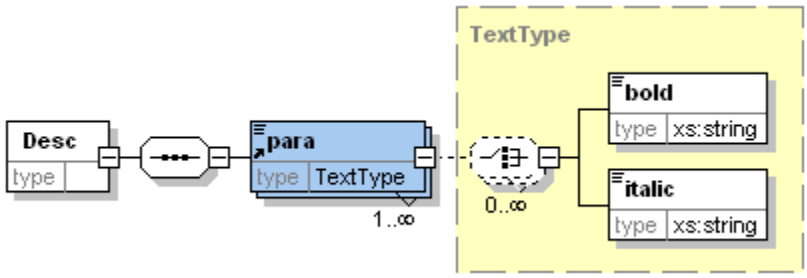
以下のスクリーンショットは混合コンテンツマッピングの例です。混合コンテンツのpara 要素にコネクタが接続されており、点線によりそれが混合コンテンツであることを表しています。**text()** ノードにはテキストデータが含まれており、ターゲットコンポーネントへテキストを表示するには、このノードをマッピングする必要があります。



(C) 2015-2021 Altova GmbH

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

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

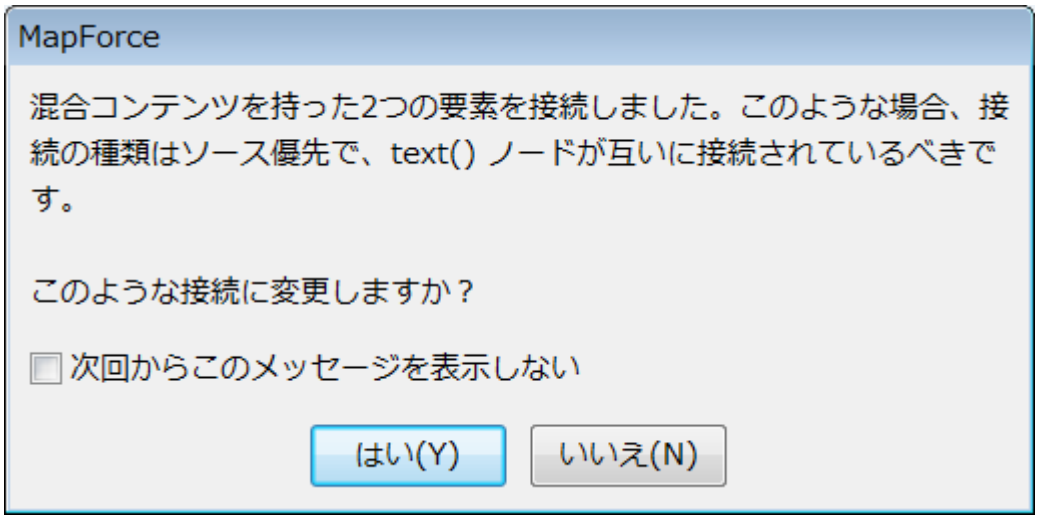


コンテンツモデルにある para 要素について、以下の事柄を理解できます:

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

アイテム間で混合コンテンツの接続を作成:

1. メニューオプションの「接続 | 子要素の自動接続」を選択して、有効にします(既に有効になっていない場合)。
2. ソーススキーマにて **para** アイテムから、ターゲットスキーマの **para** アイテムへの接続を行います。MapForce にコネクタをソース優先にするかを問うメッセージが表示されます。



3. 「はい」をクリックして混合コンテンツ接続を作成します。

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

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

4. 出力タブをクリックして、マッピングの結果を確認します。

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

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

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

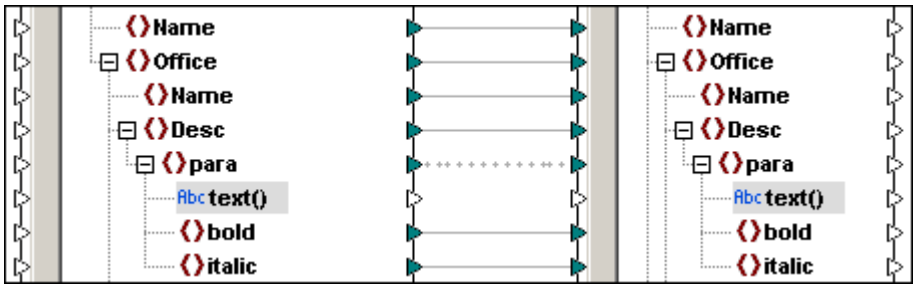
各オフィスに関する記述である混合コンテンツテキストが正しくマッピングされました。テキストだけでなく bold ならびに italic タグコンテンツが、XML ソースファイルにある通りにマッピングされました。

6. マッピングビューに切り替えます。

(C) 2015-2021 Altova GmbH

混合コンテンツからテキストノードを削除する:

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



2. 出力タブをクリックしてマッピングの結果を確認します。

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

結果:

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

処理情報とコメントのマッピング:

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

4.2.2.2 混合コンテンツのサンプル

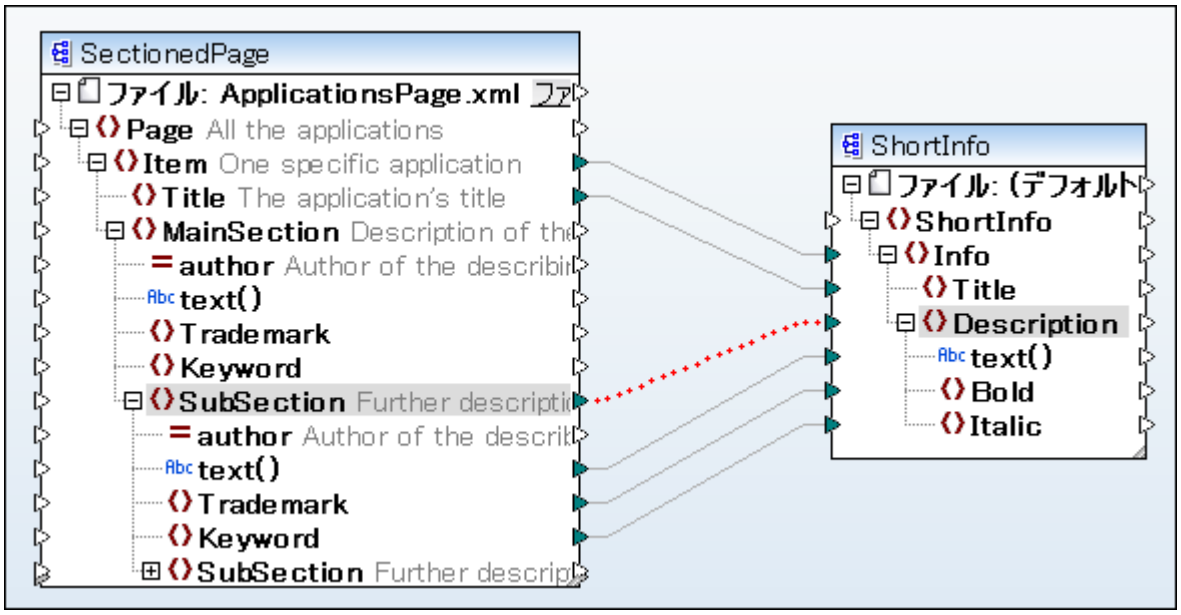
以下のサンプルでは **...\MapForceExamples** フォルダー以下にある **ShortApplicationInfo.mfd** を使用します。

この例で使用する XML ソースファイルの一部を示します:

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

以下にマッピングを示します。以下の点に注意してください:

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



マッピングの結果

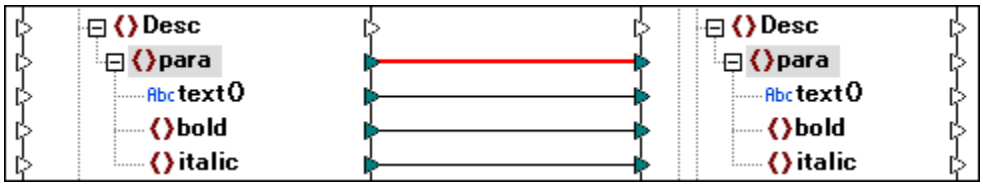
各説明文の混合コンテンツテキストが正しくマッピングされ、テキストだけではなく、bold と italic タグコンテンツが XML ソースファイルにある順序の通り、正しくマッピングされます。

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

4.2.2.3 混合コンテンツアイテムに対して標準接続を使用する

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

1. <マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥Tutorial¥ フォルダーからマッピング **Tut-OrgChart.mfd** を開きます。
2. ソース内の `para` ノード とターゲット内の `para` ノード 間に接続を作成します。MapForce が接続をソース優先と定義するかを問うメッセージが表示されます。「**いいえ**」をクリックします（これにより、MapForce による提案を破棄し、通常の接続を作成します）。



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

3. 「**出力**」タブをクリックして、マッピングの結果を表示します。

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

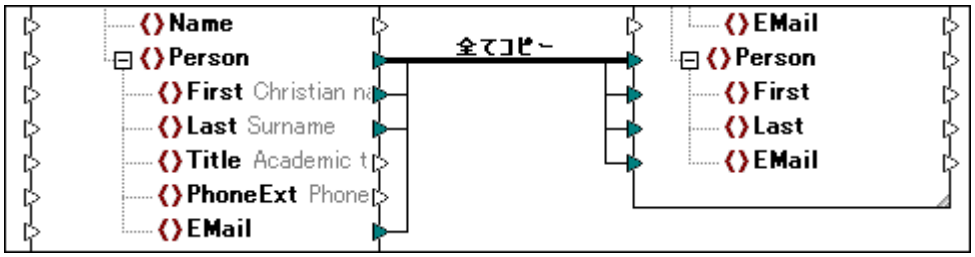
上に示されるように、混合コンテンツのアイテムに対して標準マッピングを適用すると、以下のような結果となります：

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

4.2.3 全てコピー接続

全てコピーという種類の接続方法では、作業領域にあるマッピングをより単純なものとして、ソースならびにターゲットコンポーネントにある全ての同一アイテムを自動的に接続します。つまり、ソースならびにターゲットコンポーネントにおける型に従うかたちで、ソースならびにターゲットアイテムの型が同一の場合か、またはターゲットの型がxs:anyType の場合、該当する全てのソースアイテムがターゲットコンポーネントへコピーされます。

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



全てコピー接続

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

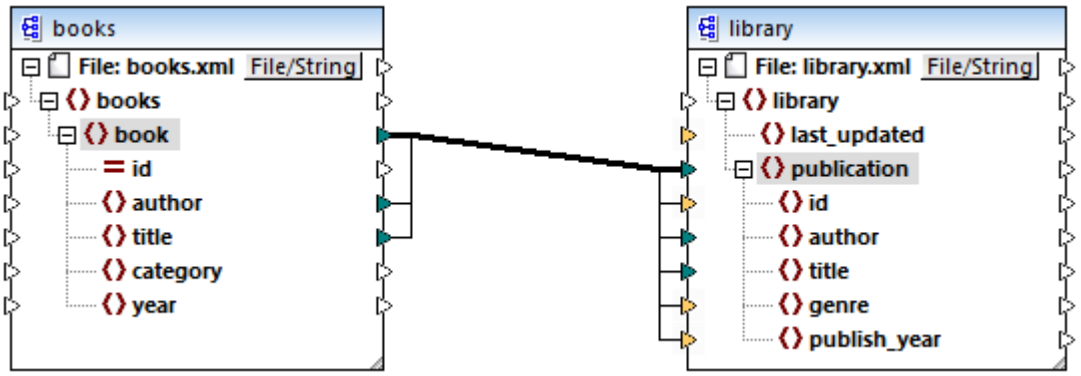
「全てコピー」接続を手動で作成するには、子アイテムを持つ同類の2 つのノード間の既存の接続を右クリックし、コンテキストメニューから全てコピー(子アイテムのコピー) を選択します。

次に注意してください:

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

「全てコピー」接続の例は次のステップで作成することができます:

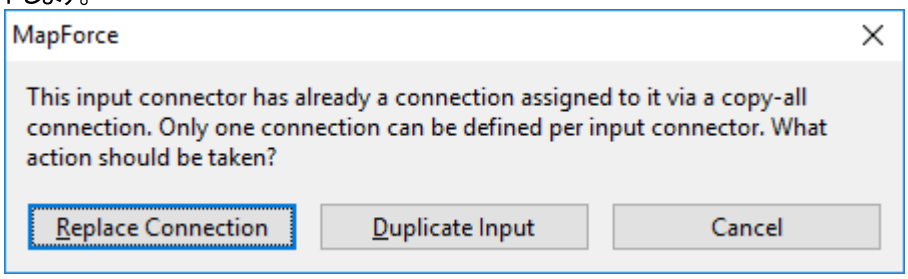
1. 新規マッピングを作成します。
2. 「挿入」メニューからXML スキーマ/ファイル をクリックし、フォルダー<マイドキュメント>\Altova\MapForce2021\MapForceExamples\Tutorial\ 内にあるbooks.xml ファイルを参照してください。
3. 「挿入」メニューからXML スキーマ/ファイル をクリックし、フォルダー<マイドキュメント>\Altova\MapForce2021\MapForceExamples\Tutorial\ 内にあるlibrary.xsd ファイルを参照してください。
4. 「books」コンポーネントのbook ノード と「library」コンポーネントのpublication ノード間にマッピング接続を描きます 。
5. 新規の接続を右クリックして、コンテキストメニューから 全てコピー(子アイテムのコピー) を選択します。



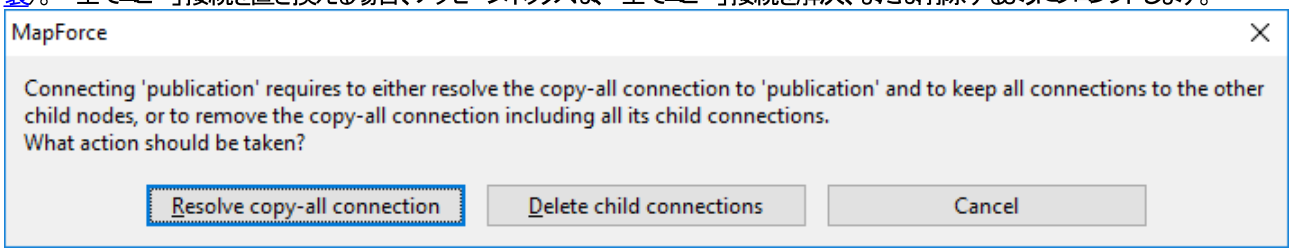
ソースとターゲット 構造間に差異が存在する場合、「全てコピー」接続が列挙します。マッピングのランタイムでは、（要素と属性などの）ソースアイテムは、ターゲット型内に存在するものだけがコピーされます。これは再帰的に繰り返されます。

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

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



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

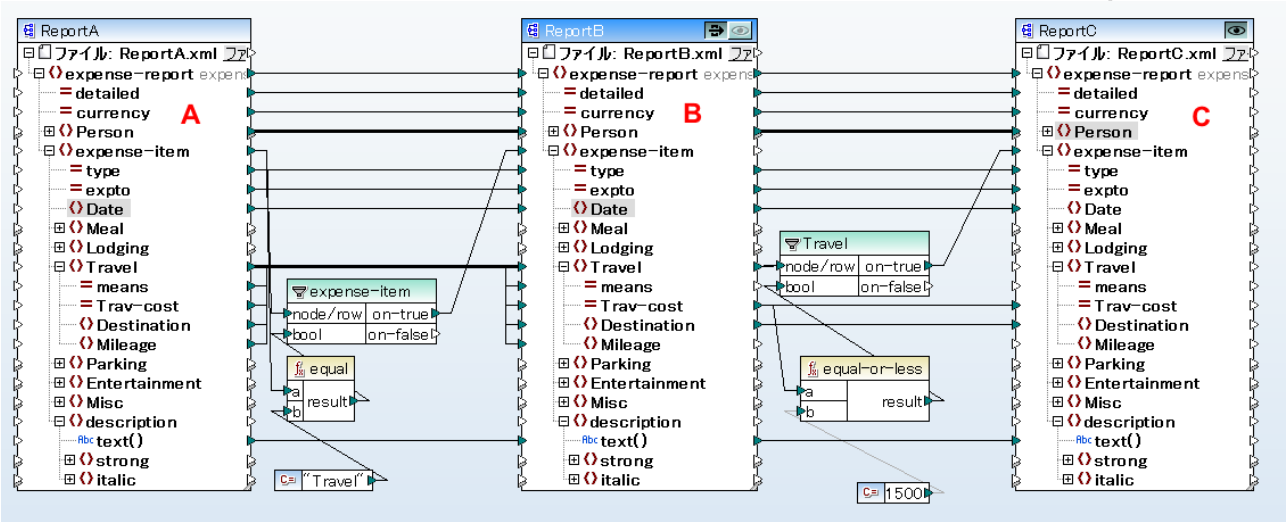


「全てコピー」接続を対応する子アイテムの標準のターゲット優先 接続と置換えるには、**全てコピー 接続を解決する** をクリックします。「全てコピー」接続を削除するには、**「子接続の削除」**をクリックします。

4.3 チェーンマッピング

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

例えば、下に示されるマッピングは、(XML 書式の) 2 つの段階で処理される経費報告書を表示しています。A から B へのマッピングの部分は、「Travel」とマークされる経費のみをフィルターします。B から C へのマッピングの部分は、1500 以下の「Travel」経費のみをフィルターします。コンポーネント B は、入力と出力接続の両方が存在するため、「中間」コンポーネントです。このマッピングは、次のパスで見つけることができます: <マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥Tutorial¥**ChainedReports.mfd**。



ChainedReports.mfd

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

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

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

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

プレビューボタン

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

(C) 2015-2021 Altova GmbH

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

パススルーボタン

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

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

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

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

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

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

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

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

次のセクションである 例: パススルーが有効な場合 と 例: パススルーが無効な場合 は、パススルーボタンが有効化、または無効化されている場合に、どのようにソースデータが転送されるかについて説明しています。

4.3.1　例: パススルーが有効な場合

このサンプルで使用されているマッピング (**ChainedReports.mfd**) は、**<マイドキュメント>\Altova\MapForce2021\MapForceExamples\Tutorial** フォルダー内にあります。このマッピングは、旅費の経費を含む **ReportA.xml** という名前の XML ファイルを処理します。簡素化のために、名前空間宣言と `expense-item` 要素の一部は省略されています:

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

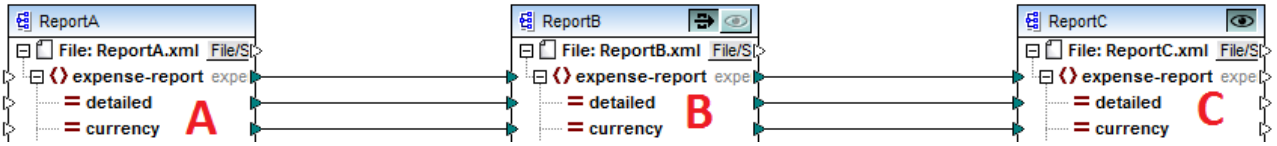
```
      <Phone>123-456-78</Phone>
      <Email>f.landis@nanonull.com</Email>
   </Person>
   <expense-item type="Travel" expto="Development">
      <Date>2003-01-02</Date>
      <Travel Trav-cost="337.88">
         <Destination/>
      </Travel>
      <description>Biz jet</description>
   </expense-item>
   <expense-item type="Lodging" expto="Sales">
      <Date>2003-01-01</Date>
      <Lodging Lodge-cost="121.2">
         <Location/>
      </Lodging>
      <description>Motel mania</description>
   </expense-item>
   <expense-item type="Travel" expto="Marketing">
      <Date>2003-02-02</Date>
      <Travel Trav-cost="2000">
         <Destination/>
      </Travel>
      <description>Hong Kong</description>
   </expense-item>
</expense-report>
```

ReportA.xml

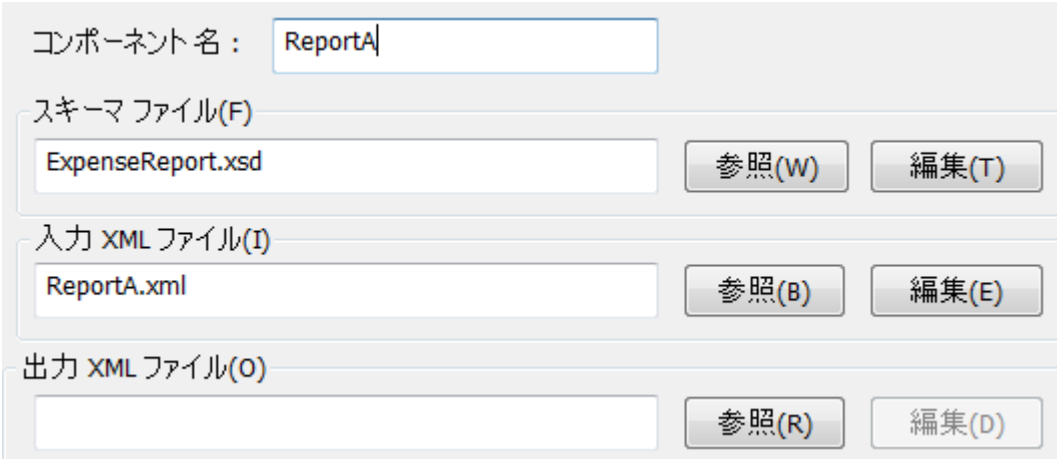
マッピングの目的は、上のファイルを基にして、2つの更なるレポートを生成することです:

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

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



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



ソースコンポーネントの設定

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

中間コンポーネントの設定

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

ターゲット コンポーネントの設定

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

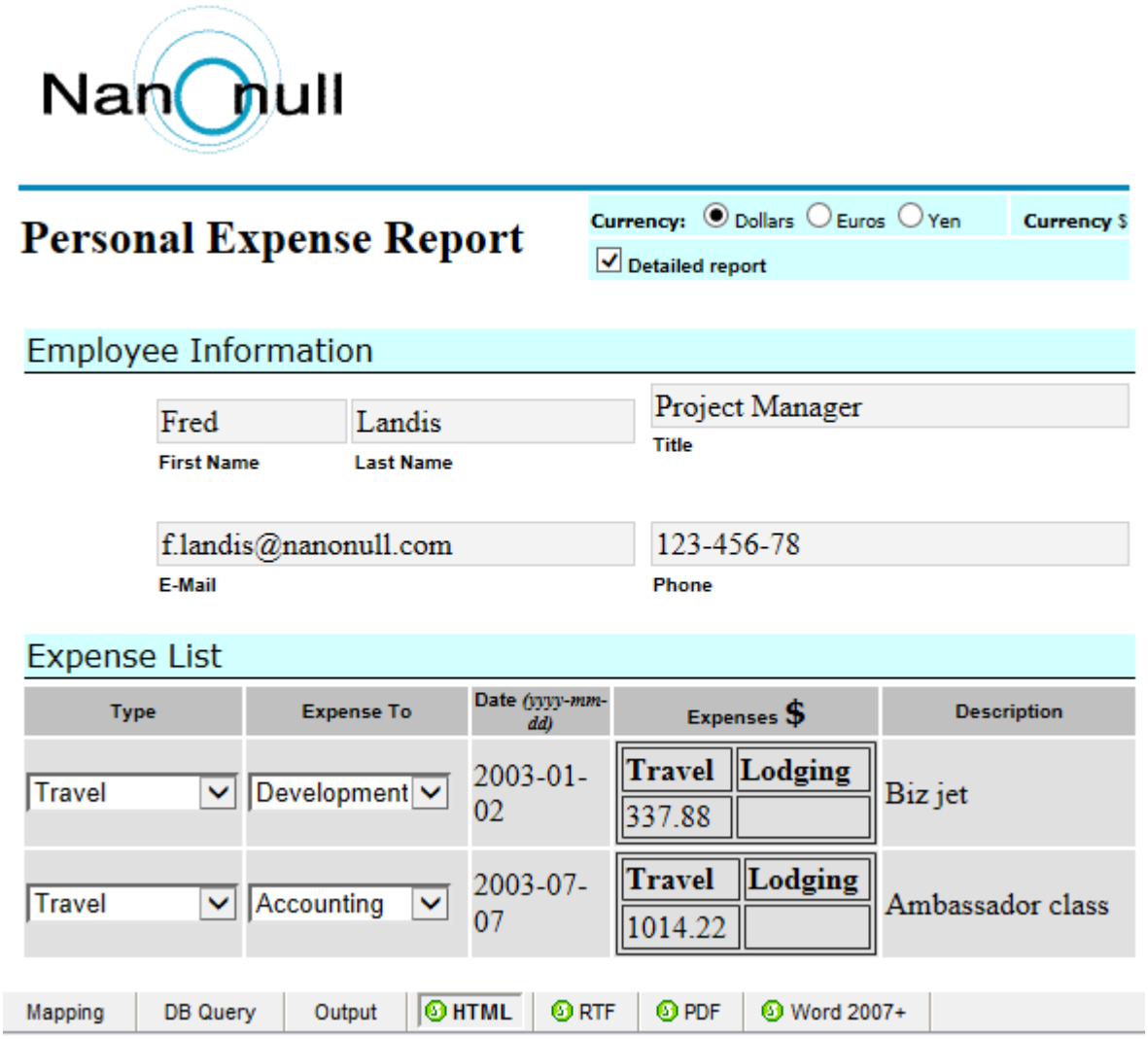
1. **ReportB.xml** は、A から B へのマッピングの結果を表します。
2. **ReportC.xml** は、B から C へのマッピングの結果を表します。

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

生成された出力ファイル

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

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

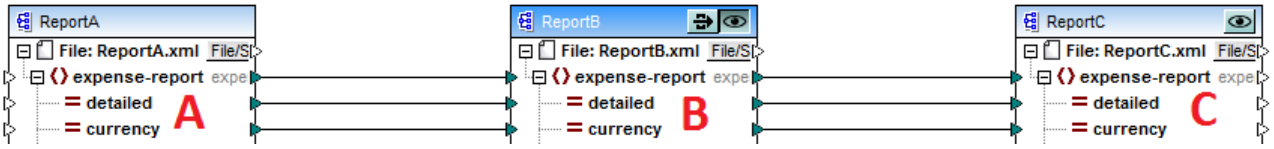


生成されたHTML出力

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

4.3.2 例: パススルーが無効な場合

このサンプル（**ChainedReports.mfd**）で使用されているマッピングは、**<マイドキュメント>\Altova\MapForce2021\MapForceExamples\Tutorial** フォルダー内で見つけることができます。このサンプルは、中間コンポーネントのパススルーボタン が無効化されている場合、出力がどのように生成されるかを説明しています。



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

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

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

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

中間コンポーネントの設定

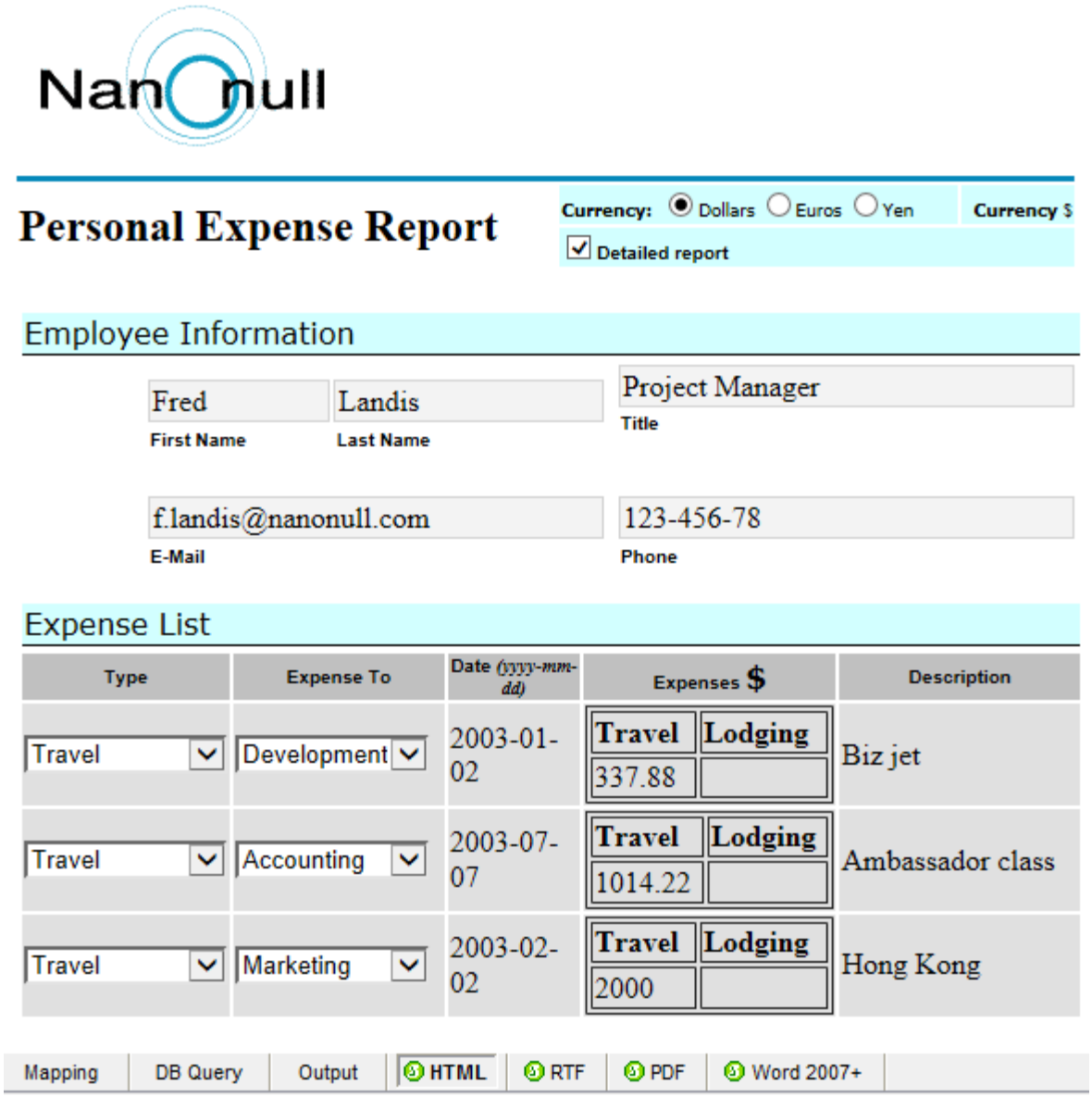
中間コンポーネント上に同じ入力と出力 ファイルが存在することは、マッピングからコードを生成する場合、またはマッピングを MapForce Server 上で実行する場合、特に重要です。前記のように、この環境は、マッピングチェーン内の各コンポーネントにより作成される出力により生成されます。ですから、中間コンポーネントが処理するファイル(この場合は **ReportB.xml**) を受け取り、異なるファイルを検索するよりも、同じファイルを次のマッピングに送ることは意味があります。(パススルーボタンが無効化されており) 中間コンポーネント上で同じ入力と出力ファイル名が存在しないと、生成されたコード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

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

1. 中間コンポーネントのプレビューボタンをクリックします。
2. マッピングをプレビューするために、**出力** タブをクリックします。
3. マッピング(<マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥Tutorial¥) と同じフォルダー内に結果出力ファイルを **ReportB.xml** として保存します。

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

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



中間コンポーネントのHTML 出力

4.4 複数の入力または出力ファイルを動的に処理

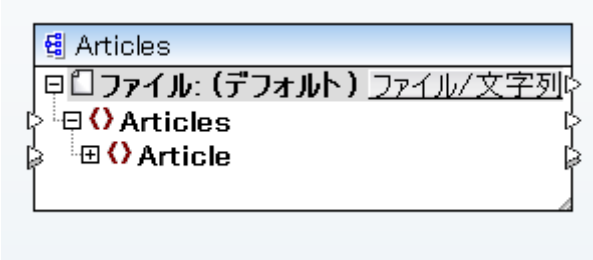
マッピングが実行される際、MapForce に複数のファイルを処理するように構成することができます（例: ディレクトリ内の全てのファイル）。この機能を使用して、以下のようなタスクを解決することができます:

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

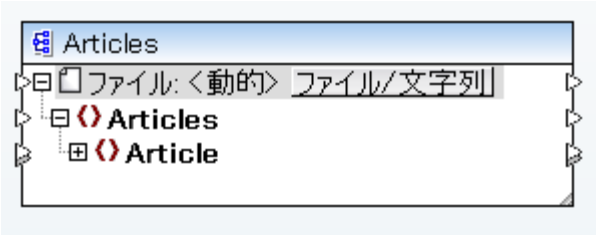
MapForce コンポーネントに複数のファイルを処理するように以下の方法で構成することができます:

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

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

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

(C) 2015-2021 Altova GmbH

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

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

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

- プロトコルバッファーファイル**

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

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

ターゲット言語	動的入力ファイル名	入力ファイル名のためにサポートされているワイルドカード	動的な出力ファイル名
XSLT 1.0	*	XSLT 1.0 によりサポートされていません	XSLT 1.0 によりサポートされていません
XSLT 2.0	*	*(1)	*
XSLT 3.0		*(1)	*
XQuery	*	*(1)	XQuery によりサポートされていません
C++	*	*	*
C#	*	*	*
Java	*	*	*
BUILT-IN	*	*	*

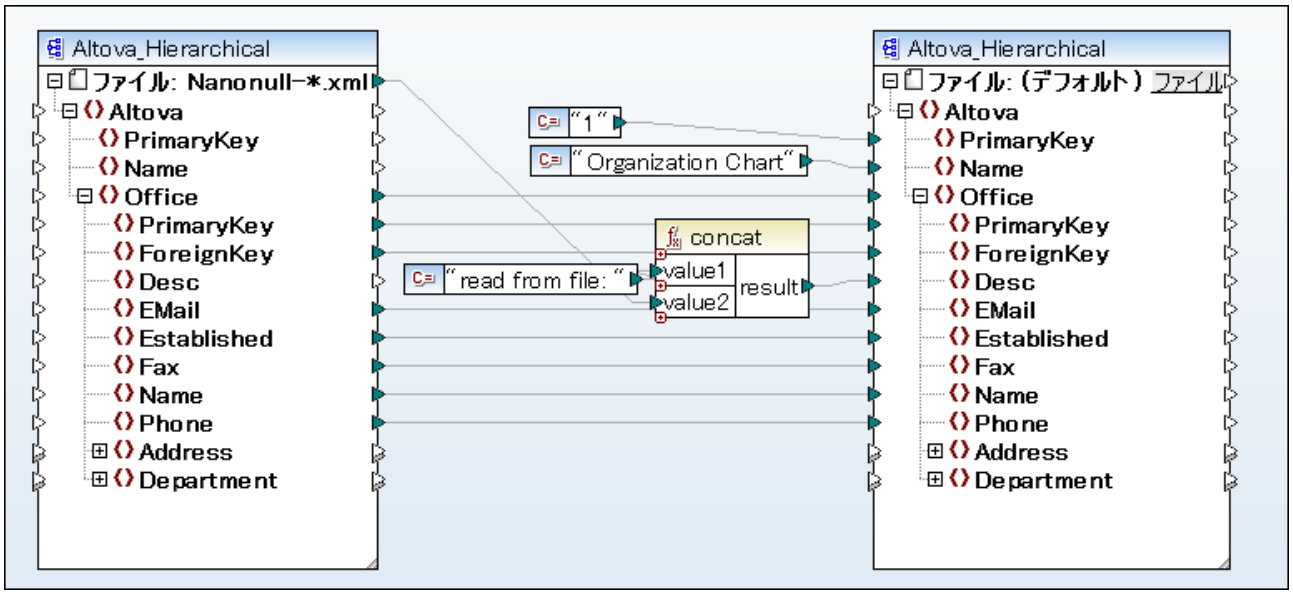
レジェンド:

*	サポートされている
(1)	XSLT 2.0、XSLT 3.0 とXQuery は `fn:collection` 関数を使用します。Altova XSLT 2.0、XSLT 3.0 と XQuery エンジン内の実装はワイルドカードを解決します。他のエンジンは異なる振る舞いをする可能性があります。

4.4.1 複数の入力ファイルを単一の出力ファイルにマップする

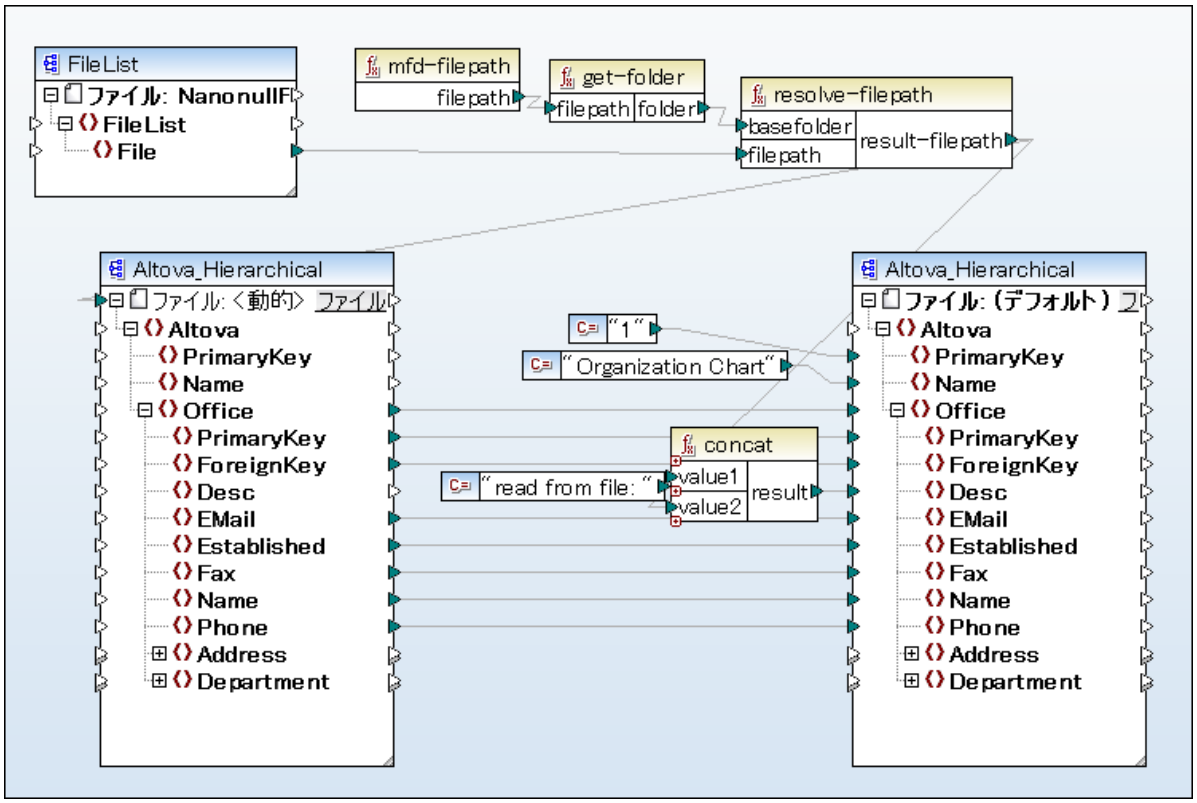
複数の入力ファイルを処理するには、以下を行います:

- コンポーネント設定ダイアログボックスに入力ファイルとしてワイルドカード(*または?)を持つのファイルパスを入力します。一致する全てのファイルが処理されます。下のサンプルは、入力 XML ファイルフィールド内でマッピング入力として名前が"Nanonull-" で開始する全てのファイルを提供する* ワイルドカード文字を使用します。ターゲットコンポーネントに動的コネクタが存在せず、ソースコンポーネントはワイルドカード * を使用して複数のファイルにアクセスしているため、複数の入力ファイルは単一の出力ファイルとしてマージされます。ターゲットコンポーネント内のルートノードの名前は**File: <default>** です。これはコンポーネント設定ダイアログボックス内で出力ファイルパスが定義されていないことを示します。複数のソースファイルは、ですから、ターゲットドキュメント内で追加されます。



MergeMultipleFiles.mfd (MapForce Basic Edition)

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

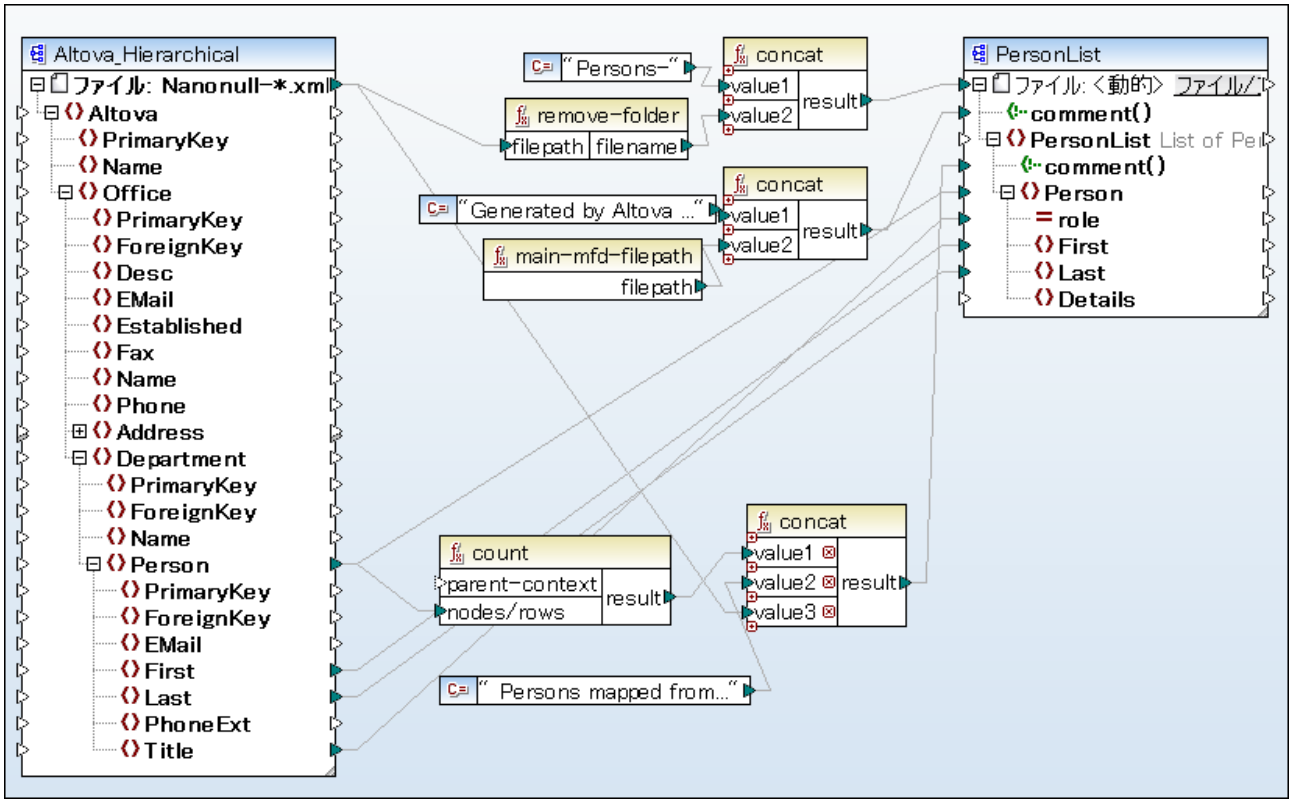


MergeMultipleFiles_List.mfd (MapForce Basic Edition)

4.4.2 複数の入力ファイルを複数の出力ファイルにマッピングする

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

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



MultipleInputToMultipleOutputFiles.mfd (MapForce Basic Edition)

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

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

4.4.3 ファイル名をマッピングパラメーターとして提供する

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

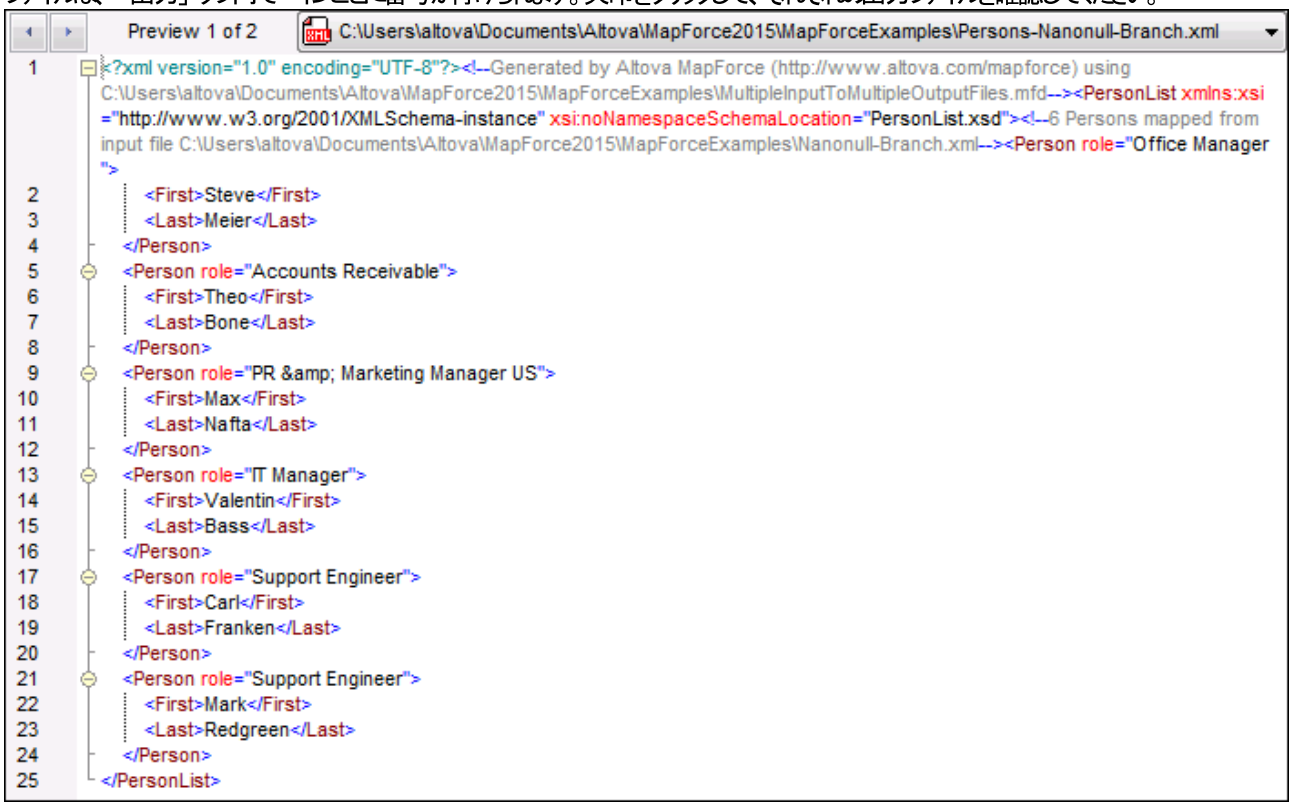
1. 単純型入力コンポーネントをマッピングに追加する（「**関数**」メニューから「**入力の挿入**」をクリックします）。コンポーネントについての更に詳しい情報は、マッピングにパラメーターを提供するを参照してください。
2. ソースコンポーネントの「**ファイル**」（ File ）または「**ファイル/文字列**」（ File/String ）ボタンをクリックし、「**マッピングから与えられた動的なファイル名を使用する**」を選択します
3. マッピングソースとして振舞う、コンポーネントのルートノードに単純型入力パラメーターを接続します。

(C) 2015-2021 Altova GmbH

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

4.4.4 複数の出力ファイルをプレビューする

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



MultipleInputToMultipleOutputFiles.mfd

生成された出力ファイルを保存するためには以下を行います:

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

4.4.5 例: 1つの XML ファイルを複数のファイルに分割

このサンプルは、単一のソースXML ファイルから動的に複数のXML ファイルを作成する方法を説明します。このサンプルに使用されるマッピングは以下で検索することができます: **<マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥Tutorial¥Tut-ExpReport-dyn.mfd**。

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

Person

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

expense-item (5)

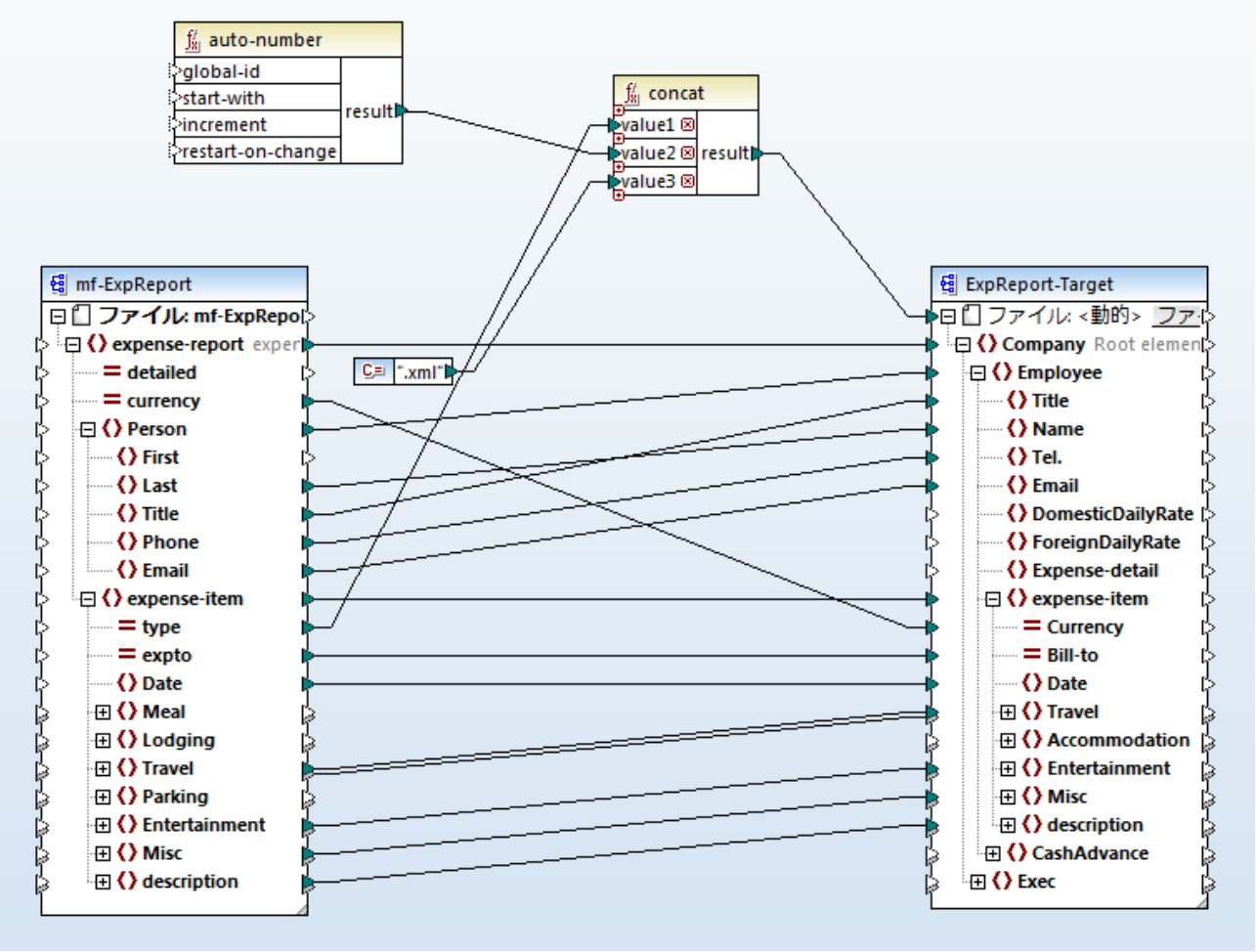
	type	expto	Date	Travel	Lodging
1	Travel	Development	2003-01-02	Travel Trav-cost=337.88	
2	Lodging	Sales	2003-01-01		Lodging
3	Travel	Accounting	2003-07-07	Travel Trav-cost=1014.22	
4	Travel	Marketing	2003-02-02	Travel Trav-cost=2000	
5	Meal	Sales	2003-03-03		

mf-ExpReport.xml (XMLSpy グリッドビューにて表示)

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

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

(C) 2015-2021 Altova GmbH



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

結果的出力ファイルは、以下のとおり動的に名前が付けられます:

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

4.4.6 例: データベーステーブルに対して複数の XML を生成

この例は、データベーステーブルのそれぞれのレコードに対して、動的に複数の XML ファイルを生成する方法を説明します。この例は、以下で参照することができます: **<マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥Tutorial¥PersonDB-dyn.mfd**。

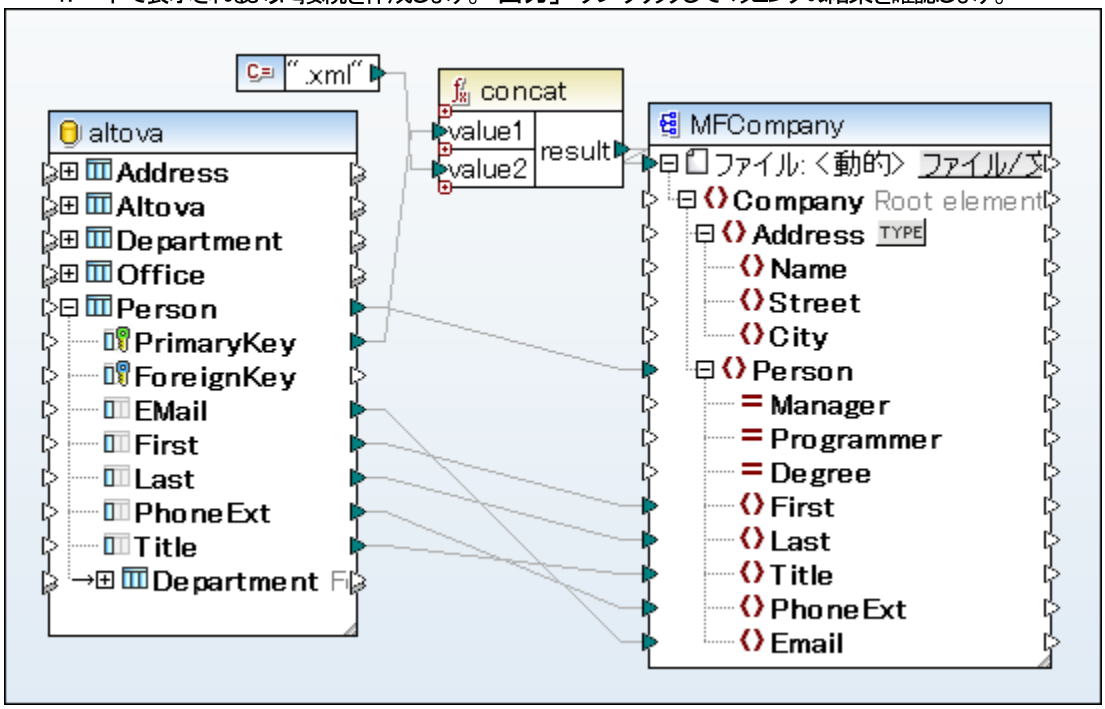
ソースデータベースファイルには 5 つのテーブルがあり、Person テーブルには 21 名の人物に関するレコードが収められています。Person テーブル内のそれぞれのレコードに対し、個別の XML ファイルを生成することがこのサンプルの目的です。

Person

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

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

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



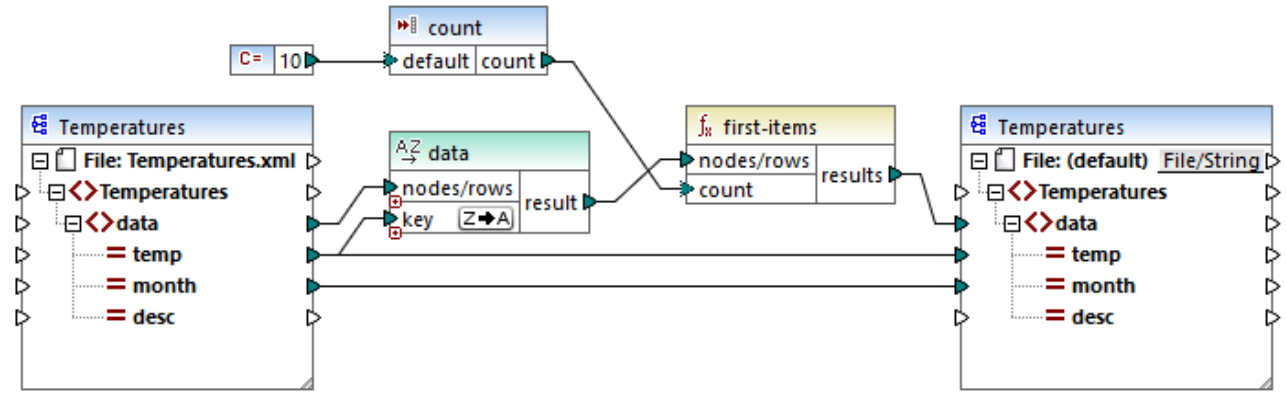
PersonDB-dyn.mfd (MapForce Professional Edition)

結果的出力ファイルは、以下のとおり動的に名前が付けられます:

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

4.5 マッピングにパラメーターを与える

パラメーターを入力としてとるマッピングを作成する場合、「単純な入力コンポーネント」と呼ばれる特別なコンポーネント型を追加して行うことができます。単純型入力コンポーネントは常に（例えば、文字列、整数などの）単純型データ型をアイテムとシーケンスの構造の代わりに持っています。例えば、下に表示されるマッピングでは単純型入力コンポーネント **count** が存在します。役割はパラメーターとして（値 **10** をデフォルトとして持つ）ソースXMLファイルから抽出される行の最大数を提供します。重要な点は、*N* がパラメーターの値である個所でマッピングが最高 *N* 気温のみを出力するように first-items 関数に入力として提供されるノードは並べ替えコンポーネントを使用して並べ替えられます。



FindHighestTemperatures.mfd

他の単純型入力コンポーネントの比較的一般的な使用方法はマッピングへのファイル名の提供です。これは入力ファイルを読み取る、または出力ファイルを動的に読み取るマッピング内で役に立ちます。複数の入力または出力ファイルを動的に処理するを参照してください。

単純型入力 コンポーネントを次のMapForce 変換言語内で使用することができます:

- BUILT-IN (「プレビュー」タブからMapForce 内で直接マッピングの変換をプレビューする場合)
- BUILT-IN (コンパイルされたMapForce Server 実行ファイルを実行する場合)
- XSLT 1.0、XSLT 2.0、XSLT 3.0、
- XQuery
- C++
- C#
- Java

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

単純型入力コンポーネント（またはパラメーター）を任意または必須で作成することができます 単純型入力コンポーネントを追加を参照してください。必要であれば、マッピング入力パラメーターのためにデフォルトの値を作成することができます デフォルトの入力値を作成するを参照してください。マッピングの実行時にパラメーターの値を明示的に提供していない場合でもこれによりマッピングを安全に実行することができます。例：ファイル名をマッピング パラメーターとして使用するを参照。

メインマッピングエリアに追加される入力パラメーターをユーザー定義関数内の入力パラメーターと区別してください。これら2つの類似点と相違点は、以下のとおりです。

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

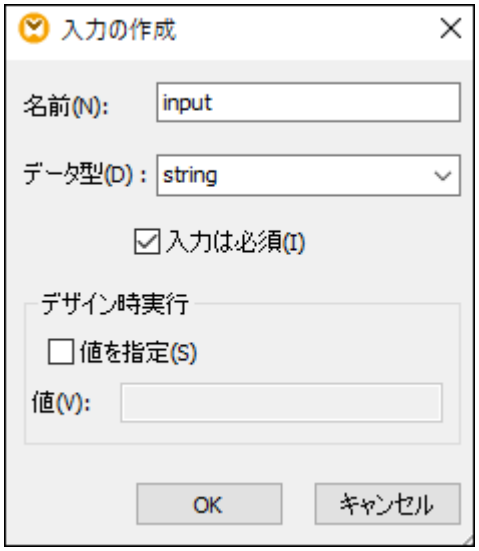
逆のマッピングを作成する場合、（メニューコマンド「ツール | 逆順マッピングの作成」を使用すると）単純型入力コンポーネントは、単純型出力コンポーネントになります。

4.5.1 単純型入力コンポーネントの追加

マッピングに単純型入力コンポーネントを追加する

1. マッピング ウィンドウに（ユーザー定義関数ではない）メインマッピングが表示されていることを確認してください。
2. 以下の1つを行ってください：

   - 「関数」メニューから「入力」をクリックします。
   - 「挿入」メニューから「入力の挿入」をクリックします。
   - 「入力の挿入」 ツールバーボタンをクリックします。



3. 名前を入力して、この入力に必要なデータ型を選択します。入力が必須のマッピングパラメーターとして扱われる必要がある場合、「入力は必須」チェックボックスを選択します。設定の完全なリストは、単純型入力コンポーネント設定を参照してください。

メモ パラメーター名は文字、数値、およびアンダースコアのみを含むことができます。他の文字は許可されていません。これによりマッピングがコード生成言語すべてのために使用できます。

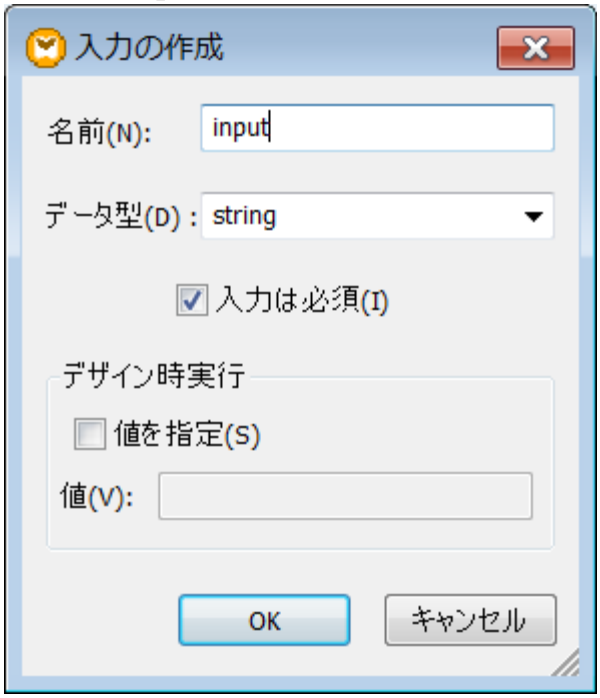
4. 「OK」をクリックします。

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

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

メモ　パラメーターの名前は、文字、数字、アンダースコアを含むことができますが、その他は許可されていません。これにより、マッピングは全てのコード生成言語と作業可能になります。

4. 「OK」をクリックします。



入力の作成 ダイアログボックス

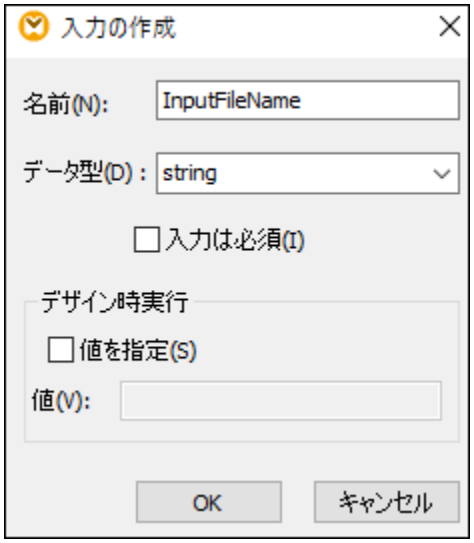
ここで定義されている設定は後に変更することができます（単純型入力コンポーネント設定を参照）。

4.5.2　単純型入力コンポーネント設定

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

「入力の編集」ダイアログボックスを開くには、以下を行います:

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



入力の編集 ダイアログボックス

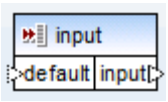
.使用することのできる設定は、以下のとおりです:

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

(C) 2015-2021 Altova GmbH

4.5.3 デフォルトの入力値を作成する

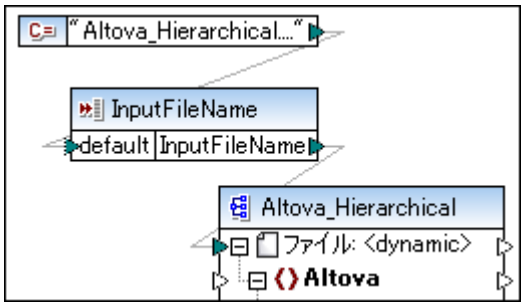
入力コンポーネントをマッピングエリアに追加すると、**default** アイテムがコンポーネントの左側に表示されます。



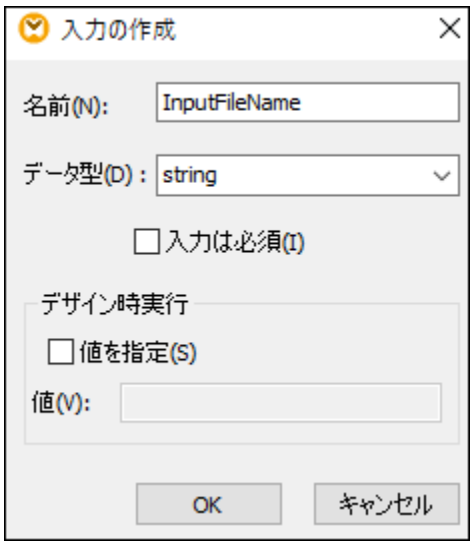
単純型入力コンポーネント

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

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



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



3. 「OK」をクリックします。

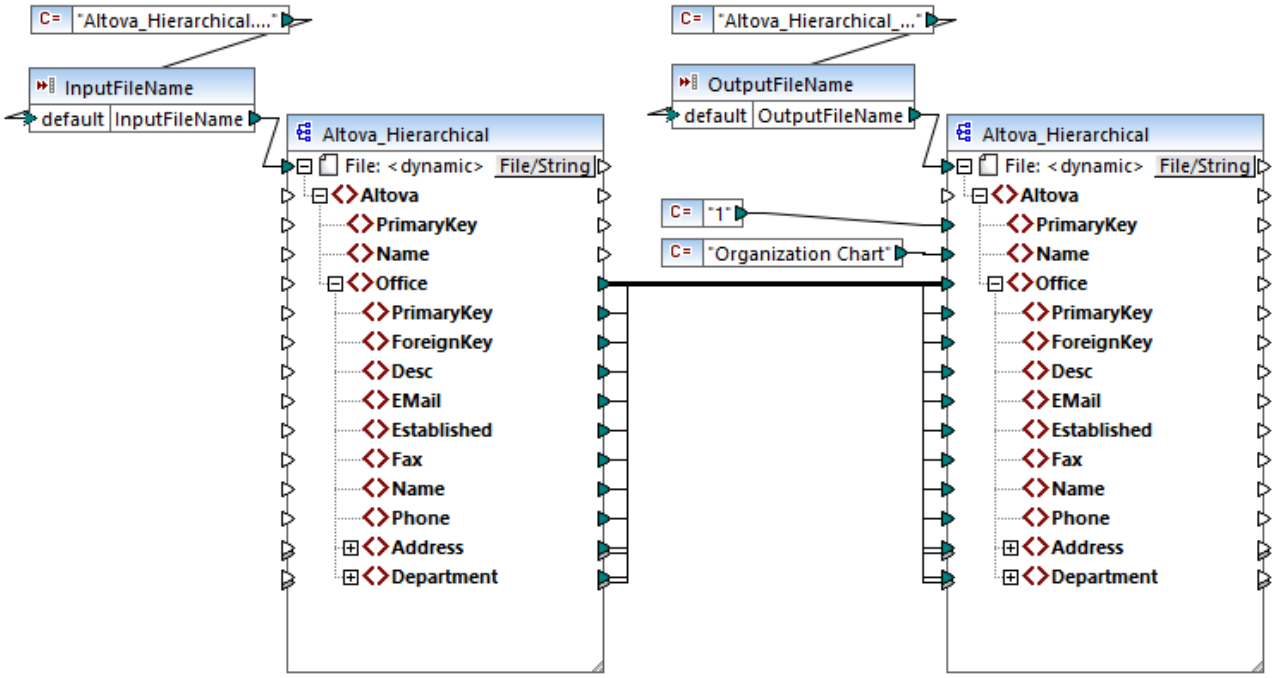
メモ　**値の指定** チェックボックスをクリックし横のボックスに値を入力した場合、入力された値は、マッピングをプレビューする際に、以前入力された(すなわち、デザイン時の実行時の)デフォルトの値を上書きします。しかしながら、デザイン時の値はMapForce Server による実行またはFlowForce Server へのデプロイ時に生成されたXSLT、XQuery、またはプログラムコードコードに影響を与えません。

4.5.4 例: ファイル名をマッピングパラメーターとして使用する

このサンプルはランタイムに入力パラメーターを取り込むマッピングの実行に必要なステップを説明します。マッピングデザインファイルは、以下のパスにあるサンプルを使用します: **<マイドキュメント >\Altova\MapForce2021\MapForceExamples\FileNamesAsParameters.mfd**.

このマッピングはXML ファイルからデータを抽出し、ターゲット XML ファイルに書き込みます。データはターゲットファイルにほぼ変更されることなく書き込まれます。属性 **PrimaryKey** と**Name** のみがマッピングからの定数の一部と共に作成されます。マッピングのメインの目的はマッピングのランタイムで呼び出し元がマッピングのパラメーターとして入力ファイルの名前と出力ファイルの名前を指定できるようになることです。

これを達成するには、マッピングには2 つの入力コンポーネントが存在します: **InputFileName** and **OutputFileName**。これらのコンポーネントはソースとターゲット XML ファイルの入力ファイル名（および出力ファイル名をそれぞれ）与えます。この理由のため、これらのコンポーネントは、**ファイル: <動的>** アイテムに接続されています。**ファイル/文字列** (File/String) ボタンをクリックして、**マッピングにより提供された動的なファイル名を使用する**を選択してコンポーネントをこのモードに切り替えることができます。



FileNamesAsParameters.mfd (MapForce Enterprise Edition)

InputFileName と**OutputFileName** コンポーネントのタイトルバーをダブルクリックすると、コンポーネントをビューすることができます。例えば、単純型入力コンポーネント設定で説明されているとおり、入力パラメーターのデータ型を指定、または、入力パラメーターの名前を変更することができます。このサンプルでは、入力と出力パラメーターは以下のように構成されています:

- **InputFileName** パラメーターは型 [string] で、同じマッピング内で定義されている定数により与えられているデフォルトの値を持っています。定数は型 [string] で、その値は [Altova_ Hierarchical.xml] です。このために、他の値がパラメーターとして与えられないと想定し、マッピングが実行されると [Altova_ Hierarchical.xml] と呼ばれるファイルからデータを読み取ろうと試みます。
- **OutputFileName** パラメーターは型 [string] で、同じマッピング内で定義されている定数により与えられているデフォルトの値を持っています。定数は型 [string] で値は [Altova_ Hierarchical_ output.xml] ですこのため、このために、他の値がパラメーターとして与えられないと想定し、マッピングは実行時に [Altova_ Hierarchical_ output.xml] と呼ばれるXML 出力ファイルを作成します。

以下のセクションはマッピングの実行方法と以下の変換言語でパラメーターを与える方法を説明しています:

- RaptorXML Server を使用したXSLT 2.0
- MapForce Server を使用したビルドイン（MapForce Server 実行ファイル）
- Java
- C#
- C++

XSLT 2.0

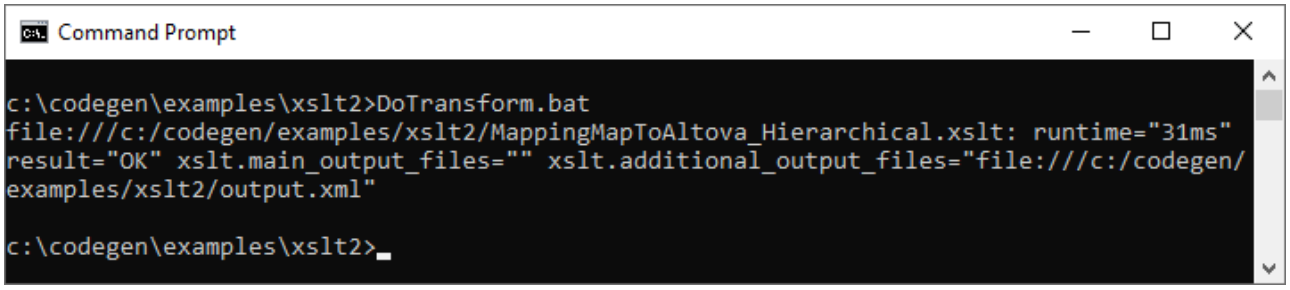
XSLT 1.0、XSLT 2.0、またはXSLT 3.0 でコードを生成する場合、XSLTファイルに加え、**DoTransform.bat** バッチファイルが選択されたターゲットディレクトリで生成されます。**DoTransform.bat** によりRaptorXML Server を使用してマッピングを実行できるようになります。 RaptorXML Server を使用した自動化を参照してください。

異なる入力（または、出力）ファイルを使用するために**DoTransform.bat** ファイルを必要とするパラメーターが含まれるように以下のように編集します:

1. 最初にXSLT コードを生成します。例えば、XSLT 2.0, を生成する場合「**ファイル | コード生成 | XSLT 2.0**」を選択します。
2. <**マイドキュメント**>¥Altova¥MapForce2021¥MapForceExamples¥から**Altova_Hierarchical.xml** ファイルをXSLT 2.0 コードが直接生成されるディレクトリにコピーします（このサンプルでは、**c:\codegen\examples\xslt2**）。**InputFileName** パラメーターにカスタム値を提供しない場合、前記の通り、マッピングはこのファイルを読み取ろうと試みます。
3. **DoTransform.bat** が%* の前または後にカスタム入力パラメーターを含むように編集します。パラメーターの値は一重引用符で閉じられていることに注意してください。使用することのできる入力パラメーターは**rem**（リマーク）セクションにリストされています。**output.xml** と呼ばれる出力ファイルを生成すると想定します。これを達成するには**DoTransform.bat** ファイルを以下のように変更します:

```
@echo off

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

DoTransform.batファイルを実行すると、RaptorXML Server は**Altova_Hierarchical.xml** を入力パラメーターとして使用して、変換を完了します。上のステップに従うと生成されたファイルの名前は**output.xml** になります。



MapForce Server 実行ファイル

カスタム入力パラメーターをMapForce Server 実行ファイルに提供する方法:

1. これをまだ行っていない場合 **FileNamesAsParameters.mfd** を<マイドキュメント>**\Altova\MapForce2021\MapForceExamples** ディレクトリから開きます。
2. [ファイル] メニューで [MapForce Server 実行ファイルにコンパイルする] をクリックします。 MapForce Server 実行ファイルにマッピングをコンパイルするも参照してください。プロンプトされると、.mfx 実行ファイルを(この場合は **c:\codegen\examples\mfx**)コンピューターのディレクトリに実行ファイルを保存します。
3. <マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥ディレクトリから **Altova_Hierarchical.xml** ファイルを .mfx ファイルを保存した場所にコピーします。
4. MapForce Server を以下のコマンドで実行します:

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

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

メモ [MapForceServer.exe が内部または外部コマンド、操作可能なプログラム、またはバッチファイルとして識別されていません] というエラーメッセージが表示されると、現在のディレクトリを MapForce Server 実行可能ファイルがインストールされている場所に変更してください。マッピングを実行する度にパスを変更することを回避するために、オペレーティングシステムの PATH 環境変数に MapForce Server 実行可能ファイルがインストールされているディレクトリへのパスを追加します(例えば、**C:\Program Files (x86)\Altova\MapForceServer2021\bin**)。

With MapForce Server を使用すると、マッピングの実行は(C++、C#、またはJava などの言語を呼び出す) MapForce Server API を呼び出しても行うことができます。このシナリオに関する詳細は、MapForce Server ドキュメンテーション (https://www.altova.com/ja/documentationr)を参照してください。

Java

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

1. これを既に行っていない場合 **FileNamesAsParameters.mfd** サンプルを<マイドキュメント>**\Altova\MapForce2021\MapForceExamples** ディレクトリから開きます。
2. [ファイル] メニューから[コードの生成 | Java]をクリックします。
3. Java コードを実行可能 JAR ファイルにコンパイルします。Eclipse を使用してこれを行う方法の例は、 例: Java コードの生成と実行を参照してください。
4. **Altova_Hierarchical.xml** ファイルを<マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥から .jar ファイルが存在するディレクトリにコピーします。**InputFileName** パラメーターにカスタム値を提供しない場合、前記の通り、マッピングはこのファイルを読み取ろうと試みます。
5. アプリケーションを以下のコマンドで実行する

```
java -jar Mapping.jar /OutputFileName "output.xml"
```

上のコマンドでは、入力パラメーター `/OutputFileName` は生成される出力ファイルの名前を提供します。

メモ .jar ファイルにパラメーターをパスする際にワイルドカードを使用する場合、ワイルドカードパラメーターを引用符で囲んでください。例:

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

C#

カスタム入力パラメーターを MapForce により生成された C# コマンドラインアプリケーションに提供する方法 :

1. これをまだ行っていない場合 **FileNamesAsParameters.mfd** を<マイドキュメント>\Altova\MapForce2021\MapForceExamples\ ディレクトリから開きます。
2. ファイル ファイルメニューから [コードの生成 | C#] を選択しターゲットディレクトリを選択します（このサンプルの場合、**C:\codegen\examples\cs** ）。
3. ソリューションを Visual Studio で開きビルドします（**Ctrl + Shift + B**）。
4. **Altova_Hierarchical.xml** ファイルを<マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥から **Mapping.exe** が生成されたディレクトリにコピーします（このサンプルでは **C:\codegen\examples\cs\Mapping\bin\Debug**）。**InputFileName** パラメーターにカスタム値を提供しない場合、前記の通り、マッピングはこのファイルを読み取ろうと試みます。
5. コマンドプロンプトウィンドウを開き **Mapping.exe** がある場所にディレクトリを変更します。

```
cd C:\codegen\examples\cs\Mapping\bin\Debug
```

6. アプリケーションを以下のコマンドで実行する

```
Mapping.exe /OutputFileName output.xml
```

上のコマンドでは、入力パラメーター `/OutputFileName` は生成される出力ファイルの名前を提供します。

C++

カスタム入力パラメーターを MapForce により生成された C++ コマンドラインアプリケーションに提供する方法：

1. これをまだ行っていない場合 **FileNamesAsParameters.mfd** を<マイドキュメント>\Altova\MapForce2021\MapForceExamples\ ディレクトリから開きます。
2. ファイル ファイルメニューから [コードの生成 | C++] を選択しターゲットディレクトリを選択します（このサンプルの場合、**C:\codegen\examples\cpp** ）。
3. ソリューションを Visual Studio で開きビルドします（**Ctrl + Shift + B**）。
4. **Altova_Hierarchical.xml** ファイルを<マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥から **Mapping.exe** が生成されたディレクトリにコピーします（このサンプルでは **C:\codegen\examples\cpp\Mapping\Debug**）。**InputFileName** パラメーターにカスタム値を提供しない場合、前記の通り、マッピングはこのファイルを読み取ろうと試みます。
5. コマンドプロンプトウィンドウを開き **Mapping.exe** がある場所にディレクトリを変更します。

```
cd C:\codegen\examples\cpp\Mapping\Debug
```

6. アプリケーションを以下のコマンドで実行する

```
Mapping.exe /OutputFileName output.xml
```

上のコマンドでは、入力パラメーター `/OutputFileName` は生成される出力ファイルの名前を提供します。

4.6 マッピングから文字列の値を返す

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

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

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

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

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

言語	作動のしくみ
BUILT-IN（マッピング変換のプレビュー時）	ファイルベースのマッピング出力をプレビューする同様、マッピングウィンドウの「出力」 タブをクリックすることにより、出力コンポーネントをプレビューすることができます。
BUILT-IN（ MapForce Server 実行ファイルの実行時）	コンパイルされたMapForce Server 実行ファイルを実行すると（MapForce のマッピングをコンパイルを参照）、マッピング出力は、標準出力 ストリーム（stdout）内に返され、表示またはファイルへリダイレクトすることができます。例: MapForce server 実行ファイルの名前が **MyMapping.mfx** とします。以下の構文を使用して、マッピング出力を **output.txt** ファイルにリダイレクトし、全てのエラーを **log.txt** ファイルにリダイレクトします: `MapForceServer.exe run MyMapping.mfx >output.txt 2>log.txt`
XSLT 1.0、XSLT 2.0、XSLT 3.0	生成されたXSLT ファイル内で、マッピング内で定義されている単純型出力コンポーネントは XSLT 変換の出力になります。 RaptorXML Server を使用する場合、RaptorXML Server にマッピング出力をファイルに書き込み、`--output` パラメータに値としてパスするように命令することができます。 ファイルに出力を書き込む、**DoTransform.bat** ファイル内の`--output` パラメーターに追加または編集する場合。例: 次のDoTransform.bat ファイルは**Output.txt** ファイルへのマッピング出力に書き込むために編集されました（ハイライトされたテキスト参照してください）。

(C) 2015-2021 Altova GmbH

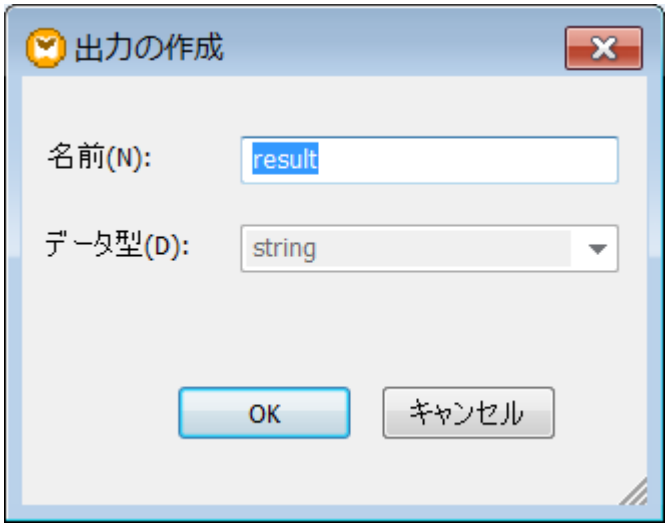
	``` RaptorXML xslt --xslt-version=2 -- input="MappingMapToResult1.xslt" --output="Output.txt" %* "MappingMapToResult1.xslt" ``` `--output` パラメーターが定義されていない場合、マッピングが実行される際、マッピング出力が標準出力ストリーム(stdout) に書き込まれます。
C++、C#、Java	生成されたC++、C#、とJava コード内で、マッピング出力は、生成されたアプリケーションの標準出力に書き込まれます。 マッピングが複数のターゲットコンポーネントを含む場合、生成されたアプリケーションは、各ターゲットコンポーネント標準出力を連結して、1つのまとめられた標準出力を返します。

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

4.6.1　単純型出力コンポーネントの追加

マッピングエリアに出力コンポーネントを追加する

1. ウィンドウが(ユーザー定義関数ではない)インのマッピングを表示していることを確認してください。
2. 以下の一つを行います:
   a. 「**関数**」メニューから、「**出力の挿入**」をクリックします。
   b. 「**出力の挿入**」 ツールバーボタンをクリックします。
3. コンポーネントのための名前を入力します。
4. 「OK」をクリックします。



出力の作成ダイアログボックス

コンポーネント名を以下の方法で後に変更することができます:

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

4.6.2 例: 関数出力のプレビュー

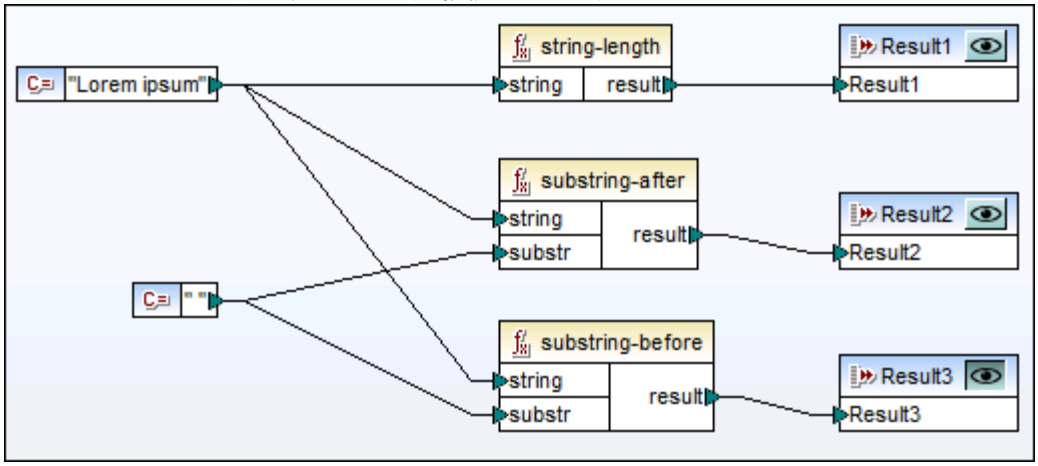
この例は、単純型出力コンポーネントを使用してMapForce関数により返された出力を確認する方法を説明します。関数に関する基本的理解と、MapForce関数に関する全般的な知識があると、このサンプルをよく理解することができます。MapForce関数の基礎知識がない場合、例を開始する前に関数の使用を参照してください。

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

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

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

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



単純型出力コンポーネントを使用して関数の出力をテストする

(C) 2015-2021 Altova GmbH

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

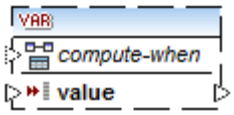
関数の出力をプレビューする方法:

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

4.7 変数の使用

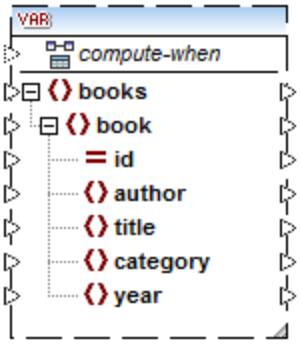
変数とは更なる処理のために中間マッピングの結果を保管するために使用される特別なコンポーネントです。マッピング上のデータを一時的に「記憶」し、処理する必要がある場合があります。例えば、ターゲットコンポーネントにコピーされる前に、フィルター、または関数を適用するなど。

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



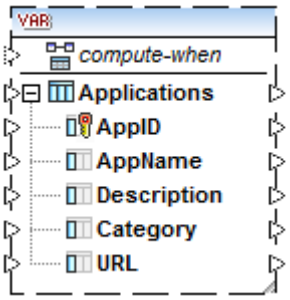
単純型変数

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



XMLスキーマから作成された複合型変数

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

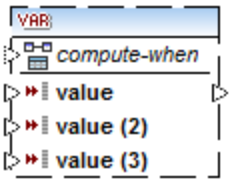


データベーステーブルから作成された複合型変数

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

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

(C) 2015-2021 Altova GmbH



複製された入力を持つ単純型変数

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

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

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

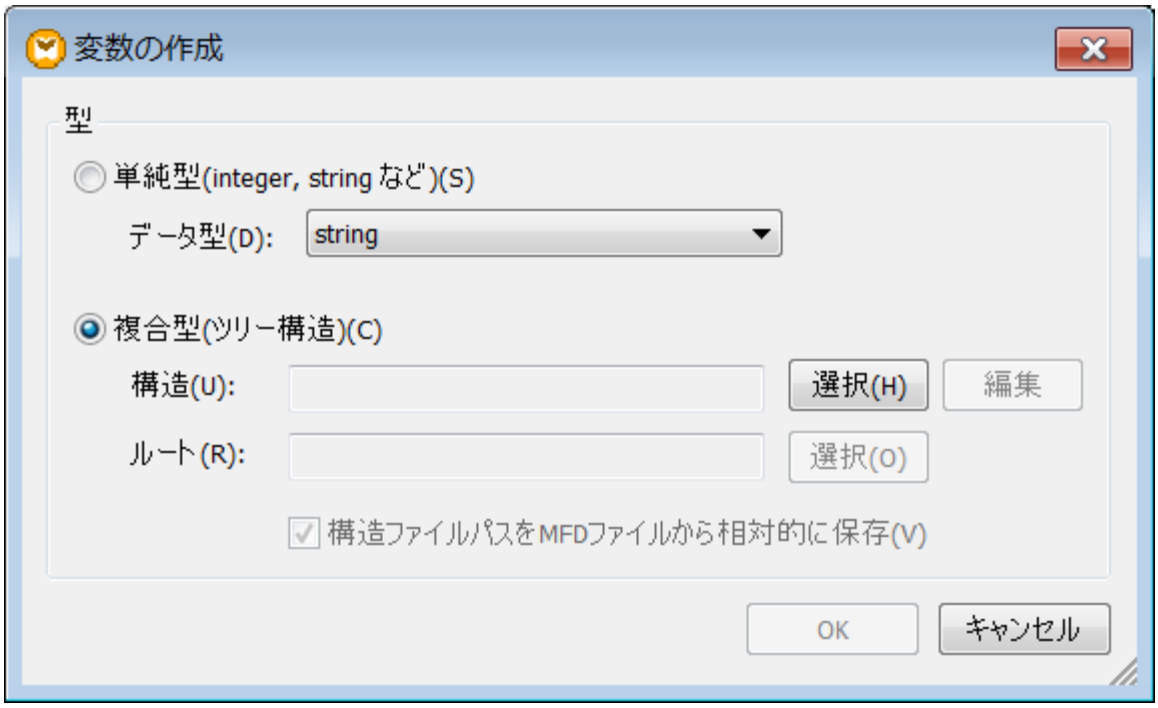
メモ 変数は、マッピング変換言語がXSLT 1.0 に設定されていない場合サポートされません。

4.7.1 変数の追加

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

メニュー、またはツールバーコマンドの使用

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

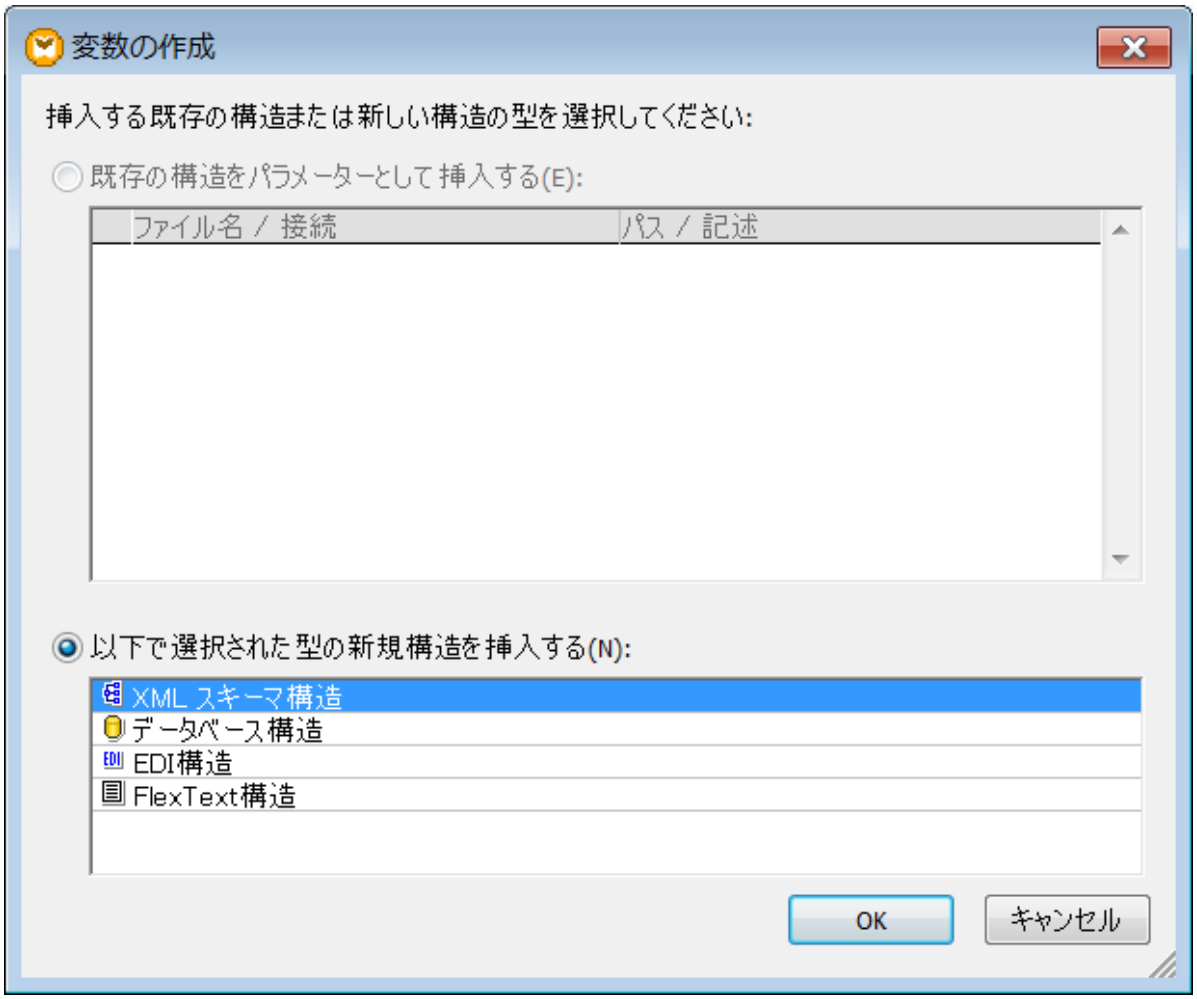


2. 挿入する変数の型を選択します（単純型 または 複合型）。

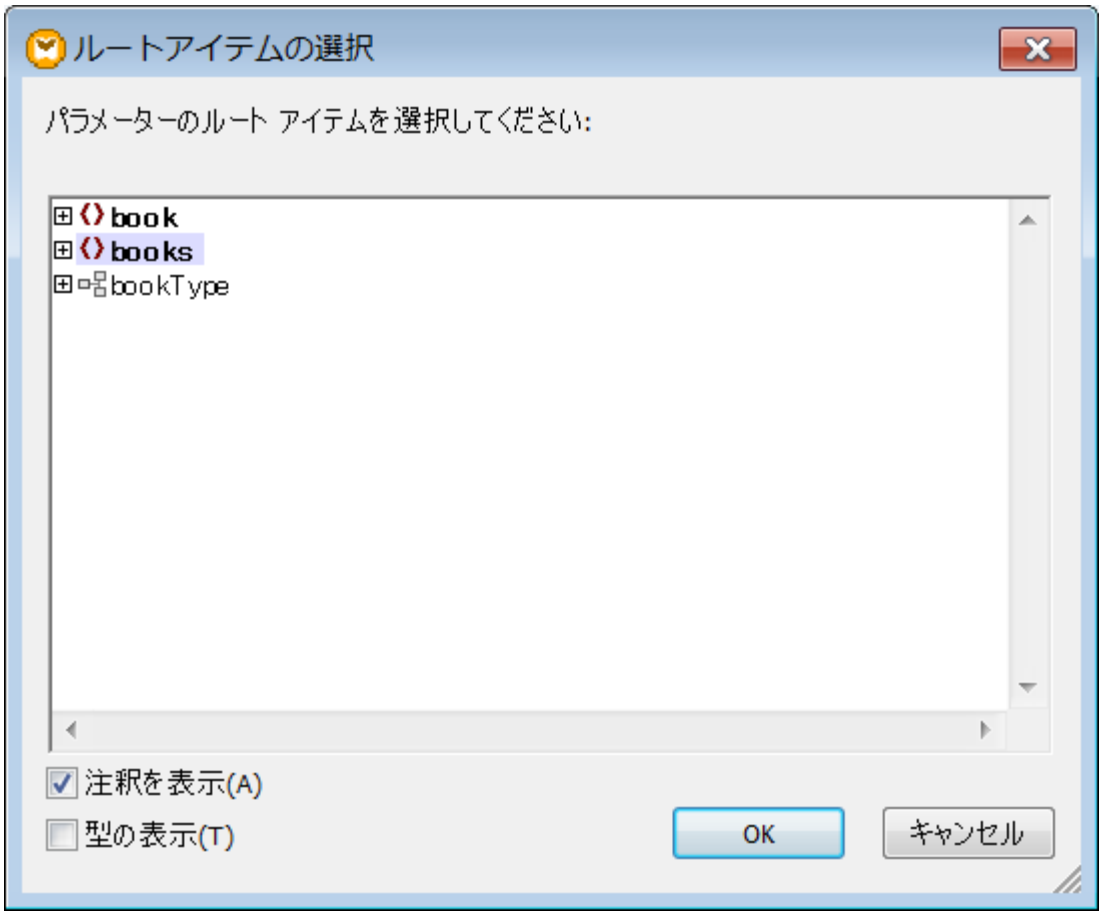
"複合型" を選択した場合、追加ステップが存在します:

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

(C) 2015-2021 Altova GmbH

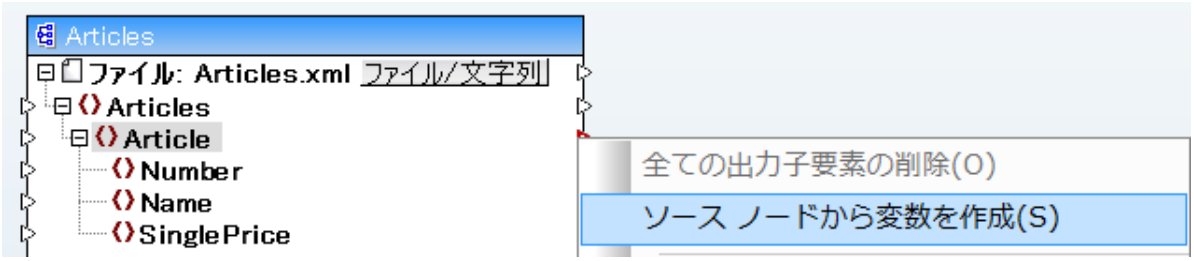


4. プロンプトされると、構造のルートアイテムを指定します。XML スキーマの場合、ルート アイテムは、グローバルに定義されているすべての要素であることができます。データベースの場合は、ルートアイテムはすべてのテーブルであることができます。



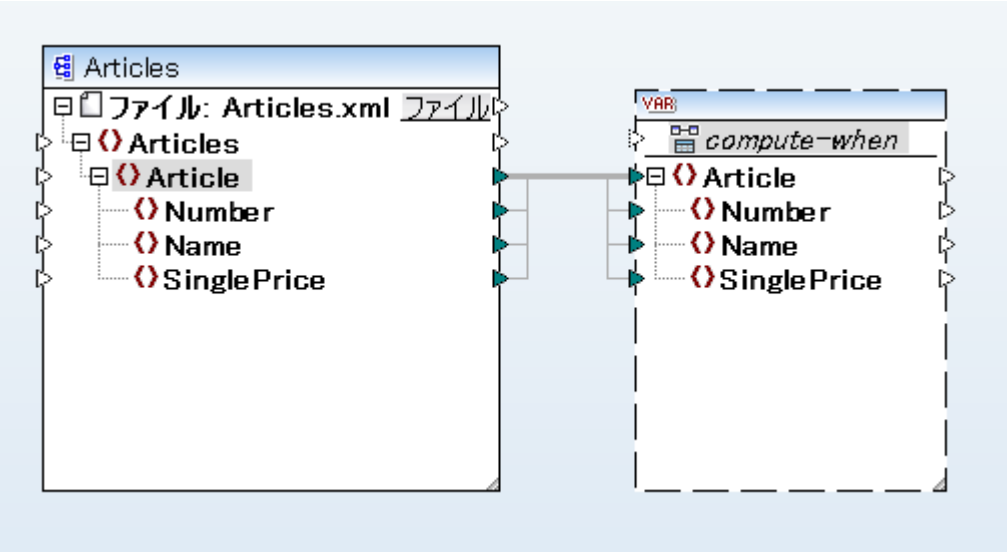
コンテキストメニューの使用

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



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

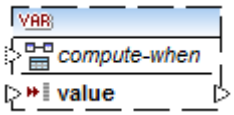
(C) 2015-2021 Altova GmbH



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

4.7.2 変数のコンテキストとスコープの変更

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



「Compute-when」アイテム

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

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

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

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

「Compute-when」が接続されていない場合（デフォルト）

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

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

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

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

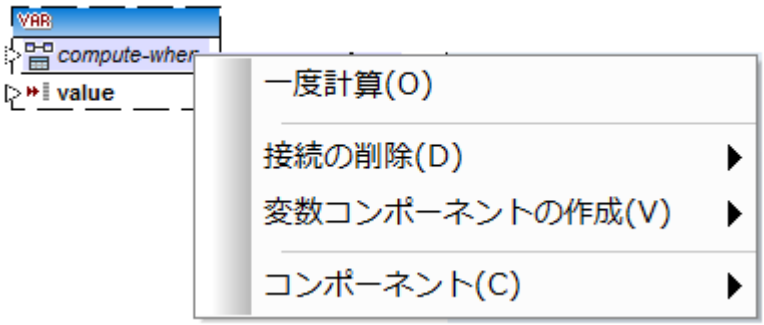
「Compute-when」が接続されている場合

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

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

「Compute-once」

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



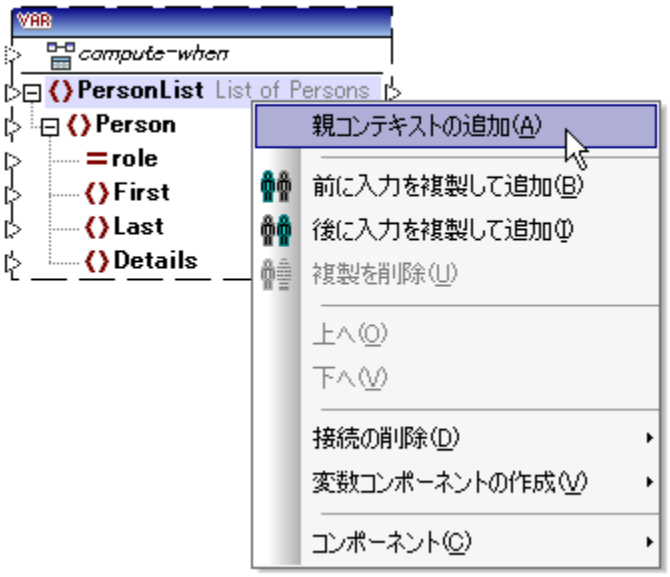
変数のスコープを compute-when=once に変更する場合、このような変数は1度のみ評価されるため、入力コネクタは、compute-when アイテムから削除されます。

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

親コンテキスト

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

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

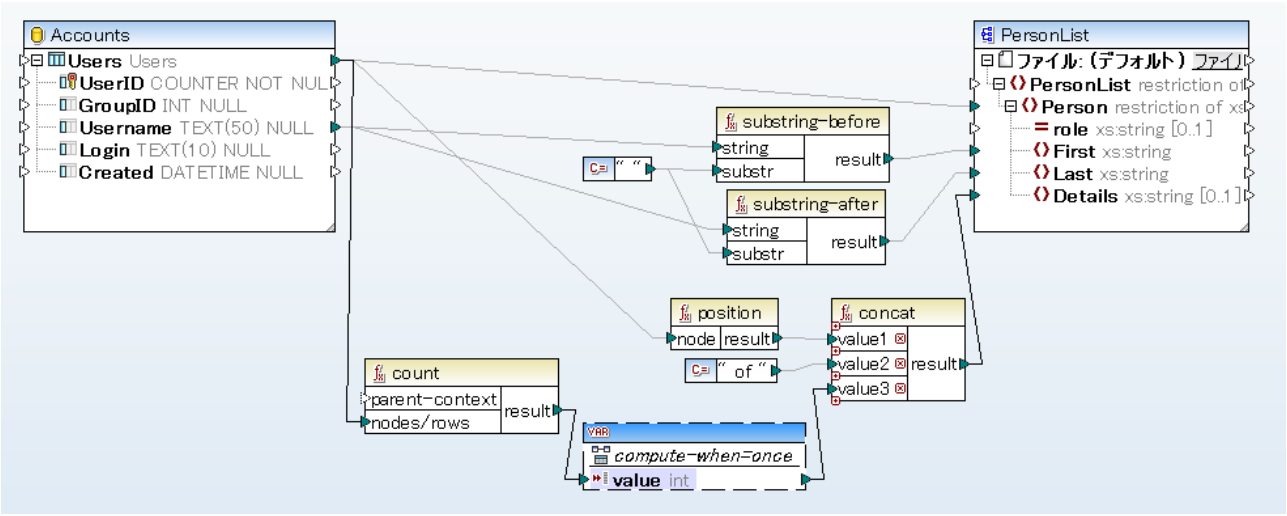


親コンテキストは、仮想の「親」ノードをコンポーネント内の階層構造に追加します。これにより、同じ、または異なるソースコンポーネント内で追加ノードを反復することができます。

4.7.3 例: データベースのテーブルの行の計算

このサンプルで説明されているマッピングは、<マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥フォルダー内の **DB_UserList.mfd** です。このマッピングは、ユーザーのレコードを "Users" という名前のデータベーステーブルから抽出し、XML ファイルに書き込みます。データベースの列 "Username" には、個人の姓と名の両方が含まれています（例えば、"Vernon Callaby"）。このマッピングの目的は以下のとおりです:

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



DB_UserList.mfd

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

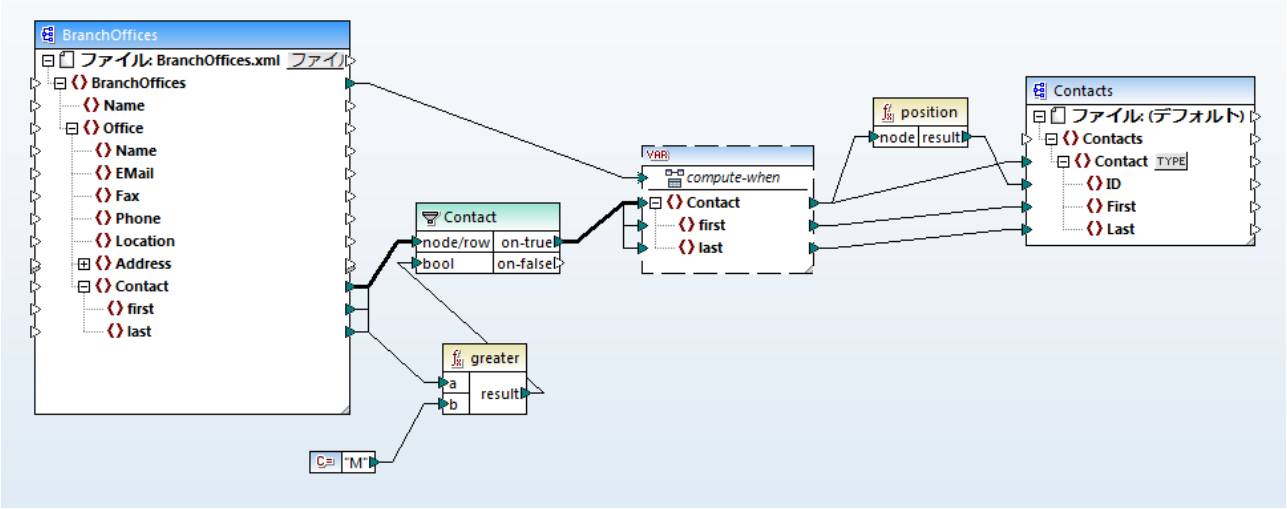
フィールド"Username"の値が substring-before と substring-after 関数に与えられています。これらの2つの関数は、スペース文字(" ")の前後のテキストをそれぞれ抽出し、このマッピングの目的を達成します。

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

4.7.4 例: ノードのフィルターと番号付け

このサンプル内で説明されているマッピングのシーケンスは、<マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥フォルダー内の **PositionInFiltered.mfd** で使用することができます。

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



PositionInFilteredSequence.mfd

上の目的を達成するには、次のコンポーネント型 がマッピングに追加されました

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

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

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

(C) 2015-2021 Altova GmbH

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

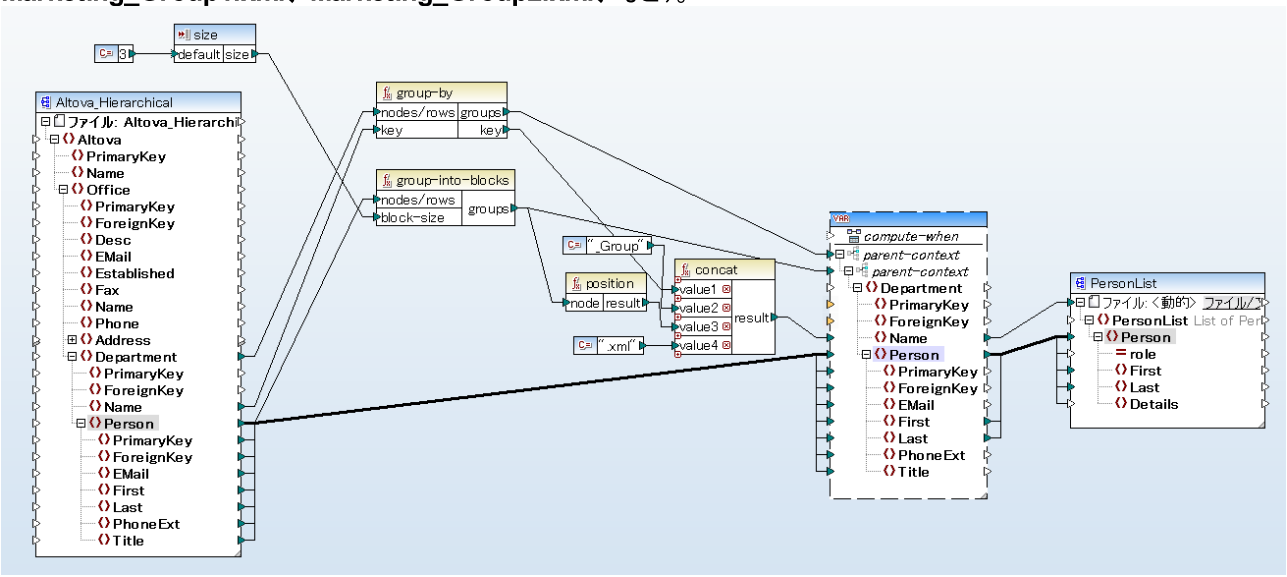
position 関数は、変数の各反復のために現在のシーケンスの番号を返します。8 件の連絡先がフィルターの条件を満たしています。ですから、マッピングをプレビューして、出力を確認すると、1 から 8 の ID が、ターゲットコンポーネントの ID 要素に書き込まれていることが確認することができます。

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

4.7.5　例:記録のグループ化、および、サブグループ化

このサンプルで説明されているマッピングは、<マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥フォルダー内の **DividePersonsByDepartmentIntoGroups.mfd** です。

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



DividePersonsByDepartmentIntoGroups.mfd

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

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

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

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

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

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

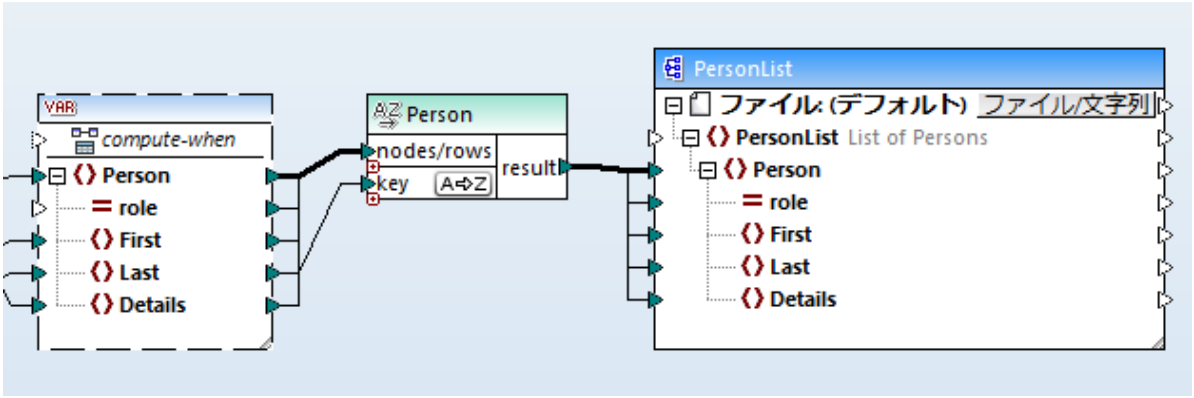
(C) 2015-2021 Altova GmbH

4.8 データの並べ替え

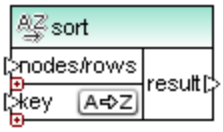
特定のソートキーをベースにした入力データの並べ替えを行う場合、ソートコンポーネントを使用してください。ソートコンポーネントは、XSLT2、XQuery、内蔵の実行エンジンにてサポートされます。変換言語が「Built-in」の場合、並べ替えコンポーネントを使用して、データベーステーブルのデータを並べ替えることができます。しかしながら、パフォーマンスを向上するには、SQL-WHERE/ORDER コンポーネントを使用することにより達成することができます。詳細に関しては、次を参照してください：[データベースデータのフィルターと並べ替え（SQL WHERE/ORDER）](#)。

マッピングに並べ替えコンポーネントを追加するには、以下を行います:

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



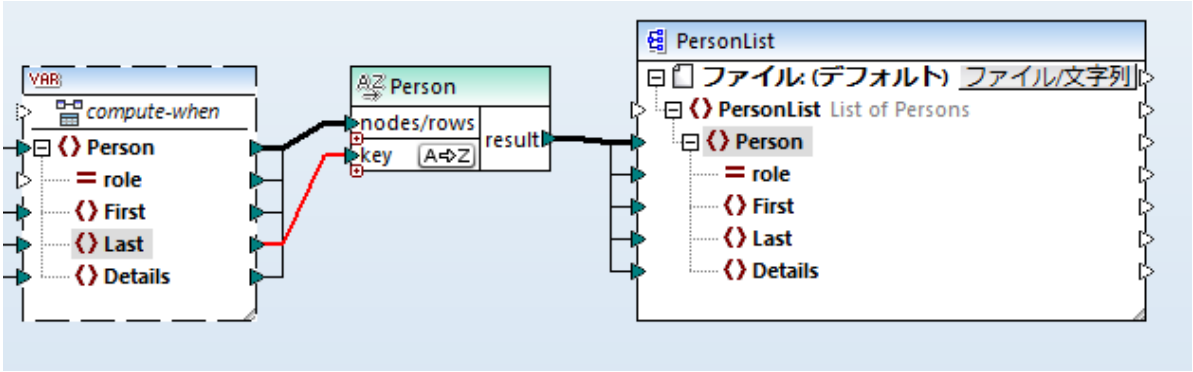
- 「挿入」メニューから「並べ替え」（または、「並べ替え」 ツールバーボタン）をクリックします。「未接続」フォーム内に並べ替えコンポーネントが挿入されます。



ソースコンポーネントに接続が構築されると，タイトルバー名は、`nodes/rows` アイテムに接続されているアイテムの名前に変更されます。

並べ替えるアイテムを定義する:

- 並べ替えコンポーネントの `key` パラメーターを並べ替えるアイテムに接続します。例えば、下のマッピングでは、`Person` nodes/rows は、フィールド `Last` に従い並べ替えられます。

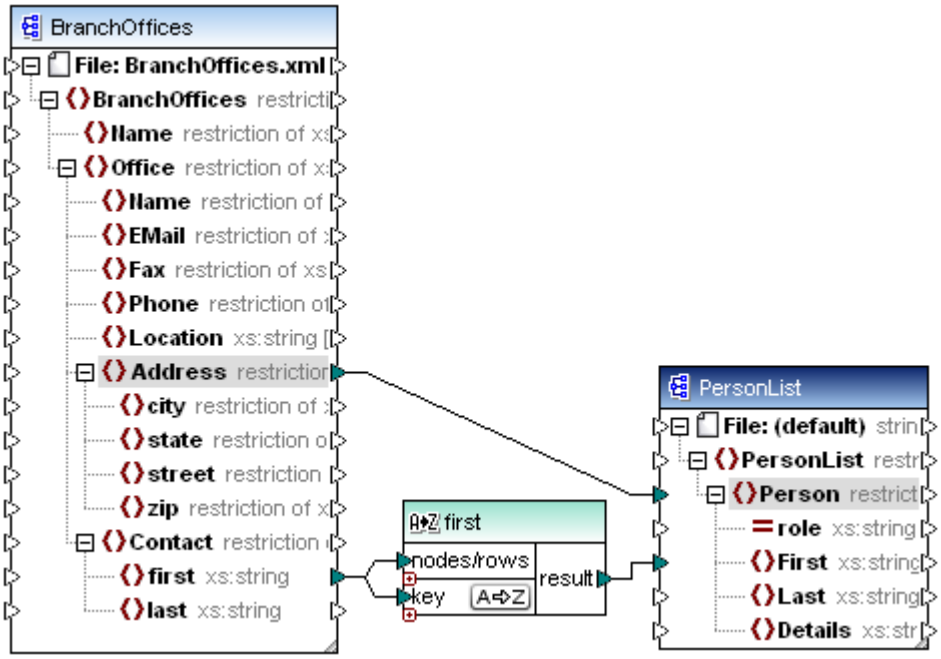


並べ替えの順序の変更:

- 並べ替えコンポーネント内の A⇨Z アイコンをクリックすると、Z➔A に変更され、並べ替えの順序は逆の順序になったことを示します。

単純型アイテムにより構成される入力データの並べ替え:

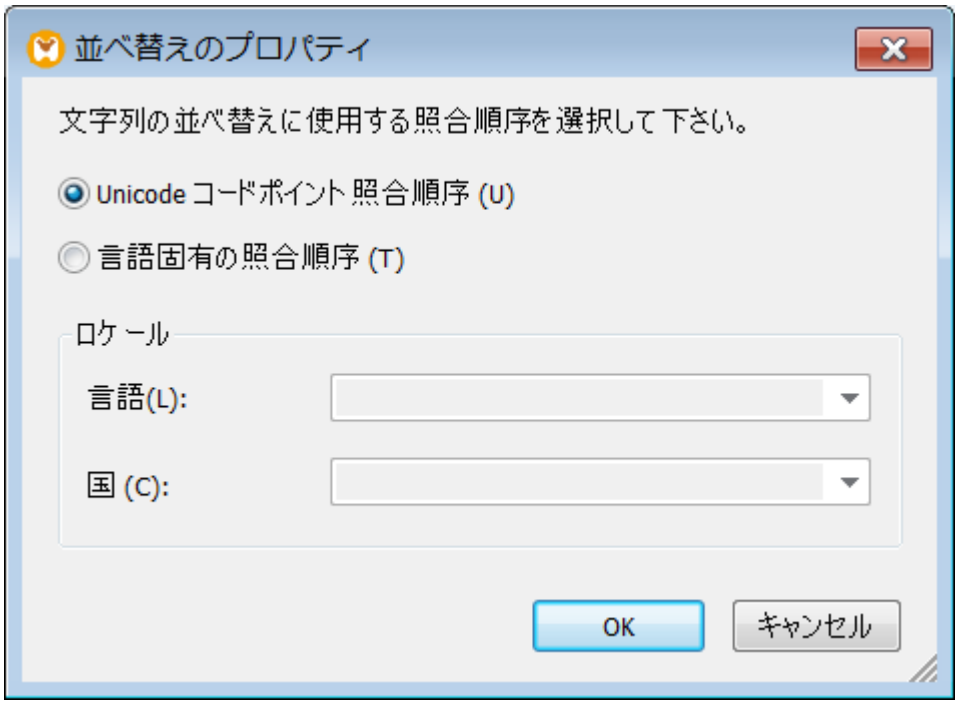
- アイテムを並べ替えコンポーネントの `nodes/rows` と `key` パラメーターに接続します。下のマッピングでは、単純型 `first` 要素が並べ替えられます。



言語特有のルールを使用して文字列を並べ替える:

- 並べ替えコンポーネントのヘッダーをダブルクリックして、「並べ替え」プロパティダイアログボックスを開く。

(C) 2015-2021 Altova GmbH

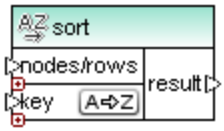


Unicode コードポイント照合: この(デフォルト)オプションにより、コードポイントの値をベースにした比較/並べ替えが行われます。コードポイント値とはUnicode コンソーシアムにより採用されたUniversal Character Set(UCS)における絶対文字に割り当てられた整数のことです。このオプションでは複数の言語やスクリプト間で並べ替えを行うことが可能になります。

言語固有の照合: このオプションにより、並べ替えを行う際にベースとなる言語や国を指定することができます。このオプションは内蔵の実行エンジン(BUILT-IN)が選択されている際にサポートされ、XSLT におけるサポートはコードの実行に使用されるエンジンに左右されます。

4.8.1　複数のキーを使用した並べ替え

マッピングに並べ替えコンポーネントを追加した後、`key` という名前の並べ替えキーがデフォルトで作成されます。

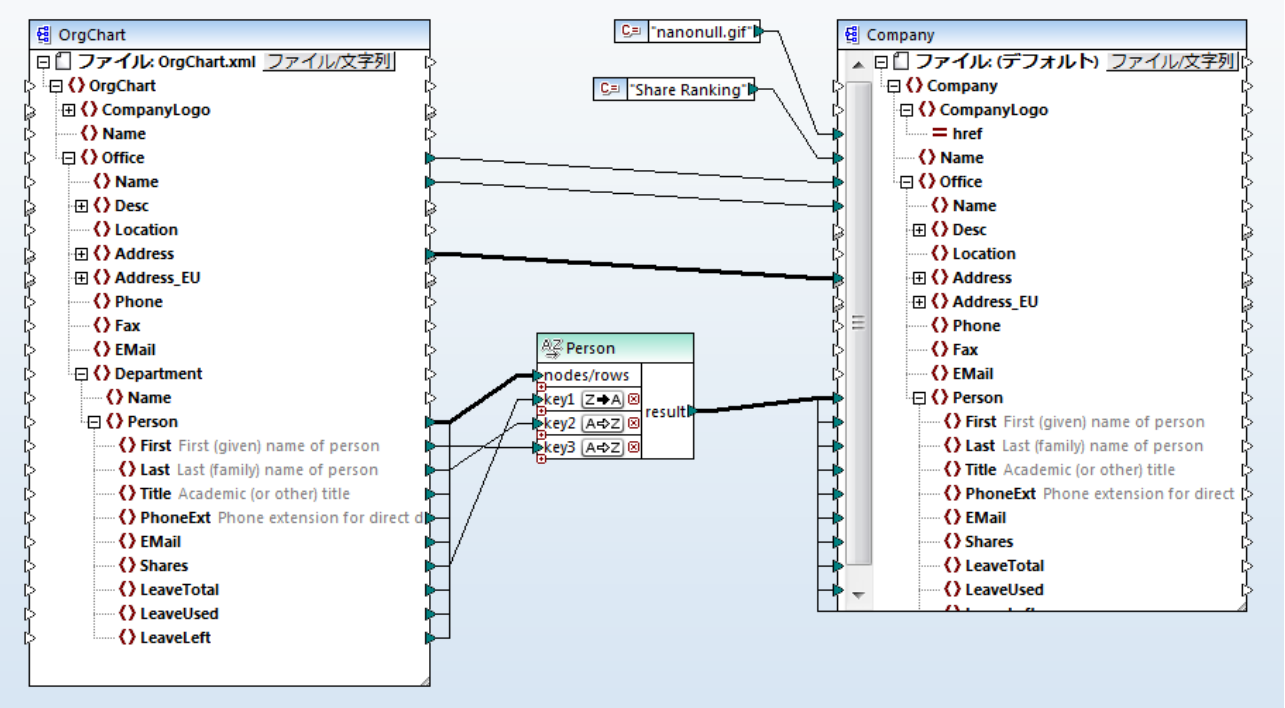


デフォルトの並べ替えコンポーネント

複数のキーを使用して並べ替える場合、並べ替えコンポーネントを以下のように調整します:

- **「キーの追加」**(⊕)アイコンをクリックして新しいキーを追加します(例えば、下のマッピング内の`key2`)。
- **「キーの削除」**(⊠)アイコンをクリックしてキーを削除します。
- ⊕アイコンに接続をドロップして、キーを追加し、接続します。

複数のキーによる並べ替えを示すマッピングは、次のパスで検索することができます: **<マイドキュメント>\Altova\MapForce2021\MapForceExamples\SortByMultipleKeys.mfd**.



SortByMultipleKeys.mfd

上のマッピングでは、`Person` レコードは3つの並べ替えキーにより並べ替えられています:

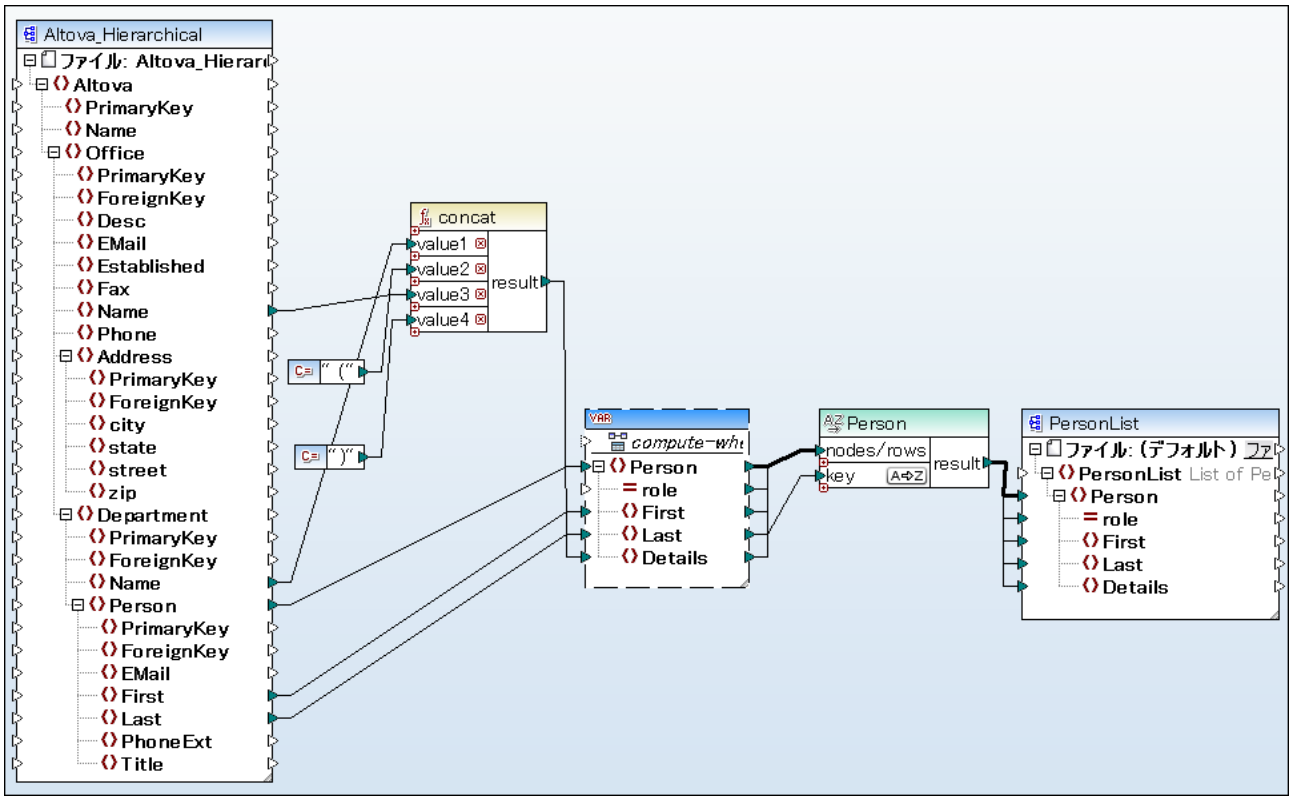
1. `Shares` (個人が所有するシェア数)
2. `Last` (姓)
3. `First` (名)

並べ替えコンポーネント内の並べ替えキーの位置は、並べ替えの優先順序を決定します。例えば、上のマッピングでは、所有するシェア数により並べ替えられています。これは、最も高い優先順位を持つ並べ替えキーです。所有するシェア数が同じの場合も同様で、姓により並べ替えられます。最後に、同数のシェアを持つ同じ性を持つ個人は、名を考慮して並べ替えられます。

各キーの並べ替えは異なる順番であることができます。上のマッピングでは、キー `Shares` は降順の並べ替え順序(Z-A)が与えられていますが、その他2つのキーには昇順の並べ替え順序 (A-Z) が与えられています。

4.8.2 変数を使用して並べ替える

希望する結果を達成するためにマッピングに中間変数を追加する必要がある場合があります。このサンプルでは、XML ファイルから記録を抽出し、中間変数を使用して並べ替える方法について説明されています。このサンプルには、次のパスで検索することのできるマッピングのサンプルが存在します: <マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥Altova_Hierarchical_Sort.mfd。



Altova_Hierarchical_Sort.mfd

このマッピングは、**Altova_Hierarchical.xml** という名前のソースXMLファイルからデータを読み取り、ターゲットXMLファイルに書き込みます。上に示されるとおり、ソースXMLには架空の企業の情報が含まれています。企業はオフィスに分岐されています。オフィスは部署に分岐されており、部署は社員に分岐されています。

ターゲットXMLコンポーネント、`PersonList`、は`Person`レコードのリストが含まれています。`Details`アイテムは、社員が属するオフィスと部署の情報が保管されています。

ソースXMLから全ての社員を抽出し、姓をアルファベット順に並べ替えることが目的です。また、オフィスと部署名は、`Details`アイテムに書き込まれる必要があります。

この目的を達成するために、このサンプルは、次のコンポーネントの型を使用します:

1. **`concat`** 関数: このマッピング内では、この関数は、フォーマット`Office(Department)`で文字列を返します。オフィス名、部署名、かっこ開始し終了する2つの定数を入力として取ります。関数をマッピングに追加するも参照してください。
2. 中間変数: 変数の役割は同じマッピングコンテキストに個人に関連する全てのデータを運ぶことです。変数によりマッピングがそれぞれの個人のコンテキストで、個人の部署とオフィスを探します。すなわち、変数は、個人が所属するオフィス名と部署名を「記憶」することを意味します。変数が不在の場合、コンテキストは、(同じXMLスキーマを使用して)正確ではなくマッピングは希望しない結果を生成します。(マッピングの実行方法の詳細に関しては、次を参照してください: マッピングルールと戦略)。変数はXMLファイルの構造を複製することに注意してください。これにより、全てコピー接続を使用して、ターゲットに並べ替えの結果を接続することができます。次も参照してください: 変数の使用 と 全てコピー接続.
3. 実際の並べ替えを行う、並べ替えコンポーネント: `Sort` コンポーネントのキー入力は、姓により個人の記録を並べ替える変数の`Last`アイテムに接続されていることに注意してください。

4.9 データのグループ分け

マッピングがソート、またはお行をグループ分けする必要がある場合、次のMapForce 内蔵関数を使用してこの目的を達成することができます:

- `group-by`
- `group-adjacent`
- `group-into-blocks`
- `group-starting-with`
- `group-ending-with`

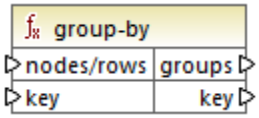
マッピング上でこれらの関数を使用する場合、ライブラリウィンドウからドラッグしてマッピングエリアにドロップします。関数をマッピングに追加するも参照してください。

メモ グループ関数は次の変換言語内で使用することができます: XSLT 2.0、XSLT 3.0、C++、C#、Java、BuiltIn。

次のセクションではグループ関数のために使用することのできる一般的なサンプルが挙げられています。これらのサンプルは次のデモマッピングが伴います: <**マイドキュメント**>¥Altova¥MapForce2021¥MapForceExamples¥Tutorial¥**GroupingFunctions.mfd**。デモマッピングには各関数のために1つずつ与えるための複数の変換が含まれています。一度にプレビュー可能な出力は1つであるため、希望する変換に適用するプレビュー ボタンを「出力」タブをクリックする前に、にクリックしてください。

group-by

`group-by` 関数は、指定するグループキーに従いレコードのグループを作成します。



例えば、下で示される抽象的な変換では、グループキーは、「Department」です。トータルで3つの一意の部署が存在するため、`group-by` を適用すると3つのグループが作成されます:

Department	Name
Administration	Vernon Callaby
Marketing	Susi Sanna
Engineering	Michelle Butler
Engineering	Fred Landis
Administration	Frank Further

group-by()

Department	Name
Administration	Vernon Callaby
Administration	Frank Further

Department	Name
Marketing	Susi Sanna

Department	Name
Engineering	Michelle Butler
Engineering	Fred Landis

詳細に関しては、group-by 関数を参照してください。

(C) 2015-2021 Altova GmbH

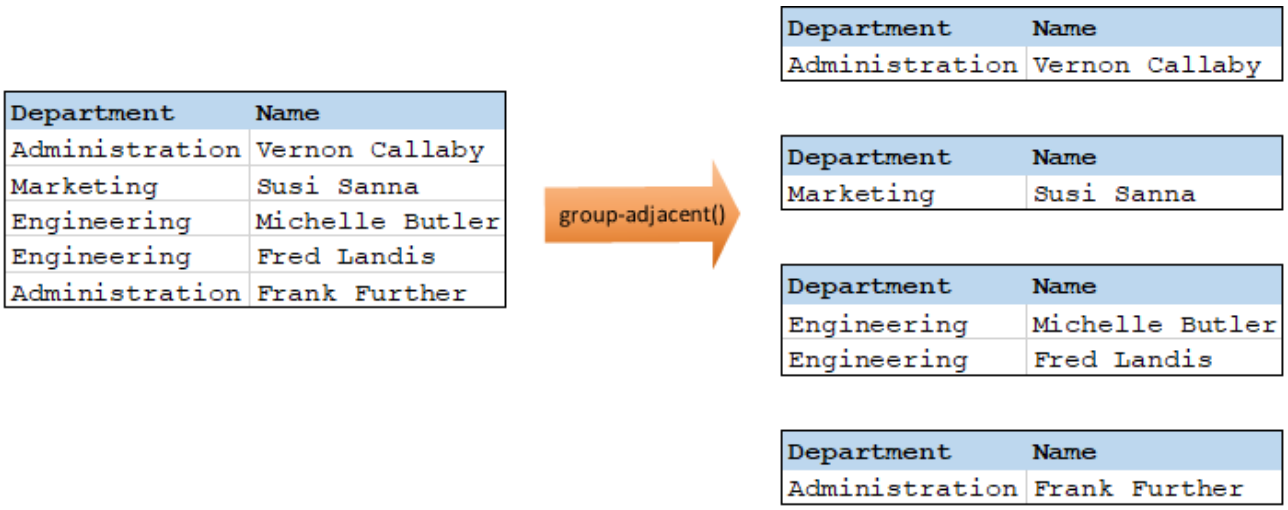
group-adjacent

group-adjacent 関数は**group-by** 関数と同様、グループキーを引数としてグループ分けすることを必要とします。**group-by** とは異なり、この関数は次のキーが異なる都度新規のグループを作成します。2つの隣接するレコードが同じキーを持つ場合、これらのアイテムは同じグループに配置されます。

例えば、下記の抽象的な変換に示されるように、グループキーは「Department」です。このダイアグラムの左側は入力データを示し、右側はグループ分けの後の出力データを示しています。次は、変換の実行中に発生します:

- 最初は、最初のキー「Administration」は新規のグループを作成します。
- 次のキーは異なるため、2番目のグループである「Marketing」が作成されます。
- 3番目のキーもまた異なるための、「Engineering」と呼ばれる他のグループが作成されます。
- 4番目のキーは3番目と同じキーです。このため、レコードは既存のグループに配置されます。
- 最後に、5番目のキーは4番目のキーと異なるため、最後のグループを作成します。

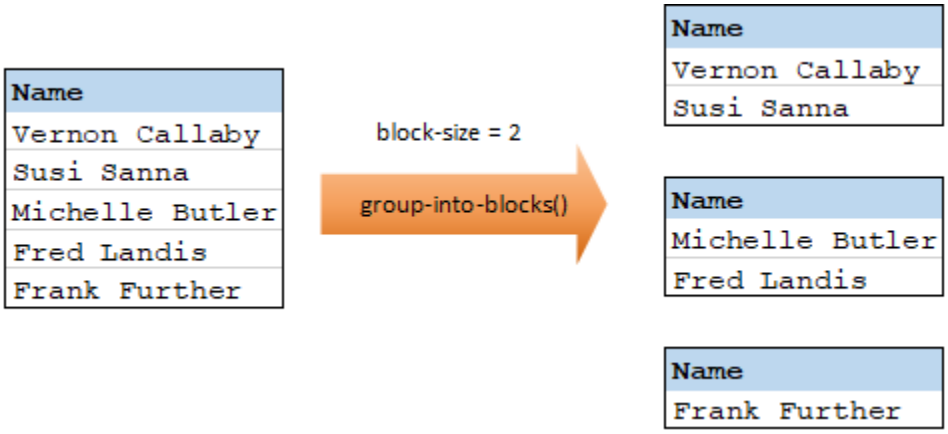
下に説明されるとおり、「Michelle Butler」と「Fred Landis」は隣接する同じキーを持つために一緒にグループ分けされています。しかしながら、「Vernon Callaby」と「Frank Further」は同じキーを所有しているにもかかわらず隣接していないため、異なるグループに分けれています。



詳細に関しては、group-adjacent 関数を参照してください。

group-into-blocks

group-into-blocks 関数は、N が`block-size` 引数に与えられる値である、N アイテムを含む等価のグループを作成します。最後のグループは、ソース内のアイテムの数に従い、N つのアイテム、または少ない数のアイテムを含むことができます。下のサンプルでは、`block-size` は2 です。総数で5つのアイテムが存在し、最後のアイテムを除き、それぞれのグループは2つのアイテムを含んでいます。



詳細に関しては、group-into-blocks 関数を参照してください。

group-starting-with

group-starting-with 関数はブール式の条件を引数として取ります。関数はブール式の条件がtrue の場合、条件を満たすレコードから、新規のグループが作成されます。下のサンプルでは、「Key」が「heading」と等価であることが条件です。最初と4番目のレコードのために、この条件がtrue の場合、2つのグループが結果として作成されます:

メモ 条件を満たす最初のレコードの前にレコードが存在する場合、追加のグループが作成されます。例えば、「line」レコードが最初の「heading」レコードよりも多く存在する場合、これらは新しいグループ内に配置されます。

詳細に関しては、group-starting-with 関数を参照してください。

group-ending-with

group-ending-with 関数はブール式の条件を引数として取ります。関数はブール式の条件がtrue の場合、条件を満たすレコードが最後になるように、新規のグループが作成されます。下のサンプルでは、「Key」が「trailing」と等価であることが条件です。3番目と5番目のレコードのために、この条件がtrue の場合、2つのグループが結果として作成されます:

(C) 2015-2021 Altova GmbH

メモ　　条件を満たす最後のレコードの前にレコードが存在する場合、追加のグループが作成されます。例えば、「line」レコードが最後の「heading」レコードよりも多く存在する場合、これらは新しいグループ内に配置されます。

詳細に関しては、group-ending-with 関数を参照してください。

4.9.1　例: キー別にレコードをグループ分けする方法

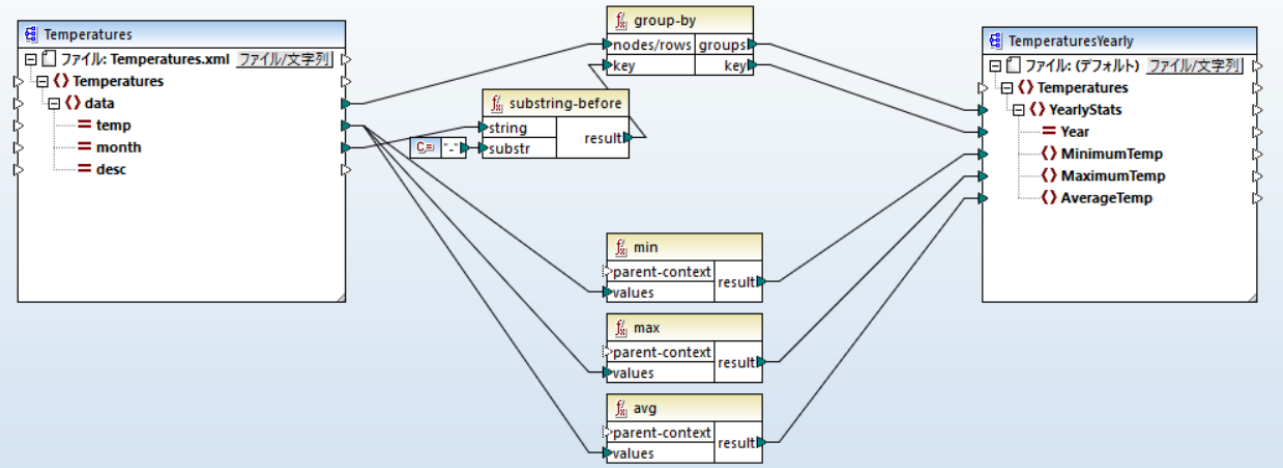
このサンプルは、**group-by** 関数の手助けと共にグループ分けする方法、および、データを集計する方法を示しています。このサンプルには次のパスで使用することのできるデモマッピングが付随しています: **<マイドキュメント>\Altova\MapForce2021\MapForceExamples\GroupTemperaturesByYear.mfd**。このマッピングは、月ごとの気温を含むXMLファイルからデータを読み取ります:

```
<Temperatures>
   <data temp="-3.6" month="2006-01" />
   <data temp="-0.7" month="2006-02" />
   <data temp="7.5" month="2006-03" />
   <data temp="12.4" month="2006-04" />
   <data temp="16.2" month="2006-05" />
   <data temp="19" month="2006-06" />
   <data temp="22.7" month="2006-07" />
   <data temp="23.2" month="2006-08" />
   <data temp="18.7" month="2006-09" />
   <data temp="11.2" month="2006-10" />
   <data temp="9.1" month="2006-11" />
   <data temp="0.8" month="2006-12" />
   <data temp="-3.2" month="2007-01" />
   <data temp="-0.3" month="2007-02" />
   <data temp="6.5" month="2007-03" />
   <data temp="10.6" month="2007-04" />
   <data temp="19" month="2007-05" />
   <data temp="20.3" month="2007-06" />
   <data temp="22.3" month="2007-07" />
   <data temp="20.7" month="2007-08" />
   <data temp="19.2" month="2007-09" />
   <data temp="12.9" month="2007-10" />
   <data temp="8.1" month="2007-11" />
   <data temp="1.9" month="2007-12" />
</Temperatures>
```

このマッピングのビジネスの条件は2段階で構成されています:

1. 各年の気温をグループ分けします。
2. 各年度の最低、最高、および平均機能を検索します。

最初の目的を達成するには、マッピングは**group-by** 関数を呼び出します。2番目のゴールを達成するには、**min**、**max** および**avg** 集計関数を呼び出します。これらすべての関数はMapForce ビルトイン関数で、ライブラリウィンドウからマッピングにドラッグして追加することができます。関数をマッピングに追加する を参照してください。



GroupTemperaturesByYear.mfd

MapForce がマッピングを実行する方法は（そして読み取りを開始するために奨励されるアプローチ）ターゲットコンポーネントの一番上のアイテムを確認することです。このサンプルでは、**YearlyStats** アイテムが**group-by** 関数により返される各グループのために作成されます。**group-by** 関数は**key** 入力に接続が作成されると、最初の引数として、すべての**data** アイテムをソースとグループから取ります。条件が気温を年度ごとにグループ分けすることであり、年度が最初に取得される必要があります。これを達成するには、**substring-before** 関数は、各 **data** 要素の**month** 属性から年度の部分を抽出します。具体的には、**month** の値を引数としてとり、の最初の**substr** の発生の前の部分に返します。上記の通り、このサンプルでは、**substr** は点線の文字のセットであるため、値「2006-01」を与えられると、関数は「2006」を返します。

最後に、**MinimumTemp**, **MaximumTemp**、と**AverageTemp** の値は、これらのアイテムを対応する集計関数に接続することにより取得することができます: **min**、**max**、と**avg**。全ての3つの関数は、ソースコンポーネントから読み取られた入力として温度のシーケンスを取ります。
これらの関数は、各グループのコンテキストと既に作業しているため**parent-context** 引数を必要としません。すなわち、作業する各集計関数のためのコンテキストを提供する**data** から**YearlyStats** への親コネクションが存在します。

マッピングの出力をプレビューするには、**Output** タブをクリックしてください。ソースファイルの読み取りで取得された年数とグループの数量が一致することに注意してください。例えば、

```
<Temperatures>
   <YearlyStats Year="2006">
      <MinimumTemp>-3.6</MinimumTemp>
      <MaximumTemp>23.2</MaximumTemp>
      <AverageTemp>11.375</AverageTemp>
   </YearlyStats>
   <YearlyStats Year="2007">
      <MinimumTemp>-3.2</MinimumTemp>
      <MaximumTemp>22.3</MaximumTemp>
      <AverageTemp>11.5</AverageTemp>
   </YearlyStats>
</Temperatures>
```


(C) 2015-2021 Altova GmbH

メモ　　簡単化のために、上記のコードリストには、デモマッピングで使用されている実際の入力と出力より少ないデータを含んでいます。

4.10 フィルターと条件

データをフィルターする場合、または値を条件に対して取得する場合、以下のうち1つのコンポーネント型を使用することができます:

- フィルター: Nodes/Rows ( )
- SQL WHERE/ORDER ( )
- If-Else 条件 ( )

「**挿入**」メニューから、または「**コンポーネントの挿入**」ツールバーからこれらのコンポーネントをマッピングに追加することができます。上記のそれぞれのコンポーネントには特定の振る舞いと必要条件があることにご注意してください。差異は下のセクションで説明されています。

ノードと行をフィルターする

XML ノード または CSV 行を含むデータをフィルターする場合、**Nodes/Rows をフィルター** コンポーネントを使用します。この **Nodes/Rows をフィルター** コンポーネントにより、true または false 条件をベースにし、大きなデータセットからノードのサブセットを抽出することができます。マッピングエリアでのこの構造は以下のように表されます:

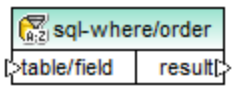
上の構造では、に接続されている条件は、**bool** に接続された **node/row** が、**on-true** または **on-false** 出力に接続されているかを決定します。具体的には、条件が true の場合、**node/row** は、**on-true** 出力にリダイレクトされます。一方、条件が false の場合、**node/row** は、**on-false** 出力にリダイレクトされます。

マッピングがフィルターの条件を満たすアイテムのみを消費する必要がある場合、**on-false** 出力を未接続のままにできます。フィルターの条件を満たさないアイテムを処理する必要がある場合、このようなアイテムがリダイレクトされる、**on-false** 出力をターゲットに接続します。フィルターの条件が満たされ例外を追加する場合は、**on-false** 出力への接続は必須です（以下を参照してください: 例外の追加）。

手順を追ってのマッピングのサンプルは、例: ノードのフィルターを参照してください。.

データベースのデータをフィルター

Nodes/Rows をフィルター コンポーネントは、MapForce により、他のコンポーネント 構造からデータをフィルターすることができます。しかしながら、データベースからデータをフィルターする場合、**SQL WHERE/ORDER** コンポーネントを代わりに、使用することが奨励されます。**SQL WHERE/ORDER** コンポーネントは、データベースと作業し、**Nodes/Rows をフィルター** コンポーネントよりも良いパフォーマンスを与えるために最適化されています。



このようなコンポーネントに関する詳細に関しては、SQL WHERE／ORDER コンポーネントを参照してください。

値を条件付きで返す

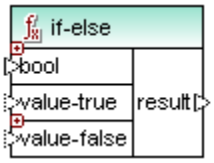
（ノードまたは行ではない）単一の値を条件付きで取得する必要がある場合、**If-Else 条件** を使用します。If-Else 条件は、ノードまたは行をフィルターするには、適していません。**Nodes/Rows をフィルター** コンポーネントとは異なり、**If-Else 条件** は、（文字列または整数などの）単純型を返します。ですから、**If-Else 条件** は、単純型を条件付きで処理するシナリオのみに適しています。例えば、各月の平均気温のリストがあると仮定します:

(C) 2015-2021 Altova GmbH

```
<Temperatures>
   <data temp="19.2" month="2010-06" />
   <data temp="22.3" month="2010-07" />
   <data temp="19.5" month="2010-08" />
   <data temp="14.2" month="2010-09" />
   <data temp="7.8" month="2010-10" />
   <data temp="6.9" month="2010-11" />
   <data temp="-1.0" month="2010-12" />
</Temperatures>
```

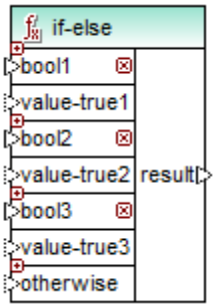
If-Else 条件 により、リスト内の各アイテムのために気温が摂氏20度を超える場合は、値 "高" を、気温が摂氏5度以下の場合は、値 "低" を返すように設定することができます。

マッピング上で If-Else 条件 の構造は、以下のようになります:



bool に接続された条件がtrue の場合、**value-true** に接続された **result** としての出力になり、条件がfalse の場合、**value-false** に接続された **result** としての出力になります。**result** のデータ型を事前に知ることはできず、これは 値のデータ型、により異なり、**value-true** または **value-false**. に接続された値により異なります。重要な点は、常に(文字列および整数などの)。単純型であるということです。 入力値を(ノードまたは行などの)複合型に接続することは If-Else 条件によりサポートされていません。

If-Else 条件は拡張することが可能です。これは、**追加** (⊕)ボタンをクリックすることにより、コンポーネントに複数の条件を追加できることを意味します。以前に追加された条件を削除するには、ボタンを**削除** (⊗) をクリックします。この機能により、条件がtrue の場合は、複数の条件をチェックして、各条件のために異なる値を返すことができます。



展開された If-Else 条件 は上から下へと評価されます(最初の条件が最初にチェックされ、2番目が次にチェックされます)。条件の全てが真ではない場合に値を返す場合は、**otherwise** に接続してください。

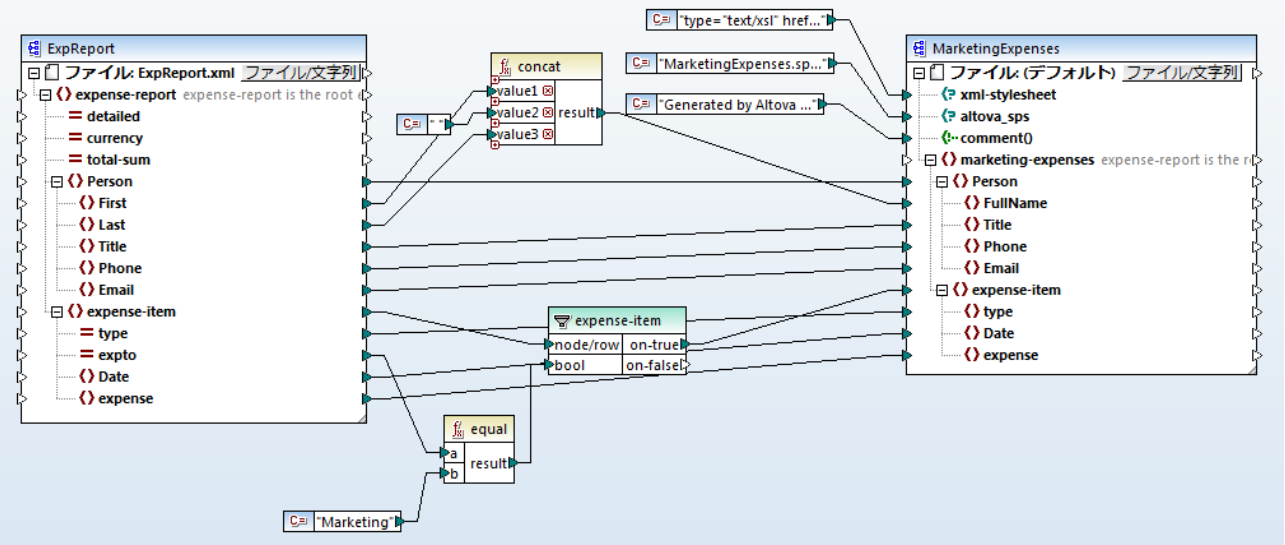
手順を追ってのマッピングのサンプルは、例: 値を条件付きで返すを参照してください。

4.10.1 例: ノードのフィルター

この例は、true/false 条件を基にしてノードをフィルターする方法について説明しています。**Filter: Nodes/Rows** ( ) コンポーネント がこの目的を達成するために使用されています。この例で説明されている技術は、XML のみではなく、CSV または テキストなどの他のコンポーネント型などとでも作業することができます。データベースの場合、フィルターを使用することはできますが、パフォーマンスのためには、代

わりにSQL WHERE/ORDER コンポーネント を使用することが奨励されます（以下を参照してください: SQL WHERE / ORDER コンポーネント）。

この例で説明されているマッピングは以下のパスで検索することができます: **<マイドキュメント>\Altova\MapForce2021\MapForceExamples\MarketingExpenses.mfd**。



上に示されるように、このマッピングは、経費の方向k（"ExpReport"）のデータを含むソースXML からデータを読み取り、ターゲット XML（"MarketingExpenses"）に書き込みます。ターゲットとソースの間には複数の他のコンポーネントが存在します。最も関連性の高いコンポーネントは、このトピックの主題を表す **expense-item** フィルター（ ）です。

このマッピングのゴールは、マーケティング部署に属する経費アイテムのみをフィルターすることです。 この目的を達成するために、フィルターコンポーネント がマッピングに追加されました。（フィルターを追加するには、「挿入」メニューをクリックして、「Filter: Nodes/Rows」をクリックします）。

各経費がマーケティング部署に属するかを識別するために、このマッピングは、ソース内の"expto" 属性の値を確認します。この属性は経費がマーケティングの経費である場合、値 "Marketing" を有します。例えば、 下にリストされるコードでは、最初と3番目のアイテムは、マーケティングに属し、2番目のアイテムは開発部署に、4番目は販売部署に属することが示されています:

```
...
   <expense-item type="Meal" expto="Marketing">
      <Date>2003-01-01</Date>
      <expense>122.11</expense>
   </expense-item>
   <expense-item type="Lodging" expto="Development">
      <Date>2003-01-02</Date>
      <expense>122.12</expense>
   </expense-item>
   <expense-item type="Lodging" expto="Marketing">
      <Date>2003-01-02</Date>
      <expense>299.45</expense>
   </expense-item>
   <expense-item type="Entertainment" expto="Sales">
      <Date>2003-01-02</Date>
      <expense>13.22</expense>
   </expense-item>
...
```

マッピングが実行される前のXML入力

マッピングエリアで、フィルターの**node/row** 入力が、ソースコンポーネント内の**expense-item** ノードに接続されています。これにより、フィルターコンポーネントが処理されなければならないノードのリストを取得することを保証します。

フィルタリングが発生する条件を追加するには、MapFlorce core ライブラリから`equal` 関数 を追加しました（詳細に関しては、以下を参照してください：関数をマッピングに追加する）。`equal` 関数は、"type" 属性を値 "Marketing" を持つ定数と比較します（定数を追加するには、**「挿入」**メニューをクリックして、**定数**をクリックします）。

条件を満たすアイテムのみをフィルターする必要があるため、フィルターの**on-true** 出力のみをターゲットコンポーネントに接続します。

マッピングの結果をプレビューする準備が整うと、**「出力」**タブをクリックします。MapForce は、フィルターの**bool** 入力に接続されている条件である、それぞれの経費アイテムノードを評価します。条件がtrue の場合、経費アイテムノードはターゲットにパスされます。それ以外の場合は、無視されます。結果、条件を満たす経費アイテムのみが出力内に表示されます:

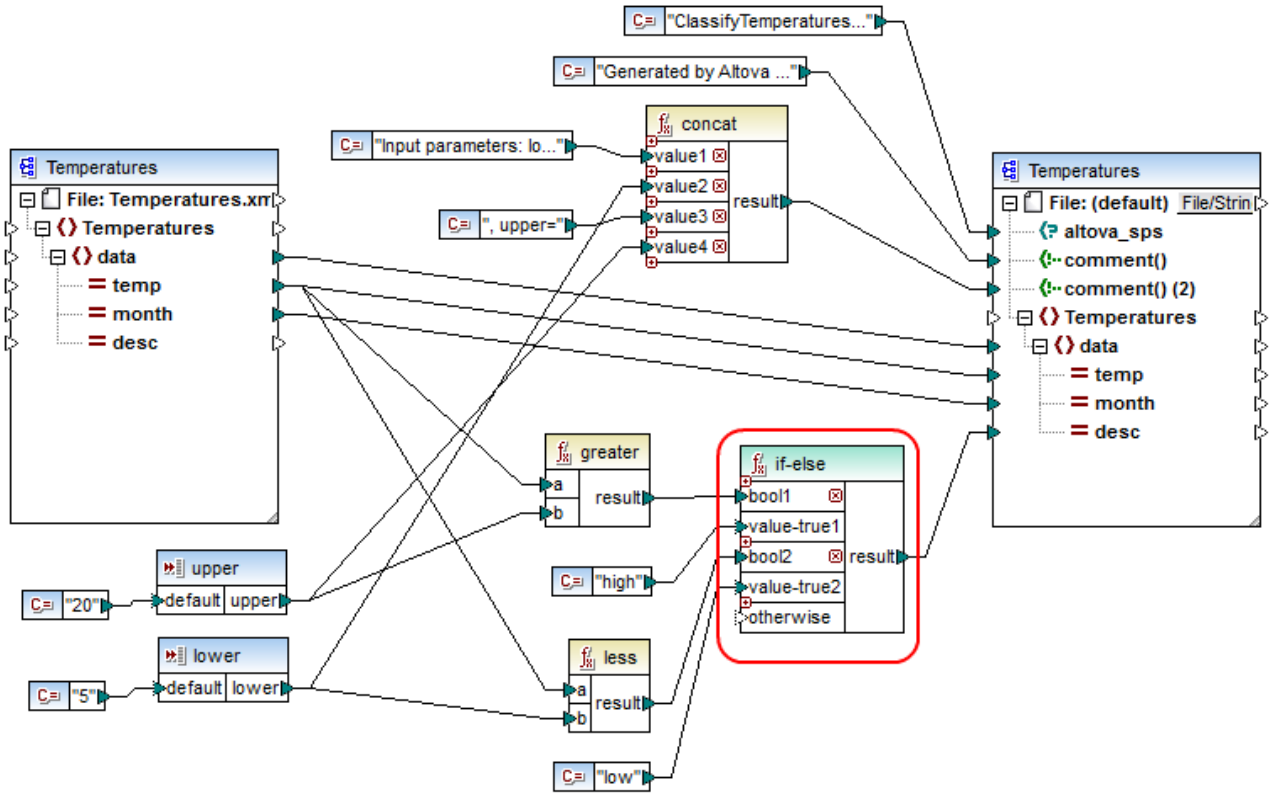
```
...
   <expense-item>
      <type>Meal</type>
      <Date>2003-01-01</Date>
      <expense>122.11</expense>
   </expense-item>
   <expense-item>
      <type>Lodging</type>
      <Date>2003-01-02</Date>
      <expense>299.45</expense>
   </expense-item>
...
```

マッピングの実行後のXML出力

4.10.2　例: 条件付で値を返す

この例では、true/false 条件を基にして、コンポーネントから単純型の値を返す方法を説明しています。**「If-Else 条件」**（ ）がこの目的を達成するために使用されています。**If-Else 条件** とフィルターコンポーネントを混同しないように注意してください。**「If-Else 条件」**は、（文字列、整数などの）単純な値を条件付きで処理する場合のみに適しています。ノードなどの複合値をフィルターする場合、代わりにフィルターを使用してください（以下を参照してください：例: ノードのフィルター）。

この例で説明されるマッピングは以下のパスで検索することができます: **<マイドキュメント>\Altova\MapForce2021\MapForceExamples\ClassifyTemperatures.mfd**。



このマッピングは、気温データを含むソースXML（"Temperatures"）からデータを読み取り、データを同じスキーマに準ずるターゲットXMLに書き込みます。ターゲットとソースの間には複数の他のコンポーネントが存在します。そのうちの1つが、このトピックの主題である（赤でハイライトされている）**if-else** 条件です。

このマッピングのゴールは、ターゲット内の各気温の記録に短い説明を追加することです。具体的には、摂氏20度以上の気温の説明は、"high" と表示され、摂氏5度以下の気温は、"low" と表示されるように設定します。上記以外の場合は、説明が表示されません。

この目的を達成するために、条件付き処理が必要とされます。ですから、If-Else 条件 がマッピングに追加されました。（If-Else 条件を追加するには、**「挿入」** メニューをクリックして、**If-Else 条件**をクリックします）。このマッピングでは、以下の2つの条件を受け入れるために（ボタンを使用して）If-Else 条件が展開されています: **bool1** と**bool2**。

条件自身は、MapForce core ライブラリから追加された`greater` と`less` 関数により追加されます。詳細に関しては、関数をマッピングに追加するを参照してください。関数は、"upper" と"lower" と呼ばれる2つの入力コンポーネントより与えられた値を評価します（入力コンポーネントを追加するには、**「挿入」** メニューをクリックして、**「入力の挿入」**をクリックします。入力コンポーネントに関する詳細については、マッピングにパラメーターを提供するを参照してください

`greater` と`less` 関数は、true または false のどちらかを返します。関数の結果は、ターゲットインスタンスに何が書き込まれるかを決定します。具体的には、ソース内の"temp" 属性の値が、20 よりも大きい場合、定数値 "high" が**if-else** 条件にパスされます。ソース内の"temp" 属性の値が5 よりも小さい場合、定数値 "low" は、**if-else** 条件にパスされます。**otherwise** 入力が接続されていませんので、上記の条件が満たされない場合は、**result** 出力コネクターにパスされる値はありません。

最後に、**result** 出力コネクタは、（各気温の記録に1度ずつ与えられる）この値をターゲット内の"desc" 属性に与えます。
マッピングの結果をプレビューする準備が整うと、**「出力」** タブをクリックします。気温が20度より高い、または5度より低い場合、結果 XML 出力は、"desc" 属性を含むことに注意してください。

```
...
```


(C) 2015-2021 Altova GmbH

```
   <data temp="-3.6" month="2006-01" desc="low"/>
   <data temp="-0.7" month="2006-02" desc="low"/>
   <data temp="7.5" month="2006-03"/>
   <data temp="12.4" month="2006-04"/>
   <data temp="16.2" month="2006-05"/>
   <data temp="19" month="2006-06"/>
   <data temp="22.7" month="2006-07" desc="high"/>
   <data temp="23.2" month="2006-08" desc="high"/>
...
```

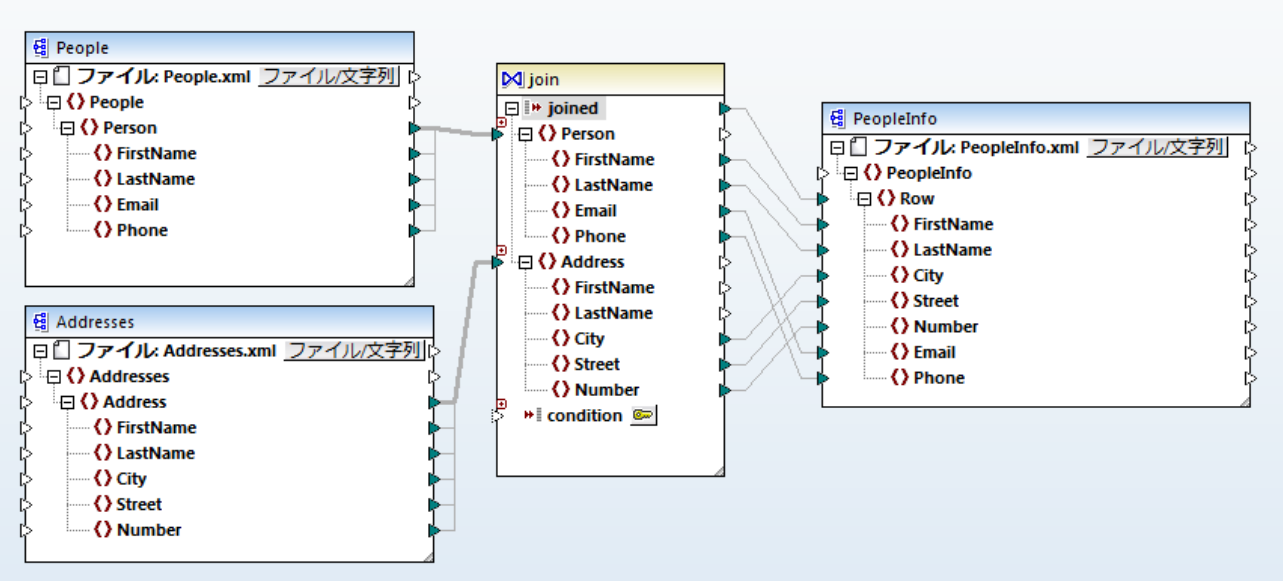
マッピングの実行後の XML 出力

4.11 データのジョイン

(例えば、最初の構造内のフィールド A が2 番目の構造内のフィールド Bと同じ値を持つなど)の条件をベースに2つの構造からのデータを結合する必要がある場合があります。このようなマッピングの条件には、ジョインコンポーネントを使用することができます。

ジョインコンポーネントは、カスタムに定義付けられた条件をベースにして2つの構造をジョインすることを有効化するMapForce コンポーネントです。条件を満たすアイテムのアソシエーション(ジョインされたセット)を返します。ジョインは、(ID などの)共通のフィールドを共有する2つの構造からのデータを結合する際にとても役に立ちます。

例えば、下のマッピングでは、中間コンポーネントは「ジョイン」コンポーネントです。このマッピングでは、2つのXML 構造(個人のリストと住所のリスト)がジョインされます。このサンプルの目的は、各個人の情報をターゲット XML ファイルに書き込むことです。 `FirstName` と `LastName` フィールドは、ジョインするキーとしての役割を果たします。すなわち、(`Person` の下の) `FirstName` と`LastName` の値が(`Address` の下の) `FirstName` と`LastName` と同じ値の場合、同じ個人に属する住所の詳細は「ジョイン」されます。ジョインした構造からのアイテムは、次のターゲット(この場合、 XML ファイル)にマップすることができます。ジョインの条件は、ジョインコンポーネント プロパティ内で「**ジョインの条件の定義**」( ) ボタンをクリックして定義されます。このサンプルにはマッピングのサンプルが存在し、例: XML 構造のジョイン内で詳しく説明されています。



JoinPeopleInfo.mfd

上記のとおり、ソース構造 とジョインコンポーネントは、マッピングを優先順位の低い「全てコピー」接続を使用して接続されています。一般的には、コンテキストが関連する場合、このような接続は自動的にMapForce により作成されます(詳細に関しては、次を参照してください: 全てコピー接続)。

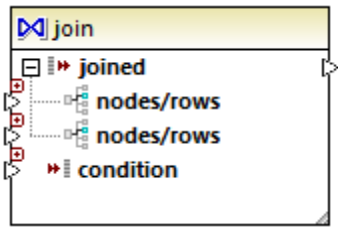
(上のマッピングのような)個別のコンポーネント、または同じコンポーネントに属するコンポーネントから2つの構造をジョインすることができます。(例えば、 XML 構造とデータベーステーブルなどの)異なる型 の構造をジョインすることもできます。データベースに関連したジョインの詳細に関しては、次を参照してください: データベースのデータのジョイン(SQL JOIN)。

ジョインコンポーネントを追加する方法:

1. マッピング 変換言語をBUILT-IN に設定します(これを行うには ツールバーボタンをクリックします、または[出力 | ビルトイン実行エンジン] メニューコマンドを使用します)。

(C) 2015-2021 Altova GmbH

2. [挿入] メニューから [ジョイン] をクリックします。または [ジョイン] ツールバーボタンをクリックします。ジョインコンポーネントがマッピングに表示されます。デフォルトでは2つの構造からデータを受け入れるため、2つの `nodes/rows` 入力が存在します。必要であれば [入力の追加] ( ) ボタンをクリックして、ジョインに新しい入力を追加することができます。3つ以上の構造をジョインするを参照してください。



3. ジョインコンポーネントの `nodes/rows` アイテムとジョインされる構造を選択します。
4. ジョイン（または複数の条件）のための条件を追加してください。これを行うには、ジョインコンポーネントを右クリックし [プロパティ] を選択します。ジョインの条件は、関数のブール値の結果をジョインコンポーネントの `condition` アイテムに接続することにより、マッピングから直接追加することができます。データベースがジョインされる特定の場合、ジョインの条件は自動的にMapForce により作成されます。詳細に関しては、ジョインの条件の追加を参照してください。

メモ

- ジョインコンポーネントは、BUILT-IN 実行エンジン内でのみサポートされています。C#、C++、またはJava 内でのコードの生成はサポートされていません。
- 構造がサポートされていない、またはジョインのための入力ソースによりサポートされて以内場合、MapForce マッピング上直接ヒントを表示するか、またはマッピングを検証する際に、メッセージウィンドウ内にヒントを表示します。（マッピングの検証参を照してください）。
- ジョインコンポーネントをインラインユーザー定義関数の結果または入力パラメーターに接続しないでください。このような接続が存在すると、マッピングの検証中に検証エラーが発生します。
- 使用することのできるデータベースコンポーネント（テーブルまたはビューなどの）を直接ジョインコンポーネントに接続すると、SQL モード SQL ボタンが自動的にジョインコンポーネントの右上に表示されます。有効化されると、このボタンは、ジョイン操作に適用することのできる特別な SQL 機能を提供します（ SQL モード内のジョインを参照してください）。
- ジョインされたアイテムの出力を他のジョインコンポーネントに接続することはできません。しかしながら、ジョインの結果の一部を他のジョインに接続することは可能です。

他のコンポーネント型と比較したジョインコンポーネント

一部の場合、同じ結果を得るために、ジョインコンポーネントの代わりに複合型変数、またはフィルターを使用することができます。次を参照してください：変数の使用 とフィルターと条件）。しかしながら、他のコンポーネント型とは異なり、ジョインコンポーネントは、同じコンテキストに与えられるデータを確認することができるため、マッピングのロジックをわかり易くします。また、SQL モード がジョインコンポーネント上で有効化されている場合、マッピングのパフォーマンスが大幅に向上します（これはデータベースジョインにも適用されます、次を参照してください：データベーステーブルのジョイン）。

親コンテキストの追加

特別な場合は、特定のマッピングの結果を達成するために、ジョインコンポーネントに接続されているデータのために いわゆる「親コンテキスト」と呼ばれる）マッピングのコンテキストを明示的に提供することができます。親コンテキストを追加するには、ジョインコンポーネントのジョインされたアイテムをクリックし、コンテキストメニューから「**親コンテキストの追加**」を選択します。ジョインコンポーネントは、必要とするソースアイテムに接続することのできる追加 `parent-context` 入力を含む外観に変更します。詳細に関しては、以下を参照してください 例: 親コンテキストの変更。

4.11.1 ジョインの条件の追加

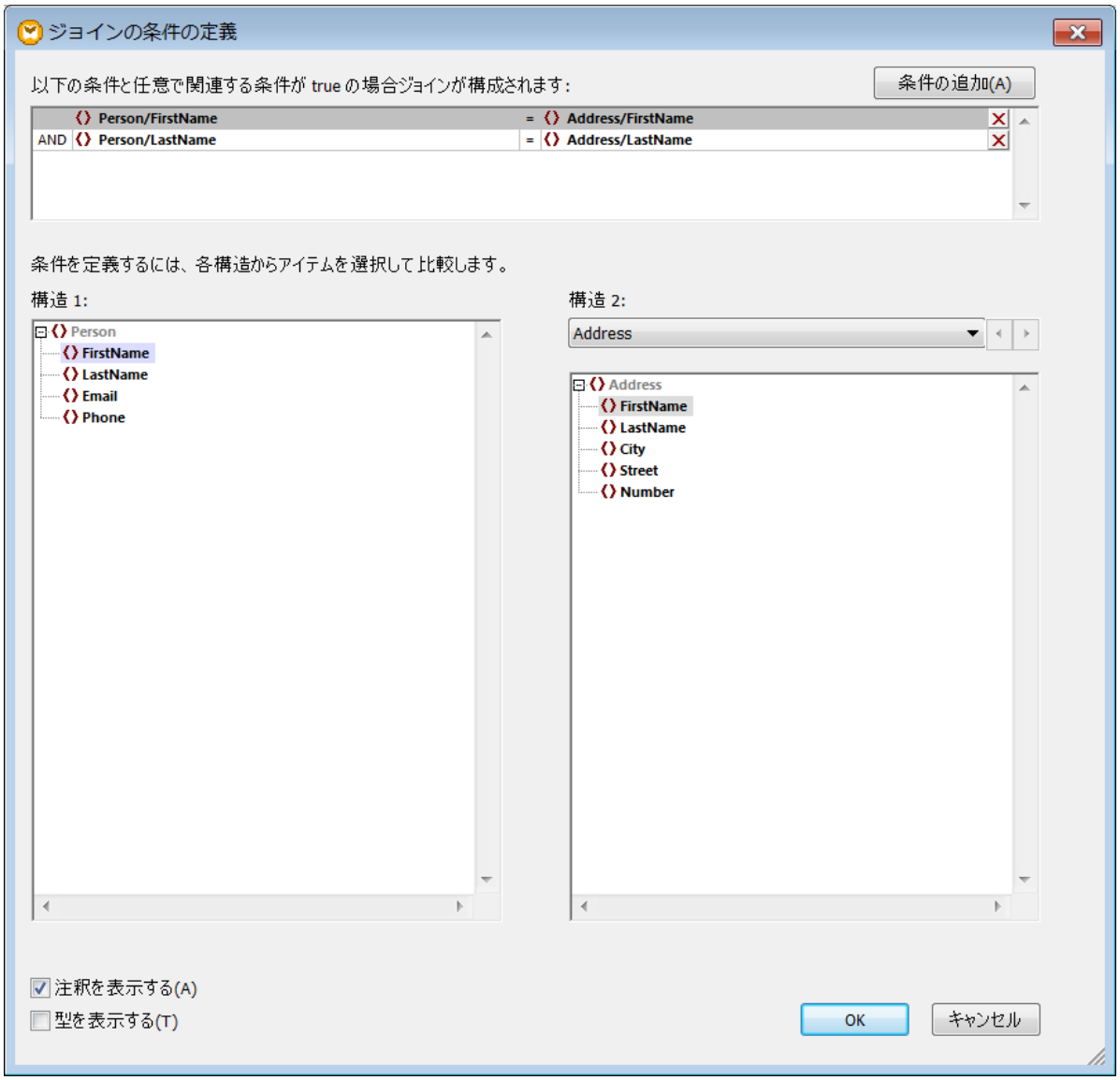
条件に従い2つの構造のアイテムを組み合わせることによりジョインは作動します。このため、ジョインには常に条件が必要になります。下に示されるようにジョインの条件を追加するには複数の方法があります。

メモ SQL モード内でデータベーステーブルがジョインされると、MapForce は、テーブル間で検出された外部キーリレーションシップをベースにして、ジョインの条件を自動的に作成します。自動的なジョインの条件が起こるには、データベーステーブルが MapForce コンポーネント上で親子関係内に存在する必要があります（すなわち、1つのテーブルが「親」または「子」で、他のテーブルがコンポーネント上に存在する必要があります）。例: SQL モード内のテーブルのジョインを参照してください。

アプローチ 1: コンポーネント プロパティからジョインの条件を追加する

1. マッピング上で、2つ以上の構造（または、データベーステーブル）がジョインコンポーネントに接続されている必要があります。このサンプルで説明されているジョインコンポーネントは、フォルダー <**マイドキュメント>\Altova\MapForce2021\MapForceExamples\Tutorial** 内で使用することのできる **JoinPeopleInfo.mfd** マッピングの一部です。詳細に関しては、例: XML 構造のジョインを参照してください。
2. ジョインコンポーネントで、「**ジョインの条件の定義**」（ ）ボタンをクリックします（または、コンポーネントのヘッダーを右クリックし、コンテキストメニューから **プロパティ** を選択します）。
3. 左の構造からアイテムを選択し、その他を右の構造から選択します（これは、このペアの比較が true を返すと、左と右の構造がジョインされます）。

(C) 2015-2021 Altova GmbH



複数の条件を追加する必要がある場合、をクリックして、「**条件の追加**」新しいアイテムのペアを選択します。例えば、上のイメージは内では、2つのジョインの条件が定義されています:

1. 構造 1 内の`FirstName` は構造 2 内の`FirstName` と等価である必要があります。
2. 構造 1 内の`LastName` は、構造 2 内の`LastName` と等価である必要があります。

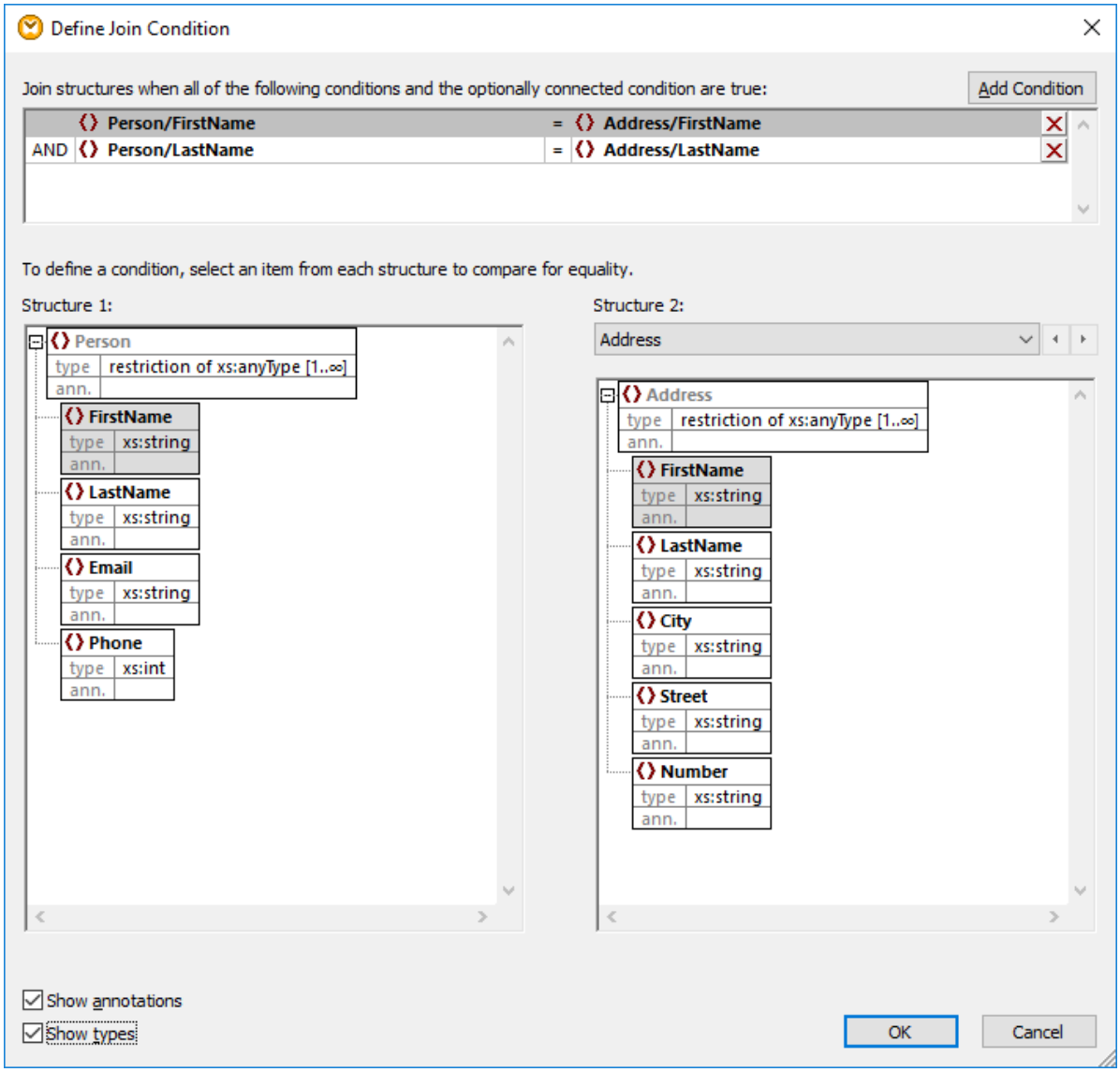
ジョインの条件を削除するには、横の「**削除**」ボタンをクリックします。

メモ

- 複数のジョインの条件が存在する場合、2つの構造がジョインされるためには、全ての条件が満たされている必要があります。すなわち、複数の条件は理論的な AND オペレーションによりジョインされていることを意味します。また、マッピングから追加された任意の条件も含んでいます（次を参照してください：下のアプローチ 2 ）。
- 2つ以上の構造がジョインコンポーネントに接続されている場合、これらの追加構造は、「構造 2」の下のドロップダウンリストに表示されます。このような追加構造をドロップダウンリストから選択すると、左側のペインに、ジョインコンポーネント前の全ての構造が表示

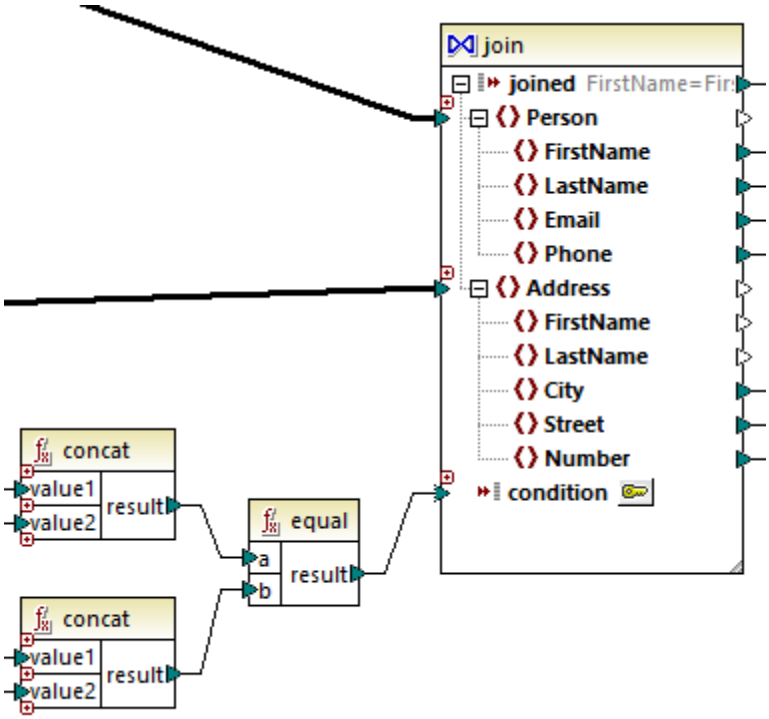
されます。この方法で、任意の複数の構造間のジョインの条件を定義することができます。例に関しては、以下を参照してください 例: CSV レポートを複数のテーブルから作成する。

- 各構造内のアイテムのデータ型をビューするためには、「**型の表示**」チェックボックスを選択します。基になるスキーマ（またはデータベース）内に情報が存在することが前提で、「**注釈の表示**」オプションは、アイテムの追加情報を表示します。両方のチェックボックスが選択されていると、レイアウトは、注釈と型の両方を表示するために調整されます。例:



アプローチ 2: マッピングからジョインの条件を追加する

- マッピング上で、ブール値を生成するコンポーネントを追加し、`condition` アイテムの入力にブールの出力を接続します。例えば、`equal` 関数は、一部のマッピングアイテムの値を比較し、ジョインコンポーネントの`condition` アイテムの入力として、ブール値の結果を提供します。



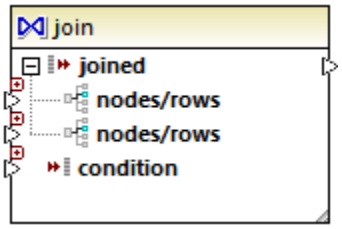
メモ　ジョインコンポーネント プロパティ（アプローチ 1）で条件が定義されていない場合、ジョインコンポーネントの `condition` アイテムが接続されている必要があります（アプローチ 2）。

アプローチ 3: 複合型のアプローチ

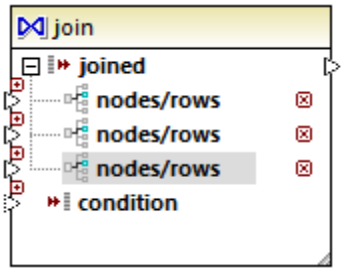
同じマッピング内で、 コンポーネント プロパティ（アプローチ 1）内のジョインの条件を定義し、マッピング（アプローチ 2）からの 1 つと結合することができます。しかしながら、SQL モードを使用して、データベーステーブルをジョインする場合は、条件はアプローチ 1 を使用して定義される必要があります。SQL モード内のジョインを参照してください。

4.11.2　3つ以上の構造をジョインする

メニューコマンド「**挿入 | ジョイン**」を使用して、ジョイン コンポーネントをマッピングに追加する場合、2 つの構造をデフォルトで受け入れることができます（すなわち、コンポーネントには 2 つの `nodes/rows` 入力が含まれています）。



2 つの構造をジョインする場合、 **入力を追加**（ ⊕ ）ボタンをクリックし、必要な数だけ `nodes/rows` を作成します。`nodes/rows` 入力を削除する場合、 **入力を削除**（ ⊠ ）ボタンをクリックします。ジョインは少なくとも 2 つの構造を必要とします、ですから ⊠ ボタンは、2 つの入力が存在する場合のみ使用することができます。



ジョインに複数の入力が存在する場合、ジョインの条件は、ジョインする各入力を考慮する必要があります。以下を参照してください ジョインの条件を追加する。複数のデータベーステーブルをジョインするサンプルのステップバイステップの例に関しては、以下を参照してください 例: CSV レポートを複数のテーブルから作成する。

4.11.3 例: XML 構造のジョイン

このサンプルは、ジョインコンポーネントを使用して、2つのXML 構造からのデータを結合する方法について説明しています。このサンプルは、次のパスにあるマッピングサンプルが存在します: <マイドキュメント>**\Altova\MapForce2021\MapForceExamples\Tutorial\JoinPeopleInfo.mfd**.

このマッピングの目的は、2つのソースXML ファイルから個人の情報（名、姓、住所、電子メール、と電話番号）を収集し、単一のターゲット XML ファイルに書き込むことです。

最初のXML ファイルは、各個人の名と姓、電子メールと電話番号を、下のサンプルコードで示されているとおり保管します（ XML 宣言、名前空間、とレコードの一部は簡素化のために省略されています):

```
<People>
   <Person>
      <FirstName>Marquita</FirstName>
      <LastName>Bailey</LastName>
      <Email>m.bailey@nanonull.com</Email>
      <Phone>555323698</Phone>
   </Person>
   <Person>
      <FirstName>Totie</FirstName>
      <LastName>Rea</LastName>
      <Email>t.rea@nanonull.com</Email>
      <Phone>555598653</Phone>
   </Person>
</People>
```

People.xml

2 番目のXML ファイルは、各個人の名と姓、および電子メールアドレスの詳細を保管します。:

```
<Addresses>
   <Address>
      <FirstName>Marquita</FirstName>
      <LastName>Bailey</LastName>
      <City>Bridgedell</City>
      <Street>Olive Street</Street>
      <Number>4</Number>
```

```
   </Address>
   <Address>
      <FirstName>Totie</FirstName>
      <LastName>Rea</LastName>
      <City>Roseford</City>
      <Street>Evergreen Lane</Street>
      <Number>34</Number>
   </Address>
</Addresses>
```

Addresses.xml

マッピングの目的は、<Address> 情報を持つ最初のファイルからの<Person> 情報と2 番目のファイルからの情報を姓が一致するか比較します。特に、 最初のファイル内の各 <Person> と 2 番目のファイル内の各 <Address> で、 FirstName とLastName が比較されます。両方の値が同じの場合、対応する<Person> と<Address> レコードが同じ個人を参照している場合、ジョインが実行されます。ターゲット XML 構造 は以下のようになります:

```
<PeopleInfo>
   <Row>
      <FirstName>Marquita</FirstName>
      <LastName>Bailey</LastName>
      <City>Bridgedell</City>
      <Street>Olive Street</Street>
      <Number>4</Number>
      <Email>m.bailey@nanonull.com</Email>
      <Phone>555323698</Phone>
   </Row>
   <Row>
      <FirstName>Totie</FirstName>
      <LastName>Rea</LastName>
      <City>Roseford</City>
      <Street>Evergreen Lane</Street>
      <Number>34</Number>
      <Email>t.rea@nanonull.com</Email>
      <Phone>555598653</Phone>
   </Row>
</PeopleInfo>
```

PeopleInfo.xml

このマッピングの目的は、ジョインコンポーネントをマッピングに追加することで簡単に達成することができます。他のコンポーネント型を使用して、同じ結果を達成することもできることに注意してください。しかしながら、下のステップでは、このサンプルの主題である、ジョインコンポーネントを使用してサンプルが説明されています。

必要とされるマッピングを作成するには、以下のステップに従います。

ステップ 1: ソース XML ファイルをマッピングに追加する

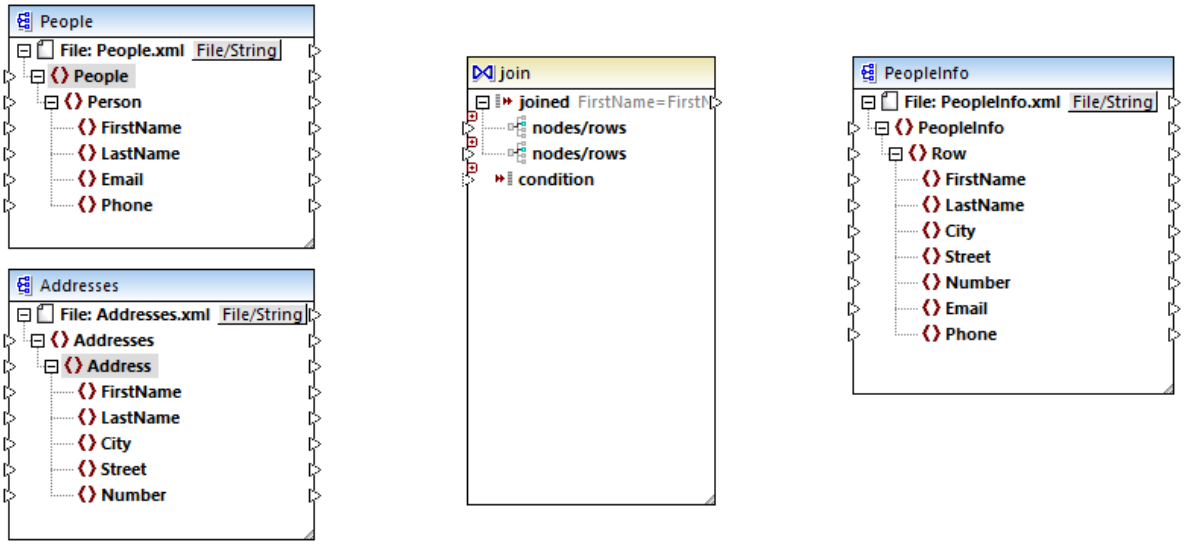
1. 「挿入」メニューからXML スキーマ/ファイル をクリックします。次のソースファイルを参照します: <**マイドキュメント >\Altova\MapForce2021\MapForceExamples\Tutorial\People.xml**。
2. **Addresses.xml** (2 番目のソースファイル) のために上のステップを繰り返します。

ステップ 2: ターゲット スキーマファイルをマッピングに追加する

- 「挿入」メニューから XML スキーマ/ファイル をクリックします。<マイ ドキュメント>**\Altova\MapForce2021\MapForceExamples\Tutorial\PeopleInfo.xsd** (ターゲット XSD スキーマファイル) を参照しますXML ファイルを与えるようにプロンプトされると、「スキップ」をクリックします。ルート要素を選択するようにプロンプトされると、ルート要素として `PeopleInfo` を選択します。

ステップ 3: ジョインコンポーネントを追加する

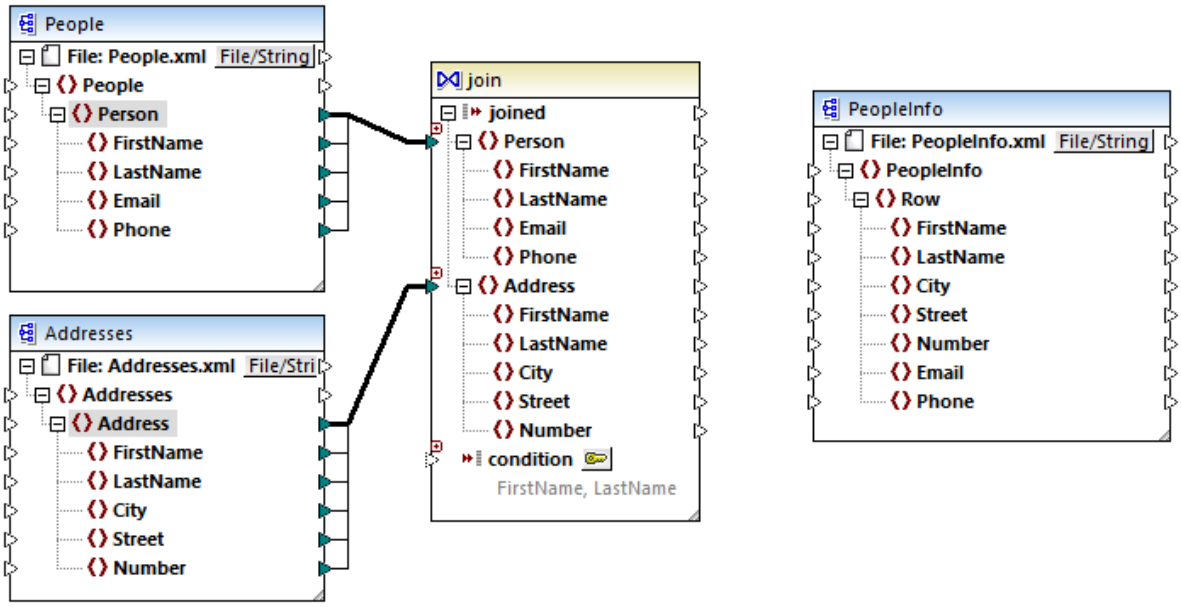
1. 「挿入」メニューから「ジョイン」(または、「ジョイン」 ツールバーボタン) をクリックします。この時点でマッピングは以下のようになります (下のマッピングと同じ表示にするには、コンポーネントをドラッグしてサイズを調整する必要があります):



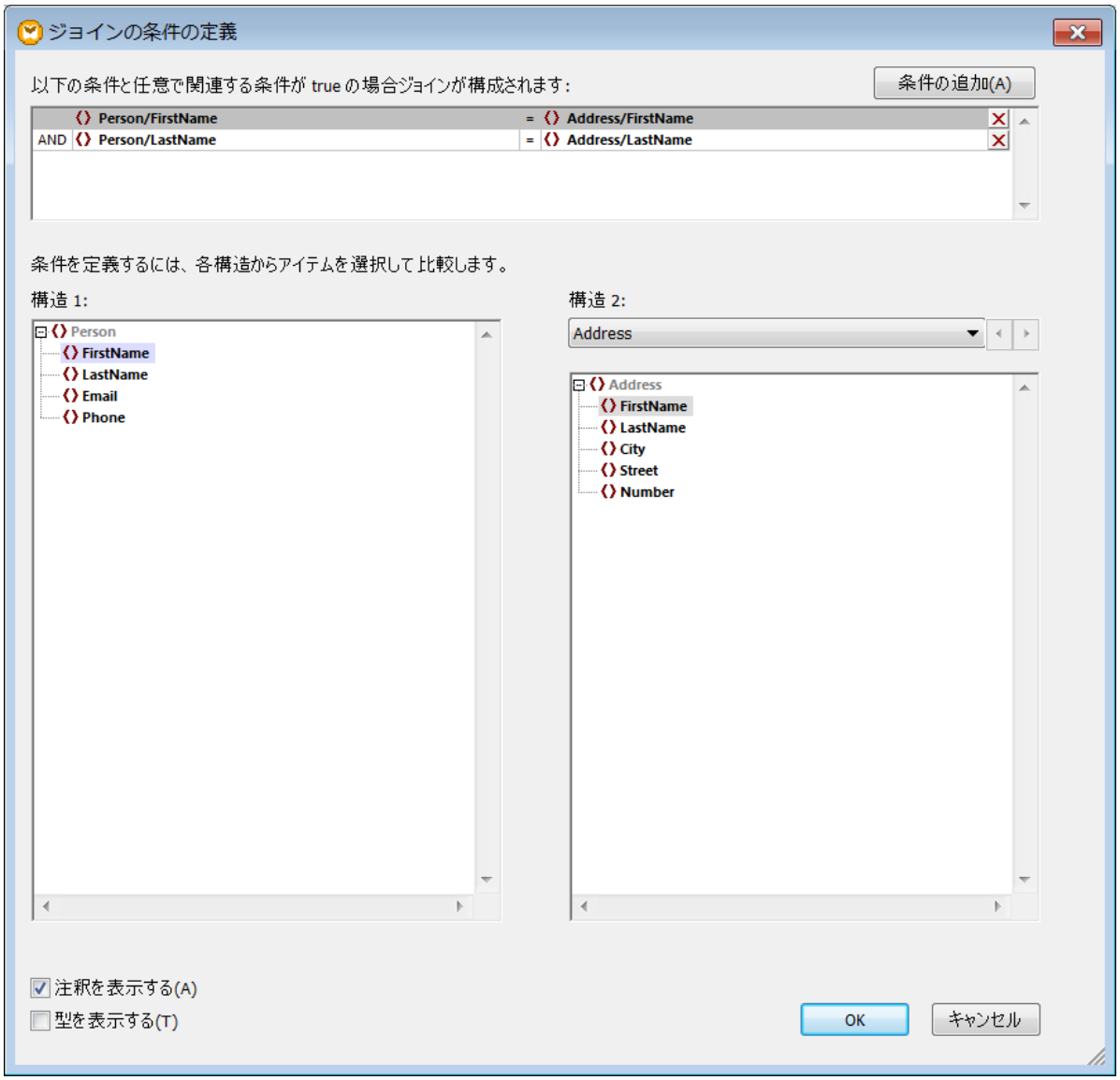
ジョインコンポーネントの構造に注目してください、比較する2つの構造(この場合、`Person` と`Address` 構造)を接続する2つのnodes/rows アイテムが存在します。

2. ジョインコンポーネントの最初の`nodes/rows` アイテムに`Person` から接続を描きます。同様に、2 番目の`nodes/rows` アイテムに`Address` を接続します。

(C) 2015-2021 Altova GmbH



3. 既に説明されているとおり、`FirstName` と`LastName` 値が両方の構造内で等価の場合のみジョインは実行されます。この条件を定義するために、「ジョインの条件の定義」ボタンをクリックしてください。
4. 最初のジョインの条件（構造 1 の下の`FirstName` と構造 2 の下の`FirstName` ）を定義するアイテムのペアを選択します。
5. 「条件の追加」をクリックして、上記のステップを`LastName` のために繰り返します。



マッピングの一部では、1つの比較から構成される条件のみでジョインを実行することを可能な場合があります。しかしながら、このサンプルでは、2つの比較が作成されることが重要です:

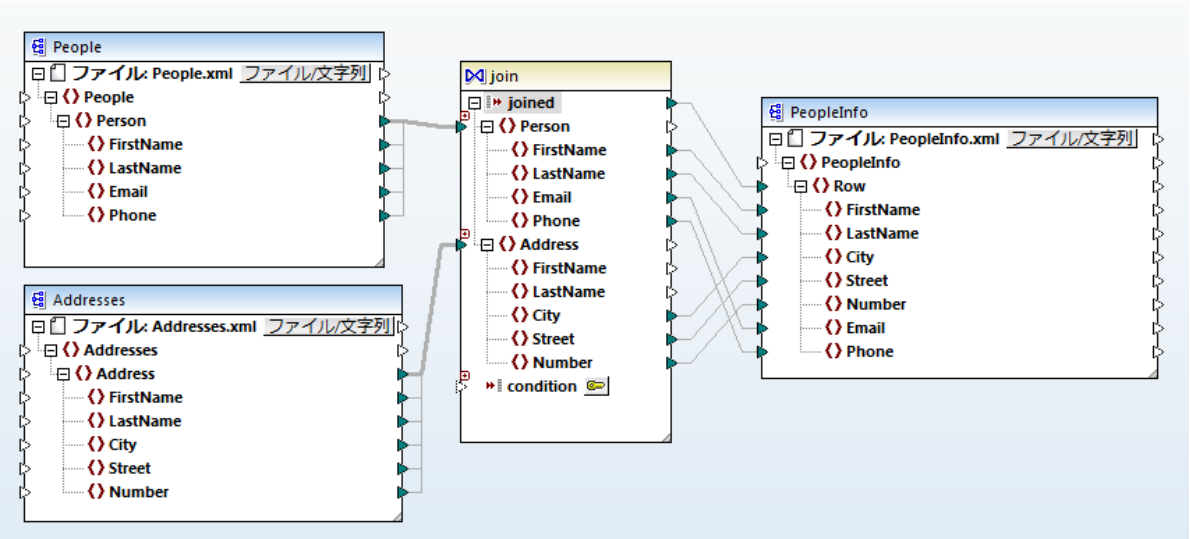
1) 構造1 内の`FirstName` = 構造 2 内の`FirstName`。
2) 構造1 内の`LastName` = 構造 2 内の`LastName`。

複数の条件が定義されている場合、ジョインが実行されるには全ての条件が満たされている必要があります。このため、このサンプルでは、(期待される結果である)ジョインは両方の比較の条件が満たされている場合のみ実行されます。それ以外の場合、1つの比較の条件のみが定義されている場合、ジョインは同じ名と姓を持つ個人に対して実行される可能性があります。

ステップ 4: ジョインコンポーネントをターゲット スキーマにマップする

2つの構造はジョインされ、ジョインした構造のどのアイテムがターゲットにマップされるかを定義することができます。これを行うには、ターゲットコンポーネントへジョインされた構造のアイテムからの接続を下に示されるように作成します。`joined` と`Row` 間の接続の目的は次のとおりです: ジョインの条件が満たされると、ターゲット内に新規の`Row` アイテムを作成する。

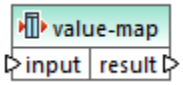
(C) 2015-2021 Altova GmbH



出力 タブをクリックしてマッピング出力をプレビューすることができます。期待通り、各レコード（<Row>）には、2つの異なるソースからジョインされた住所の詳細が含まれています。

4.12 Value-Map の使用

Value-Map コンポーネントにより *ルックアップ* を使用して値を他の値に置き換えることができます。このようなコンポーネントは1度に1つの値を処理します。このためにマッピング上に1つの **入力** と1つの **結果** が存在します。



Value-Map は*アイテムを置き換えるために*2つのセット内の個別のアイテムをマップする場合役に立ちます。例えば、数値（1、2、3、4、5、6 および 7）で表示された曜日をそれぞれの曜日（「月曜」、「火曜」など）と置き換えます。同様に月の名前（「1月」、「2月」、「3月」など）を各月の数値表記（1、2、3, etc）に置き換えます。マッピングの*ランタイム*では、一致する値は*カスタム ルックアップ テーブル*に従い置き換えられます。両方のセット内の値は異なる型であることができますが、各セットは同じデータ型の値を保管する必要があります。

Value-Map コンポーネントは最初のセット内の各値が2番目のセット内の単一の値に対応する単純な*ルックアップ*に適しています。*ルックアップ テーブル*内で値が見つからない場合、カスタム値、または空の値と置き換える、またはそのままパスすることができます。更に複雑な条件をベースにした値を*ルックアップ*または*フィルター*する場合、コンポーネントのフィルターを代わりに使用します。

重要な点は、コードを生成またはMapForce Server 実行可能ファイルをマッピングからコンパイルする場合、*ルックアップ テーブル*データは生成された*コード*、または*ファイル*内に埋め込まれる点です。この結果、*ルックアップ テーブル*を直接マッピング上で定義することは、データが頻繁に変更されず、（例えば、数百エントリより*も*少ないなど）データの総量が大きくない場合のみ良い選択でしょう。*ルックアップ*データが定期的に変更する場合、マッピング、および生成された*コード*を定期的に管理することは難しく、*ルックアップ*データをテキスト、XML、データベース、またはExcel として管理する方法のほうが簡単です。

*ルックアップ テーブル*が大きい場合、*ルックアップ テーブル*によりマッピングの実行が遅くなる場合があります。この場合、代わりにSQL-Whereを使用するデータベースコンポーネントを使用することが奨励されます。ポータビリティを考慮すると、SQLite データベースは良い候補です。他方、MapForce Server またはMapForce Server Advanced Edition を使用してマッピングを実行し*ルックアップ テーブル*のパフォーマンスを向上することもできます。

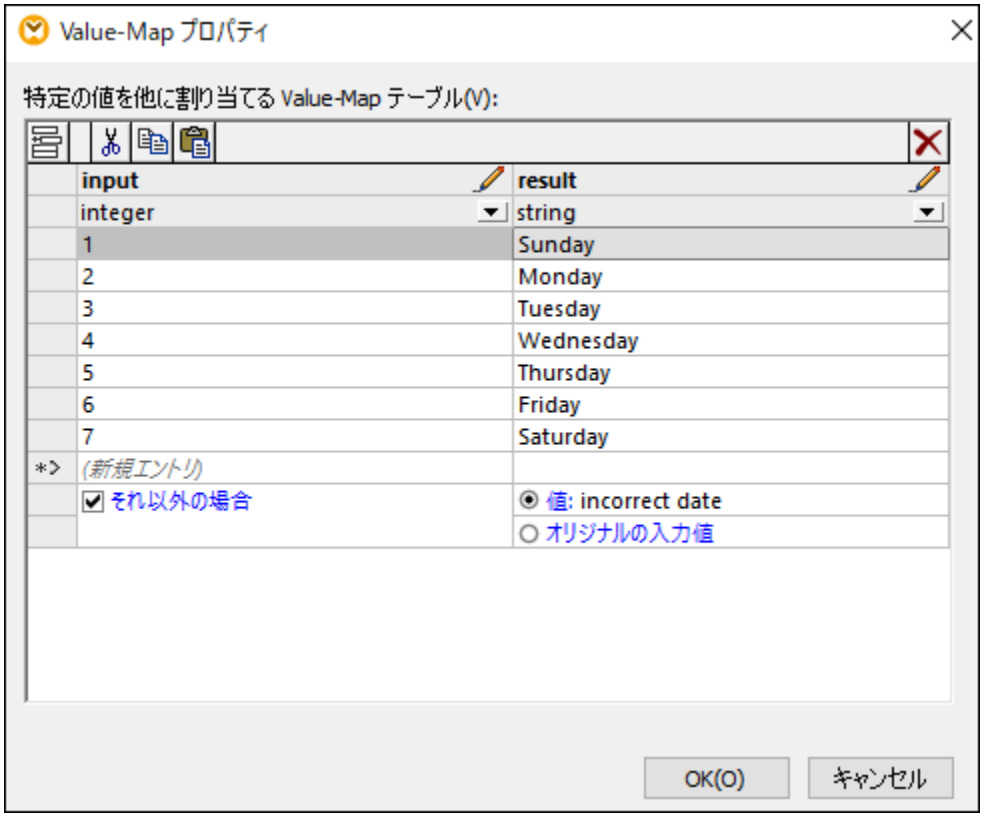
Value-Map の作成方法

Value-Map コンポーネントをマッピングに追加するには次のいずれかの操作を実行してください：

- 「**Value-Map の挿入**」 ツールバーボタンをクリックします。
- 「**挿入**」メニューから「**Value-Map**」をクリックします。
- 接続を右クリックし「**Value-Map の挿入**」をコンテキストメニューから選択します。

この操作は新規のValue-Map コンポーネントをマッピングに追加します。*ルックアップ テーブル*にアイテムのペアを追加することから開始します。これを行うにはコンポーネントのタイトルバーをダブルクリック、または右クリックし、「**プロパティ**」をコンテキストメニューから選択します。

(C) 2015-2021 Altova GmbH



マッピングのランタイムで、MapForce は Value-Map の **入力** に各値が到達するかをチェックします。*ルックアップ* テーブル内の *左側の列* に一致する値が存在する場合、元の入力値を *右側の列* からの値と置き換えます。それ以外の場合、任意で以下の *オプション* の一つに戻るように構成することができます:

- 置換の値。下のサンプル内では、置換の値はテキスト「incorrect date」です。置換の値をテキストを入力せずに空に設定することもできます。
- 元の入力の値。これはテーブル内で一致が存在しない場合、元の入力値がマッピングにパスされることを意味します。

それ以外の場合の条件を構成しない場合、Value-Map は一致が見つからない場合 *空のノード* を返します。この場合、ターゲットコンポーネントには何もパスされず、出力には空のフィールドが含まれます。このような場合が発生することを回避するために、それ以外の場合の条件を構成、または、`substitute-missing` 関数 を使用します。

空の置換値の設定と、それ以外の場合の条件を設定しない事は異なります。最初のケースは、出力でフィールドは生成されますが、空の値が存在します。後者のケースは値を囲むフィールド（または XML 要素）が作成されません。詳細に関しては、例: 職位の置換 を参照してください。

Value-Map の設定方法

ルックアップ テーブル内で、必要な数の値のペアを定義することができます。値を手動で入力する、または、テキスト、CSV、または Excel ファイルからテーブルのデータをコピーして貼り付けることができます。一般的なブラウザーを使用して HTML ページからテーブルをコピーして貼り付けることができます。 DB クエリペイン 内のデータベースグリッドからデータを張り付けることができます。テキストファイルからデータをコピーするとフィールドはタブ文字で区切られる必要があります。更に、MapForce は多くの場合コンマ、またはセミコロンにより区切られているテキストを認識することができます。

ルックアップ テーブルを作成する際に以下の点を考慮してください:

1. 左の列内のすべてのアイテムは一意である必要があります。それ以外の場合、具体的に一致するアイテムを決定することは不可能です。
2. 同じ列に属するアイテムは同じデータ型である必要があります。ルックアップ テーブル内の各列の上のドロップダウン リストからデータ型を選択することができます。ブール型を変換する必要がある場合、テキスト「true」または「false」を文字通り入力してください。このシナリオの説明は例: 曜日の置換を参照してください。

MapForce がルックアップ テーブル内で無効なデータを検知すると、ピンク色でハイライトされたエラーメッセージが表示されます。例:

Value-Map コンポーネントに外部ソースからデータをインポートする方法:

1. (例えば、Excel などの) ソース プログラム内の注目するセルを選択します。これは単一のデータ列、または2つの隣接した列であることができます。
2. 外部プログラムの「コピー」コマンドを使用してクリップボードにデータをコピーします。
3. Value-Map コンポーネント上で、データを張り付ける行の前をクリックします。
4. Value-Map コンポーネント上の「**クリップボードからテーブルを張り付ける**」 ボタンをクリック、または「Ctrl+V」または「Shift+Insert」を押します。

メモ 「**クリップボードからテーブルを張り付ける**」ボタンはソースからデータがコピーされている場合のみ有効化されています(すなわち、クリップボード上にデータが存在します)。

クリップボード データに複数の列が含まれる場合、最初の2列の行のデータのみがルックアップ テーブルに挿入されます。以降の行は無視されます。既存の値のトップの単一行からデータを張り付ける場合、コンテキスト メニューが表示され、新規の行としてクリップボード上のデータが挿入されるか、または既存の行が上書きされるかが問われます。このため、テーブル内の既存の値を上書きする必要がある場合、クリップボードには値だけではなく1行のみのデータが含まれていることを確認してください。

既存の行の前に行を手動で挿入するには、最初に注目する行をクリックし、「**挿入**」 ボタンをクリックします。

既存の行を削除するには、左側にマウスのボタンを押したまま行をドラッグして(上または下の) 新しい位置に移動します。

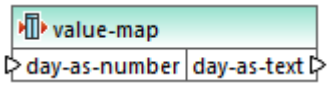
他の箇所で後に張り付けるためのコピーまたは切り取りを行うには、行を選択し「**コピー**」 ボタン(または「**切り取り**」 ボタンをそれぞれクリックします。必ずしも連続していない複数の行をコピーまたは切り取ることもできます。複数の行を選択するには、行を選択しながら「Ctrl」キーを押し続けます。切り取られた、またはコピーされたテキストは両方の列からの値を含むことに注意してください。1つの行からの値のみを切り取る、またはコピーすることはできません。

行を削除するには「**削除**」 ボタンをクリックします。

左と右の列をスワップするには、「**スワップ**」 ボタンをクリックします。

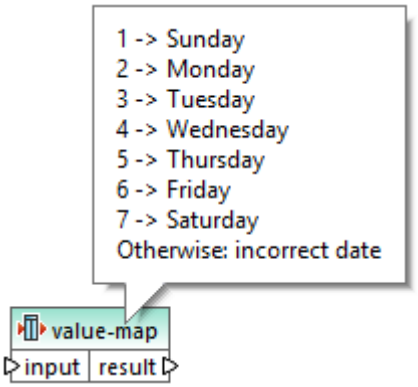
Value-Map パラメーターの名前を変更する方法

デフォルトではValue-Map コンポーネントの入力パラメータは「入力」と呼ばれ、出力パラメーターは「結果」と呼ばれます。マッピングをより簡潔にするために、それぞれの名前の横の「**編集**」 ボタンをクリックしてこれらのパラメーターの名前を任意で変更することができます。以下は、カスタム パラメーター名を持つValue-Map のサンプルです:



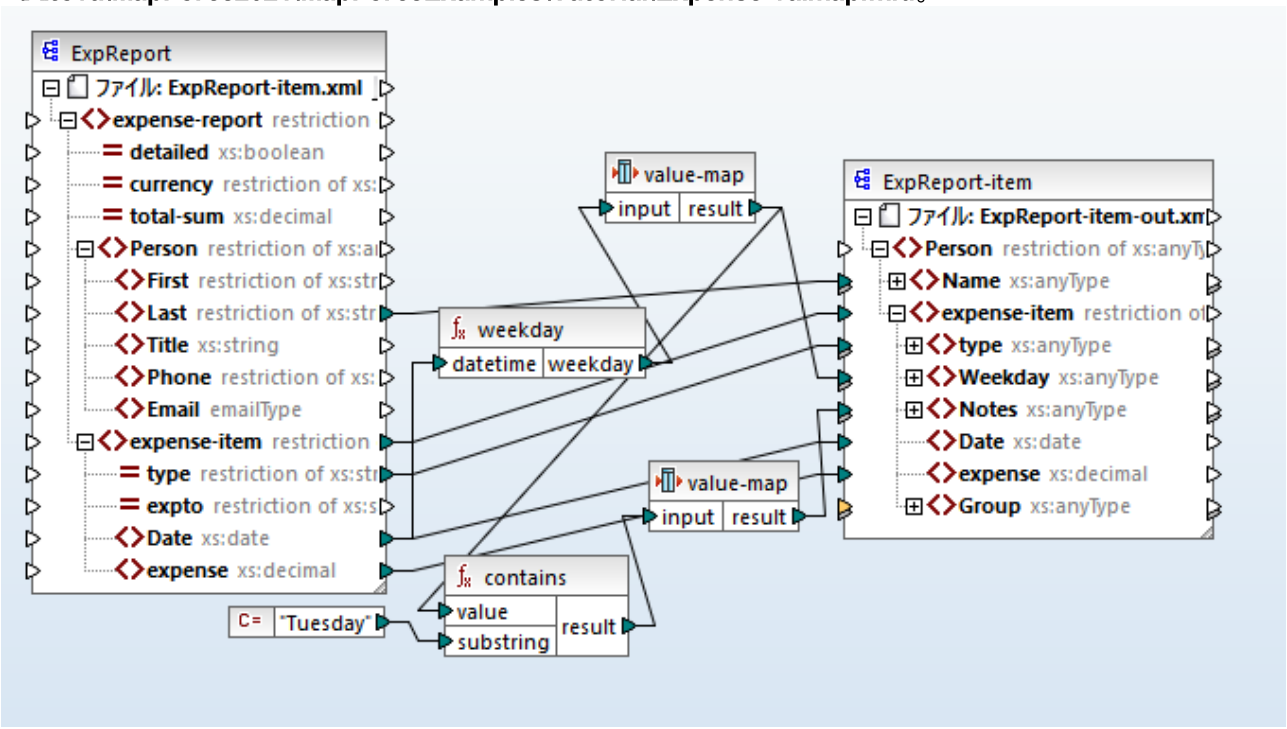
Value-Map をプレビューする方法

Value-Map の作成後、コンポーネントのタイトルバーにマウスをかざすことによりマッピングから直接実装をプレビューすることができます:



4.12.1 例: 曜日の置換

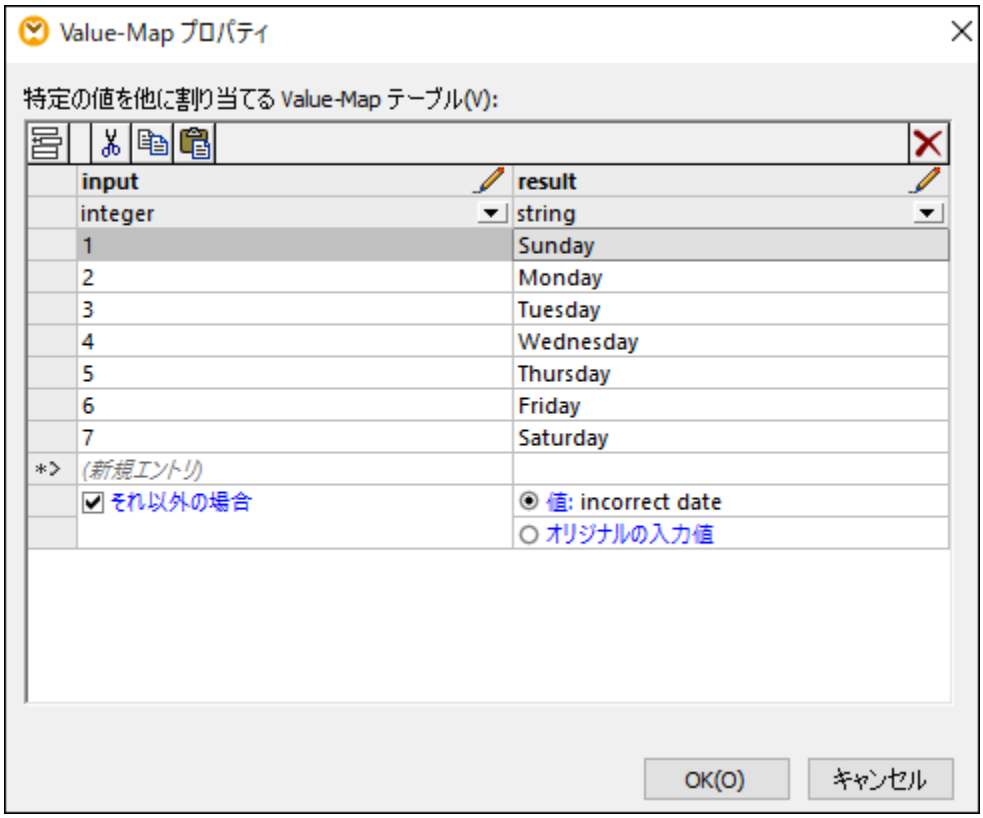
このサンプルは整数の値を曜日の名前 (1 = 日曜日、2 = 月曜日 など) と置き換える Value-Map を説明しています。このサンプルは以下のパスで見つけることのできるマッピングが存在します: **<マイドキュメント>\Altova\MapForce2021\MapForceExamples\Tutorial\Expense-valmap.mfd**。



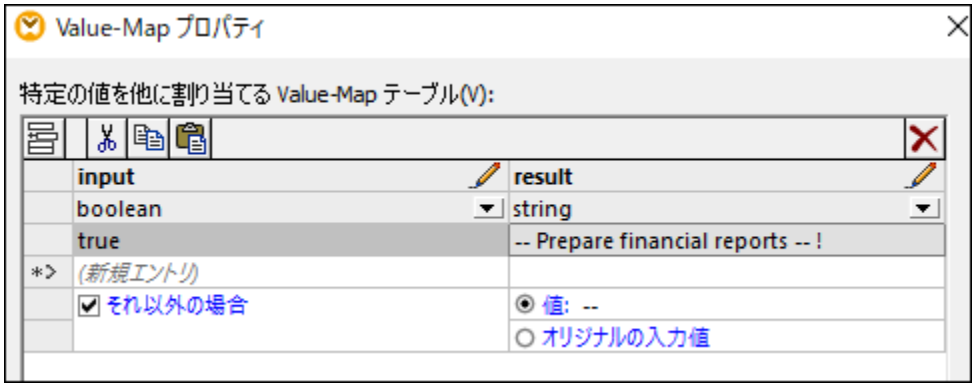
Expense-valmap.mfd

このマッピングはソースファイル内の**Date** アイテムから曜日を抽出し、テキスト内で数値に変換し、ターゲットコンポーネントの**Weekday** アイテムに書き込みます。具体的には、以下が行われます:

- `weekday` 関数はソースファイル内の**Date** アイテムから曜日の数値を抽出します。この関数の結果は1 から7 の範囲の整数です。
- 最初のValue-Map コンポーネントは整数を曜日（1 = 日曜日、2 = 月曜日 など）に変換します。コンポーネントは1-7 範囲外の無効な整数が発生するとテキスト「incorrect date」（無効な日付）を返します。



- 曜日に「Tuesday」が含まれている場合、テキスト「Prepare Financial Reports」はターゲットコンポーネント内で**Notes** アイテムに書き込まれます。これはブール型 **true** または**false** 値を2番目のValue-Map コンポーネントに渡す `contains` 関数を使用して達成することができます。2番目のValue-Map には次の構成が存在します:



上で説明されているValue-Map は以下のように理解されるべきです:

- ブール値 **true** が発生すると、テキスト「— Prepare financial reports — ! 」に変換されます。それ以外の場合、テキスト「—」が返されます。

(C) 2015-2021 Altova GmbH

最初の列のデータ型が「boolean」に設定されていることにご注意してください。これは入力ブールの値 **true** が認識されていることを保証しています。

4.12.2 例: 職位の置換

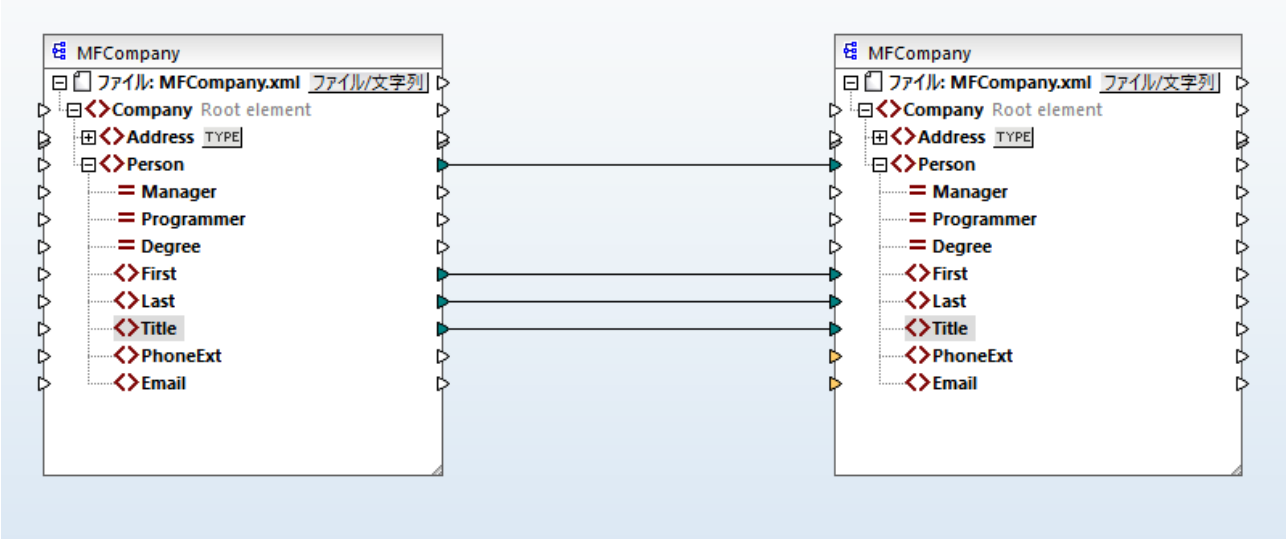
このサンプルはValue-Map コンポーネントを使用してXML ファイル内の特定の要素の値を置き換える方法(すなわち定義済みのルックアップテーブルを使用する方法)を示しています。

このサンプルで必要とされるXML ファイルは以下のパスで見つけることができます: **<マイドキュメント>\Altova\MapForce2021\MapForceExamples\Tutorial\MFCompany.xml**。他のデータ以外に、企業の従業員と職位などの情報を保管します。例:

```
<Person>
   <First>Michelle</First>
   <Last>Butler</Last>
   <Title>Software Engineer</Title>
</Person>
<Person>
   <First>Lui</First>
   <Last>King</Last>
   <Title>Support Engineer</Title>
</Person>
<Person>
   <First>Steve</First>
   <Last>Meier</Last>
   <Title>Office Manager</Title>
</Person>
```

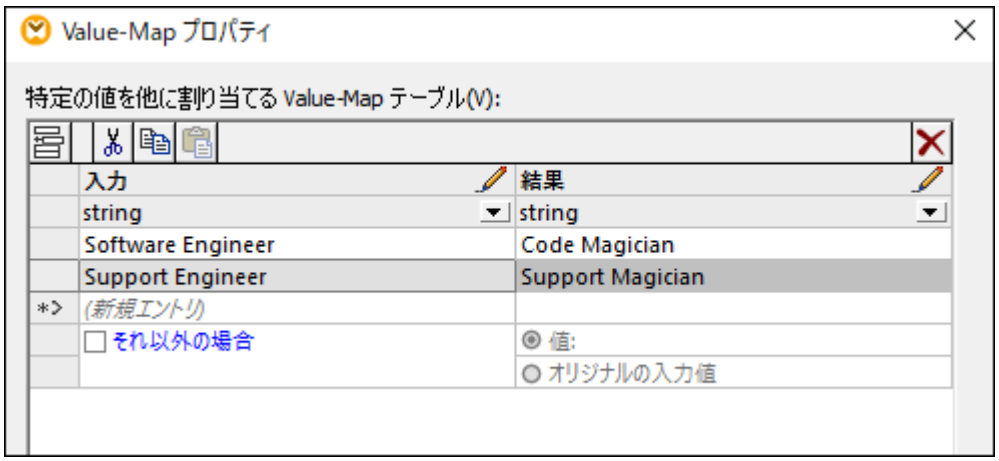
XML ファイル内の職位の一部を置き換える必要があると仮定します。具体的には職位「Software Engineer」は「Code Magician」と置き換えられます。また、職位「Support Engineer」は「Support Magician」と置き換えられます。その他の職位はすべて変更されません。

この目的を達成するために、**「XML スキーマ/ファイルの挿入」** ツールバーボタンをクリック、または**「挿入 | XML スキーマ/ファイル」**メニューコマンドを実行してマッピングエリアにファイルを追加します。次に、マッピング上のXML コンポーネントをコピーして貼り付け、以下に示される通りに接続を作成します。自動的に不必要な接続の作成を防ぐために **「子要素の自動的な接続の切り替え」** ツールバーオプションを最初にオフに切り替える必要がある場合があります。



First、**Last**、および**Title** 要素に変更を加えることなく作成されたマッピングは**Person** 要素をターゲットファイルにコピーします。

必要とされる職位を置き換えるために、Value-Map コンポーネントを追加します。2つの**Title** 要素間の接続を右クリックし、コンテキストメニューから「**Value-Map の選択**」を選択します。Value-Map プロパティを下に表示される通りセットアップします:



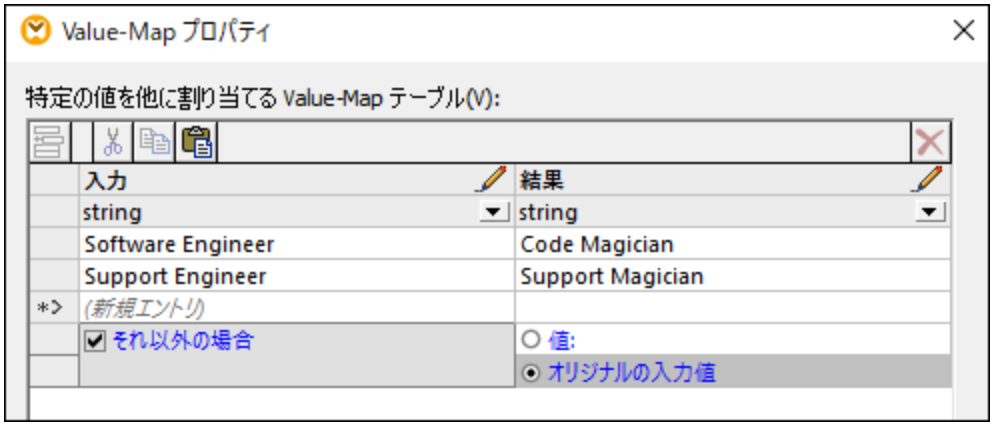
上記のセットアップに従い、「Software Engineer」は「Code Magician」と置き換えられます。そして、「Support Engineer」は「Support Magician」と置き換えられます。**それ以外の場合** 条件はまだ指定されていません。このため、Value-Map は職位が「Software Engineer」と「Support Engineer」以外の場合は *空のノード* を返します。この結果、出力タブをクリックし、マッピングをプレビューすると、**Person** 要素の一部には不足している**Title** が存在します。例:

```
<Person>
   <First>Vernon</First>
   <Last>Callaby</Last>
</Person>
<Person>
   <First>Frank</First>
   <Last>Further</Last>
</Person>
```


(C) 2015-2021 Altova GmbH

```
<Person>
   <First>Michelle</First>
   <Last>Butler</Last>
   <Title>Code Magician</Title>
</Person>
```

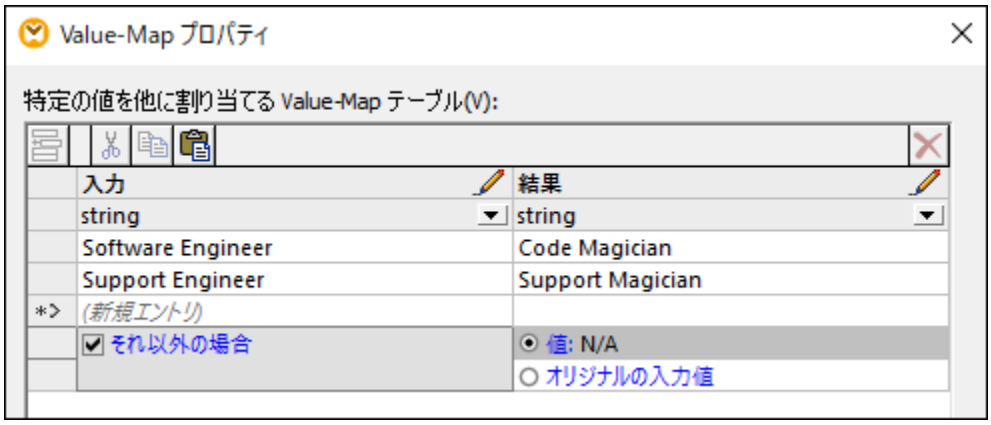
上記の通り、空のノードは生成される出力内で不足するエントリを引き起こします。このため、上記のXML フラグメント内では、ルックアップテーブル内で職位が存在するためMichelle Butler の職位のみが置き換えられています。ここまでに作成されている構成はオリジナルの必要条件を満たしていません。正しいセットアップは以下の通りです:



上記の構成を使用すると、マッピングのランタイムに以下が発生します:

- 「Software Engineer」は「Code Magician」と置き換えらえます。
- 「Support Engineer」は「Support Magician」と置き換えらえます。
- ルックアップテーブルでオリジナルのタイトルが見つからない場合、Value-Map は変換しないままタイトルを返します。

説明のために、「Software Engineer」と「Support Engineer」以外の職位を「N/A」などのカスタム値に変えることができます。これを達成するには、Value-Map プロパティを以下に示される通り設定します:



マッピングをプレビューする場合、各職位が出力に存在しますが、一致しない項目には「N/A」値が与えられます。例:

```
<Person>
   <First>Vernon</First>
```

```
   <Last>Callaby</Last>
   <Title>N/A</Title>
</Person>
<Person>
   <First>Frank</First>
   <Last>Further</Last>
   <Title>N/A</Title>
</Person>
<Person>
   <First>Michelle</First>
   <Last>Butler</Last>
   <Title>Code Magician</Title>
</Person>
```

Value-Map サンプルは終了です。上記のロジックを適用することにより、他のマッピング内で希望する結果を得ることができます。

(C) 2015-2021 Altova GmbH

4.13 例外の追加

例外は、フィルターにより返された条件が発生すると、マッピングの処理を停止し、エラーを表示する特別なコンポーネントの型です。true/false 条件をチェックするフィルターをマッピングが含む場合、例外を追加することができます（次を参照: フィルターと条件）。例えば、カスタム化されたしきい値よりマッピングアイテムの一部の値が大きい場合、例外をスローするなどが挙げられます。

MapForce を使用して、WSDL Web サービスをデザインする際にも例外を追加することができます。（次を参照: Web サービスのフォールト）。

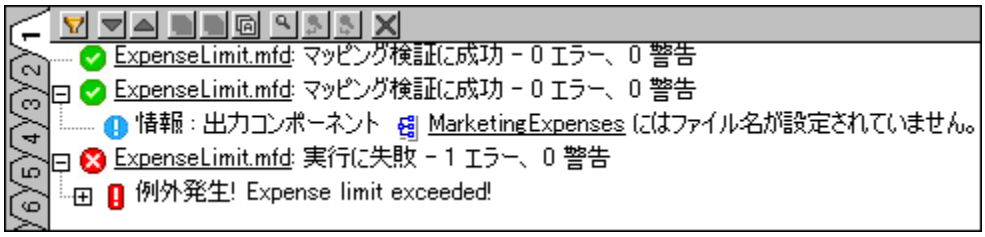
例外をマッピングに追加する:

1. 「**挿入**」メニューから、「**例外**」をクリックします。
2. 「**例外の挿入**」（ ）ツールバーボタンをクリックします。
3. 例外 **throw** 入力をフィルターの**on-true** または**on-false** 出力に接続します。
4. この例外がスローされると、任意で、例外の**error-text** 入力を、エラーのテキストを与える他の（通常、定数である）コンポーネントに接続します。

メモ フィルターの**on-true** と**on-false** 出力が双方接続されている必要があります。具体的には、これら出力の内の1つが、例外に（中間関数または、コンポーネントを使用することなく）直接接続されていなければなりません。

マッピングに例外が発生すると、以下のような通知が送信されます:

- MapForce 内でメッセージウィンドウはエラーと例外テキストを表示します。"Expense limit exceeded")。



マッピング言語がXSLT 2.0 またはXQuery の場合、「実行に失敗しました」" エラーがメッセージウィンドウ内で表示され、対応するXSLT2 またはXQuery タブが開かれます。エラーの生じたラインはメッセージウィンドウ内でハイライトされます。

- MapForce Server からマッピングを実行する場合は、MapForce 内で定義されたカスタムの例外テキストに続きエラー「例外がスローされました」が返されます。

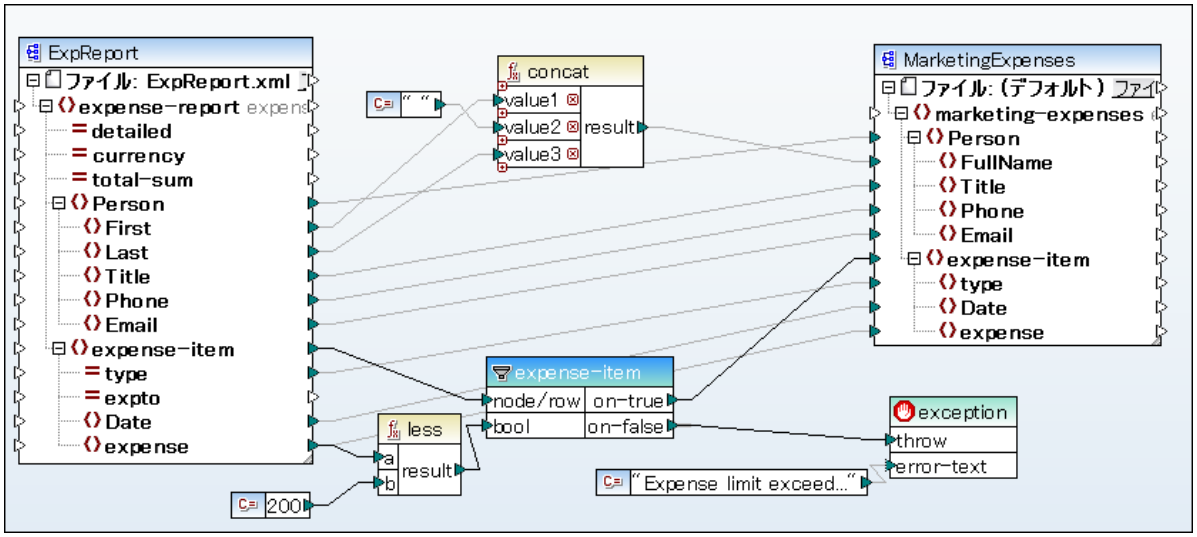
```
c:\codegen\mfx\ExpenseLimit>"C:\Program Files (x86)\Altova\MapForceServer2016\bi
n\MapForceServer.exe" run ExpenseLimit.mfx
Exception was thrown! Expense limit exceeded!
```

- 生成されたC#、C++、またはJava コードからマッピングを実行すると、エラー"USER EXCEPTION" が、内で定義されたカスタム例外テキストの前に返されます。

```
C:\codegen\cs\ExpenseLimit\Mapping\bin\Debug>Mapping.exe
Mapping Application
USER EXCEPTION: Expense limit exceeded!
```

4.13.1 例: "より大きい" 条件の例外

この例では、「より大きい」条件が発生すると例外を発生するマッピングの作成方法を説明します。このマッピングのためのサンプルマッピングは以下にあります: **<マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥ExpenseLimit.mfd**。



このマッピングは、ソースXML インスタンス内の**expense** アイテムが200 以上の値を有すると、例外をスローします。値 "200" は、定数により与えられます。`less` 関数 は、2つの値を比較するために使用されます。**expense** の値が200 より小さい場合、その親である**expense-item** をフィルターにパスされ、例外はスローされません。それ以外の場合、カスタムテキスト "Expense limit exceed" と共に、例外がスローされます。

上に示されるように、例外は、 アイコンにより識別され、次の2つのアイテムより構成されます: **throw** と**error-text**。 **throw** アイテムは、フィルターの**on-false** または**on-true** 出力に接続されている必要があります。**error-text** は、例外のカスタムテキストを与える定数に接続されています。

重要な点は、フィルターの両方の出力は接続されています。それ以外の場合、例外は、スローされません。この特別な例では、**on-false** 出力は、例外に接続されていますが、**on-true** 出力は、ターゲットコンポーネントに接続されています。

4.13.2 例: ノードが存在しない場合の例外

この例では、ソースXML スキーマが存在しない場合例外をスローする方法を説明します。説明を簡素化するために、この例は、同じ XML スキーマをソースとターゲットコンポーネントで使用します。

ソーススキーマをマッピングに追加する:

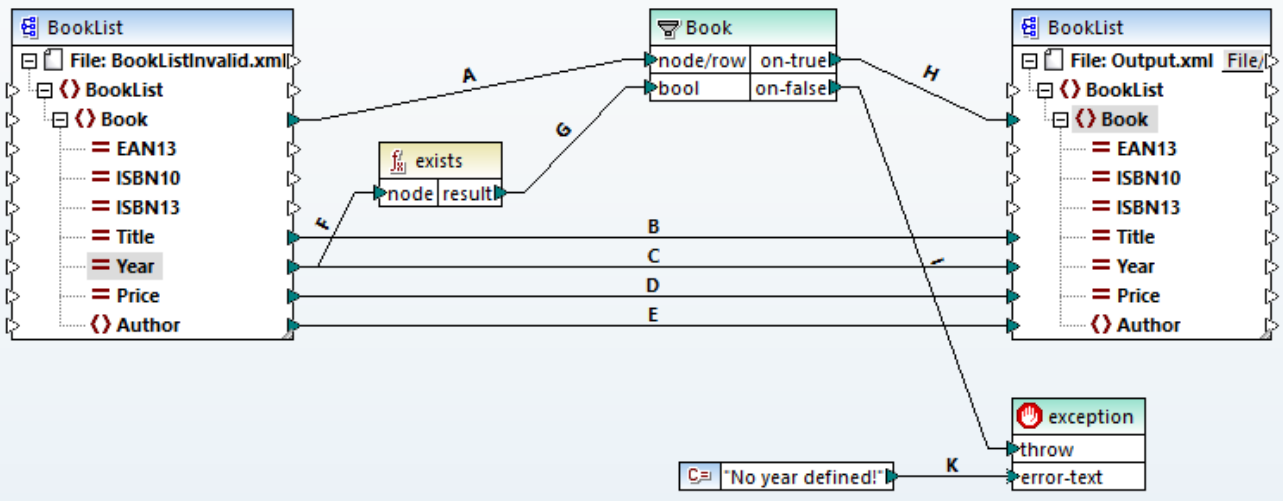
1. 「**挿入**」メニューから「XML スキーマ/ファイル」をクリックし <**マイドキュメント>\Altova\MapForce2021\MapForceExamples\BookList.xsd** を参照します。
2. インスタンスファイルを提供するようにプロンプトされると、「**スキップ**」をクリックします。
3. スキーマルート要素を選択するようにプロンプトされると、**BookList** をルート要素として選択します。

ターゲットスキーマを追加するには、同じステップに従います。そして、対応するコマンド(または、対応するツールバーボタン)を「**挿入**」メニューから使用し、以下を追加します:

(C) 2015-2021 Altova GmbH

- **Filter: Nodes/Rows** コンポーネント（以下も参照してください: フィルターと条件）
- テキスト "No year defined!" を伴う定数
- 例外

最後に、**ライブラリ** ウィンドウから、`exists` 関数をマッピングエリアにドラッグして、下に表示されるように接続を描きます。



XML スキーマに従うと、書籍の名前以外の**Book** 要素の全ての属性は、任意です。ですから、"Year" 要素は、有効な XML インスタンス内に存在する、もしくは、しない場合があります。このマッピングのゴールは、各書籍のために "Year" 要素が存在する箇所で XML インスタンスを正確に処理することです。それ以外の場合、マッピングは、例外をスローします。

マッピングの例外の成功をテストする:

1. ソースコンポーネントのヘッダーをダブルクリックします。「**入力** XML **ファイル**」の横から、次のファイルを参照します: <**マイドキュメント**>¥Altova¥MapForce2021¥MapForceExamples¥BookList.xml。
2. 「**出力**」ボタンをクリックしてマッピングを実行します。

例外をテストする:

1. 同じディレクトリ内に、**BookListInvalid.xml** と呼ばれる **BookList.xml** ファイルのコピーを作成します。
2. Book 要素から "Year" 要素を削除するように修正します。
3. ソースコンポーネントのヘッダーをダブルクリックします。「**入力** XML **ファイル**」の横から、**BookListInvalid.xml** ファイルを参照します。
4. 「**出力**」ボタンをクリックしてマッピングを実行します。

マッピングの作動のしくみに注目します。

接続 **A** は、ターゲット インスタンスがソース インスタンス内の各書籍のために作成されていることを保証します。接続 **B**、**C**、**D**、**E** は "Title"、"Year"、"Price"、と "Author" が各書籍のためにソースからターゲットにコピーされることを保証します。

接続 **F** は、`exists` 関数が "Year" 属性の存在をチェックするようをトリガーします。接続 **G** は関数の結果（`true` または `false`）をフィルターにパスします。結果が `true` の場合、"Year" 属性は存在し、書籍はフィルターにパスされ、フィルターに書籍はパスされ、後に、接続 **H** を介してターゲットにパスされます。

ソースコンポーネントの**Year** 出力にフィルターが直接接続されていないことに注意してください。これを行った場合、フィルターは自身の存在により**Year** をフィルターするので、意味が無く、例外は決してスローされません。

ルールに従い、フィルターの**on-false** または**on-true** 出力へ例外が接続されている必要があるため、**I** が存在します。

最後に、接続 **K** がカスタムエラーテキストを定数から例外コンポーネントにパスします。

(C) 2015-2021 Altova GmbH

4.14 文字列の解析とシリアル化

文字列の解析とシリアル化は、文字列からデータを解析、またはデータを文字列にシリアル化するコンポーネントを構成する高度なマッピング技術です。この技術は、ファイルからデータを読み込む(またはファイルにデータを書き込む)代替として使用することができます。文字列を解析する、またはデータを文字列にシリアル化するMapForce コンポーネントは、以下のようなシナリオにとても役にたちます。例:

- XML、テキスト、またはJSON ファイル などの構造を データベースフィールド またはExcel スプレッドシートセルに挿入する必要がある場合。
- データベースフィールド内に保管されたXML フラグメントを スタンドアロンのXML ファイルに変換する必要がある場合。
- (例えば、FlexText を使用する)レガシーデータをテキスト(例: 単一のデータベースフィールド内の固定値コンテンツ)として保存しており、このデータを並べ替え可能なフィールドベースの構造 にするために変換する場合。

文字列の解析とシリアル化 以下のMapForce コンポーネント型で使用することができます:

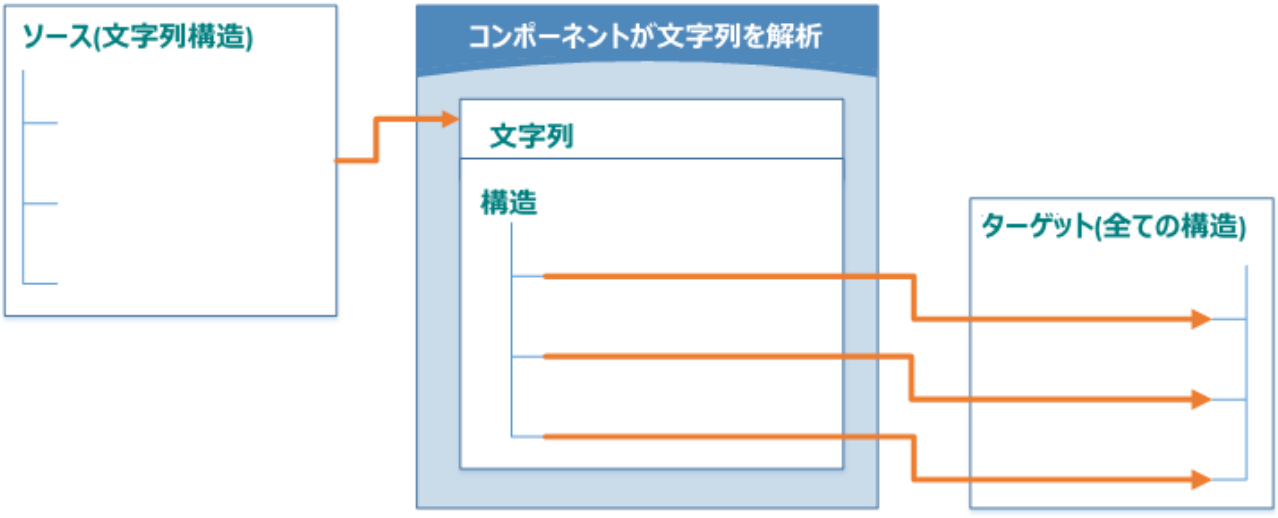
- テキスト(CSV、固定値フィールド テキスト、EDI、とMapForce Flexテキスト テンプレート)
- JSON スキーマファイル
- XML スキーマファイル

上記のコンポーネント型のために、文字列の解析とシリアル化はBUILT-IN ターゲット言語でサポートされています。更に、JSON への文字列の解析、または文字列からのJSON シリアル化はBUILT-IN、C#、またはJava でサポートされています。

4.14.1 コンポーネントの解析/シリアル化について

MapForce 内のコンポーネントの解析/シリアル化 コンポーネントは、ソースもしくはターゲットコンポーネントではない、ハイブリッドのコンポーネントです。マッピングデザイン内での役割を考えると、このようなコンポーネントはソースとターゲットコンポーネントの間に置かれることが適切です。

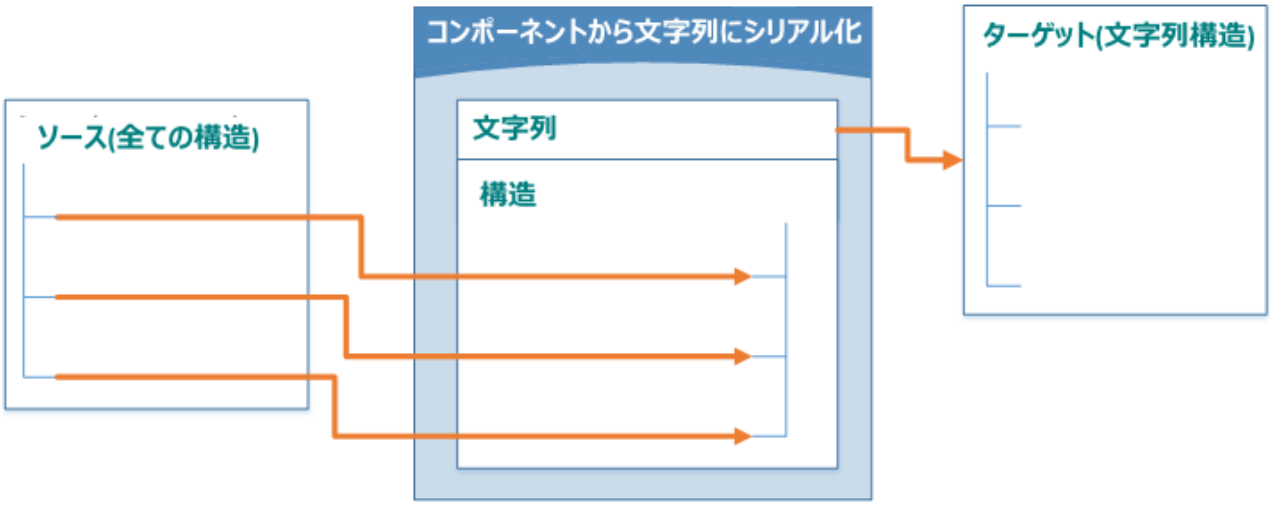
文字列の解析ためのコンポーネントの「文字列の解析/シリアル化」のためには、構造を持つ文字列を他のフォーマットに変換する必要があります(例: データベース内に文字列として保管されているXML)。ソース文字列からデータを解析するには、コンポーネントの"解析/シリアル化"は、ソース文字列がMapForce 構造に変換されることを意味します。ですから、文字列として保管されたソースXML の要素または属性へのアクセスを取得することができます。



ジェネリックな"文字列の解析"コンポーネント

上の図は、MapForce コンポーネントが文字列を解析する全般的な構造を示しています。コンポーネントの"文字列の解析/シリアル化"は、マッピングのソースとターゲットの間に配置されています。このコンポーネントは、文字列の構造の一部をトップ文字列ノードに接続されている単一のMapForce コネクタを入力として受け入れません。出力構造は、MapForce によりサポートされるデータターゲットとなることができます。例えば、 例: 文字列の解析 (固定長テキストをExcel へ)を参照。

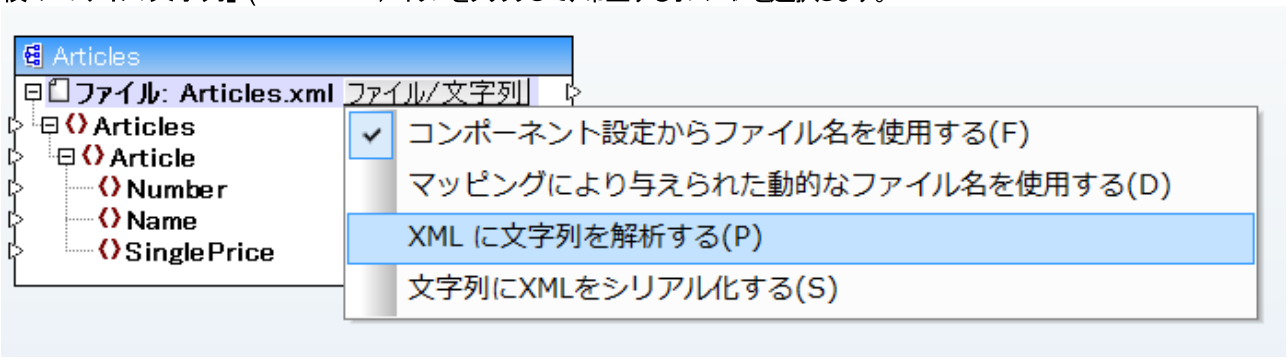
コンポーネントから文字列にデータをシリアル化すると、逆の効果が発生します。特に、MapForce コンポーネントの全体の構造が、必要であれば更に操作することが可能な文字列 構造になります。例: これによりXML ファイル(またはXML フラグメント) から、データベースフィールドまたはExcel スプレッドシートの単一のセルに書き込むことが可能になります。



ジェネリックな "文字列にシリアル化" コンポーネント

上の図は、MapForce でのジェネリックなコンポーネントの"文字列にシリアル化" を示しています。このコンポーネントは 標準 MapForce コネクタを使用して MapForce によりサポートされるデータソースを入力として受け入れます。出力構造は、コンポーネントのトップ文字列からターゲットコンポーネントアイテムに描かれた単一のMapForce コネクタを使用して、更にパスすることのできる文字列(例: スプレッドシートセル)です。 例: 文字列にシリアル化 (XML からデータベースへ)を参照してください。

文字列の解析またはシリアル化のためにコンポーネントをマッピングウィンドウからいつでも指定することができます。これを行うには、ルートノード横の「ファイル/文字列」(File/String) ボタンをクリックして、希望するオプションを選択します。



コンポーネントモードの変更

メモ "文字列の解析/シリアル化" コンポーネントは、文字列からデータを読み込み、同時に文字列に書き込むことはできません。ですから、ルートノードは入力コネクタ、または出力コネクタの(両方ではなく)どちらかを持つことができます。両方のオペレーションに同じコンポーネントを使用しようとすると、エラーが生成されます。

コンポーネントを文字列の解析またはシリアル化に指定する場合、コンポーネントの外観の変更は以下の通りです:

- コンポーネントはタイトル内に「解析」または「シリアル化」のプレフィックスを与えられます。
- タイトルバーは関数コンポーネントに類似した黄色の背景色が与えられます。
- トップのノードは **文字列: プレフィックス** から始まり、 アイコンで識別されます。
- コンポーネントが文字列を解析する場合、ルートノードからの出力コネクタは有効ではなく、使用することはできません。
- コンポーネントが文字列にシリアル化される場合、ルートノードへの入力コネクタは、有効ではなく使用することはできません。

コンポーネントが「文字列の解析/シリアル化」モード内にある場合、ファイルベースのモードと同様の法要で設定を変更することができます(コンポーネント設定を変更するを参照)。コンポーネントが「解析」または「シリアル化」モードの場合、全てのコンポーネント設定を使用できない場合があります。

4.14.2　例: 文字列の解析 (固定長テキストから Excel へ)

このサンプルは、文字列データを解析するマッピングデザインの方法について説明します。このサンプルにはサンプルファイルが存在します。このサンプルを開始する前に、サンプルファイルを確認するには、ファイルを以下のパスで検索することができます: **<マイドキュメント>\Altova\MapForce2021\MapForceExamples\Tutorial\ParseString.mfd**。

単一のデータベースフィールドに保管されているレガシーテキストデータがあると仮定します。テキストデータは従業員のリストであり、RESOURCE 列内に保管されています。固定値としてフォーマットされたフィールドは、以下のとおりです:

```
|----|-----------------------------------------------------------------------------------------|------------|
| ID | RESOURCE                                                                                | TYPE       |
|----|-----------------------------------------------------------------------------------------|------------|
| 1  | P00001Callaby       Vernon        582Office Manager          v.callaby@nanonull.com    | Employees  |
|    | P00002Further       Frank         471Accounts Receivable     f.further@nanonull.com    |            |
|    | P00003Matise        Loby          963Accounting Manager      l.matise@nanonull.com     |            |
|    | P00004Firstbread    Joe           621Marketing Manager Europe j.firstbread@nanonull.com |            |
|    | P00005Sanna         Susi          753Art Director            s.sanna@nanonull.com      |            |
|    | P00006Landis        Fred          951Program Manager         f.landis@nanonull.com     |            |
|    | P00007Butler        Michelle      654Software Engineer       m.landis@nanonull.com     |            |
|    | P00008Little        Ted           852Software Engineer       t.little@nanonull.com     |            |
|    | P00009Way           Ann           951Technical Writer        a.way@nanonull.com        |            |
|    | P00010Gardner       Liz           753Software Engineer       l.gardner@nanonull.com    |            |
|    | P00011Smith         Paul          334Software Engineer       p.smith@nanonull.com      |            |
|----|-----------------------------------------------------------------------------------------|------------|
```

単一のデータベースフィールドにテキストデータが保管されているため、個々の従業員の記録には簡単にアクセスすることができません。これにより、従業員を追加、または削除、および並べ替えすることが難しくなります。このサンプルの目的は、RESOURCE データベースフィールドからテキストデータを抽出し、記録を簡単に処理できるように構造に分割することです。

この目的は、コンポーネントの"文字列の解析/シリアル化"を使用することにより達成することができます。最初に、コンポーネントの"文字列の解析/シリアル化" はテキストデータを入力として受け入れます。そして、データを解析し、構造に変換します。最後に、ターゲットフォーマットに構造が書き込まれます。このサンプルでは、ターゲットフォーマットは Excel スプレッドシートです。かしながら、全般的に、MapForce によりサポートされる他の出力フォーマットに変換することができます。

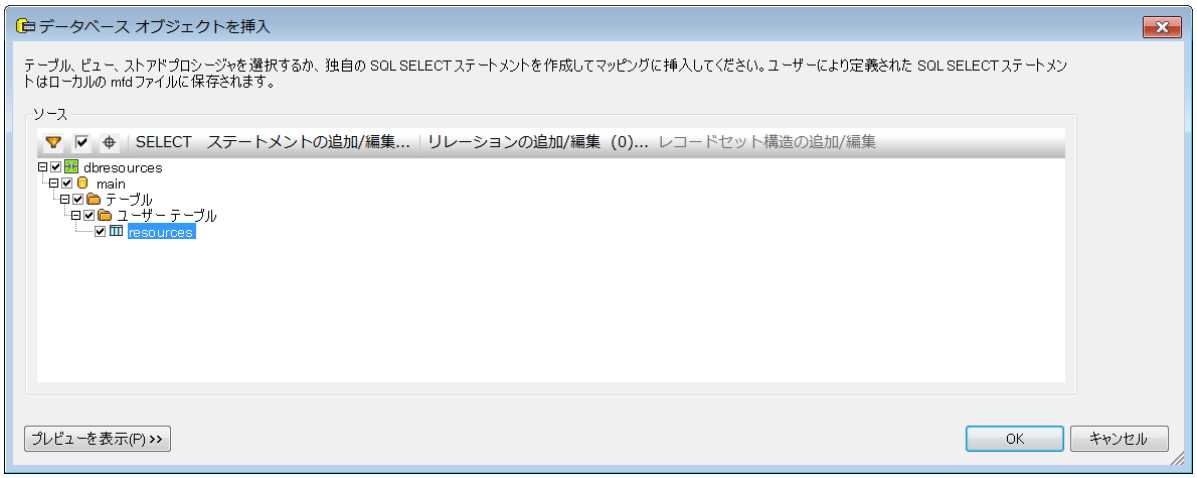
このサンプルのマッピングを要約すると、RESOURCE データベースフィールドのコンテンツをテーブルに変換します。マッピングが実行された後、各テーブル行は従業員に対応し、各列は固定長フィールドの1つに以下の順序で対応します: ID、Last Name、First Name, Extension、Job Title、Email。

	A	B	C	D	E	F	G
1	P00001	Callaby	Vernon	582	Office Manager	v.callaby@nanonull.com	
2	P00002	Further	Frank	471	Accounts Receivable	f.further@nanonull.com	
3	P00003	Matise	Loby	963	Accounting Manager	l.matise@nanonull.com	
4	P00004	Firstbread	Joe	621	Marketing Manager Europe	j.firstbread@nanonull.com	
5	P00005	Sanna	Susi	753	Art Director	s.sanna@nanonull.com	
6	P00006	Landis	Fred	951	Program Manager	f.landis@nanonull.com	
7	P00007	Butler	Michelle	654	Software Engineer	m.landis@nanonull.com	
8	P00008	Little	Ted	852	Software Engineer	t.little@nanonull.com	
9	P00009	Way	Ann	951	Technical Writer	a.way@nanonull.com	
10	P00010	Gardner	Liz	753	Software Engineer	l.gardner@nanonull.com	
11	P00011	Smith	Paul	334	Software Engineer	p.smith@nanonull.com	
12							
13							

文字列の解析後の期待される出力

目的を達成するには、以下のステップを行ってください：

1. マッピングエリアにソースデータベースを追加します。データベースは、スタンドアロン SQLite データベースファイルとして、以下のパスから参照することができます: **<マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥Tutorial¥\dbresources.db**。（データベースコンポーネントを追加するには、「**挿入 | データベース**」メニューコマンドを使用します。データベースへ接続するも参照）。
2. データベースオブジェクトの挿入を問われると、**resources** テーブルを選択します。



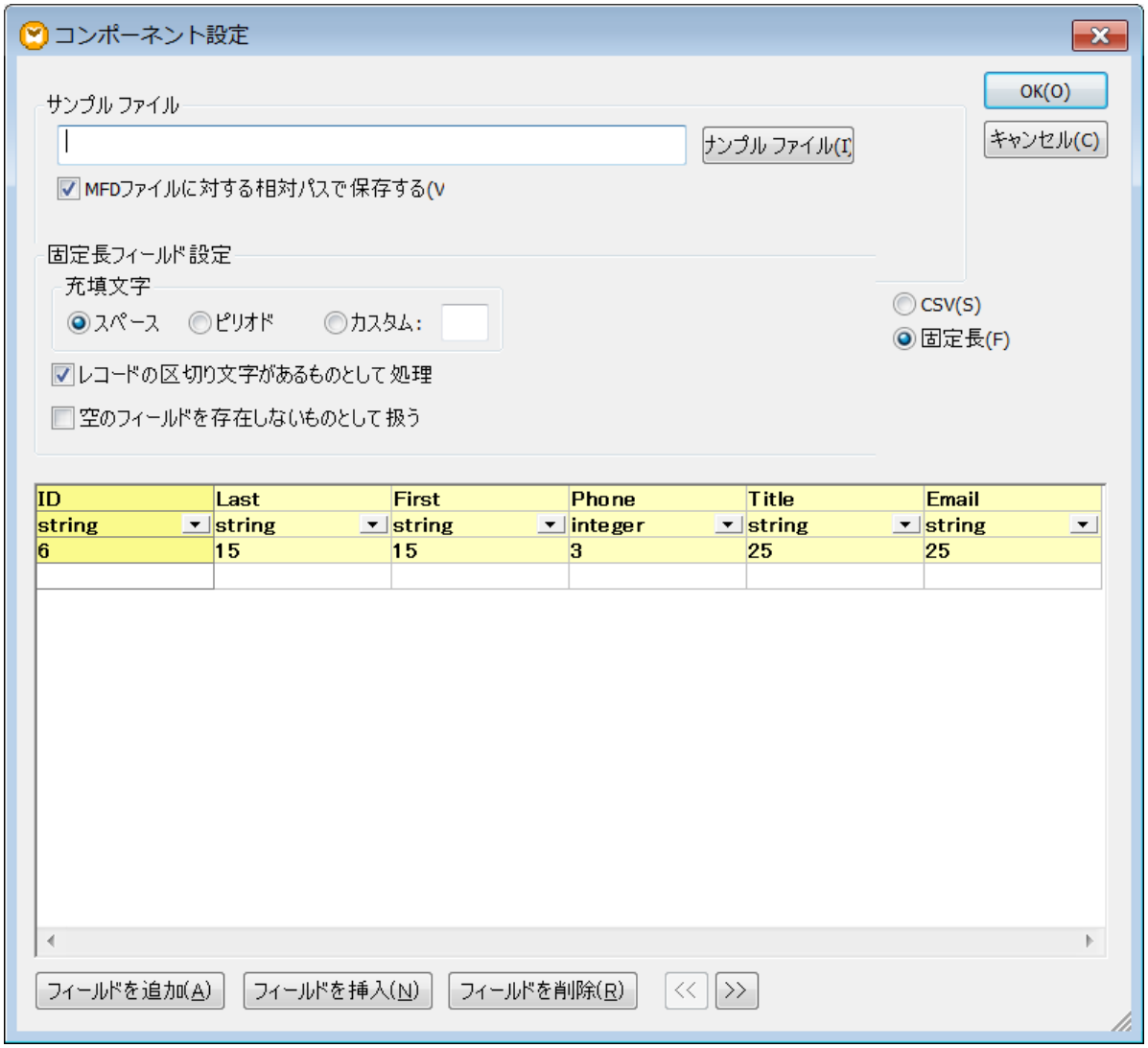
3. (「**挿入 | テキストファイル**」メニューコマンドを使用して) マッピングエリアにテキストコンポーネントを追加します。ソースデータはフィールドカンマ区切りテキストであるため、MapForce によりプロンプトされると、簡単な処理を選択します。
4. データベース内に保管されたソーステキストデータの構造をマップするために、テキストコンポーネントの構造を構成します。既に気が付いているかもしれませんが、このサンプルでは、ソーステキストは6つの固定長フィールドから以下のように構成されています:

文字サイズ	フィールド
6	ID
15	Surname
15	First Name

(C) 2015-2021 Altova GmbH

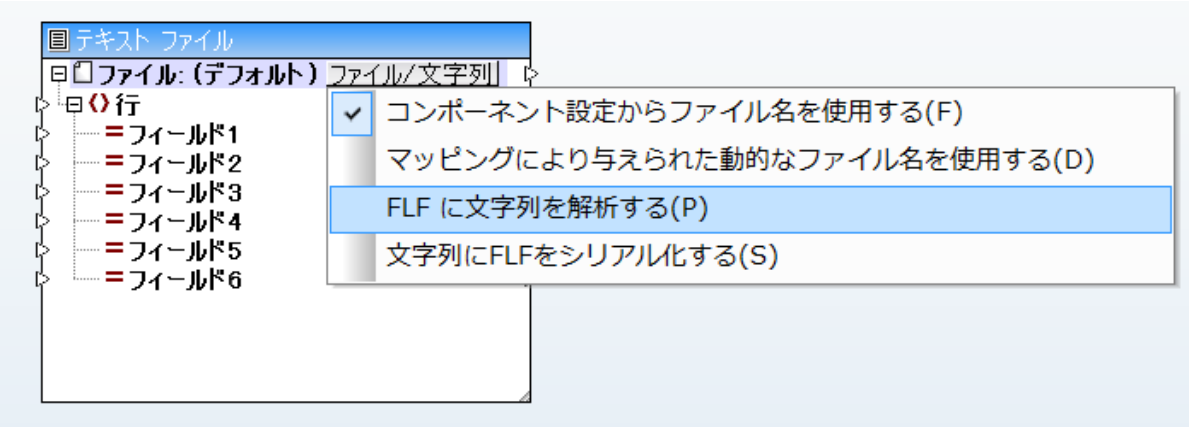
文字サイズ	フィールド
3	Extension
25	Job Title
25	Email

これを達成するには、テキストコンポーネントを固定長として宣言し、上記のポジションに適用することのできる6つのフィールドを追加します。(コンポーネント設定ダイアログボックスから開くには、コンポーネントを右クリックして、コンテキストメニューから、「プロパティ」を選択します)。



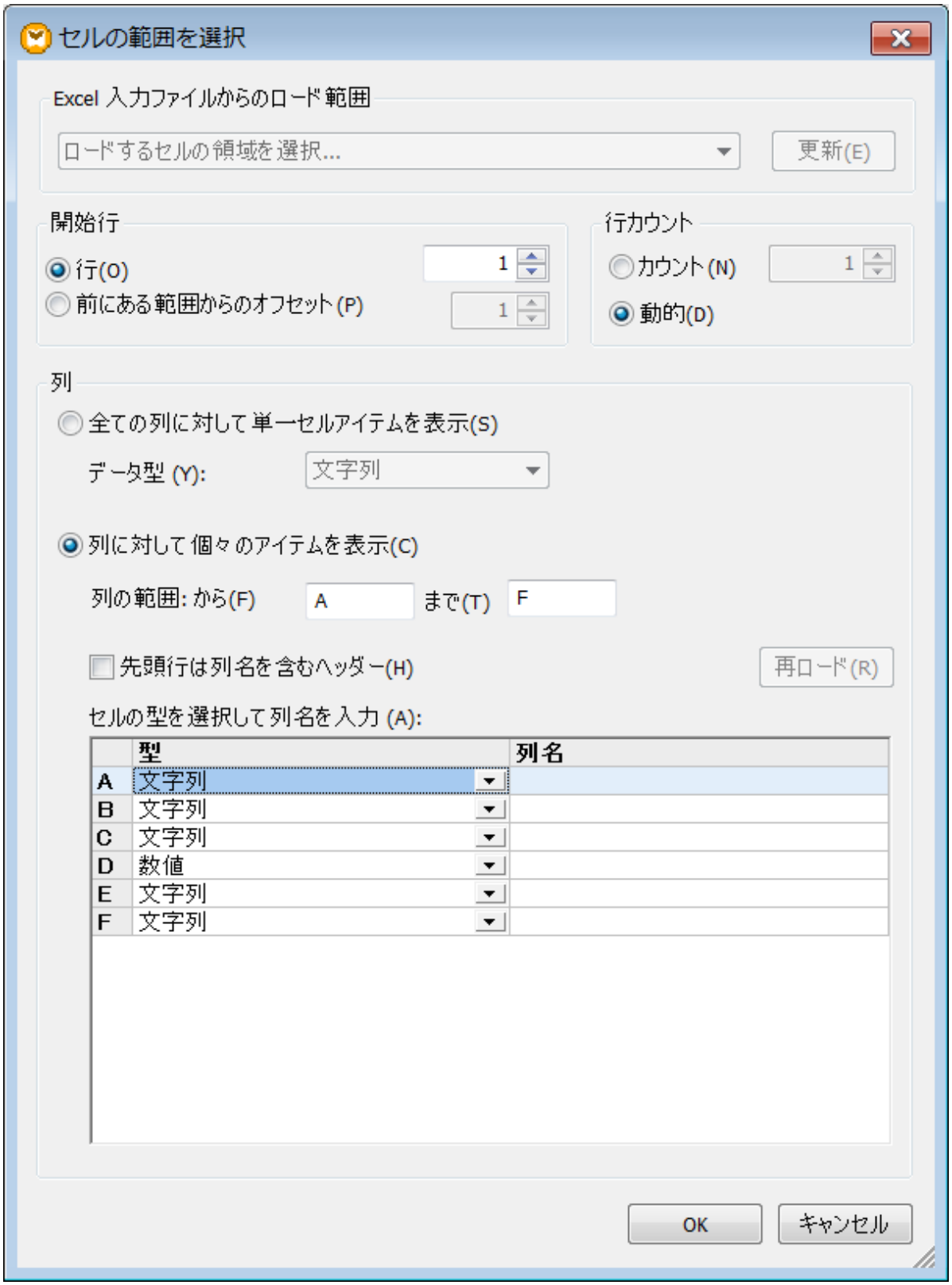
整数をデータ型として使用する **Field4** フィールドに注意してください。このサンプルのスコープのためには、**Field4** を数値型として宣言することは任意ですが、内線番号(**Field4**)が抽出するソーステキストが数値として検証されることを確認することができます。

5. File/String をクリックし、「FLF に文字列を解析する」を「コンテキスト」メニューから選択します。これは、このコンポーネントに文字列を 固定値フィールド (FLF) フォーマットに解析するように、MapForce に命令します。

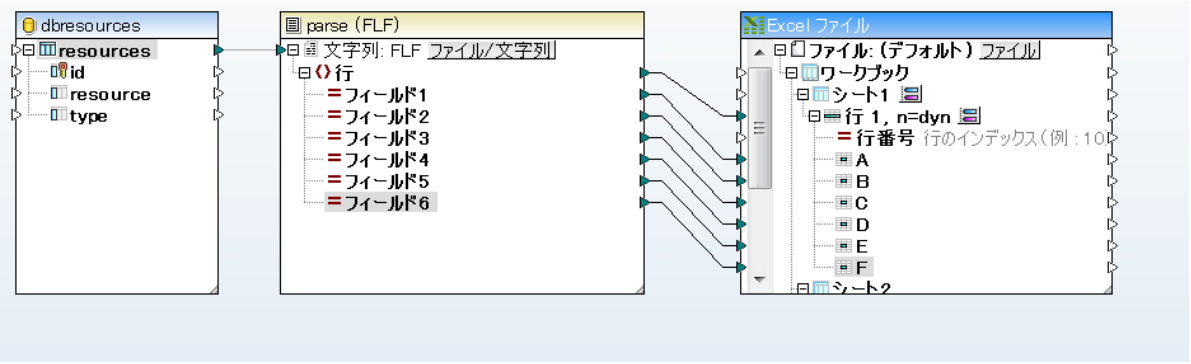


6. マッピングエリアにターゲット Excel 2007+ コンポーネントを挿入します。サンプルファイルを選択するようにプロンプトされると、「スキップ」をクリックします。(「挿入 | Excel 2007+」メニューコマンドを使用して Excel コンポーネントを追加することができます。Excel 2007+ ファイルをマッピングコンポーネントとして追加する も参照)。
7. **Row 1, n=dyn** ノードの横の ボタンをクリックし、Excel コンポーネントが、最初の列から各テキストフィールドのために行を書き込むことを構成します。(Excel 2007+ コンポーネント型とその構成 に関する詳細は、 Excel 2007+ コンポーネントについて を参照してください)。

(C) 2015-2021 Altova GmbH



8. コンポーネントアイテム間の接続の描き方は以下の通りです。



マッピングの左側では、リソースデータベース列のコンテンツが文字列の値からMapForce 構造に変換(解析) されます。マッピングの右側では、解析*(テキストファイル)* コンポーネントのアイテムが個別のExcel 列に接続され、この結果、ソース文字列を個別の並べ替えることのできるセルに分割します。

文字列データを解析し、構造を作成するMapForce デザインファイルが作成されました。「**出力**」 タブをクリックすると、レガシーテキストデータがExcel スプレッドシートの個別の行と列に変換されます。

4.14.3 例: 文字列にシリアル化 (XML からデータベースへ)

このサンプルは、データを文字列にシリアル化するマッピングデザインの作成に必要なステップを説明します。このサンプルにはサンプルファイルが存在します。このサンプルを開始する前に、サンプルファイルを確認するには、ファイルを以下のパスで検索することができます: **<マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥Tutorial¥SerializeToString.mfd**。

複数の<Person> 要素により構成されるXML ファイル(関連したスキーマ) があると仮定します。各 <Person> 要素は、それぞれの姓、名、役職、内線番号、電子メールアドレスを表します:

```
<Person>
      <First>Joe</First>
      <Last>Firstbread</Last>
      <Title>Marketing Manager Europe</Title>
      <PhoneExt>621</PhoneExt>
      <Email>j.firstbread@nanonull.com</Email>
</Person>
```

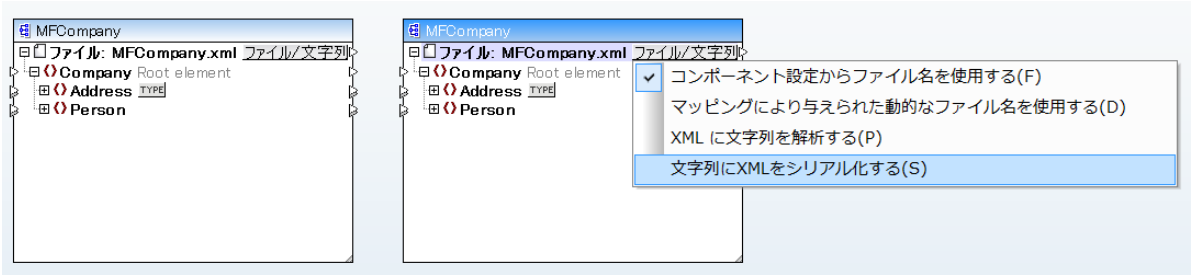
目的は、XML ファイルから各 <Person> 要素 を抽出し、リテラルに(XML タグも含め) SQLite データベースの PEOPLE 新規データベースの記録として挿入することです。PEOPLE テーブルは、2 つの列のみを含んでいます: ID とPERSON。定義は、以下の通りです:

```
CREATE TABLE PEOPLE (ID INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL, PERSON TEXT);
```

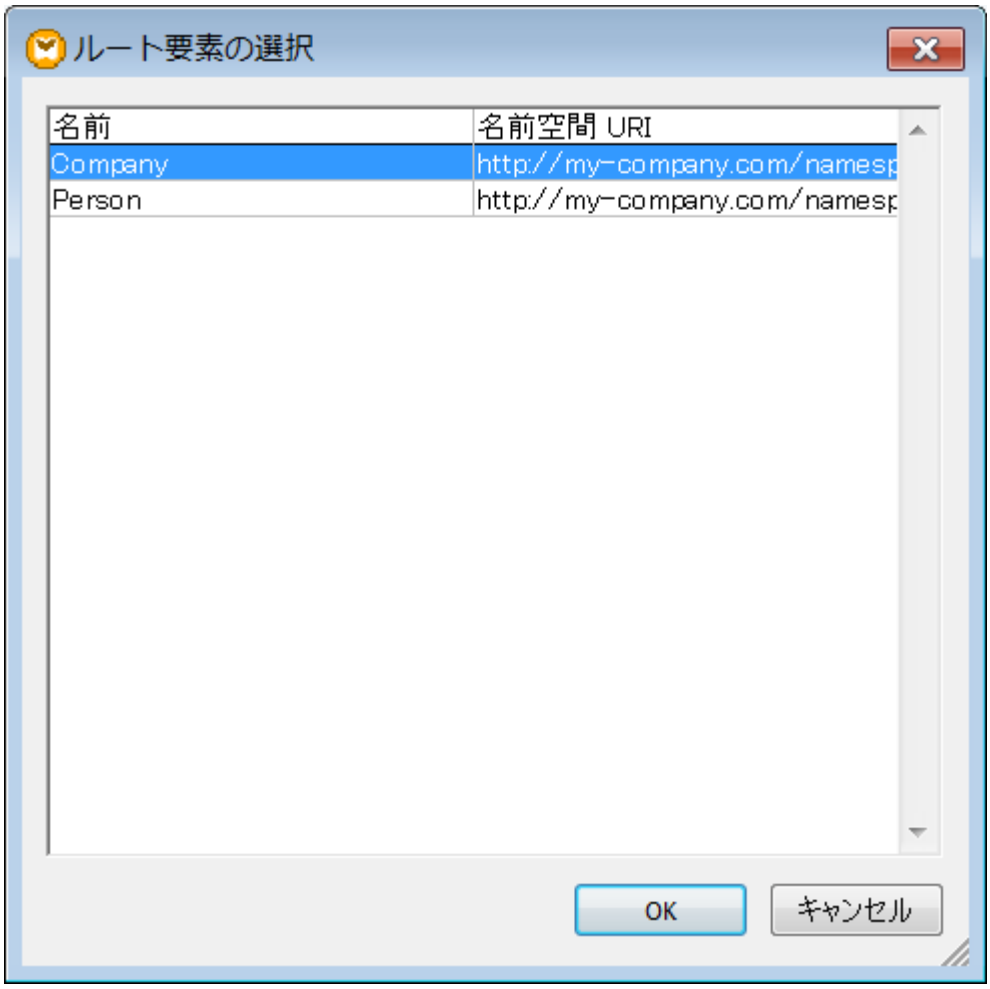
マッピングが実行された後、PEOPLE テーブルは、XML ファイル内の<Person> 要素の行数と同じ数の行を持つようになります。

このゴールを達成するには、以下を行います:

1. マッピングエリアソースXML コンポーネントに(「**挿入 | XML スキーマ/ファイル**」メニューコマンドを使用して)追加します。サンプルファイルは以下で検索することが出来ます: **<マイドキュメント >\Altova\MapForce2021\MapForceExamples\Tutorial\MFCompany.xml**。
2. (コピー—貼り付けを使用して) XML コンポーネントを複製します。
3. 複製されたXML コンポーネントで、File/String をクリックして、「**XML を文字列にシリアル化する**」 を選択します。

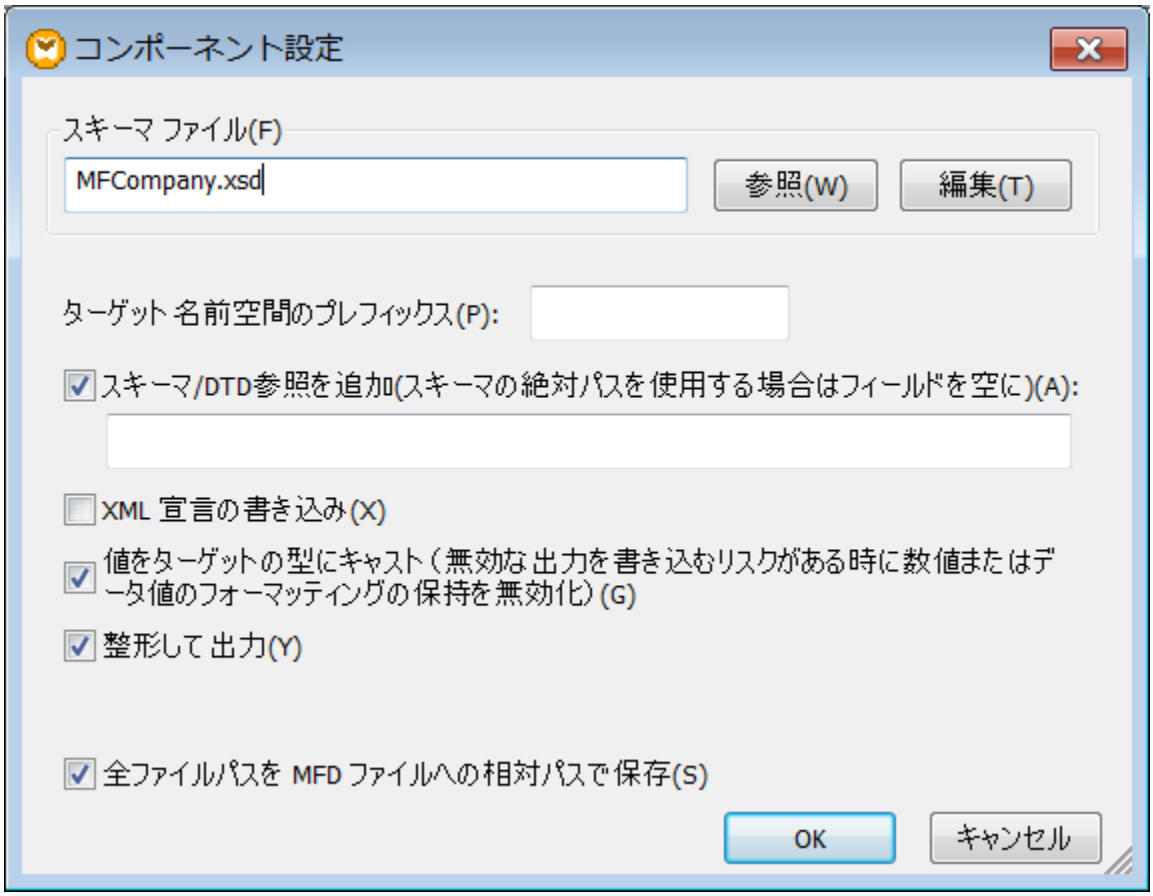


4. コンテキストメニューから、複製されたコンポーネントを右クリックして、「ルート要素の変更」を選択します。ルート要素を <Person> に変更します。

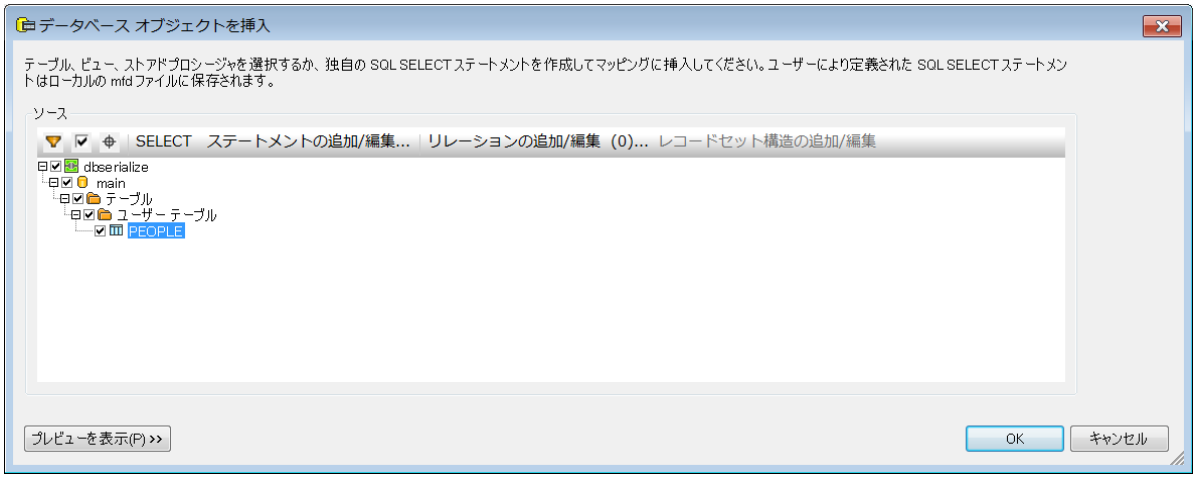


全般的に、ルート要素を XML スキーマ内で（ローカルではない）グローバル宣言を持つ要素に変更することができます。スキーマ内でグローバルに定義されていない要素は "ルート要素の選択" ダイアログボックスにリストされません。

5. コンポーネントをダブルクリックし、「XML 宣言の書き込み」チェックボックスをクリアします。これにより、XML 宣言が各 <Person> 要素のために書き込まれることを回避します。

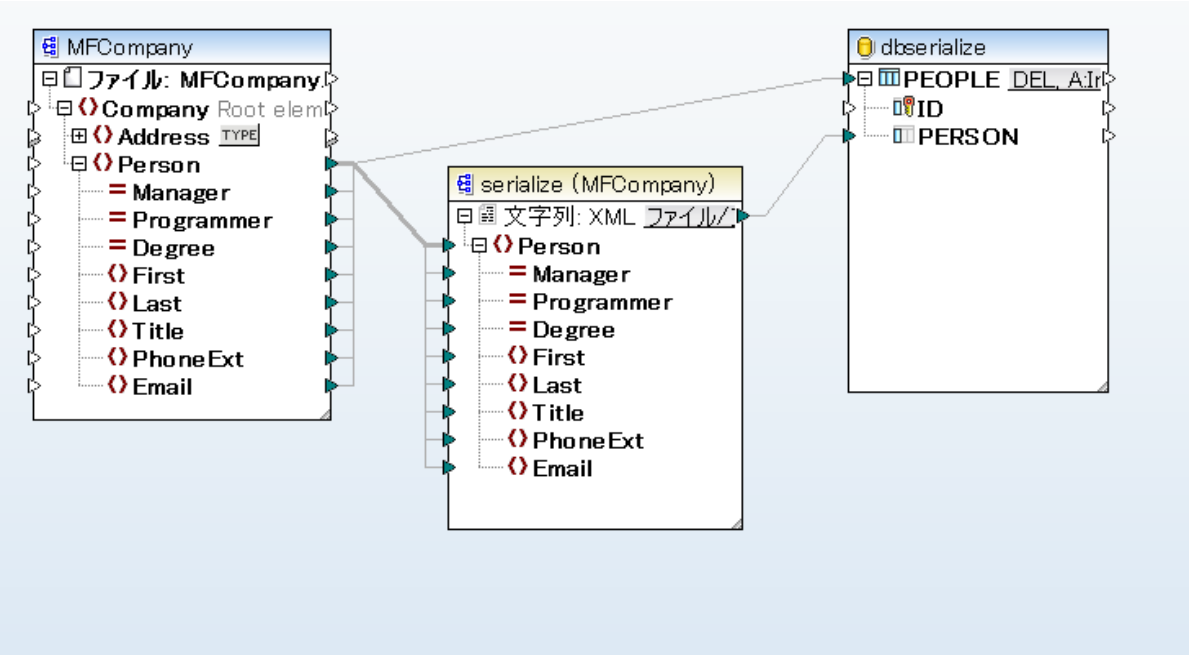


6. マッピングエリアにターゲット SQLite データベースコンポーネントを以下のパスから追加します: **<マイドキュメント>\Altova\MapForce2021\MapForceExamples\Tutorial\\dbserialize.db**。(「挿入 | データベース」メニューコマンドを使用して データベースコンポーネントを挿入します。データベースへ接続する も参照)。データベースオブジェクトを挿入するようにプロンプトされると、PEOPLE テーブルを選択します。

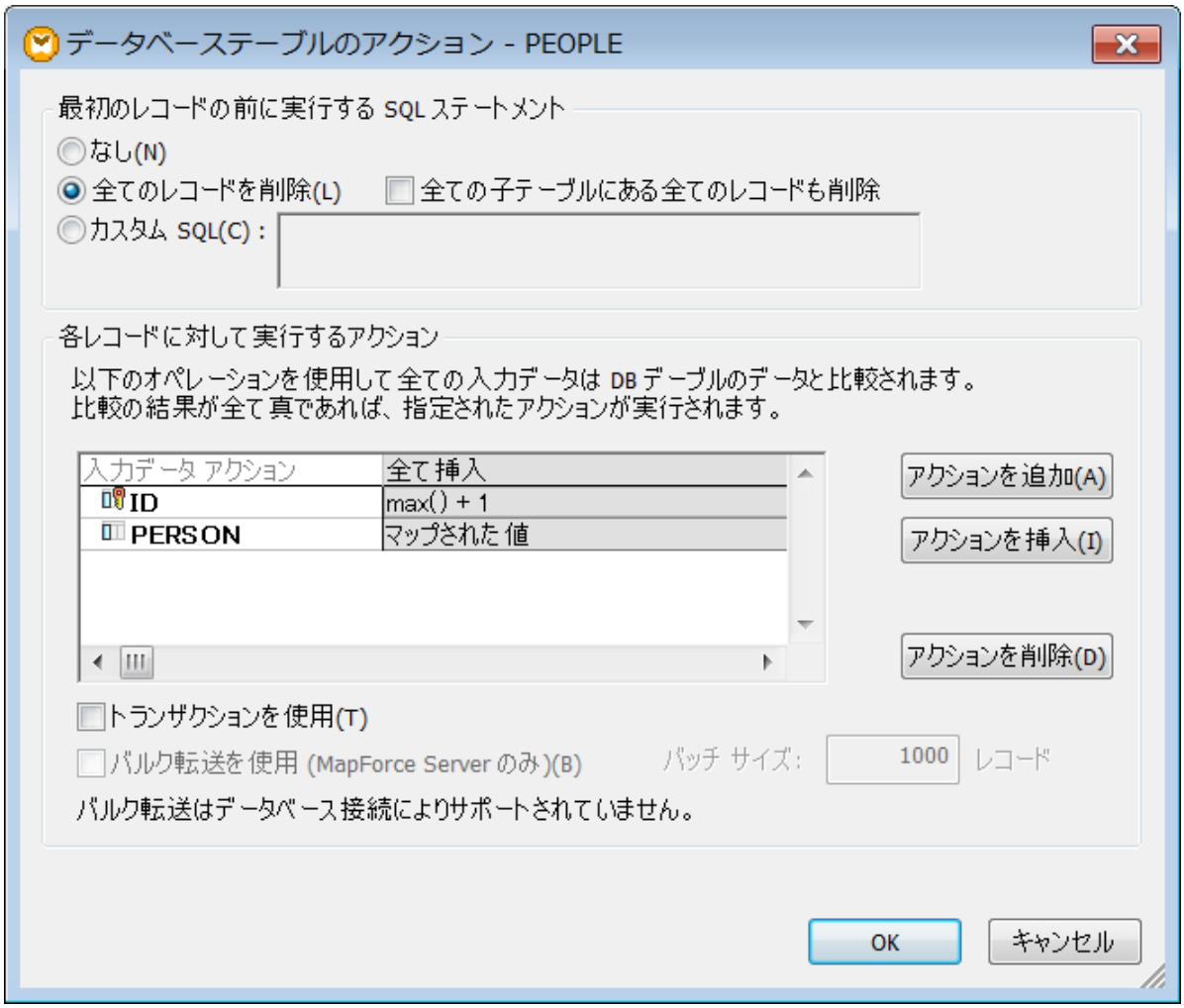


7. コンポーネントを下で表示されるように、リンクします。マッピングの左側では、<Person> 要素は、コンポーネントはシリアル化コンポーネントにマップされます。マッピングの右側では、PEOPLE データベーステーブルのシリアル化された文字列の値が PERSON 列に挿入されます。最後に、<Person> から PEOPLE テーブルへ描かれたコネクタは、MapForce に当てはまる各 <Person> 要素のために新しい記録を作成するように命令します。

(C) 2015-2021 Altova GmbH



8. データベースコンポーネントの **A:In** ボタンをクリックし、MapForce にマッピング変換が実行される都度、以下のアクションを行うように命令します:
   a. 「**全てのレコードを削除**」オプションを選択します。マッピングランタイムでは、これは既存のレコードを新規のものが挿入される前にデータベースから削除します。
   b. **ID** 列の横の「**DB-生成された**」オプションを選択します。これはレコードの ID がデータベースにより生成されることを確保します。「**DB-生成された**」オプションは列がこのオプションをサポートする場合のみ表示されます。識別、または自動インクリメントフィールドではない場合、**max+1** オプションを代わりに使用することができます。このオプションは、その列内で既存の最高値が何であるかを確認し、次に使用することのできる 1 つずつ増やされた整数を入力します。



ID 列のために選択されていた **max()+1** アクションに注意してください。これによりデータベース内の既存の ID 値の最大値を MapForce に分析するように命令します。

データを文字列にシリアル化するマッピングデザインが作成されました。「出力」 タブをクリックすると、プレビュー SQL クエリは、XML ファイル内の各 `<Person>` 要素 ために個別の記録がデータベースに挿入されることを示します。

(C) 2015-2021 Altova GmbH

4.15　ノード名のマッピング

MapForce を使用して、マッピングを作成すると、多くの目的はソースから値を読み取り、ターゲットに値を書き込むことです。しかしながら、ソースからだけではなく、ノード名から値を取得することを希望する場合があります。例えば、ソース XML から(値ではなく) 要素または属性の名前を読み取り、ターゲット XML内で(名前では無く)要素または属性の値 に変換することを希望する場合があります。

以下の例を考慮してください： XML ファイルが製品のリストを含む場合、各製品は次のフォーマットを有します:

```
<product>
   <id>1</id>
   <color>red</color>
   <size>10</size>
</product>
```

目的は、各製品の情報を名前と値のペアに変換することです。例:

```
<product>
   <attribute name="id" value="1" />
   <attribute name="color" value="red" />
   <attribute name="size" value="10" />
</product>
```

このようなシナリオには、マッピングからのノード名へのアクセスが必要になります。このトピックの主題であるノード名への動的なアクセスにより、このようなデータ変換を行うことができます。

メモ　node-name と static-node-name コアライブラリ関数を使用して、上記の変換を行うことができます。しかしながら、この場合は、ソースから期待する正確な要素名が必要であり、ターゲットに手動で要素を接続する必要があります。また、これらの関数は、十分ではない場合があります。例えば、名前別にフィルター、またはグループ分けを行う場合、また、ノードマッピングからデータ型を操作する場合などが挙げられます。

ノード名への動的なアクセスは、ノード名を読み取る場合だけではなく、書き込む必要がある場合にも使用することができます。標準マッピングでは、基となるコンポーネントのスキーマからくる、ターゲット内の属性または要素の名前は、マッピングの実行前に既知です。動的なノード名を使用して、マッピングの実行前に既知ではない、新規の属性または要素を作成することができます。具体的には、属性または要素の名前が、MapForce によりサポートされるソースからマッピング自身により与えられます。

> ノードの子要素、または属性への動的なアクセスを可能にするには、ノードが実際に子要素または属性を持つ必要があり、XML ルートノードであってはなりません。

動的なノード名は、次のコンポーネント型に、または型からマッピングをする際にサポートされています:

- XML
- CSV/FLF*

* MapForce Professional または Enterprise Edition が必要です。

メモ　CSV/FLF の場合、 CSV/FLF 構造には、ノードが存在しないため、動的なアクセスは、ノードの代わりにフィールドへのアクセスを意味します。

マッピングのターゲットがCSV またはFLF（固定長フィールド）ファイルである場合、フィールドは、コンポーネント設定内で定義されている必要があります（名前、順序、またはターゲットフィールドの数量を変更することはできません）。 XML とは異なりテキストファイルのフォーマットは固定されており、実際のフィールドの値のみを操作することができますが、フィールド名、数量、順序を操作することはできません。

ノード名への動的なアクセスは次のマッピング言語によりサポートされています: BUILT-IN、XSLT 2.0、XSLT 3.0 、XQuery、C#、C++、Java。*

*これらの言語はMapForce Professional またはEnterprise Edition を必要とします。

ノードへの動的なアクセスに関する情報は、以下を参照してください： ノード名へのアクセスを取得する。順序を追ったマッピングの例に関しては、例: 要素名を属性値にマップするを参照してください。

4.15.1 ノード名へのアクセスを取得する

XML コンポーネント（またはは、CSV/FLF コンポーネント内のフィールド）内のノードに子ノードが存在する場合、マッピングの名前と各子ノードの値をマッピングで直接取得することができます。このテクニックは、「動的なノード名」へのアクセスと呼ばれます。「動的」とは、処理がランタイム中に素早く行われることを意味し、マッピングが実行される前に既知の静的なスキーマ情報をベースに行われます。このトピックは、ノード名への動的なアクセスの有効化の方法と、その使用方法について説明します。

ソースからデータを読み取る場合、「動的なノード名」とは以下を行うことができることを意味します:

- ノードの全ての子ノード（または属性）のリストをシーケンスとして取得します。MapForce では、"シーケンス" は、ゼロのリスト、またはターゲットに接続することのできるアイテムであり、ソース内にアイテムが存在すると、ターゲットに同じ数のアイテムを作成します。ですから、例えば、ノードがソース内に5つの属性 を持つ場合、ターゲット内に属性に対応する5つの新規の要素を作成します。
- （標準マッピングが行う）子ノードの値を読み取るだけではなく、名前も読み込みます。

ターゲットにデータを書き込む場合、「動的なノード名」は、以下を行うことができることを意味します:

- コンポーネント設定（いわゆる「静的」名前）により与えられた名前とは対照的な、マッピング（いわゆる「動的」名前）により与えられる名前を使用して、新しいノードを作成します。

動的なノード名を説明するために、このトピックは次のXML スキーマを使用しています: **<マイドキュメント>\Altova\MapForce2021\MapForceExamples\Tutorial\Products.xsd**。このスキーマは、サンプルインスタンスドキュメント **Products.xml** を伴っています。スキーマとインスタンスファイルをマッピングエリアに追加するには、「**挿入**| XML スキーマ/ファイル」メニューコマンドを選択して、<**マイドキュメント**>¥Altova¥MapForce2021¥MapForceExamples¥Tutorial¥Products.xml を参照してください。ルート要素を選択するように問われると、`products` を選択します。

`product` ノードのための動的なノード名を有効化するには、右クリックし、次のコンテキストメニューコマンドの1つを選択します:

- ノードの属性にアクセスする場合、「**動的な名前を持つ属性を表示**」
- ノードの子要素にアクセスする場合、「**動的な名前を持つ子要素を表示**」

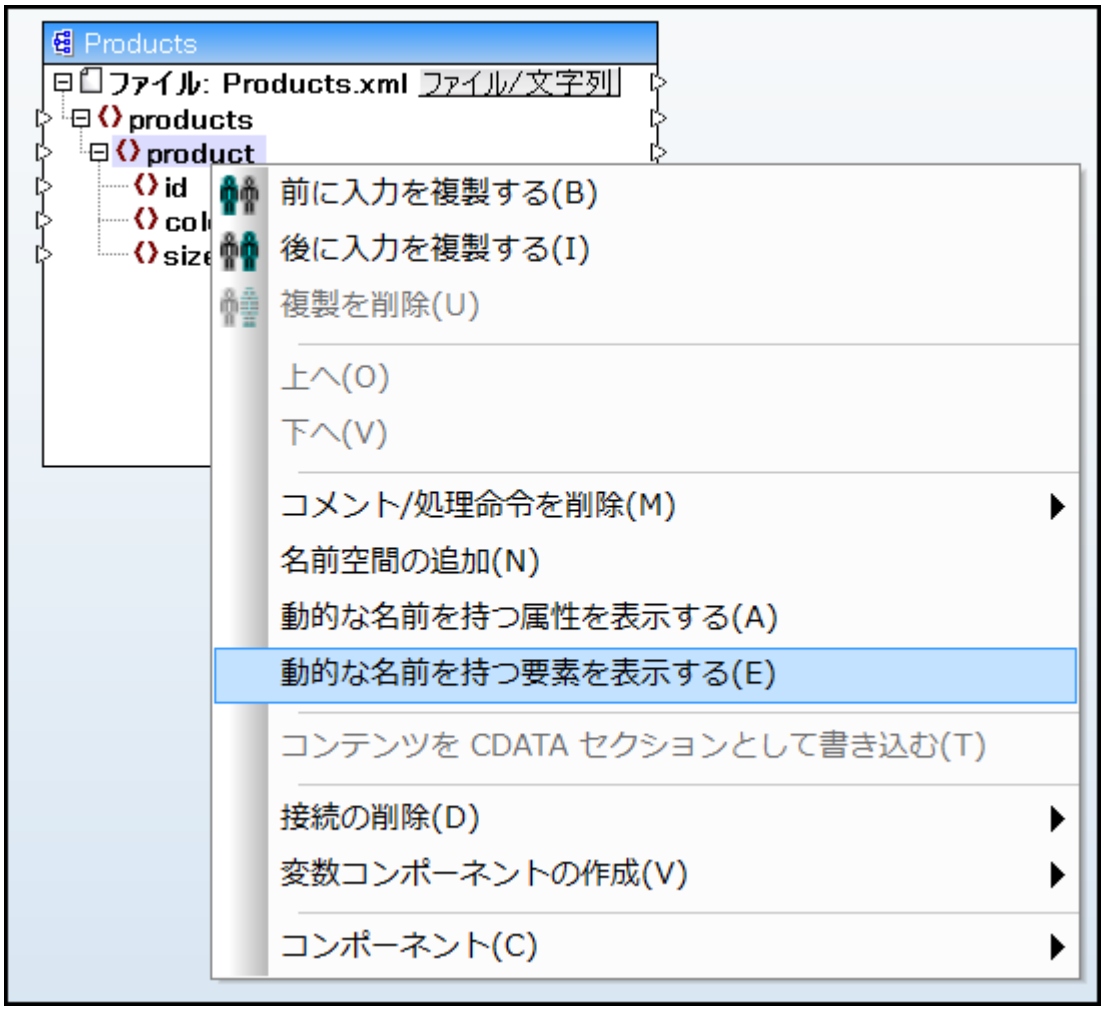


図. 1　(子要素のための) 動的なノード名の有効化

メモ　上のコマンドは、子ノードを持つノードのみに対して使用することができます。また、コマンドはアトミックノードには使用することができません。

ノードを動的なモードに切り替えると、下に表示されるダイアログボックスに類似したダイアログボックスが表示されます。このトピックの目的のために、オプションを下に表示されるように設定します。これらのオプションは特定の型のノードにアクセスする内で説明されています。

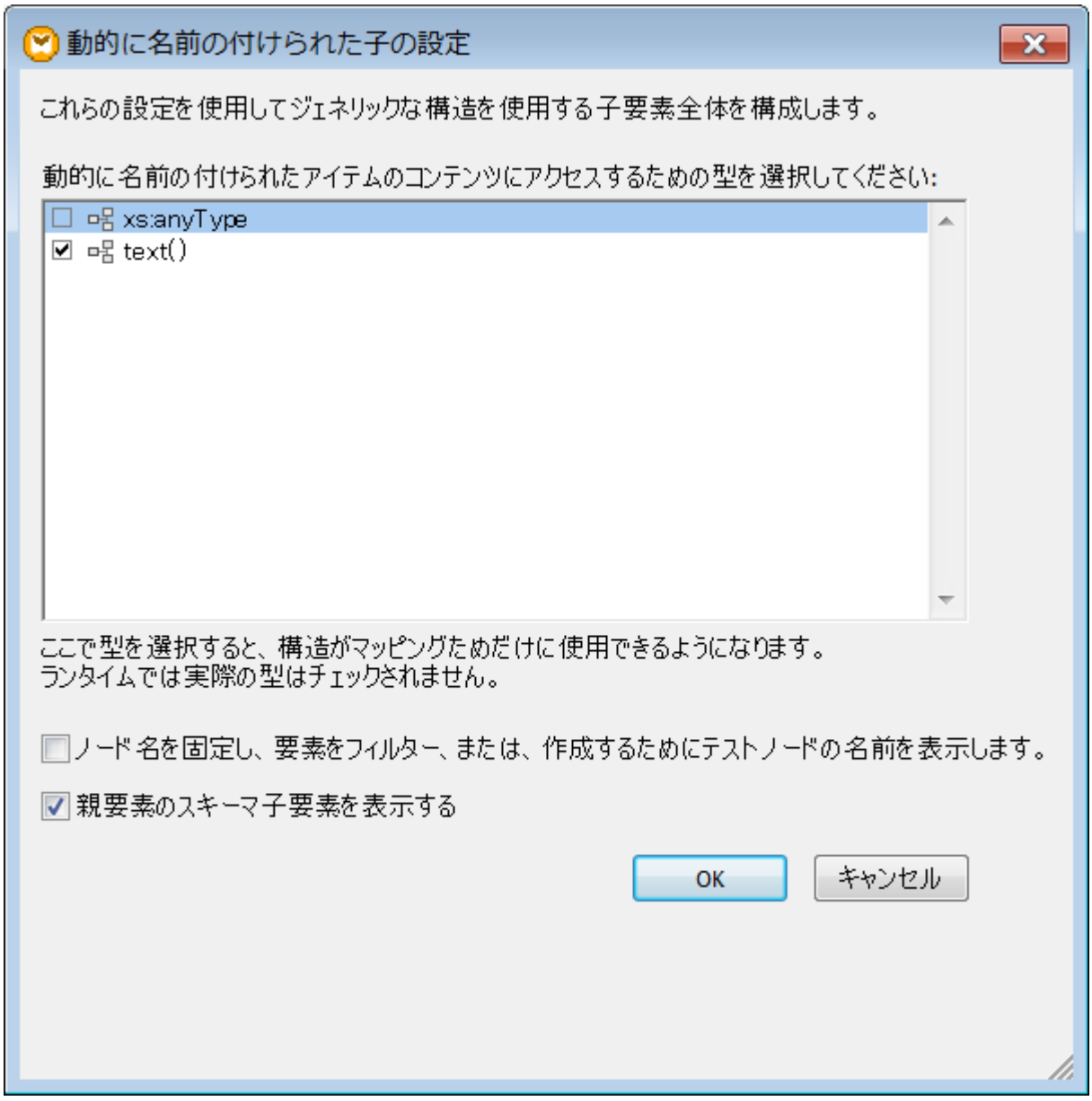


図 2 「動的に名前がつけられた子の設定」ダイアログボックス

図 3 は、`product` ノードのために動的なノード名が有効化されている場合、どのようにコンポーネントが表示されるかを示しています。コンポーネントの外観が大幅に変更されていることにご注意してください。

(C) 2015-2021 Altova GmbH

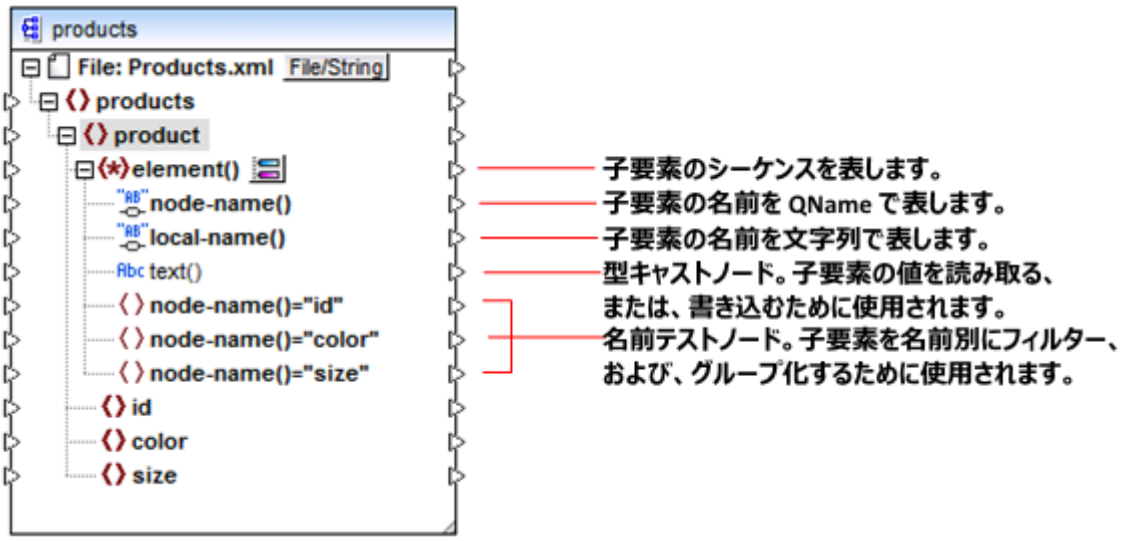


図3　（要素のために）動的なノード名が有効化されている場合

コンポーネントを標準モードに切り替えるには、`product` ノードを右クリックして、コンテキストメニューからオプション「動的な名前を持つ子要素を表示する」を無効化します。

下のイメージは、ノードの属性への動的なアクセスが有効化された時、同じコンポーネントがどのように表示されるかを示しています。`product` 要素を右クリックして、コンポーネントが取得され、コンテキストメニューから「動的な名前を持つ属性を表示する」を選択します。

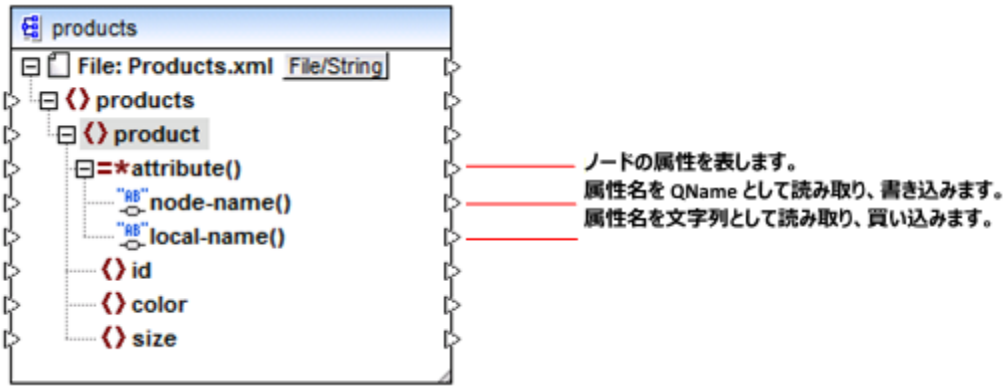


図 4　（属性のために）動的なノード名が有効化されている場合

コンポーネントを標準モードに切り替えるには、`product` ノードを右クリックして、コンテキストメニューから、「動的な名前を持つ属性を表示する」オプションを無効化します。

図 3 と図 4 に示されているように、コンポーネントの外観はノード（この場合は、`product`）が「動的なノード名」なモードに切り替えられると変更されます。新しい外観は、次のアクションを取る可能性を広げます:

- すべてのノードの子要素または属性のリストを読み取り、または書き込みます。これらは、`element()` または `attribute()` アイテムによりそれぞれ提供されます。
- 各子要素または属性の名前を読み取り、または書き込むます。名前は、`node-name()` と `local-name()` アイテムにより提供されます。

- 要素の場合、各子要素の値を特有のデータ型として読み取る、または書き込みます。この値は、型キャストノード（この場合は、`xs:string` アイテム）により与えられます。要素のみが型キャストノードを持つことができることに注意してください。属性は、常に、「文字列」型として処理されます。
- 名前別に子要素をグループ化します。例に関しては次を参照してください：例: 名前別にノードをグループ化またはフィルターする。

「動的なノード名」モード内で作業することのできるノード型は、下で説明されています。

element()

ターゲットコンポーネントと比較すると、このノードはソースコンポーネント内で異なる振る舞いをします。ソースコンポーネント内で、ノードの子要素をシーケンスとして提供します。図 3 `element()` は、全ての `product` の子要素のリスト（シーケンス）を与えています。例えば、次の XML から作成されたシーケンスは、3 つのアイテムを含みます（これは、`product` 3 つのつの子要素が存在するからです）:

```
<product>
   <id>1</id>
   <color>red</color>
   <size>10</size>
</product>
```

シーケンス内の各アイテムの実際の名前と型は、`node-name()` ノードと型キャストノードによりそれぞれ与えられています 上記を理解するには、ソースからのデータ XML をターゲット XML に次のように変換する必要があると仮定してください：

図 6 シーケンス内の各アイテムの実際の名前と型は、`node-name()` ノードと型キャストノードによりそれぞれ与えられています 上記を理解するには、ソースからのデータ XML をターゲット XML に次のように変換する必要があると仮定してください：

マッピングの目的を達成するマッピングは以下のようになりますマッピング:

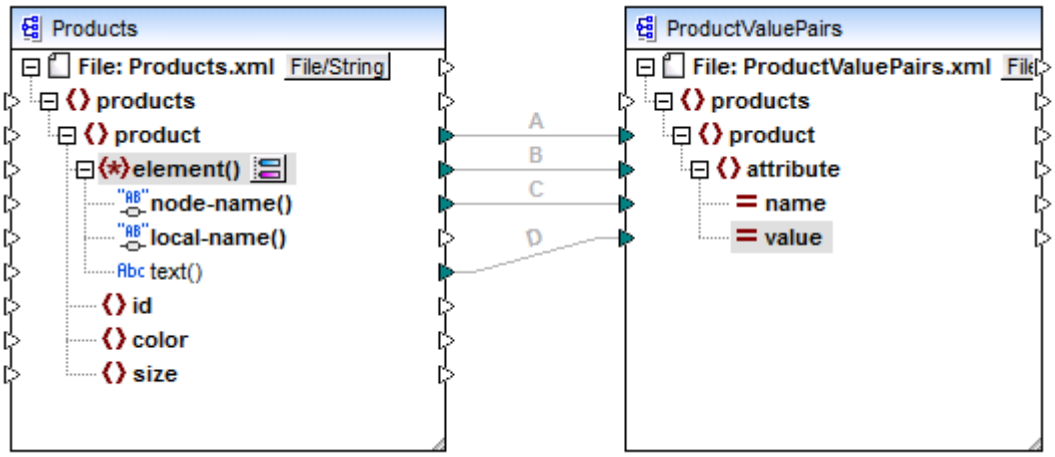


図 7　(MapForce 内で) XML 要素名 を属性値 にマッピングする

node-name() と text() は、シーケンス内の各アイテムの値と実際の名前を与えますが、element() のここでの役割は、product の子要素のシーケンスを提供します。このマッピングは、チュートリアルのサンプルが付随しており、以下で更に詳しく説明されています: 例: 要素名を属性値にマップする。

ターゲットコンポーネント内では、element() は、ソース内の各アイテムのマッピングの基本ルールの例外であるとおり自身では何も作成しませんが、1 つのターゲットアイテムを作成します。実際の要素 は、(node-name() の値を使用して) 型キャストノード と自身の名前を使用して) 名前テストノードにより作成されます。

attribute()

図 4 内で表示されているように、このアイテムは、マッピングのランタイムでノードの全ての属性へのアクセスを有効化します。ソースコンポーネント内で、接続されたソースノードの属性をシーケンスとして提供します。例えば、次の XML 内で、シーケンスは、(product には 2 つの属性が存在するため) 2 つのアイテムを含みます:

```
<product id="1" color="red" />
```

attribute() ノードは、シーケンス内の各属性の値のみを文字列の型として与えます。各属性の名前は node-name() ノードにより与えられます。

ターゲットコンポーネント内では、このノードは、接続されたシーケンスを処理し、シーケンス内の各アイテムのために属性の値を作成します。属性の名前は、node-name() により与えられます。例えば、ソースからのデータ XML を ターゲット XML を以下のように処理すると想定します:

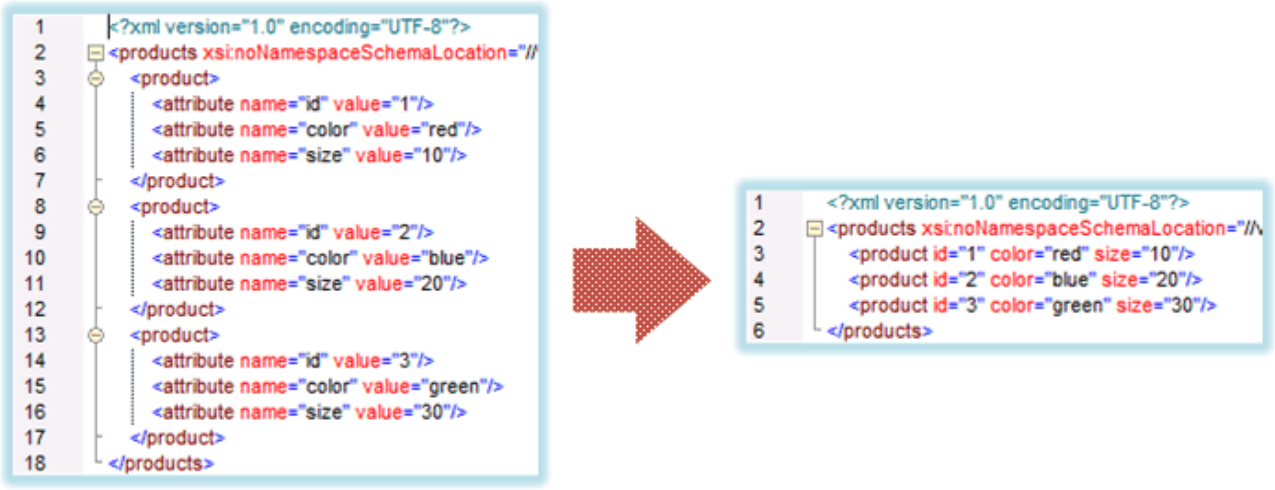


図 8 属性の値を属性名にマッピングする(必須)

この目的を達成するマッピングは以下のようになります:

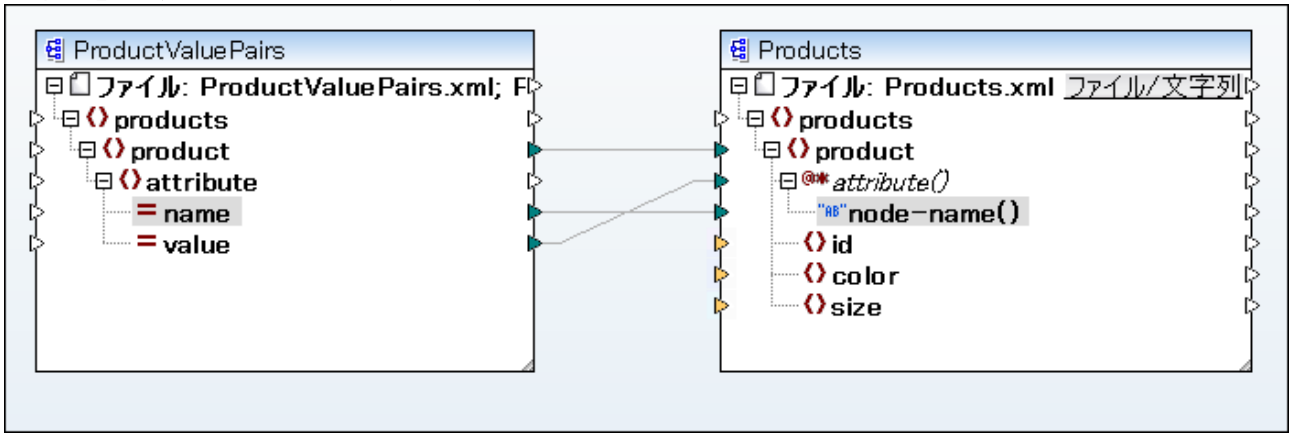


図 9 (MapForce 内で)属性の値を属性名にマッピングする

メモ この変換は、ノードの属性へのアクセスを有効化せずに行うことができます。ここでは、どのように`attribute()`がターゲットコンポーネント内で動作するかを表示しています。

このマッピングを再作成する場合は、<マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥Tutorial¥フォルダー内の**ConvertProducts.mfd** マッピングと同じ XML コンポーネントを使用しています。マッピングとして使用します。唯一の違いは、ターゲットがソースになり、ソースがターゲットになることのみです。ソースコンポーネントのための入力データには、属性の値を実際に含む XML インスタンスが必要になります。例:

```
<?xml version="1.0" encoding="UTF-8"?>
<products>
   <product>
      <attribute name="id" value="1"/>
      <attribute name="color" value="red"/>
      <attribute name="size" value="big"/>
   </product>
</products>
```

上記のコードリスティングでは、名前空間とスキーマ宣言が、ここでは簡単にするために省略されていることにご注意してください。

node-name()

ソースコンポーネント内では、node-name() は、element() の子要素の名前、または attribute() の要素の名前をそれぞれ提供します。デフォルトでは、提供される名前は、xs:QName の型です。名前を文字列として取得するには、 local-name() ノードを（図 3 を参照してください）、または、関数 QName-as-string を使用します。

ターゲットコンポーネント内では node-name() は、element() または attribute() 内に含まれる各要素または属性の名前を書き込みます。

local-name()

このノードは、node-name() と同様の作動をしますが、異なる点は、xs:QName の代わりに xs:string が使用されていることです。

型キャストノード

ソースコンポーネント内では、型キャストノードは、element() 内の各子要素の値を提供しますこのノードの名前と構造は、「動的に名前がつけられた子の設定」ダイアログボックス（図 2）から選択された型により異なります。

ノードの型を変更する場合は、「**選択の変更**」( ) ボタンを使用して、スキーマワイルドカード（xs:any）を含む、使用できる型から希望する型を選択します。詳細に関しては、次を参照してください： 特定の型のノードにアクセスする。

ターゲットコンポーネント内で、型キャストノードは、element() 内に含まれる各子要素の値特有のデータ型として書き込みます。希望するデータ型は「**選択の変更**」( ) ボタンをクリックすることにより選択することができます

名前テストノード

ソースコンポーネント内では、名前テストノードは、ソースインスタンスから名前別に子要素をグループ化、またはフィルターする方法が提供されています。正確な型を使用してマッピングが、インスタンスデータにアクセスしていることを保証するために子要素を名前別にフィルターする必要がある場合があります。（次を参照してください： 特定の型のノードへのアクセス）。例えば、例: 名前別にノードをグループ化またはフィルターするを参照してください。

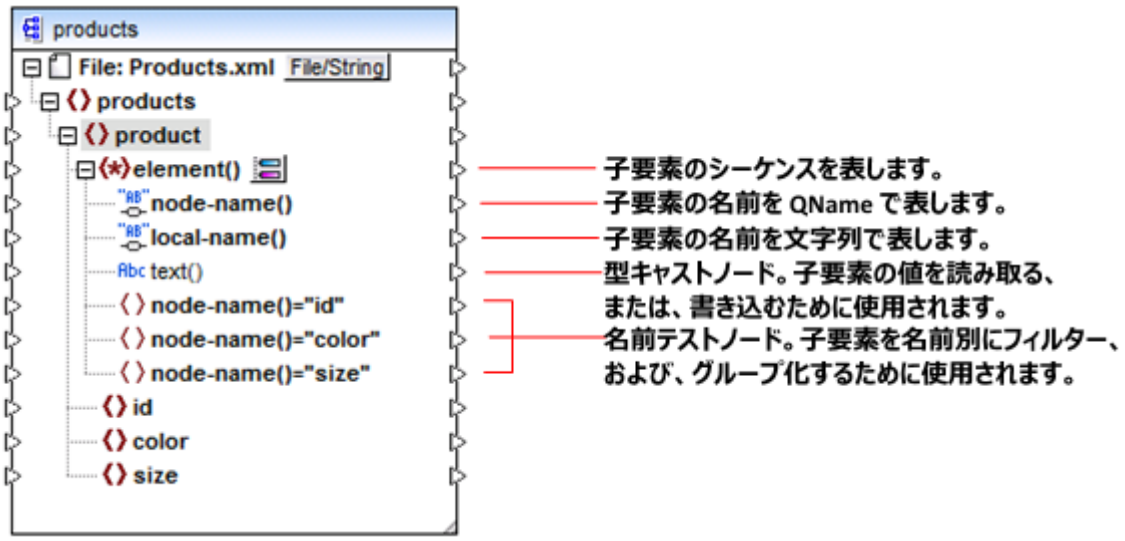
一般的には、名前テストノードは、値とサブ構造の読み取りと書き込みをする通常の要素ノードとほぼ同様の作動をします。しかしながら、動的なアクセスが有効化されていると、マッピングのセマンティクスが異なるため、制限が発生します。例えば、2 つの名前テストノードを連結することはできません。

ターゲット側では、名前テストノードは、接続されているソースシーケンスにアイテムが存在するため、出力内と同じ数量の要素を作成します。これらの名前は、node-name() にマップされている値を上書きします。

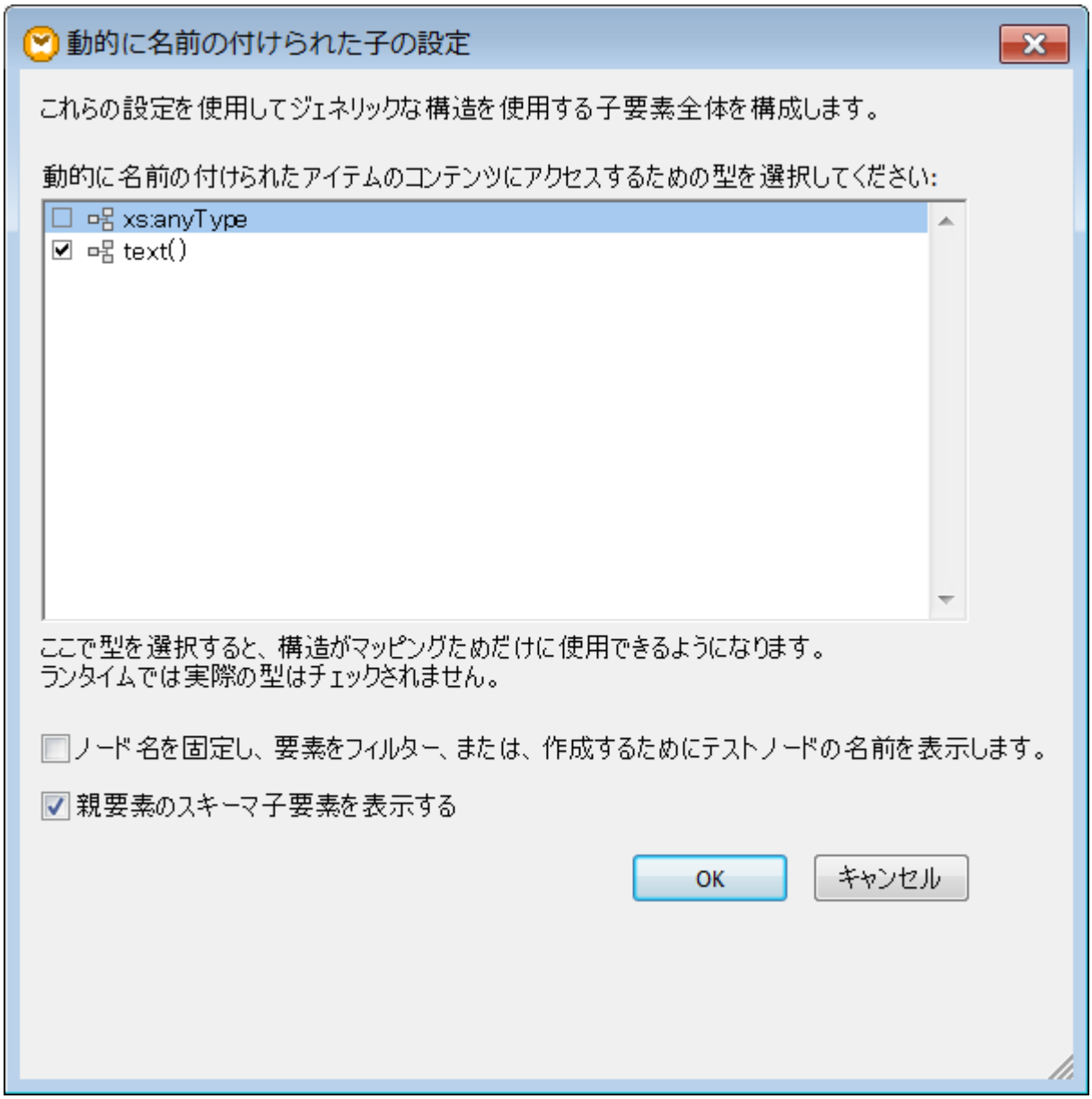
必要であれば、コンポーネントから名前テストノードを非表示にすることができます。これを行うには、 element() ノードの横の「**選択の変更**」( ) ボタン をクリックします。そして、「動的な名前を持つ子の設定」ダイアログボックスから、「**名前テストノードの表示**」チェックボックスのチェックを解除します。

4.15.2　特定の型のノードへのアクセス

前のセクションで説明されているとおり、ノード名へのアクセスの取得 は、ノードを右クリックして「**動的な名前を持つ子要素を表示する**」コンテキストメニューコマンドを選択し、ノードの全ての子要素にアクセスすることができます。特定の型のノードにより値をアクセスすることはできますが、マッピングのランタイムでは、 node-name() ノードから各子要素の名前にアクセスすることができます。下のイメージでは、 型キャストノードは、text() ノードの横にあります。



子要素のデータ型は、マッピングのランタイム前にはわかりません。 また、各子要素により異なる可能性があります。例えば、XML インスタンスファイル内の `product` ノード は、型 `xs:integer` の子要素 `id` と型 `xs:string` の子要素 `size` を持つ可能性があります。特定の型のノードコンテンツにアクセスするには、下に表示されるダイアログボックスがノードの子要素への動的なアクセスを有効化するたびに開かれます。`element()` ノードの横の「**選択の変更**」( ) ボタンをクリックすることにより、このダイアログボックスを後で開くことができます。



「動的に名前がつけられた子の設定」ダイアログボックス

マッピングのランタイムに各子要素のコンテンツにアクセスするには、いくつかのオプションがあります:

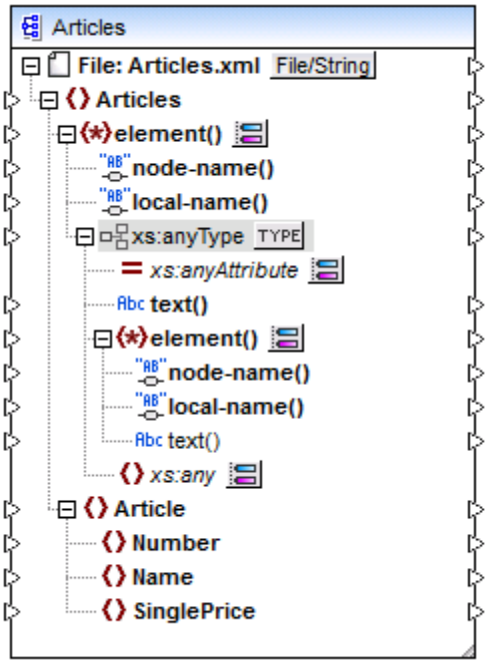
1. コンテンツを文字列としてアクセスするには、上のダイアログボックスで **text()** チェックボックスを選択します。この場合、ダイアログボックスが閉じられると、`text()` ノードがコンポーネント上に作成されます。このオプションは、コンテンツが `xs:int`、`xs:string` などの単純型 の場合、適切です。例: 要素名を属性値にマップするで詳しく説明されています。**text()** ノードは、現在のノードの子ノードがテキストを含むことができる場合表示されることに注意してください。
2. スキーマに許可されている特定の複合型としてコンテンツにアクセスすることができます。カスタム複合型が、選択されているのためにノードグローバルに許可されていると、上のダイアログボックスで使用することが可能になり、横のチェックボックスを選択することができます。上のイメージでは、グローバルに定義されている複合型は存在せず、この選択は使用することができません。
3. コンテンツを型としてアクセスする。これは、高度のマッピングのシナリオで役に立ちます（次を参照してください「更に深い構成にアクセスする」を参照してください）。これを行うには、**xs:anyType** の横のチェックボックスを選択してください。

マッピング ランタイムでは、MapForce（型 キャスト ノードを介して）には、実際のインスタンス ノード型に関する情報がないことにご注意してください。ですから、マッピングは、正確な型を使用してノード コンテンツにアクセスする必要があります。例えば、ソース XML インスタンスが多種の複合型の子ノードを持つことを希望する場合、以下を行います:
a）型キャスト ノードを一致させる必要が有る複合型に設定します（上のリストの2番目のアイテムを参照してください）。
b）一致する必要のあるインスタンスのみから読み込むためにフィルターを追加します。このテクニックは、次で説明されています: 例: 名前別にノードをグループ化またはフィルターする.

更に深い構成にアクセスする

スキーマ ワイルドカードの選択により、ノードをノードの直下の子よりもスキーマ内の更に深いレベルでアクセスすることができます。高度なマッピングのシナリオではとても役に立ちます。マッピングは、XML ノードの直下の子しかアクセスすることができないため、例: 要素名を属性値にマップする 等の簡単なマッピングでは、このテクニックを必要としません。しかしながら、動的に更に深い構造にアクセスする必要がある場合、例えば「孫」にアクセスが必要な場合、以下の方法でアクセスすることが可能です。

1. 新規マッピングを作成する。
2. 挿入メニューから「XML スキーマ/ファイルを挿入する」をクリックし、XML インスタンス ファイルを参照します。（この例では、<マイ ドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥Tutorial¥ フォルダーからの **Articles.xml** ファイルです）。
3. `Articles` ノードを右クリックして、「**動的な名前を持つ子要素を表示する**」 コンテキスト コマンドを選択します。
4. 「動的に名前がつけられた子の設定」ダイアログ ボックスから **xs:anyType** を選択します。
5. `xs:anyType` ノードを右クリックして、「**動的な名前を持つ子要素を表示する**」コンテキスト コマンドに対して選択します。
6. 「動的に名前がつけられた子の設定」ダイアログ ボックスから **text()** を選択します。



上記のコンポーネントで、2つの `element()` ノードが存在することにご注目してください。2番目の `element()` ノードは、**Articles.xml** インスタンス内の `<Articles>` ノードの孫に動的なアクセスを与えます。

```
<?xml version="1.0" encoding="UTF-8"?>
<Articles xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="Articles.xsd">
```

```
   <Article>
      <Number>1</Number>
      <Name>T-Shirt</Name>
      <SinglePrice>25</SinglePrice>
   </Article>
   <Article>
      <Number>2</Number>
      <Name>Socks</Name>
      <SinglePrice>2.30</SinglePrice>
   </Article>
   <Article>
      <Number>3</Number>
      <Name>Pants</Name>
      <SinglePrice>34</SinglePrice>
   </Article>
   <Article>
      <Number>4</Number>
      <Name>Jacket</Name>
      <SinglePrice>57.50</SinglePrice>
   </Article>
</Articles>
```

Articles.xml

例えば、「孫」要素名（Number、Name、SinglePrice）を取得するには、2番目のelement()ノードの下のlocal-name()ノードからターゲットノードに接続を描きます。同様に、「孫」要素の値（1、T-Shirt、25）を取得するには、text()ノードから接続を描きます。

この例には適用することはできませんが、実際のシナリオでは、更に深いレベルにアクセスするために、動的なノード名を後のxs:anyTypeノードに対して有効化することができます。

以下の点に注意してください：

- TYPE ボタンを使用して、生成された型を現在のスキーマから選択し、異なるノード内に表示することができます。この方法は、生成されたスキーマの型から、またはスキーマの型へマップする必要がある場合のみに役に立ちます（次を参照してください：生成されたXML スキーマ型）。
- element()ノードの横の「選択の変更」（ ）ボタンは、このトピックで説明されている「動的に名前の付けられた子の設定」ダイアログボックスを開きます。
- xs:anyAttribute 属性の横の「選択の変更」（ ）ボタンにより、スキーマ内でグローバルに定義された属性を選択することができます。同様に、 xs:any 要素の横の「選択の変更」（ ）によりスキーマ内でグローバルに定義されている要素を選択することができます。これにより、スキーマワイルドカードへ、またはスキーマワイルドカードからのマッピングと同じように動作します（次も参照してください：ワイルドカード – xs:any / xs:anyAttribute）。このオプションを使用する場合、選択された属性または要素がスキーマに従い特定のレベルで存在できることを確認してください。

4.15.3　例: 要素名を属性値にマップする

この例は、XML ドキュメントから、要素名をターゲット XML ドキュメント内の属性の値にマップする方法について説明しています。この例には、付随するサンプルマッピングがあり、次のパスを使用して検索することができます：**<マイドキュメント>\Altova\MapForce2021\MapForceExamples\Tutorial\ConvertProducts.mfd**。

例を理解するために、XML ファイルは製品のリストを含むと想定しましょう。各製品は次のフォーマットを有します：

```
   <product>
```

```
    <id>1</id>
    <color>red</color>
    <size>10</size>
  </product>
```

目的は、各製品の情報を名前と値ペアに変換することです。例:

```
  <product>
    <attribute name="id" value="1" />
    <attribute name="color" value="red" />
    <attribute name="size" value="10" />
  </product>
```

上記のようなデータマッピングを行うには、この例でも使用されていますが、"ノード名への動的なアクセス"と呼ばれる MapForce 機能を使用しています。マッピングを実行する際、「動的」とは、(値だけではなく)ノード名も読み取ることができ、これらの名前を値として使用することを意味します。必要とするマッピングをいくつかのシンプルなステップを以下に示されるように作成することができます。

ステップ 1: ソース XML コンポーネントをマッピングに追加する

- 「挿入」メニューから、「XML スキーマ/ファイル」をクリックして、<マイドキュメント>**\Altova\MapForce2021\MapForceExamples\Tutorial\Products.xml** を参照します。この XML ファイルは、同じフォルダー内にある **Products.xsd** スキーマを指します。

ステップ 2: ターゲット XML コンポーネントをマッピングに追加する

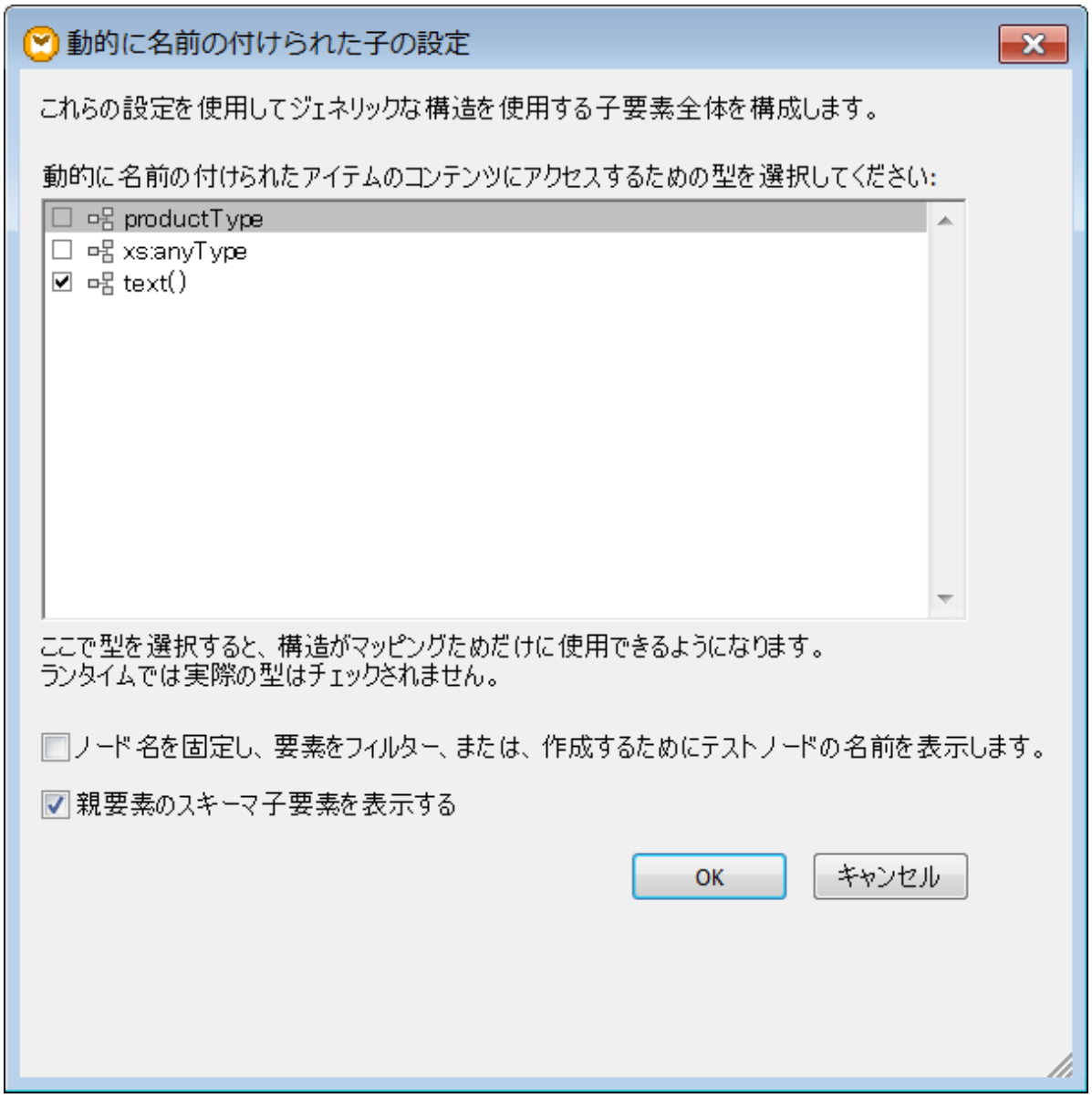
- 「挿入」メニューから、「XML スキーマ/ファイル」をクリックして、次のファイルを参照します: <マイドキュメント>**\Altova\MapForce2021\MapForceExamples\Tutorial\ProductValuePairs.xsd**。インスタンスファイルを与えるように問われると、「スキップ」をクリックします。ルート要素を選択するようにプロンプトされると、ルート要素として、`products` を選択します。

この時点では、マッピングは、以下のようになります:

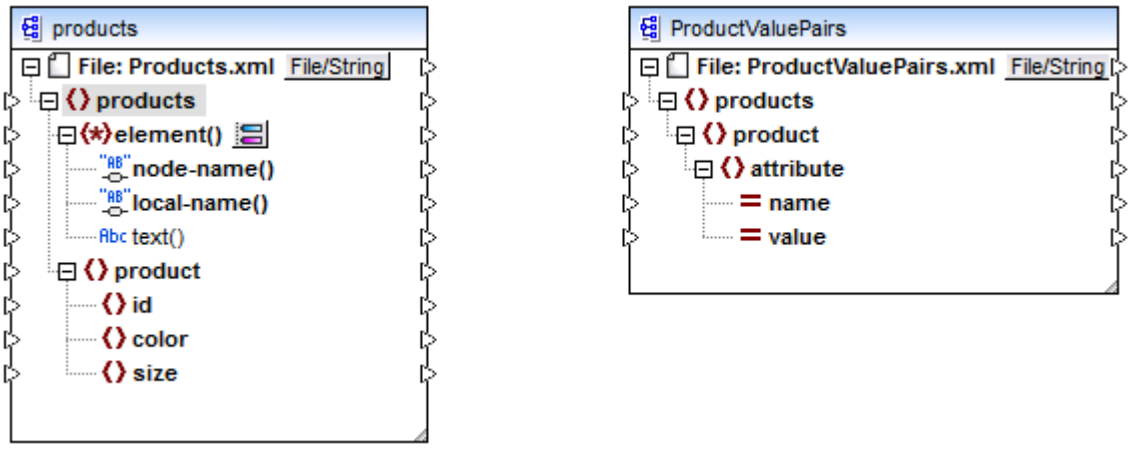
ステップ 3: 子ノードへの動的なアクセスを有効化する

1. `products` ノードを右クリックして、コンテキストメニューから、「**動的な名前を持つ子要素を表示する**」を選択します。
2. 開かれるダイアログボックス内から、**text()** を型として選択します。他のオプションをそのままにします。

(C) 2015-2021 Altova GmbH

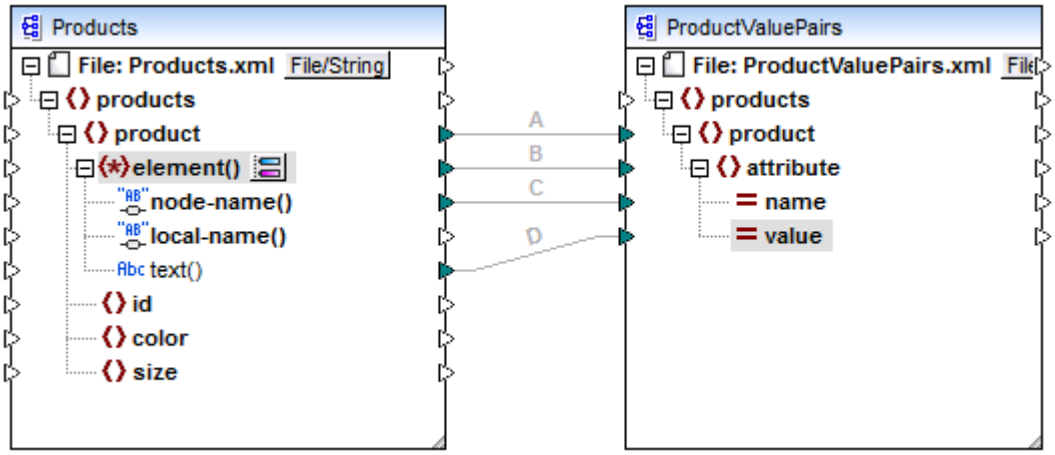


`text()` ノードがソースコンポーネントに追加されていることにご注意してください。このノードは、それぞれの子アイテムをマッピングに与えます（この場合は、「id」、「color」、「size」の値です）。



ステップ 4: マッピング 接続を描く

マッピング 接続 A、B、C、D 下に表示されるとおり、描きます。任意で、各接続上からをダブルクリックし、テキスト「A」、「B」「C」、および「D」をそれぞれ、詳細ボックスに入力します。



ConvertProducts.mfd

上で説明されているマッピングでは、接続 A は、ソース内の各製品のために、ターゲット内に製品を作成します。これまでは、これはノード名を指さない標準的な MapForce 接続でした。しかしながら、接続 B は、`product` の各対応する子要素のために、新規の要素を `attribute` と呼ばれるターゲット内に作成します。

> 接続 B は、マッピング内でとても重要です。この接続の目的である `product` の子要素のシーケンスを、ソースからターゲットに運びます。実際の名前または値を運ばないため、以下として理解される必要があります: ソース **element()** に N 子要素が存在する場合、ターゲット内のアイテムの N インスタンスを作成します。この特定の場合は、ソース内の `product` には、3 つの子要素 (`id`、`color` と `size`) が存在します。これは、ターゲット内の各 `product` が `attribute` という名前の 3 つの子要素を持つことを意味します。
>
> この例で説明されていませんが、同じルールが、**attribute()** の子要素をマップするために使用されます。ソース **attribute()** アイテムが、N 子属性を持つ場合、ターゲット内のそのアイテムの N インスタンスを作成します。

次に、接続 C は、`product` の各子要素の実際の名前をターゲットにコピーします (文字通り、「id」、「color」、と「size」)。

(C) 2015-2021 Altova GmbH

最後に、接続 D は、製品の各子要素の値を文字列の型として、ターゲットにコピーします。

マッピングの出力をプレビューするには、「出力」タブをクリックして、生成された XML を確認してください。マッピングの意図とする目的である通り、出力は、データが名前と値のペアとして保管されている、複数の製品を含んでいます。

```
<?xml version="1.0" encoding="UTF-8"?>
<products xsi:noNamespaceSchemaLocation="ProductValuePairs.xsd"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <product>
      <attribute name="id" value="1"/>
      <attribute name="color" value="red"/>
      <attribute name="size" value="10"/>
   </product>
   <product>
      <attribute name="id" value="2"/>
      <attribute name="color" value="blue"/>
      <attribute name="size" value="20"/>
   </product>
   <product>
      <attribute name="id" value="3"/>
      <attribute name="color" value="green"/>
      <attribute name="size" value="30"/>
   </product>
</products>
```

生成されたマッピング出力

4.15.4　例: 名前別にノードをグループ化またはフィルターする

この例は、XML プロパティリスト（または XML プロパティリスト）からキー値ペアを読み取り CSV ファイルへ書き込むマッピングをデザインする方法を説明します。（XML プロパティリストは、macOS と iOS オブジェクトの情報を XML フォーマットで保存する方法を示します。詳細に関しては、次を参照してください：
https://developer.apple.com/library/mac/documentation/Cocoa/Conceptual/PropertyLists/UnderstandXMLPlist/UnderstandXMLPlist.html。）このサンプルは、マッピングサンプルが伴い、次のパスで検索することができます: **<マイドキュメント>\Altova\MapForce2021\MapForceExamples\Tutorial\ReadPropertyList.mfd**。

下のコードリストはソース XML ファイルを表しています。

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist SYSTEM "https://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
   <dict>
      <key>First Name</key>
      <string>William</string>
      <key>Last Name</key>
      <string>Shakespeare</string>
      <key>Birthdate</key>
      <integer>1564</integer>
      <key>Profession</key>
      <string>Playwright</string>
      <key>Lines</key>
      <array>
         <string>It is a tale told by an idiot,</string>
         <string>Full of sound and fury, signifying nothing.</string>
      </array>
   </dict>
</plist>
```

マッピングの目的は、プロパティリストファイル内の<dict>ノードの下で検出された特定のキー値ペアからCSV ファイル内に新しいラインを作成することです。具体的には、マッピングは**<key>** - **<string>** ペアのみをフィルターしなくてはなりません。他のキー値ペア（例えば、**<key>** - **<integer>**）は、無視されます。CSV ファイル内で、リストは、プロパティの値からコンマで区切られているプロパティ名を保存しなくてはなりません。すなわち、出力は以下のようになります:

```
First Name,William
Last Name,Shakespeare
Profession,Playwright
```

この目的を達成するには、マッピングは、dict ノードのすべての子ノードへの「動的」なアクセスを使用します。第2に、XML ファイルから取得されたマッピングはキー値ペアをグループ化するために group-starting-with 関数を使用します。最後に、マッピングは、フィルターを使用してノード名が「文字列」であるノードをフィルターします。

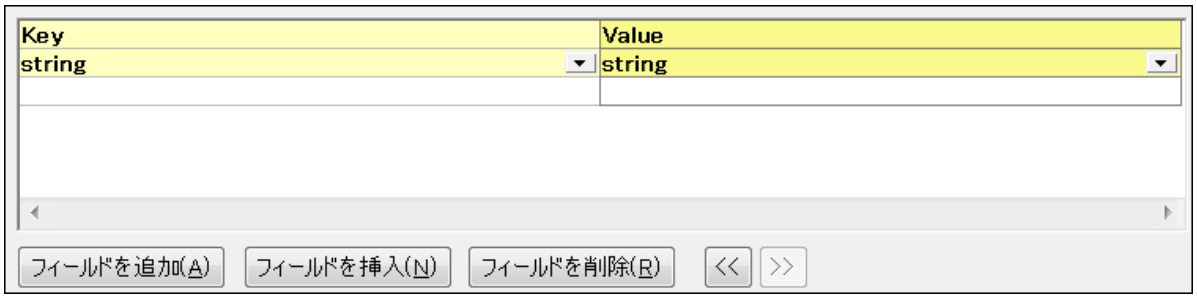
下のステップは必要とされるマッピングの作成方法を示しています。

ステップ 1: ソース XML コンポーネントをマッピングに追加する

- 「**挿入**」メニューから「**XML スキーマ/ファイル**」をクリックします。次のファイルを参照します: **<マイドキュメント>\Altova\MapForce2021\MapForceExamples\Tutorial\plist.xml**。このXML ファイルは、同じフォルダー内の **plist.dtd** スキーマを指します。

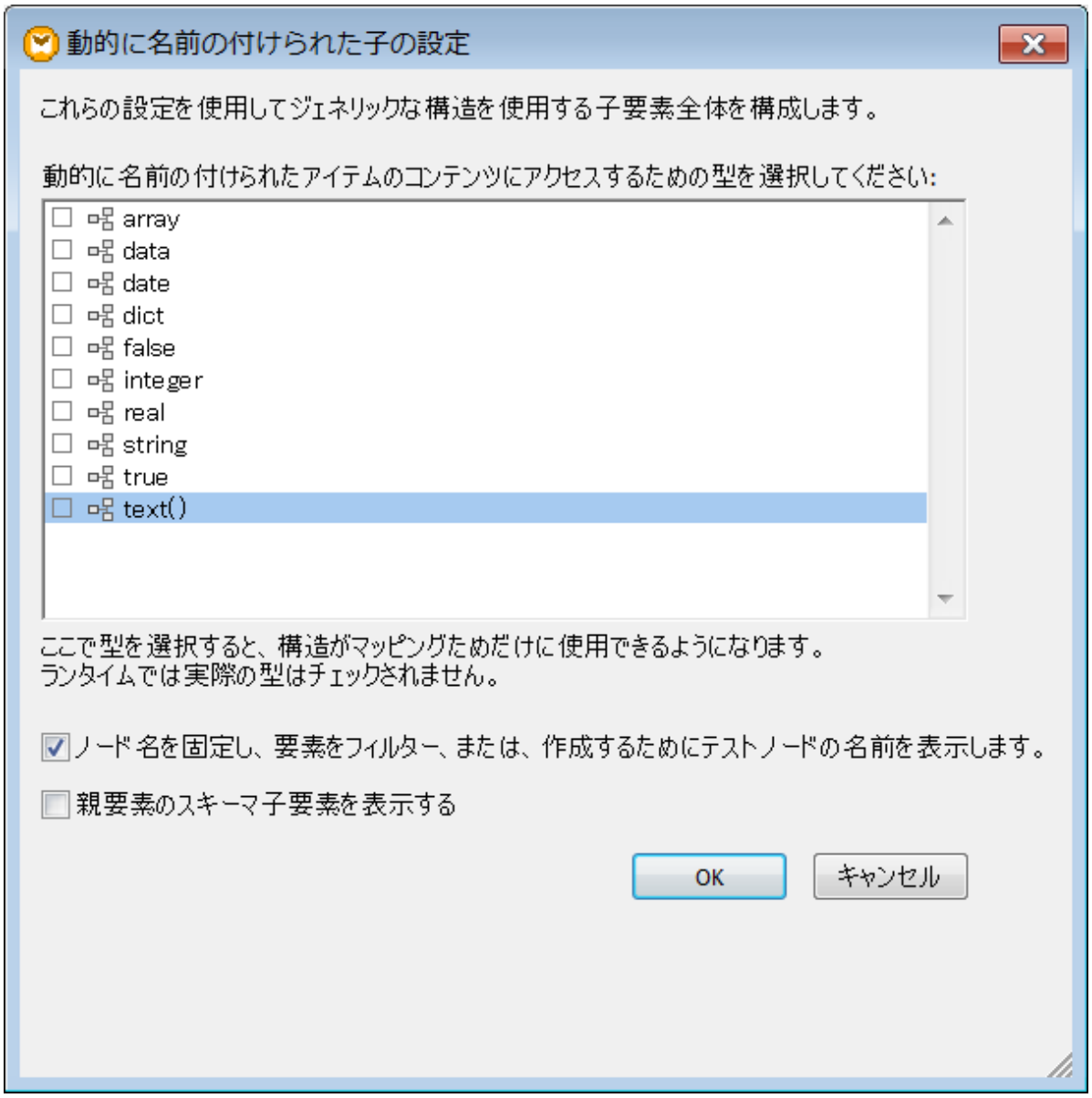
ステップ 2: マッピングにターゲット CSV コンポーネントを追加する

1. 「**挿入**」メニューから「**テキストファイル**」をクリックします。プロンプトされると、「**標準 CSV のための単純処理を使用する...**」オプションを選択します。
2. 「**フィールドの追加**」をクリックして、CSV フィールドをコンポーネントに追加します。ト
3. 各フィールド名をダブルクリックして、最初のフィールド名として「Key」を入力し、「Value」を2番目のフィールドの名前として入力します。「Key」フィールドは、プロパティ名を保管し、「Value」フィールドはプロパティの値を保管します。CSV コンポーネントの詳細に関しては、次を参照してください: CSV とテキストファイル。



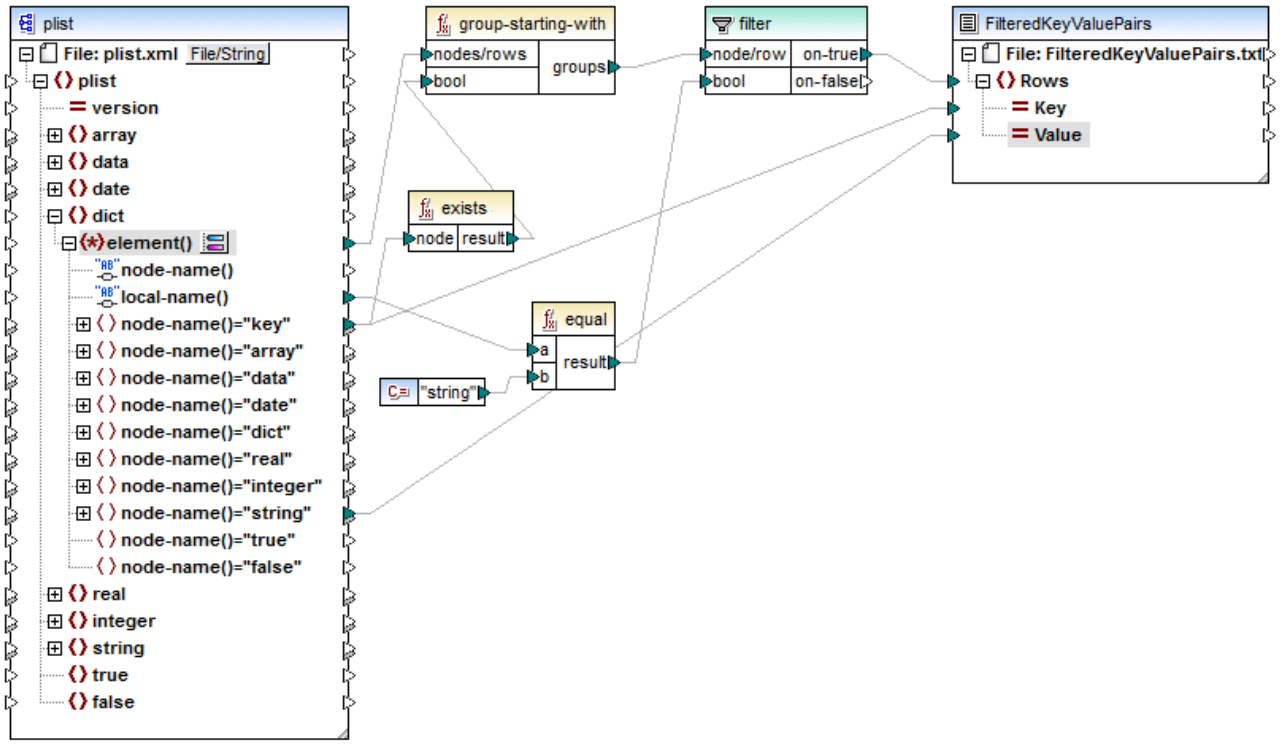
ステップ 3: フィルターと関数を追加する

1. **equal**, **exists** と**group-starting-with** 関数をライブラリウィンドウからドラッグしマッピングにドロップします。関数に関する一般的情報は次を参照してください: 関数。
2. フィルターを追加するには、「**挿入**」メニューから「Filter: Nords/Rows」をクリックします。フィルターに関する一般的情報は次を参照してください: フィルターと条件。
3. 「**挿入**」メニューから「**定数**」をクリックし、テキスト「string」を入力します。
4. ソースコンポーネント内で、`dict` ノードを右クリックし、コンテキストメニューから「**動的な名前を持つ子要素を表示する**」を選択します。「動的に名前がつけられた子の設定」ダイアログボックスでは、「**ノード名を固定し、要素をフィルター、または作成するために名前テストノードを表示する**」のチェックボックスが選択されていることを確認してください。



5. 下に表示されるように接続を描きます。

(C) 2015-2021 Altova GmbH



ReadPropertyList.mfd

マッピングの説明

ソースコンポーネント上の要素( ) アイテムは、`dict` ノードの全ての子を**`group-starting-with`** 関数へのシーケンスとして提供します。**`group-starting-with`** 関数は、名前 `key` を持つノードが発生すると新しいグループを作成します。**`exists`** 関数は、この条件をチェックして、結果をブール値 true/false としてグループ関数に返します。

各グループのために、フィルターは、**`equal`** 関数を使用して現在のノードが「文字列」と等しいかをチェックします。名前自身は、文字列としてノードの名前を与える`local-name()` から読み込まれます。

ターゲットコンポーネントへの接続には以下の役割があります:

- フィルターの条件が、true の場合のみ、新しい行がターゲット CSV 内に作成されます。
- `Key`（プロパティ名）は、ソース内の`key` 要素の値から得られます。
- `Value`（プロパティ値）は、`string` 名前 テスト ノードから得られます。

4.16 認証を使用してマッピングを実行する方法

MapForce を使用して作成されたマッピングは（例えば、W eb サービスなどの）認証を必要とする外部リソースにアクセスする必要がある場合があります MapForce（とサーバー側のMapForce Server とFlowForce Server）は以下の認証方法をサポートします:

- ユーザー名とパスワード。これらは通常基本のHTTP 認証を必要とするW eb サービスを呼び出すために必要です。
- OAuth 2.0 資格情報。これらはOAuth 2.0 検証が必要とされるW eb サービスを呼び出すために必要とされます。サーバー側では、OAuth 2.0 資格情報はMapForce Server Advanced Edition とFlowForce Server Advanced Edition によりサポートされていることに注意してください。

W eb サービスを認証し、マッピングの実行に成功するには、MapForce 内でマッピングデザインする際に以下のオプションを選択することができます:

- *資格情報* の定義。これは最も奨励され、最も柔軟的なアプローチです。資格情報オブジェクトには、フル認証の詳細、または、機密なデータなしの名前が付けられた参照が含まれている場合があります。後者の場合実際の詳細（例えば、パスワード）はマッピングのランタイムに提供される必要があります。任意で、マッピングデザインファイル（*.mfd）の一部として暗号化された書式内で機密な詳細を関連付けて保存することができます。MapForce Server を他の環境で手動の操作を使用せずにマッピングを実行する場合、資格情報をポータブルにすることができます。ポータブルな資格情報は実際のユーザー名とパスワードを含む必要がありません（含むことができる場合がありますが、選択可能です）。マッピングランタイムにこれらの詳細を設定、またはオーバーライドすることができます。詳細に関しては、資格情報を参照してください。
- *動的な認証* の使用とはすなわちユーザー名とパスワードを入力パラメーターとしてマッピングに提供することです。このアプローチでは、マッピングが実行される都度パラメーターの値をマッピングに提供する必要があります。任意で、MapForce 内でマッピングをプレビューするには、ユーザー名とパスワードを「デザインタイム」の実行値として保管することができます。しかしながら、クリアなテキストでユーザー名とパスワードは保管されます。OAuth 資格情報をマッピングに入力パラメーターとして提供することはできません。ユーザー名とパスワードのためにのみこのアプローチを使用することができます。詳細に関しては、動的な認証を参照してください。
- （レガシーアプローチ）MapForce 内でユーザー名とパスワードを名前を持つ資格情報オブジェクトとではなく、スタンドアロンのユーザー名とパスワードとして定義します。マッピングを保存する場合、ユーザー名とパスワードはデザインファイル（*.mfd）の一部として暗号化されて保存されます。このアプローチはMapForce の以前のバージョン内でデザインされたマッピングの下位互換性のためにサポートされています。詳細に関しては、保管されているユーザー名とパスワードを参照してください。

以下のセクションでは上記のシナリオのそれぞれの構成方法について説明します。

4.16.1 資格情報

（ユーザー名、パスワード、およびOAuth 認証の詳細などの）認証データを異なるマッピング実行環境で安全なポータブル化する方法を資格情報オブジェクトは提供します。資格情報は基本的な HTTP 認証 、またはOAuth 2.0 認証を必要とするマッピング内で役に立ちます。MapForce とFlowForce Server 内で資格情報を定義することができます。MapForce 内で資格情報が定義されている場合、マッピングデプロイされるようにFlowForce Server にデプロイすることができます。

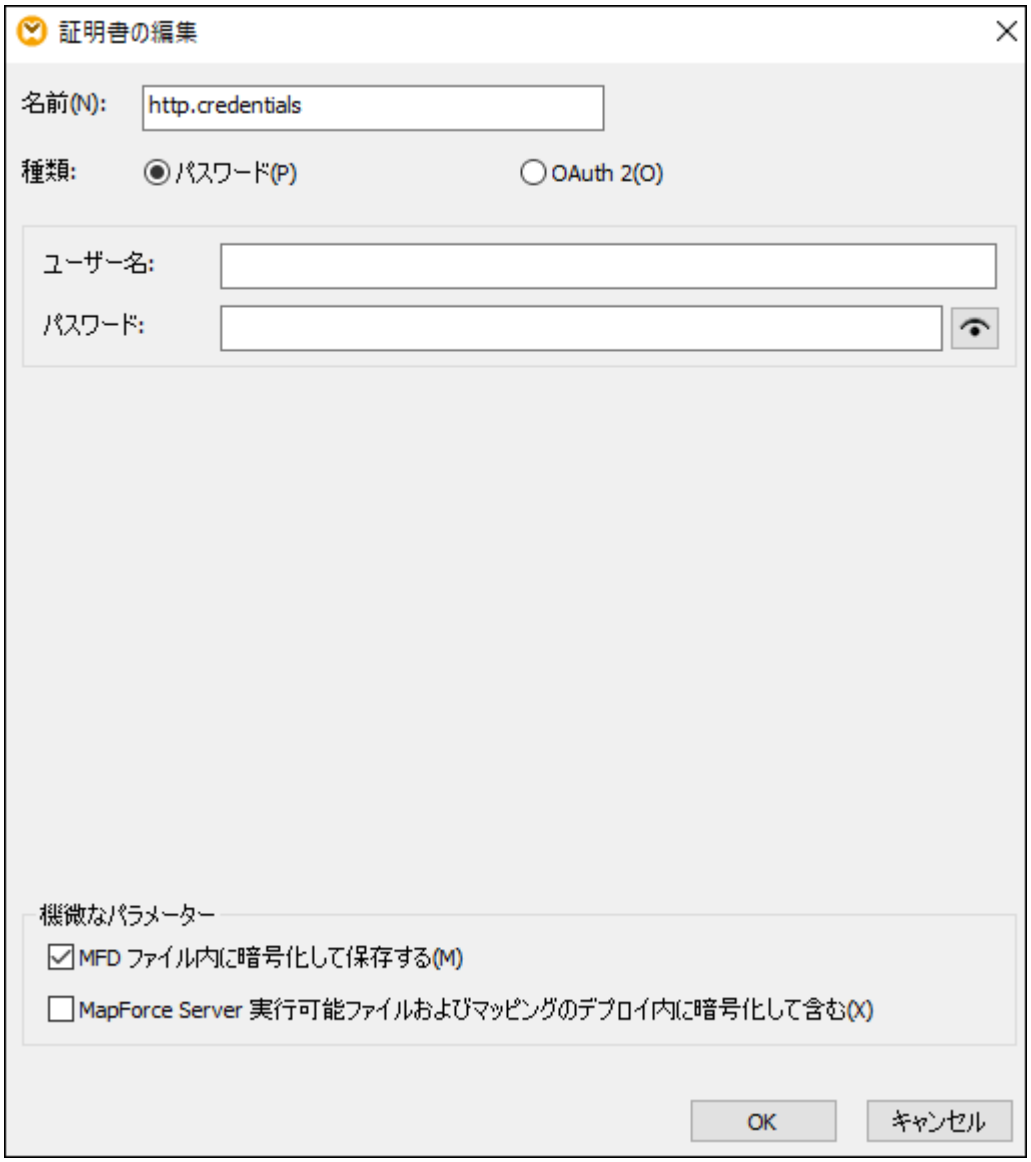
資格情報には名前と型が必要です。他のすべての詳細は任意です。名前は実行エンジンを示す名前で（MapForce または MapForce Server と名付けてください）マッピングを実行するために資格情報を必要とします。資格情報に関連する機密な情報をマッピング内で暗号化して保存するか、またはマッピングのランタイムに提供するかを決定することができます。マッピング内に機密な情報を保管する場合、サーバー環境内でマッピングの実行時に上書きすることができます。

資格情報の型は **Password** または **OAuth 2.0** であることができます。

資格情報の作成奉納:

1. マッピング上の空白を右クリックし、「**資格情報マネジャーを開く**」をコンテキストメニューから選択します。資格情報マネジャーグリッドがマッピングペインの上の部分に表示されます。

2. 「**追加**」をクリックします。資格情報情報を入力することのできるダイアログボックスが表示されます。（Web サービスコンポーネントのためにHTTP セキュリティ設定を作成、または編集することのできる同じダイアログボックスが表示されます）。
3. 資格情報名を入力します。
4. 資格情報の型を選択します（**Password** または **OAuth 2.0**）。



5. デザイン時でマッピングを認証するには、次のフィールドを記入してください：
   - 「password」資格情報の場合、Web サービスに認証するために必要なユーザー名と パスワード を入力してください。
   - **OAuth 2.0** 資格情報の詳細に関しては、OAuth 2.0 資格情報を参照してください。

マッピングデザイン内に機密なデータを任意で保存し、暗号化された書式で他の環境にデプロイすることができます。または保管しないことも可能です。機密なデータを保管すると、マッピングを閉じた後に「**表示**」ボタンをクリックすると確認することができます。

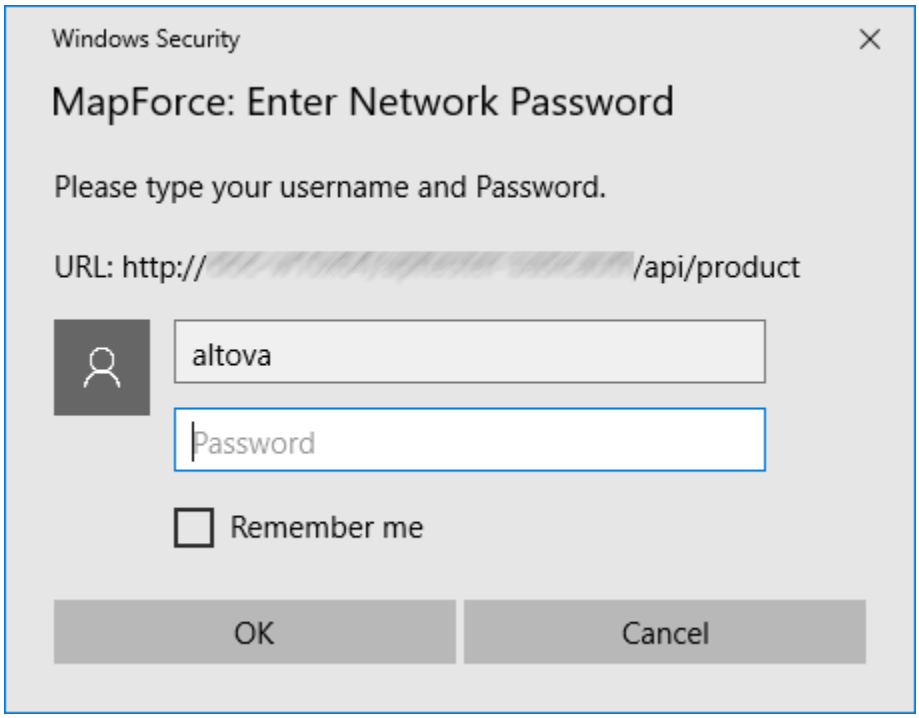
password 資格情報の場合、**Password** が機密なフィールドです。OAuth 2 資格情報の場合、**クライアントの秘密**、**アクセス**

トークン と **更新トークン** が機密なフィールドです。グラフィカルなインターフェイスでは後者の2つは表示されないことご注意してください。

次のチェックボックスにより機密なデータの保管に関する希望する振る舞いを管理することができます。

MFD ファイル内で暗号化して保存する	このチェックボックスを選択して、暗号化された書式でマッピングデザインファイル(.mfd) 内に機密なデータを保存します。
MapForce Server 実行ファイルとマッピングデプロイを含む	MapForce Server 実行可能ファイルにマッピングをコンパイルするで説明されているとおり、コンパイル時にMapForce Server 実行可能ファイル(.mfx) に含む場合、このチェックボックスを選択します。 チェックボックスが選択されていると、機密なデータはFlowForce Server にデプロイされるパッケージに含まれます。FlowForce Server にマッピングをデプロイする を参照してください。

マッピング実行時に暗号化を解除する必要があるため、機密なフィールドは元に戻すことが可能な暗号化された書式で保管されます。マッピング実行時に動的に機密なデータを提供でき、機密なデータを保管しない場合、上記のチェックボックスのチェックを解除してください。マッピングを実行しようとするとMapForce はパスワードをプロンプトする可能性があります。例:

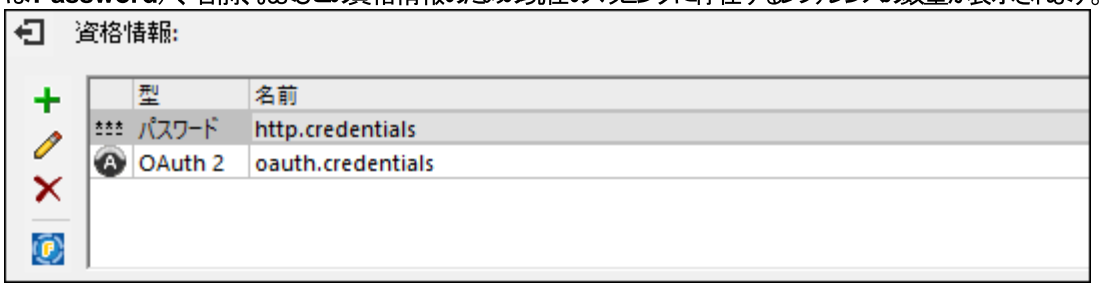


サーバ側では、マッピングの実行時にパスワードを動的に提供することができます。詳細に関しては、MapForce Server 内の資格情報 と FlowForce Serverr 内の資格情報 を参照してください。

(C) 2015-2021 Altova GmbH

4.16.1.1 資格情報マネジャー

資格情報マネジャーはマッピング内の資格情報を管理することのできるグラフィカルなユーザーインターフェイスです。マッピングに資格情報が含まれる場合、これらはマッピングウィンドウの上のグリッド内に表示されます。各資格情報オブジェクトではグリッドは資格情報型（**OAuth** または **Password**）、名前、およびこの資格情報のための現在のマッピングに存在するレファレンスの数量が表示されます。



資格情報マネジャー

資格情報マネジャー を隠すには、「閉じる」または「Esc」キーをクリックします。

資格情報マネジャーを表示するには、マッピングを右クリックして、「**資格情報マネジャーを開く**」をコンテキストメニューから選択します。

資格情報マネジャー から次のアクションを実行することができます:

アイコン	詳細
+	新規の資格情報オブジェクトを作成します。資格情報を参照してください。
鉛筆	既存の資格情報オブジェクトを編集します。
FF	資格情報オブジェクトを FlowForce Server にデプロイします。FlowForce Server に資格情報をデプロイするを参照してください。
×	資格情報オブジェクトをマッピングから削除します。

資格情報レコードを右クリックし、対応するコマンドをコンテキストメニューから選択して切り取り、コピーし、張り付けることができます。または、標準的な Windows ショートカット（「**切り取り**」、「**コピー**」および「**張り付け**」のためにそれぞれ **Ctrl+X**, **Ctrl+C**、と **Ctrl+V** を使用します）を使用します。マッピング上のコンポーネントにより参照される資格情報を検索するには、資格情報マネージャーを右クリックし、コンテキストメニューから「**全ての参照するコンポーネントを検索する**」を選択します。出力はマネージャーウィンドウ内に表示されます。

4.16.1.2 UDF 内の資格情報

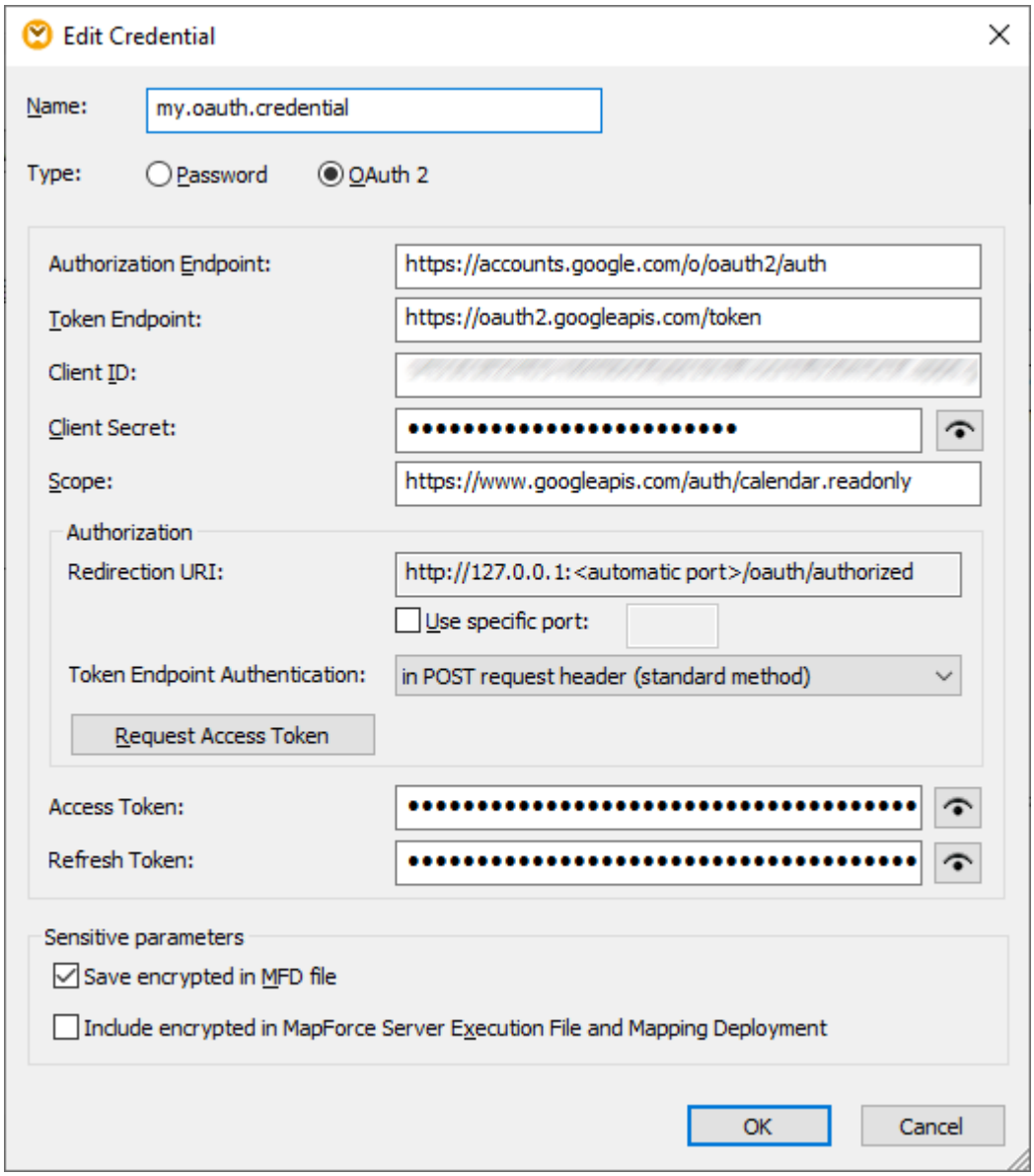
ユーザー定義関数（UDF）を含むマッピングは他のマッピングにインポートすることができます。ユーザー定義関数の呼び出しとインポートを参照してください。インポートされたマッピングに資格情報が含まれる場合、資格情報マネージャー内に資格情報はインポート済みと表示されます。インポートされた資格情報は黄色い背景色で表示されます。

デフォルトでは、インポートされた資格情報はメインマッピングと共に保存されませんが、ローカルコピーをオプションで作成し、メインマッピング内に保存することもできます。メインマッピング内からインポートされたマッピングを編集しようとすると警告ダイアログボックスが表示されます。

メモ　インポートされたマッピングとメインマッピングが同じ資格情報を含む場合、名前の競合が発生する場合があります。名前の競合を回避するために、資格情報名がマッピングとインポートされた UDF のスコープ内で一意であることを確認してください。

4.16.1.3 OAuth 2.0 資格情報

MapForce 内のOAuth 2.0 承認を必要とするマッピングを実行するには資格情報マネジャーを使用して型 **OAuth 2.0** の資格情報を作成する必要があります。OAuth 2.0 詳細を記入し、以下で説明される通り手動の認証プロセスに従ってください。



OAuth 2.0 資格情報オブジェクトに関するフィールドは下にリストされています。これらの値を取得するには（例えば、Google API Console、Facebook API、Bitbucket API などの） W eb サービスプロバイダーに登録する必要があります

認証エンドポイント	MapForce が自動化フローを開始するURI を指定します。W eb サービスプロバイダーを使用して登録後にこの値を取得することができます。
トークンエンドポイント	MapForce がトークンフローを開始するURI を指定します。W eb サービスプロバイダーを使

(C) 2015-2021 Altova GmbH

	用して登録後にこの値を取得することができます。
クライアント ID	クライアントアプリケーションのID です（この場合、MapForce ）。W eb サービスプロバイダーを使用して登録後にこの値を取得することができます。
クライアントの秘密	クライアントアプリケーションに関連する秘密です。W eb サービスプロバイダーを使用して登録後にこの値を取得することができます。
スコープ	プロバイダーにより必要とされる場合のクライアントアプリケーションのスコープです。W eb サービスプロバイダーを使用して登録後にこの値を取得することができます。
リダイレクト URI	OAuth 2.0 認証サーバーはMapForce. へのレスポンスを送信するコールバックURL を必要とします。MapForce は認証サーバーとの通信のために自動的にURL とポートを割り当てます。 認証サーバーが通信が特定のポートを使用して行われることを指定する場合、「**特定のポート**」を選択し、希望するポート名を入力します。
トークンエンドポイント認証	OAuth 2.0 認証サーバーの多くは**POST リクエストヘッダー**に認証の詳細が送信されることを必要とします。これもデフォルトでドロップダウンリストから選択された値です。 OAuth 2.0 認証サーバーの一部はPOST リクエストのボディ内のみで認証の詳細を受け入れる場合があります。このような認証サーバーのために **POST リクエストボディ内** でドロップダウンリストから値を選択してください。
アクセストークン	認証サーバーにより返されるアクセストークンです。リソースサーバーがアクセストークンが正確で有効であると判断し場合のみマッピングの実行は成功します。
更新トークン	認証サーバーにより返される更新トークンです。「**アクセストークン**」の期限が切れると必要になります（下を参照）。稀なケースとしてアクセストークンの期限が切れない場合、これは必要ありません。

上記のフィールドを記入した後、「**アクセストークンのリクエスト**」をクリックして、マッピングを実行するために必要なアクセストークンを取得します。W eb サーバープロバイダーにより実際の認証プロセスは異なります。識別を決定するためにW eb サービスにアクセスを許可するブラウザーウィンドウ内で手動で確認することを通常必要とします（例えば、マッピングがGoogle API を呼び出す場合はGoogle アカウントを指します）。プロセスの終わりに、MapForce はサーバーからアクセストークンが取得されたことを確認するメッセージを表示します。資格情報 で説明されているとおり「**MFD ファイル内で暗号化して保存する**」チェックボックスが選択されている場合のみアクセストークンが保存されることに注意してください。「**MFD ファイル内で暗号化して保存する**」チェックボックスが選択されていない場合、マッピングを実行する都度、手動で認証する必要があります。

アクセストークンはある一定の期間が過ぎると期限が切れる可能性があります。トークンの有効期限はW eb サービスにより異なります。更に、アクセストークンはW eb サーバープロバイダーにより明示的に呼び出される場合があります。このような場合、メッセージウィンドウはマッピングをプレビューしようとすると認証エラーを表示します。この場合、「**アクセストークンのリクエスト**」をクリックして、認証プロセスをもう一度行うとマッピングを実行することができます。

4.16.1.4 MapForce Server 内の資格情報

MapForce Server 実行可能ファイル（.mfx）にマッピングをコンパイルした後、資格情報の作成 で説明されるとおりMapForce Server をマッピングデザイン時に従い.mfx ファイルを実行します。

MapForce 内で資格情報を作成する場合、「**実行可能ファイルとマッピングのデプロイに含まれる**」チェックボックスが選択されている場合、MapForce Server はマッピングランタイムで.mfx ファイル内に保管されている資格情報を使用します。これはマッピングを以下のようなコマンドで実行可能なことを意味します:

```
<exec> run mapping.mfx
```

<exec> がMapForce Server 実行可能ファイルへのパスである場所です。このパスは絶対的、または、現在のディレクトリが実行可能と同じ場合、実行可能ファイル名のみを入力することができます。

(ユーザー名とパスワード無しで) 資格情報名のみをMapForce 内に入力した場合、`run` コマンドのために使用できる **--credential** コマンドラインオプションを使用してマッピングランタイムにこれらの情報を明示的に提供する必要があります。このようにして、例えば、マッピングのデザインとは異なる生産の資格情報の異なるセットを使用することができます。`--credential` オプションは書式 `--credential=KEY:VALUE` を以下の場所で使用します

- **KEY** はMapForce 内で定義されている通りの資格情報の名前です。
- **VALUE** は資格情報プロパティ、または、アンパサンド (&) により区切られているプロパティのリストです。型「パスワード」の資格情報のために使用することが可能なプロパティは **username** と **password** です。型 OAuth 2.0 の資格情報のためにサポートされるプロパティは **oauth:token** のみです。
- "=" サインを使用して実際のプロパティ値はURL 内のクエリパラメーターと同様に提供されます。

例:

```
<exec> run mapping.mfx --credential="mycredential:username=admin&password=4xJ38dnx7"
```

上記のコードリスティングでは、ユーザー名とパスワードがアンパサンド文字により区切られているため値をそのまま使用するために `--credential` オプションの値は引用符により囲まれています。

実行するためにマッピングが名前を持つ資格情報の複数のセットを必要とする場合、**--credential** オプションを複数回指定することができます。

> コマンドラインオプションとして提供された資格情報は保管された資格情報よりも優先されます。

「実行可能ファイルとマッピングのデプロイに含まれる」 チェックボックスを選択しなかった場合、重要なフィールドが不足することになります。これは、資格情報を名前で参照しつつ、コマンドラインでパスワードを提供する必要があることを意味します。例:

```
<exec> run mapping.mfx --credential=mycredential:password=4xJ38dnx7
```

> 以下のフィールドは重要なデータと考慮されます:
>
> - (型「Password」の証明書のための) **パスワード**
> - (型「OAuth 2.0」のための資格情報) **クライアントの秘密、アクセストークン、および更新トークン**

OAuth 2.0 認証が必要なマッピングのために、MapForce Server コマンドラインはOAuth 2.0 アクセストークンをマッピングのランタイムで入力として受け入れます。MapForce Server コマンドラインはインタラクティブな GUI を提供しないため、具体的にコマンドラインを使用する際に外部の方法を使用して OAuth 2.0 アクセストークンを取得する必要があります。これは後者がランタイムに新規のOAuth 2.0 アクセストークンを自身で取得することができるため、(例えば、MapForce を使用してリクエストするなど) FlowForce Server の管理下でMapForce Server が作動する場合必要ありません。

コマンドラインでは、保管されたOAuth 2.0 トークンの期限が切れていない場合、または、W eb サービスプロバイダーにより呼び出されていない場合、保管された資格情報を使用してマッピングを実行することが可能です。これに対処するには、**--credential** オプションを使用して(外部の方法で取得された) 新規のOAuth 2.0 アクセストークンを提供してください。例:

(C) 2015-2021 Altova GmbH

```
<exec> run mapping.mfx --
credential=my_oauth_credential:oauth:token=jdsaflkajlkewsaiurthczv904215-jhd
```

場所:

- **my_oauth_credential** はMapForce から作成されたOAuth 2.0 資格情報の名前です。
- **oauth:token** は新規のOAuth 2.0 アクセストークンがランタイムに提供されてるMapForce Server を示す方法です。

MapForce Server API

MapForce Server API は資格情報を作成し、資格情報にプロパティを追加し、宣言後に資格情報を閉じるメソッドを提供します。詳細に関してはMapForce Server ドキュメントを参照してください(https://www.altova.com/documentation)

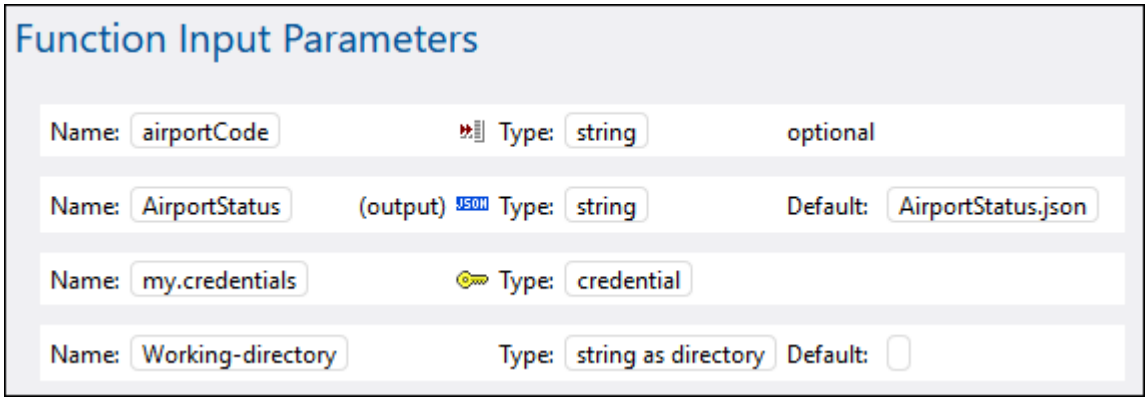
4.16.1.5 FlowForce Server 内の資格情報

MapForce からFlowForce Server に資格情報を含むマッピングをデプロイする場合、サーバーに資格情報もデプロイされます。デプロイされた情報には資格情報レコードを作成した際に記入されたフィールドのみが含まれます。例えば資格情報名のみを保管することを選択し) 空の資格情報である場合、または、ユーザー名とパスワードの両方を含む資格情報オブジェクトの場合。

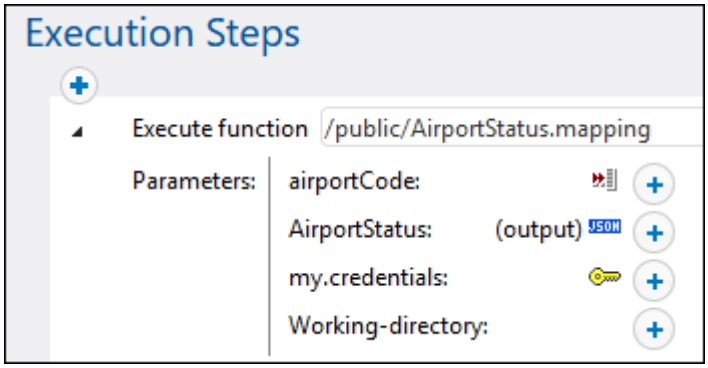
MapForce からFlowForce Server へ資格情報オブジェクトをスタンドアロンオブジェクトとしてデプロイすることができます。FlowForce Server に資格情報をデプロイする を参照してください。

資格情報の作成 で説明されている通り「**実行可能ファイルとマッピングのデプロイに含まれる**」チェックボックスを選択した場合のみ重要なデータがデプロイされます。これはマッピングをデプロイする際、およびスタンドアロン資格情報をデプロイする際に適用されます。

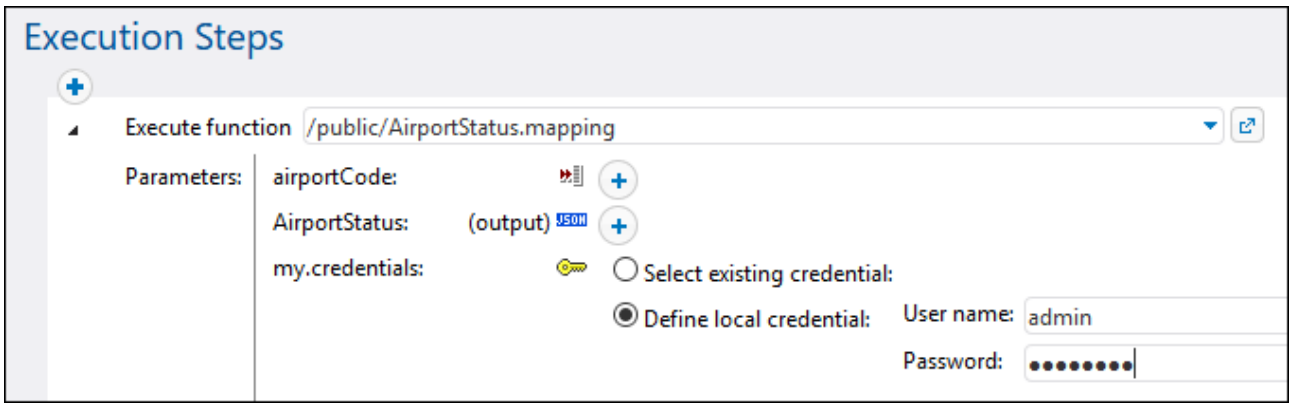
FlowForce Server 内では、対応するマッピング関数のページを開くことにより、マッピング関数が資格情報を必要とするか確認します。例:



「**実行可能ファイルとマッピングのデプロイに含まれる**」チェックボックスを資格情報を作成する際に選択した場合、マッピングと共に資格情報をジョブはデプロイします。この場合、ジョブ構成ページから指定する必要はありません。例えば、次の実行ステップは存在する場合保管された資格情報と共にマッピング関数を実行します(“my.credentials” パラメーターが展開されていないことに注目してください):



FlowForce Server 内で直接定義された、または、ローカルの資格情報により定義された他の資格情報オブジェクトと保管された資格情報を上書きすることができます。これを行うには、「+」ボタンを押して、FlowForce Server 内に既存の資格情報オブジェクトを選択、または、直接ユーザー名とパスワードを入力します。例:



実行ステップにパラメーターとして提供された資格情報は、マッピング関数内に保管されている資格情報よりも優先順位が高いです。

「**実行可能ファイルとマッピングのデプロイに含まれる**」チェックボックスを MapForce 内で資格情報を作成する際に選択しなかった場合、実行ステップにパラメーターとして資格情報を提供することは必須です。それ以外の場合、実行は失敗します。

4.16.1.6 FlowForce Server に資格情報をデプロイする

資格情報は頻繁に変更する可能性があります。これは資格情報が変更される都度、マッピングを変更して再デプロイする必要があることを意味しません。代わりに、マッピングに変更を加えることなく資格情報オブジェクトを便利に FlowForce Server にデプロイすることができます。これにより複数の FlowForce Server ジョブ内で資格情報オブジェクトを再利用することができます。

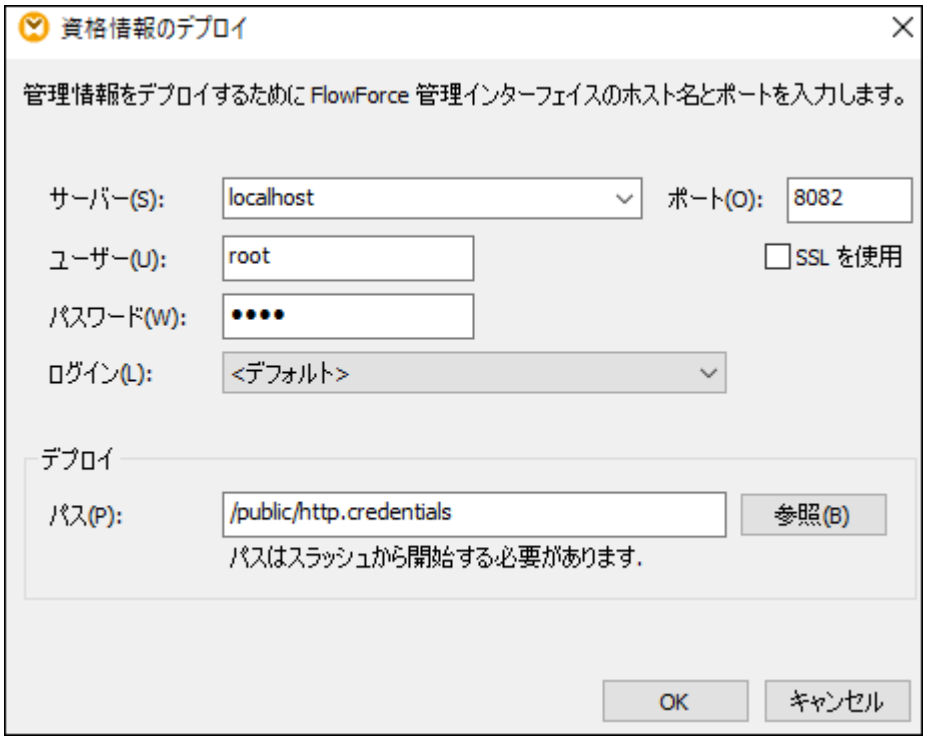
または、FlowForce Server 内で直接資格情報オブジェクトを作成し、他の FlowForce 資格情報と同様に実行ステップにパラメーターとして提供することもできます。

MapForce 資格情報 FlowForce Server にデプロイする方法:

1. 資格情報を含むマッピングを開きます。
2. マッピングの空のエリアを右クリックし、「**資格情報マネジャーを開く**」をコンテキストメニューから選択します。

(C) 2015-2021 Altova GmbH

3. 資格情報マネジャーグリッドからレコードを選択し、「FlowForce Server に資格情報をデプロイする」ボタンをクリックします。ダイアログボックスFlowForce Server 接続詳細を入力しサーバー上のターゲットパスを指定することのできるダイアログボックスが開かれます:



4. 「OK」をクリックし、メッセージウインドウに表示される結果を確認します。

デプロイオプションは以下にリストされています。

設定	詳細
サーバー、ポート、SSL の使用	サーバーホスト名（またはIP アドレス）およびFlowForce Server のポートを入力します。FlowForce Server がデフォルトのポートで同じマシン上で作動する場合、これらは **localhost** と **8082** である可能性があります。確かではない場合、FlowForce Server Web 管理インターフェイスにログオンして、I.P. アドレスとポートをWeb ブラウザーのアドレスバーで確認してください。 接続エラーが発生した場合、目的のアドレスとポート上で受信する接続を受け入れるようにFlowForce Server が作動するマシンが構成されていることを確認してください。 SSL-暗号化済みの接続を介して、資格情報をデプロイするには、「**SSL の使用**」チェックボックスをチェックしてください。FlowForce Server は既に接続を受け入れるように構成されていると仮定します。詳細に関しては、FlowForce Server ドキュメントを参照してください (https://www.altova.com/ja/documentation)。
ユーザーとパスワード	入力されるユーザー名とパスワードはログインドロップダウンリストにより異なります（次のオプションを参照してください）。ログインドロップダウンリストが **<Default>** または **Directly** に設定されている場合、FlowForce Server ユーザー名とパスワードを入力します。それ以外の場合、ドメインのユーザー名とパスワードを入力し、ログインドロップダウンリストからドメイン名を選択します。

設定	詳細
ログイン	FlowForce Server 内でディレクトリサービス統合が有効化されている場合、子のドロップダウンリストからドメイン名を選択し、ユーザーとパスワードフィールドにドメイン資格情報を入力します（前のオプションを確認してください）。
パス	「参照」をクリックし、FlowForce Server コンテナーハイアラーキー内で資格情報が保存されるパスを選択します。デフォルトでは、パスはFlowForce Server の**/public** コンテナーに設定されています。 FlowForce Server パーミッションと特権を有することが前提で、新規のコンテナーを作成、または、既存のコンテナーとマッピングこのダイアログボックスから直接を削除します。

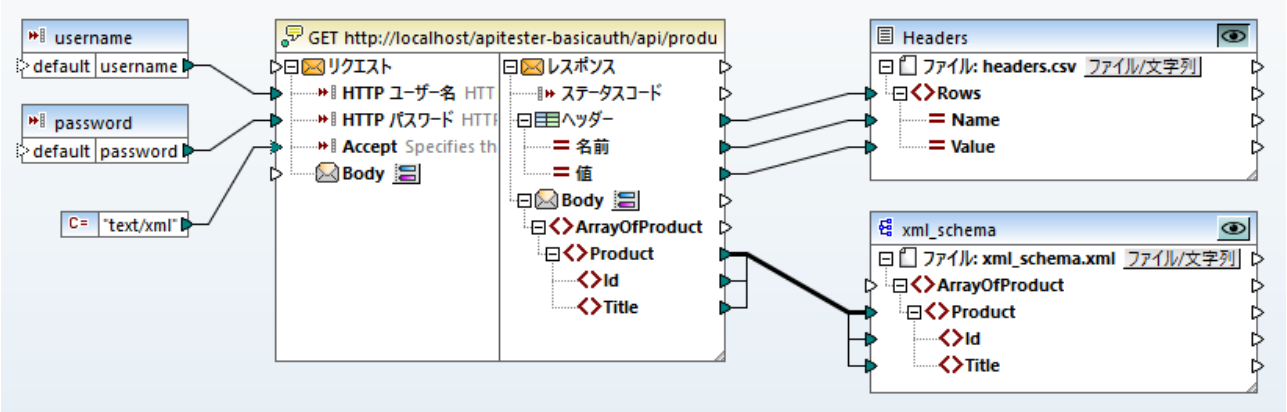
4.16.2 動的な認証

動的な認証とは、ランタイムでマッピングにユーザー名とパスワードを入力パラメーターとして提供することのできる機能を指します。動的な認証はHTTP 認証を必要とするW eb サービスを呼び出すマッピングのためにサポートされています。

動的な認証のセットアップ方法:

1. W eb サービスの呼び出しを含むマッピングを開く、またはW eb サービスが新規のマッピングに追加します。詳細に関しては、W eb サービス呼び出しの追加（REST-Style）とW eb サービス呼び出しの追加（SOAP）を参照してください。
2. 設定ダイアログボックスを開くために、W eb サービスコンポーネントのタイトルバーをダブルクリックします。
3. 「HTTP セキュリティ設定」の横の「編集」をクリックします。
4. 「動的な認証」を選択します。
5. 「OK」をクリックします。

動的な認証が有効化されると、W eb サービスコンポーネントの「リクエスト」側に以下の2つの新規の入力接続が追加されます: HTTP ユーザー名 とHTTP パスワード。実際の値を提供するには、「挿入 | 入力を挿入」メニューコマンドを使用し、それぞれ「HTTP ユーザー名 and HTTP パスワード」に追加しマッピングに2つの新規の入力コンポーネントを追加します。



MapForce 内でマッピングをプレビューするには、ユーザー名とパスワードを「デザイン-実行」値を任意で提供することができます。これを行う場合、各入力コンポーネントのタイトルバーをダブルクリックし、「値」テキストボックスにそれぞれユーザー名とパスワードを入力します。詳細に関しては、マッピングにパラメーターを与えるを参照してください。

(C) 2015-2021 Altova GmbH

ユーザー名とパスワードはクリアなテキストとしてマッピングデザイン内に保存できるように他のデザイン時の実行の値と同様に扱われます。これを回避するには、デザイン時実行の値を入力しない、または資格情報アプローチを使用してください。

4.16.2.1 MapForce Server 内の動的な認証

動的な認証のためにマッピングを構成する場合、対応するユーザー名とパスワードはマッピングのランタイムにパラメーターとして提供される必要があります。これはマッピングへ他のパラメーターの型を提供することと違いはありません。例えば、MapForce Server をコマンドラインで呼び出す場合、マッピングのための構文は下記のようになります:

```
<exec> run mapping.mfx --p=username:admin --p=password:dj9JaVax
```

場所:

- **<exec>** はMapForce Server 実行可能ファイルへのパスです。このパスは絶対的、または、現在のディレクトリが実行可能と同じ場合、実行可能ファイル名のみを入力することができます。
- **username** と**password** はMapForce マッピング上の対応する入力パラメーターの名前です。

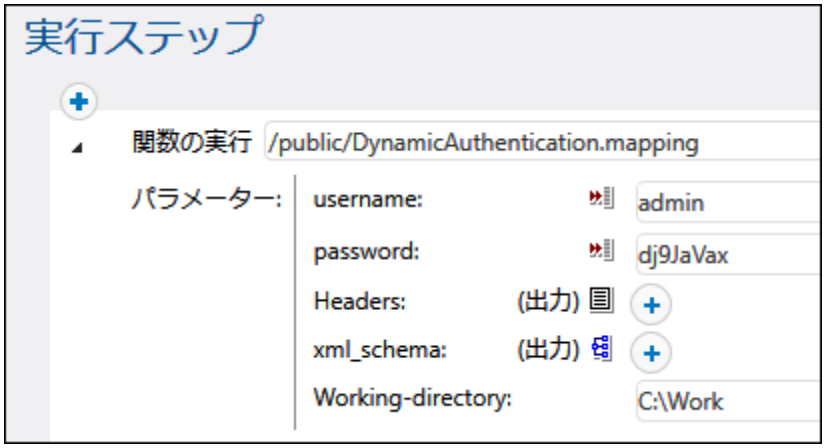
MapForce Server API を呼び出す際、**Run** メソッドを**AddParameter** メソッドを呼び出す前に呼び出して、マッピングを認証することができます。例えば、C# では、アーカイブするコードは以下のようになります:

```
try
{
    Altova.MapForceServer.Server mfs = new Altova.MapForceServer.Server();
    mfs.AddParameter("username", "amdin");
    mfs.AddParameter("password", "dj9JaVax");
    mfs.WorkingDirectory = "C:\\Work";
    if(mfs.Run("C:\\Work\\mapping.mfx"))
    {
      Console.WriteLine("Success");
    }
    else
    {
      Console.WriteLine(mfs.LastExecutionMessage);
    }
    Console.ReadLine();
}
catch(Exception ex)
{
  Console.WriteLine(ex);
}
```

username と**password** (**AddParameter** メソッドの最初の引数) はMapForce マッピング上の対応する入力パラメーターの名前と同じである必要があります。

4.16.2.2 FlowForce Server 内の動的な認証

FlowForce Server に動的な認証を含むマッピングをデプロイする場合、ユーザー名とパスワードはマッピング関数の入力パラメーターになります。このようなマッピング関数を呼び出す FlowForce Server ジョブは実行に成功するためにユーザー名とパスワードが必要です。例:



上記のサンプルでは、ユーザー名とパスワードはそれぞれのテキストボックス内に入力されています。しかしながら、ジョブに入力パラメーターとして提供することもできます。。詳細に関しては、FlowForce Server ドキュメントを参照してください (https://www.altova.com/ja/documentation)。

4.16.3 保管されたユーザー名とパスワード

ユーザー名とパスワードをマッピング内で暗号化して保管することは認証を使用してマッピングを実行する方法です。機敏なデータがマッピングの一部として暗号化されて保管されるため、または、サーバー実行中にサーバー実行ファイルの一部として暗号化されるため、このアプローチにより実行環境に関わらずマッピングを簡単に実行することができます。しかしながら、削減された複雑性を考慮しても、ユーザー名とパスワードが変更される都度、更新し、新規のサーバー実行 .mfx ファイルに再コンパイルする、またはFlowForce Server に再デプロイするという必要があるという欠点があります。また、資格情報と同様にマッピングランタイムにユーザー名とパスワードを変更することができます。

> このアプローチはMapForce の前のバージョンの下位互換性によりのみサポートされています。資格情報 または 動的な認証 を代わりに使用することが奨励されます。

保管されたユーザー名とパスワードと共に認証をセットアップする方法:

1. W eb サービスの呼び出しを含むマッピングを開く、または、新規のマッピングにW eb サービスを追加します。
2. 設定ダイアログボックスを開くために、W eb サービスコンポーネントのタイトルバーをダブルクリックします。
3. 「HTTP セキュリティ設定」の横の「編集」をクリックします。
4. 「ユーザー名」の横のラジオボタンを選択し、保管するユーザー名とパスワードを入力します。プロンプトされるとパスワードを確認してください。
5. 「OK」をクリックします。

他の必要条件無しにMapForce 内でマッピングをプレビューすることができます。MapForce Server 内では、資格情報を必要としないマッピングと同様マッピングを実行することができます。例えば、コマンドラインでMapForce Server を呼び出す場合、マッピングを実行する構文は以下のようになります:

```
<exec> run mapping.mfx
```

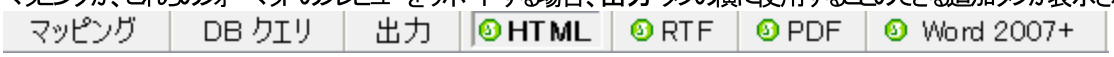
マッピング関数として作動するためにFlowForce Server にマッピングにデプロイする場合、FlowForce Server 内ではマッピングに保管された資格情報が含まれることを示す特別な通知はありません。マッピング関数からジョブを作成し、特別な条件無しに実行することができます。

(C) 2015-2021 Altova GmbH

4.17 StyleVision を使用してマッピングの出力のスタイルを指定する方法

Altova StyleVision が使用中のコンピューターにインストールされていると仮定して、マッピングのターゲットコンポーネントがXML または XBRL である箇所は、マッピングの出力をHTML、RTF、PDF とW ord 2007+ ドキュメントとしてプレビューし、保存することができます。StyleVision のEnterprise エディションが使用されている場合、プレビュー内でチャートを表示することができます。

マッピングが、これらのフォーマットのプレビューをサポートする場合、出力 タブの横に使用することのできる追加タブが表示されます。例えば



StyleVision プレビュータブ (MapForce Enterprise Edition)

以下に注意してください:

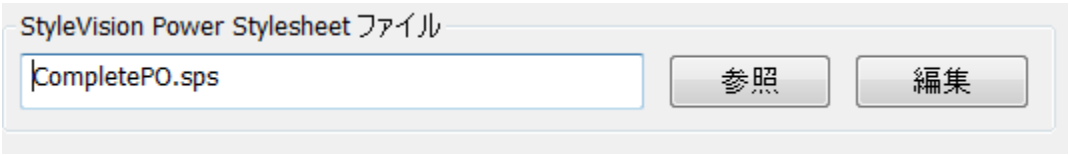
- StyleVision Professional がインストールされていると、**HTML** と**RTF** 出力をプレビューすることができます。StyleVision Enterprise を使用すると、**HTML**、**RTF**、**PDF** と**Word 2007+** 出力をプレビューすることができます。
- マッピングの出力をPDF としてプレビューするには、Java、Acrobat Reader、とFOP (Formatting Objects Processor) バージョン0.93 または1.0. FOP が、StyleVision をインストールしない オプションが選択されていない限り、StyleVision と共にインストールされている必要があります。
- MapForce の64-ビットエディションの場合、W ord 2007+ とRTF プレビューは、埋め込まれていない アプリケーションとして開かれます。
- マッピングがソースとターゲット (パススルーコンポーネント) としての役割を果たすコンポーネントを含む場合、コンポーネントの「プレビュー」ボタン が有効化されている場合のみ、StyleVision でのプレビューが可能になります。このようなマッピングの詳細に関しては、次を参照してください: チェーン マッピング.

マッピングからデータをこのようにプレビューするには、以下の必要条件が存在します:

- Altova StyleVision がコンピューターに、スタンドアロン、またはAltova MissionKit の一部として、インストールされている必要があります。
- ターゲットコンポーネントには、関連付けられたStyleVision Power Stylesheet (SPS) ファイルが必要です。StyleVision を使用して、スタイルシートを作成、または編集することができます。MapForce で直接スタイルシートを編集、または変更することはできませんが、StyleVision 内のMapForce から開くことはできます。スタイルシートの準備ができると、下に示されるようにターゲット MapForce コンポーネントに割り当てることができます。

ターゲットコンポーネントにStyleVision パワースタイルシートを割り当てる

1. StyleVision 内で、必要とするスタイルシートファイルを作成します。MapForce コンポーネントと同じ XML スキーマがソースとして使用されていることを確認してください。
2. MapForce 内で、ターゲット XML またはXBRL コンポーネントを右クリックし、「プロパティ」を選択します。
3. コンポーネント設定ダイアログボックス内で、「StyleVision Power スタイルシートファイル」の横から、作成済みのスタイルシートファイルを選択します。



メモ StyleVision Power スタイルシートファイルへのパスは、絶対または相対的であることができます。次を参照してください: 相対的および絶対パスの使用。

(C) 2015-2021 Altova GmbH

StyleVision により生成された出力を保存する

StyleVision により生成されたHTML、PDF、RTF、またはW ord 2007+出力 を、他のマッピングの保存と同様に保存することができます。具体的には、マッピングのプレビューを行うには、以下の内の1つをおこなってください:

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

Altova 製品のパッケージを使用して、HTML、PDF、RTF、W ord 2007+ ファイルを自動生成する

マッピングが同じ、または異なるマシン、またはプラットフォームで HTML、PDF、RTF、およびW ord 2007+ ファイルを自動的に生成する場合、(MapForce とStyleVision の機能性を拡張する個別にライセンスされるサーバー製品である) MapForce Server と StyleVision Server を使用して行うことができます。このシナリオの場合、各アプリケーションが以下の特別な役割を果たします:

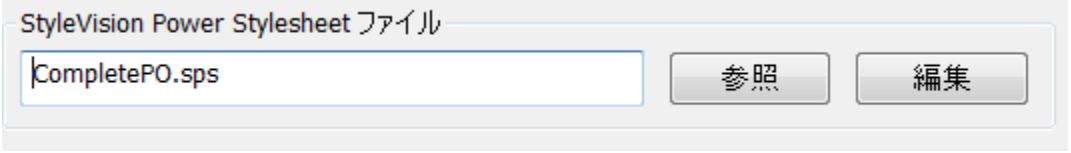
- MapForce – (例えば、データベースからXML への)データ変換の入力と出力 を定義するマッピング(.mfd ファイル) のデザインを行うことができます。
- MapForceServer – マッピング(.mfx ファイル) をコマンドライン またはAPI から、(同じ、または異なるオペレーティングシステム上で)実行可能なマッピングを実行します
- StyleVision – マッピングの出力をHTML、PDF、RTF、W ord 2007+に出力するために必要な、スタイルシート (.sps ファイル) をデザインすることができます。
- StyleVision Server – マッピングの出力をターゲットフォーマットに変換する.sps スタイルシートを作成します。これは、コマンドライン、またはAPI (同じ、または異なるオペレーティングシステム) から行うことができます。
- (個別のライセンスを必要とする)StyleVision Server とMapForce Server は、任意でFlowForce Server の管理下で作動します。このシナリオの場合、MapForce マッピング とStyleVision 変換は、予定通り、実行、トリガー、または需要に応じて実行することができ、完全に自動化することができます。

4.17.1 StyleVision スタイルシートを使用したマッピングの例

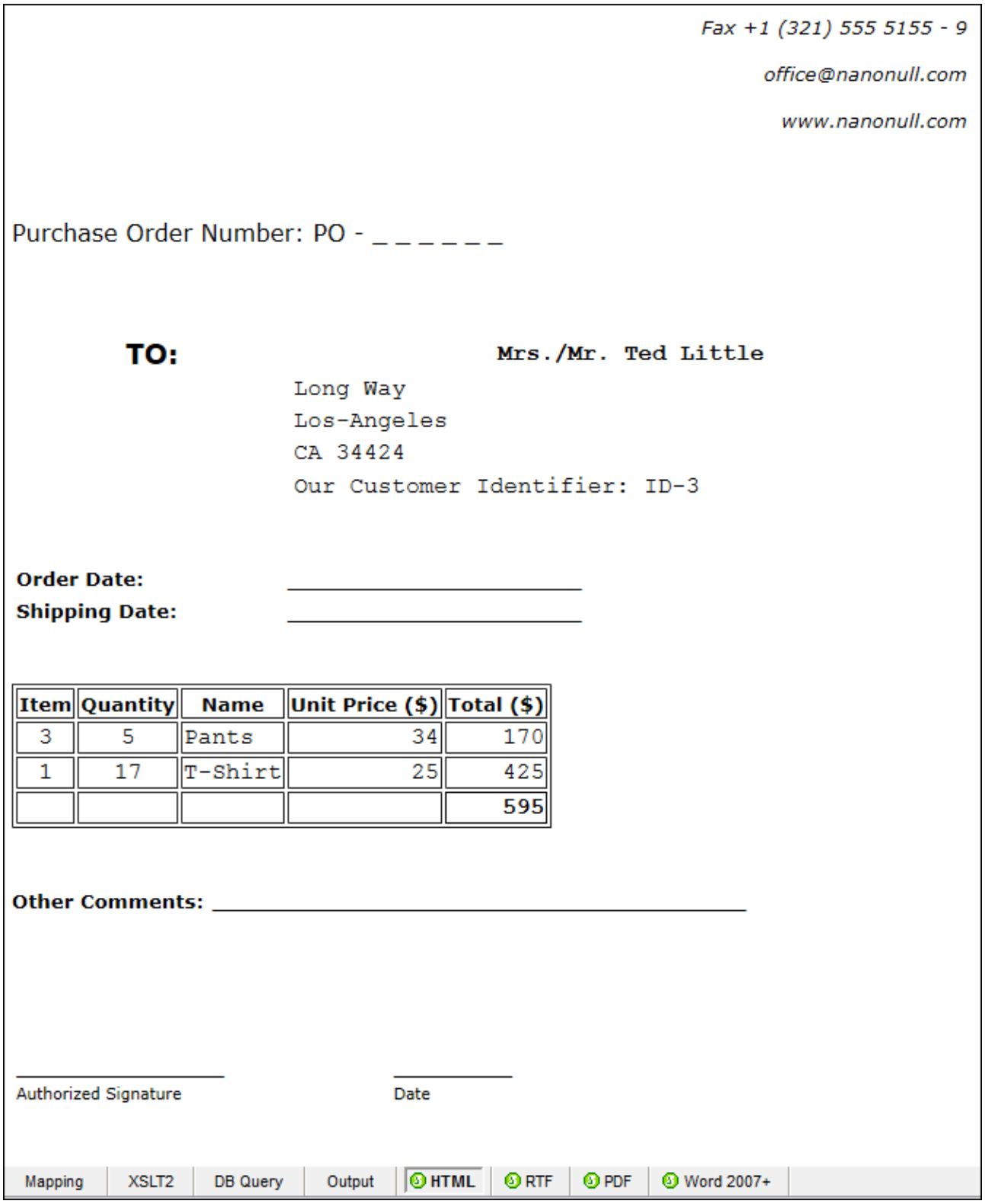
MapForce サンプルフォルダー(<マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥) 内のマッピングの多くは、ターゲットコンポーネントに割り当てられているStyleVision Power Stylesheets (.sps ファイル) を含んでいます。この場合、マッピングはをStyleVision プレビュータブ含んでいます。

StyleVision プレビュータブ(MapForce Enterprise Edition)

次のパスで見つけることのできる **CompletePO.mfd** はスタイルシートを使用したマッピングの一例です: **<マイドキュメント>\Altova\MapForce2021\MapForceExamples\CompletePO.mfd**。このマッピングは、XML フォーマットで注文を表示します。ターゲットコンポーネントを右クリックして、プロパティを選択すると、.sps ファイルが割り当てられていることに気が付きます。



HTML フォーマットで表示される出力データを確認するために出力 タブをクリックします。



Fax +1 (321) 555 5155 - 9

office@nanonull.com

www.nanonull.com

Purchase Order Number: PO - _ _ _ _ _ _

TO: **Mrs./Mr. Ted Little**

Long Way
Los-Angeles
CA 34424
Our Customer Identifier: ID-3

Order Date: _____

Shipping Date: _____

Item	Quantity	Name	Unit Price ($)	Total ($)
3	5	Pants	34	170
1	17	T-Shirt	25	425
				595

Other Comments: _____

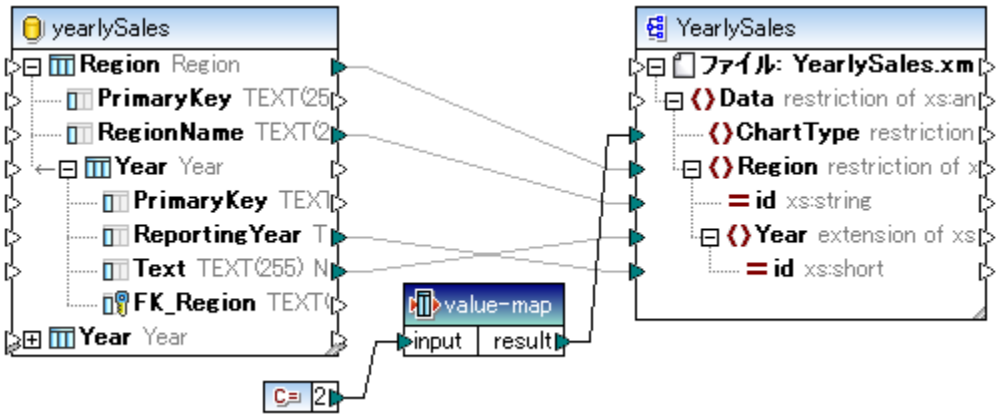
_____ _____
Authorized Signature Date

Mapping | XSLT2 | DB Query | Output | HTML | RTF | PDF | Word 2007+

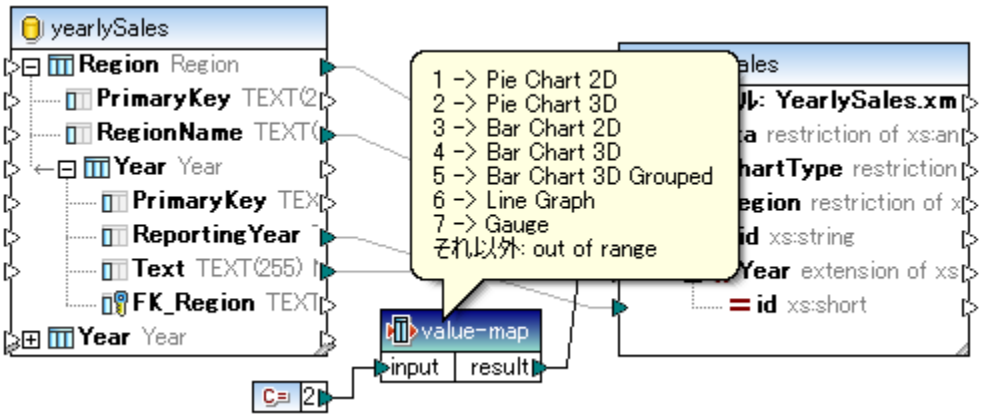
CompletePO.mfd のHTML プレビュー(MapForce Enterprise edition)

(C) 2015-2021 Altova GmbH

他のサンプルは**YearlySales.mfd** です。このマッピングは、以下のパスで見つけることができます: **<マイドキュメント>\Altova\MapForce2021\MapForceExamples\Tutorial\YearlySales.mfd**。



このマッピングに割り当てられているスタイルシートは、(StyleVision を使用して) デザインされており、`ChartType` 要素の値を変更して、チャートの型の管理が可能することができます。これにより、マッピングから直接チャートの型を変更することができます。具体的な、値の定数のデフォルトの値を 1 から 7 に変更することができます。**value-map** コンポーネントにマウスをポイントすると、使用可能な値が表示されます。



出力内での 3D 円グラフを生成する定数のデフォルトの値は、"2" です。他の種類のチャートを表示するには、他の許可されている値に変更し、結果を確認するために**出力** タブをクリックします。

4.18 マッピングドキュメンテーションの生成

ドキュメンテーションの生成コマンドにより、マッピングに関する詳細なドキュメンテーションをHTML、Microsoft Word (.doc)、RTF 書式で生成することができます。StyleVision がインストールされているとPDF 書式でドキュメントを生成することもできます。

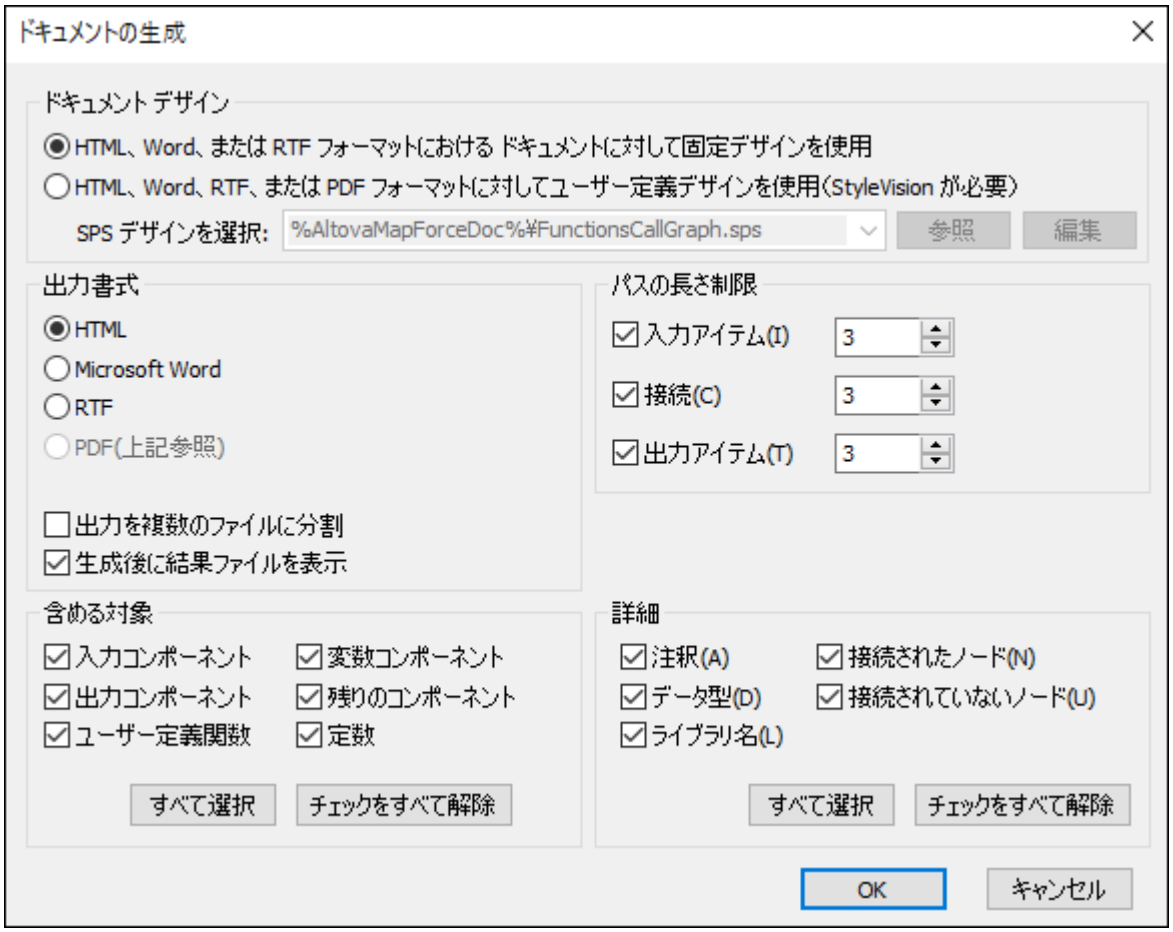
メモ

- MS Word フォーマットにてドキュメンテーションの生成を行うには、MS Word (バージョン2000 以降) がインストールされている必要があります
- PDF 書式でドキュメントを生成、または生成されたドキュメントのデザインをカスタマイズする場合 StyleVision がインストールされている必要があります。

デフォルトではドキュメンテーションは含まれるコンポーネント、表示されるパスの深さ、および他の設定を構成することのできる固定されたデザインで生成されます。StyleVision がインストールされている場合、含まれているStyleVision Power Stylesheets (SPS) ファイルの追加機能を活用、またはStyleVision 内で自身のデザインを作成することができます。

マッピングドキュメンテーションの生成方法:

1. 「ファイル」メニューから「ドキュメントの生成」をクリックします。これにより「ドキュメントの生成」ダイアログボックスが開かれます。



2. 必要とされる設定を選択し「OK」をクリックします。

(C) 2015-2021 Altova GmbH

構成可能な設定は下に説明されています。

ドキュメントのデザイン

- 「固定デザインを使用」を選択して内蔵のドキュメンテーションテンプレートを使用します。
- 「ユーザー定義デザインを使用」を選択して StyleVision 内で作成された定義済み StyleVision Power Stylesheet を使用します。**...\Documents\Altova\MapForce2021\Documentation\MapForce** フォルダー内で SPS ファイルを使用することができます。詳細に関しては、定義済み StyleVision パワースタイルシートを参照してください。
- 「**参照**」をクリックして、定義済みの SPS ファイルを参照します。
- 「**編集**」をクリックして StyleVision をローンチし、StyleVision ウィンドウ内で選択された SPS を開きます。

出力書式

- 次の出力書式の一つを選択します: HTML、Microsoft W ord、RTF、または PDF. Microsoft W ord ドキュメントは固定されたデザインを使用する場合 **.doc** ファイル拡張子と共に作成され、StyleVision SPS を使用して生成する場合 **.docx** ファイル拡張子と共に作成されます。ます。PDF 出力書式は StyleVision を必要とし、StyleVision SPS を選択した場合のみ使用することができます。
- 入力、または出力コンポーネントなどの個別のコンポーネント別に 1 つのファイルを与えるなど、複数のドキュメンテーションファイルを生成する場合「**出力を複数のファイルに分割**」を選択します。固定されたデザインを使用する場合、複数のドキュメント間のリンクが自動的に生成されます。
- 「**生成後に結果ファイルを表示**」オプションが選択されている場合、MapForce は生成されたファイルを適用できる場合デフォルトのブラウザー、またはアプリケーション内で開きます。

パスの長さの制限

入力、または出力接続のために表示される パスの長さの最大値を定義するためにこれらのオプションを使用します。例えば、デフォルトの長さが **3** の場合、アイテム パスは以下のようになります: **.../ShortPO/LineItems/LineItem**。

含まれる内容

生成されたドキュメント内に含まれる特定のコンポーネントをここで選択します。

詳細

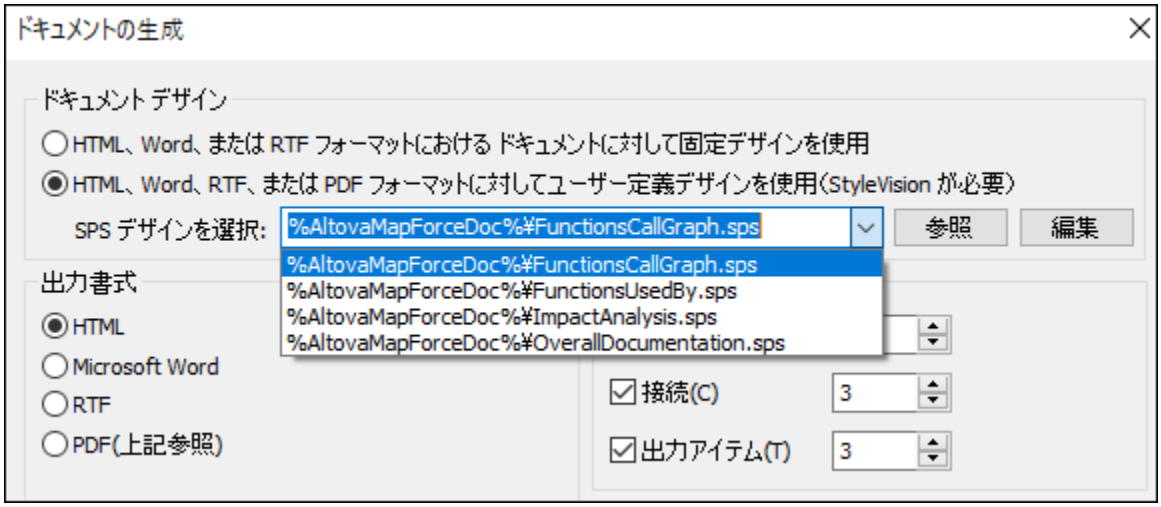
生成されたドキュメンテーション内での詳細のレベルをカスタマイズするオプションを使用します。「**ライブラリ名**」オプションは巻数のために "core" プレフィックスを挿入します。

4.18.1　定義済みの StyleVision パワースタイルシート

W hen StyleVision がコンピューターにインストールされると、ビルトインの固定されたデザインの代わりに定義済みの StyleVision Power Stylesheet（SPS）ファイルをテンプレートとして選択してマッピングドキュメンテーションを生成することができます。次の定義済みの SPS スタイルシートを使用することができます:

- **FunctionCallGraph.sps** – メインマッピングとユーザー定義関数の呼び出しグラフを表示します。
- **FunctionsUsedBy.sps** – マッピング内で直接、または間接的に使用された関数を表示します。
- **ImpactAnalysis.sps** – ソース、およびターゲット ノードとターゲット ノードまでの多種の関数を使用したすべてのルートをリストします。
- **OverallDocumentation.sps** – 全てのノード、コネクタ、関数、およびターゲット ノードを表示します。このテンプレートは最大の詳細を出力し、ビルトインの固定デザイン出力に合致しています。

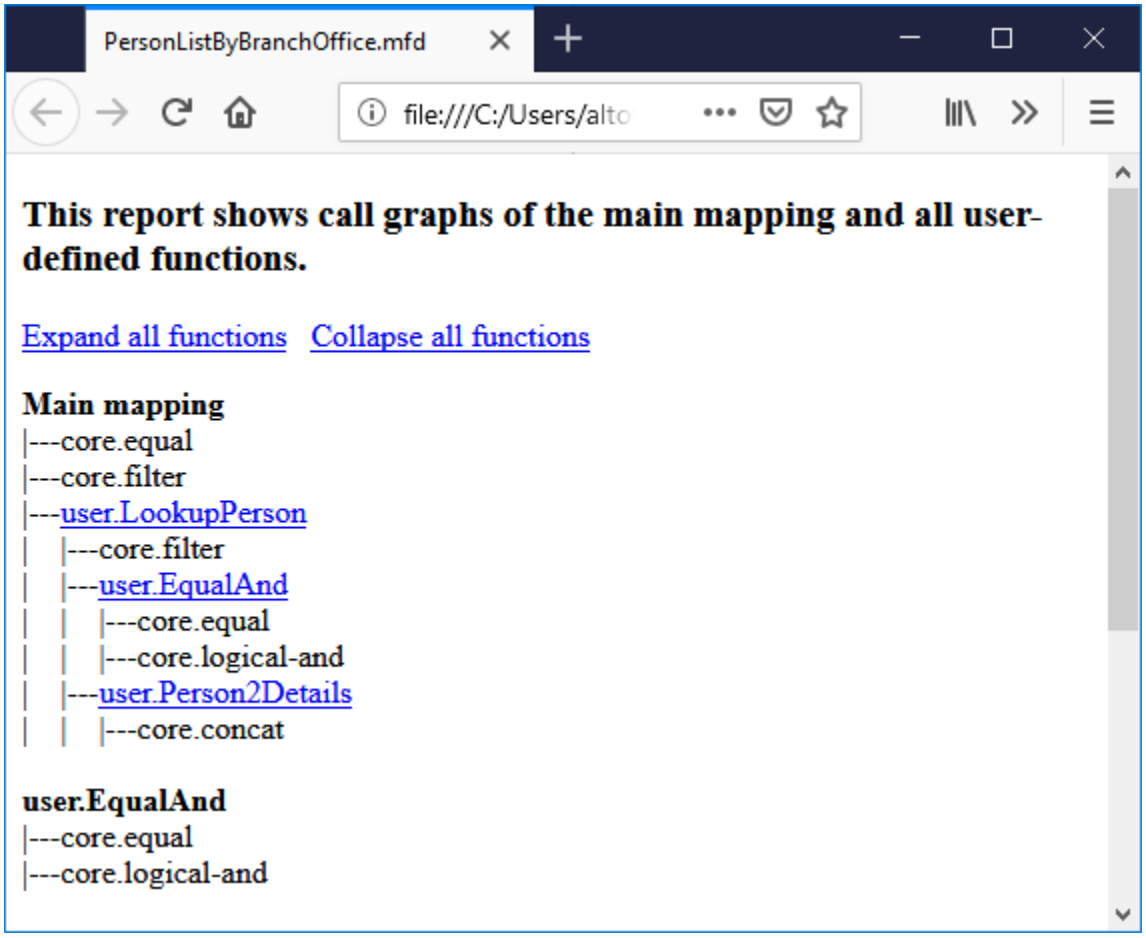
以下に示されるようにドキュメントを生成する前に必要とするスタイルシートを選択することができます。ファイルは... **\MapForce2021\Documentation\MapForce** フォルダー内で見つけることができます。



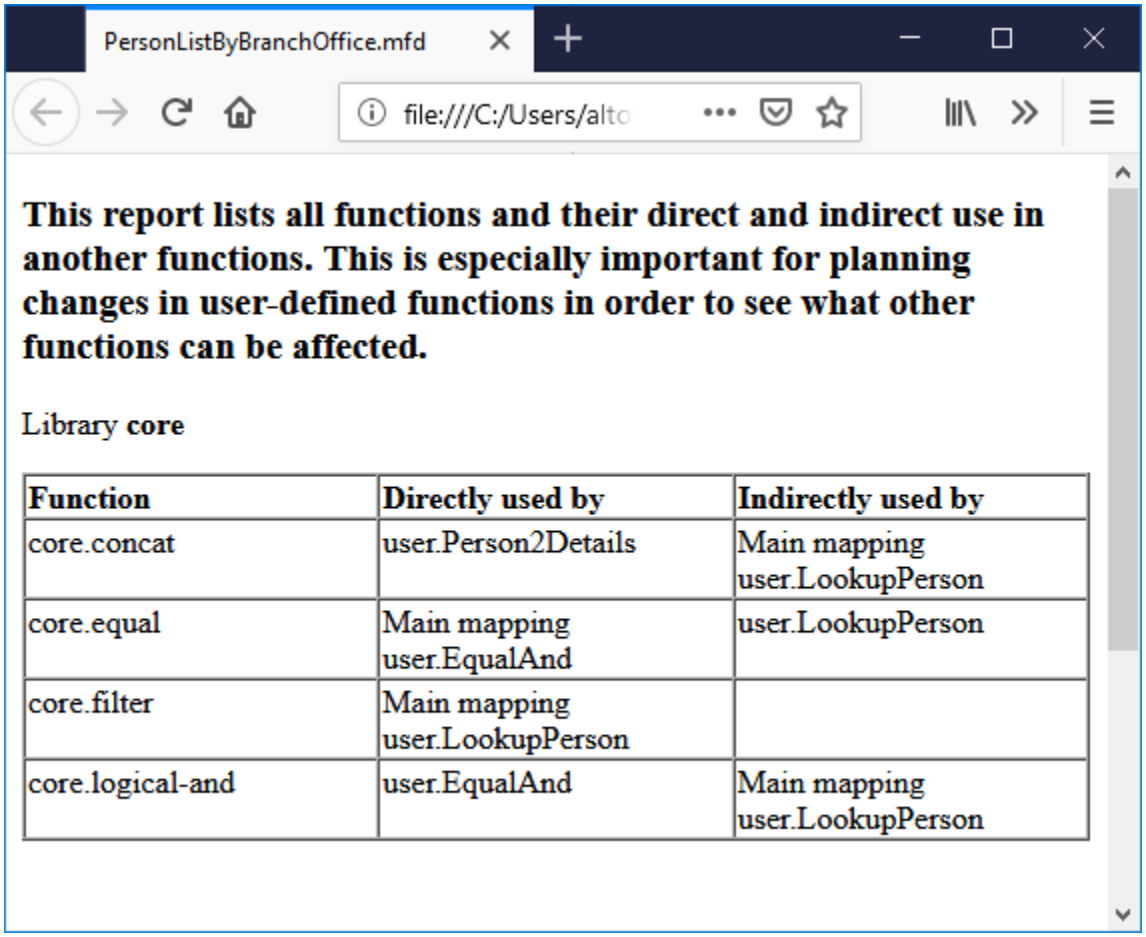
下のサンプルはこれらのスタイルシートにより生成される出力を示しています。サンプルはMapForce を使用してインストールされたデモマッピング の1つ(**PersonListByBranchOffice.mfd**)により生成されています。これらのサンプルはHTML 出力を示していますが、他のフォーマットのレイアウトもこれに類似しています。SPS ファイルの作成、およびカスタム化に関しての情報は、カスタムスタイルシートを参照してください。

(C) 2015-2021 Altova GmbH

スタイルシート "FunctionCallGraph.sps"



スタイルシート "FunctionsUsedBy.sps"



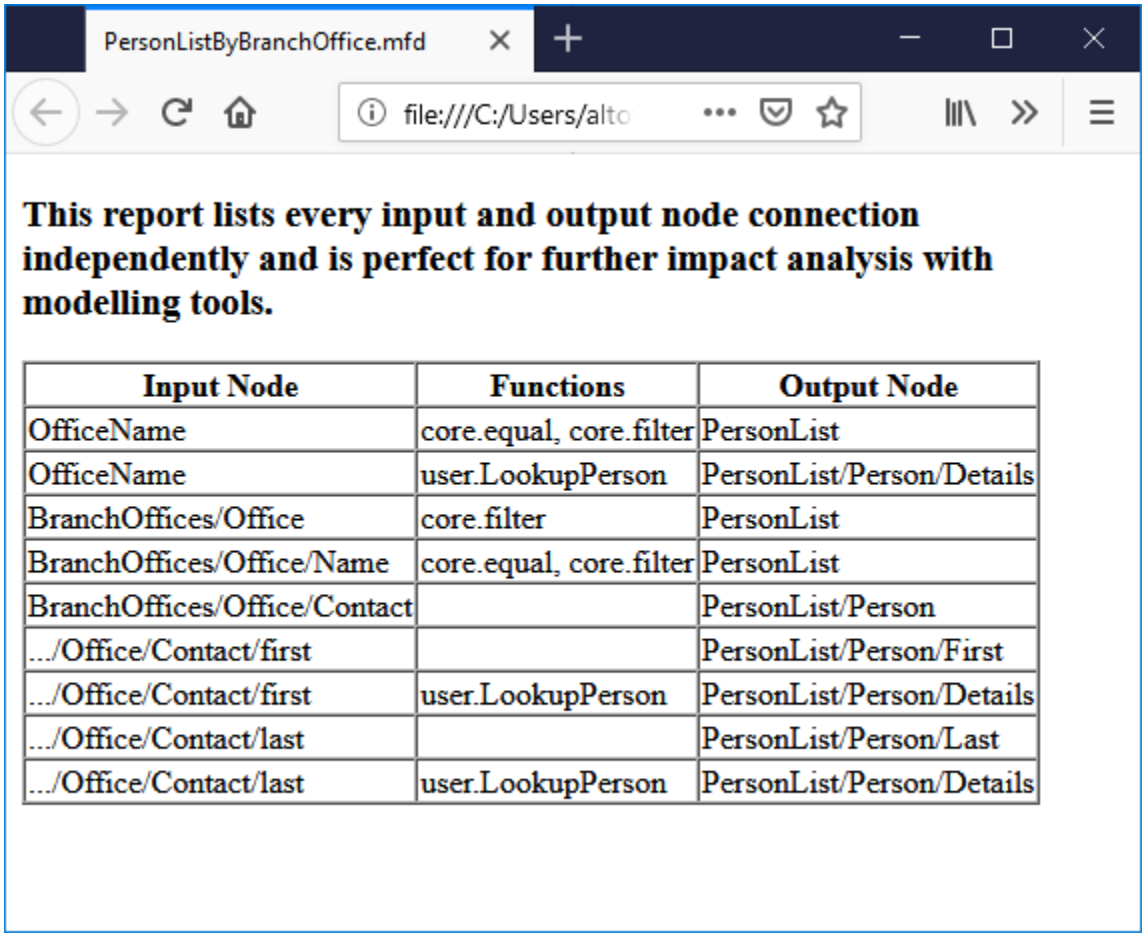
This report lists all functions and their direct and indirect use in another functions. This is especially important for planning changes in user-defined functions in order to see what other functions can be affected.

Library **core**

Function	Directly used by	Indirectly used by
core.concat	user.Person2Details	Main mapping user.LookupPerson
core.equal	Main mapping user.EqualAnd	user.LookupPerson
core.filter	Main mapping user.LookupPerson	
core.logical-and	user.EqualAnd	Main mapping user.LookupPerson

(C) 2015-2021 Altova GmbH

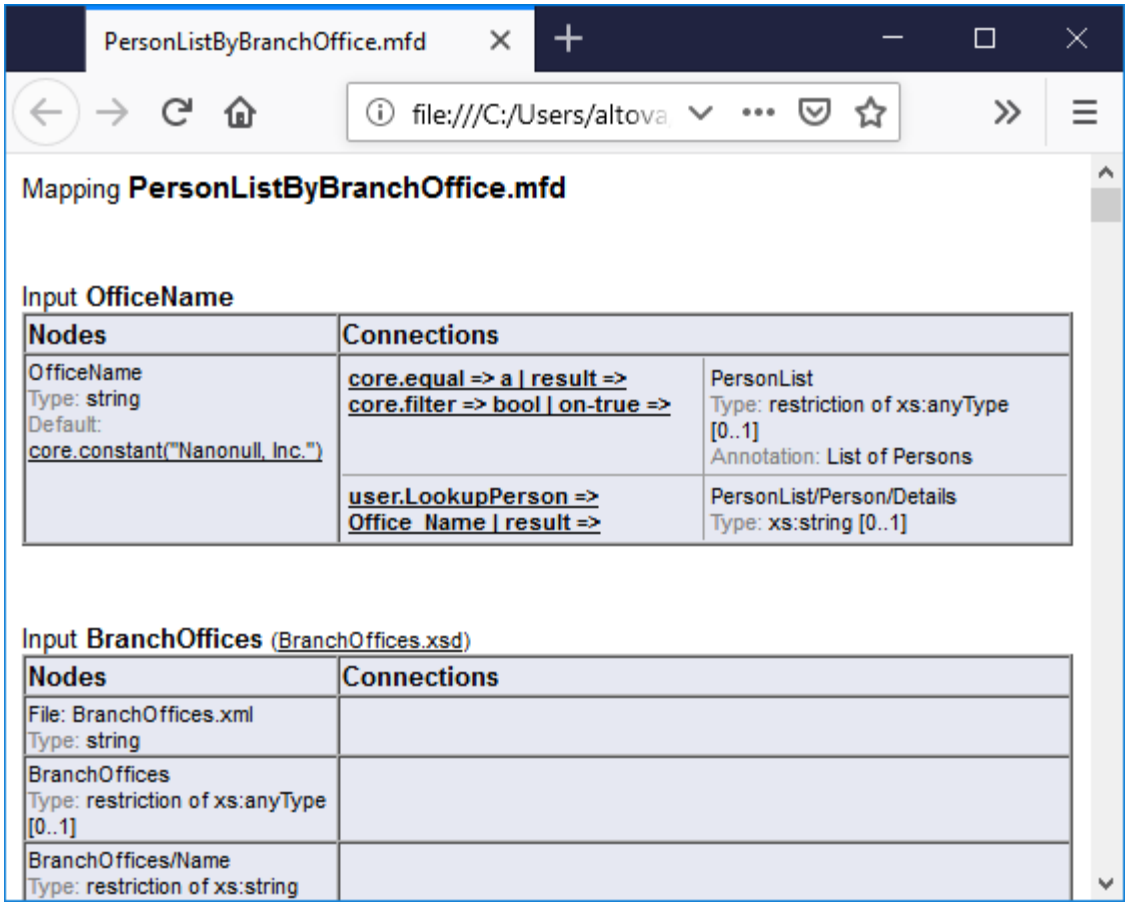
スタイルシート "ImpactAnalysis.sps"



This report lists every input and output node connection independently and is perfect for further impact analysis with modelling tools.

Input Node	Functions	Output Node
OfficeName	core.equal, core.filter	PersonList
OfficeName	user.LookupPerson	PersonList/Person/Details
BranchOffices/Office	core.filter	PersonList
BranchOffices/Office/Name	core.equal, core.filter	PersonList
BranchOffices/Office/Contact		PersonList/Person
.../Office/Contact/first		PersonList/Person/First
.../Office/Contact/first	user.LookupPerson	PersonList/Person/Details
.../Office/Contact/last		PersonList/Person/Last
.../Office/Contact/last	user.LookupPerson	PersonList/Person/Details

Stylesheet "OverallDocumentation.sps"



4.18.2 カスタムスタイルシート

ビルトインデザインに加え、生成されたマッピングのドキュメンテーションのために StyleVision (https://www.altova.com/ja/stylevision) を使用してカスタムスタイルシートを作成することができます。フォントと他のスタイルを調整して 定義済みのスタイルシート を変更することもできます。例えば、

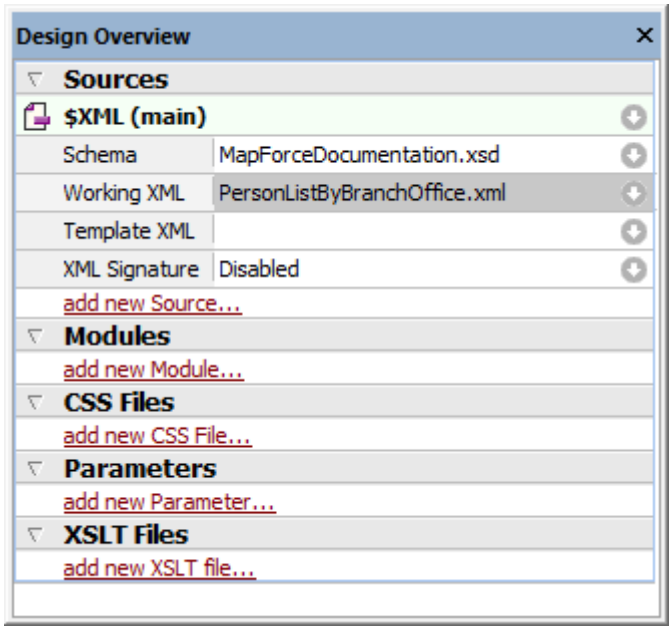
カスタムデザインとは StyleVision Power StyleSheet (SPS) です。マッピングドキュメンテーションの生成に SPS を使用する利点はドキュメントのデザインを完全に管理できる点です。

カスタム SPS ファイルを作成するには、以下を必要とします:

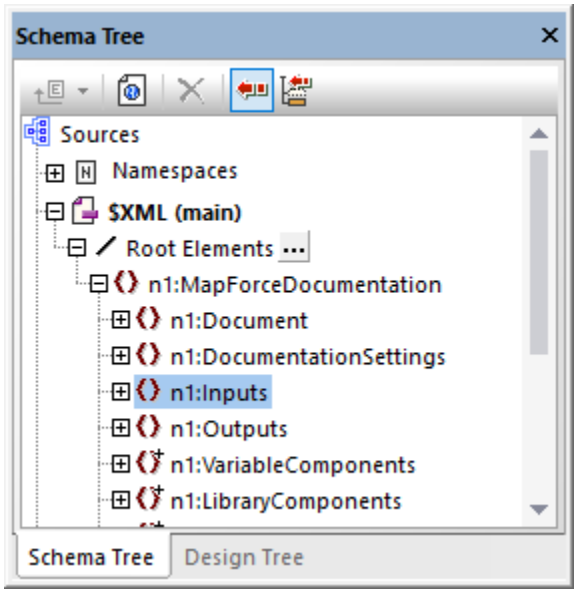
1. 生成された MapForce ドキュメンテーションの構造を提供する XML スキーマ。このスキーマは **MapForceDocumentation.xsd** と呼ばれ MapForce インストールパッケージに搭載されています。... **\Documents\Altova\MapForce2021\Documentation\MapForce** フォルダー内に保管されています。**MapForceDocumentation.xsd** には上のフォルダー内に **Documentation.xsd** ファイルが含まれていることに注意してください。

2. カスタムデザインをテストしプレビューするデータ。次のXML ファイルをサンプルデータとして使用することができます：**...\Documents\Altova\MapForce2021\Documentation\MapForce\SampleData\PersonListByBranchOffice.xml**.

上記のファイルはStyleVision 内のデザイン概要ウィンドウ内で参照される必要があります。例:



StyleVision 内ではスキーマツリーウィンドウからノードをデザインエリアにドラッグし、プロパティを割り当ててデザインを作成することができます。



リンクおよびイメージなどの追加コンポーネントをSPS デザインに追加することもできます。特定のフォーマットをデザイン内でプレビューするには、次のタブをクリックしてください：**HTML**、**RTF**、**PDF**、または**Word 2007+**。詳細に関してはStyleVision ドキュメンテーションを参照してください(https://www.altova.com/ja/documentation)。

4.19 マッピングのルールと戦略

一般的には、MapForce はデータを直観的にマップしますが、出力に多すぎる、または少なすぎるアイテムが含まれているシチュエーションが存在します。このチャプターは性格ではない接続やマッピングコンテキストが理由でマッピングが希望しない出力を生成するシチュエーションを回避するための手助けを意図としています。

マッピングのルール

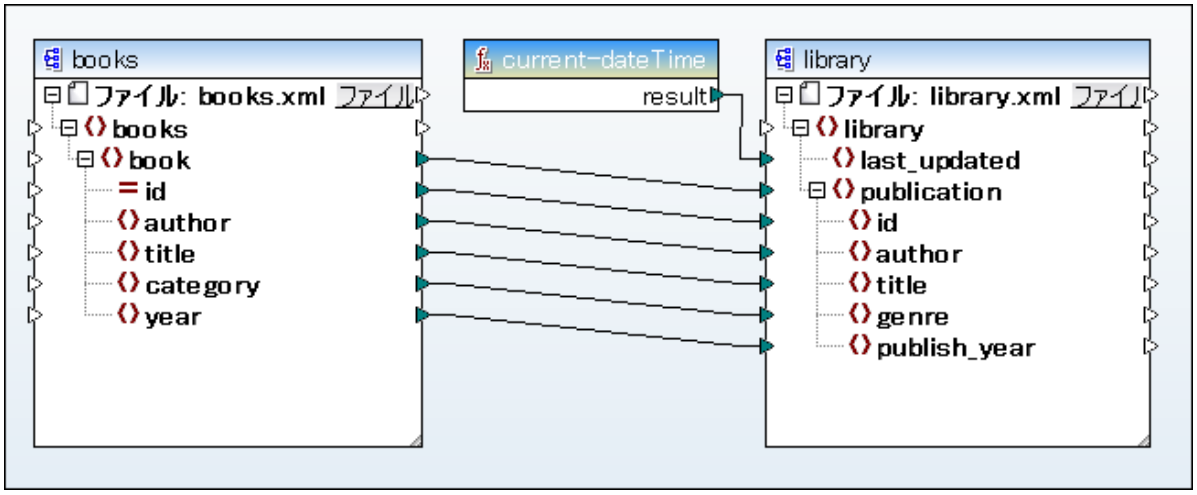
有効であるためには、マッピングには少なくとも 1 つのソースコンポーネントと少なくとも 1 つのターゲットコンポーネントが含まれている必要があります。ソースコンポーネントはデータを通常はファイルまたはデータベースから読み取ります。ターゲットコンポーネントは通常ファイルまたはデータベースでデータを書き込みます。上記が正しくない個所でマッピングを保存しようとすると、メッセージウィンドウにエラーが表示されます:「マッピングは少なくとも2 つの接続された構造を必要とします。例えば、スキーマまたはデータベース構造を必要とします。」

データマッピングを作成するには、マッピング接続をソースとターゲットコンポーネント間に描きます。

描かれたマッピング接続全てはマッピングアルゴリズムを作成しています。マッピングランタイムでは、MapForce はアルゴリズムを評価し、それに基づきデータを処理します。マッピングアルゴリズムの統一性と効率性は接続により異なります。マッピング レベル、コンポーネント レベル、または 接続 レベルで設定の一部を調整することもできますが、基本的にはマッピング接続がデータの処理方法を決定します。

接続を作成する際に以下のルールが存在することに注意してください。:

1. ソースアイテムから接続を描く場合、マッピングはソースファイルまたはデータベースからのアイテムと関連したデータを読み取ります。データは0、1、または複数の発生が存在する場合があります(すなわち、シーケンスである場合もあります。例えば、マッピングが書籍を含むXML ファイルからデータを読み込む場合、ソースXML ファイルは0、1、または複数の**book** 要素を含む可能性があります。下のマッピングでは、ソース(*インスタンス*)ファイルには複数の**book** 要素、または要素が含まれない場合がありますが、**book** アイテムはマッピングコンポーネント上で一度のみ表示されます。



2. ターゲットアイテムに接続を描く場合、マッピングはその種類のインスタンスデータを生成します。ソースアイテムが単純なコンテンツを含む場合(例えば、文字列または整数)およびターゲットアイテムが単純なコンテンツを受け入れる場合、MapForce はコンテンツをターゲットアイテムにコピーし、必要な場合は、データ型を変換します。受信するソースデータに従いゼロ、1、複数の値を生成することができます。次の点を参照してください。
3. ソース内の各(*インスタンス*)アイテムのために 1 つの(*インスタンス*)アイテムがターゲット内で作成されます。**MapForce 内の一般的なマッピングルールです。**上のマッピングを例として、ソースXML が3つの**book** 要素を含む場合、3つの**publication** 要素がターゲット側で作成されます。特別なケース シーケンス がいくつか存在することにも注意してください。を参照してください。
4. 各接続は*現在のマッピングコンテキスト* を作成します。コンテキストはどのデータが現在のターゲットノードのためにどのデータが使用できるかを決定します。コンテキストは、ですから、子のソースアイテムが実際にソースからターゲットコンポーネントにコピーされるかを決定します。接続を作成、または削除する場合、現在のコンテキストが不注意に変更される可能性があり、マッピングの出力に

(C) 2015-2021 Altova GmbH

影響を与える場合があります。例えば、同じマッピングが不必要にデータベース、または、Web サービスを呼び出す場合があります。このコンセプトは下のマッピングコンテキストで詳細に説明されています。

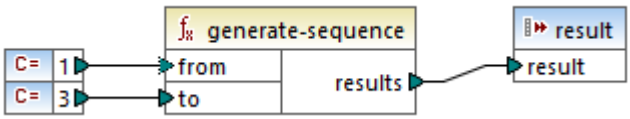
4.19.1 シーケンス

以前に記述されている通り一般的なマッピングルールは「ソース内の各アイテムのために、ターゲット内に 1 つ作成する」です。ここでは、「アイテム」は以下の 1 つを意味します:

- 入力ファイルまたはデータベースの単一のインスタンスノード
- 入力ファイルまたはデータベースのゼロから複数のインスタンスノードのシーケンス

マッピングの実行中、シーケンスがターゲットアイテムに達成すると、ソースノードの数と同じ数量のターゲットを生成するルールを作成します。いくつかの例外がこのルールには存在します:

- ターゲットアイテムが XML ルート要素の場合、1 度のみ作成されます。シーケンスを接続すると、結果はスキーマに対して有効でない場合があります。ルート要素の属性も接続されている場合 XML のシリアル化はマッピングのランタイムで失敗します。このため、ルート XML 要素にシーケンスを接続することを回避してください。
- ターゲットが 1 つの値のみ受け入れる場合、1 度のみ作成されます。アイテムのサンプルは 1 つの値にのみより受け入れられます: XML 属性、データベースフィールド、単純型出力コンポーネント。例えば、下のマッピングは `generate-sequence` 関数を使用して 3 つの整数 (1, 2, 3) のシーケンスを生成します。しかしながら、ターゲットは単一の値を受け入れる単純型出力コンポーネントのため出力は 1 個の整数を含みます。他の 2 つの値は無視されます。



- ソーススキーマが特定のアイテムが 1 度のみ発生するが、インスタンスファイルには多数のアイテムが存在する場合、MapForce は (スキーマに従うと必ず存在する必要がある) ソースから最初のアイテムを抽出し、ターゲット内で 1 個のアイテムを作成します。この振る舞いを無効化するには、チェックボックスをコンポーネント設定から解除します。この振る舞いを無効化するには、チェックボックス「min/maxOccurs をベースに入力処理最適化するを有効化する」コンポーネント設定内でクリアしてください。この振る舞いを無効化するには、コンポーネント設定からクリア、または、 XML コンポーネント設定 を参照してください。

シーケンスが空の場合、ターゲット側には何も生成されません。例えば、ターゲットが XML ドキュメントで、ソースシーケンスが空の場合、XML 要素はターゲット内に作成されません。

関数も同様な方法で作動します: 入力としてシーケンスを取得すると、シーケンス内のアイテムの数量だけ呼び出し (その数の出力を) 作成します。

> 関数が空のシーケンスを入力として取得すると、空の結果を返し、この結果出力は生成されません。

しかしながら、関数の一部のカテゴリでは、デザインのために入力として空のシーケンスを受けると空の値を返します。

- `exists`, `not-exists`, `substitute-missing`
- `is-null`, `is-not-null`, `substitute-null` (これらの 3 つの関数は前の 3 つとエイリアスです)
- 集計関数 (`sum`, `count`, など)
- (インラインではなく) 正規 関数であるユーザー定義関数

空の値を置き換えるには、add the `substitute-missing` 関数をマッピングに追加し、空の値を代替地と置き換えます。代わりに、同じ結果を デフォルトとノード関数を使用してえることもできます。

関数には複数の入力が存在する場合があります。シーケンスが各入力に接続されている場合、通常希望される出力ではありませんが、全ての入力の連結された結果を出力することができます。これを回避するために、1つのシーケンスのみを複数のパラメーターを持つ関数に接続してください。他のすべてのパラメーターは親または他のコンポーネントからの「単純型」アイテムに接続されている必要があります。

4.19.2 マッピングコンテキスト

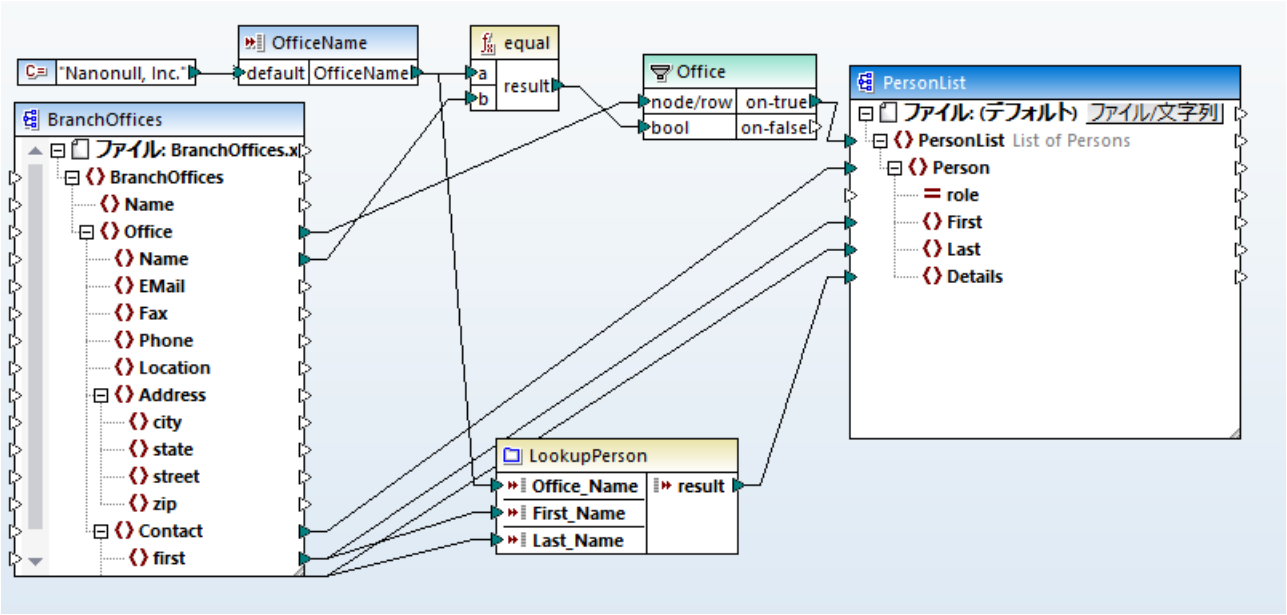
マッピングコンポーネントは深さの多くのレベルを含む階層的な構造です。他方、マッピングは複数のソースとコンポーネントおよび関数、フィルター、値マップなどの中間コンポーネントを持ちます。これは複数の関連性の内コンポーネントが接続されているとマッピングアルゴリズムに複雑性を追加します。マッピングをこのポジションで実行するには、現在のコンテキストがそれぞれの接続で確立される必要があります。

現在のコンテキストは処理済みの各接続と共に変更されるため、複数の「現在のコンテキスト」がマッピングの実行の期間に構築されていると言うことができます。

MapForce は *ターゲットルートアイテム（ノード）* 空開始する現在のコンテキストを常に構築します。これはマッピングの実行が実際に開始する箇所です。ターゲットルートアイテムへの接続は、関数または他の中間コンポーネントを含む直接または間接的に接続されている全てのソースアイテムにトレースバックされます。関数により生成された全てのソースアイテムと結果は現在のコンテキストに追加されます。

ターゲットノードの処理後、MapForce は階層的に作動します。具体的には、全てのターゲットコンポーネントのマップされたアイテムが階層的に処理されます。具体的には、上から下にターゲットコンポーネントが処理されます。新規の各アイテムのために、親コンテキストのすべてのアイテムを含む新規のコンテキストが作成されます。このために、ターゲットコンポーネント内のマップされたすべての兄弟アイテムはそれぞれに対して独立していますが、親アイテムのソースデータにアクセスすることができます。

上記が実際にどのように作動するかサンプルマッピング、**PersonListByBranchOffice.mfd** をベースに確認してみましょう。<マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥ディレクトリ内でこのマッピングを見つけることができます。



上のマッピングでは、ソースとターゲットコンポーネントの両方がXML です。ソースXML ファイルには2つの**Office** 要素が含まれています。

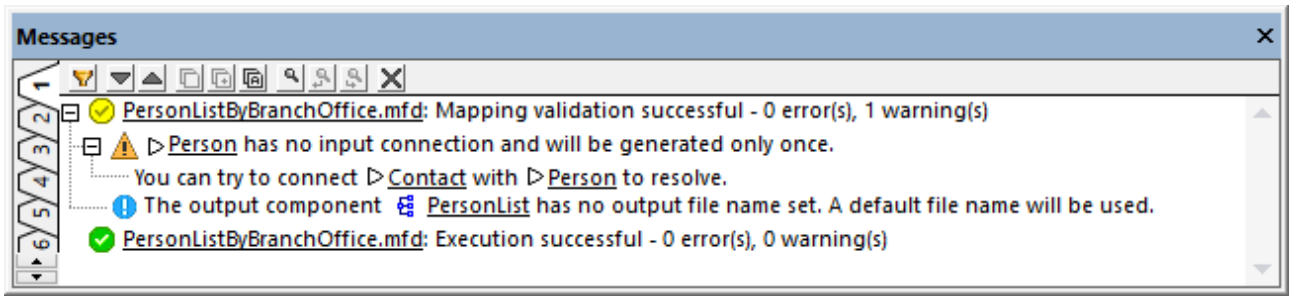
前述の通り、いつも通り、マッピングの実行はターゲットルートノード（このサンプルでは**PersonList** ）から開始されます。（フィルターと関数を介して）接続をソースアイテムにトレースバックすることで、ソースアイテムが**Office** であることが結論付けることができます。（他の接続パスは入力パラメーターへ接続されており、その目的に関しては以下で説明されています）。

Office と**PersonList** の間にわかりやすい接続が存在する場合、一般的なマッピングルールに従い、ソースファイル内の**Office** アイテムの数だけ、マッピングは**PersonList** インスタンスアイテムを作成します。しかしながら、フィルターが存在するためこれが発生しない場合があります。フィルターはフィルターの**bool** 入力に接続されているブール値の条件を満たすデータをターゲットコンポーネントに提供します。`equal` 関数は、オフィス名が「Nanonull, Inc.」に等しい場合 **true** を返します。ソースXML ファイル内のオフィス名が1つしか存在しないため、この条件は一度のみ満たされることができます。

結果的に**Office** と**PersonList** の間尾の接続はターゲットドキュメント全体のために単一のオフィスを定義します。これは**PersonList** アイテムのすべての子孫が現在のコンテキストの他のオフィスではなく "Nanonull, Inc." オフィスのデータにアクセス可能なことを意味します。

次は**Contact** と**Person** 間の接続です。一般的なマッピングルールに従うと、各ソース**Contact** のために1つのターゲット **Person** を作成します。各反復で、この接続は新規の現在のコンテキストを作成します。このため、子接続（**first** to **First**, **last** to **Last**）はデータをソースから各 **Person** のコンテキスト内でターゲットアイテムに提供します。

Contact と**Person** 間の接続をそのままにすると、マッピングにより複数の**First**、**Last** と**Details** ノードを持つ1つの**Person** が作成されます。このような場合、MapForce はメッセージウィンドウ内に警告と提案を表示します。例えば



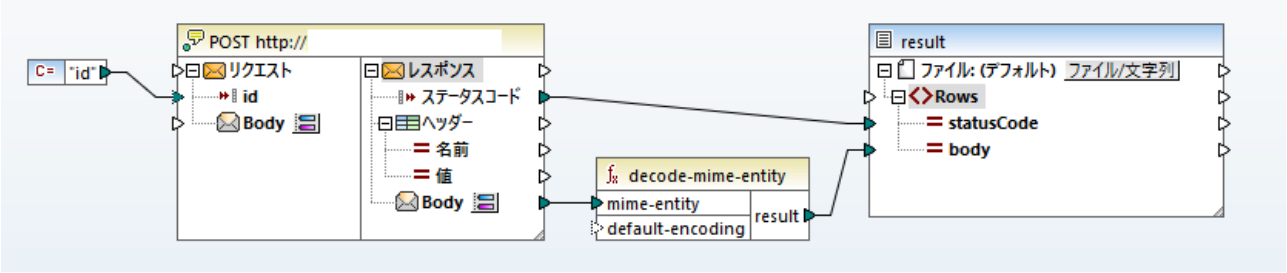
最後に、マッピングにはユーザー定義関数 `LookupPerson` が含まれています。 **Contact** と**Person** 間の親接続のためユーザー定義関数はそれぞれの**Person** コンテスト内で実行されます。新規のPerson アイテムがターゲット側で作成される都度、関数が個人の**Details** 要素を作成するために呼び出されます。この関数は3つの入力パラメーターを取ります。最初のパラメーター（**OfficeName**）はマッピングの入力パラメーターからデータを読み取るように設定されています。マッピング出力を変更することなくこのパラメーターのための**Name** ソースデータはソースアイテムにより提供されることもできます。いずれにせよ、ソース値は親コンテキストからの値と同じです。内部では、ルックアップ関数は引数として取得された値を連結し単一の値を出力します。 詳細に関しては`LookupPerson` 関数のしくみと例: ルックアップと連結を参照してください。

4.19.2.1 Web サービス

Web サービスを含むマッピング内では、コンテキストが許可する限り、通常は回避されるべきですが、同じ Web サービスが複数回呼び出されます。以下のケースを考慮してください:

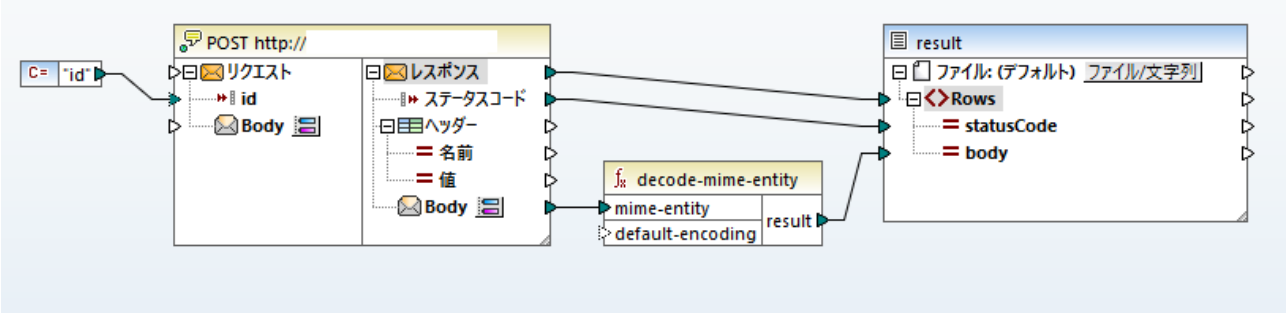
- Web サービスコンポーネントのリクエスト アイテムの接続が *値のシーケンス*を持つ場合、Web サービスはシーケンス内の各アイテムのために複数回呼び出されます。
- Web サービスがユーザー定義関数内にある場合、またはマッピングの実行の間ユーザー定義関数が複数回呼び出される場合 Web サービスは同様複数回呼び出されます。UDF 設定を構成してこれを回避することができます。詳細に関しては、ユーザー定義関数 を参照してください。
- 通常 Web サービスの レスポンス アイテムをターゲットアイテムに接続します:
  - このアイテムは一度のみ作成されることが期待されます（例えば、単純な出力コンポーネント、または複雑なターゲットコンポーネントの一部のトップアイテムなど）
  - また、アイテムの複数の反復を要求しない（結果複数回 Web サービスの呼び出さない）親コンテキストです。

上記の3番目のケースに関しては、マッピング出力は期待されるものですが、 Web サービスが複数回呼び出されることを引き起こします。例えば下で説明される Web サービスはレスポンス から 行への接続が不足しているため2回呼び出されています。



通常通り、ターゲットルートアイテム（この場合行）からマッピングの実行は開始されます。行 は接続を持たないため、MapForce は階層的に下に進行します。**statusCode** ターゲットアイテムを作成するために、最初にW eb サービスへの呼び出しが行われます。サービスへの2番目の呼び出しを行うために次にマッピングは**body** アイテムを作成する必要があります。W eb サービスに接続されている 行 の子が他に存在する場合、それぞれのために新規の呼び出しが実行されます。

この理由は、上のマッピングで**statusCode** または**body** が明示的な親コンテキストを有していないからです（すなわち、親はソースアイテムに接続されていないことを意味します）。この結果、親コンテキストからデータを再利用することができず、この場合はW eb サービスであるソースコンポーネントから個別にデータを抽出しなくてはなりません。以下に表示されるとおり**レスポンス** から 行 に接続線を描くことでこの問題は簡単に解決されます:



レスポンス と 行 間の接続線によりW eb サービスは一度のみ呼び出されます。最初の呼び出しが行われると、W eb サービスデータは現在のコンテキスト内に存在し 行 の子を使用できるようになり、冗長なW eb サービスの呼び出しは行われません。

4.19.2.2 データベース

効率性を向上し、ハードウェアとネットワークリソースの使用を削減するために同じマッピング内で同じデータベースへの不必要な複数回の呼び出しを回避することが奨励されます。マッピングの性質から同じデータベースを複数回呼び出すことを回避できないシチュエーションが存在する場合がありますが、以下の一般的な点を考慮してください：

- データベースを一度のみ呼び出す場合、データベースを複数回呼び出すことを必要とするデータベースコンポーネントを親コンテキスト内に配置することを回避してください。これは、例えば、データベースコンポーネントを値のシーケンスを入力として受け取り、このためにシーケンス内で各アイテムのために呼び出しを受けるユーザー定義関数内に追加すると発生する場合があります。ユーザー定義関数 も参照してください。ターゲットコンポーネントにパスする前に同じコンテキストにデータを集めるために変数は役に立ちます。
- データベースから値を集計する必要がある場合（ `count` 関数を使用してレコードの数量を計算する場合など）、集計関数の出力を**compute-when=once** である箇所の変数に接続することが奨励されます。これは例: テーブル行のカウント内で説明されている通りデータベースへの複数回の呼び出しを回避します。
- マッピングに複数回同じデータベースコンポーネントを追加する代わりに（例えば、SQL-SELECT ステートメント、またはストアドプロシージャなどの）一度の呼び出しでデータベースのデータを抽出することを試みてください。
- 同じデータベースから複数のテーブルまたはビューのデータを抽出する必要がある場合、（ SQL モード）ジョインコンポーネント、またはSQL-SELECT ステートメントを使用することが奨励されます。SQL SELECT ステートメントを自身で作成する場合、後者

のオプションが便利です。データベースデータを非データベースデータ、または異なるデータベースのデータにジョインする必要がある場合、非-SQL ジョインを使用してください。データ集中マッピング内で非-SQL ジョインの実行を最適化するには、マッピングを MapForce Server Advanced Edition を使用して実行してください。

- データベースからデータをフィルターする場合、通常のフィルターの代わりに SQL-WHERE コンポーネントを使用することが効率が良いです。これは前者のコンポーネントが対応するデータベースの文法上、データベースと作業するために最適化されているためです。

4.19.2.3 ユーザー定義関数

ユーザー定義関数（UDF）はマッピング内に埋め込まれている入力、出力、および処理ロジックを定義するカスタム関数です。各ユーザー定義関数にはWeb サービスとデータベースを含むメインマッピングと同じコンポーネントが含まれている可能性があります。

> デフォルトではUDF がデータベースまたはWeb サービスコンポーネントを含む場合、そして、UDF への入力データが複数の値のシーケンスの場合、各入力値はUDF を呼び出し、データベースまたはWeb サービス呼び出しが結果として行われます。

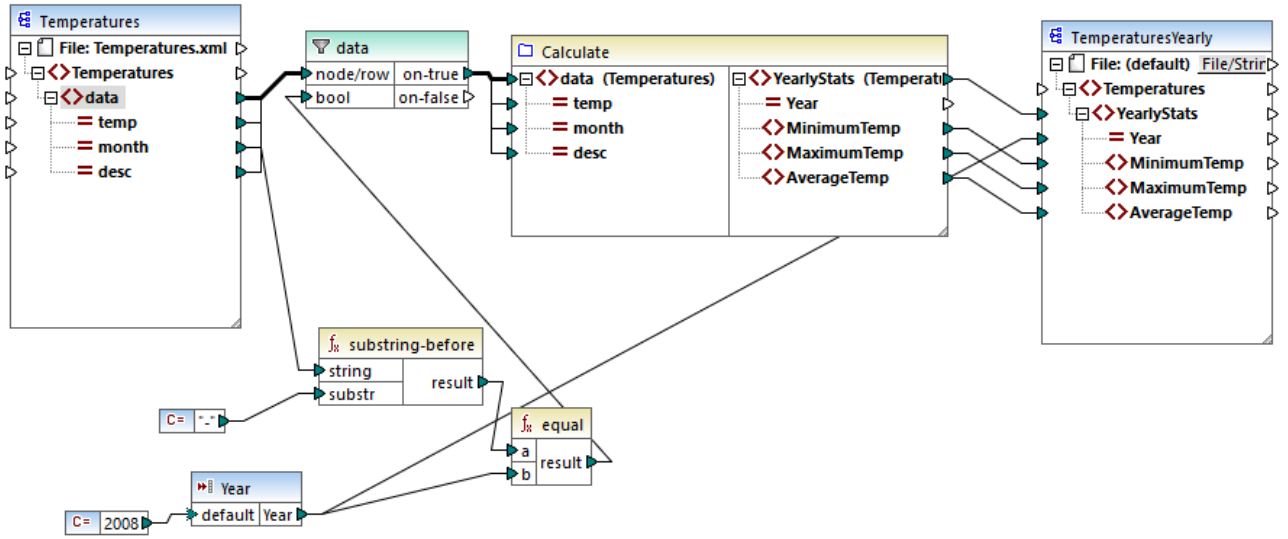
これらの振る舞いは、代替法が無く *入力の値の数量の回数* UDF を呼び出す必要のあるマッピングの箇所で受け入れられます。

この発生を希望しない場合、UDF を構成して、入力として値のシーケンスを取得するように呼び出しを1 回に制限するように構成することができます。通常これを返される前に値のセットを操作する（平均または総数を計算する）UDF のために行います。

同じ呼び出し内でUDF を複数の値を受け入れるように構成するにはUDF が型「インライン」ではなく「標準」の場合可能です。（詳細に関しては、ユーザー定義関数 チャプターを参照してください。）標準関数では、「入力はシーケンスです」チェックボックスを選択することにより入力パラメーターがシーケンスであることを指定することができます。入力パラメーターのタイトルバーをダブルクリックするとこのチェックボックスはコンポーネント設定で表示されます。このチェックボックスは以下に示される通り関数の呼び出しの頻度に影響を与えます:

- 入力データがシーケンスパラメーターに接続されていると、ユーザー定義関数は*一度のみ呼び出され*シーケンス全体がユーザー定義関数にパスされます。
- 入力データが非―シーケンスパラメーターに接続されていると、ユーザー定義関数は *シーケンス内の各単一アイテムのために それぞれ*呼び出されます。

サンプルとして以下のデモマッピングを開いてください：<マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥**\InputIsSequence.mfd**.

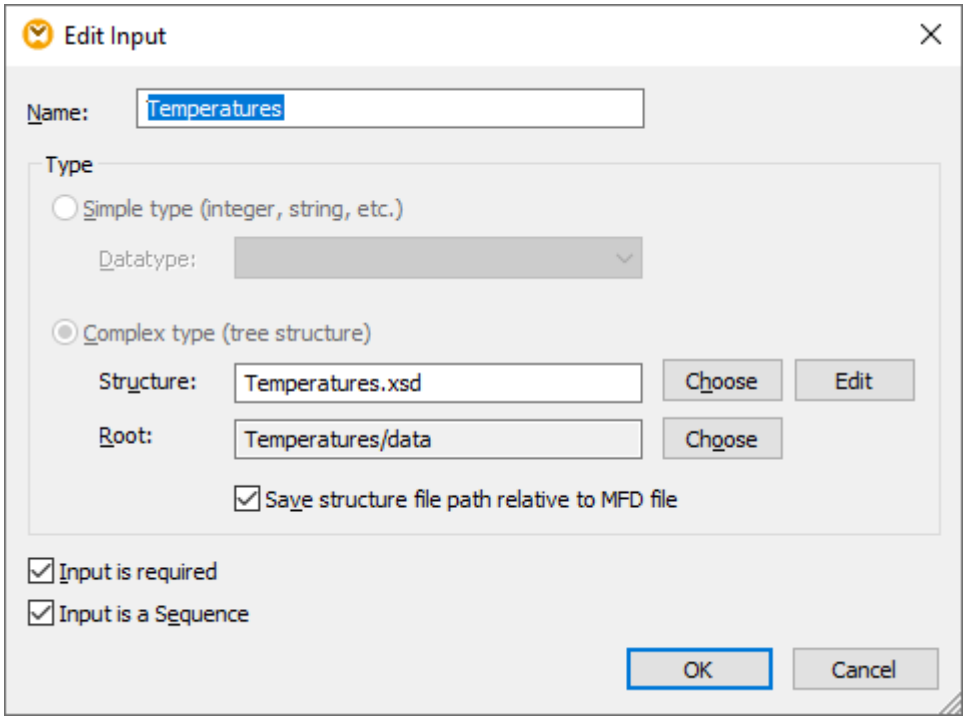


上のマッピングは値のセットを操作し、一度の呼び出し内で全ての入力値を必要とするUDF の典型的なケースを表しています。具体的には`Calculate` ユーザー定義関数は、XML ファイルから入力データを取り、最小、最大、および平均の気温をかえします。期待されるマッピングの出力は以下のとおりです:

```
<Temperatures>
    <YearlyStats Year="2008">
        <MinimumTemp>-0.5</MinimumTemp>
        <MaximumTemp>24</MaximumTemp>
        <AverageTemp>11.6</AverageTemp>
    </YearlyStats>
</Temperatures>
```

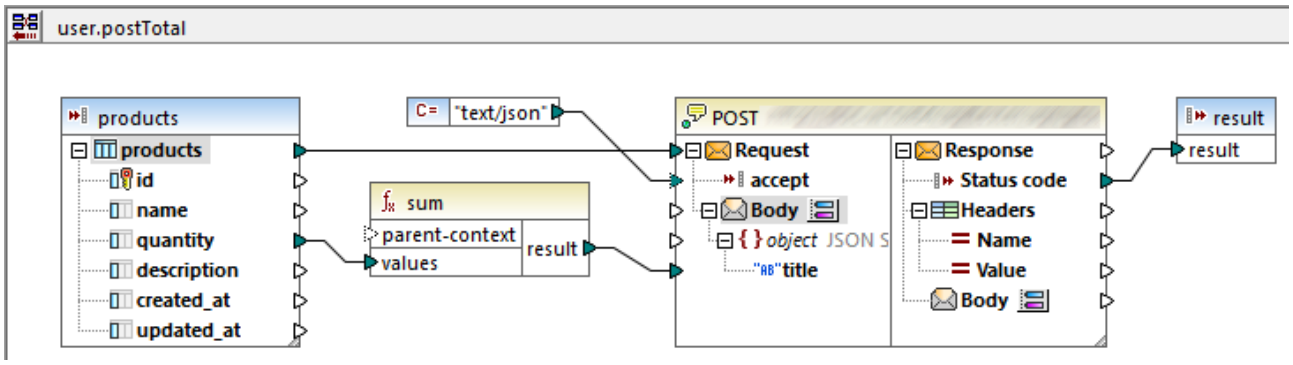
通常どおりターゲットコンポーネントのトップアイテム(このサンプルでは**YearlyStats**)からマッピングの実行は開始されます。このノードを作成するにはフィルターをトリガーするUDF からソースデータを取得しようと試みます。このマッピング内のフィルターの役割は2008 からの気温のみをUDF にパスすることです。

チェックボックス「**入力はシーケンスです**」がUDF の入力パラメーターのために選択されています(このチェックボックスを確認するには、`Calculate` 関数のタイトルバーをダブルクリックし関数のマッピングを入力し、入力パラメーターのタイトルバーをダブルクリックします)。上記の通り「**入力はシーケンスです**」オプションは関数に入力として提供された値のシーケンスを完了し、関数は一度のみ呼び出されます。



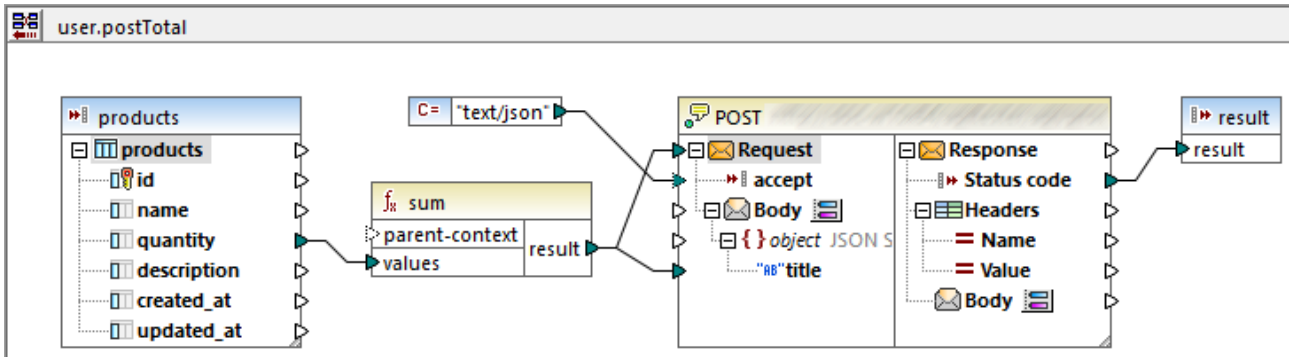
「入力はシーケンスです」チェックボックスが選択されていない場合、UDF はソース内の各値のために呼び出されます。この結果、最低、最高、および平均の値が各単一の値のために計算され、正確ではない出力が生成されます。

データベースまたは Web サービスの呼び出しを含む更に複雑な UDF に適用することで、実行を最適化し、データベースまたは Web サービスへの不必要な呼び出しを回避することができます。「入力はシーケンスです」チェックボックスは関数に入力 *後* に値のシーケンスに発生する事項を管理することはできません。すなわち、受信する値のシーケンスを Web サービスの入力に接続し複数回呼び出しが行われることを防ぐものはありません次の例を考慮してください：



上記の UDF は外部マッピングから値のシーケンスを受け取ります。具体的には、入力パラメーターに与えられたデータはデータベースから来るものです。入力パラメーターはオプション「入力がシーケンス」を選択しており、シーケンス全体が 1 つの呼び出し内で関数に提供されています。この関数は **quantity** 値を集計し、結果を Web サービスにポストします。1 つの Web サービスの呼び出しが期待されます。しかしマッピングが実行されると Web サービスは正確に複数回呼び出されます。これは Web サービスの **Request** 入力が単一の値ではなく *値のシーケンス* を受け取るからです。

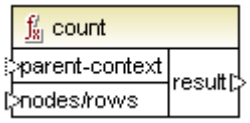
問題を解決するために Web サービスの **Request** 入力を `sum` 関数の結果に接続してください。Web サービスが一度のみ呼び出されるように、関数は 1 つの単一の値を処理します:



通常 sum、count などの集計関数は単一の値を生成します。しかしながら、許可する親接続が存在する場合、値のシーケンスを生成することができます。詳細に関しては以下を参照してください：例: 親コンテキストの変更で詳細に説明されています。

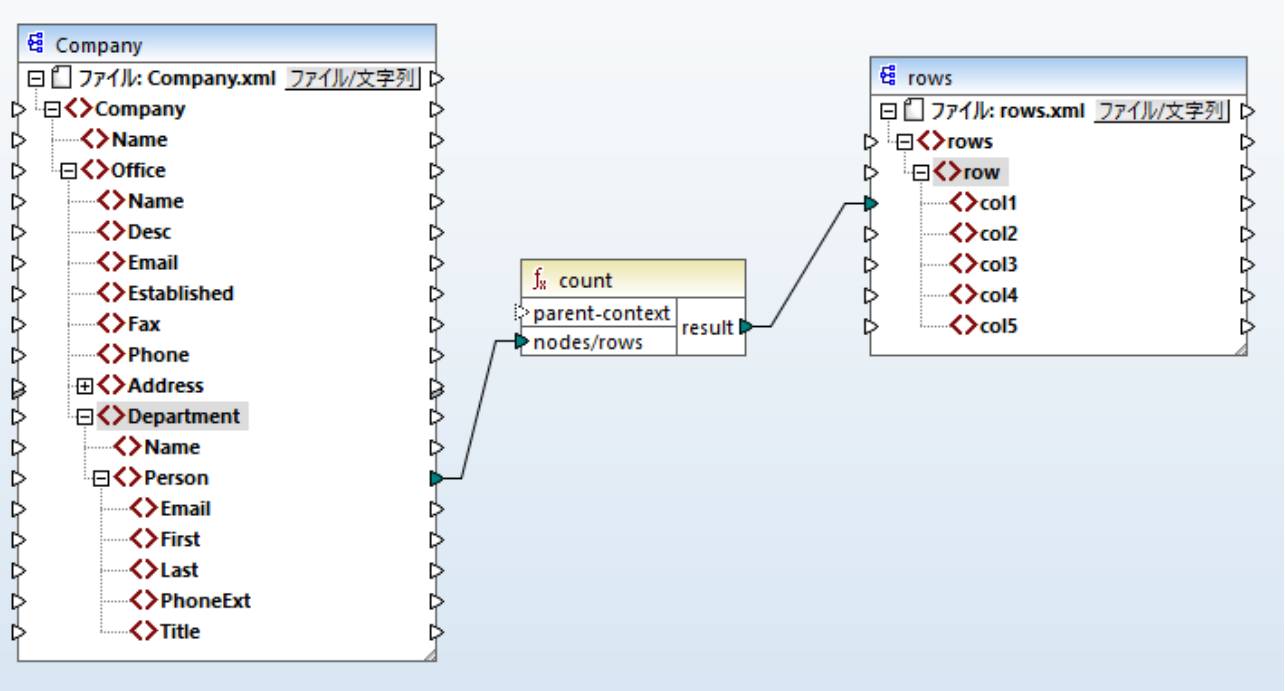
4.19.2.4 例: 親コンテキストの変更

マッピングコンポーネントの一部は任意の**parent-context** アイテムを持っている場合があります。このアイテムを使用して、コンポーネントが作動し結果的にマッピングの出力を変更するようにマッピングコンテキストに影響を与えることができます。任意の**parent-context** を持つコンポーネントは以下の通りです: 集計関数、変数、ジョインコンポーネント。



親コンテキストを変更する方法に関しては次のマッピングを開いてください：<マイドキュメント**>\Altova\MapForce2021\MapForceExamples\Tutorial\ParentContext.mfd**。

(C) 2015-2021 Altova GmbH



上のマッピング内のソースXMLでは、2つの**Office**ノードを含む単一の**Company**が存在します。各**Office**ノードは複数の**Department**ノードを含み、**Department**は複数の**Person**ノードを含んでいます。XMLエディターでXMLファイルを開くと、オフィス内の職員の配属は以下のようになっています:

オフィス	部署	職員数
Nanonull, Inc.	管理	3
	マーケティング	2
	エンジニアリング	6
	IT & 技術サポート	4
Nanonull Partners, Inc.	管理	2
	マーケティング	1
	IT & 技術サポート	3

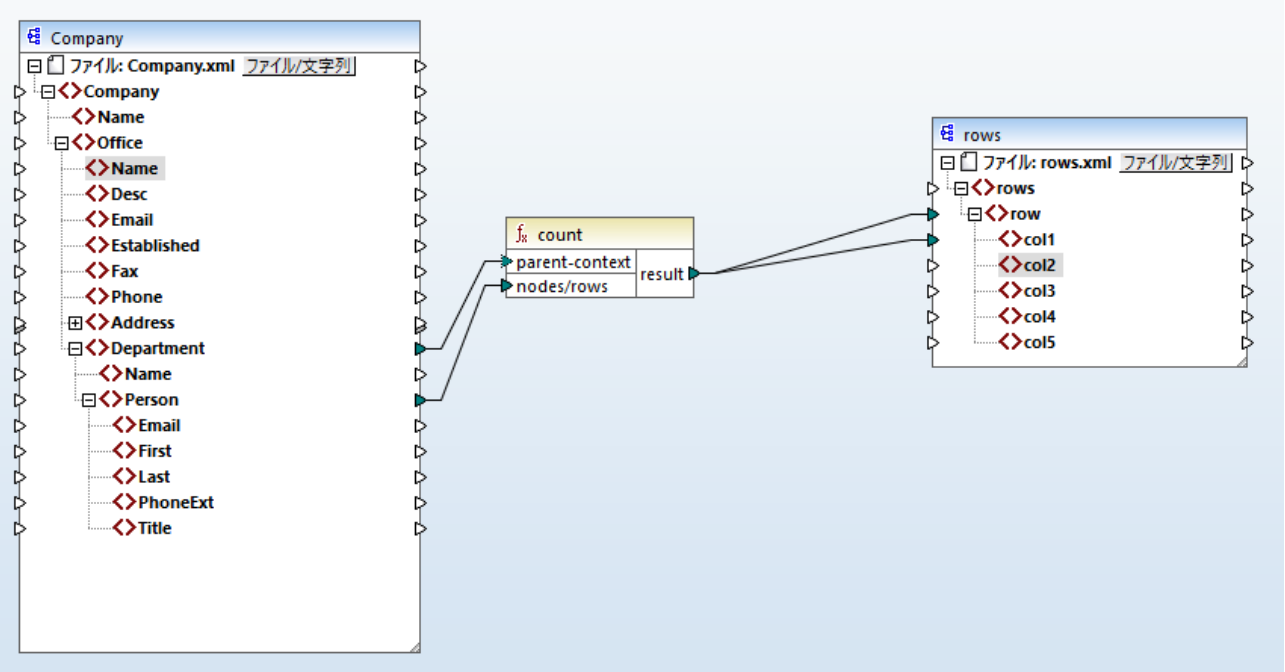
マッピングはすべての部署の職員数を計算します。この目的のために`core`ライブラリから`count`関数を使用します。マッピングをプレビューするために「出力」タブをクリックすると、ソースXMLファイル内の総職員数である単一の値**21**が生成されます。

マッピングの仕組みは以下のとおりです:

- 通常どおりターゲットコンポーネントのトップノード（このサンプルでは**rows**）からマッピングの実行は開始されます。**rows**への接続線は存在しません。この結果**Company**（ソースコンポーネントの上のアイテム）と**rows**（ターゲットコンポーネントの上のアイテム）間の明示的なマッピングコンテキストが確立されます。
- ソースファイルに1つの会社が存在するため関数の結果は単一の値です。

- **col1** ターゲットアイテムを作成するために、MapForce は、全ての部署のすべてのオフィスから **Person** ノードを数えるために、上記の *明示的な親コンテキスト* 内の `count` 関数を実行します。

関数の **parent-context** 引数によりマッピングコンテキストを変更することができます。これにより、例えば、各部署内の職員数を数えることができます。これを行うために、以下に示される通り2本の接続線を描きます:

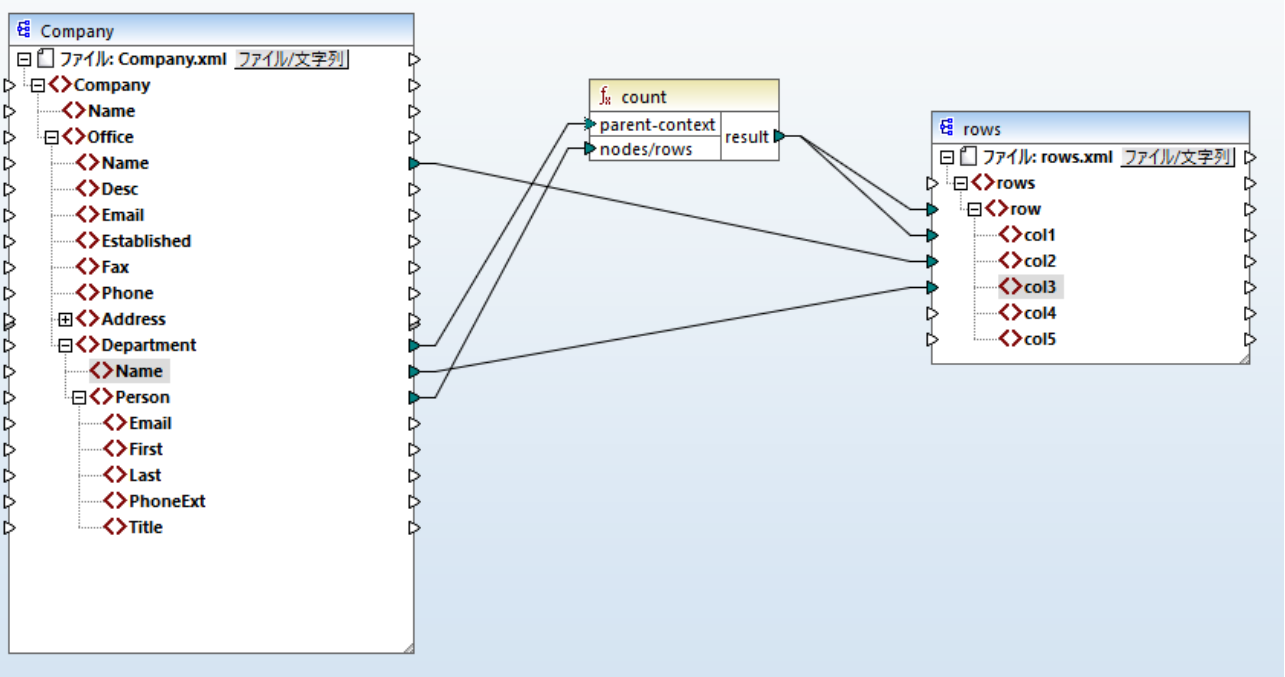


上のマッピングでは、接続線 A はthe `count` 関数の親コンテキストを **Department** に変更します。これにより、関数は各部署内の職員数を数えることができます。重要な点は、ソース内に複数の部署が存在するため関数は単一の結果の代わりに結果の *sequence* シーケンスを返します。このために接続線 B が存在します: 結果シーケンス内の各アイテムのためにターゲットファイル内に新規の行が作成されます。マッピングの出力は上記に従い変更されています（各部署内の職員数に結果が対応していることにご注意してください）:

```
<rows>
    <row>
        <col1>3</col1>
    </row>
    <row>
        <col1>2</col1>
    </row>
    <row>
        <col1>6</col1>
    </row>
    <row>
        <col1>4</col1>
    </row>
    <row>
        <col1>2</col1>
    </row>
    <row>
        <col1>1</col1>
    </row>
    <row>
        <col1>3</col1>
```

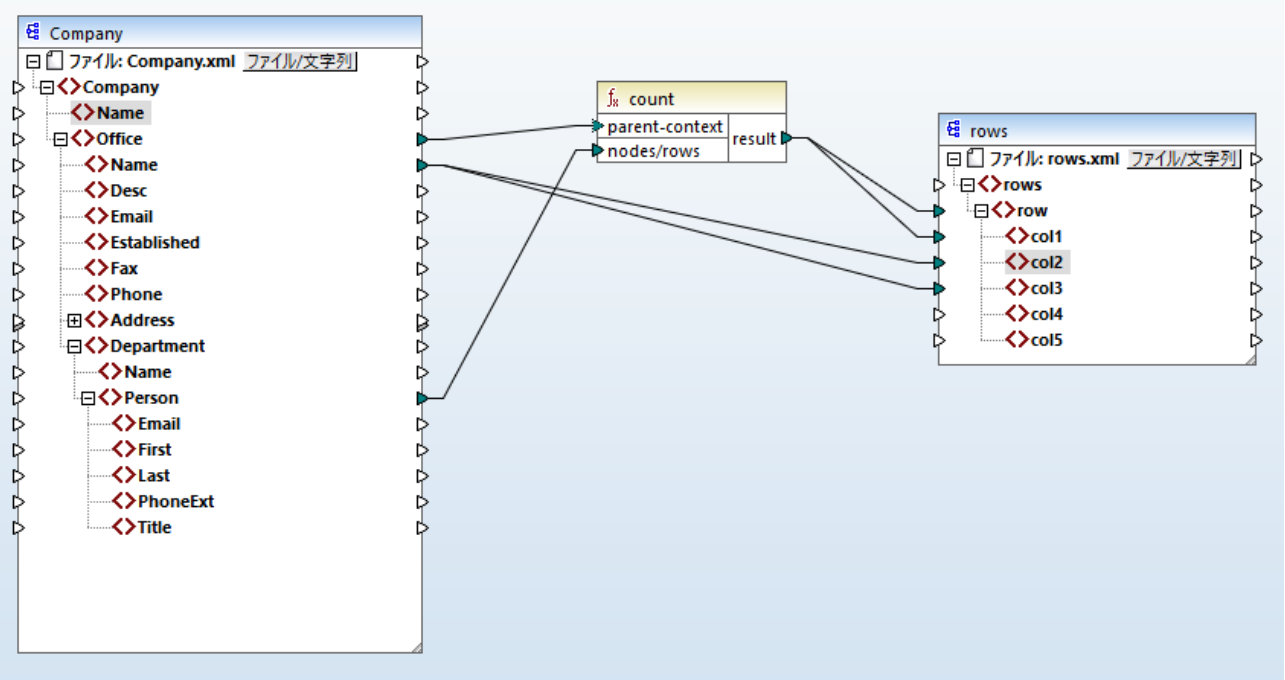
```
    </row>
</rows>
```

現在のマッピングは各部署のために行を作成することから、接続線 C と D を描くことにより、ターゲットファイル内にオフィス名と部署名もコピーすることができます:



このようにして、出力で職員数だけではなく、対応するオフィス名と部署名も表示されます。

各オフィス内の職員数を数えるためには、`count` 関数の親コンテキストをソース内の**Office** アイテムに接続します。



上記の接続では `count` 関数は各オフィスのために1つの結果を返します。ソースファイル内に2つのオフィスが存在するために、関数は2つのシーケンスを返します。この結果、出力内には各行がそのオフィス内の職員数を示す2つの行が存在します:

```
<rows>
    <row>
        <col1>15</col1>
        <col2>Nanonull, Inc.</col2>
    </row>
    <row>
        <col1>6</col1>
        <col2>Nanonull Partners, Inc.</col2>
    </row>
</rows>
```

4.19.3 優先コンテキスト

優先コンテキストは関数の入力パラメーターが評価される順序を影響する方法です。優先コンテキストの設定はマッピングがデータを2つの関連しないソースからジョインする場合必要になる場合があります。

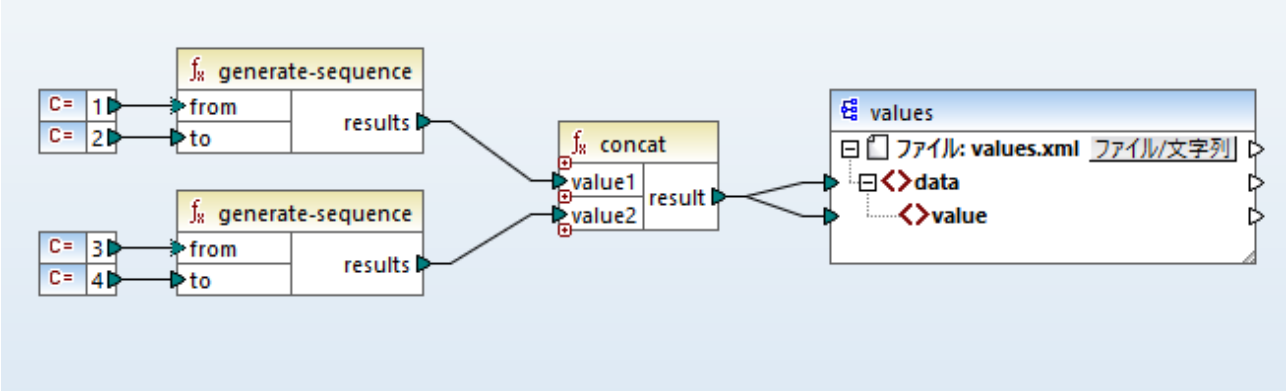
優先コンテキストのしくみを理解するには、マッピングの実行時入力アイテムへの接続が複数の値の *シーケンス* を持つ場合があることを思いだしてください。2つの入力パラメーターを持つ関数の場合、これはMapForce が1つのループが最初に処理される必要がある2つのループを作成する必要があることを意味します。最初に処理されるループは「外側の」ループです。例えば `equal` 関数は2つのパラメーターを受け取ります: *a* と *b*。*a* と *b* の両方が値のシーケンスを受け取る場合 MapForce は以下のように処理します:

- *a* の個別の発生のために
  - *b* の個別の発生のために
    - *a* は *b* と等価か?

上記で明確のようにb は各 a のコンテキスト内で評価されます。優先コンテキストは処理ロジックを変更することを許可するため、各 a は各 b のコンテキストで評価されます。すなわち、内部ループと外部ループを置き換えることが可能になります。例:

- b の個別の発生のために
  - a の個別の発生のために
    - a は b と等価か?

マッピングの結果に影響を与える優先コンテキストを見てみましょう。下のマッピングでは `concat` 関数には2 つの入力パラメーターが存在します。各入力パラメーターは `generate-sequence` 関数の助けを使用して生成されるシーケンスです。最初のシーケンスは「1,2」で2番目のシーケンスは「3,4」です。



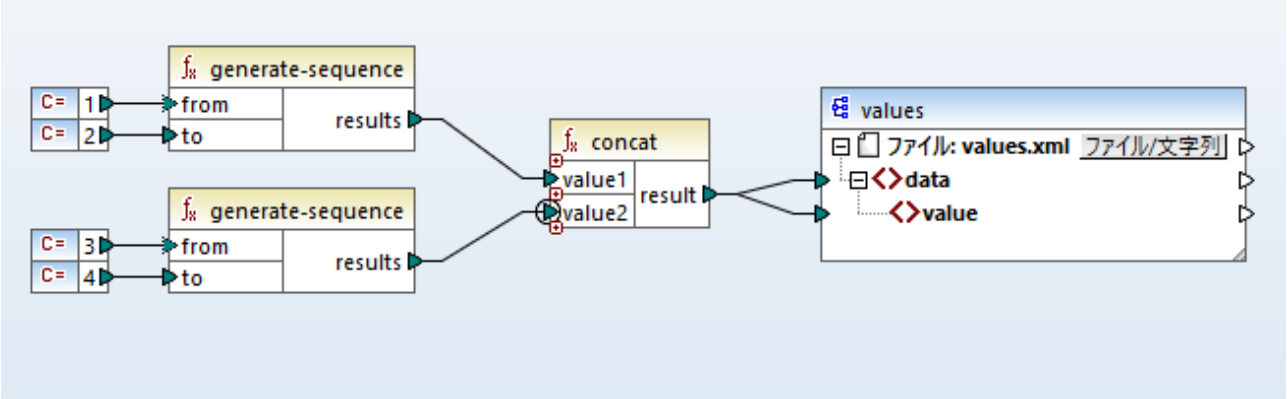
最初に、優先コンテキストを設定せずにマッピングを実行してみましょう。`Concat` 関数はトップのシーケンスの評価を開始し、次の順序で値を連結します:

- 1 と3
- 1 と4
- 2 と3
- 2 と4

これはマッピングにも反映されています:

```
<data>
    <value>13</value>
    <value>14</value>
    <value>23</value>
    <value>24</value>
</data>
```

2番目の入力パラメーターを右クリックし 優先コンテキスト をコンテキストメニューから選択すると、優先コンテキストに設定されます。下に示されているように、優先コンテキスト入力は丸で囲まれています。



今回は2番目の入力パラメーターが最初に評価されます。`concat` 関数は同じ値と連結されていますが、シーケンス `3,4` を最初に処理します。結果的に、出力は以下のとおりになります:

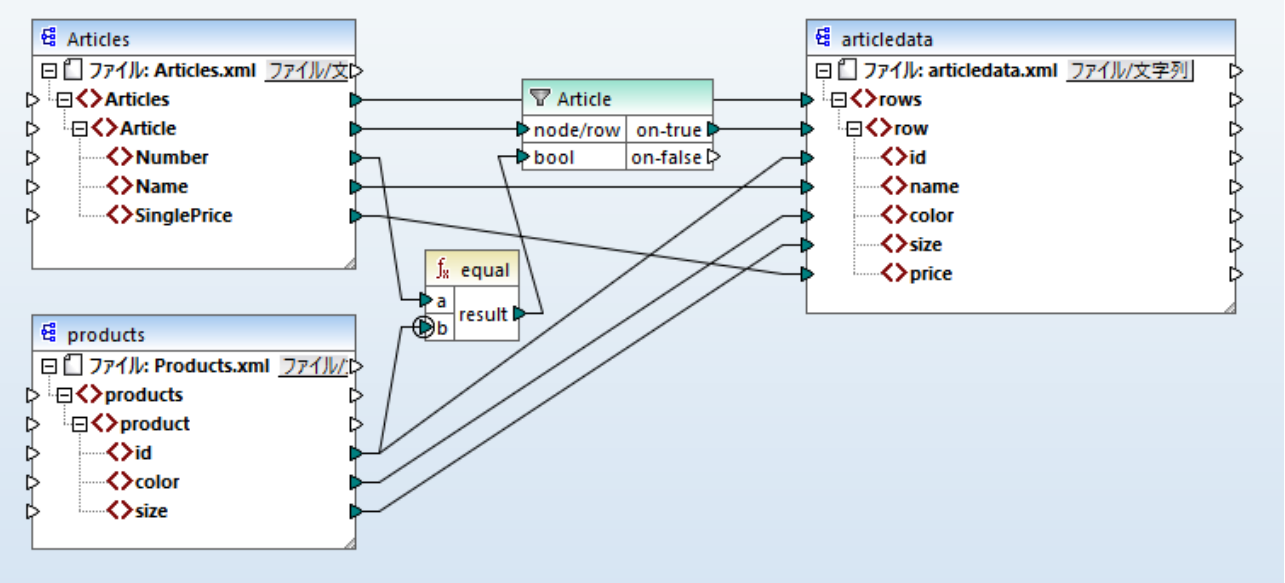
```
<data>
    <value>13</value>
    <value>23</value>
    <value>14</value>
    <value>24</value>
</data>
```

ここまでは、優先コンテキストの理論的な部分のみが紹介されています。実質的なシナリオに関しては[例: 優先コンテキストを使用してフィルターする]を参照してください。

4.19.3.1 例: 優先コンテキストを使用してフィルターする

関数がフィルターに接続されている場合、優先コンテキストは巻数のみではなく、フィルターの評価にも影響を与えます。下のマッピングは正確な出力を取得するために優先コンテキストの設定が必要とされる典型的なケースを説明しています。このマッピングを次のパスで見つけることができます: <マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥Tutorial¥**FilterWithPriority.mfd**。

メモ このマッピングはXML コンポーネントを使用しますが、下に説明されている同じロジックがMapForce 内のEDI, JSON, などを含む他のすべてのコンポーネントに適用されます。データベースに関しては標準のフィルターを使用する代わりに[SQL WHERE]の使用が奨励されます。



このマッピングの目的は**Articles.xml** からデータを 異なるスキーマ**articledata.xml**を持つ新規のXML ファイルにコピーすることです。同時にマッピングは**Products.xml** ファイル内の各アーティクルの詳細をルックアップし、対応するアーティクルのレコードにジョインします。**Articles.xml** 内の各レコードは**Number** を持ち、**Products.xml** 内の各レコードは**id** を持っています。これら2 つの値が等価の場合、他のすべての値（**Name**、**SinglePrice**、**color**、**size**）はターゲット内の同じ **row** にコピーされます。

この目的はフィルターを追加することにより達成することができます。各フィルターはブール条件を入力として必要とします。条件を満たす nodes/row のみがターゲットにコピーされます。この目的のために`equal` 関数がマッピング上に存在します。`equal` 関数はアーティクル番号と製品 ID が両方のソースで等価化をチェックします。結果は入力としてフィルターに提供されます。**true** の場合 **Article** アイテムはターゲットにコピーされます。

2番目の`equal` 関数の2番目の入力パラメーター上で優先コンテキストが定義されていることに注意してください。このマッピングでは、優先コンテキストは大きな違いをもたらし、優先コンテキストを設定しない場合、間違ったマッピングの出力が結果として生成されます。

初期のマッピング: 優先コンテキストなし

優先コンテキストを持たないマッピングロジック:

- 一般的なマッピングルールに従うとフィルターの条件を満たす **Article** と新規の**row** がターゲット内で作成されます。（関数とフィルターを使用して） **Article** と**row** 間の接続線はこの部分を対応します。
- フィルターは各アーティクルのために条件をチェックします。これを処理するには、全ての製品内で反復し、現在のコンテキスト内に複数の製品をもたらします。
- ターゲット側で**id** を作成するには、MapForce は一般的なルールに従います（ソース内の核アイテムのためにターゲット内でアイテムが作成されます）。しかしながら、上記の通り**Products.xml** からのすべての製品が現在のコンテキスト内に存在します。特定の製品の**id** を読み取るための**product** と他の箇所への接続線が存在しません。この結果、複数の**id** 要素はターゲット内の各 **Article** のために作成されます。**color** と**size** にも同様な状況が発生します。

要約: **Products.xml** からのアイテムは各製品を反復する必要がある）フィルターコンテキストを有します。このため、**id**、**color**、および**size** 値は各ターゲット **row** にソースファイル内の製品の数だけコピーされ、以下のように不正確な出力を生成します:

```
<rows>
    <row>
        <id>1</id>
```

```
        <id>2</id>
        <id>3</id>
        <name>T-Shirt</name>
        <color>red</color>
        <color>blue</color>
        <color>green</color>
        <size>10</size>
        <size>20</size>
        <size>30</size>
        <price>25</price>
    </row>
</rows>
```

ソリューション A: 優先コンテキストの使用

上記の問題は関数にフィルターのブール条件を計算する優先コンテキストを追加することで解決することができます。

具体的には、`equal` 関数の2番目の入力パラメーターが優先テキストとして選択されている場合、**Products.xml** からのシーケンスが優先されます。これは次のマッピングロジックに反映されます:

- 各製品のために`equal` 関数の入力 **b** が作成されます（すなわち **b** が優先されます）。この段階では、現在の製品の詳細はコンテキスト内に存在します。
- 各アーティクルのために`equal` 関数の入力 **a** は条件がtrue かをチェックします。条件を満たしている場合、アーティクルの詳細を現在のコンテキストにも配置します。
- 次に、アーティクルと製品の詳細を現在のコンテキストからターゲット内の対応するアイテムにコピーします。

製品のマッピングロジックは正確な出力です。例:

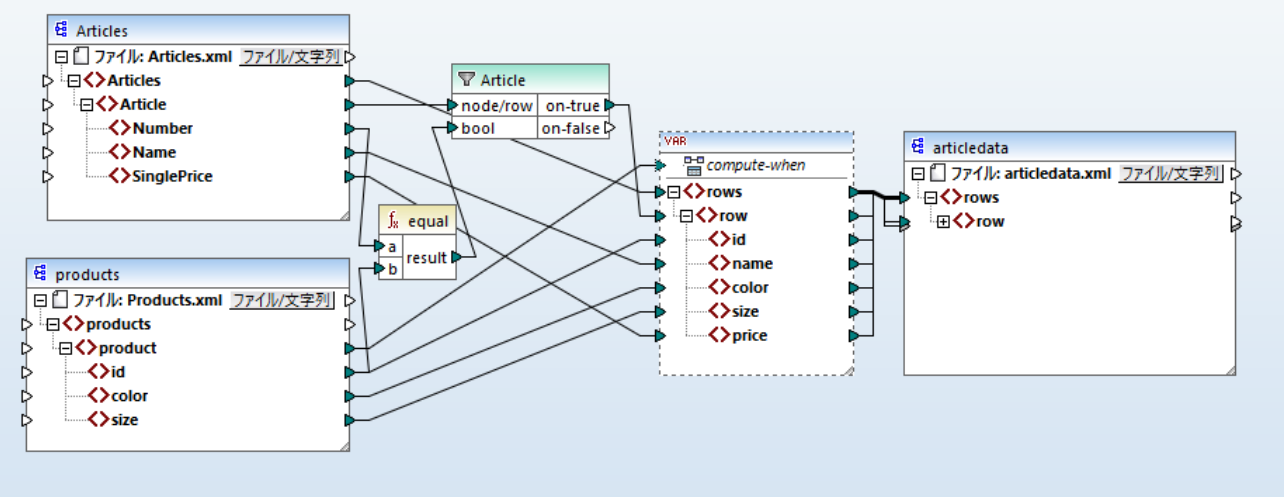
```
<rows>
    <row>
        <id>1</id>
        <name>T-Shirt</name>
        <color>red</color>
        <size>10</size>
        <price>25</price>
    </row>
</rows>
```

ソリューション B: 変数の使用

代替ソリューションとして、中間変数を使用してフィルターの条件を満たす各アーティクルと製品を同じコンテキストに置くことができます。マッピング上にデータを一時的に保管しコンテキストを必要に応じて変更することができるため、変数はこのようなシナリオに適しています。

このようなシナリオのために、マッピングにターゲットコンポーネントと同じスキーマを持つ変数を追加することができます。「**挿入**」メニューで「**変数**」をクリックして **articledata.xsd** スキーマをプロンプトされると構造として提供します。

(C) 2015-2021 Altova GmbH



上記のマッピングでは次が発生します:

- 優先コンテキストはもう使用されません。代わりにターゲットコンポーネントと同じ構造を持つ変数が存在します。
- いつも通り、マッピングの実行はターゲットルートノードから開始されます。ターゲットを作成する前に、マッピングはデータを変数に収集します。
- 各製品のコンテキスト内で変数が計算されます。これが発生する理由は**product** から変数の**compute-when** 入力への接続が存在するからです。
- フィルターの条件はこのため各製品のコンテキスト内でチェックされます。条件がtrue の場合のみ変数の構造は作成され、ターゲットにパスされます。

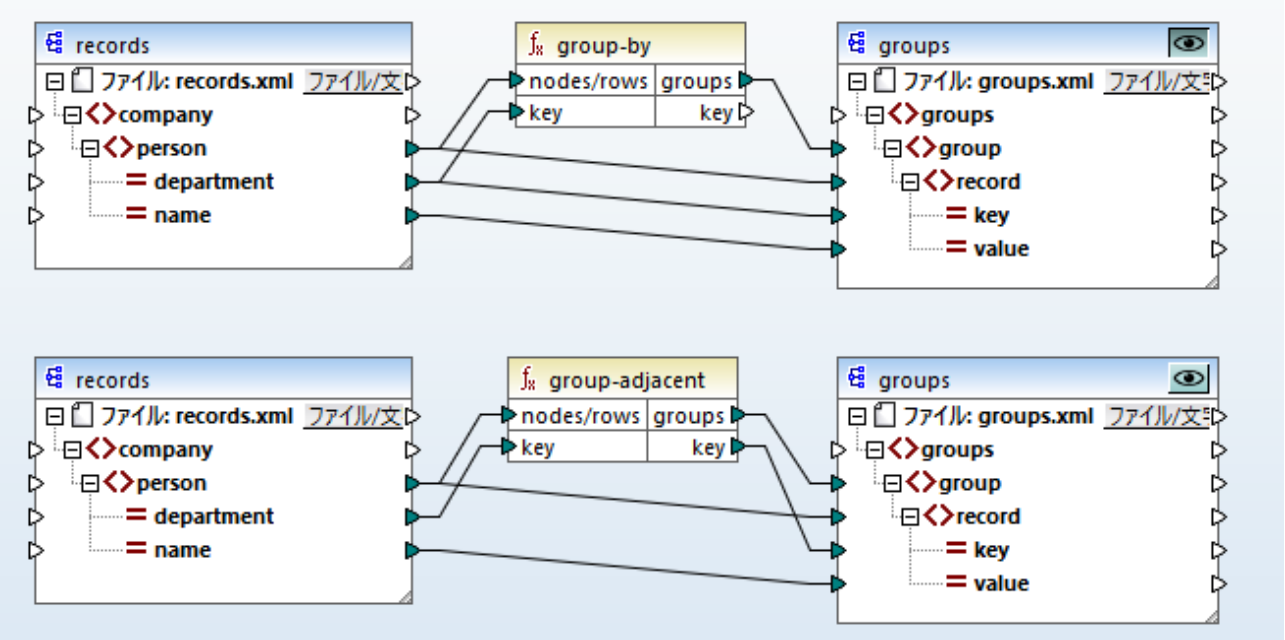
4.19.4　複数のターゲットコンポーネント

マッピングには複数のソースとターゲットコンポーネントが存在する場合があります。複数のターゲットコンポーネントが存在する場合 MapForce 内では一度に1つのコンポーネント出力のみ プレビューボタンをクリックすることによりプレビューすることができます。他の実行環境では(MapForce Server または生成されたコード) 全てのターゲットコンポーネントがシーケンスで実行され、各コンポーネントの対応する出力が生成されます。

デフォルトでは、ターゲットコンポーネントは上から下、左から右に処理されます。必要であれば、マッピングウィンドウ内でターゲットコンポーネントの位置を変更することにより処理順序に影響を与えることができます。レファレンスポイントは各コンポーネントの左角です。以下の点に注意してください:

- 2つのコンポーネントが同じ垂直方向の位置にある場合、左側が優先されます。
- 2つのコンポーネントが同じ水平方向の位置にある場合、上側が優先されます。
- コンポーネントが同じ位置に存在することは少なく、定義された順序を保証するが、変更不可能な一意の内部コンポーネント ID が自動的に使用されます。

この仕組みを説明するサンプルとしては、以下のデモマッピングを開いてください: **<マイドキュメント>\Altova\MapForce2021\MapForceExamples\Tutorial\GroupingFunctions.mfd**。このマッピングは複数のソースと複数のターゲットコンポーネントから構成されています。マッピングの一部のみが表示されています。



ルールに従うと、MapForce Server 内のこのマッピングと生成されたコード内のデフォルトの処理順序は上から下です。この場合 XSLT 2.0 コードを生成してチェックすることができます。

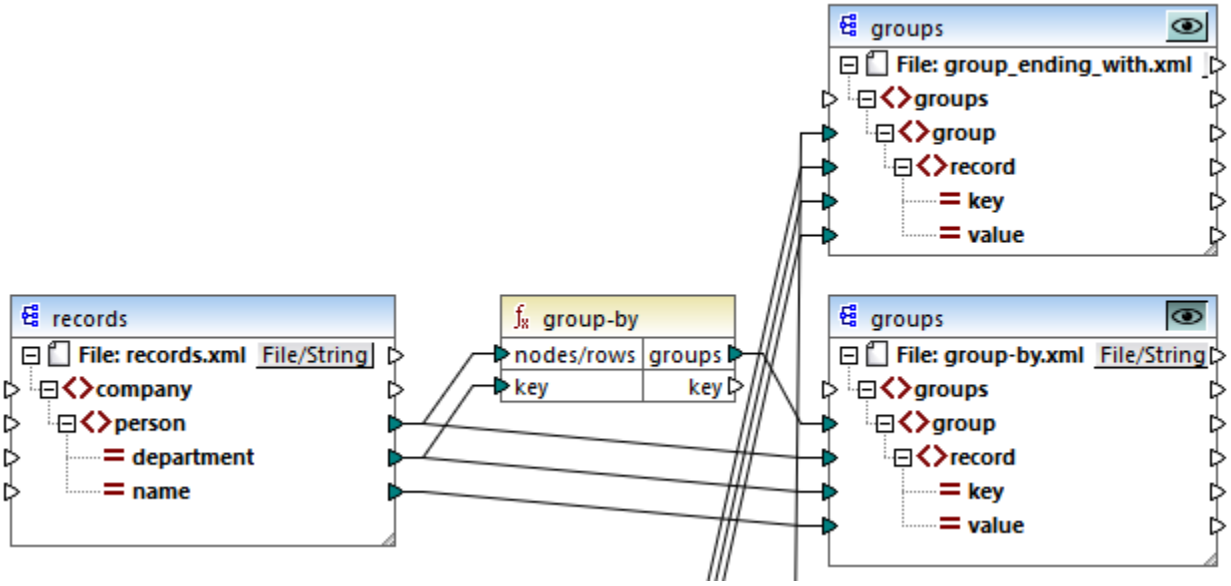
1. 「ファイル」メニューから「コード生成 | XSLT 2.0」をクリックします。
2. プロンプトされると、生成されるコードのためのターゲットディレクトリを選択します。

生成後、ターゲットディレクトリには複数の XSLT ファイルと **DoTransform.bat** ファイルが含まれています。後者は(個別のライセンスを必要とする) RaptorXML Server により実行されることができます。**DoTransform.bat** ファイルはマッピングの定義と同じ順序で上から下にコンポーネントを処理します。これは各変換の `--output` パラメーターを確認することで検証可能です。

```
RaptorXML xslt --xslt-version=2 --input="records.xml" --output="group-by.xml" --xml-
validation-error-as-warning=true %* "MappingMapTogroups.xslt"
IF ERRORLEVEL 1 EXIT/B %ERRORLEVEL%
RaptorXML xslt --xslt-version=2 --input="records.xml" --output="group-adjacent.xml" --
xml-validation-error-as-warning=true %* "MappingMapTogroups2.xslt"
IF ERRORLEVEL 1 EXIT/B %ERRORLEVEL%
RaptorXML xslt --xslt-version=2 --input="records.xml" --output="group-into-blocks.xml"
--xml-validation-error-as-warning=true %* "MappingMapTogroups3.xslt"
IF ERRORLEVEL 1 EXIT/B %ERRORLEVEL%
RaptorXML xslt --xslt-version=2 --input="records-v2.xml" --output="group-starting-
with.xml" --xml-validation-error-as-warning=true %* "MappingMapTogroups4.xslt"
IF ERRORLEVEL 1 EXIT/B %ERRORLEVEL%
RaptorXML xslt --xslt-version=2 --input="records-v3.xml" --
output="group_ending_with.xml" --xml-validation-error-as-warning=true %*
"MappingMapTogroups5.xslt"
IF ERRORLEVEL 1 EXIT/B %ERRORLEVEL%
```

最後の変換は **group-ending-with.xml** と呼ばれる出力ファイルを生成します。このターゲットコンポーネントをマッピングの上の部分に移動します:

(C) 2015-2021 Altova GmbH



XSLT 2.0 コードをもう一度生成すると、処理順序が変更に従い変更されます:

```
RaptorXML xslt --xslt-version=2 --input="records-v3.xml" --
output="group_ending_with.xml" --xml-validation-error-as-warning=true %*
"MappingMapTogroups.xslt"
IF ERRORLEVEL 1 EXIT/B %ERRORLEVEL%
RaptorXML xslt --xslt-version=2 --input="records.xml" --output="group-by.xml" --xml-
validation-error-as-warning=true %* "MappingMapTogroups2.xslt"
IF ERRORLEVEL 1 EXIT/B %ERRORLEVEL%
RaptorXML xslt --xslt-version=2 --input="records.xml" --output="group-adjacent.xml" --
xml-validation-error-as-warning=true %* "MappingMapTogroups3.xslt"
IF ERRORLEVEL 1 EXIT/B %ERRORLEVEL%
RaptorXML xslt --xslt-version=2 --input="records.xml" --output="group-into-blocks.xml"
--xml-validation-error-as-warning=true %* "MappingMapTogroups4.xslt"
IF ERRORLEVEL 1 EXIT/B %ERRORLEVEL%
RaptorXML xslt --xslt-version=2 --input="records-v2.xml" --output="group-starting-
with.xml" --xml-validation-error-as-warning=true %* "MappingMapTogroups5.xslt"
IF ERRORLEVEL 1 EXIT/B %ERRORLEVEL%
```

上のコードリスティングでは、最初の呼び出しは **group-ending-with.xml** を生成します。

他のコード言語と生成された MapForceServer 実行ファイル(.mfx) 内で同様に処理順序を変更することができます。

チェーンされたマッピング

上記と同様の処理シーケンスがチェーンされたマッピングにも適用されますチェーンされたマッピンググループは1つのユニットとしてとられます。単一のチェーンされたマッピングの中間または最終ターゲットコンポーネントの位置変更は処理シーケンスに影響を与えません。マッピング内に存在する複数の「チェーン」または複数のターゲットコンポーネントのみが各グループの最初に処理される最終ターゲットコンポーネントの位置を決定します。

- 最終ターゲットコンポーネントが同じ垂直の位置にある場合、左側が優先されます。
- 最終ターゲットコンポーネントが同じ水平の位置にある場合、上側が優先されます。
- コンポーネントが同じ位置に存在することは少なく、定義された順序を保証するが、変更不可能な一意の内部コンポーネント ID が自動的に使用されます。

5 マッピングのデバッグ

MapForce は、MapForce 内蔵の変換言語のためのマッピングデバッガーを含みます。マッピングデバッガーは、以下の目的を達成する手助けをします:

- マッピングにより生成された値を個別のコネクタレベルで確認して分析します。
- 特定の値を作成するためのコンテキスト（ノードのセット）をマッピングでハイライトします。
- 各値をリアルタイムで処理または計算するかを確認するため、または生成されるマッピング出力をプレビューするため、MapForce をステップバイステップで実行します。
- 目的（ブレークポイント）を、マッピングの実行に設定し、現在処理されている値を停止し、表示します。
- 現在の実行の箇所までのマッピングの実行により処理された値の履歴を確認します。

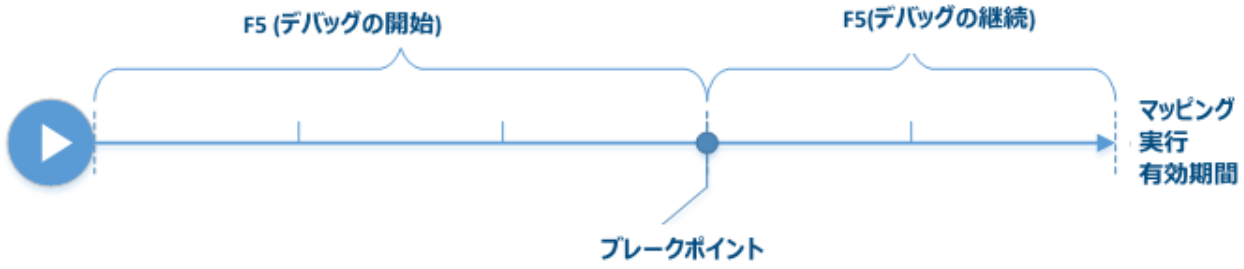
マッピングの変換言語がBUILT-IN の場合、マッピングデバッガーを使用することができます。異なる言語のために デザインされたマッピングのデバッグを開始すると、マッピングの言語を BUILT-IN に変換するようにプロンプトされます。メニューコマンド「**出力 / 内蔵の実行エンジン**」を選択することにより、マッピングを BUILT-IN にすることにより変更することもできます。どちらの場合でも、マッピングがBUILT-IN 言語内(例: XSLT 関数) で使用することのできないコンポーネントを含む場合 BUILT-IN への変換は成功します。

MapForce デバッガーは、(MapForceを使用してコードを書く必要がないため) 従来のデバッガーと異なり、プログラムコードをラインごとにトラバースしません。代わりに、MapForce により生成されたコードの結果は、デバッガーにより表示されます。具体的な方法は、デバッガーは、入力と出力コネクタを介して パスされた値をログします。ログされた値は、マッピング専用のウィンドウを介して、直接分析のために使用することができます。

以下のセクションは、マッピングデバッガーを使用する複数の方法をハイライトします。

ブレークポイントを使用するデバッグ

マッピングの特定の箇所で、デバッグの実行を停止する必要がある場合、従来の開発環境で行われるように、ブレークポイントを設定します。相違点は、ブレークポイントはコードのラインに追加されるのではなく、マッピングコンポーネントの入力または出力コネクタに追加されます。ブレークポイントの条件も追加することができます（これは、設定された条件が満たされると実行を停止する場合役に立ちます）。

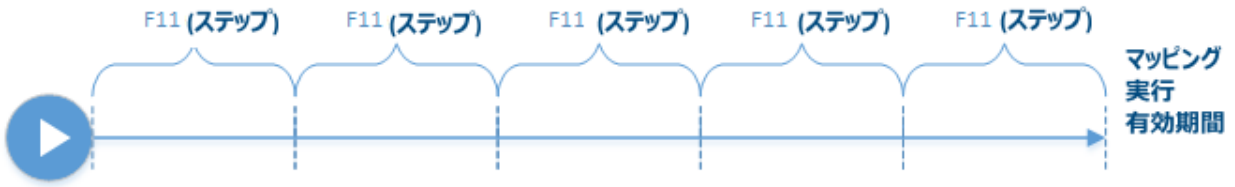


ブレークポイントを希望するコネクタ上で定義し、最初のブレークポイントに到達するまでマッピングを実行し、また次のブレークポイントまでマッピングを継続することができます。このようにして、選択されたコネクタと関連したマッピングコンテキストと値を分析することができます。デバッガーにより与えられた「**ステップ イン**」、「**ステップアウト**」、「**ステップオーバー**」、および「**最小ステップ**」コマンドを使用して実行のスピードを遅くしたり速くしたりすることができます。これらのコマンドによりマッピングの一部をスキップすることができ、またその反対に、必要であればマッピングの一部を更に詳しく実行することもできます。

ステップバイステップのデバッグ

デバッグをステップバイステップで行い、マッピングコンテキストとそれぞれのステップに関連した値を分析することができます。このシナリオは前のものと類似しており、「**ステップイン**」、「**ステップアウト**」、「**ステップオーバー**」および「**最小ステップ**」コマンド.を使用して実行のスピードを速くまたは遅くすることができます。

(C) 2015-2021 Altova GmbH

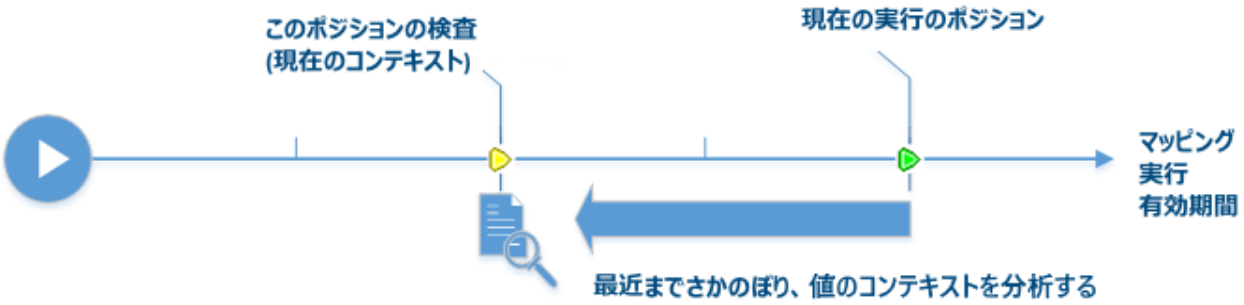


値のログを分析する

MapForce にマッピングのデバッグ中に全てのコネクタにより処理された全ての値を記憶(履歴のトレース)するように構成することができます。履歴のトレースを保持することは、データを中心としたマッピングには、向いていないかもしれません。ですから、このオプションは、必要であれば無効化することができます。オプションが有効化されていると、各コネクタにより処理された値の現在の実行のポジションまでの、完全なログを分析することができます。値の作成の理由を理解することのできる、MapForce に特定の値に関連したマッピングを再作成することもできます。

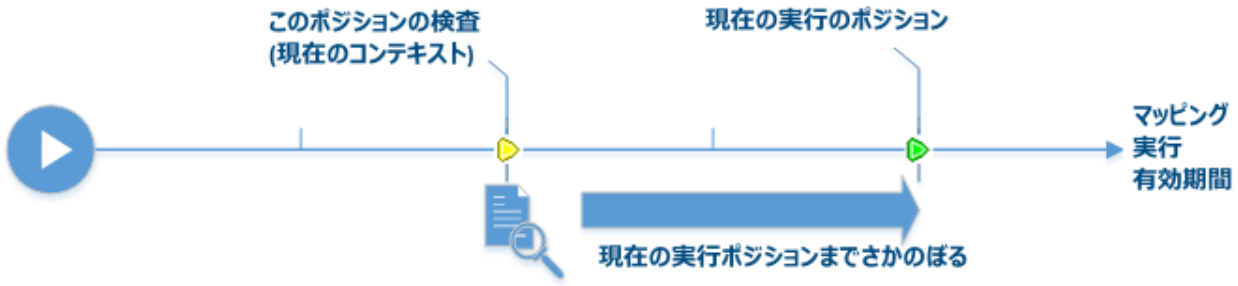
現在の実行の箇所に関連した値のコンテキストを設定する

デバッガーがマッピングで特定の実行の位置にある場合、現在の実行の箇所に対応した過去の値のコンテキストを分析することができます。(これは、少し前の時間に戻ることにも比較することができます):



コンテキストはなぜ値が計算されるかを説明します。すなわち、生成されるマッピングの特定の値について説明します。コンテキストとは、MapForce により近い過去のコンテキストとして設定することもできますが、通常現在の実行の箇所です。コンテキストが特定の値に設定されている場合、MapForce は関連するノードをマッピングで直接ハイライトし、マッピングコネクタの横にヒントを表示し、デバッガーに関連したウィンドウ内(**値**、**コンテキスト** と**ブレークポイント** ウィンドウ) で追加情報を表示します。

現在の実行の箇所と同じではない、マッピングのコンテキストを確認した後、コンテキストを現在の場所に再設定することができます:



制限

- 現在、マッピングデバッガーは、Visual Studio のためのMapForce プラグイン またはEclipseではサポートされていません。
- MapForce がマッピングを実行する際、内部でコードを最適化します（例: データをキャッチ、または任意の箇所で中間結果を計算する場合）。これにより、特定のコネクタ（つまりブレークポイント）はデバッグで到達できない場合があります、この場合、MapForce は通知を表示します。MapForce コード最適化は、マッピングに対するマッピング出力（をしてこの結果、デバッガーの振る舞い）は同じ場合でも、MapForce のリリースにより異なる場合があります。
- デバッガーは1つのターゲットコンポーネントのための出力をデバッグすることができます。マッピング上に複数のターゲットコンポーネントが存在する場合、デバッガーにより実行されるコンポーネントを選択する必要があります。
- 現在、データベーステーブルアクションデータベースコンポーネントの（"全て挿入" や "If の場合更新"など）では、バッギングはサポートされません。
- ブレークポイントは次のエンティティに追加することはできません: 定数、`core | position` 関数、「全てコピー」接続の子孫アイテム、"インライン" ユーザー定義関数のパラメーター。

(C) 2015-2021 Altova GmbH

5.1 デバッガーの準備

デバッグの準備は、大きなシステムメモリの実行を必要とする通常大きなデータマッピングに対して必要とされます。大きな入力または出力ファイルを処理するマッピングの場合、または、大きなデータのコレクションを繰り返す場合などが当てはまります。

デバッグのスピードを向上し、メモリの使用量を減らすために、デバッグを開始する前に以下を行うことが奨励されます:

- マッピングが複合型の場合、デバッグが必要のないマッピングの一部の接続を削除または解除してください。
- マッピング大きな入力ファイルを使用する場合、小さなサイズのファイルと置き換えてください。
- 「**フルトレース履歴を保持**」が無効化されていることを確認してください（デバッガー設定を参照）。

正確な出力をデバッグするために、以下を適用することができるか確認してください:

- マッピングに複数のターゲットコンポーネントがある場合、ターゲットコンポーネントを選択して、「**プレビュー**」ボタン（ ）をクリックします。
- マッピングがチェーンマッピング（チェーンマッピングを参照）の場合、中間コンポーネントの「**パススルー**」（ ）ボタンをリリースします。「パススルー」コンポーネントのデバッグは現在サポートされていません。

オプションで、デバッガーが重要なコネクタの一部を停止する場合、値を分析するこれらのコネクタへブレークポイントを追加します（ブレークポイントの追加と削除を参照）。

5.2 デバッガーコマンド

次のデバッガーコマンドにアクセスすることができます:

- 「デバッグ」メニュー内
- キーボードのショートカット
- デバッグツールバー内

メニューコマンド	キーボードショートカット	ツールバーボタン	説明
デバッグ	デバッグの開始	F5		デバッグを開始または、ブレークポイントに到達またはマッピングが完了するまで継続します。
デバッグ	デバッグの中止	Shift + F5		デバッグを停止します。このコマンドはデバッグモードを終了して、MapForce を標準モードに切り替えます。
デバッグ	ステップイン	F11		マッピング内で単一のステップを終了するまでマッピングを実行します。マッピングでデバッガーは、シーケンスの単一アイテムを生成する独立した計算の理論的なグループです。 マッピングコンテキストにより、このコマンドは「左へ移動/ターゲットに移動 子/ソースの親へ移動」と解釈することができます。
デバッグ	ステップオーバー	F10		現在のステップが終了するまでまたは他のシーケンスのアイテムを終了するまで実行を継続します。このコマンドは、現在のステップの入力である計算をステップオーバーします。
デバッグ	ステップアウト	Shift + F11		現在のステップの結果が使用されるまで、または入力または使用の子ではないステップが実行されるまで実行を継続します。このコマンドは現在の計算をステップアウトします。 マッピングコンテキストにより、このコマンドは「右へ移動/ターゲットへ移動 親/ソースの子へ移動」と解釈することができます。
デバッグ	最小ステップ	Ctrl + F11		値が生成または使用されるまで、実行を継続します。このコマンドは、ステップを細分化し、接続ごとに通常2回停止します: ソースが値を生成した際、ターゲットが値を使用した際。MapForce は、マッピングの提示に従い値を計算する必要はなく、生成と使用イベントは順序に従わない場合があります。

デバッグツールバー

(C) 2015-2021 Altova GmbH

5.3 デバッグモードについて

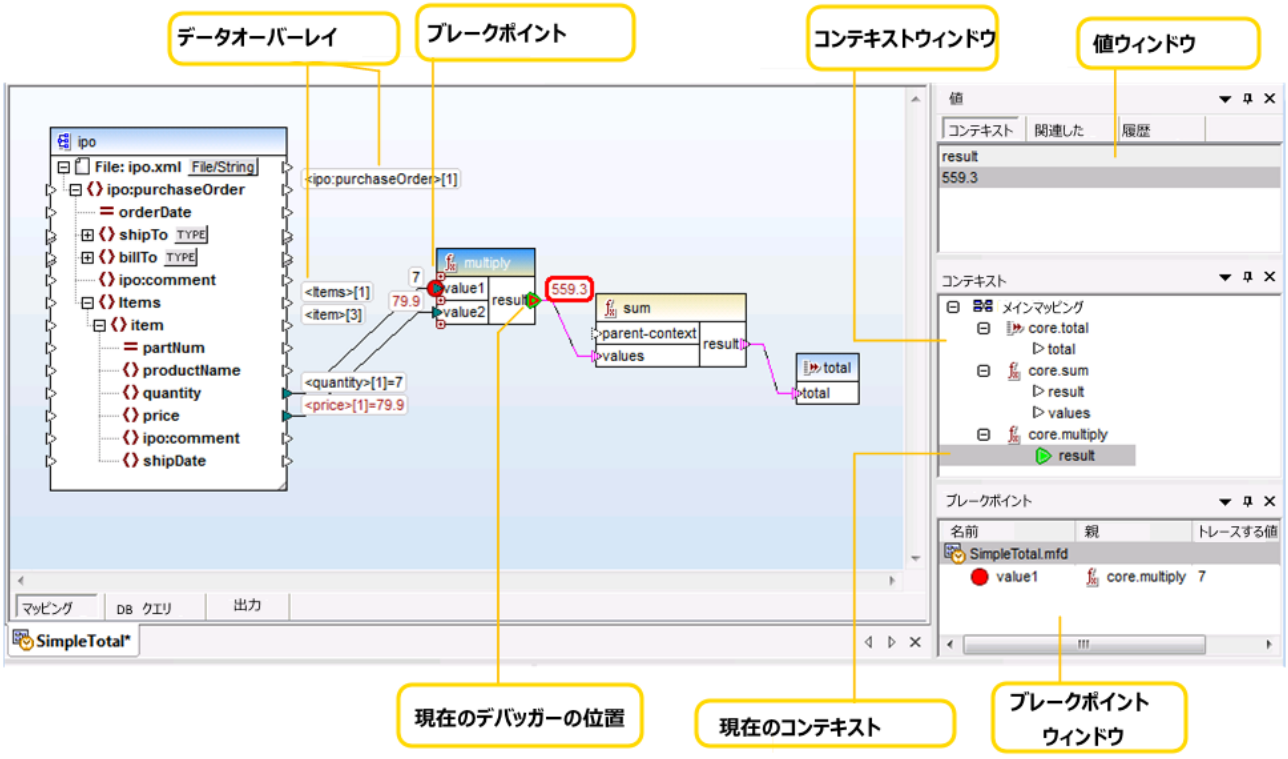
デバッグを開始すると、MapForce は、(「F5」または「F11」、または「Ctrl + F11」を押して)マッピングをデバッグモードで実行します。

> MapForce がデバッグモードの場合、マッピングは読み取り専用です。マッピングエリアでコンポーネントを移動させることはできますが、コマンドの多くを使用することはできません。これは、マッピングの検証、デプロイ、コードの生成、マッピングのドキュメント、マッピングエリアに新規コンポーネントを追加する、または既存のコンポーネントを再ロードするなどが含まれます。

デバッグモードにより特定の値を生成するコンテキストを分析することができます。この情報は、マッピング上および、値、コンテキスト、とブレークポイント ウィンドウで直接使用することができます。デフォルトでは、これらのウィンドウは、デバッグを開始すると表示され、デバッグを中止すると隠されます。

MapForce は、「Shift + F5」を使用して、(または「**デバッグの中止**」 ツールバーボタンをクリックすることにより、)デバッグが中止されるまで、デバッグモードにあります(マッピングは読み取り専用です)。

以下のイメージは段階的にデバッグされるサンプルのマッピングを表しています(<**マイドキュメント>\Altova\MapForce2021\MapForceExamples** ディレクトリからの**SimpleTotal.mfd**)(「F11」を押すと、先に進みます)。



デバッグモード中のMapForce 開発環境

デバッグモード中にMapForce により提供される他の情報と視覚的ヒントは以下で説明されています。

マッピングペイン

デバッグ中、マッピングペインは追加情報を表示します:

- データオーバーレイ（下を参照してください）は現在の値と関連した値をコネクタの近くに表示しています。
- （コンテキストウィンドウ内で構造と表示されている）現在のコンテキストは以下のようにハイライトされています:
  - コンテキスト内のコネクタ 緑のマゼンタで表示されています（ ）。
  - 曖昧なコンテキスト内のコネクタは点線のマゼンタで表示されています（ ）。
- 現在の実行の場所は、緑のコネクタアイコン（ ）で表示されています。

データオーバーレイ

各コネクタにより処理された値は、対応するコネクタの近くに（小さな三角形の）データオーバーレイとして表示されています。最近選択されたオーバーレイは赤い枠で表示されます。前回のステップから変更された値は、暗い赤で表示されています。単純なコンテンツを持つノードには、データオーバーレイは以下の2つの値を結合します －ノード名と値。ノード名が現在の実行ポジションの前に複数回反復されると、現在の反復のインデックスは角かっこ内の数値で表されます。

データオーバーレイは以下の振る舞いを行います:

- データオーバーレイにマウスをポイントすると、一時的に前景に常時されます。クリックすると常時前景に表示されます。また、クリックすると対応するコネクタを選択します。
- データオーバーレイはドラッグして移動することができます。
- データオーバーレイはコンポーネントが移動されると、移動します。ですから、データオーバーレイが重なって表示される場合はコンポーネント同士が近すぎるためであり、コンポーネントをマッピングエリア内で移動してスペースを与えてください。データオーバーレイもコンポーネントと共に移動します。
- データオーバーレイをクリックすると、値ウィンドウ内に値が表示されます。
- コネクタをクリックすると、データオーバーレイを選択します。

ブレークポイント

ブレークポイントは、デバッグモード中にマッピングの実行を中断するマイルストーンを指します。この用語は、他の統合された開発環境などで使用されているため、既知かもしれません。コードのラインにブレークポイントを追加する他の開発環境とは異なり、MapForce は、入力または出力コネクタに（接続の左または右側に小さな三角形で表示される）ブレークポイントを追加することができます。マッピングペインでは、ブレークポイントは赤いサークルで表示されています。定義されたブレークポイントはブレークポイントウィンドウ内に表示されます。ブレークポイントの追加と削除を参照してください。

現在のデバッガーのポジション

緑の三角形（ ）はデバッガーを表しています。このポジションはコンポーネントの入力または出力コネクタです。

現在処理されている値は、「**コンテキスト**」タブの「値」ウィンドウでも表示されます。

接続 および/または コネクタのセットは、緑のマゼンタで現在のマッピングコンテキストを表します。同じ情報が「コンテキスト」ウィンドウ内で階層構造として表示されています（コンテキストウィンドウの使用 を参照）。

値のコンテキストを手動で設定する場合、現在のデバッガーのポジションは最近の実行ポジションに相対的な過去にあるポジションです。最近のポジションと過去のポジションの違いを明確にするために、現在のポジションのコネクタは、デバッガーインターフェイス内で以下の色別に表示されています。

	緑は"現在"を意味し、現在の実行のポジションを示します（コネクタの現在の値を表示するを参照してください）。
	黄色は"過去"を意味し、現在の実行のポジションに相対して、過去のコネクタを観察していることを示します。これは、コンテキストを手動で設定した後に発生する場合があります（ コンテキストを値に設定するを参照してください）。

(C) 2015-2021 Altova GmbH

値 ウィンドウ

値ウィンドウは、マッピングにより処理された値についての情報を提供します。現在の実行のポジション、またはユーザーが設定した特定のコンテキストでどのマッピング処理されたかについての情報が提供されます。値ウィンドウの使用を参照してください。

コンテキスト ウィンドウ

コンテキスト ウィンドウは、現在のデバッガーのポジションに関連したノードのセットと関数の階層的なビューを提供します。 コンテキスト ウィンドウの使用を参照してください。

ブレークポイント ウィンドウ

ブレークポイントは、MapForce が開始されてから作成されたデバッグ ブレークポイントのリストを表示します。複数のマッピングでブレークポイントを定義した場合、ブレークポイント ウィンドウ内に全てのブレークポイントが表示されます。ブレークポイント ウィンドウの使用を参照してください。

5.4 ブレークポイントの追加と削除

ブレークポイントは、デバッグモード中の実行でどのマッピングをブレークするかを指定するマイルストーンです。作成されるブレークポイントは、すべてのマッピングのためにグローバルに保管され、ブレークポイントウィンドウで表示されます。ブレークポイントは明示的に削除されるか、MapForce が閉じられるまで有効です。

メモ ブレークポイントは以下のエンティティには追加することはできません: 定数、`core | position` 関数、「全てコピー」接続の子孫アイテム、"インライン" ユーザー定義関数のパラメーター。

ブレークポイントは簡単または条件付きで作成することができます。簡単なブレークポイントはマッピングの実行を条件無しに停止します。条件付ブレークポインはマッピングの実行を、割り当てられた条件が満たされた場合のみ停止します。条件はカスタム化された値を提供する、MapForce ビルドインライブラリ関数の書式を取ります。すなわち、条件が true を返す場合、ブレークポイントはマッピングの実行を停止します。

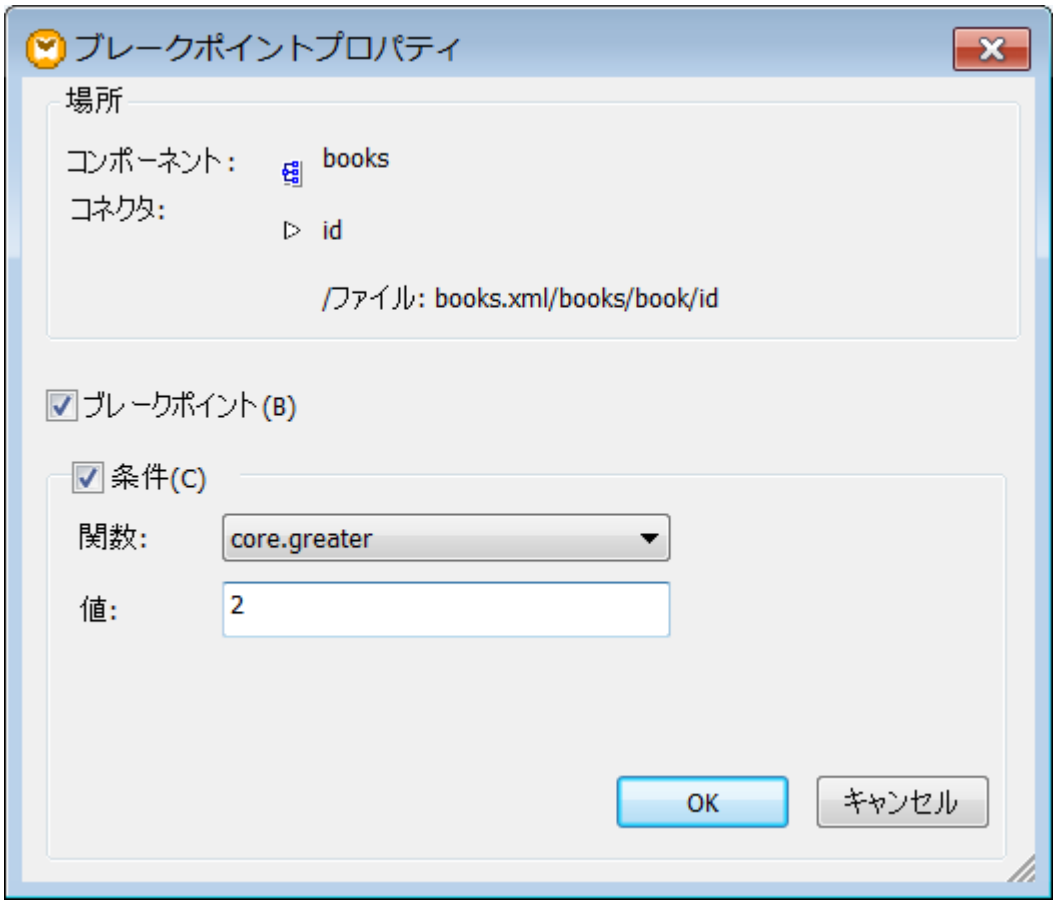
ブレークポイントを作成するには、以下を行います:

- 入力または出力コネクタ(コンポーネントの右または左横にある小さな三角形)を右クリックして、「**デバッガー ブレークポイント**」を選択します。
- 入力または出力コネクタをクリックして、「**F9**」を押します。

条件付ブレークポイントの作成:

1. コネクタを右クリックして、「**ブレークポイント プロパティ**」を選択します。

(C) 2015-2021 Altova GmbH



2. 両方の「ブレークポイント」を選択するために「条件」チェックボックスをクリックします。
3. リストから必要な関数を選択し、関数の値を、適用できる場合は、入力します。例: 上のサンプルでは、ブレークポイントは、通過する値が2より大きい場合、マッピングの実行を停止します。

> 条件ブレークポイントを追加するコネクタのデータ型が、関数により期待される型と一致しない場合、MapForce は、データ型を自動的に変換しようと試みます。自動変換が無理な場合は、マッピングの実行は、失敗します。これを回避するために、互換性のあるデータの型を使用するようにしてください。例: 関数 `core.starts-with` は、文字列の値を期待します、ですからブレークポイントのコネクタは同じ型を有する必要があります。

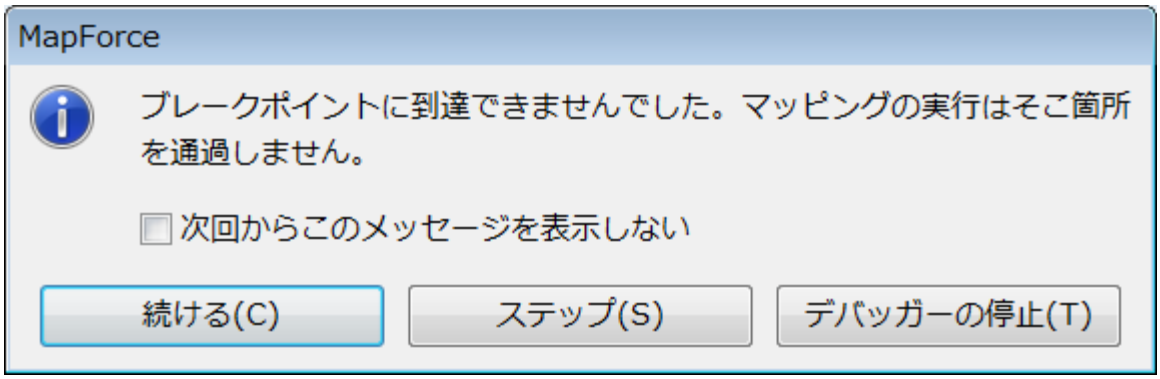
ブレークポイントの削除

ブレークポイントを削除するには、ブレークポイントが存在するコネクタを右クリックして、「デバッガーブレークポイント」を選択します。または、ブレークポイントが存在する入力または出力コネクタをクリックし、「F9」を押します。

ブレークポイントウィンドウからブレークポイントを削除することもできます（ブレークポイントウィンドウの使用 を参照）

到達することのできないブレークポイント

MapForce は、"ブレークポイントに到達することができません" メッセージを表示する場合があります：



これは、以下の理由のためブレークポイントにデバッガーが到達できないことを意味します:

- マッピングの一部ではないコネクタでブレークポイントが定義されている場合。
- 実行の最適化のためMapForce がブレークポイントに到達できない場合(制限を参照)。

次の定義されたブレークポイントに到達するために、**続ける** をクリックします。次に定義されたブレークポイントに移動する場合、デバッグの実行の終わりに移動する場合、デバッグをステップで実行する場合、**ステップ** をクリックします。

デバッガーが直面する 到達することのできないブレークポイントに関する通知を無効化することが、**このメッセージを再度表示しない** をクリックすることにより設定できます。手順は以下のとおりです:

1. 「**ツール**」メニューから「**オプション**」をクリックします。
2. 「**メッセージ**」をクリックします。
3. 「**到達することのできないブレークポイントに関する情報**」チェックボックスのチェックを解除するためにチェックボックスをクリックします。

(C) 2015-2021 Altova GmbH

5.5 値ウィンドウの使用

値ウィンドウは、デバックモード中にマッピングにより処理される値についての情報を表示します。値ウィンドウに表示される情報は、現在のデバッガーのポジション、およびクリックするユーザーインターフェイスにより異なります。値ウィンドウには、以下のタブが含まれます:

"コンテキスト"タブ

「**コンテキスト**」タブは、現在処理されている値を表示します。(コンテキストウィンドウで同じ値のコンテキストが表示されています)。これは、デバッガーの現在の実行のポジションの値、または過去に処理されたコネクタの値を表示します。MapForce はこれらの2つの違いを異なる色を使用して区別します:

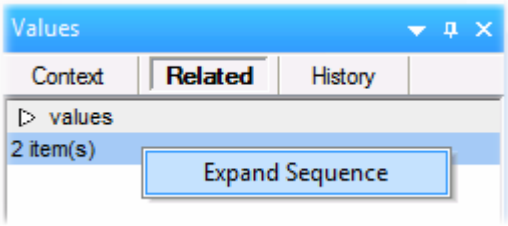
	緑は"現在"を意味し、現在の実行のポジションを示します(コネクタの現在の値を表示するを参照してください)。
	黄色は"過去"を意味し、現在の実行のポジションに相対して、過去のコネクタを観察していることを示します。これは、コンテキストを手動で設定した後に発生する場合があります(コンテキストを値に設定するを参照してください)。

"関連した"タブ

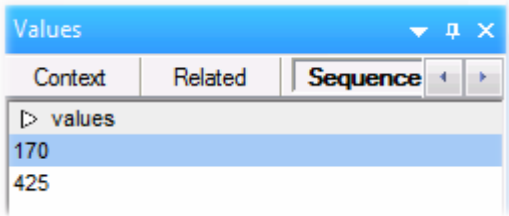
関連したタブは、関連した値、または現在処理されている(または、"近い過去"の値)を表示します。通常、このタブを明示的にクリックする必要があります。MapForce は自動的にデバッガーの現在の実行のポジションに関連したコネクタのデータオーバーレイをクリックすると切り替えます。最近の過去にステップバックを参照してください。

"シーケンス"タブ

存在する場合、**シーケンス**タブは、シーケンスを処理するコネクタの値にアクセスすることを有効化します。このタブはコネクタがアイテムのシーケンスを処理した場合のみ表示されます(例: `sum` または `count` などの関数)。 アイテムのシーケンスを処理したコネクタのデータオーバーレイをクリックすると、値ウィンドウは、フォーマットで"n アイテム"エントリを、n がコネクタにより処理されたアイテム数である箇所で表示します。それぞれの値にアクセスするには、このエントリをダブルクリックします。(または右クリックして、コンテキストメニューから「**シーケンスの展開**」を選択します)。



値はシーケンスタブ内に表示されます。



履歴 タブ

「履歴」タブは、デバッグが開始してから現在の実行のポジションまでに処理された値を表示します。 コネクタにより処理された値の履歴を確認するを参照してください。

(C) 2015-2021 Altova GmbH

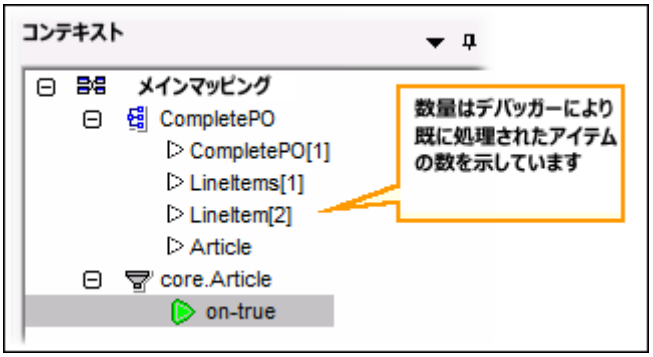
5.6 コンテキストウィンドウの使用

MapForce がデバッグモードの場合、コンテキストウィンドウは、デバッガーの現在のポジションに対して関連のあるコネクタの構造を表示します。すなわち、現在のマッピングの値を作成するマッピングコンテキストを与えます。

MapForce は、現在のコンテキストを以下の通り作成します:

1. ターゲット構造のルートノードから開始します。
2. 現在のターゲットノードに降順します。
3. 現在のターゲットノードから、マッピング内で現在のポジションにリードするコンポーネントを左側に移動します。これらのコンポーネントは、コンポーネント、ビルドインまたはユーザー関数、変数などをフィルターおよび並べ替える可能性があります。

コンテキストウィンドウは、情報およびナビゲートを助ける役割を果たします。マッピング内の現在のコンテキストから直接 マッピング内で特定のノードを選択するには、コンテキストウィンドウ内のノードを右クリックして、「マッピングの選択」をクリックします。これはマッピングが大きいとスクロールを減らすことにより役に立ちます。
コンテキストウィンドウ以下の特別なアイコンと表記を表示する場合があります:

アイコン	説明
	コンテキストが属するマッピングを示します。これはメインマッピングまたはユーザー定義関数のマッピングであることができます。
▷	コネクタを表します。ターゲットノードは角かっこで表示されています。 
	現在のコネクタ（最近の実行ポジション）を表しています。これは値ウィンドウ内の現在の値のソースです。

アイコン	説明
	稀に、計算された値が複数のコネクタのために使用される場合があります。この場合、複数の緑色のアイコン が表示される可能性があります。
	デバッガーが最近の実行ポストに相対的な過去に存在する場合を表示します。これは値にコンテキストを設定すると生じる可能性があります（ コンテキストを値に設定するを参照）。

上のアイコンに加え、コンテキストウィンドウは、マッピング内に存在するコンポーネントの標準アイコンを含みます。

コンテキストウィンドウとユーザー定義関数

現在のコンテキストがユーザー定義関数を含む場合、これらはコンテキストウィンドウにも表示されます。現在のコンテキストがユーザー定義関数の入力の値を計算するための場合、コンテキストは以下のように決定されます:

1. ターゲットからユーザー定義関数の出力コネクタ、ユーザー定義関数の入力コネクタへ。
2. 更に左側に。

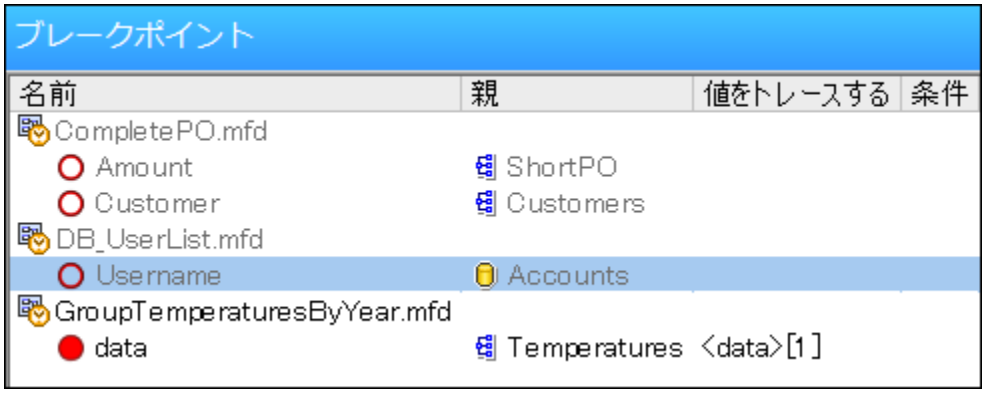
メモ ユーザー定義関数は、コンテキスト内で複数回生じる可能性があります。これは、関数の呼び出しの一部がチェーンされている、またはユーザー定義関数が再帰的と定義されているためです。

(C) 2015-2021 Altova GmbH

5.7 ブレークポイントウィンドウの使用

ブレークポイントを使用することによりブレークポイントをグローバル的に表示し管理することができます。デフォルトでは、 MapForce がデバッグモードの場合、ブレークポイントウィンドウは表示されます。ブレークポイントウィンドウを常に表示するには、「**表示 | デバッグ ウィンドウ | ブレークポイント**」を選択します。

ブレークポイントウィンドウは、 MapForce を開始してから作成された全てのブレークポイントをマッピングファイルごとにグループ化し表示します。MapForce が開かれていると、マッピングに関連する全てのブレークポイントは MapForce により記憶されており、ブレークポイントウィンドウで表示されます。現在デバッグされているマッピングは標準テキストの色でブレークポイントウィンドウ内で表示され、他のマッピング(閉じられているまたは非アクティブなマッピング)は灰色で表示されています。



ブレークポイントウィンドウ内の全てのマッピング(またはその中のブレークポイント)をダブルクリックすることで素早く開くことができます。

メモ　　MapForce を閉じる、または再起動すると、すべてのブレークポイントは削除されます。

ブレークポイントに関する情報は下の表にまとめられています:

カラム	説明
名前	ブレークポイントが属するノードの名前
親	ブレークポイントが属するマッピングコンポーネントの名前
トレースする値	ブレークポイントで接続をパススルーする値。トレースする値は、値デバッグの実行中に表示されます。
条件	ブレークポイントに条件がある場合、この列はブレークポイントの条件を表示します。

ブレークポイント次のアイコンに関連している可能性があります。

アイコン	説明
	アクティブなブレークポイント。現在デバッグされているブレークポイントをマッピングから表します。
	非アクティブなブレークポイント。現在デバッグされていない、開かれているブレークポイントをマッピングから表します。
	到達することのできないブレークポイント。デバッガーにより到達することのできないブレークポイントを表します。
	条件付のブレークポイント。条件付のブレークポイントを表します。

ブレークポイントのプロパティを表示または変更する方法:

- 右クリックして、コンテキストメニューから、「**ブレークポイントプロパティ**」を選択します

ブレークポイントを削除する方法。:

- 削除するブレークポイントを右クリックして、コンテキストメニューから「**ブレークポイントの削除**」を選択します。
- ブレークポイントをクリックして、「**削除**」を押します。

コンテキストコマンド「**全てのブレークポイントを削除**」は、マッピング内の存在箇所に関わらずブレークポイントウィンドウ内に表示される全てのブレークポイントを削除します。

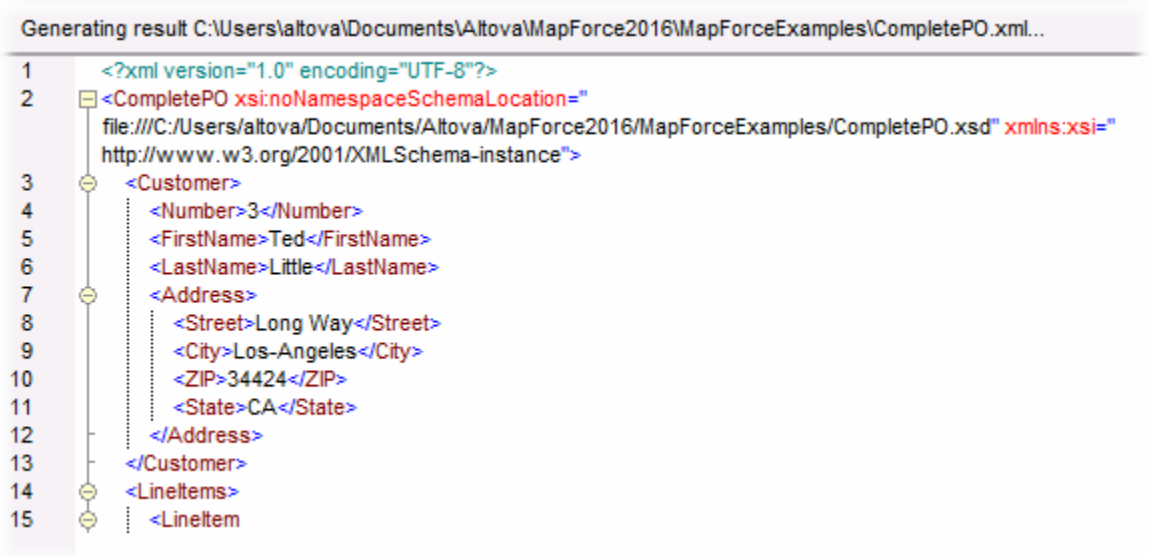
以下を参照してください: ブレークポイントの追加と削除

(C) 2015-2021 Altova GmbH

5.8 部分的に生成された出力のプレビュー

段階的に、またはブレークポイントを使用してデバッグを行う場合、現在のデバッガーのポジションまでの出力されたマッピングの出力を確認することができます。生成された出力の一部のプレビューはXML、flat text、およびEDI ターゲットコンポーネント によりサポートされています。

デフォルトでは、「F5」を押すと、(ブレークポイントを定義せずに) MapForce は、デバッグモードで全体のマッピングを実行し、「出力」 タブに切り替え、最終的に生成された出力を表示します。しかしながら、ブレークポイントを定義している場合、またはデバッグを段階的に行っている場合 (**F11** または **Ctrl + F11**) デバッガー実行はマッピング出力が生成されているにもかかわらず停止されます。マッピング出力が部分的に書かれている場合でも、「出力」 タブをクリックすることができ、また、プレビューすることができます。



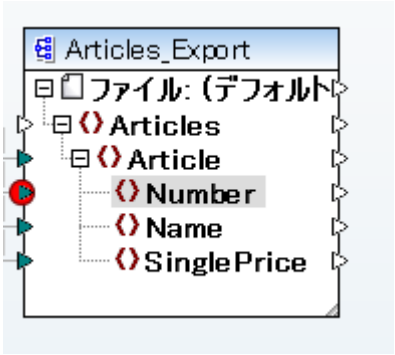
制限

- 現在計算されるターゲットノードは、「出力」 タブに常に表示されているわけではありません。例: XML 属性は内部で集められ、一度に書き込まれます。
- 出力が複数のファイルを作成する場合、現在書き込まれているファイルのみが表示されます。他の出力ファイルに切り替えると無効化されます。

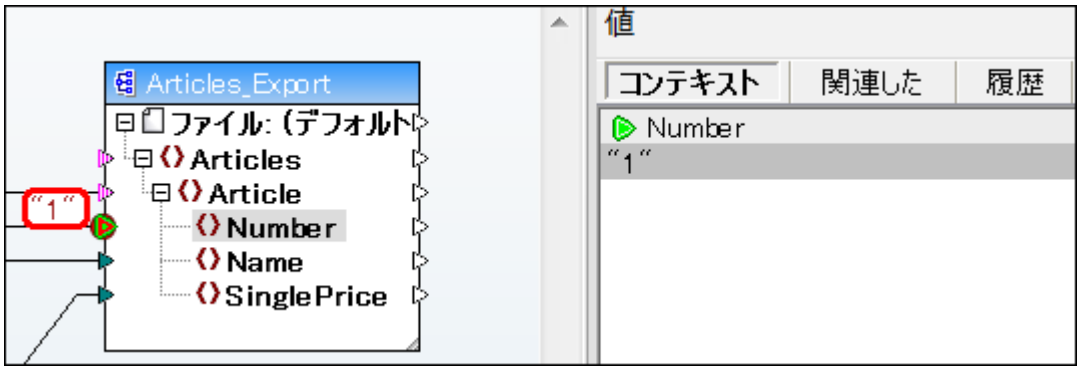
5.9 コネクタの現在の値を表示する

(ステップでデバッグを実行しているか、コネクタ上にブレークポイントが定義されているため) デバッガーの現在の実行のポジション ( ) が特定のコネクタ上にある場合、コネクタにより処理される現在の値は値ウィンドウの「コンテキスト」タブに表示されます。この値が出力に書き込まれようとしている値であり、すなわち、"現在"の値になります。これはまたコンテキストウィンドウにコンテキストが表示される値でもあります (コンテキストウィンドウの使用を参照)。

このシチュエーションを理解するには、<マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥ディレクトリから **PreserveFormatting.mfd** を開きます。ターゲットコンポーネントの `Number` ノードの入力コネクタをクリックし、「F9」を押して、ブレークポイントを追加します。



次に、**F5** を押して、デバッグを開始して、結果全体を確認します。



イメージで表示されるように、現在のデバッガーのポジション (およびブレークポイント) ターゲットコンポーネントの `Number` ノード上にあります。値ウィンドウはこのノードが値 "1"を有することを示しています (この値もマッピング上では、赤い縦じまハイライトされています)。

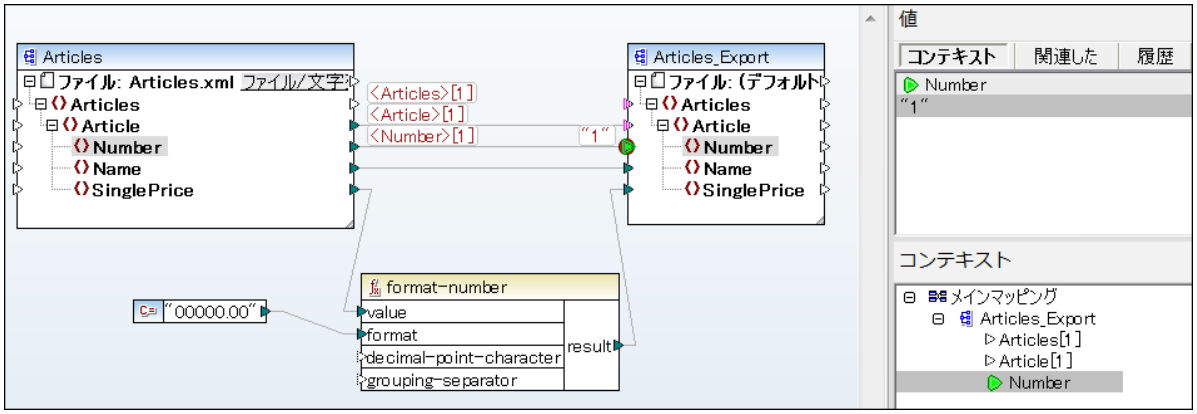
(C) 2015-2021 Altova GmbH

5.10　最近の過去にステップバックする

マッピングコネクタの横のデータオーバーレイ（小さな表方形のボックス）をクリックすると、値 ウィンドウは名前、とオプションで、選択されたコネクタに関連した値を表示します。フォーカスは現在のデバッガーのポジションでは無く、選択されたデータオーバーレイです。このビューをデバッグの履歴を少し戻ったビューと考えることができます。マッピングは、現在のデバッガーのポジションに関連した最後のいくつかのコネクタのためのオーバーレイのみを表示するため、これは、"近い" 過去です。関連した" データオーバーレイをクリックすると、値 ウィンドウは自動的に関連したタブに切り替えます。

このシナリオの説明に関しては、<マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥ ディレクトリからマッピング **PreserveFormatting.mfd** を開いてください。

マッピングを開いた後、ターゲットコンポーネントの `Number` ノードの横のコネクタをクリックし、「F9」を押して、ブレークポイントを追加します。「F5」を押してデバッグを開始します。ソースコンポーネントの `Number` ノードの横にあるデータオーバーレイ（小さな表方形のボックス）をクリックします。



マッピング内でコネクタは、通常、複数回反復されるため、反復の現在のインデックスは各かっこ内に表示されます: <Number>「1]。また、コネクタは値を持ち、その値がまた等号サインの後に以下のように表示されます: <Number>「1]=1。同じ値は以下のように値ウィンドウの新規の行に表示されます。

特定の値についての追加情報が必要な場合、その値を生成したコンテキストを再作成できることに注意してください（ コンテキストを値に設定する を参照）。

5.11 コネクタにより処理された値の履歴を確認する

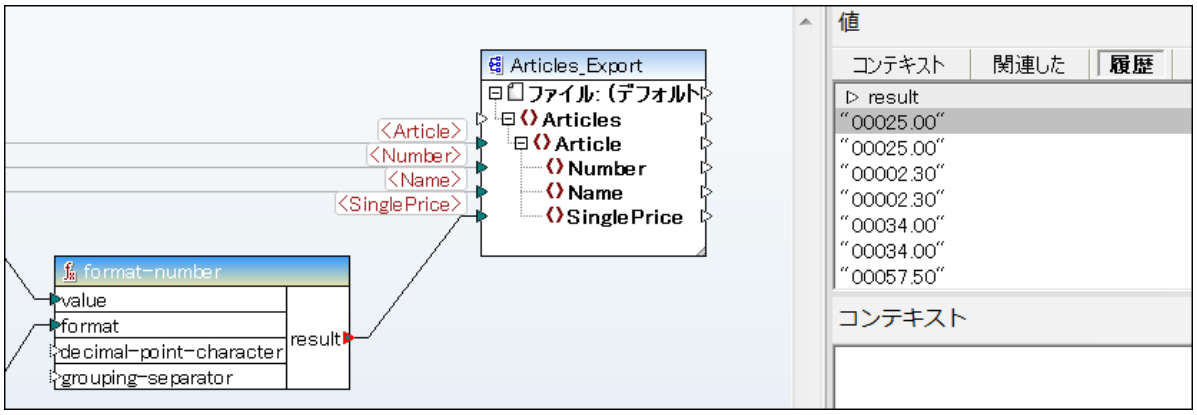
オプション「**トレースの履歴の全体を保管**」が有効化されている場合、(現在の実行のポジションまで)そのコネクタにより処理された全ての値の履歴をビューすることができます(デバッガー設定 を参照) 。

コネクタをクリックすると、履歴が表示されます。値ウィンドウの「**履歴**」タブをクリックします。このオペレーションはマッピングの実行の開始から、現在のデバッガーのポジションまでの間、値を処理したコネクタに対してのみ意味があることに注意してください。

このケースを説明するには、ブレークポイントを使用せず、始めからマッピングをデバッグし、特定のコネクタにより処理されたすべての値の履歴を確認してみましょう。最初に、<**マイドキュメント**>¥Altova¥MapForce2021¥MapForceExamples¥ディレクトリから マッピング **PreserveFormatting.mfd** を開きます。既に開かれている場合は、以下を行います:

- ブレークポイントが存在する場合、全てクリアします(ブレークポイントの追加と削除を参照)。
- Shift + F5 を押すと、現在実行中のデバッグを中止することができます。

準備ができると、「F5」を押して、新規のデバッグの操作を開始します。「F5」を押すと、MapForce はマッピングをデバッグモードで実行し、「**出力**」タブに切り替えます。「**マッピング**」タブをクリックして、メインマッピング ウィンドウ `format-number` 関数 の `result` ノードに移動します(下のイメージでは赤でハイライトされています)。最後に、値ウィンドウの「**履歴**」タブをクリックします。表示される値を確認してください。

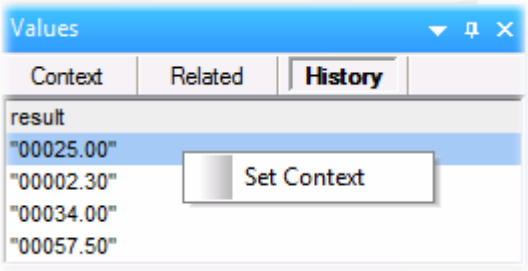


上のイメージで表示されるとおり、この特定のノード(`result`)は全部で4つの値を処理しました。特定の値についての追加の情報が必要な場合、その値を作成したコンテキストを再作成することができることに注意してください(コンテキストを値に設定する を参照)。

(C) 2015-2021 Altova GmbH

5.12　コンテキストを値に設定する

コンテキストの値に設定することは、その値を生成したマッピングコンテキストを詳細に確認する、過去にさかのぼり比較するアクションです。コンテキストを(**関連した** タブ、**シーケンス** タブ、または「**履歴**」タブ内の)値ウィンドウに表示される値に設定することができます。「**トレースの履歴の全体を保管**」オプションを有効化した場合(デバッガー設定 を参照)、「**履歴**」タブは、現在選択されているコネクタにより処理されたすべての値を表示します。ですから、この場合、追加して、そのコネクタのために過去にさかのぼりコンテキストの値を設定することができます。

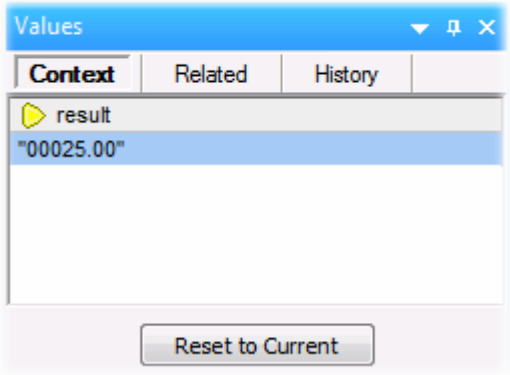


コンテキストを値に設定するには、以下を行います:

- 値を右クリックして、コンテキストメニューから、「**コンテキストの設定**」を選択します
- 値をダブルクリックします。

値をコンテキストに設定すると値を生成した状況を再作成し、**値**ウィンドウと**コンテキスト**ウィンドウを選択したコンテキストに従い作成するために、MapForce は、マッピングエリアをハイライトします。コンテキスト内のマッピングで使用されたビジュアルヒントのレジェンドに関しては、デバッグモードについてを参照してください。コンテキスト自身についての詳細は、コンテキストウィンドウの使用を参照してください。

手動で設定されたコネクタは、最近の実行ポジションでないことを示す黄色（ ）です。最近の実行ポジションに切り替えるには、(適用できる場合) 値ウィンドウの「**コンテキスト**」タブから、「**現在にリセット**」ボタンをクリックします。



5.13 デバッガー設定

MapForce デバッガーに適応することのできる設定へのアクセスの方法は、以下のとおりです。メニューコマンド「ツール | オプション」 を選択して、から「デバッガー」をクリックします。 使用することのできる設定は、以下のとおりです:

値の最高ストレージの長さ

値ウィンドウ内で表示される文字列の長さの値を定義します（最低 15 文字）。ストレージの長さを高い値に設定すると、使用可能なシステムメモリを消耗する点に注意してください。

トレースの履歴の全体を保管

MapForce に、デバッグ中マッピング内の全てのコンポーネントのコネクタにより処理された全ての値を保管するように命令します。このオプションが有効化されていると、デバッグの始めから MapForce により処理された全ての値が、メモリ内に保管され、デバッグを中止するまで、ウィンドウ内で分析のために使用することができます。データ優先型マッピングをデバッグする場合にこのオプションを有効化することは、デバッグの実行を遅らせ、使用できるシステムメモリを消耗するため奨励されません。 このオプションが無効化されている場合、MapForce は、現在の実行のポジションに相対するノードの最も新しいトレース履歴のみを保管します。

(C) 2015-2021 Altova GmbH

6 データソースとターゲット

このチャプターでは**MapForce Enterprise Edition** がマッピングに必要なソースとターゲットコンポーネントの型について説明されています。

- XML とXML スキーマ
- データベース
- CSV とテキストファイル
- MapForce FlexText
- EDI (ASC X12、HIPAA X12、HL7 Version 2、IATA PADIS、NCPDP SCRIPT、SAP IDoc、TRADACOMS、UN/EDIFACT などのEDI ベースまたはEDI 関連のフォーマットが含まれます)
- JSON
- Microsoft OOXML Excel 2007+
- XBRL
- HL7 バージョン3
- プロトコルバッファー
- 生バイナリファイル

6.1 XML と XML スキーマ

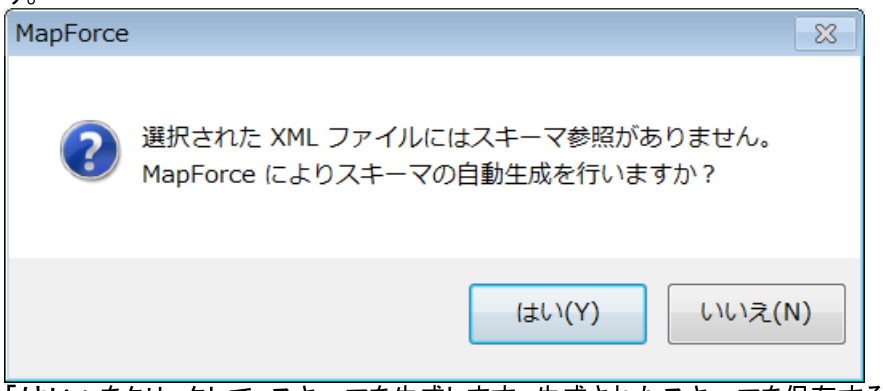
Altova W eb サイト: XML マッピング

このドキュメントのはじめに、XML とXML スキーマファイルをソースまたはターゲットコンポーネントとして、使用する簡単なマッピングのサンプルについて説明しました。このセクションはマッピング内でXML コンポーネントの使用についての詳細について説明します。以下のトピックが含まれます:

- XML スキーマの生成
- XML コンポーネント設定
- DTD をスキーマコンポーネントとして使用
- 派生したXML スキーマ型
- QName
- Nil 値 / Nillable
- コメントと処理命令
- CData セクション
- ワイルドカード – xs:any / xs:anyAttribute
- 複数のスキーマからのデータのマージ
- カスタム名前空間の宣言

6.1.1 XML スキーマの生成

スキーマがない場合、MapForce は、自動的にXML スキーマを既存のファイルをベースに生成することができます。(メニューコマンド「挿入 | XML スキーマ/ファイル」を使用して) マッピングエリアにスキーマを持たないXML ファイルを追加すると、次のダイアログボックスが表示されます。



「はい」をクリックして、スキーマを生成します。生成されたスキーマを保存するディレクトリを選択するようにプロンプトされます。

MapForce がXML ファイルからスキーマを生成する場合、要素/属性は、XML インスタンスドキュメントから推定されなければならず、予期しないものの可能性があります。生成されたスキーマがインスタンスデータを正確に表していることを確認することが奨励されます。

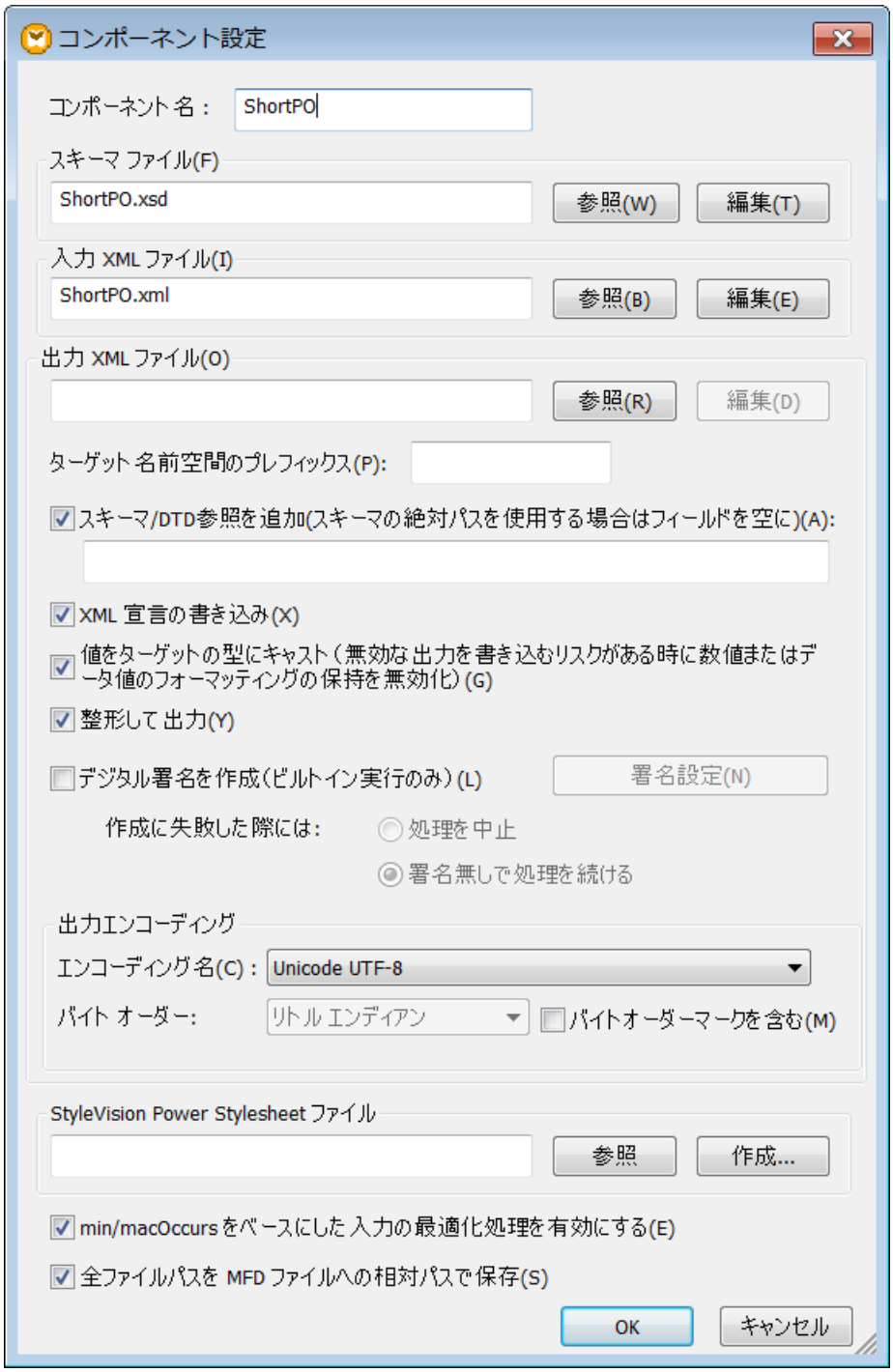
要素と属性が1つ以上の名前空間に存在する場合、MapForce は個別のXML スキーマをそれぞれの名前空間のために生成します。このため、複数のファイルがディスク上で作成される可能性があります。

(C) 2015-2021 Altova GmbH

6.1.2 XML コンポーネント設定

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

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



XML コンポーネント設定 ダイアログボックス

使用することのできる設定は、以下のとおりです.

コンポーネント名	コンポーネント名はコンポーネントを作成する際に自動的に生成されます。ですが、後に名前を変更することが可能です。

(C) 2015-2021 Altova GmbH

	コンポーネント名が自動的に生成され、インスタンスファイルをその後に選択した場合、コンポーネント名もオプションで更新するようにMapForce は、プロンプトします。 コンポーネント名 はスペース（例: "ソースXML ファイル"）またフルストップ文字も含むことができます（例: "Orders.EDI"）。コンポーネント名は、スラッシュ、バックスラッシュ、コロン、二重引用符、行頭および末尾スペースを含んではなりません。全般的に、コンポーネントの名前を変更するさいの以下の影響を考慮してください: • マッピングをFlowForce Server にデプロイする場合、コンポーネント名は一意である必要があります。 • コマンドラインに入力できる文字のみを使用することが奨励されます。コマンドラインとW indows 内では、それぞれの文字で異なったエンコードを使用する場合があります。
スキーマファイル	MapForce によりデータを検証とマップするために使用されたXML スキーマファイルの名前とパスを指定します。 スキーマファイルを変更するには、「参照」をクリックして、新しいファイルを選択します。XMLSpy 内のファイルを編集するには、「編集」をクリックします。
入力 XML ファイル	MapForce がデータを読み込むXML インスタンスファイルを指定します。このフィールドはソースコンポーネントにより重要で、最初にコンポーネントを作成した際に入力され、XML インスタンスファイルに割り当てられます。 ソースコンポーネント内で、インスタンスファイル名はXML ルート要素とレファレンススキーマを検出するため、また選択されたスキーマに対して検証するために使用されます。 ファイルのロケーションを変更するには、「参照」新しいファイルを選択します。 XMLSpy 内のファイルを編集するには、「編集」をクリックします。
出力 XML ファイル	MapForce がデータを書き込むXML インスタンスファイルを指定します。このフィールドはターゲットコンポーネントにより重要です。 ファイルのロケーションを変更するには、「参照」をクリックして、新しいファイルを選択します。XMLSpy 内のファイルを編集するには、「編集」をクリックします。
ターゲット 名前空間のためのプレフィックス	ターゲット名前空間のためにプレフィックスを入力することを許可します。プレフィックスを割り当てる前に、ターゲットスキーマ内でターゲット名前空間が定義されている必要があります。
スキーマ/DTD レファレンスの追加	参照されたXML スキーマファイルをXML 出力のルート要素に対してパスを追加します。このフィールドに入力されたスキーマのパスは、`xsi:schemaLocation` 属性またはDTD が使用される場合、`DOCTYPE` 宣言 内の生成されたターゲットインスタンスファイルに書き込まれます。 XQuery またはC++ 出コードを生成する場合、DTD レファレンスの追加はサポートされません。 このフィールドにパスを入力することにより、XML インスタンスファイルにより参照されたスキーマファイルがどの箇所に位置するかを決定することができます。これにより、マッピングが実行される際、出力インスタンスがマッピングのデスティネーションで検証されることを保証できます。また、**http://**アドレスや絶対及び相対パスをこのフィールドに入力することもできます。

<table>
<tr><td></td><td>このオプションを無効化することにより、XML インスタンスを参照されたXML スキーマまたはDTD から、切り離すことができます(例: 結果するXML 出力を基となるXML スキーマを持たない相手に送信する場合など)。</td></tr>
<tr><td>XML 宣言の書き込み</td><td>このオプションにより生成された出力からXML 宣言 を表示しないようにすることができます。デフォルトでは、オプションが有効化されていると、XML 宣言 が出力に表示されます。

この機能はMapForce ターゲット 言語と実行エンジンでサポートされています。

<table>
<tr><th>ターゲット言語 / 実行エンジン</th><th>出力がファイルの場合</th><th>出力が文字列の場合</th></tr>
<tr><td>ビルドイン</td><td>はい</td><td>はい</td></tr>
<tr><td>MapForce Server</td><td>はい</td><td>はい</td></tr>
<tr><td>XSLT, XQuery</td><td>はい</td><td>いいえ</td></tr>
<tr><td>コード ジェネレータ(C++、C#、Java)</td><td>はい</td><td>はい</td></tr>
</table>
</td></tr>
<tr><td>ターゲット型に値をキャストする</td><td>Aターゲット XML スキー型がマッピングで使用されるかを定義します。またはターゲットコンポーネントにマップされる全てのデータが文字列値として扱われるかを定義します。デフォルトでは、この設定 が有効化されています。

このオプションを無効化することにより、正確なフォーマットの値を保持することができます。例: これは、数値内で特定の小数点を必要とするスキーマ内のパターンファセットを満足させる場合に役にたちます。

マッピング関数を使用して、数値を文字列として必要とされるフォーマット内で使用し、この文字列をターゲットにマップすることができます。

このオプションを無効化にすると、無効な値の検出も無効化されます。例: 数値フィールドに文字が入力された場合。</td></tr>
<tr><td>整形出力</td><td>出力 XML ドキュメントに構造化された外見を与えるために整形出力します。それぞれの子ノードは、単一タブ文字のそれぞれの親のオフセットです。</td></tr>
<tr><td>デジタル署名の作成</td><td>XML 出力インスタンスファイルにデジタル署名を追加することを許可します。"ビルドイン"を変換言語として選択してもデジタル署名を追加することができます。(デジタル署名も参照)。</td></tr>
<tr><td>出力エンコード</td><td>出力インスタンスファイルの次の設定を指定することを許可します:

• エンコード名
• バイトオーダー
• バイトオーダーマーク(BOM) 文字が含まれるか否か。

デフォルトでは、「新しいコンポーネントのためのデフォルトのエンコード」オプション内で定義されたエンコードを持つ新しいコンポーネント。「ツール | オプション」からこのオプションにアクセスし、タブを生成します。</td></tr>
</table>

(C) 2015-2021 Altova GmbH

	マッピングがXSLT 1.0/2.0 を生成すると、**バイトオーダーマーク** チェックボックスを有効化しても、これらの言語はバイトオーダーマークをサポートしないため効果はありません。
StyleVision Power スタイルシートファイル	このオプションにより、Altova StyleVision スタイルシートファイルを選択または作成することができます。このようなファイルは、XML インスタンスファイルからのデータを HTML、RTF などの多種のレポートに適したフォーマットで出力することができます。 以下も参照: コンポーネント上で相対パスを使用する
min/maxOccurs をベースに入力処理の最適化の有効化	このオプションにより、`minOccurs` と`maxOccurs="1"` を持つ必要な属性または子要素などの1つのアイテムのみを含むシーケンスを特別に処理することが許可されます。この場合、シーケンスの最初のアイテムが抽出され、アイテムは シーケンスとしてではなく直接動的な値として処理されます 入力データがスキーマに対して場合、**有効でない場合**、エラーメッセージと共にマッピングを停止する空のシーケンスがマッピング内で生じる可能性があります。このような無効な入力を処理するには、このチェックボックスを無効化にします。
MFD ファイルに相対的なすべてのファイルパスを保存する	このオプションが有効化されると、MapForce は、コンポーネント設定ダイアログボックスに表示された、MapForce Design (.mfd) ファイルの場所に相対したファイルパスを保存します。以下も参照: コンポーネント上で相対パスを使用する。

6.1.3 "スキーマ" コンポーネントとしてを DTD 使用する

MapForce 2006 SP2 以降では、ソースならびにターゲットコンポーネントにて名前空間を意識したDTD がサポートされます。名前空間 URI がDTD の"xsIns" 属性宣言から抽出され、マッピングを行うことが可能になります。

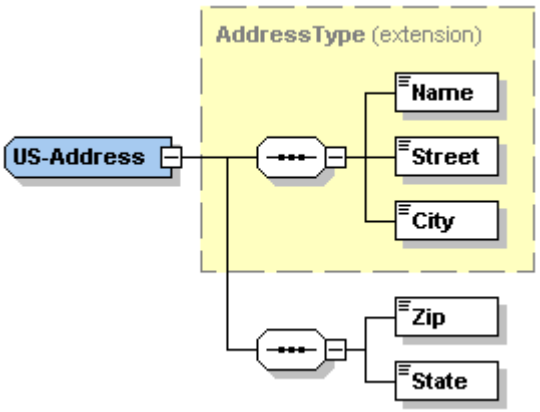
しかし StyleVision により使用されるDTD の様に、名前空間 URI を持たずにxmlns* 属性を含むDTD も存在します。このようなDTD は、MapForce で使用することができるように拡張する必要があります。以下に示されるように、DTD を修正して名前空間 URI を含むxmlns 属性を定義する必要があります:

```
<!ATTLIST fo:root
   xmlns:fo CDATA #FIXED 'http://www.w3.org/1999/XSL/Format'
   ...
>
```

6.1.4 派生した XML スキーマの型

MapForce では複合型の派生型に対するマッピングがサポートされます。派生型は、**xsi:type** 属性により特定の派生型を指定する XML スキーマの複合型です。

以下のスクリーンショットは、XMLSpy にて派生型の"US-Address" の定義を示したものです。基底型(または オリジナルとなっている複合型)はこの場合 AddressType となります。`Zip` と`State` の2 つの要素を新たに追加することで、派生型のUS-Address が作成されます。

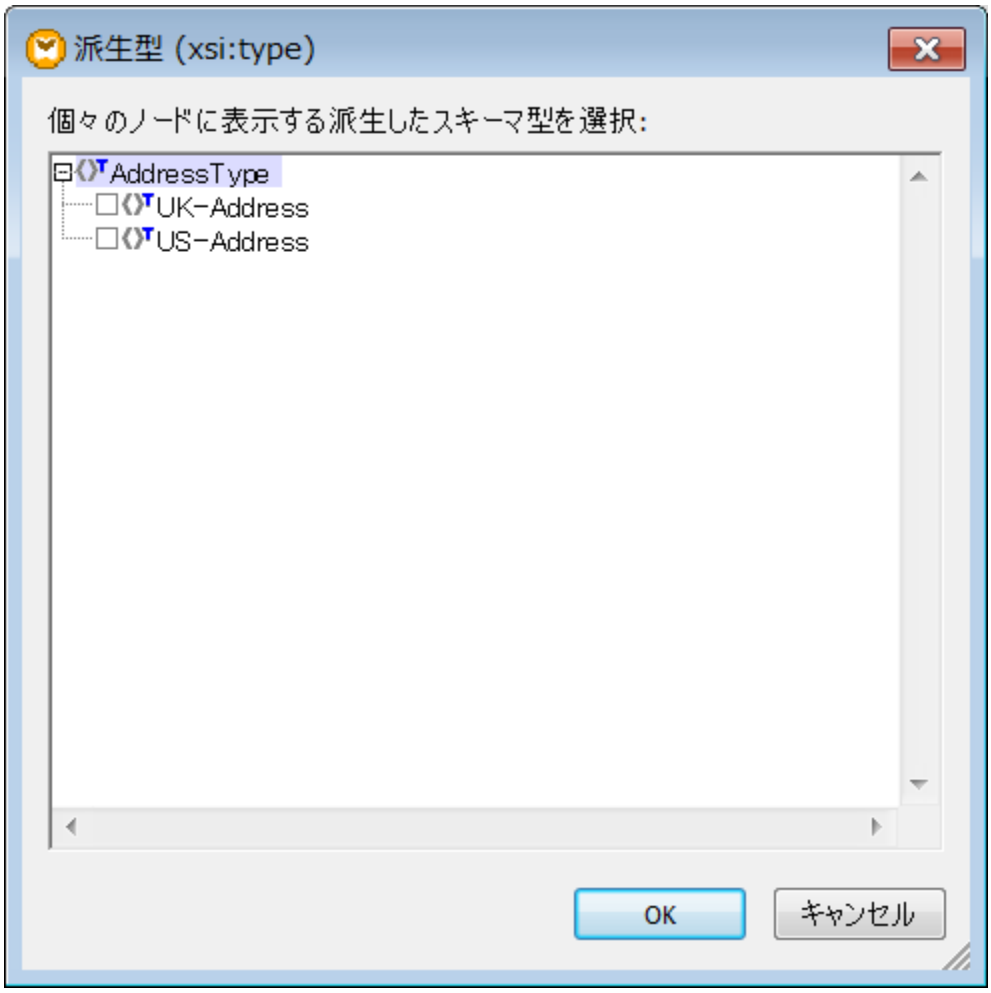


生成された型のサンプル(XMLSpy スキーマビュー)

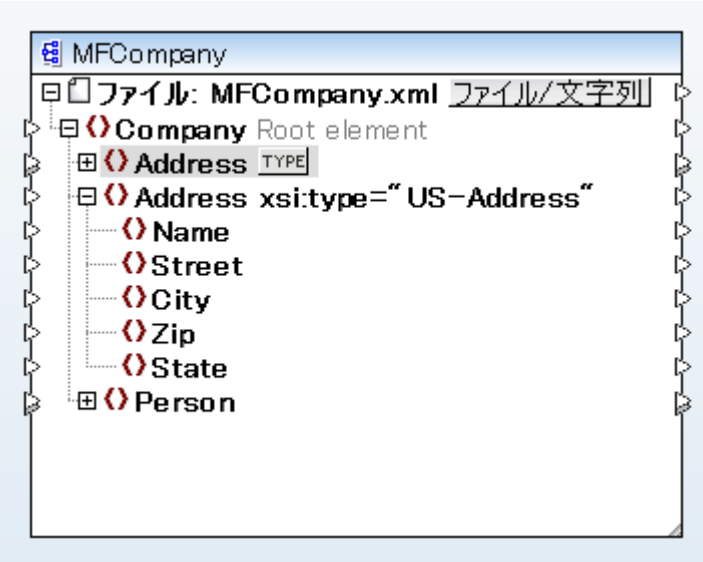
次のサンプルは、派生したXML スキーマ型からデータをマップ、またはスキーマ型へのデータのマップの方法を説明しています。

1. 「挿入」メニューから「XML スキーマ/ファイル」をクリックします。以下のXML スキーマ: **<マイドキュメント>\Altova\MapForce2021\MapForceExamples\Tutorial\MFCompany.xsd** を開きます
2. インスタンスファイルを提供するようにプロンプトされると、「スキップ」をクリックして、ルート要素をとして `Company` 選択します。

3. `Address` 要素の横の TYPE ボタンをクリックします。このボタンは、スキーマ内のこの要素のために存在する派生した型であることを示しています。



4. 使用する派生した型の横のチェックボックスを選択し、(この場合、`US-Address`)、「OK」を押して確認します。新規の要素 `Address xsi:type="US-Address"` がコンポーネントに追加されます。



これらのアイテムに対して直接、または US-Address から、マッピングを行うことが可能になります。

派生型ダイアログボックスにて複数のアイテムを選択することで、複数の派生型を追加/挿入することができます。各ノードにはそれぞれ独自の xsi:type 要素が与えられます。

6.1.5 QNames

MapForce は、マッピングの実行ランタイムで XML ファイルからデータを読み込む際に QName (修飾名) プレフィックス (http://www.w3.org/TR/xml-names/#ns-qualnames) を解決します。

QNames は、XML と XBRL インスタンスドキュメント内の名前空間 URI を参照および省略するために使用されます。QNames には、2 つの種類があります: プレフィックスを持つ、またはプレフィックスを持たない QNames.

PrefixedName	Prefix ':'	LocalPart
UnPrefixedName		LocalPart
		LocalPart が要素 または 属性の名前の箇所

例: 下のリストでは、<x:p/> が QName です:

- プレフィックス "x" は名前空間 "**http://myCompany.com**" の省略形です。
- p は、ローカルパートです。

```
<?xml version='1.0'?>
<doc xmlns:x="http://myCompany.com">
   <x:p/>
</doc>
```

MapForce は、**xpath2 | qname-related** 関数 ライブラリに QName に関連した関数の一部を含みます。

6.1.6 Nil の値 / Nillable

XML スキーマ仕様に従えば、nillable="true" 属性をスキーマ内の要素に対して指定することで、(コンテンツのデータ型により空のコンテンツが許されていない場合でも)コンテンツを含まない要素でも妥当とみなすことができるようになります。XML インスタンスドキュメント内に記述される属性のxsi:nil="true" により、要素は存在するが、コンテンツは含まれない(つまり空要素である)ことが示されます。このセクションはどのように、MapForce がソースとターゲットコンポーネント内のnil 要素を扱うかを説明します。

'xsi:nil' と 'nillable' の比較

xsi:nil="true" 属性はXML インスタンスファイルにおいてのみ定義されます。

```
14   <Person>
15     <PrimaryKey>2</PrimaryKey>
16     <ForeignKey>1</ForeignKey>
17     <EMail>biff@amail.com</EMail>
18     <First>biff</First>
19     <Last>bander</Last>
20     <PhoneExt>22</PhoneExt>
21     <OrderID xsi:nil="true"/>
22     <Title>IT services</Title>
23   </Person>
```

xsi:nil 属性はMapForce のマッピングにて視覚的(かつ明示的)に示されることは無く、自動的に処理されます。通常、null 値が許容されたノード(xsi:nil="true" 属性が含まれるノード)は存在するが、コンテンツには何も含まれない状態になります。

`xsi:nil` 属性は、通常自動的に処理されるため、MapForce グラフィカルなマッピングで明示的に表示されません。特に"nilled" ノード(`xsi:nil="true"` 属性を持つ) が存在する場合、そのコンテンツは存在しません。

マッピングソースにある null 値が許容された要素

マッピングにて null 値を許容する(nillable な) データを XML または XBRL 要素から読み取った場合、xsi:nil 属性が自動的にチェックされます。xsi:nil の値がtrue となっている場合、コンテンツは存在しないものとして扱われます。

nillable なソース要素から*単純コンテンツの*(属性は含まれるが子要素を含まない) nillable なターゲット要素に対して、target-driven 接続を作成し、xsi:nil がソース要素にセットされている場合、xsi:nil 属性がターゲット要素にも挿入されます(例: `<OrderID xsi:nil="true"/>`)。

全てコピー 接続を nillable なソース要素から nillable なターゲット要素へマッピングして、xsi:nil がソース要素にてセットされている場合、xsi:nil 属性がターゲット要素にも挿入されます(例: `<OrderID xsi:nil="true"/>`)。

ソース要素のxsi:nil にtrue がセットされているかを明示的にチェックするには、**`is-xsi-nil`** 関数を使用します。「nil 化」された要素に対して true が返され、それ以外のノードに対して false が返されます。

nil 化された(存在しない)ソース要素の値を別の値で代用するには、**`substitute-missing`** 関数を使用してください。

メモ

- コンテンツが無くても要素ノードが実際存在するため、**`exists`** 関数をnill 化されたソース要素 に接続すると、TRUE を返します。

- xsi:nil が設定されている箇所で（multiply と concat などの）単純型の値を期待する関数を要素に対して使用すると、要素コンテンツが存在しなく、結果を取得することはできません。これらの関数はソースノードが存在しないかのように振舞います。

マッピングターゲットとしての Nillable 要素

ターゲット優先 接続から nillable ソース要素に*単純型コンテンツ*（任意の追加属性は含むが子要素を含まない単一の値）を持つソース要素が xsi:nil に設定されているターゲット要素に作成する場合、MapForce は xsi:nil 属性をターゲット要素に挿入します（例: <OrderID xsi:nil="true"/>）。 xsi:nil="true" 属性が XML ソース要素にセットされていない場合、通常の方法で要素コンテンツがターゲット要素へマッピングされます。

マッピングが（子要素を持つ）*複合型*の nillable 要素に対して行われている場合、xsi:nil 属性が自動的に書き込まれることはありません。これはマッピングが行われた後で子要素が書き込まれるか分からない為です。全てコピー 接続を定義することで、ソース要素から xsi:nil 属性をコピーすることができます。

空のシーケンスまたはデータベースの NULL 値をターゲット要素へマッピングする場合、それが nillable であるかどうかにかかわらず、要素が作成されることはありません。

空のターゲット要素を xsi:nil="true" 属性がセットされた状態で（強制的に）作成するには、**set-xsi-nil** 関数をターゲット要素へ直接接続してください。この方法は単純型ならびに複合型のターゲット要素に対して使用することができます。

substitute-missing-with-xsi-nil 関数を使用することで、マッピングソースから値を得ることができない場合に xsi:nil をターゲットへ挿入することができます。これはソースノードが全く存在しない場合、または（multiply のような）計算に対して nil 化されたソースノードが与えられ、結果が得られなかった場合などに使用することができます。

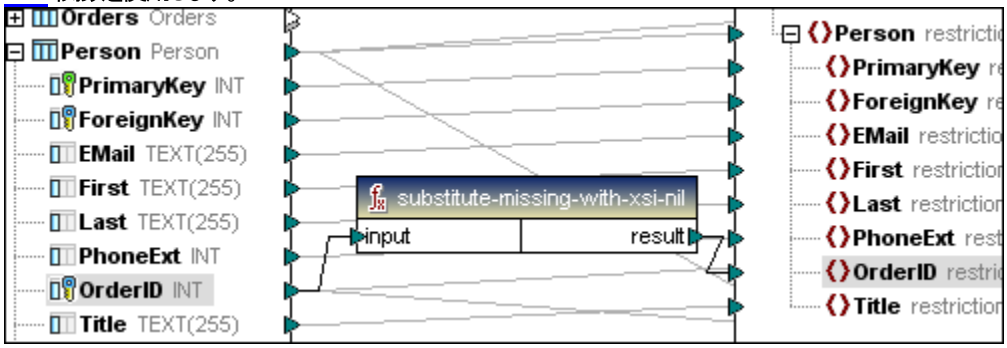
この関数はデータベースの NULL 値から xsi:nil="true" を持つ要素を作成するために使用することができます。この関数は単純型のコンテンツを持つノードに対してのみ使用することができます。

xsi:nil を生成する関数は、（**if-else** 関数などの）値に対して動作する関数またはコンポーネントにパススルーされることはできません。

NULL データベースフィールドから xsi:nil へマッピング

NULL データベースフィールドが XML スキーマの nillable 要素へマッピングされた場合、実際にデータを含んでいるフィールドに対応したターゲット要素だけが生成されます。NULL データベースフィールドの要素はターゲットコンポーネントにて作成されません。

ターゲットコンポーネント内ですべての要素の作成を強制するには、アライブラリのノード関数から **substitute-missing-with-xsi-nil** 関数を使用します。



(C) 2015-2021 Altova GmbH

上のスクリーンショットは、どのように**substitute-missing-with-xsi-nil** 関数が全てのデータベースフィールドのためにターゲット要素を作成するかを示しています:

- ターゲット要素内で<OrderID xsi:nil="true"/> を含むすべての不足しているNULL データベースフィールド
- データベースフィールドがターゲット要素に直接マップされる既存のデータ例: <OrderID>1</OrderID>.

データベースコンポーネントのNull フィールドを確認するには、**データベース クエリ** ボタンをクリックして、データベーステーブルに対してクエリを実行します。Null フィールドは「結果」ウィンドウ内で「*NULL*」と表示されます。

	EMail	First	Last	PhoneExt	OrderID	Title
1	v.callaby@nanonull.com	Vernon	Callaby	582	*[NULL]*	Office Manager
2	f.further@nanonull.com	Frank	Further	471	*[NULL]*	Accounts Recei
3	l.matise@nanonull.com	Loby	Matise	963	1	Accounting Man
4	j.firstbread@nanonull.com	Joe	Firstbread	621	*[NULL]*	Marketing Manag

xsi:nil を NULL データベースフィールドにマップする

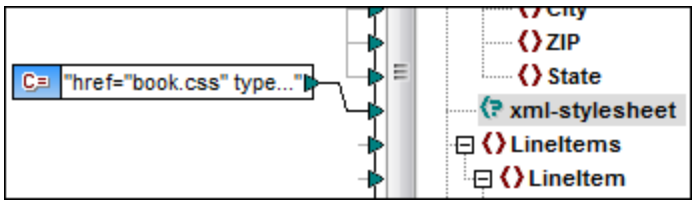
nill 化されたXML 要素をデータベース列にマップするには、MapForce はNULL 値をデータベースに書き込みます。**set-null** 関数を使用して、データベースフィールドをNULL に無条件で設定することもできます。

6.1.7 コメントと処理命令

コメントと処理命令はターゲット XML コンポーネントに挿入することができます。処理命令は、XML ドキュメントを更に処理するアプリケーションに情報をパスするために使用されます。コメントと処理命令は、全てエピーマップグループのためのノードのためには定義することができないことに注意してください。

処理命令の挿入方法:

1. ターゲットコンポーネント内の要素を右クリックして、コメント/処理命令を選択し、メニュー(前、後) から処理命令 オプションを選択します。
2. 処理命令(ターゲット) 名をダイアログ内に入力し、「OK」を押して確認します。例: xml-stylesheet。
   これによりこの名前をコンポーネントツリーに追加します。

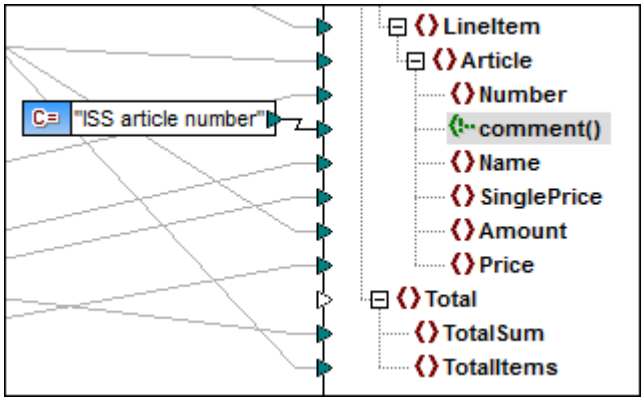


3. 以下を使用することができます。例: 処理命令属性の値を与えるために定数コンポーネントを使用する。例:
   `href="book.css" type="text/css".`

メモ
複数の処理命令は、ターゲットコンポーネント内の要素の前または後ろに追加することができます。

コメントの挿入:

1. ターゲットコンポーネント内の要素を右クリックして、コメント/処理命令を選択し、メニューからコメントオプション(前、後ろ)を選択します。



これは、コンポーネントツリーにコメントノード(`<!--comment()`)を追加します。
2. 定数コンポーネントを使用して、コメントコンテキストを与えます、またはソースノードをコメントノードに接続します。

メモ
単一のターゲットノードには1つのコメントのみを追加することができます。複数のコメントを作成するには、入力関数の複製を使用します。

コメント/処理命令の削除:

- 対応するノードを右クリックして、コメント/処理命令を選択し、ブライアウトメニューから「削除」コメント/処理命令を選択します。

6.1.8 CDATA セクション

CDATA セクションは、通常マークアップとして解釈される文字を含むテキストのブロックをエスケープするために使用されます。CDATA セクションは"<![CDATA[" で開始し、"]]>"で終わります。

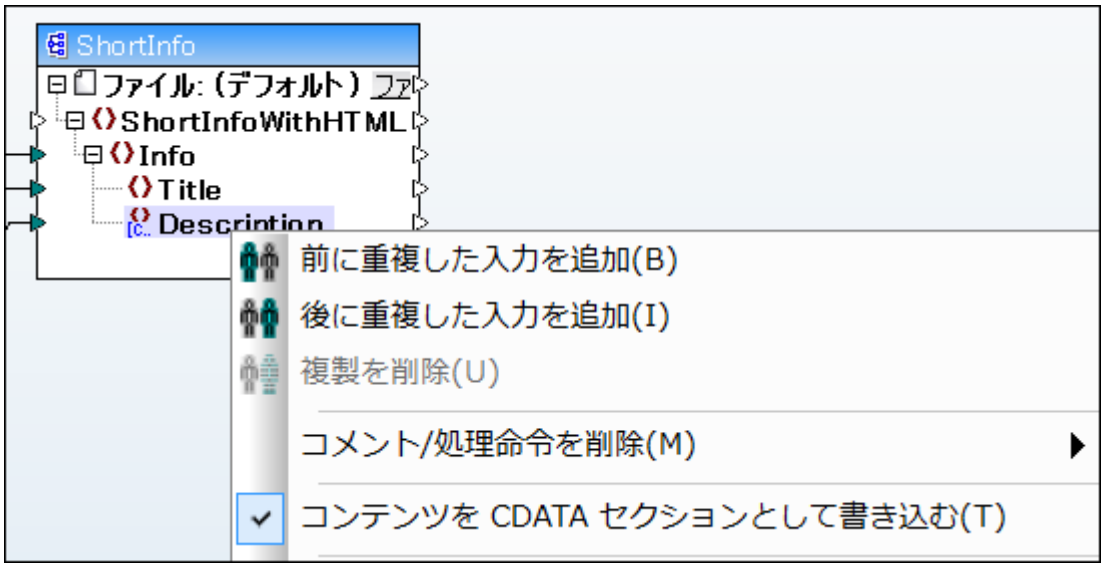
ターゲットノードは、CDATA セクション として受信される入力データを書き込むことができます。ターゲットノードコンポーネントは以下であることができます:

- XML データ
- データベースフィールド内に埋め込まれたXML データ
- XBRL ターゲット内の型指定されたディメンションのXML 子要素

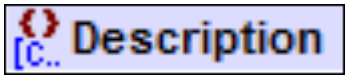
CDATA セクションの作成:

1. CDATA セクションとして定義するターゲットノードを右クリックして、 "CDATA セクションとしてコンテンツを書き込む"を選択します。

(C) 2015-2021 Altova GmbH



入力データは、区切り記号 ']]>' で終了されたCDATA セクションを含まない 用に警告プロンプトを表示します。「OK」をクリックしてプロンプトを閉じます。
下に表示される [C.. アイコン では、要素タグがこのノードがCDATA セクションとして定義されていることを示しています。

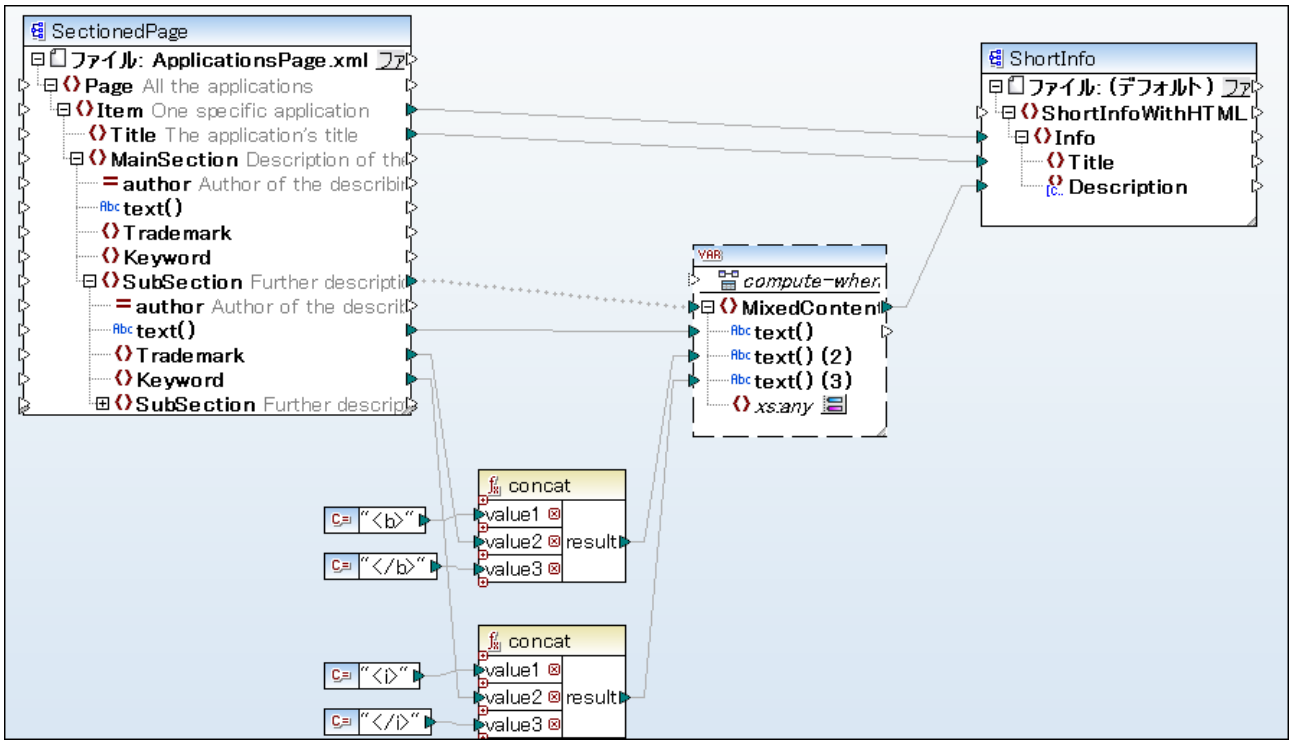


メモ
CDATA セクションは、複製されたノード、および、 xsi:type ノード上でも定義されることができます。

例:
...¥MapForceExamples フォルダー内で使用することのできる **HTMLinCDATA.mfd** マッピングファイルは、CDATA セクション が有効利用できる例を表示しています。

この例:

- 太字の開始 () と終了 () タグは **Trademark** ソース要素のコンテンツに追加することができます。
- イタリックの開始 () と終了 () タグは **Keyword** ソース要素のコンテンツに追加することができます。
- 結果のデータは、サブセクション要素コネクタが ソース優先 (複合型コンテンツ) ノードとして定義されているため、ソースドキュメント内に表示されるために複製 **text()** ノードにパスされます。
- 複合型ノードの出力は、CDATA セクションとして定義されている ShortInfo ターゲット コンポーネント内の **詳細** ノードにパスされます。

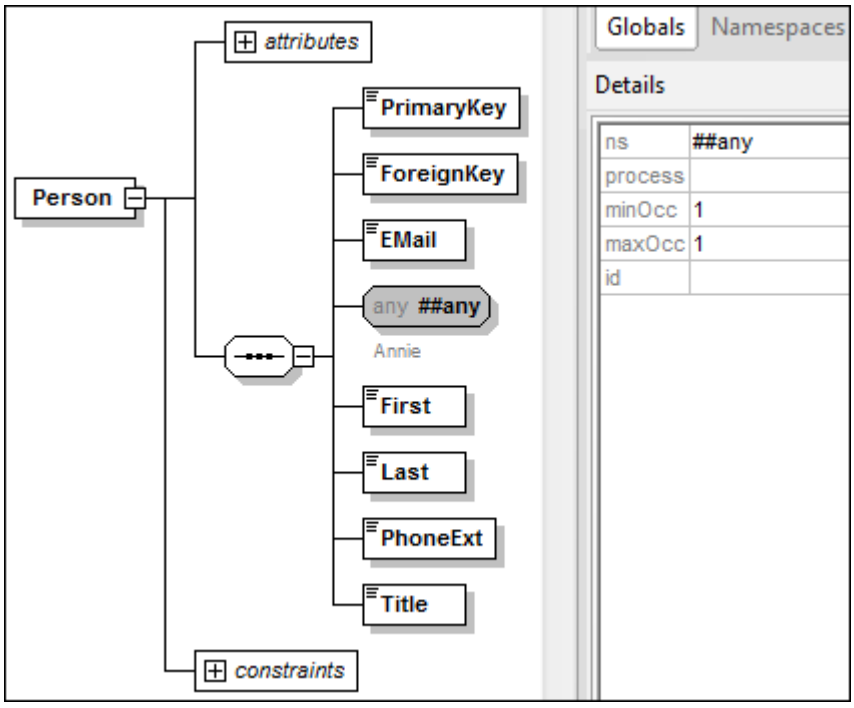


「出力」ボタンをクリックすることにより、マークアップされたテキストを含むCDATA セクションが表示されます。

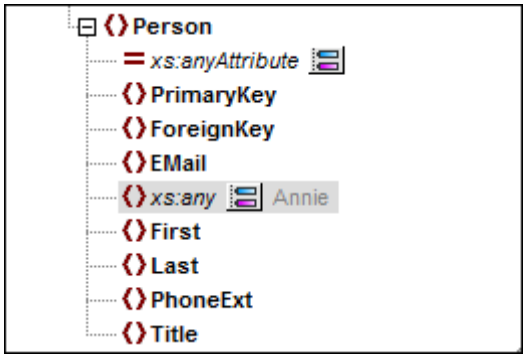
```
7     <Info>
8       <Title>MapForce</Title>
9       <Description><![CDATA[Altova <b>MapForce</b> 2014 Enterprise Edition is the premier <i>XML</i>
/ <i>database</i> / <i>flat file</i> / <i>EDI</i> data mapping tool that auto-generates mapping code in
<i>XSLT</i> 1.0/2.0, <i>XQuery</i>, <i>Java</i>, <i>C++</i> and <i>C#</i>. It is the definitive tool for
data integration and information leverage.]]></Description>
10    </Info>
```

6.1.9 ワイルドカード – xs:any / xs:anyAttribute

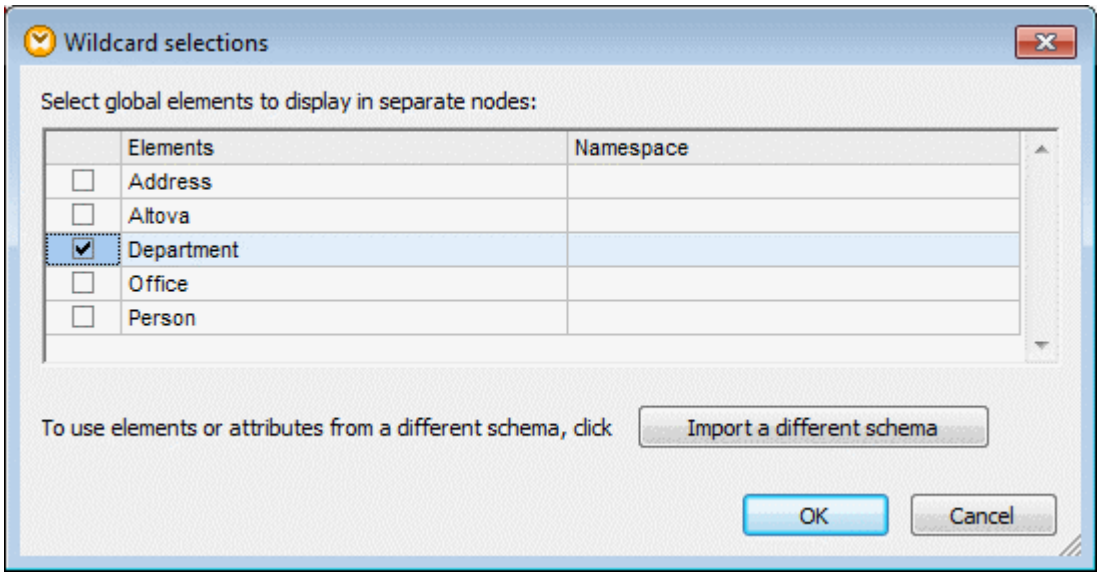
ワイルドカード xs:any (および xs:anyAttribute) は、すべての要素/属性をスキーマから使用することを許可します。スクリーンショットは XMLSpy のスキーマビュー内の"any" 要素を表示しています。



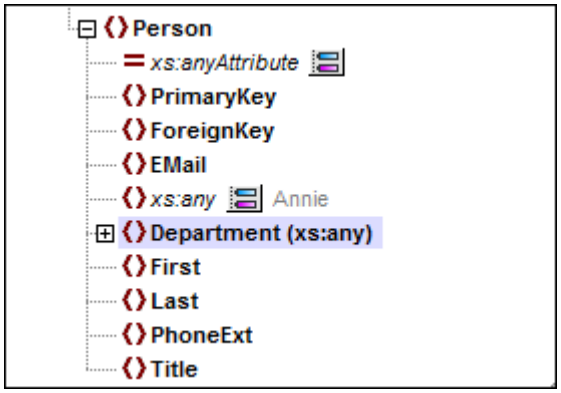
MapForce 内では、選択の変更 (excel1-compicon) ボタンが xs:any (または xs:anyAttribute) 要素の横に表示されます。



クリックすると、選択の変更 ボタン excel1-compicon が開かれ、"ワイルドカードの選択"ダイアログボックスが開かれます。エントリは、現在のスキーマ内で宣言されているグローバルな要素と属性を表示します。



クリックすると、選択の変更 ボタン excel1-compicon が開かれ、"ワイルドカードの選択"ダイアログボックスが開かれます。エントリは、現在のスキーマ内で宣言されているグローバルな要素と属性を表示します。



.これらのノードから、またはノードへ、他の要素と同様にマップすることができます。

コンポーネント上では、ワイルドカード要素、または属性は、名前に追加されている **(xs:any)** テキストとして識別されることができます。

ワイルドカード要素を削除するには、選択の変更 (excel1-compicon) ボタンをクリックして、"ワイルドカードの選択"ダイアログボックスから選択の解除を行います。

ワイルドカードと動的なノード名

ワイルドカードから、またはワイルドカードへのマッピングは、通常、使用することのできる、要素または属性がインスタンス内に表示されており、XML インスタンスがコンポーネントの XML スキーマにより宣言されている場合(または、外部のスキーマからインポートされている場合)に適しています。しかしながら、インスタンス内に表示される要素または属性が多すぎ、スキーマ内で宣言されるには多すぎる場合があります。`<message>` の子要素の数が任意である次のインスタンスの場合を考慮してみてください:

```
<?xml version="1.0" encoding="UTF-8"?>
<message>
   <line1>1</line1>
   <line2>2</line2>
   <line3>3</line3>
   ...............
   <line999></line999>
</message>
```

上記のようなシチュエーションでは、ノード名への動的なアクセスがワイルドカードの使用より先適しています（ノード名のマッピング を参照）。

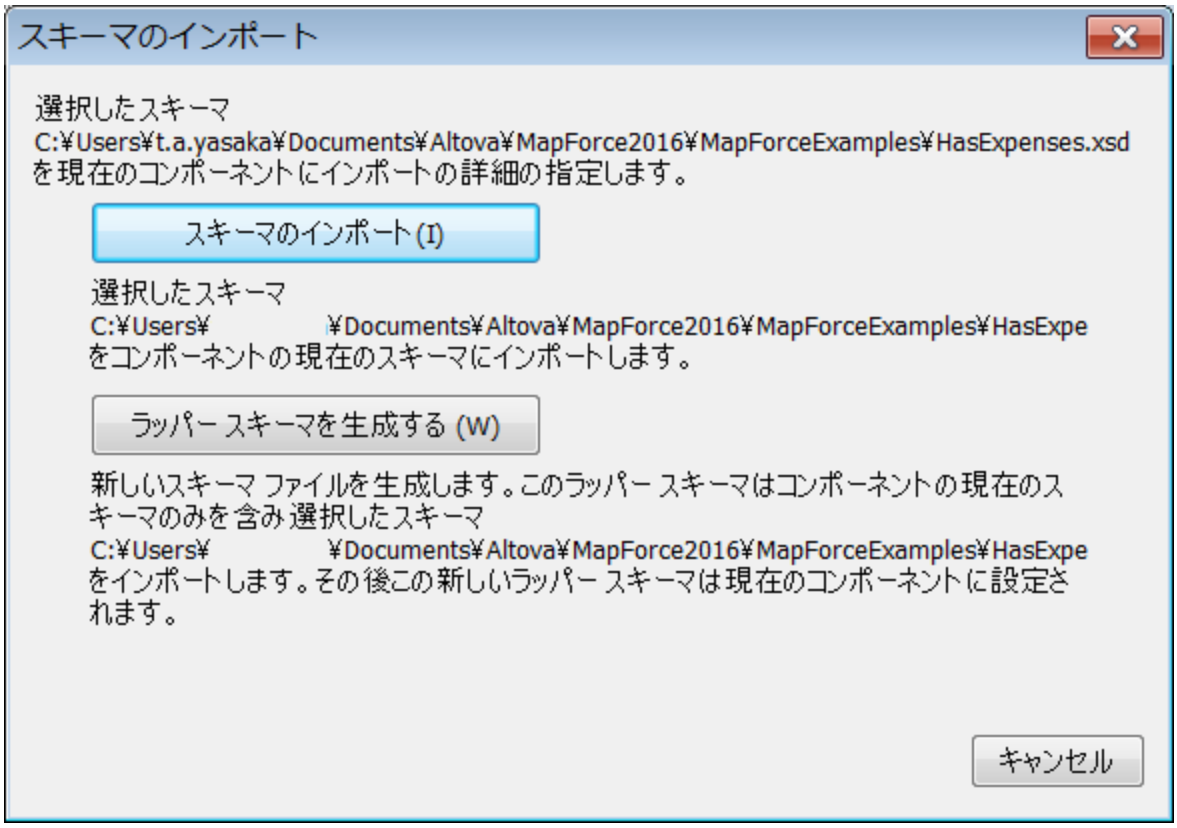
異なるスキーマからワイルドカードとして要素を追加する

コンポーネントに割り当てられた要素以外のスキーマからの要素は、ワイルドカードとして使用することができます。このような要素をコンポーネント上で表示するには、"ワイルドカードの選択" ダイアログボックス上の**異なるスキーマのインポート** ボタンをクリックします。これにより以下の2つのオプションを持つ新規ダイアログボックスが開かれます:

1. スキーマのインポート

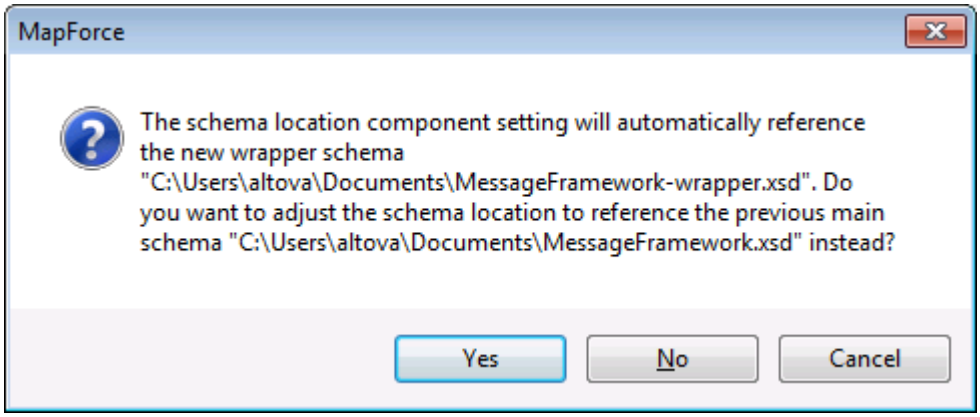
2. ラッパースキーマの生成

例えば、下のイメージは、**HasExpenses.xsd** という外部スキーマを現在のスキーマにインポートしようと試みると何が起こるかを表しています。



スキーマのインポート オプションは外部スキーマをコンポーネントに割り当てられている現在のスキーマにインポートします。このオプションはディスク上のコンポーネントの既存のスキーマをオーバーライドすることに注意してください。現在のスキーマがディスクではなくURL により開かれたリモートのスキーマである場合、変更することはできません（URL からコンポーネントを追加する を参照してください）。この場合、ラッパースキーマの生成オプションを使用してください。

ラッパースキーマの生成オプションは "ラッパー" スキーマという名前の新しいスキーマファイルを作成します。このオプションの利点は、コンテンツの既存のスキーマが変更されない点です。代わりに、既存のスキーマとインポートされるスキーマの両方を含む(ラッパースキーマという) 新しいスキーマが作成されます。このオプションを選択すると、ラッパースキーマを保存する場所を問われます。デフォルトでは、ラッパースキーマは somefile-wrapper.xsd. という書式の名前を持ちます。ラッパースキーマを保存した後、デフォルトでコンポーネントに割り当てられ、ダイアログボックスがプロンプトします:

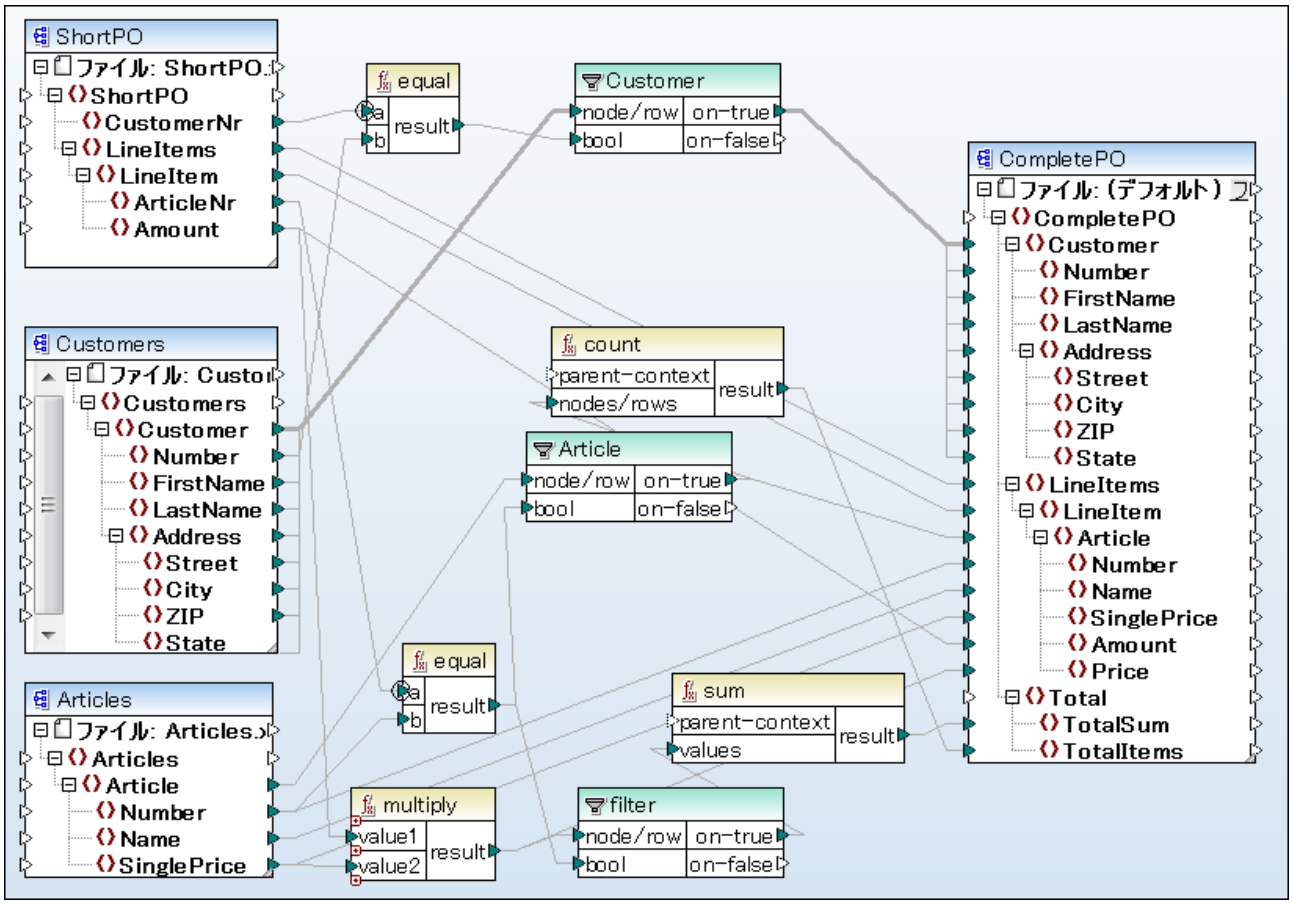


「はい」をクリックして、前のスキーマに戻します。それ以外の場合は、「いいえ」をクリックして新規に作成された、コンポーネントに割り当てられたラッパースキーマを保持するためにいいえをクリックします。

6.1.10 複数のスキーマからのデータのマージ

このサンプルは複数のファイルを単一のファイルにマージする方法について説明しています。具体的には、異なるスキーマを持つ複数のソースコンポーネントをターゲットスキーマにマージします。同一のスキーマをベースにした複数のファイルをマージする方法については、複数の入力または出力ファイルを動的に処理を参照ください。

CompletePO.mfd ファイルは **...\MapForceExamples** フォルダー内にあり、3 つのファイルを 1 つの XML ファイルへマージする方法を表示しています。



単一のターゲット XML ファイル **CompletePO.xml** に複数のソースコンポーネントデータが組み合わされていることにご注意してください。

- **ShortPO** は XML インスタンスファイルに関連付けられたスキーマで、番号と数量などの顧客番号と品物に関するデータのみが含まれています。(このファイルには顧客一人だけが収められています。)
- **Customers** は XML インスタンスファイルに関連付けられたスキーマで、顧客番号と、Name や Address といった顧客に関する詳細情報が収められています。
- **Articles** は XML インスタンスファイルが関連付けられたスキーマで、品物名やその数、単価といった品物に関する情報が収められています。
- **CompletePO** はインスタンスファイルを持たないスキーマファイルです。全てのデータは 3 つの XML インスタンスファイルより与えられています。このファイルの階層構造により、XML データのマージと出力を行うことが可能になります。

このスキーマファイルは XMLSpy のような XML エディターにより作成されており、MapForce から生成されたものではありません（**CompletePO.xml** インスタンスファイルが存在する場合、スキーマファイルの生成を行うことも可能です）。

CompletePO の構造はソースとなる XML ファイルの構造を組み合わせたものとなります。

filter コンポーネント（Customer）を使用することで、ShortPO と Customer XML ファイルで顧客番号が一致するデータを検索することができ、関連するデータをターゲットの CompletePO コンポーネントへ渡すことができます。

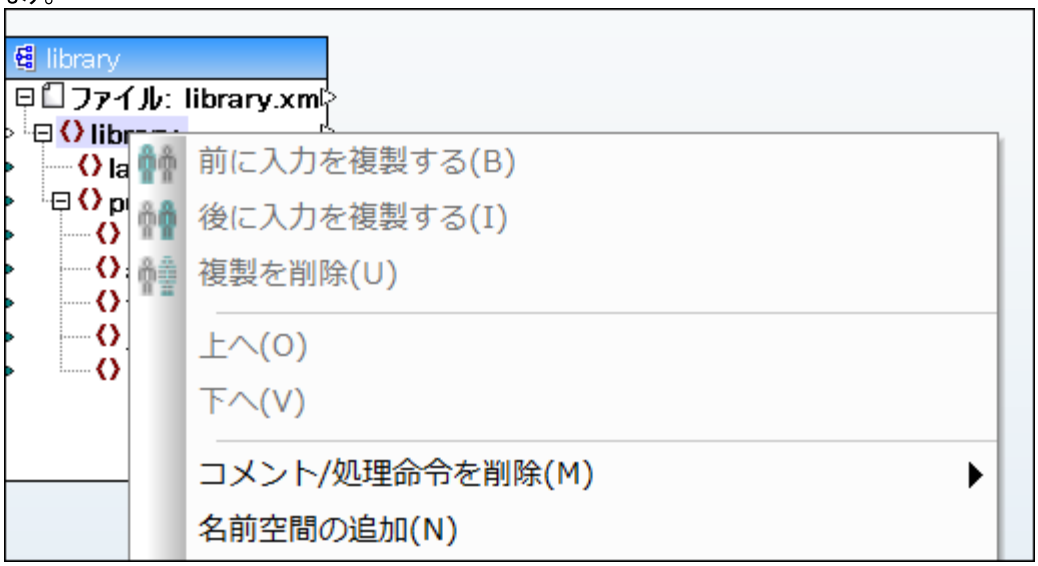
- `equal` 関数により、ShortPO の **CustomerNr** が Customer 以下にある **Number** と比較されます。
- ShortPO には 1 つの顧客情報（番号が 3）しか含まれていないため、顧客番号が 3 となっている顧客の情報だけがフィルターコンポーネントにお渡されます。
- フィルターコンポーネントの **node/row** パラメーターは **bool** パラメーターが true の時に Customer データを "on-true" ノードへ渡します。つまり、顧客番号が 3 の時だけ出力が行われます。
- 顧客と品物データの残りは、他にある 2 つのフィルターコンポーネントによりターゲットスキーマへパスされます。

6.1.11 カスタム名前空間の宣言

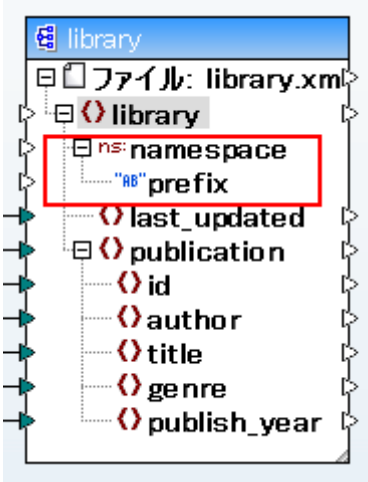
デフォルトでは、マッピングが XML 出力を生成する際に、各要素と属性の名前空間（または、名前空間のセット）が自動的に MapForce によりターゲットコンポーネントと関連するスキーマから生成されます。これは、MapForce 内のデフォルトの振る舞いで、XML 出力の生成を含むマッピングのシナリオの大半に適切です。

しかしながら、結果の XML 出力内で要素の名前空間の管理を更に行う場合、マッピングから要素の名前空間を手動で宣言することが必要な可能性があります。

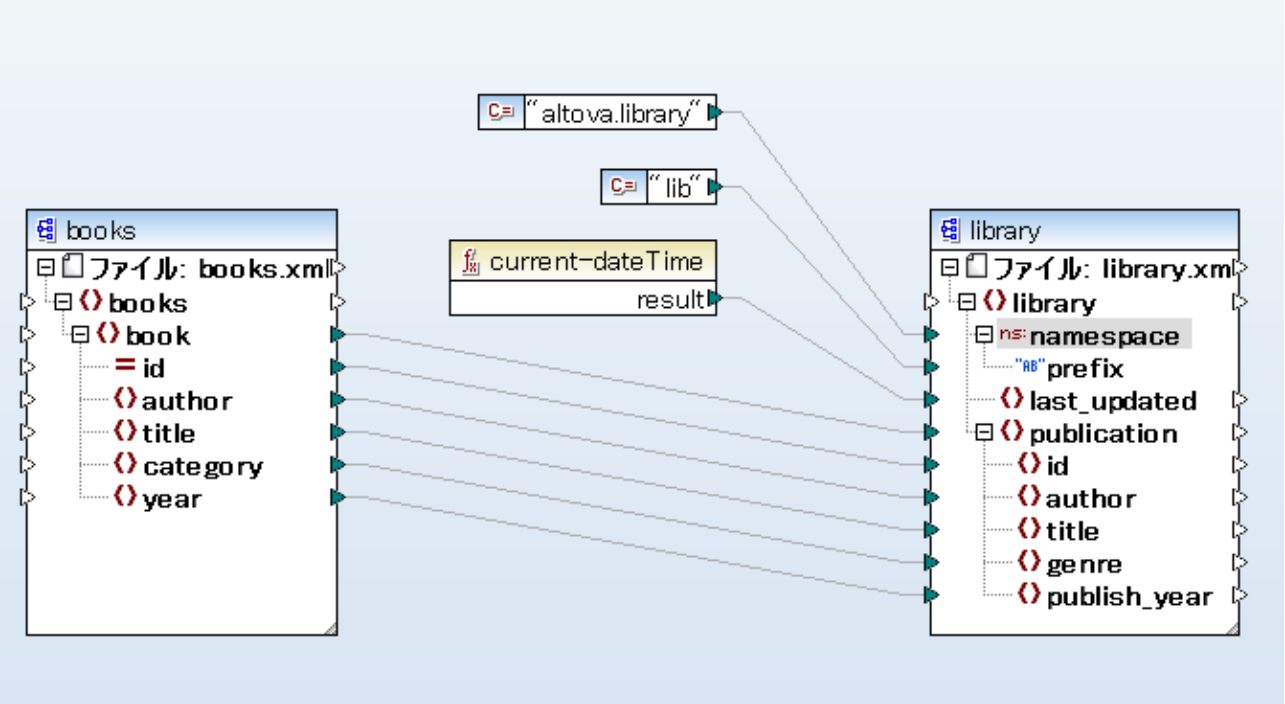
この仕組みを理解するには、＜マイドキュメント＞¥Altova¥MapForce2021¥MapForceExamples¥Tutorial¥ 内にある **BooksToLibrary.mfd** マッピングを開きます。`library` ノードを右クリックし、コンテキストメニューから「名前空間の追加」を選択します。



`library` ノードの下で次の 2 つの新しいノードが使用することができるようになっていることにご注意してください： `namespace` と `prefix`.



マッピングからの文字列の値をマップすることができます。下のイメージでは、名前空間「altova.library」とprefix「lib」を与える、2つの定数が「**挿入 | 定数**」メニューコマンドを使用して定義されています:



生成された出力内の結果としては、`xmlns:<prefix>="<namespace>"` 属性が、`<prefix>` と`<namespace>` がマッピング(この場合は定数)から来る値である箇所の要素に追加されます。生成された出力は、以下のようになります(ハイライトされている部分に注意してください):

```
<?xml version="1.0" encoding="UTF-8"?>
<library xmlns:lib="altova.library" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:noNamespaceSchemaLocation="library.xsd">
...
```

メモ カスタム名前空間の宣言(と「**名前空間の追加**」コマンド)は、ターゲット XML コンポーネントに取り有益であり、要素のみに適用することができます。属性に対して、「**名前空間の追加**」コマンドは、属性とワイルドカードノードに対しては使用することが

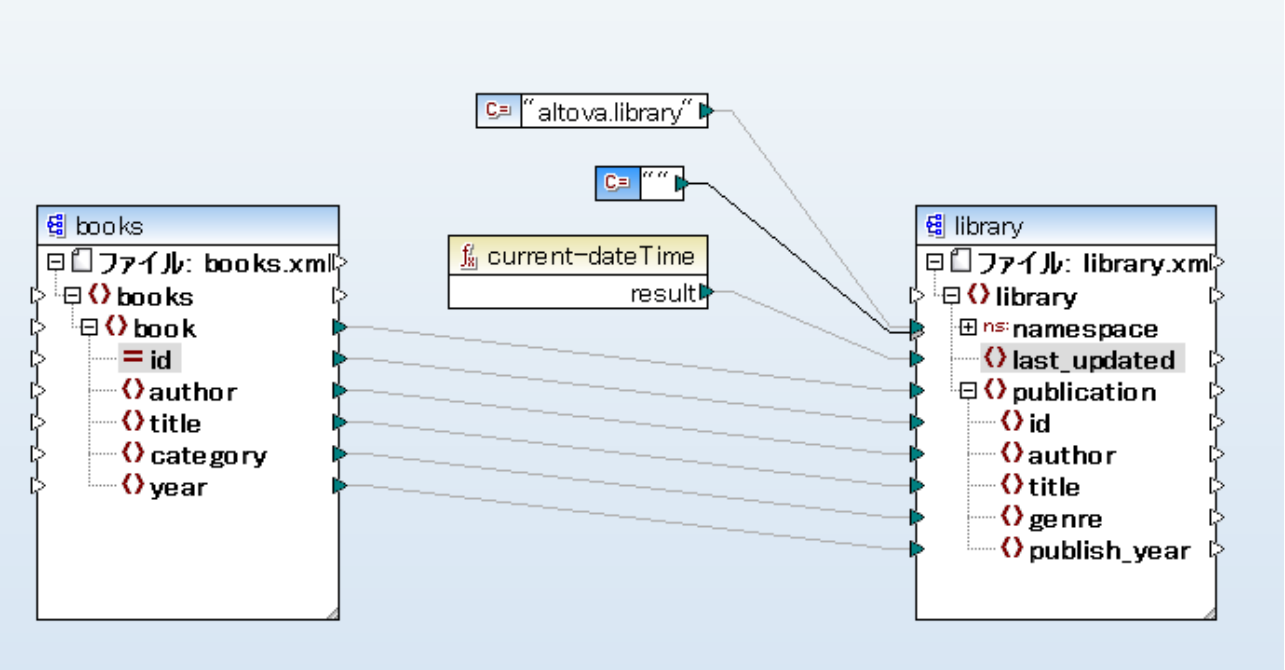
できません。ノードの子への動的なアクセスが有効化されている場合でも、または「全てをコピー」接続を使用してノードがデータを受け取るノードの場合でも使用することはできません（ノード名のマッピングを参照してください）。

必要に応じて、複数の名前空間を同じ要素のために宣言することができます。これを行うには、ノードをもう一度右クリックし、コンテキストメニューから「**名前空間の追加**」を選択します。新規のプレフィックスと名前空間の値に接続することのできる名前空間とプレフィックスノードの新規のペアを使用できるようになります。

以前に追加された名前空間の宣言を削除するには、 `ns:namespace` ノードを右クリックし、コンテキストメニューから「**名前空間の削除**」を選択します。

> 空の値が与えられる場合でも、`namespace` と `prefix` 入力コネクタの両方がマップされている必要があります。

（`xmlns="mydefaultnamespace"` のフォーマットの）デフォルトの名前空間を宣言する場合、`prefix` に対して空の文字列をマップします。このケースをマップ上で確認する場合は、2番目の定数が空の文字列になるように上のサンプルマッピングを編集します。



結果の出力は以下のようになります:

```
<?xml version="1.0" encoding="UTF-8"?>
<library xmlns="altova.library" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="library.xsd">
...
```

属性の名前のためにプレフィックスを作成する場合、例えば、`<number prod:id="prod557">557</number>` 、ノードの属性への動的なアクセスを有効化する、または`<number>` のために`prod:id` 属性を持つようにスキーマを編集することにより属性の名前のためにプレフィックスを作成することができます（次を参照してください: ノード名のマッピング）。

(C) 2015-2021 Altova GmbH

6.1.12 デジタル署名

XML デジタル署名はXMLドキュメントを暗号化されたコードにより署名するためのW3C 仕様で、XMLドキュメントが変更されていないことを保証するために使用することができます。MapForce のXML 署名機能ではUSA-SHA1 ならびにDSA-SHA1 型の証明書がサポートされます。

XML 署名に関する詳細については、XML 署名に関するW3C の仕様 https://www.w3.org/TR/xmldsig-core/ を参照ください。

MapForce ではXML ならびにXBRL 出力ファイルに対してXML デジタル署名の作成がサポートされます。デジタル署名は出力ターゲットがビルトインとなっている場合のみ生成することができます。出力ボタンがクリックされた時、また出力ファイルが保存された時に、生成された結果ファイルに対して署名が作成されます。

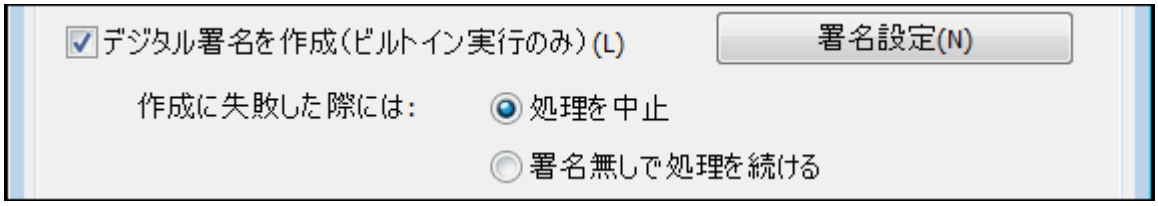
メモ MapForce Server はデジタル署名をサポートしません。

デジタル署名は出力ドキュメント最後の要素として埋め込むことができるほか、独立した署名ファイルとしても保管することができます。

- "Enveloped" が選択されている場合、XML ファイルにあるルート要素直下の最後の子要素として署名が作成されます。
- "Detached" が選択されている場合、独立したドキュメントとして署名ファイルが生成されます。

デジタル署名の生成を有効にするには:

1. 出力コンポーネントのヘッダーをダブルクリックするか、「**コンポーネント | プロパティ**」を選択することで表示される、出力コンポーネントのコンポーネント設定ダイアログボックスを開きます。
2. **デジタル署名を作成**チェックボックスを有効にします。



3. 必要な設定を入力するXML 署名設定ダイアログボックスが開かれます。(XML 署名設定を参照してください)。

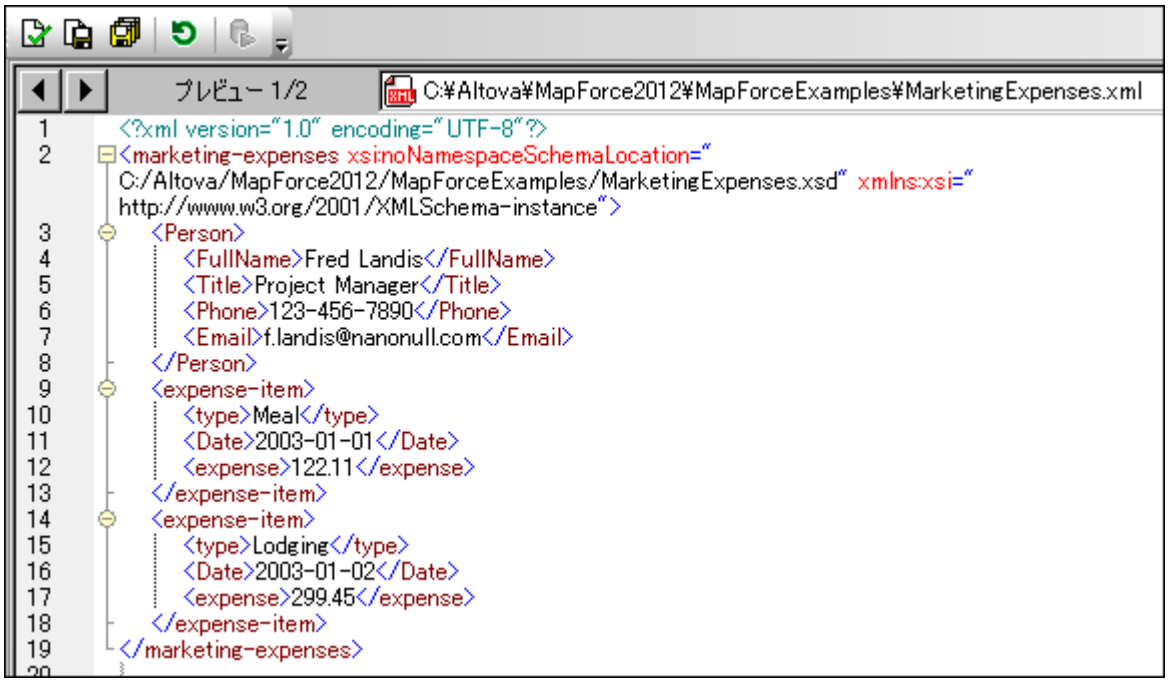
デジタル署名の設定を変更する:

1. 出力コンポーネントのヘッダーをダブルクリックするか、「**コンポーネント | プロパティ**」を選択することで表示される、出力コンポーネントのコンポーネント設定ダイアログボックスを開きます。
2. 署名設定ボタンをクリックして、XML 署名設定 ダイアログボックスを開きます。
3. 設定を入力して「OK」をクリックします。

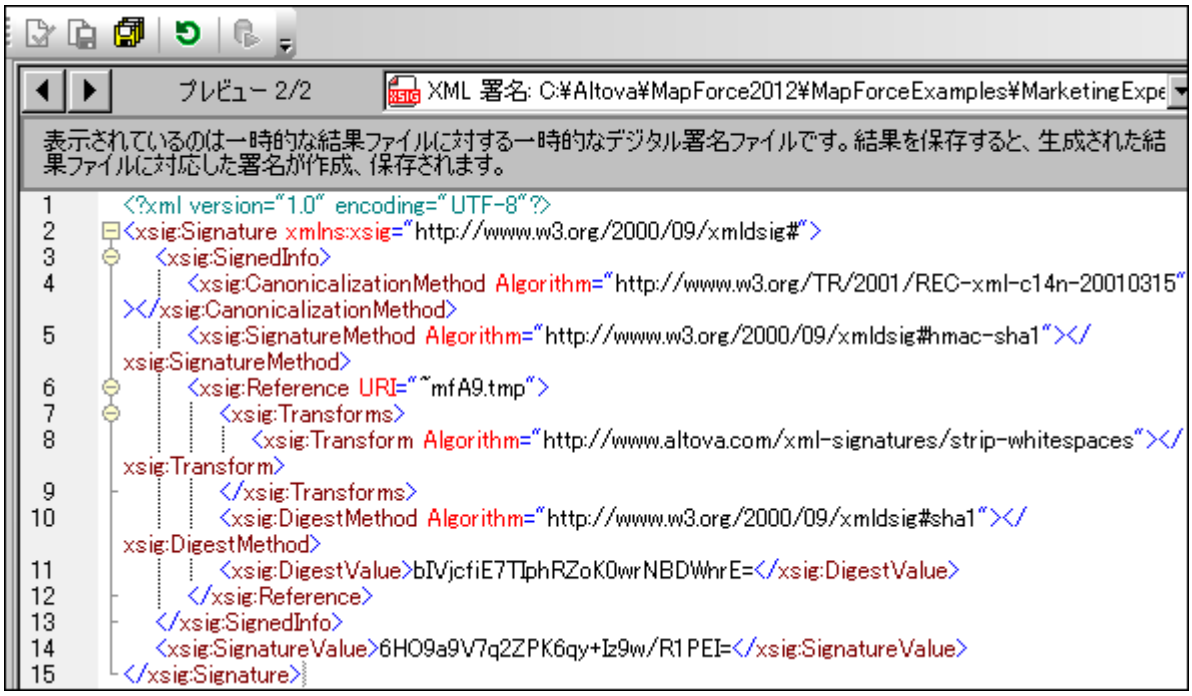
例えば、...¥MapForceExamples フォルダー内の**MarketingExpenses_DetachedSignature.mfd** ファイルを使用します:

1. MarketingExpenses ターゲットコンポーネントをダブルクリックして、署名設定ボタンをクリックします。選択済みのオプションが表示されています。
2. 「OK」をクリックして、ダイアログボックスを閉じます。
3. 出力ボタンをクリックして、マッピングの結果を確認します。

   2 つのファイルがプレビューウィンドウに生成されました。最初のファイル**MarketingExpenses.xml** はそのターゲットコンポーネントのマッピングの結果です。

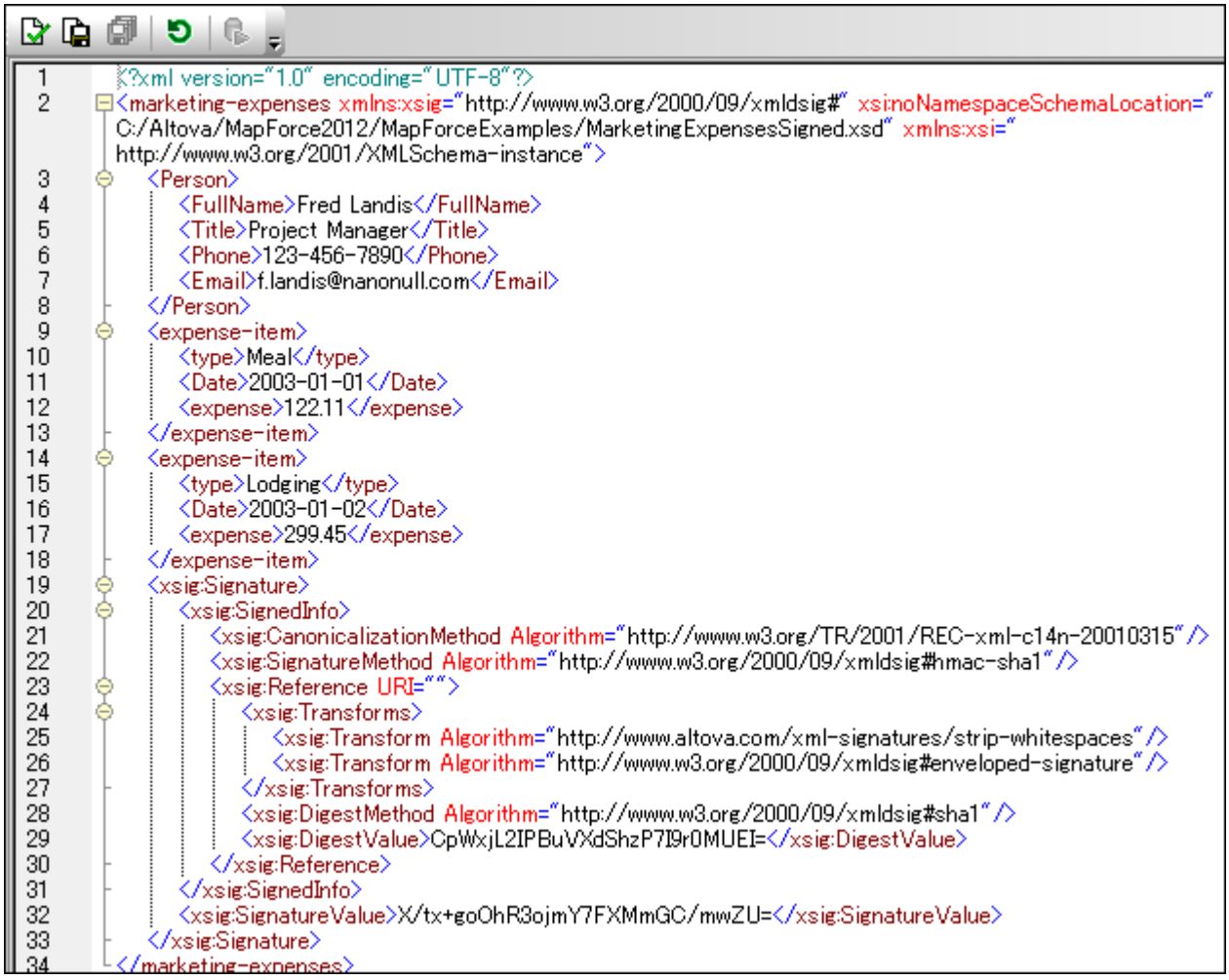


2番目のファイル**MarketingExpenses.xml.xsig** はターゲットコンポーネントにより生成された一時的なデジタル署名ファイルです。



署名ファイルを生成するには「**生成された全ての出力を保存する**」 ツールバーボタンをクリックします。
これで出力ディレクトリに.xml と.xsig ファイルが生成されます。

..¥MapForceExamples フォルダー内に収められている**MarketingExpenses_EnvelopedSignature.mfd** ファイルでは、"Enveloped" 設定で生成される署名の出力を確認することができます。



```
<?xml version="1.0" encoding="UTF-8"?>
<marketing-expenses xmlns:xsig="http://www.w3.org/2000/09/xmldsig#" xsi:noNamespaceSchemaLocation="
C:/Altova/MapForce2012/MapForceExamples/MarketingExpensesSigned.xsd" xmlns:xsi="
http://www.w3.org/2001/XMLSchema-instance">
   <Person>
      <FullName>Fred Landis</FullName>
      <Title>Project Manager</Title>
      <Phone>123-456-7890</Phone>
      <Email>f.landis@nanonull.com</Email>
   </Person>
   <expense-item>
      <type>Meal</type>
      <Date>2003-01-01</Date>
      <expense>122.11</expense>
   </expense-item>
   <expense-item>
      <type>Lodging</type>
      <Date>2003-01-02</Date>
      <expense>299.45</expense>
   </expense-item>
   <xsig:Signature>
      <xsig:SignedInfo>
         <xsig:CanonicalizationMethod Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-20010315" />
         <xsig:SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#hmac-sha1" />
         <xsig:Reference URI="">
            <xsig:Transforms>
               <xsig:Transform Algorithm="http://www.altova.com/xml-signatures/strip-whitespaces" />
               <xsig:Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature" />
            </xsig:Transforms>
            <xsig:DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
            <xsig:DigestValue>CpWxjL2IPBuVXdShzP7I9r0MUEI=</xsig:DigestValue>
         </xsig:Reference>
      </xsig:SignedInfo>
      <xsig:SignatureValue>X/tx+goOhR3ojmY7FXMmGC/mwZU=</xsig:SignatureValue>
   </xsig:Signature>
</marketing-expenses>
```

XML ドキュメントの妥当性

XML 署名が XML ドキュメント内に埋め込まれている場合、`http://www.w3.org/2000/09/xmldsig#` 名前空間内の `Signature` 要素が XML ドキュメントに追加されます。スキーマに従うかたちでドキュメントの妥当性を保つには、適切な要素宣言をスキーマ内に含める必要があります。MapForce では Enveloped オプションを使用した署名の埋込みが行われます:

- *Enveloped:* 要素がルート（またはドキュメント）要素以下にある最後の子要素として、`Signature` 要素が作成されます。

  XML ドキュメントのスキーマに変更を加えたくない場合、"Detached" オプションを使用することで XML 署名を独立した外部ファイルとして作成することができます。

Enveloped 署名の `Signature` 要素を許可するために使用される XML スキーマの抜粋を以下に示します。これらのサンプルをガイドとして、使用中のスキーマを修正することができます。

2 つのうちの最初のサンプルでは、XML 署名スキーマがユーザーのスキーマにインポートされます。XML 署名スキーマは以下のアドレスで見つけることができます: http://www.w3.org/TR/xmldsig-core/xmldsig-core-schema.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
```

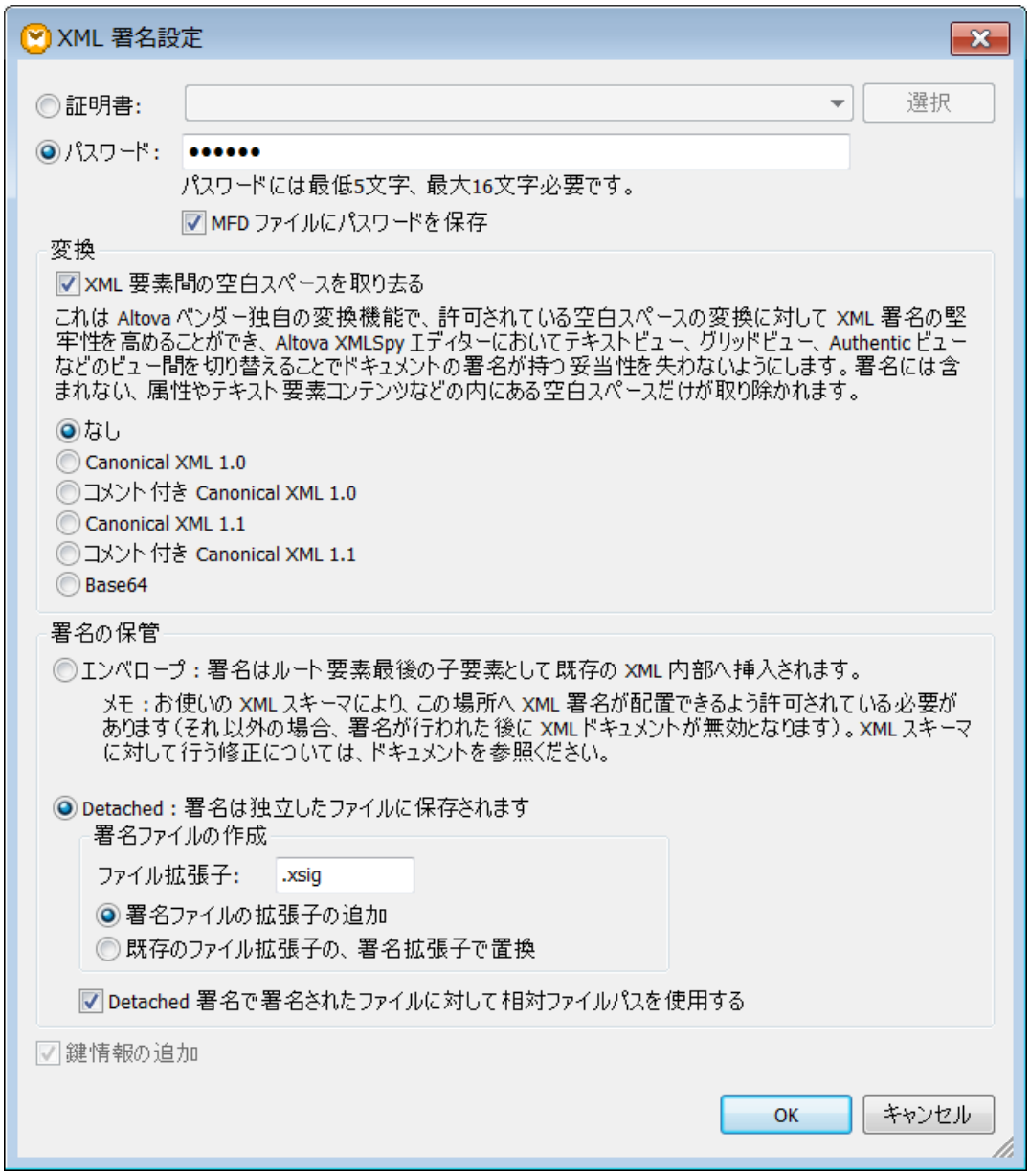
```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
           xmlns:xsig="http://www.w3.org/2000/09/xmldsig#"
           elementFormDefault="qualified"
           attributeFormDefault="unqualified">
   <xs:import namespace="http://www.w3.org/2000/09/xmldsig#"
              schemaLocation="http://www.w3.org/TR/xmldsig-core/xmldsig-core-
schema.xsd"/>
   <xs:element name="Root">
      <xs:complexType>
         <xs:sequence>
            <xs:element ref="FirstChildOfRoot"/>
            <xs:element ref="SecondChildOfRoot" minOccurs="0"/>
            <xs:element ref="ThirdChildOfRoot" minOccurs="0"/>
            <xs:element ref="xsig:Signature" minOccurs="0"/>
         </xs:sequence>
      </xs:complexType>
   </xs:element>
   ...
</xs:schema>
```

(以下にある) 2 番目のオプションでは、他の名前空間にある要素全てにマッチするような、全般的なワイルドカード要素が追加されます。processContents 属性を lax にセットすることで、検証の際にこの要素をスキップするようになります—(マッチする要素宣言が見つからないため)。結果として、ユーザーは XML 署名スキーマを参照する必要がなくなります。このオプションの欠点は、XML ドキュメントの妥当性を損なうこと無く、(Signature 要素以外の) 全ての要素を特定の場所に追加することができてしまうという点です。

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
           elementFormDefault="qualified"
           attributeFormDefault="unqualified">
   <xs:element name="Root">
      <xs:complexType>
         <xs:sequence>
            <xs:element ref="selection"/>
            <xs:element ref="newsitems" minOccurs="0"/>
            <xs:element ref="team" minOccurs="0"/>
            <xs:any namespace="##other" minOccurs="0" processContents="lax"/>
         </xs:sequence>
      </xs:complexType>
   </xs:element>
   ...
</xs:schema>
```

6.1.12.1 XML 署名設定

署名に関する設定はコンポーネント設定ダイアログボックスにて、個別のコンポーネントに対して保管することができるとともに、MFD ファイルが保存された際に全ての設定が保存されます。



XML 署名設定ダイアログボックス

認証方法: 証明書とパスワード

署名は証明書またはパスワードをベースに作成することができます。使用する方法のラジオボタンを選択してください:

- *証明書*:
  証明書を使用するには、証明書に秘密鍵が含まれており、アクセスすることができる証明書ストアに証明書が配置されている必要があります。署名は証明書の秘密鍵により生成されます。署名の検証を行うには、証明書（又はその公開鍵バージョン）へアクセスする必要があります。証明書の公開鍵は署名の検証を行うために使用されます。秘密鍵ならびに公開鍵の証明書を選択するには、選択ボタンをクリックして証明書を参照してください。

- *パスワード：*
  5 文字から16文字の長さのパスワードを入力します。このパスワードにより署名が作成され、署名の検証を行うために必要になります。ダイアログボックスの「OK」ボタンは、パスワードの必須要件が満たされたときに有効になります。

- *MFD ファイルにパスワードを保存：*
  有効になっている場合、パスワードフィールドに入力されたパスワードが、人間には読むことのできない難読化されたかたちでMFD ファイル内部に保存されます。このパスワードを使用することで、MFD ファイルへアクセスすることができる人ならば誰でも署名を作成することができるようになります。

変換

XML データが変換され、その結果が署名の作成に使用されます。正規化アルゴリズムを指定することで、署名の計算を行う前に、そのアルゴリズムがファイルのXML データ(SignedInfo コンテンツ)へ適用されます。アルゴリズム間における重要な違いを以下に記します:

- *コメント付きの Canonical XML：*
  コメントが署名の計算に含まれる場合、XML データのコメントに変更を加えることで、検証に失敗するようになります。コメントを署名の計算に含まない場合、ドキュメントの署名が行われた後でも、コメントの追加や変更を行うことで署名が失敗することはありません

  メモ
  「コメント付き...」は"Detached" が選択されている場合のみ選択可能になります。

- *Base64:*
  XML ドキュメントのルート(または ドキュメント)要素は Base64 によりエンコードされていると認識され、バイナリ形式で読み取られます。ルート要素が Base64 でない場合、要素の型により、エラーが返されるか空の要素として読み取られます。

- *無し：*
  変換は行われず、ディスクに保存されているバイナリファイルから読み取られた XML データが、署名の作成に直接使用されます。署名作成後に行われた変更により、署名の検証が失敗するようになります。

  しかし、XML 要素間の空白スペースを取り去るチェックボックスにチェックが入っている場合、全ての空白スペースが取り除かれ、空白スペースに対する変更が無視されるようになります。

  無し オプションと Canonicalization オプションの主な違いは、正規化により XML データストリームが作成され、属性の順序といった違いが正規化されるという点にあります。結果として Canonicalization 変換では属性の順序変更といったものも正規化され検証が成功する一方、変換を行わない場合そのような変更も検証の結果に反映されます(つまり検証は失敗します)。

  デフォルトの正規化は署名が埋め込まれている(Enveloped) 場合だけ行われることに注意してください。そのため、署名ファイルが独立している場合 (Detached)、無し が選択されている場合、または空白スペースを取り去るチェックボックスにチェックが入っていない場合、XML データがそのまま使用されます。

署名の配置

署名はXML ファイルの中に配置することができるほか、独立したファイルとしても作成することができます。以下のオプションが利用可能です:

- *Enveloped:*
  署名要素がルート(ドキュメント)要素直下にある最後の子要素として作成されます。出力 XML を妥当とするためには、関連する XML スキーマに署名の定義を行う要素を含める必要があります。詳細は、このセクションの上部を参照ください。

- *Detached:*
  XML 署名を独立したファイルとして作成します。この場合、署名ファイルのファイル拡張子を指定し、(i) XML ファイルに拡張子を追加する(例: test.xml.xsig)か、(ii) 拡張子で XML ファイルの XML 拡張子を置き換える(例: test.xsig)かを選択することができます。署名ファイル内部で指定される XML ファイルへのパスを相対パスとするか、絶対パスとするかも選択することができます。

メモ XML スキーマ(.xsd) ファイルならびにXBRL ファイルに対するXML 署名は外部署名ファイルとしてしか作成することができません。W SDL ファイルの場合、外部ファイルならびにW SDL ファイル内部へ署名を埋め込む(enveloped) ことができます。

メモ XML 署名が独立したファイルとして作成された場合、署名ファイル内部で行われる参照によりXML ファイルと署名ファイルの関連付けが行われます。そのため、署名が外部ファイルに収められている状態の検証は、XML ファイルがアクティブな状態ではなく、署名ファイルがアクティブな状態で行われる必要があります。

鍵情報を追加

署名が証明書をベースとしている場合、鍵情報を追加 オプションを選択することができるようになります。

鍵情報を追加オプションが有効になっている場合、公開鍵情報が署名内部に配置され、有効になっていない場合、鍵情報は署名に含められません。鍵情報を追加することにより、証明書(ならびにその中に含まれる公開鍵情報) は、検証処理にて必要となりません(鍵情報は既に署名内に収められているため) 。

署名設定 – 無効の場合

署名設定が無効の場合、MapForce はデジタル署名を作成することができません。以下の場合、署名設定が無効となります:

- 選択された証明書へアクセスすることができないか、XML ドキュメントの署名に使用することができない、または
- 「MFD ファイルにパスワードを保存」オプションにチェックされておらず、パスワードが設定されていない

出力ボタンをクリックすると、以下のメッセージが表示されます
パスワードの場合:

> コンポーネント "MarketingExpressSigned" の出力を署名するためのパスワードを指定してください

証明書の場合:

> コンポーネント "MarketingExpressSigned" の出力を署名するために使用する証明書が含まれているストアを選択してください

パスワードまたは証明書が選択されていない場合、処理が停止するか、署名なしで出力が行われます。この振る舞いはコンポーネント設定ダイアログボックスの処理を中止または署名なしで処理を続けるラジオボタンにより指定することができます。

コマンドラインからマッピングが実行された場合、ダイアログボックスは表示されません。マッピングの処理は中止されるか、署名無しで継続されることになります。

6.2 MapForce とデータベース

Altova W eb サイト: データベースマッピング

MapForce 2021 ではデータベースから、XML、フラットファイル、と他のデータベースフォーマットへのマッピングがサポートされています。MapForce Enterprise エディションでは、追加してデータベースをEDI フォーマット、Excel 2007+、JSON、XBRL とW eb サービスへのマッピングがサポートされます。

次のデータベースはサポートされています。各データベースの使用できるルート オブジェクトもリストされています。Altova は他のODBC/ADO データベースへのサポートの向上のための努力をしていますが、以下にリストされたデータベースのみで接続とデータ処理のテストに成功しています。使用中のAltova アプリケーションが、64 ビット バージョンの場合、接続されているデータベース特定の64 ビット データベース ドライバーにアクセスしてください。Altova アプリケーションはIBM iSeries データベースのロジカル ファイルをサポートし、ロジカル ファイルを表示します。

データベース	メモ
Firebird 2.5.4, 3.0	
IBM DB2 8.x, 9.1, 9.5, 9.7, 10.1, 10.5	
IBM Db2 for i 6.1, 7.1, 7.2, 7.3, 7.4	ロジカルファイルはサポートされ、ビューとして表示されています。
IBM Informix 11.70, 12.10, 14.10	Informix はADO、JDBC およびODBC を介した接続をサポートします。実装は、すべてのコード生成言語による大きなオブジェクト データ型をサポートしません。MapForce は、(コード生成中) これらのデータ型が使用された場合エラーメッセージを表示します。
MariaDB 10.2, 10.3, 10.4, 10.5	
Microsoft Access 2003, 2007, 2010, 2013, 2016, 2019	ドキュメントの作成時 (2019 年9 月初旬) Access 2019 のために使用可能な Microsoft Access Runtime が存在しませんでした。Microsoft Access 2016 Runtime がインストールされており、データベースが「大量」のデータ型を使用しない場合のみ Altova 製品から Access 2019 にデータベース接続することができます。
Microsoft Azure SQL Database	SQL Server 2016 codebase
Microsoft SQL Server 2005, 2008, 2012, 2014, 2016, 2017, 2019 Microsoft SQL Server on Linux	
MySQL 5.0, 5.1, 5.5, 5.6, 5.7, 8.0	
Oracle 9i, 10g, 11g, 12c, 18, 19	
PostgreSQL 8.0, 8.1, 8.2, 8.3, 9.0.10, 9.1.6, 9.2.1, 9.4, 9.5, 9.6, 10, 11, 12.1	PostgreSQL 接続は、ODBC またはJDBC などのインターフェイス(ドライバ)によりネイティブとドライバーベースの接続でサポートされています。ネイティブな接続はドライバを必要としません。
Progress OpenEdge 11.6	
SQLite 3.33.0	SQLite 接続は、ネイティブとしてサポートされます。SQLite データベースファイルへの直接の接続です。

(C) 2015-2021 Altova GmbH

データベース	メモ
Sybase ASE 15, 16	
Teradata 16	ADO.NET、JDBC、および ODBC を介した接続がサポートされています。 マッピングがデータをデータベースに挿入する場合、データベースにより生成される識別フィールドはサポートされていません。

多種の実行環境内でのデータベースマッピング

マッピングからプログラムコードを生成する場合、または MapForce Server 実行可能ファイルにマッピングをコンパイルする場合、FlowForce Server にマッピングをデプロイする場合、生成されたファイルと共に保存されたデータベース接続の詳細は、対応するドライバーにより適用、または選択されたターゲット環境によりサポートされています。例えば、マッピング変換言語が Java に設定されている場合、ADO 接続は、マッピングから Java コードが生成されると、JDBC に変換されます。

MapForce 以外の環境でマッピングが実行されると、データベース接続の詳細がマッピングを実行するマシンに適合していることを確認してください（例えば、データベースドライバーがインストールされている、データベースのパスが正確である、データベースサーバーへのアクセスがあるなど）。

データベースへの接続の種類の一部は、下のテーブルで示されているように、ターゲット環境の一部でサポートされていません。

接続の型/実行環境	C#	C++	Java	Windows 上の MapForce Server	Linux/Mac 上の MapForce Server
ADO	ADO ブリッジ	現状のまま	JDBC に変換されます	現状のまま	JDBC に変換されます
ADO.NET	現状のまま	ユーザー定義	JDBC に変換されます	現状のまま	JDBC に変換されます
JDBC	ユーザー定義	ユーザー定義	現状のまま	現状のまま	現状のまま
ODBC	ODBC ブリッジ	ODBC ブリッジ	JDBC に変換されます	現状のまま	JDBC に変換されます
ネイティブ PostgreSQL	サポート無し	サポート無し	サポート無し	現状のまま	現状のまま
ネイティブ SQLite	サポート無し	サポート無し	サポート無し	現状のまま	現状のまま

テーブル内のレジェンド:

- "現状のまま" は、データベース接続 型（例えば、JDBC、ODBC）が、MapForce で定義されたままという意味です。
- "JDBC に変換されます" は、データベース接続が、JDBC のようなデータベース接続 URL に変換されるという意味です。
- "ADO ブリッジ"、または "ODBC ブリッジ" は、接続文字列は、MapForce 内で定義されたままですが、生成されるコードは、ADO ブリッジ（、または ODBC ブリッジ）としての役割を果たす適切なクラスを使用するという意味です。例えば、**System.Data.OleDb.OleDbConnection**、または **System.Data.Odbc.OdbcConnection** 等があげられます。

- "ユーザー定義" は、生成されたコード内で接続が作動するためには、手動で接続の情報をデータベースコンポーネント設定 ダイアログボックスに入力するという意味です。場合により異なりますが、接続の詳細は **ADO/OLE-DB-specific settings** 、または **JDBC-specific settings** に入力される必要があります。

次も参照してください:

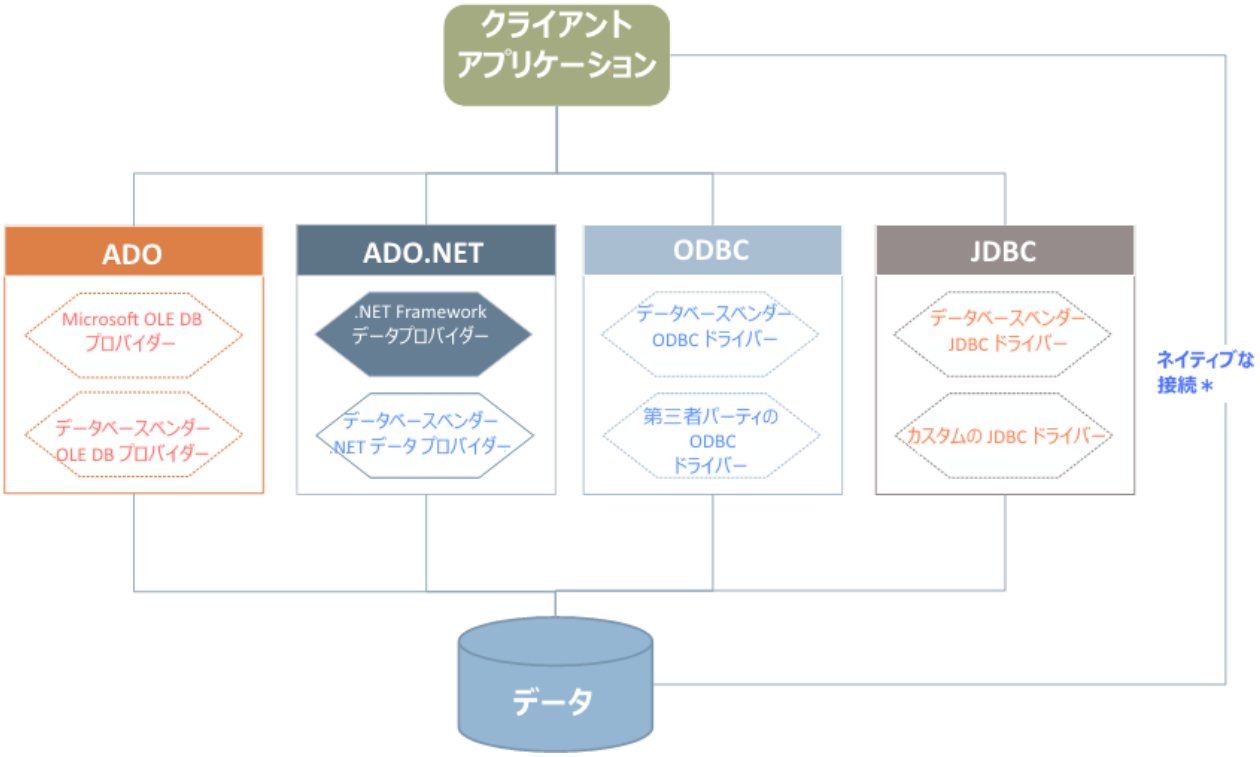
- サーバー実行のためにマッピングを準備する
- マッピングを MapForce Server 実行可能ファイルにコンパイルする
- FlowForce Server にマッピングをデプロイする
- コード ジェネレーター

6.2.1 データソースへの接続

最も簡単なケースとして、データベースは、Microsoft Access、または、SQLite データベースファイルなどのローカルファイルが挙げられます。高度なケースの例としては、データベースは、アプリケーションが接続し、データを消費するオペレーティングシステムと同じオペレーティングシステムを使用するとは限らない、リモート、または、ネットワーク データベースサーバーに存在します。例えば、MapForce が、Windows で作動し、データ(例えばMySQL) へアクセスするデータベースがLinux マシーンで作動することは可能です。

多種のデータベース型とリモートおよびローカルで作業するには、MapForce は、データ接続インターフェイスとオペレーティングシステム内で使用することのできる、または、定期的にメジャーなデータベースベンダーによりリリースされるデータベースドライバーに従います。常に進化するデータベース技術の景観の中、このアプローチは、クロスプラットフォームに対する柔軟性および相互運用性に対応します。すなわち、MapForce により、以下のデータアクセス技術により主要なデータベースにアクセスすることができます

次の図は、最も簡単な方法で、(ジェネリックなクライアントアプリケーションとして表示されている) MapForce と(データベースサーバー、または、データベースファイルであることができる) データ保管の間で使用することのできるデータ接続のオプションを示しています。



** 直接なネイティブな接続 は、SQLite とPostgreSQL データベースのためにサポートされています。使用中のシステムにインストールするために追加ドライバーは必要ではありません。*

(C) 2015-2021 Altova GmbH

上の図で示されるように、MapForce は、主なデータベース型に次のアクセス技術を使用してアクセスすることができます:

- 基となる OLE DB (Object Linking and Embedding, Database) プロバイダーを使用する ADO (Microsoft® ActiveX® Data Objects)
- ADO.NET (データとのインタアクションを有効化する Microsoft .NET Framework 内で使用することのできるライブラリのセット)
- JDBC (Java データベース接続)
- ODBC (Open データベース接続)

ADO.NET の一部はサポートされていない、または、サポートが制限されている場合があります。次を参照してください: ADO.NET サポートメモ。

選択するデータベースへの接続のインターフェイスは、既存のソフトウェアのインフラストラクチャによります。データのアクセス技術と、接続するデータベースと密に統合することのできる、データベースドライバーを通常選択します。例えば、Microsoft Access 2013 データベースに接続するには、Microsoft Office Access データベース Engine OLE DB プロバイダーなどのネイティブなプロバイダを使用する ADO 接続文字列を作成します。に接続するには、他方、最新の JDBC、ODBC、または ADO.NET インターフェイスを Oracle W eb サイトからダウンロードします。

W indows 製品 (Microsoft Access または SQL Server など) のためのドライバーは、W indows オペレーティングシステムで使用可能ですが、他のデータベースの種類では使用することができない可能性があります。メジャーなデータベースベンダーは、対応するデータベースに ADO、ADO.NET、ODBC、または JDBC の組み合わせからクロスプラットフォームアクセスをすることができる、一般使用可能なデータベースクライアントソフトウェアとドライバーを定期的にリリースします。更に、上記の技術のためのサードパーティーのドライバーを使用することができます。多くの場合は、1つ以上の方法で必要なデータベースへ使用中のオペレーティングシステムから接続することができます、この結果、MapForce からも接続することができます。使用可能な機能、パフォーマンスパラメーター、と既知の問題はデータアクセス技術および使用されるドライバーにより異なります。

6.2.1.1 接続ウィザードの開始

MapForce にはデータソースをセットアップするために必要とするステップを導くウィザードが搭載されています。

ウィザードのステップを開始する前に、データベースの一部の種類は、データベースドライバーまたはデータベースクライアントソフトウェアなどの複数のデータベースの必須条件を個別にインストールまたは構成する必要があることご注意してください。これらは、通常各データベースのベンダーにより提供され、特定の W indows バージョン用に調整されたドキュメントを含みます。データベースの種類別にグループ化されたデータベースドライバーのリストに関しては、をデータベースドライバーの概要参照してください。

データベースをソースまたはコンポーネントをマッピングに追加するには以下を行ってください:

- **挿入** メニューから、**データベース**をクリックしください。

データベースを再利用可能なグローバルリソースとして追加するには以下を行ってください:

1. **ツール** メニューから、**[グローバルリソース]**をクリックします。
2. **「追加」**をクリックして、**データベース**をクリックします。
3. **[データベースの選択]** をクリックします。

データベース型を選択し**「次へ」**をクリックすると データベースの種類、技術（ADO, ADO.NET, ODBC, JDBC）と使用されるドライバーにより異なる命令が画面に表示されます。

例えば、各データベースの種類に適用できるサンプルは[データベース接続のサンプル](#) のセクションを参照してください。各データベースへのアクセス技術に関しては以下のトピックを参照してください:

- [ADO 接続の設定](#)
- [ADO.NET 接続の設定](#)
- [ODBC 接続の設定](#)
- [JDBC 接続の設定](#)

(C) 2015-2021 Altova GmbH

6.2.1.2 データベースドライバー概要

以下のテーブルは、特定のデータベースに特定のデータアクセス技術を使用して接続するために一般的に使用されるデータベースのドライバーのリストです。このリストは、完全または規範的なものではありません。以下に表示されているドライバに加え他のネイティブまたはサードパーティー代替を使用することができます。

Windows オペレーティングシステムの複数のデータベースドライバーが既に使用されている場合がありますが、代替ドライバーをダウンロードする必要があります。データベースの一部に対して、オペレーティングシステムに付随するドライバーよりもデータベースベンダーより提供される最新のドライバーのパフォーマンスの方が良い場合があります。

データベースのベンダーは、個別にダウンロードすることのできるパッケージ、または、バンドルされたデータベースクライアントソフトウェアを提供します。後者の場合、データベースクライアントソフトウェアは、通常、データベースの管理と接続、およびデータベースクライアントとそのコンポーネントのインストールと構成を簡素化するために管理と構成ユーティリティにより構成されています。

データベースクライアントを正確に構成することは、データベースへの接続を成功させるために必須です。データベースをインストールして、使用する前に、各データベースバージョンおよび各 Windows バージョンにより通常異なるデータベースのインストールおよび構成の手順を確認することが奨励されます。

データベースの種類に関して各データアクセステクノロジーの機能と制限を理解するには、特定のデータベース製品のドキュメントを参照し、特定の環境に対しての接続をテストします。よく発生する接続問題を回避するために、以下に注意してください。:

- ADO.NET の一部のプロバイダーはサポートされない、または、サポートが制限されています。次を参照してください: ADO.NET サポートメモ。
- データベースドライバーをインストールする際、Altova アプリケーション(32-ビット または64-ビット)などの同じプラットフォームが使用されることが奨励されます。例えば、32ビットのAltova アプリケーションが64-ビットオペレーティングシステムで使用される場合、が使用される場合、32-ビットのドライバーをインストールし、32-ビットのドライバーを使用して、データベースの接続をセットアップします。詳細に関しては、使用可能な ODBC ドライバーの表示 を参照してください。
- ODBC データソースをセットアップする際、データソース名 (DSN) をユーザー *DSN* の代わりに *システム DSN* として作成することが奨励されます。詳細に関しては、使用可能な ODBC ドライバーの表示 を参照してください。
-
- ターゲットデータベースがODBC を使用したMySQL またはMariaDB の場合、オプションは「**影響された行の代わりに一致した行を返す**」がMySQL ODBC コネクタの**カーソル/結果** タブで有効化されている必要があります。または、データベース接続ウィザードを使用して接続文字列を手動で 入力する場合`Option=2` を接続文字列に追加します: `Dsn=mydsn;Option=2;`
- When setting
- JDBC データソースをセットアップする際、JRE (Java Runtime Environment) 、Java Development Kit (JDK) がインストールされている、またオペレーティングシステムのCLASSPATH 環境変数が構成されていることを確認してください。詳細に関しては、JDBC 接続のセットアップを参照してください。
- データベースベンダーからインストールするドライバーおよびデータベースクライアントソフトウェアのインストールの手順とサポートの詳細は、インストールパッケージのドキュメントを参照してください。公式なまたはサードパーティデータベースドライバーの場合、特定のオペレーティングシステムでの特定のドライバーに適用することのできる最も包括的な情報と構成手順は、通常ドライバーインストールパッケージの一部です。

データベース	インターフェイス	ドライバー
Firebird	ADO.NET	Firebird ADO.NET Data Provider (https://www.firebirdsql.org/en/additional-downloads/)
	JDBC	Firebird JDBC driver (https://www.firebirdsql.org/en/jdbc-driver/)

データベース	インターフェイス	ドライバー
	ODBC	Firebird ODBC driver (https://www.firebirdsql.org/en/odbc-driver/)
IBM DB2	ADO	IBM OLE DB Provider for DB2
	ADO.NET	IBM Data Server Provider for .NET
	JDBC	IBM Data Server Driver for JDBC and SQLJ
	ODBC	IBM DB2 ODBC Driver
IBM DB2 for i	ADO	• IBM DB2 for i5/OS IBMDA400 OLE DB Provider • IBM DB2 for i5/OS IBMDARLA OLE DB Provider • IBM DB2 for i5/OS IBMDASQL OLE DB Provider
	ADO.NET	.NET Framework Data Provider for IBM i
	JDBC	IBM Toolbox for Java JDBC Driver
	ODBC	iSeries Access ODBC Driver
IBM Informix	ADO	IBM Informix OLE DB Provider
	JDBC	IBM Informix JDBC Driver
	ODBC	IBM Informix ODBC Driver
Microsoft Access	ADO	• Microsoft Jet OLE DB Provider • Microsoft Access Database Engine OLE DB Provider
	ADO.NET	.NET Framework Data Provider for OLE DB
	ODBC	• Microsoft Access Driver
MariaDB	ADO.NET	MariaDB のための専用の.NET コネクタが不足している場合、MySQL のためのコネクタ/NET を使用してください（https://dev.mysql.com/downloads/connector/net/).
	JDBC	MariaDB Connector/J (https://downloads.mariadb.org/)
	ODBC	MariaDB Connector/ODBC (https://downloads.mariadb.org/)
Microsoft SQL Server	ADO	• Microsoft OLE DB Driver for SQL Server (MSOLEDBSQL) • Microsoft OLE DB Provider for SQL Server (SQLOLEDB) • SQL Server Native Client (SQLNCLI)
	ADO.NET	• .NET Framework Data Provider for SQL Server • .NET Framework Data Provider for OLE DB
	JDBC	• Microsoft JDBC Driver for SQL Server (https://docs.microsoft.com/en-us/sql/connect/jdbc/microsoft-jdbc-driver-for-sql-server)
	ODBC	• ODBC Driver for Microsoft SQL Server (https://docs.microsoft.com/en-us/SQL/connect/odbc/download-odbc-driver-for-sql-server)

(C) 2015-2021 Altova GmbH

データベース	インターフェイス	ドライバー
MySQL	ADO.NET	• Connector/NET (https://dev.mysql.com/downloads/connector/net/)
	JDBC	Connector/J (https://dev.mysql.com/downloads/connector/j/)
	ODBC	Connector/ODBC (https://dev.mysql.com/downloads/connector/odbc/)
Oracle	ADO	• Oracle Provider for OLE DB • Microsoft OLE DB Provider for Oracle
	ADO.NET	Oracle Data Provider for .NET (http://www.oracle.com/technetwork/topics/dotnet/index-085163.html)
	JDBC	• JDBC Thin Driver • JDBC Oracle Call Interface (OCI) Driver これらのドライバーは、通常、データベースクライアントのインストール中にインストールされます。（Thin ドライバーではない） OCI ドライバーを使用する接続は、Oracle XML DB のコンポーネントを使用しています。
	ODBC	• Microsoft ODBC for Oracle • Oracle ODBC Driver（通常、Oracle データベースクライアントのインストール中にインストールされます）
PostgreSQL	JDBC	PostgreSQL JDBC Driver (https://jdbc.postgresql.org/download.html)
	ODBC	psqlODBC (https://odbc.postgresql.org/)
	Native Connection	使用可能。ネイティブな接続を使用する場合、ドライバーをインストールする必要はありません。
Progress OpenEdge	JDBC	JDBC Connector (https://www.progress.com/jdbc/openedge)
	ODBC	ODBC Connector (https://www.progress.com/odbc/openedge)
SQLite	Native Connection	使用可能。ネイティブな接続を使用する場合、ドライバーをインストールする必要はありません。
Sybase	ADO	Sybase ASE OLE DB Provider
	JDBC	jConnect™ for JDBC
	ODBC	Sybase ASE ODBC Driver
Teradata	ADO.NET	.NET Data Provider for Teradata (https://downloads.teradata.com/download/connectivity/net-data-provider-for-teradata)
	JDBC	Teradata JDBC Driver (https://downloads.teradata.com/download/connectivity/jdbc-driver)
	ODBC	Teradata ODBC Driver for Windows (https://downloads.teradata.com/download/connectivity/odbc-driver/windows)

6.2.1.3 ADO 接続の設定

Microsoft ActiveX Data Objects (ADO) はOLE DB を介して多種のデータソースにデータベースアクセステクノロジーです。OLE DB はODBC またはJDBC への代替インターフェイスです。COM (コンポーネント オブジェクト モデル) 環境内での統一されたアクセスを提供します。ADO は新規のADO.NET 前のバージョンであり、他のデータソースのためにも使用可能ですがMicrosoft Access or SQL Server などのMicrosoft ネイティブなデータベースにアクセスを可能にする方法です。

需要な点は、複数のADO プロバイダー間から選択することができ、一部はダウンロードされワークステーションに使用する前にインストールされる必要があります。SQL Server への接続のために以下のADO プロバイダーを使用することができます:

- Microsoft OLE DB *Driver* for SQL Server (MSOLEDBSQL)
- Microsoft OLE DB *Provider* for SQL Server (SQLOLEDB)
- SQL Server Native Client (SQLNCLI)

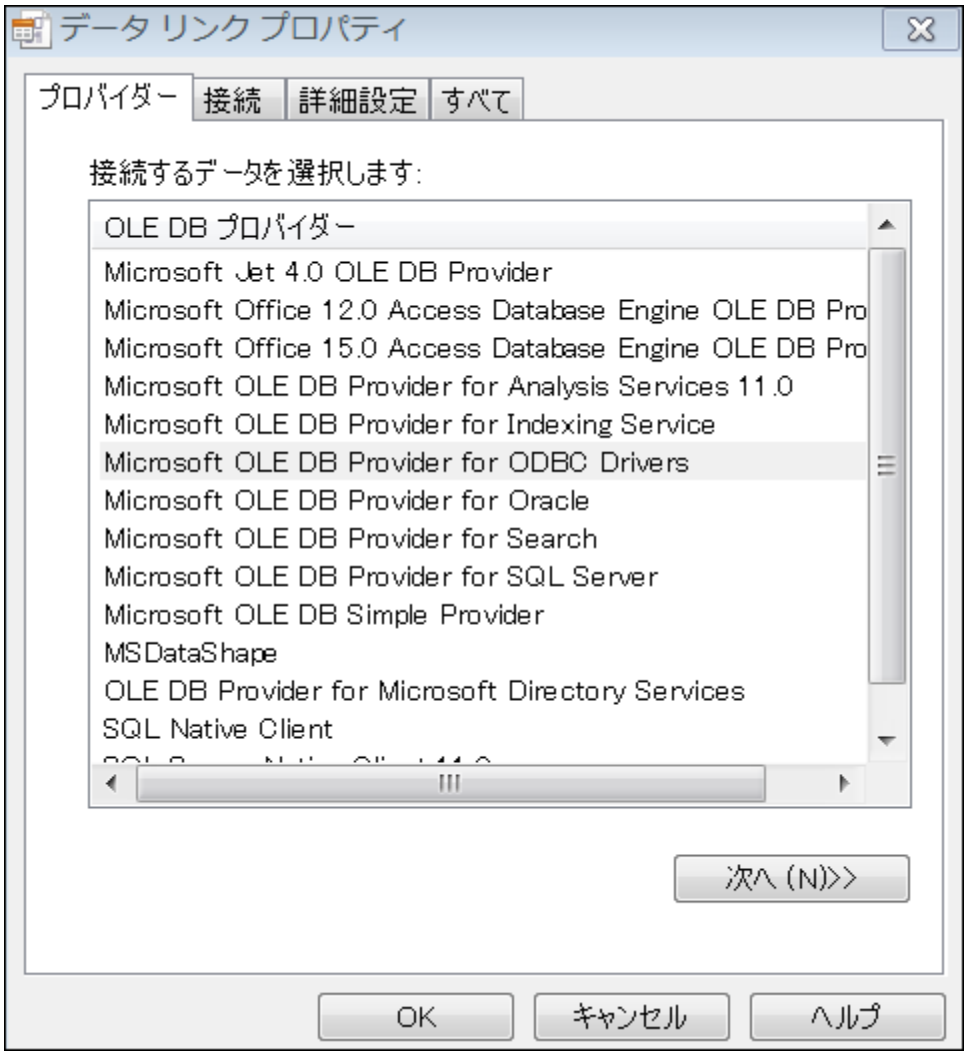
上記でリストされたプロバイダーで奨励されるプロバイダーはMSOLEDBSQL です。https://docs.microsoft.com/en-us/sql/connect/oledb/download-oledb-driver-for-sql-server?view=sql-server-ver15 からダウンロード可能です。MapForce (32-ビット または64-ビット)のプラットフォームに一致する必要があることにご注意してください。SQLOLEDB とSQLNCLI プロバイダーは非奨励であると考えられています。

Microsoft OLE DB Provider for SQL Server (SQLOLEDB) は共通テーブル式 (CTE) とネストされたSELECT ステートメントなどの複雑なクエリのパラメーターバンディングに関する問題があることが知られています。

ADO 接続の設定:

1. データベース接続ウィザードを開始します。
2. 「**ADO 接続**」をクリックします。

3. 構築] をクリックします。



4. 接続するデータプロバイダーを選択します。下のテーブルは一般的なシナリオをリストしています。

このデータベースに接続する方法...	このプロバイダーを使用する...
Microsoft Access	• **Microsoft Office Access Database Engine OLE DB Provider** (奨励) • **Microsoft Jet OLE DB Provider** **Microsoft Office Access Database Engine OLE DB Provider** がリスト内で使用できない場合 Microsoft Access または Microsoft Access Database Engine Redistributable (https://www.microsoft.com/en-us/download/details.aspx?id=54920) がコンピューターにインストールされていることを確認してください。
SQL Server	• **Microsoft OLE DB Driver for SQL Server (MSOLEDBSQL)** が OLE DB プロバイダーのために奨励されています。リストにこのプロバイダーが表示されるためには、プロバイダーが https://docs.microsoft.com/en-

(C) 2015-2021 Altova GmbH

このデータベースに接続する方法...	このプロバイダーを使用する..
	us/sql/connect/oledb/download-oledb-driver-for-sql-server?view=sql-server-ver15 からダウンロードされインストールされる必要があります。 • **Microsoft OLE DB Provider for SQL Server (OLEDBSQL)** • **SQL Server Native Client (SQLNCLI)**
他のデータベース	データベースに適用可能なプロバイダーを選択してください。 視聴中のデータベースへのOLE DB プロバイダーが使用できない場合 データベースベンダーから必要なドライバーをインストールします(データベースドライバー概要を参照してください)。代わりにADO.NET、ODBC、またはJDBC 接続を設定します。 オペレーティングシステムが必要なデータベースのためにODBC ドライバーを必要とする場合 **Microsoft OLE DB Provider for ODBC Drivers** を使用することもできます。またはODBC 接続を使用してください。

5. プロバイダーを選択後 [次へ] をクリックしてウィザードを完了します。

ウィザードの次のステップは選択されたプロバイダーにより異なります。SQL Server には、データベースサーバーのホスト名、認証メソッド、データベース名、およびデータベースユーザー名とパスワードを提供または選択する必要があります。サンプルに関してはMicrosoft SQL Server (ADO) への接続 セクションを参照ください。Microsoft Access に関しては、データベースファイルへのパスを参照または提供するように問われます。サンプルに関してはMicrosoft Access (ADO) への接続 セクションを参照ください。

初期化プロパティの完全なリスト (接続パラメーター) は接続ダイアログボックスの 全ての タブで使用可能です。これらのプロパティは選択されたプロバイダーにより異なり、接続を可能にするために明示的に設定される必要がある場合があります。以下のセクションはMicrosoft Access とSQL Server データベースのための基本的な初期化プロパティを構成する方法について説明しています。

- SQL Server データリンクプロパティの設定
- Microsoft Access データリンクプロパティの設定

6.2.1.3.1 既存の Microsoft Access データベースへの接続

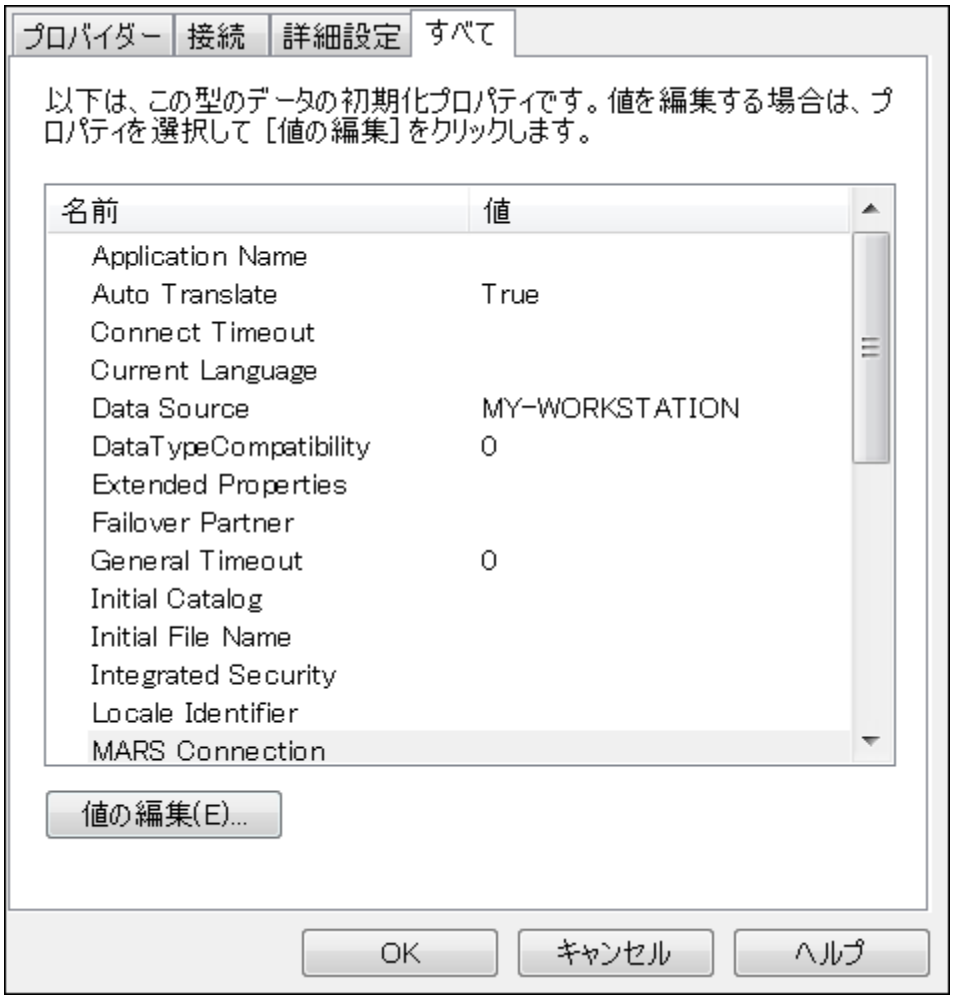
このアプローチは、パスワードで保護されていないMicrosoft Access データベースに接続する場合に適しています。データベースがパスワードにより保護されている場合、Microsoft Access (ADO) への接続 への接続で説明されている通りデータベースパスワードを設定してください。

既存のMicrosoft Access データベースへの接続方法:

1. データベース接続ウィザードを実行します(データベース接続ウィザードの開始を参照)。
2. **Microsoft Access (ADO)** を選択して、「次へ」をクリックします。
3. データベースファイルを参照、または、(相対または絶対)パスを入力します。
4. 「**接続**」をクリックします。

6.2.1.3.2 SQL Server データ リンク プロパティの設定

Microsoft SQL Server データベースにADO を介して接続する場合、次のデータリンク プロパティダイアログボックスの全ての タブ内の次の接続プロパティを設定する必要があります。

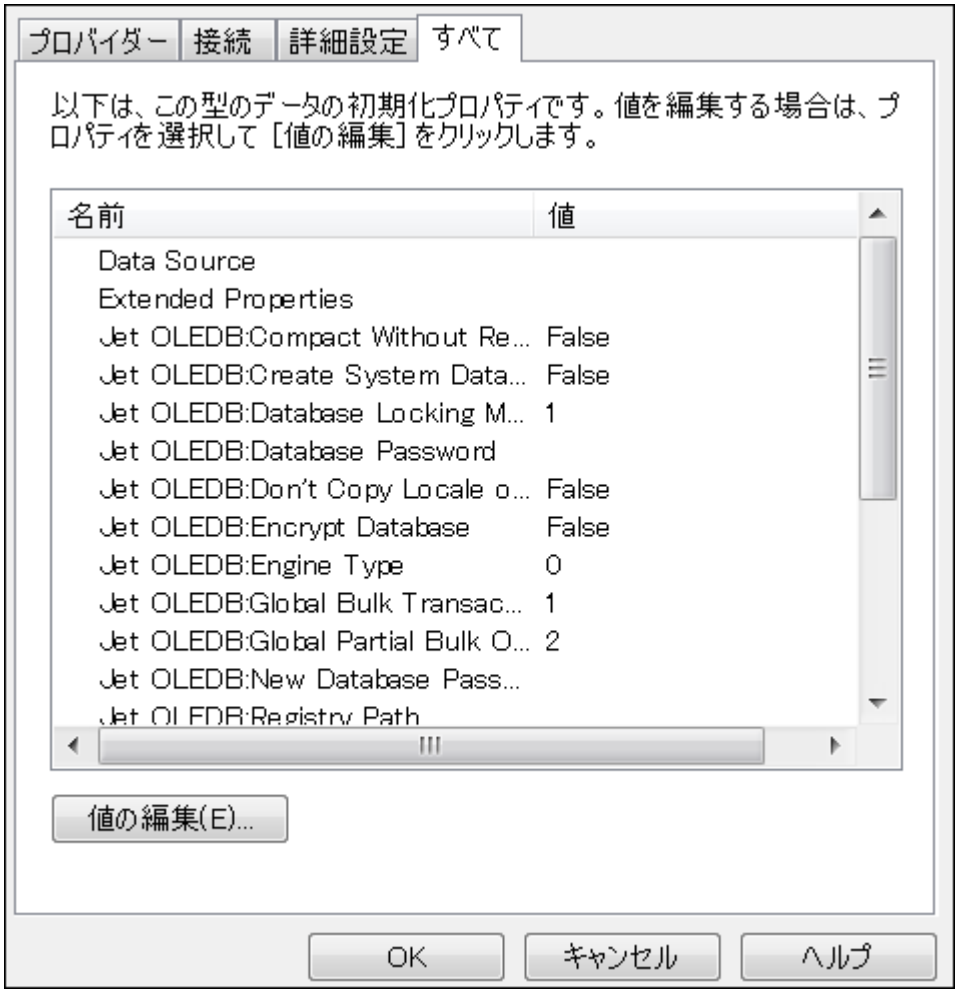


データ リンク プロパティ ダイアログ ボックス

プロパティ	メモ
統合セキュリティ	プロバイダータブで **SQL Server Native Client** がデータプロバイダーに選択された場合は、このプロパティを、空白文字に設定します。
セキュリティ情報を保持	このプロパティを**True** に設定する。

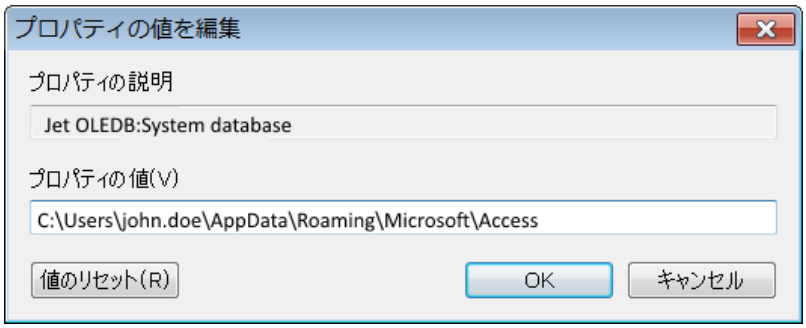
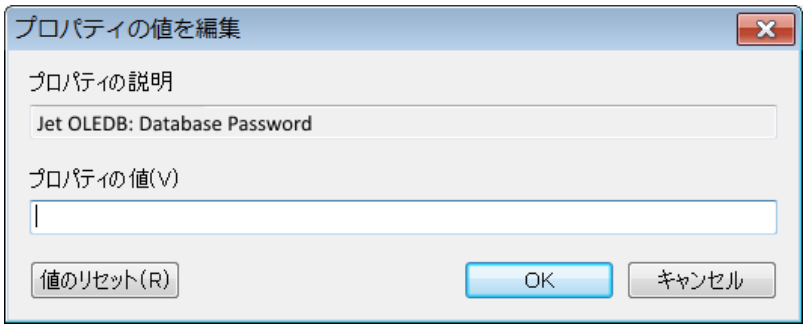
6.2.1.3.3 Microsoft Access データ リンク プロパティの設定

Microsoft Access データベースにADO を介して接続する場合、次のデータリンク プロパティダイアログボックスの全ての タブ内の次の接続プロパティを設定する必要があります。



データ リンク プロパティ ダイアログ ボックス

プロパティ	メモ
データ ソース	このプロパティは、Microsoft Access データベースファイルへのパスを保管します。データベースへの接続問題を回避するために、UNC (Universal Naming Convention 汎用名前付け規則) パスフォーマットの使用が奨励されます。例: `\\anyserver\share$\filepath`
Jet OLEDB: システムデータベース	このプロパティはワークグループ情報ファイルへのパスを保管します。データベースに接続する前に、このプロパティを明示的に設定する必要がある可能性があります。 "ワークグループ情報ファイル" エラーが理由で接続できない場合、ワークグループ情報ファイル (**System.MDW**) をユーザープロファイルに適用できるように設定します (手順は http://support.microsoft.com/kb/305542 を参照してください)、プロパティの値を **System.MDW** ファイルのパスに設定します。

プロパティ	メモ
	
Jet OLEDB: データベース パスワード	データベースがパスワードで保護されている場合は、このプロパティの値をデータベースのパスワードに設定します。 

6.2.1.4 ADO.NET 接続のセットアップ

ADO.NET は、データベースからのデータを含むデータと作業するためにデザインされた Microsoft .NET フレームワーク ライブラリのセットです。MapForce から、データベースに接続するには、ADO.NET、Microsoft .NET フレームワーク 4、または、以降を使用する必要があります。下に表示されているとおり、 .NET プロバイダーを選択して、接続文字列を提供し、データベースに ADO.NET を介して接続します。

.NET データプロバイダーは、コマンドを実行する特定の型のデータソースへの接続を可能にするクラスのコネクションです（例えば、SQL Server または Oracle データベースなど）。すなわち、ADO.NET とは、MapForce などのアプリケーションがデータベースプロバイダーを介してデータベースと作動することを意味します。各 データプロバイダーは特定の型のデータソースと作動するように最適化されています。.NET プロバイダーには以下の2つの種類があります：

1. デフォルトとして Microsoft .NET フレームワークを使用して提供されます。
2. 主要なデータベースベンダーを使用して .NET フレームワークの拡張として提供されます。このような ADO.NET プロバイダーは、個別にインストールされている必要があり、通常それぞれのデータベースベンダーのウェブサイトからダウンロードすることができます。

メモ 特定の ADO.NET プロバイダーは、サポートされていない、または、サポートが制約されています。ADO.NET サポートに関するメモを参照してください。

ADO.NET 接続の作成:

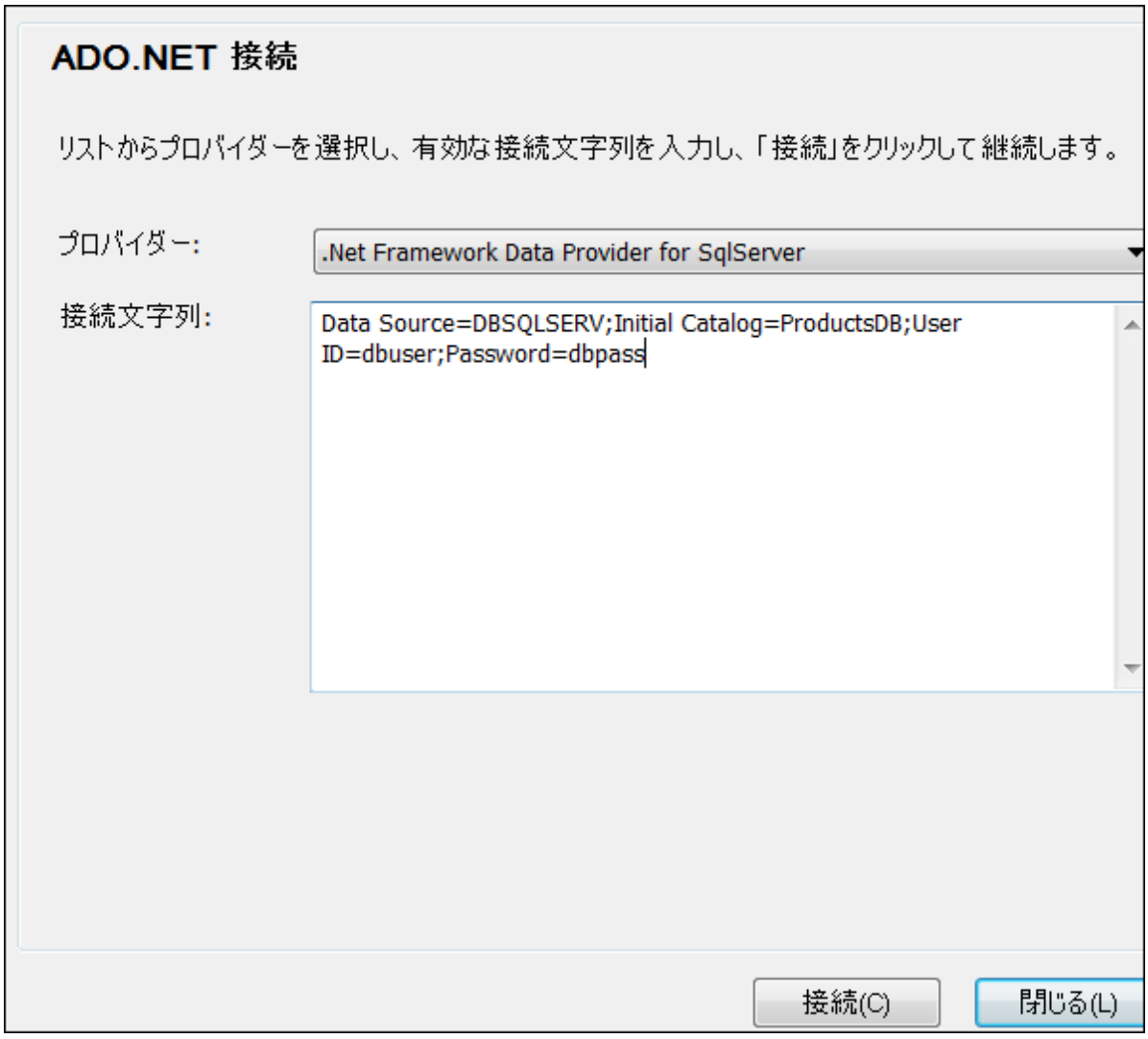
1. データベース接続ウィザードを開始します。
2. 「ADO.NET 接続」をクリックします。

3. .NET データプロバイダーをリストから選択します。

   .NET フレームワークと共に使用することのできるプロバイダーのリストはデフォルトでプロバイダーのリストに表示されます。ベンダーに特有の.NET データプロバイダーは、使用中のシステムに既にインストールされている場合のみ使用することができます。.NET データプロバイダーは、データベースベンダーにより提供される、.msi、または、.exe ファイルを実行して、GAC（Global Assembly Cache）内にインストールされる必要があります。

4. データベース接続文字列を入力します。接続文字列は、データベース接続情報をセミコロンで区切られた接続パラメーターのキー値ペアとして定義します。例えば、`Data Source=DBSQLSERV;Initial Catalog=ProductsDB;User ID=dbuser;password=dbpass` などの文字列は、サーバー `DBSQLSERV` 上で、ユーザー名 `dbuser` とパスワード `dbpass` を使用して、SQL Server データベース `ProductsDB` に接続します。接続文字列を、キー値ペアを直接「接続文字列」ダイアログボックスに入力することにより作成することができます。他の接続オプションは Visual Studio で作成することができます（次を参照: Visual Studio 内で接続文字列を作成する）。

   接続文字列の構文は、プロバイダーリストより選択されたプロバイダーにより異なります。例えば、次を参照してください: サンプル ADO.NET 接続文字列。



5. 「接続」をクリックします。

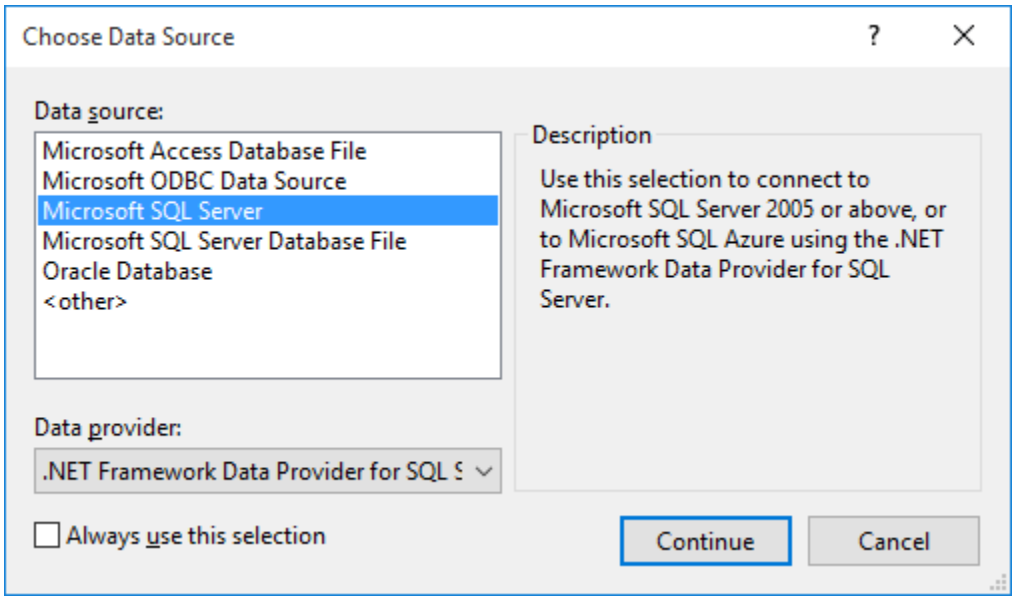
6.2.1.4.1 Visual Studio 内で接続文字列を作成する

ADO.NET を使用してデータソースに接続するには、有効な接続文字列が必要です。次の手順は、Visual Studio から接続文字列を作成する方法を説明しています。

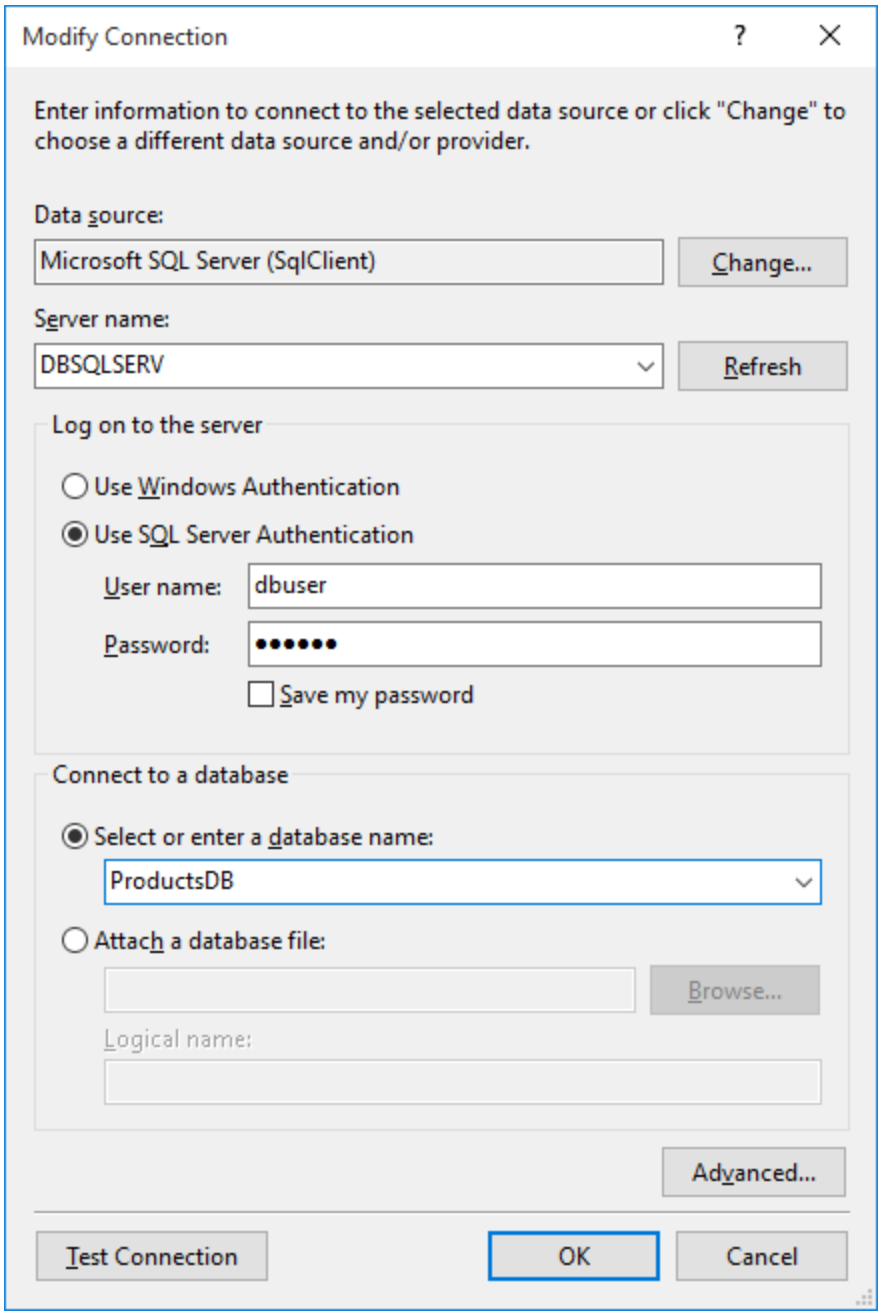
Visual Studio 内で接続文字列を作成する:

1. 「ツール」メニューから、「データベースへの接続」をクリックします。
2. リストからデータソースを選択します(この例の場合、Microsoft SQL Server)。データプロバイダーは、選択に従い自動的に記入されます。

(C) 2015-2021 Altova GmbH



3. 「接続」をクリックします。



4. データベースへのサーバーホスト名、ユーザー名、およびパスワードを入力します。この例の場合、サーバー `DBSQLSERV` 上のデータベース `ProductsDB` にSQL Server 認証を使用して接続します。

5. 「OK」をクリックします。

データベースへの接続に成功した場合、サーバーエクスプローラーウィンドウに表示されます。サーバーエクスプローラーウィンドウを、メニューコマンド「**表示 | サーバーエクスプローラー**」を使用して表示することができます。データベース接続文字列を取得するには、サーバーエクスプローラーウィンドウ内の接続を右クリックし、「**プロパティ**」を選択します。接続文字列が、Visual Studio プロパティウィンドウに表示されます。MapForce の「接続文字列」ボックスに文字列を貼り付ける前に、アスタリスク（＊）文字を実際のパスワードと置き換える必要があることにご注意ください。

(C) 2015-2021 Altova GmbH

6.2.1.4.2 サンプル ADO.NET 接続文字列

ADO.NET 接続をセットアップするには、データベース接続ダイアログボックスから ADO.NET プロバイダーを選択し、接続文字列を入力します。(次も参照してください ADO.NET 接続のセットアップ)。異なるデータベースのサンプル ADO.NET 接続文字列は .NET プロバイダーの下にリストされています。

Teradata のための .NET Data

このプロバイダーは、次の Teradata W eb サイトからダウンロードすることができます (http://downloads.teradata.com/download/connectivity/net-data-provider-for-teradata)。サンプル接続は以下のとおりです:

```
Data Source=ServerAddress;User Id=user;Password=password;
```

IBM i のための .NET Framework データプロバイダー

このプロバイダーは、*IBM i Access Client Solutions - Windows Application Package* の一部としてインストールされます。サンプル接続文字列は以下のとおりです:

```
DataSource=ServerAddress;UserID=user;Password=password;DataCompression=True;
```

詳細に関しては、インストールパッケージ内に含まれる「.NET プロバイダー技術レファレンス ヘルプファイル」を参照してください。

MySQL のための .NET Framework データプロバイダー

このプロバイダーを MySQL ウェブサイトからダウンロードすることができます (https://dev.mysql.com/downloads/connector/net/)。サンプル接続文字列は以下のとおりです:

```
Server=127.0.0.1;Uid=root;Pwd=12345;Database=test;
```

次も参照してください: https://dev.mysql.com/doc/connector-net/en/connector-net-programming-connecting-connection-string.html

SQL Server のための .NET Framework データプロバイダー

サンプル接続文字列は以下のとおりです:

```
Data Source=DBSQLSERV;Initial Catalog=ProductsDB;User ID=dbuser;Password=dbpass
```

次も参照してください: https://msdn.microsoft.com/en-us/library/ms254500(v=vs.110).aspx

.NET Framework 4.0 のための IBM DB2 データプロバイダー 10.1.2

```
Database=PRODUCTS;UID=user;Password=password;Server=localhost:50000;
```

メモ このプロバイダーは、通常 IBM DB2 Data Server Client パッケージと共にインストールされます。IBM DB2 Data Server Client パッケージをインストール後 ADO.NET プロバイダーのリスト内でプロバイダーが不足している場合、次の技術メモを参照してください: http://www-01.ibm.com/support/docview.wss?uid=swg21429586。

次も参照してください:
http://www.ibm.com/support/knowledgecenter/en/SSEPGG_10.1.0/com.ibm.swg.im.dbclient.adonet.ref.doc/doc/DB2ConnectionClassConnectionStringProperty.html

.NET (ODP.NET) のための Oracle データプロバイダー

ODP.NET プロバイダーを含むインストールパッケージを Oracle ウェブサイトからダウンロードすることができます (次を参照してください: http://www.oracle.com/technetwork/topics/dotnet/downloads/index.html). サンプル接続文字列は以下のとおりです:

```
Data Source=DSORCL;User Id=user;Password=password;
```

Oracle への接続(ODBC)内で説明されているように **tnsnames.ora** ファイル内で定義された Oracle サービス名をさす DSORCL がデータソース名の場合。

tnsnames.ora ファイル内のサービス名を構成せず接続する場合は、以下のような文字列を使用してください:

```
Data Source=(DESCRIPTION=(ADDRESS_LIST=(ADDRESS=(PROTOCOL=TCP)(HOST=host)(PORT=port)))
(CONNECT_DATA=(SERVER=DEDICATED)(SERVICE_NAME=MyOracleSID)));User
Id=user;Password=password;
```

次も参照してください: https://docs.oracle.com/cd/B28359_01/win.111/b28375/featConnecting.htm

6.2.1.4.3 ADO.NET サポートに関するメモ

下のテーブルは MapForce 内で現在サポートされていない、または、制限のあるサポートのある既知の ADO.NET データベースドライバーがリストされています。

データベース	ドライバー	サポートノート
全てのデータベース	ODBC のための Net Framework **Data Provider**	制限つきサポート。Microsoft Access 接続の際に既知の問題があります。ODBC への直接の接続を代わりに使用することが奨励されます。
	OleDb のための .Net Framework **Data Provider**	制限つきサポート。Microsoft Access 接続の際に既知の問題があります。ADO への直接の接続を代わりに使用することが奨励されます。
Firebird	**Firebird ADO.NET Data Provider**	制限つきサポート。ODBC または JDBC を代わりに使

データベース	ドライバー	サポートノート
		用することが奨励されます。
Informix	.NET Framework 4.0 のための IBM **Informix Data Provider**	サポートされていません。**DB2 Data Server Provider** を代わりに使用してください。
IBM DB2 for i (iSeries)	i5/OS のための .Net Framework **Data Provider**	サポートされていません。*IBM i Access Client Solutions - Windows Application Package* の一部として提供されている **.Net Framework Data Provider for IBM i** を代わりに使用してください。
Oracle	Oracle のための .Net Framework **Data Provider**	制限つきサポート。.NET Framework と共にこのドライバは提供されていますが、Microsoft により使用を奨励されていません。
PostgreSQL	-	このベンダーのための ADO.NET ドライバはサポートされていません。ネイティブな接続を代わりに使用してください。
Sybase	-	このベンダーのための ADO.NET ドライバはサポートされていません。

6.2.1.5 ODBC 接続の設定

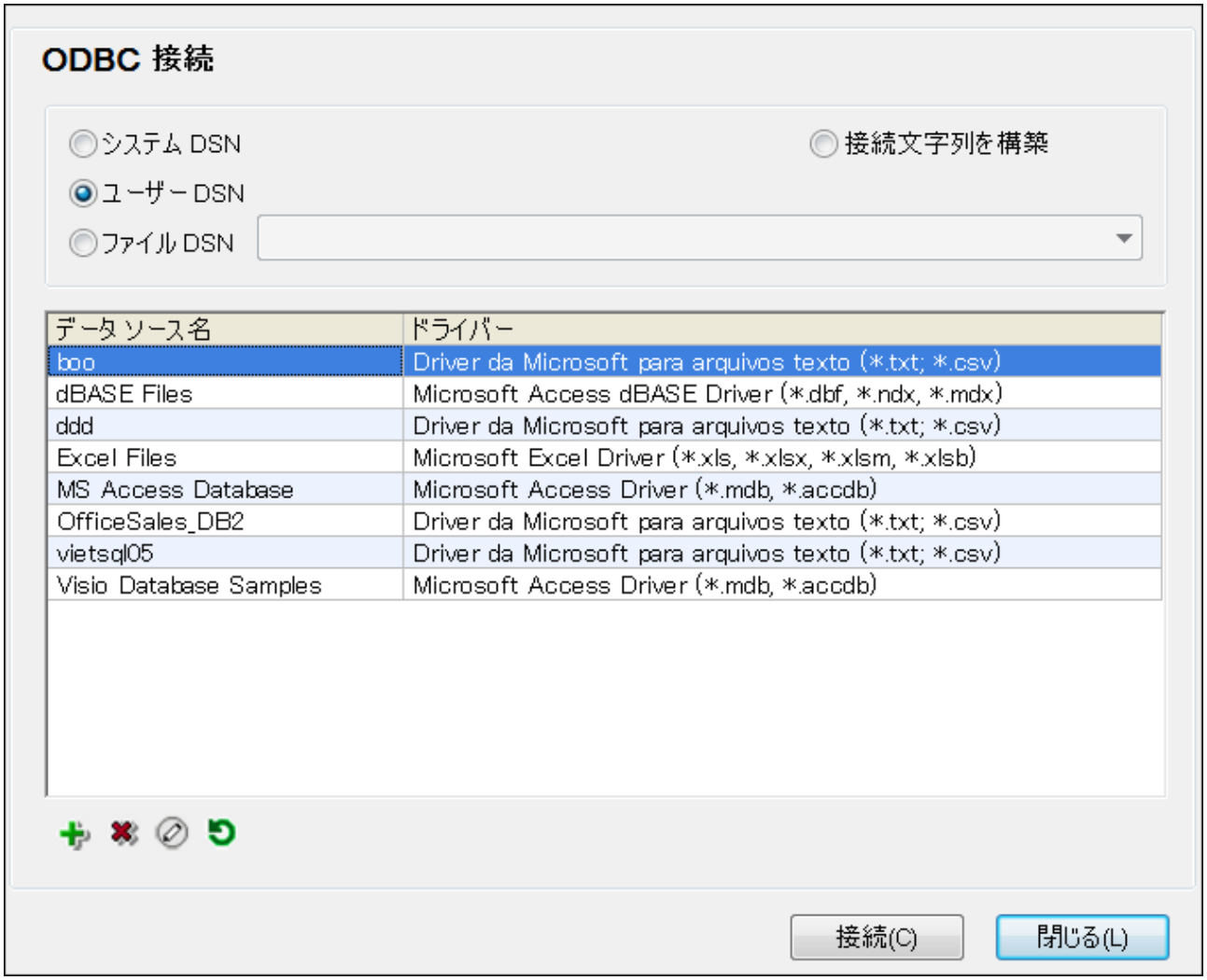
ODBC (Open Database Connectivity) は、一般に使用されているデータアクセス技術で、MapForce からデータベースに接続することができます。データベースにアクセスするための主要な手段として、またはネイティブな OLE DB- や JDBC-による接続の代替として使用することができます。

ODBC を使用してデータベースに接続するには、最初に ODBC データソース名(DSN) を作成する必要があります。このステップは、他のユーザーまたはオペレーティングシステムにより DSN が既に作成されている場合必要ありません。DSN は、統一された、MapForce を含むオペレーティングシステム上の ODBC に対応したクライアントアプリのデータベース接続方法を表します。DSN には以下の種類があります:

- ファイル DSN
- ユーザー DSN
- ファイル DSN

オペレーティングシステムで特権を持つすべてのユーザーはシステム データ ソースにアクセスすることができます。作成したユーザーのみがユーザー データ ソースにアクセスすることができます。*ファイル DSN* を作成した場合、データソースは .dsn 拡張子を持つファイルとして作成され、データソースにより使用されているドライバーがインストールされていることが前提で、他のユーザーと共有することができます。

ODBC 接続ダイアログボックス内の [**ODBC 接続**] をクリックすると、使用中のコンピューターで既に使用することのできる DSN はデータベース接続ダイアログボックスに表示されています。



ODBC 接続ダイアログボックス

データベースにより必要とされるDSN が無い場合、MapForce データベース接続ウィザードが作成の手助けをします。しかし、W indows オペレーティングシステムに直接作成することもできます。いずれにせよ、続行する前に、使用中のデータベースにODBC ドライバーがオペレーティングシステムのODBC ドライバーリストに適用できることを確認してください。(使用可能な ODBC ドライバーの表示を参照)。

新規のDSNを使用しての接続:

1. データベース接続ウィザードを開始します。
2. データベース接続ダイアログボックスの[ODBC 接続] をクリックします。
3. システムDSN を作成するには、オペレーティングシステムの管理者権限が必要です。

   > また、MapForce が管理者として作動している必要があります。

4. 「追加」 をクリックします。

5. ドライバーを選択して、(作成するDSN の種類に応じて) [ユーザーDSN] または[システムDSN] をクリックします。使用中のデータベースに適用することのできるドライバーがリストされていない場合、データベースベンダーからダウンロードして、インストールしてください。(次を参照: データベースドライバー概要)。
6. ポップアップするダイアログボックスに特定の接続情報を入力してセットアップを完了します。

接続を成功させるには、データベースサーバーのホスト名(またはIP アドレス)、とデータベースのユーザー名およびパスワードを提供する必要があります。他の接続パラメーターがある可能性がありますが、これらのパラメーターはデータベースプロバイダーにより異なります。各接続方法に固有のパラメーターについての詳細は、ドライバープロバイダーのドキュメントを参照してください。一度作成されると、DSN はデータソース名のリストで使用することができます。これによりデータベース接続の詳細をデータベースの接続を希望する際いつでも使用することができます。ユーザーDSN はユーザーDSN のリストに追加されますが、システムDSN はシステムDSN のリストに追加されます。

既存のDSNを使用しての接続:

1. データベース接続ウィザードを開始します。
2. [ODBC 接続] をクリックします。
3. 既存のデータソース(ユーザーDSN、システムDSN、ファイルDSN) から種類を選択します
4. 既存 DSN レコードをクリックして、「接続」をクリックします。

既存の .dsn ファイルをベースにして接続文字列を構築する:

1. データベース接続ウィザードを開始します。
2. [ODBC 接続] をクリックします。
3. [接続文字列の構築] を選択して [構築] をクリックします。
4. ファイルDSN を使用して、接続文字列を構築する場合、[ファイルデータソース] タブをクリックします。それ以外の場合は、[マシンデータソース] タブをクリックします。(システムDSN とユーザーDSN は"マシン" データソースとして知られています。)
5. 必要とされる .dsn ファイルを選択して、「OK」 をクリックします。

用意された接続文字列を使用して接続する:

1. データベース接続ウィザードを開始します。
2. [ODBC 接続] をクリックします。
3. [接続文字列の構築] を選択しています。
4. 接続文字列を与えられたボックスに貼り付け、「接続」 をクリックします。

6.2.1.5.1 使用可能な ODBC ドライバーの表示

ODBC データソース管理で使用中のオペレーティングシステムで使用可能な ODBC ドライバーを確認することができます。W indows コントロールパネルの管理ツールからODBC データソース管理 (**Odbcad32.exe**) にアクセスすることができます。64-ビット版オペレーティングシステムには、2 種類のバージョンの実行可能ファイルがあります:

- **C:\Windows\SysWoW64** ディレクトリにある、**Odbcad32.exe** ファイルの32-ビット版バージョン(**C:** が使用中のシステムドライブとして前提されています)。
- **C:\Windows\System32** ディレクトリにある、**Odbcad32.exe** ファイルの64-ビット版バージョン

インストールされている32-ビット版データベースドライバーは、ODBC データソース管理の32-ビット版バージョンで、64-ビット版バージョン内の64-ビット版ドライバーで確認することができます。ですから、対応するバージョンのODBC データソース管理でデータベースドライバーを確認してください。

ODBC データソース管理者

ターゲットデータベースのドライバーがリストに存在しない場合、または、代替ドライバーを追加する場合、データベースのベンダーからダウンロードしてください(データベースドライバー概要 を参照)。ODBC ドライバーがシステムで使用可能になると、ODBC 接続を作成することができます(ODBC 接続のセットアップを参照)。

6.2.1.6 JDBC 接続の設定

JDBC (Java Database Connectivity) は、Oracle の Java ソフトウェアプラットフォームの一部であるデータベースアクセスインターフェイスです。JDBC 接続は ODBC 接続に比べ、通常より多くリソースを集中的に使用する接続ですが、ODBC で使用することのできない機能を使用することができます。ODBC コネクタにより使用できないデータベース機能を使用する場合 JDBC 接続を使用することが奨励されています。

前提条件

- JRE (Java Runtime Environment) または Java Development Kit (JDK) がインストールされている必要があります。これは Oracle JDK または Oracle OpenJDK などのオープンソースビルドであることができます。MapForce は次のロケーションから Java Virtual Machine (JVM) へのパスを次の順序で決定します: a) アプリケーション オプション内で設定したカスタム JVM パス Java 設定を参照してください。b) Windows レジストリ内で見つかった JVM パス。c) `JAVA_HOME` 環境変数。
- MapForce (32-ビット、64-ビット) のプラットフォームが JRE/JDK のプラットフォームに一致することを確認してください。
- データベースベンダーからの JDBC ドライバーがインストールされている必要があります。これは使用可能でありデータベースによりサポートされている場合、データベースクライアント インストールの一部として、または、JDBC ライブラリ (.jar ファイル) として個別にダウンロードされている JDBC ドライバーの可能性があります。データベース接続のサンプルも参照してください。
- Oracle データベースに接続する場合は、一部の Oracle ドライバーは特定の JRE バージョンに固有のため、追加コンポーネントと構成が必要な点に注意してください。Oracle 製品のドキュメント (例えば、「JDBC 開発者ガイドおよびリファレンス」) には、各 JDBC ドライバーの構成の手続きの詳細が説明されています。

(C) 2015-2021 Altova GmbH

- CLASSPATH 環境変数は、(.jar ファイルの1つまたは複数の) 使用中のWindows オペレーティングシステムのJDBC ドライバーのパスを含む必要があります。データベースクライアントをインストールする際に、インストーラーはこの変数を自動的に構成する可能性があります (CLASSPATH の構成を参照)。

Windows 資格情報を使用してJDBC を介してSQL Server に接続する方法

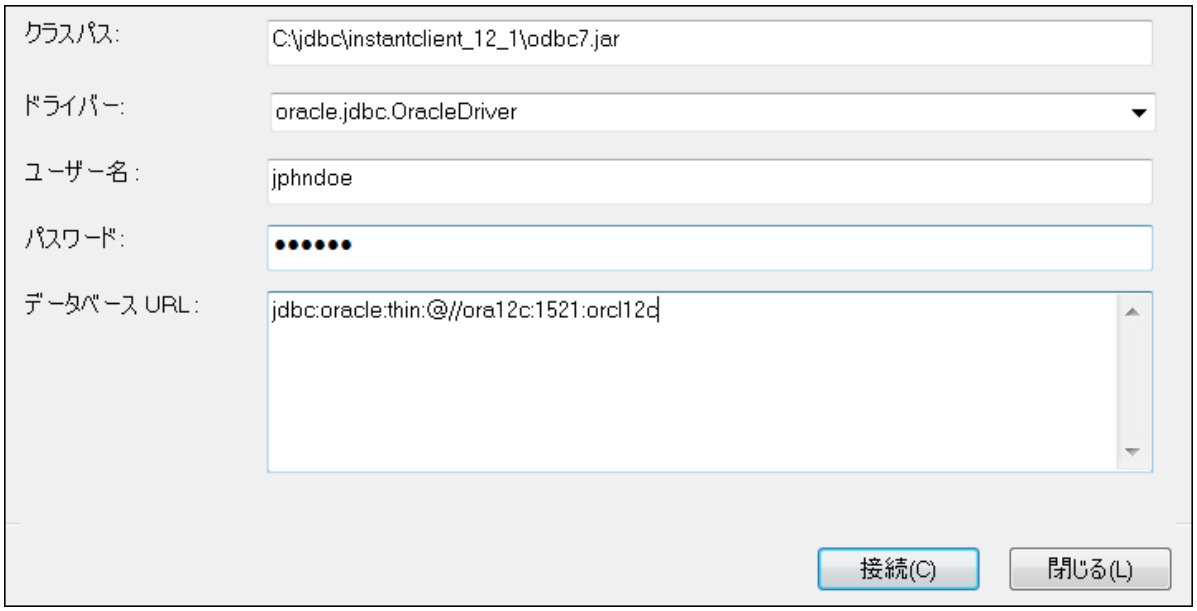
Windows 資格情報 (統合されたセキュリティ) を使用してJDBC を介してSQL Serverに接続する場合、以下に注意してください:

- JDBC ドライバーパッケージに含まれている **sqljdbc_auth.dll** ファイルはシステムPATH 環境変数上のディレクトリにコピーされる必要があります。このようなファイルは2個存在します。1つはx86 プラットフォームのために、もう1つはx64 プラットフォームのために存在します。JDK プラットフォームに対応するPATH が追加されていることを確認してください。また、環境変数の変更後MapForce (適用可能な場合マッピングを実行しているプログラム) を再起動してください。
- JDBC 接続文字列はプロパティ **integratedSecurity=true** を含む必要があります。このプロパティを異なる場所から追加することができます:
  - データベース接続ウィザードから追加することができます。以下を参照してください。
  - データベースコンポーネント設定から追加することができます。
  - 適応可能な場合、生成済みのJava コード内のデータベース接続文字列の編集をすることにより追加することができます。

詳細に関しては *SQL Server のための Microsoft JDBC ドライバー* ドキュメントを参照してください。
https://docs.microsoft.com/en-us/sql/connect/jdbc/building-the-connection-url。

JDBC 接続のセットアップ方法

1. データベース接続ウィザードを開始します。
2. 「**JDBC 接続**」をクリックします。
3. 任意で、セミコロンで区切られた.jar ファイルパスのリストを「クラスパス」テキストボックスに入力することもできます。ここで入力される.jar ライブラリは、CLASSPATH 環境変数内で既に定義されているものに加え、環境にロードされます。「クラスパス」テキストボックスの編集が完了すると、ソース.jar ライブラリ内で検索されるJDBC ドライバーは、「ドライバー」リストに自動的に追加されます (次のステップを参照してください)。



4. 「ドライバー」の横でJDBC ドライバーをリストから選択、または、Java クラス名を入力します。CLASSPATH 環境変数により構成されたJDBC ドライバーおよび、「クラスパス」テキストボックス内で検出されたドライバーが含まれます (次を参照: CLASSPATH の構成)。

CLASSPATH 変数内で定義されている JDBC ドライバーのパスを使用することができます。また、データベース接続ダイアログボックスに直接入力された .jar パスは Java Virtual Machine (JVM) によりサポートされています。JVM は接続を構築するためのドライバーを決定します。JVM にロードされる Java クラスを確認し、JDBC ドライバーの競合、および、データベースに接続する際に発生する予期されないエラーを回避するようにしてください。

5. 対応するボックスに、ユーザー名とパスワードを入力します。
6. データベース URL テキストボックス内に、JDBC 接続 URL (文字列) をデータベースの型のとおり入力します。次のテーブルは一般のデータベースの型のための JDBC 接続 URL (文字列) の構文について説明しています。

データベース	JDBC 接続 URL
Firebird	`jdbc:firebirdsql://<host>[:<port>]/<database path or alias>`
IBM DB2	`jdbc:db2://hostName:port/databaseName`
IBM DB2 for i	`jdbc:as400://[host]`
IBM Informix	`jdbc:informix-sqli://hostName:port/databaseName:INFORMIXSERVER=myserver`
MariaDB	`jdbc:mariadb://hostName:port/databaseName`
Microsoft SQL Server	`jdbc:sqlserver://hostName:port;databaseName=name`
MySQL	`jdbc:mysql://hostName:port/databaseName`
Oracle	`jdbc:oracle:thin:@hostName:port:SID` `jdbc:oracle:thin:@//hostName:port/service`
Oracle XML DB	`jdbc:oracle:oci:@//hostName:port:service`
PostgreSQL	`jdbc:postgresql://hostName:port/databaseName`
Progress OpenEdge	`jdbc:datadirect:openedge://host:port;databaseName=db_name`
Sybase	`jdbc:sybase:Tds:hostName:port/databaseName`
Teradata	`jdbc:teradata://databaseServerName`

メモ 上記のフォーマットに対する構文のバリエーションも使用することができます (例えば、データベース URL はポートを除外、またはデータベースのユーザー名とパスワードを含む可能性があります)。詳細はデータベースのベンダーのドキュメントを参照してください。

7. 「接続」をクリックします。

6.2.1.6.1 CLASSPATH の構成

CLASSPATH 環境変数はクラスと、オペレーティングシステム上の他のリソースファイルを検索するため Java Runtime Environment (JRE) または Java Development Kit (JDK) により使用されます。JDBC を介してデータベースに接続すると、この変数はオペレー

ティングシステム上のJDBC ドライバーへのパス、または一部の場合、使用中のデータベースの種類に固有の追加ライブラリファイルへのパスを含むように構成されなければなりません。

下のテーブルは、CLASSPATH 変数に通常含まれる必要のあるサンプルファイル パスをリストしています。重要なことは、システム上の情報をシステム上のJDBC ドライバーのロケーション、JDBC ドライバー名、およびオペレーティングシステム上に存在するJRE/JDK バージョンをベースにしたこの情報を調整する必要があります。接続問題を避けるために、使用中のオペレーティングシステムにインストールされているJDBC ドライバに適用されるインストールの指示、インストール前とインストール後の構成ステップを確認してください。

データベース	サンプルCLASSPATH エントリ
Firebird	`C:\Program Files\Firebird\Jaybird-2.2.8-JDK_1.8\jaybird-full-2.2.8.jar`
IBM DB2	`C:\Program Files (x86)\IBM\SQLLIB\java\db2jcc.jar;C:\Program Files (x86)\IBM\SQLLIB\java\db2jcc_license_cu.jar;`
IBM DB2 for i	`C:\jt400\jt400.jar;`
IBM Informix	`C:\Informix_JDBC_Driver\lib\ifxjdbc.jar;`
Microsoft SQL Server	`C:\Program Files\Microsoft JDBC Driver 4.0 for SQL Server\sqljdbc_4.0\enu\sqljdbc.jar`
MariaDB	**<installation directory>**\mariadb-java-client-2.2.0.jar
MySQL	**<installation directory>**\mysql-connector-java-*version*-bin.jar;
Oracle	**ORACLE_HOME**\jdbc\lib\ojdbc6.jar;
Oracle (with XML DB)	**ORACLE_HOME**\jdbc\lib\ojdbc6.jar;**ORACLE_HOME**\LIB\xmlparserv2.jar;**ORACLE_HOME**\RDBMS\jlib\xdb.jar;
PostgreSQL	**<installation directory>**\postgresql.jar
Progress OpenEdge	%DLC%\java\openedge.jar;%DLC%\java\pool.jar; メモProgress OpenEdge SDK がマシンにインストールされていると想定します。%DLC% は OpenEdge がインストールされているディレクトリです。
Sybase	C:\sybase\jConnect-7_0\classes\jconn4.jar
Teradata	**<installation directory>**\tdgssconfig.jar;**<installation directory>**\terajdbc4.jar

- CLASSPATH 変数の変更は、使用中のコンピュータのJava アプリケーションの振る舞いに影響する可能性があります。続行する前に生じえる影響を把握するために、Java ドキュメントを参照してください。
- 環境変数は、ユーザーまたはシステムであることができます。システム環境変数を変更するには、オペレーティングシステムに対する管理者権限が必要です。
- 環境変数を変更後、設定の変更を有効にするには、プログラムを再起動してください。または、オペレーティングシステムをログオフまたは再起動してください。

Windows 7 でCLASSPATH を構成する:

1. [スタート] メニューを開始して、[コンピューター] を右クリックします。

2. [プロパティ] をクリックします。
3. [システムの詳細設定] をクリックします。
4. [詳細設定] タブから [環境変数] をクリックします。
5. CLASSPATH 変数を、ユーザーまたはシステム環境変数からロケートして、編集をクリックします。CLASSPATH 変数が存在しない場合は、[新規] をクリックして作成します。
6. オペレーティングシステム内で JDBC ドライバーがある場所にパスを含むように変数の値を編集します。JDBC ドライバーパスを他の既に CLASSPATH 変数内にあるパスから区別するためにセミコロン(;)で区切ります。

Windows 10 で CLASSPATH を構成する

1. Windows キーを押して、環境編集を入力します。
2. [システム環境変数の編集] の提案をクリックします。
3. [環境変数] をクリックします。
4. CLASSPATH 変数を、ユーザーまたはシステム環境変数からロケートして、編集をクリックします。CLASSPATH 変数が存在しない場合は、[新規] をクリックして作成します。
5. オペレーティングシステム内で JDBC ドライバーがある場所にパスを含むように変数の値を編集します。JDBC ドライバーパスを他の既に CLASSPATH 変数内にあるパスから区別するためにセミコロン(;)で区切ります。

6.2.1.7 PostgreSQL 接続のセットアップ

PostgreSQL データベースへの接続は、ネイティブな接続として、または、ODBC、JDBC、および他のドライバーを使用して確立することができます。ネイティブな接続の利点は、使用中のシステムにインストールする場合、ドライバーを必要としないことです。また、Linux、または、macOS サーバー上での実行のためにファイルをデプロイする場合、ターゲットサーバー上にドライバーをインストールする必要はありません

ネイティブでは内ドライバーを使用しての接続を作成する場合は、以下のトピックを参照してください:

- JDBC 接続の設定
- PostgreSQL (ODBC) への接続

それ以外の場合、PostgreSQL へのネイティブの接続をセットアップする場合、以下の手順に従ってください。継続するには、以下の前提条件が必要です: ホスト名、ポート、データベース名、ユーザー名、およびパスワード。

ネイティブ PostgreSQL 接続のセットアップ:

1. データベース接続ウィザードを開始します.
2. 「PostgreSQL 接続」をクリックします。
3. 対応するボックスに、ホスト(PostgreSQL が同じマシンで作動している場合、*ローカルホスト*)、ポート(通常、5432 ですが、これは任意です)、データベース名、ユーザー名とパスワードを入力します。

4. 「**接続**」をクリックします。

PostgreSQL データベースサーバーが異なるマシンに存在する場合、次の点に注意してください:

- PostgreSQL データベースサーバーは、クライアントからの接続を受け入れるように構成されなければなりません。具体的には、**pg_hba.conf** ファイルがローカルではない接続を許可するように構成される必要があります。第2に、**postgresql.conf** ファイルは、特定されたIP アドレスとポートをリッスンするように構成される必要があります。詳細に関しては、PostgreSQL ドキュメントを参照してください(https://www.postgresql.org/docs/9.5/static/client-authentication-problems.html)。
- サーバーマシンがファイアウォールを越えてデザインされたポート(通常、5432)上で接続を受け入れるように構成されている必要があります。例えば、「**コントロール パネル**> **Windows ファイアウォール**> **詳細設定** > **受信の規則**」で設定すると、Windows 上で作動するデータベースサーバーでは、ファイアーウォールを越えてポート 5432 に接続を許可するルールを作成する必要がない場合があります。

6.2.1.8 SQLite 接続の設定

SQLite (http://www.sqlite.org) は、ポータブル性とインストールの簡便性が重要な場合理想的な構成のファイルベースの、内蔵のデータ型です。Since SQLite データベースは、MapForce によりネイティブにサポートされるため、接続のためにドライバーをインストールする必要はありません。

SQLite データベースサポートに関するメモ

- (マッピングをプレビューまたはMapForce Server 実行ファイルを実行する際) SQLite データベースはMapForce 内蔵 変換言語によりサポートされています。
- ユーザー定義関数(UDF) はSQLite データベースをサポートしません。

- Linux では、SQLite データベースのステートメント実行タイムアウトはサポートされません。
- フルテキスト検索テーブルはサポートされません。
- テーブルの各行内の異なるデータ型の値を SQLite は許可します。MapForce では、処理されたすべての値は宣言された列の型との互換性が必要です。ですから、SQLite データベースが宣言された列の値と異なる行の値を持つ場合ランタイムエラーが起る可能性があります。
- SQLite データベースにマッピングを書き込む場合、また、ターゲットデータベースが存在しない場合、外部にターゲットデータベースを作成する必要があります。この場合、DatabaseSpy (http://www.altova.com/ja/databasespy.html)などのツールを使用して作成するか、または SQLite コマンドラインシェルを公式の Web サイトからダウンロードして、コマンドラインからデータベースファイルを作成してください(サンプル: XML から SQLite へデータをマップ を参照)。SQLite コマンド構文に関する完全なレファレンスは、SQLite の公式のドキュメントを参照してください。

6.2.1.8.1 既存の SQLite データベースへの接続

既存の SQLite データベースに接続する方法:

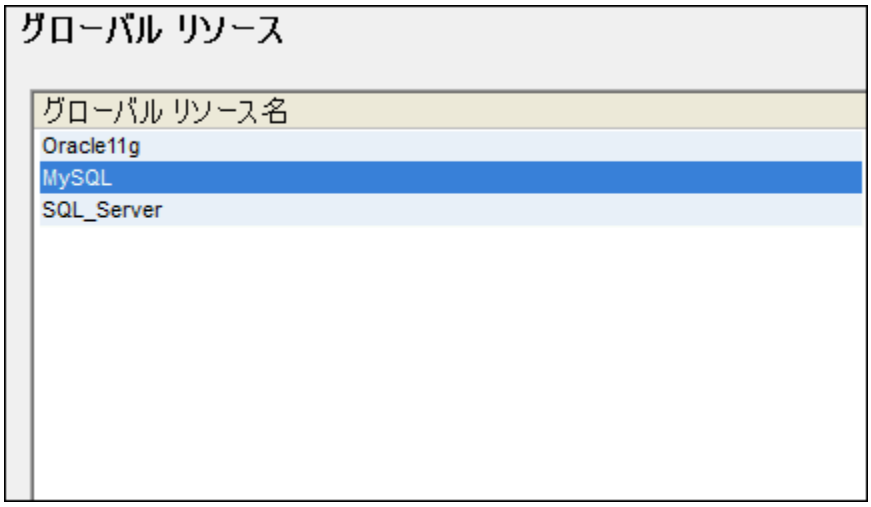
1. データベース接続ウィザードを実行します(データベース接続ウィザードの開始を参照)。
2. **SQLite** を選択して、「次へ」をクリックします。
3. SQLite データベースファイルを参照して、または、(相対または絶対)パスを入力します。「**接続**」ボタンは、データベースファイルへのパスを入力すると有効化されます。
4. 「**接続**」をクリックします。

6.2.1.9 グローバルリソースからの接続の使用

Altova グローバルリソースはファイル、フォルダー、およびデータベースへのポータブルなレファレンスです。以前にデータベースをグローバルリソースとして既に構成している場合、接続をいつでも再使用することができます(異なる Altova アプリケーション間でも使用可能)。詳細に関しては、Altova グローバルリソースを参照してください。

グローバルリソースからのデータベース接続の使用:

1. データベース接続ウィザードを開始します。
2. [**グローバルリソース**] をクリックします。グローバルリソースとして使用可能なデータベース接続がリストされます。



3. データベース接続履歴から「接続」をクリックします。

ヒント: 各グローバルリソースの追加情報を取得するために、マウスカーソルをグローバルリソースにポイントします。

6.2.1.10 データベース接続のサンプル

ADO、ODBC、またはJDBC を使用したMapForce からのデータベースへの接続のためのサンプルがこのセクションには含まれています。ADO.NET 接続サンプルは個別にリストされています サンプルADO.NET 接続文字列 を参照してください。For instructions about establishing a native connection to PostgreSQL とSQLite へのネイティブな接続の作成のための手順は、それぞれ PostgreSQL 接続のセットアップ とSQLite 接続のセットアップ を参照してください。

以下の点に注意してください:

- W indows 構成、ネットワーク環境、およびデータベースクライアント、または、サーバーソフトウェアが各サンプルと異なる場合手順は異なります。
- ほぼすべてのデータベースの型では、1つ以上のデータアクセステクノロジー(ADO、ADO.NET、ODBC、JDBC) またはドライバーを使用して接続することができます。データベース接続パフォーマンス、およびドライバーの機能と制限は、MapForce 外で構成された選択されたドライバー、(該当する場合)データベースクライアントソフトウェア、および追加接続パラメータなどによって異なります。

6.2.1.10.1 Firebird (JDBC) への接続

このサンプルではFirebird データベースサーバーへの接続のサンプルの命令について説明されています。

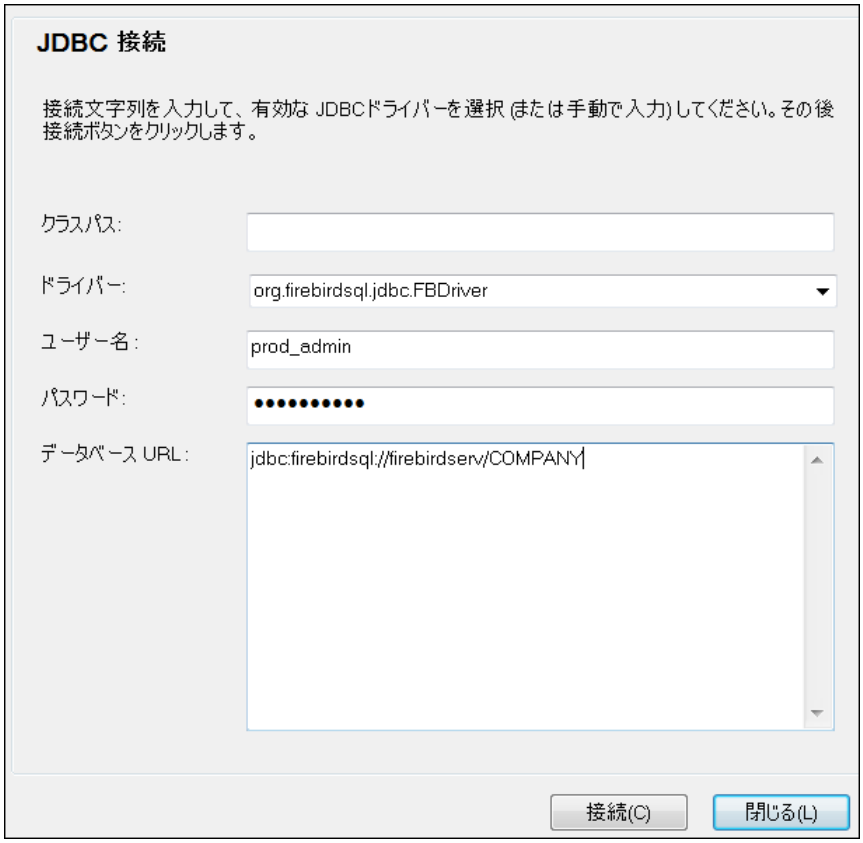
必要条件:

- JRE (Java Runtime Environment) またはJava Development Kit (JDK) がインストールされている必要があります。これはOracle JDK または Oracle OpenJDK などのオープンソースビルドであることができます。MapForce は次のロケーションからJava Virtual Machine (JVM) へのパスを次の順序で決定します: a) アプリケーション オプション内で設定したカスタムJVM パス Java 設定を参照してください。b) W indows レジストリ内で見つかったJVM パス。c) `JAVA_HOME` 環境変数。
- MapForce (32-ビット、64-ビット) のプラットフォームがJRE/JDK のプラットフォームに一致することを確認してください。

- Firebird JDBC ドライバーが使用中のオペレーティングシステムで使用可能である必要があります（データベースへの接続を可能にする .jar ファイルのフォームをとります）。ドライバーは Firebird Web サイトからダウンロードすることができます（ http://www.firebirdsql.org/ ）。このサンプルは、*Jaybird 2.2.8* を使用します。
- データベース接続の詳細があります: ホスト、データベース パス、またはエイリアス、ユーザー名、および パスワード。

JDBC を使用して Firebird に接続する

1. データベース接続ウィザードを開始します.
2. 「JDBC 接続」をクリックします。
3. 「クラス パス」の横にデータベースへの接続を与える .jar ファイルへの パスを入力します。必要であれば、セミコロンで区切られた .jar ファイル パスのリストを入力することもできます。このサンプルでは、必要とされる .jar ファイルは次の パスを使用して見つけることができます: **C:\jdbc\firebird\jaybird-full-2.2.8.jar**。.jar ファイル パスをオペレーティングシステムの CLASSPATH 環境変数に追加した場合、「クラス パス」テキスト ボックスを空白のままにすることができます。（次も参照: CLASSPATH の構成）。
4. 「ドライバー」ボックスから、**org.firebirdsql.jdbc.FBDriver** を選択します。このエントリは、有効な .jar ファイル パスが、「クラス パス」テキスト ボックス内、または、オペレーティングシステムの CLASSPATH 環境変数内で検出された場合のみ使用することができます（前のステップを参照してください）。



5. 対応するテキスト ボックスに、データベースへのユーザー名と パスワードを入力します。
6. URL テキスト ボックスに、データベース サーバーに適用することのできる ハイライトされた値を置き換える、データベース サーバーの接続文字列を入力します。

```
jdbc:firebirdsql://<host>[:<port>]/<database path or alias>
```

7. 「接続」をクリックします。

(C) 2015-2021 Altova GmbH

6.2.1.10.2 Firebird (ODBC) への接続

このサンプルでは、Linux サーバー上で実行される、Firebird 2.5.4 データベースへの接続について説明されています。

必要条件:

- Firebird データベースサーバーは、クライアントからのTCP/IP 接続を受け入れるように構成されています。
- Firebird ODBC ドライバーを使用中のオペレーティングシステムにインストールする必要があります。このサンプルは、を使用します。 Firebird ODBC ドライバー バージョン 2.0.3.154 は、Firebird ウェブサイト (http://www.firebirdsql.org/) からダウンロードすることができます。
- Firebird クライアントを、使用中のオペレーティングシステムにインストールする必要があります。 Firebird 2.5.4 クライアントのためのスタンドアロンのインストーラーが存在しない点に注意してください。また、クライアント は、サーバーのインストール パックの一部です。Firebird サーバー インストール パックを Firebird ウェブサイト(http://www.firebirdsql.org/) からダウンロードすることができます。ダウンロードの際、[スーパークラシック/クラシック または スーパーサーバーのための、W indows 実行ファイル] インストーラーを選択してください。クライアント ファイルのみをインストールする場合、[最小クライアント インストール- サーバーツール無し] を選択します。

  重要点:

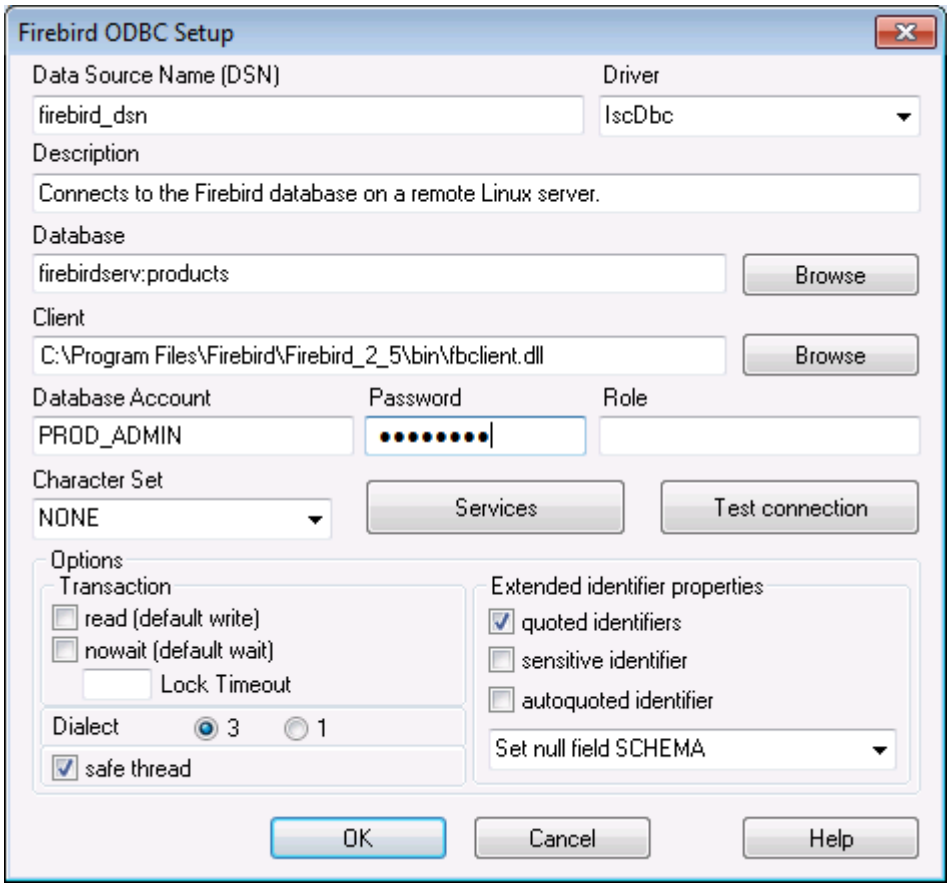
  - Firebird ODBC ドライバーおよびクライアント (32 ビット または 64 ビット) のプラットフォームは、MapForce のプラットフォームに適応する必要があります。
  - Firebird クライアントのバージョンは、接続されている Firebird サーバーのバージョンに適応する必要があります。

- データベース接続の詳細 サーバー上のサーバーホスト名 または IP アドレス、データベース パス (または エイリアス)、ユーザー名、および パスワード。

ODBC を使用して Firebird に接続する:

1. データベース接続ウィザードを開始します。
2. [ODBC 接続] をクリックする。
3. [ユーザー DSN] を選択して、(または、管理者の特権がある場合、 [システム DSN] を選択して、)[追加] をクリックします。

4. Firebird ドライバーを選択して、をクリックします。そして[ユーザーDSN](前のステップでの選択により、または、[システムDSN])。Firebird ドライバーがリスト内に存在しない場合、使用中のオペレーティングシステムにインストールされていることを確認してください(使用可能な ODBC ドライバーの表示 を参照)。



5. データベース接続の詳細を入力します:

データソース名 (DSN)	作成する詳細名とデータソースを入力します。

データベース	サーバーホスト名 またはIP アドレスをコロンとデータベース エイリアス(またはパス)の後に入力します。このサンプルでは、ホスト名は、`firebirdserv`、データベースエイリアスは、`products` です:
	
	`firebirdserv:products`
	
	データベース エイリアスを使用することにより、サーバー側上では、データベース管理者は、エイリアス製品を構成し、サーバー上の実際のFirebird (.fdb) データベースファイルをポイントします(詳細は、Firebird のドキュメントを参照してください)。
	
	ホスト名の代わりに サーバーIP アドレス、エイリアスの代わりにパスを使用することもできます。ですから、以下のサンプル接続の文字列は有効です:
	
	`firebirdserver:/var/Firebird/databases/butterflies.fdb`
	`127.0.0.1:D:\Misc\Lenders.fdb`
	データベースがW indows マシン上に存在する場合、[参照] をクリックして、Firebird (.fdb) データベースファイルを直接選択します。

クライアント	**fbclient.dll** ファイルへのパスを入力します。デフォルトでは、これはFirebird インストールディレクトリのサブディレクトリです。
データベース アカウント	データベース管理者（このサンプルの場合 PROD_ADMIN）により与えられたデータベースユーザー名を入力します。
パスワード	データベース管理者により与えられた データベース パスワード を入力します。

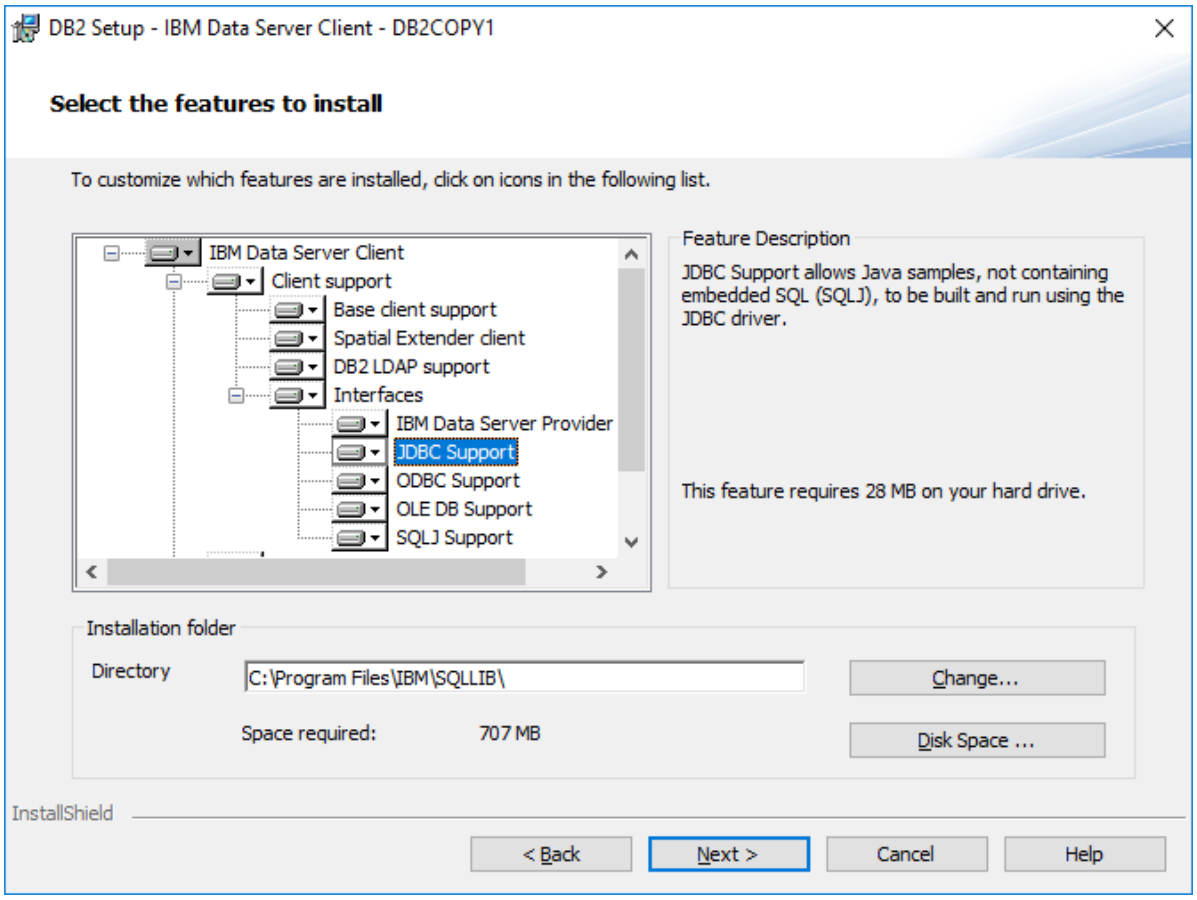
6. 「OK」をクリックします。

6.2.1.10.3 IBM DB2 (JDBC) への接続

このサンプルでは、JDBC を使用して **IBM DB2** データベースサーバーに接続する手順のサンプルについて説明されています。

必要条件:

- JRE (Java Runtime Environment) または Java Development Kit (JDK) がインストールされている必要があります。Oracle JDK または Oracle OpenJDK などのオープンソースビルドであることができます。MapForce は次のロケーションから Java Virtual Machine (JVM) へのパスを次の順序で決定します: a) アプリケーション オプション内で設定したカスタム JVM パス Java 設定を参照してください。b) Windows レジストリ内で見つかった JVM パス。c) JAVA_HOME 環境変数。
- MapForce (32-ビット、64-ビット) のプラットフォームが JRE/JDK のプラットフォームに一致することを確認してください。このサンプルは Oracle の OpenJDK 11.0 64-bit を使用し、その結果 MapForce の64 ビットバージョンを使用します。
- (データベースへの接続を与える1 つ、または、複数の .jar ファイル) JDBC ドライバーが使用中のオペレーティングシステムで使用可能である必要があります。**IBM Data Server Client** バージョン 10.1 (64-ビット) をインストール後使用することのできる JDBC ドライバーをこのサンプルは使用しています。JDBC ドライバーをインストールするためには、**通常** インストールを選択してください。または、インストールウィザードでこのオプションを明示的に選択してください。



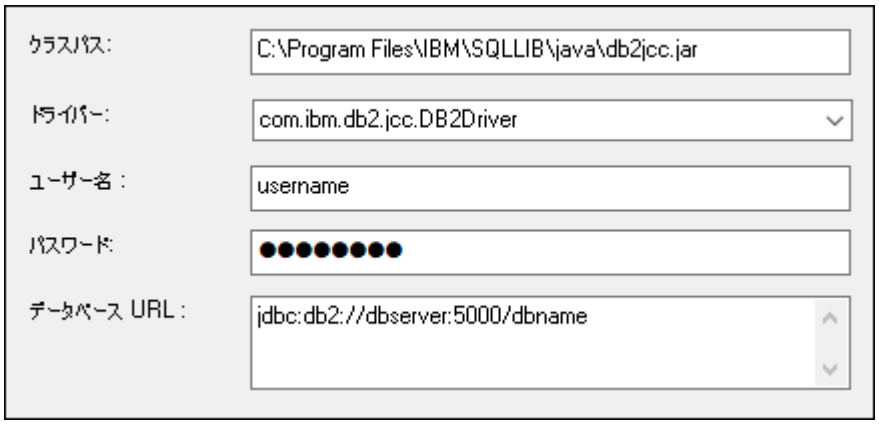
デフォルトのインストールパスを変更していない場合、インストール後必要とされる .jar ファイルは **C:\Program Files\IBM\SQLLIB\java** ディレクトリ内で見つけることができます。

- 以下のデータベース詳細が必要になります: ホスト、データベース、ポート、ユーザー名、および、パスワード。

JDBC を使用して IBM DB2 に接続する:

1. データベース接続ウィザードを開始します.
2. 「JDBC 接続」をクリックします。
3. 「クラスパス」の横に、データベースへの接続を与える .jar ファイルへのパスを入力します。このサンプルは **C:\Program Files\IBM\SQLLIB\java\db2jcc.jar** を参照します。データベースサーバーバージョンにより **db2jcc4.jar** ドライバーを参照する必要がある場合があります。ドライバーの互換性に関しては IBM ドキュメントを参照してください（http://www-01.ibm.com/support/docview.wss?uid=swg21363866）。.jar ファイルパスをオペレーティングシステムの CLASSPATH 環境変数に追加している場合クラスパステキストボックスを空にしておくことができます（CLASSPATH の構成を参照してください）。
4. 「ドライバー」ボックスから、**com.ibm.db2.jcc.DB2Driver** を選択します。このエントリは、有効な .jar ファイルパスが、「クラスパス」テキストボックス内、または、オペレーティングシステムの CLASSPATH 環境変数内で検出された場合のみ使用することができます（前のステップを参照してください）。

(C) 2015-2021 Altova GmbH



5. データベースユーザーのユーザー名とパスワードを対応するテキストボックスに入力します。
6. JDBC 接続文字列をデータベース URL テキストボックスに入力します。使用中のデータベースサーバーに適用することのできるものと接続の詳細が置き換えられていることを確認してください。

```
jdbc:db2://hostName:port/databaseName
```

7. 「接続」をクリックします。

6.2.1.10.4 IBM DB2 (ODBC) への接続

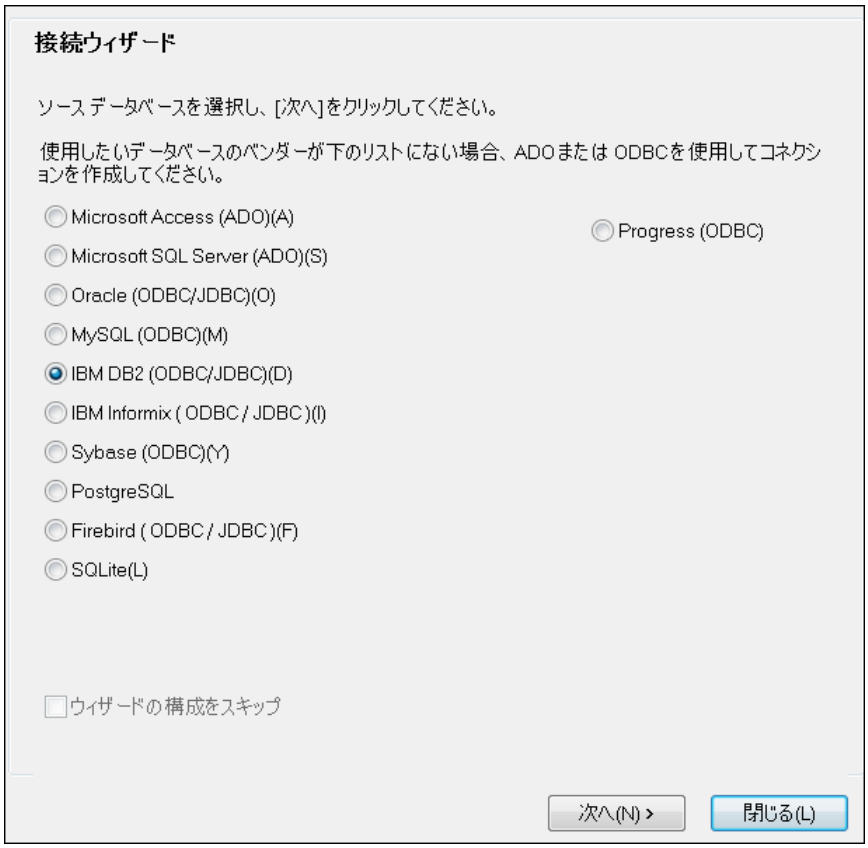
このトピックではODBC を使用した、IBM DB2 への接続のサンプル手順について説明されています。

前提条件:

- IBM Data Server Client は、使用中のオペレーティングシステムにインストールおよび構成される必要があります(このサンプルの場合はIBM Data Server Client 9.7 を使用します)。インストールの手順に関しては、IBM DB2 ソフトウェアにより提供されたドキュメントを確認してください。 IBM Data Server Client インストール後、使用中のコンピューターで ODBC ドライバーが使用可能か確認してください(使用可能な ODBC ドライバーの表示を参照)。
- データベースエイリアスを作成します。複数の作成方法があります:
  - IBM DB2 構成アシスタントから
  - IBM DB2 コマンドラインプロセッサーから
  - ODBC データソースウィザードから(この場合は、下に表示される手順を参照してください)
- データベース接続の詳細: ホスト、データベース、ユーザー名およびパスワード

IBM DB2 への接続:

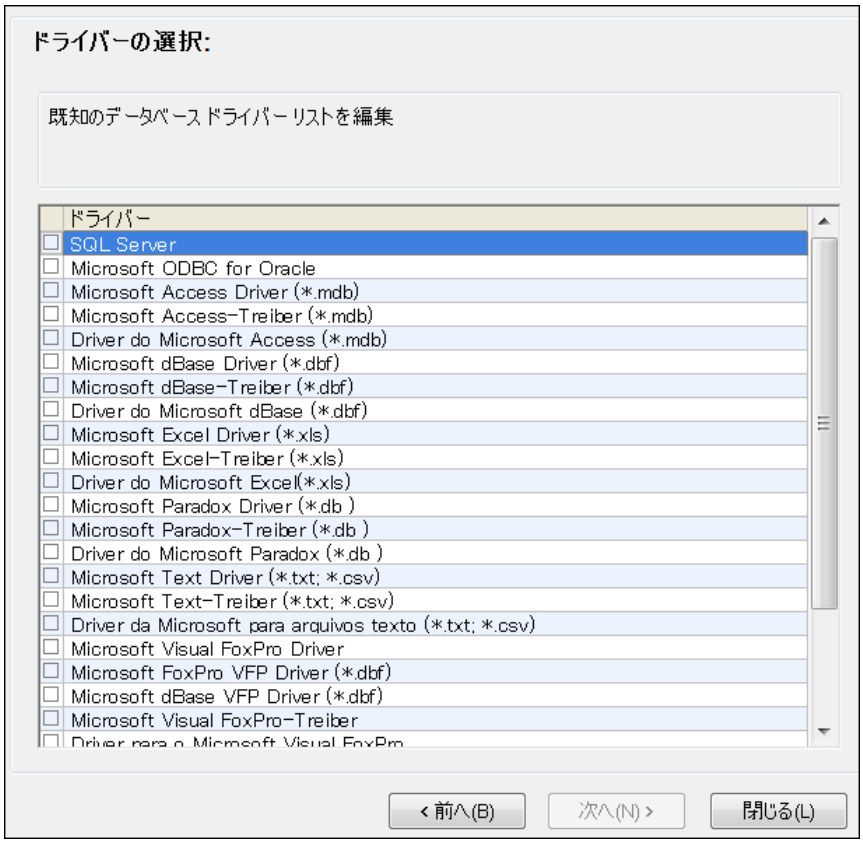
1. データベース接続ウィザードを開始します。**IBM DB2 (ODBC/JDBC)** を選択します。



2. 「次へ」をクリックします。

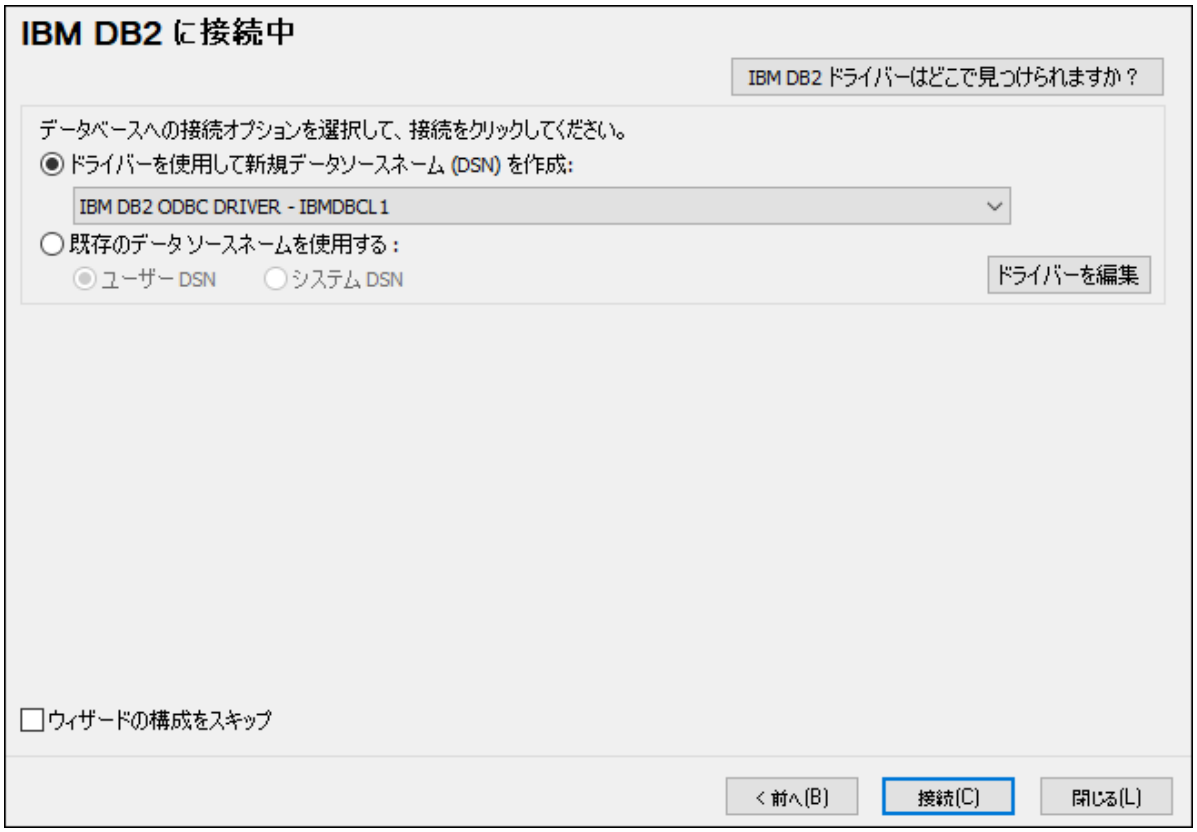
(C) 2015-2021 Altova GmbH

3. **ODBC** を選択して「次へ」をクリックします。データベースのための既知のリストを編集するようにプロンプトされると、IBM DB2 に適用することのできるドライバーを選択して(前提条件を参照)、「次へ」をクリックします。

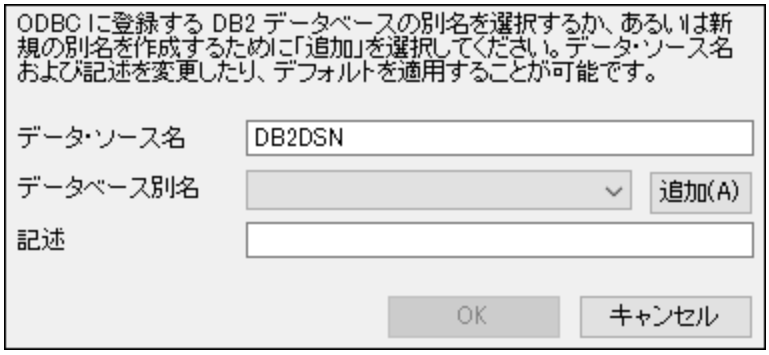


4. リストからIBM DB2 を選択して、「**接続**」をクリックします。（ドライバーで使用可能なリストを編集するには、[**ドライバーの編集**] をクリックして、使用可能なドライバーのリストを編集します。クリックして追加または削除するIBM DB2 ドライバーをチェックまたはチェックの解除をします。ドライバーを選択または選択の解除をします。）

(C) 2015-2021 Altova GmbH



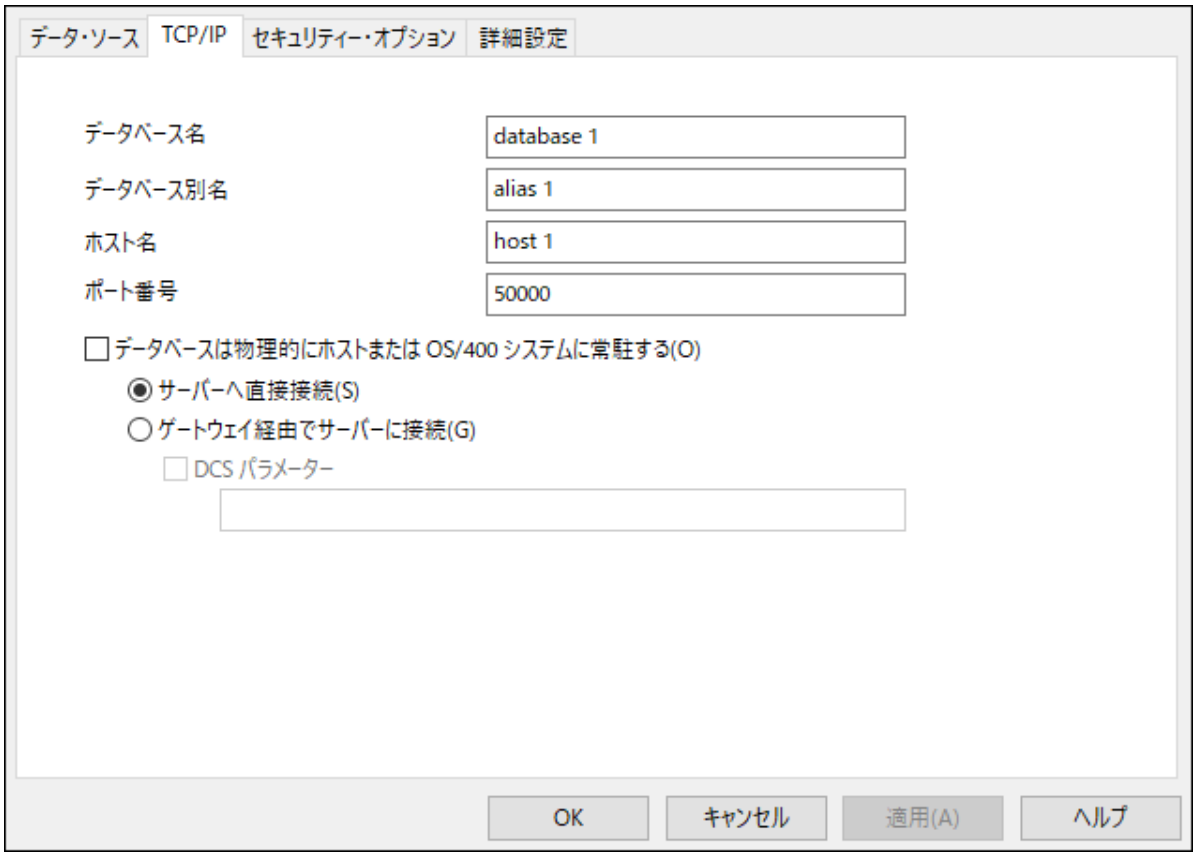
5. ソース名を入力します（このサンプルでは**DB2DSN**）「追加」をクリックします。



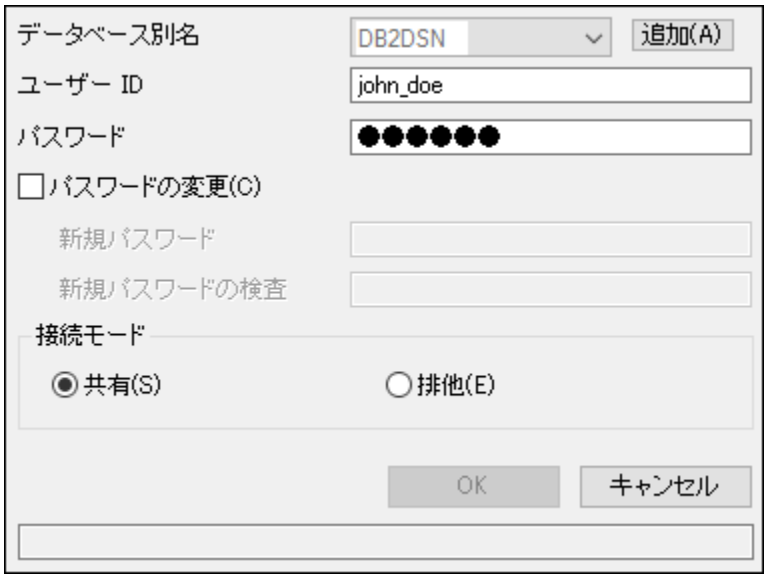
6. [**データソース**] タブで、データベースのユーザー名とパスワードを入力します。

7. **[TCP/IP]** タブで、データベース名、エイリアスの名前、ホスト名とポート名を入力して「OK」をクリックします。

(C) 2015-2021 Altova GmbH



8. ユーザ名とパスワードを再入力して「OK」をクリックします。



6.2.1.10.5 IBM DB2 for i (JDBC) への接続

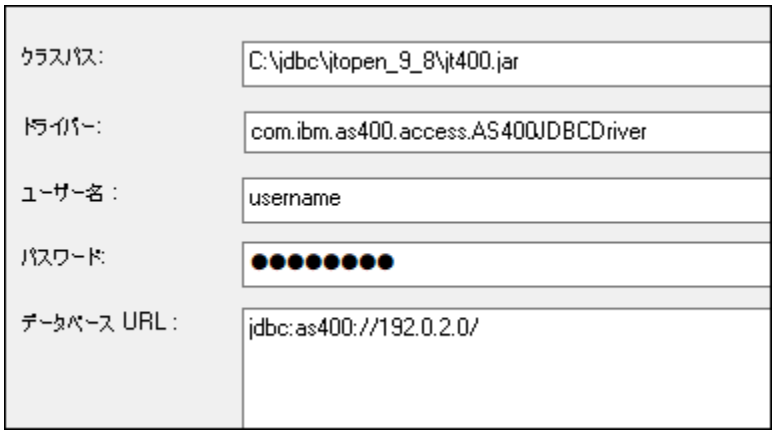
このサンプルではJDBC を介して**IBM DB2 for i** データベースサーバーに接続する方法について説明されています。

必要条件

- JRE (Java Runtime Environment) またはJava Development Kit (JDK) がインストールされている必要があります。これはOracle JDK または Oracle OpenJDK などのオープンソースビルドであることができます。MapForce は次のロケーションからJava Virtual Machine (JVM) へのパスを次の順序で決定します: a) アプリケーション オプション内で設定したカスタムJVM パス Java 設定を参照してください。b) W indows レジストリ内で見つかったJVM パス。c) `JAVA_HOME` 環境変数。
- MapForce (32-ビット、64-ビット) のプラットフォームがJRE/JDK のプラットフォームに一致することを確認してください。このサンプルはOracle のOpenJDK 11.0 64-bit を使用し、その結果 MapForce の64 ビットバージョンを使用します。
- JDBC ドライバー (データベースへの接続を提供する1 つ、または、複数の.jar ファイル) が使用中のオペレーティングシステム上で使用可能である必要があります。このサンプルは以下のオープンソースを使用します: **Toolbox for Java/JTOpen** バージョン 9.8 (http://jt400.sourceforge.net/)。パッケージをダウンロードし、ローカルディレクトリにアンパックすると、必要な.jar ファイルが**lib** サブディレクトリで使用できるようになります。
- 以下のデータベース接続の詳細が必要です: ホスト、ユーザー名、および パスワード

JDBC を使用してIBM DB2 for i に接続する方法:

1. データベース接続ウィザードを開始します。
2. 「JDBC 接続」をクリックします。
3. 「クラスパス」の横にデータベースへの接続を与える.jar ファイルへのパスを入力します。必要であれば、セミコロンで区切られた.jar ファイルパスのリストを入力することもできます。このサンプルでは、必要とされる.jar ファイルは次のパスを使用して見つけることができます: **C:\Informix_JDBC_Driver\lib\ifxjdbc.jar**。.jar ファイルパスをオペレーティングシステムのCLASSPATH 環境変数に追加した場合、「クラスパス」テキストボックスを空白のままにすることができます。(次も参照: CLASSPATH の構成)。
4. 「ドライバー」ボックスから、**com.ibm.as400.access.AS400JDBCDriver** を選択します。このエントリは、有効な.jar ファイルパスが、「クラスパス」テキストボックス内、または、オペレーティングシステムのCLASSPATH 環境変数内で検出された場合のみ使用することができます (前のステップを参照してください)。



5. データベースユーザーのユーザー名とパスワードを対応するテキストボックスに入力します。
6. JDBC 接続文字列をデータベースURL テキストボックスに入力します。`host` をデータベースサーバーのホスト名、または、IP アドレスと置き換えてください。

(C) 2015-2021 Altova GmbH

```
jdbc:as400://host
```

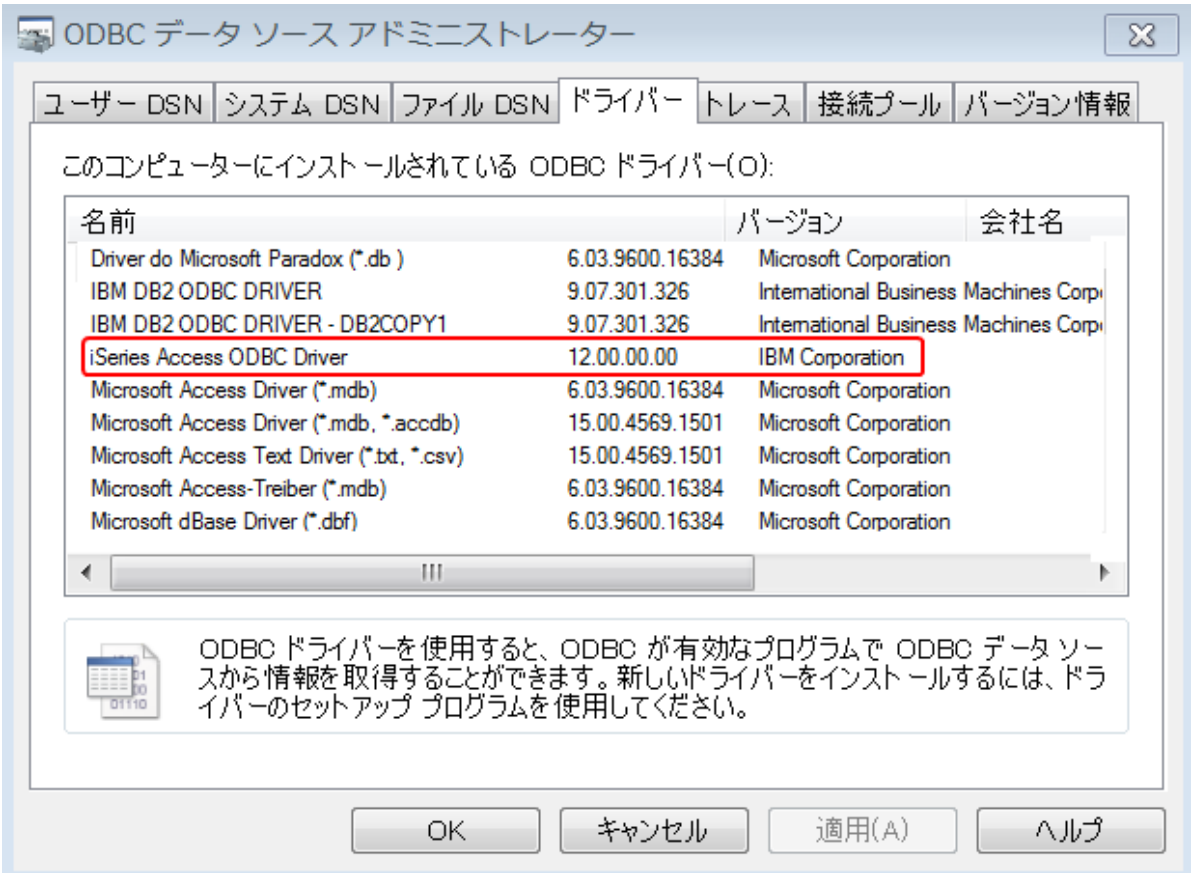
7. 「接続」をクリックします。

6.2.1.10.6 IBM DB2 for i (ODBC) への接続

このサンプルでは、ODBC を使用して *IBM DB2 for i* データベースへの接続について説明されています。

前提条件:

- *IBM System i Access for Windows* は使用中のオペレーティングシステム(このサンプルでは *IBM System i Access for Windows V6R1M0* を使用しています) にインストールされる必要があります。インストールの手順は、*IBM DB2 for i* ソフトウェアにより提供されているドキュメントを参照してください。インストール後、ODBC ドライバーが使用中のコンピューターで使用可能か確認してください (使用可能なODBC ドライバーの表示を参照)。



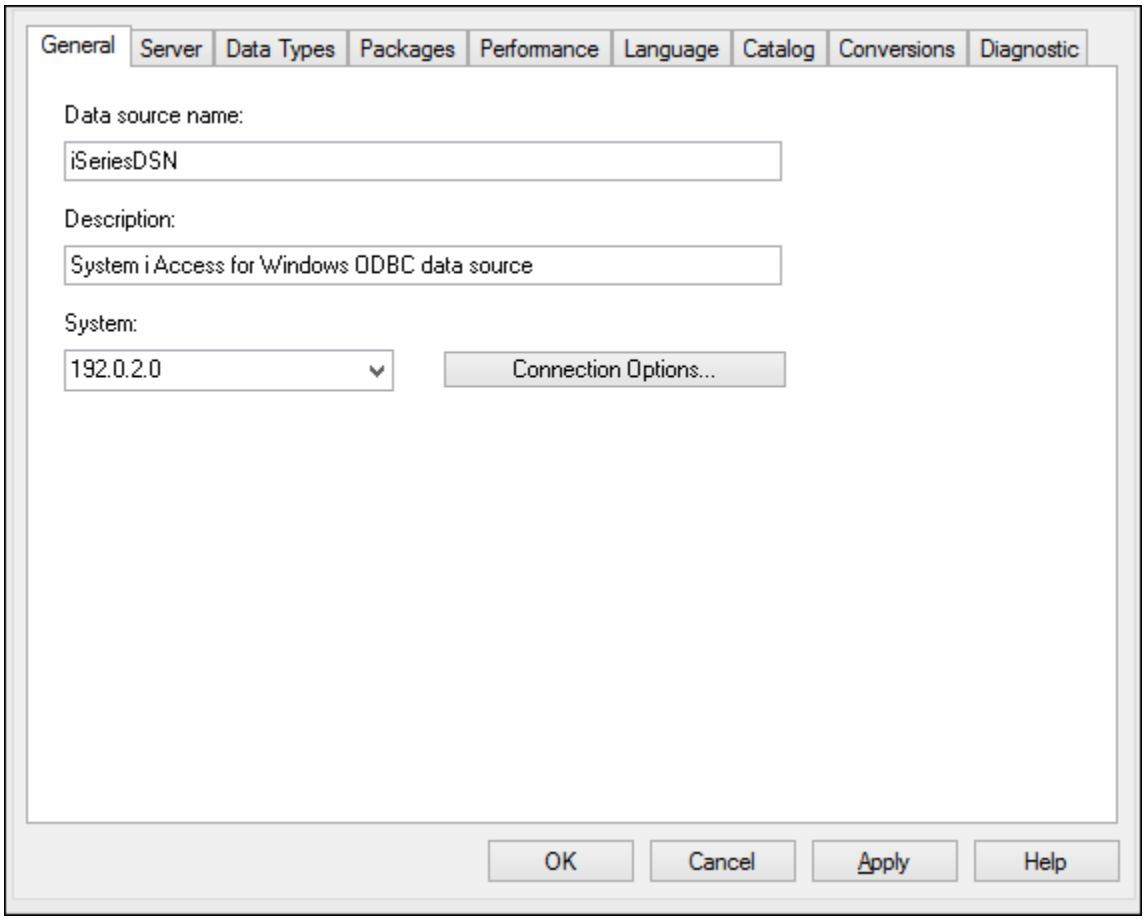
- データベース接続の詳細 データベースサーバーの I.P. アドレス、データベースユーザー名、および パスワード
- *i Navigator* を実行して、新規接続を作成するウィザードの手順を踏んで新規接続を作成してください。システムの指定をプロンプトされると、データベースサーバーのI.P. アドレスを入力してください。接続を作成した後、検証する(接続をクリックして、ファイル > 診断 > 接続の検証 を選択) ことが奨励されます。接続エラーが発生した場合、データベースサーバー管理者に連絡してください。

IBM DB2 for i への接続:

1. データベース接続ウィザードを開始します.
2. [ODBC 接続] をクリックします。
3. [ユーザー DSN] をクリックします。(または、[システム DSN] または、後に続く手順が同じ [ファイル DSN] をクリックします)。
4. 「追加」 をクリックします。
5. **[iSeries Access ODBC Driver]** をリストから選択します。[ユーザー DSN] (または、該当する場合 [システム DSN]) をクリックします。

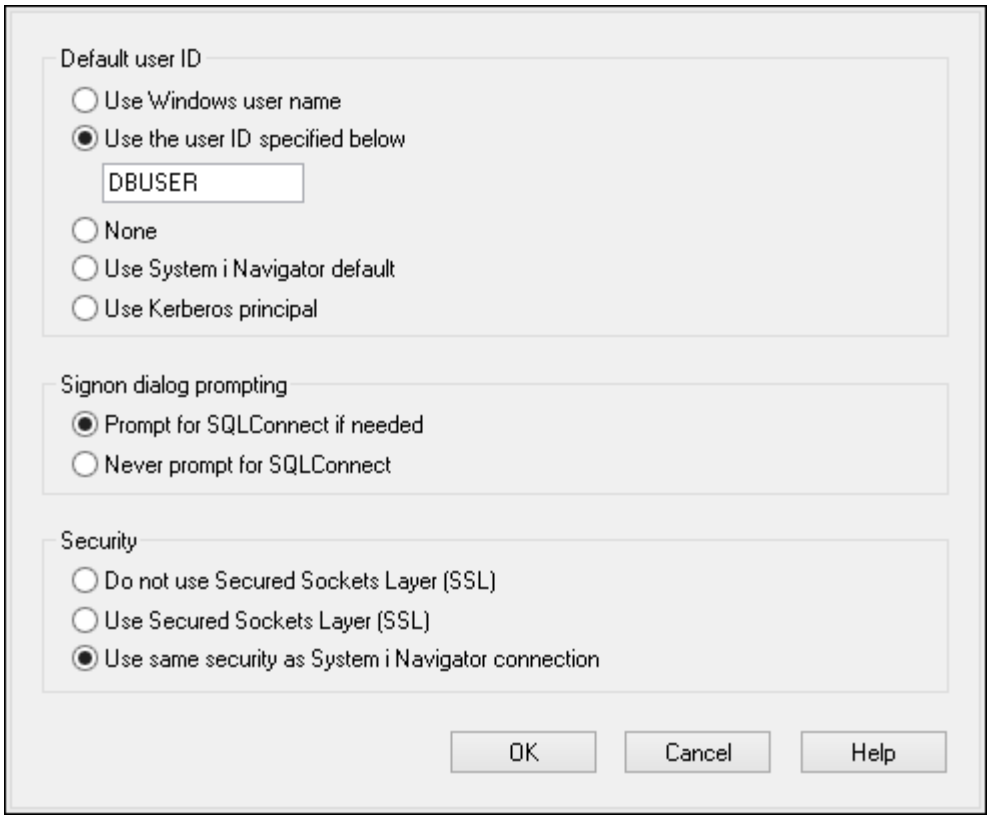
6. データソース名を入力して、システムコンボボックスから接続を選択します。このサンプルでは、データソース名は **iSeriesDSN** でシステムは **192.0.2.0** です。

(C) 2015-2021 Altova GmbH



メモ ODBC データソースを *IBM DB2 for i* database に追加する場合、クエリのタイムアウトを有効化するデフォルトのフラグが設定されます。この設定は、マッピングファイルを正確にロードするために、MapForce のために無効化されている必要があります。設定を無効化するには、**パフォーマンス** タブから、**詳細**をクリックして、**クエリのタイムアウトの許可** チェックボックスをクリアします。

7. [**接続オプション**] をクリックします。[**指定されたユーザー ID を使用**] を選択して、データベースユーザー（このサンプル内では **DBUSER**）の名前を入力します。



8. 「OK」をクリックします。DSN 内のリストで新しいデータソースを使用することができます。
9. 「接続」をクリックします。
10. データベースのユーザー名とパスワードをプロンプトされると入力して「OK」をクリックします。

6.2.1.10.7 IBM Informix (JDBC) への接続

このサンプルでは、JDBC を使用して IBM Informix データベースサーバーに接続する手順のサンプルについて説明されています。

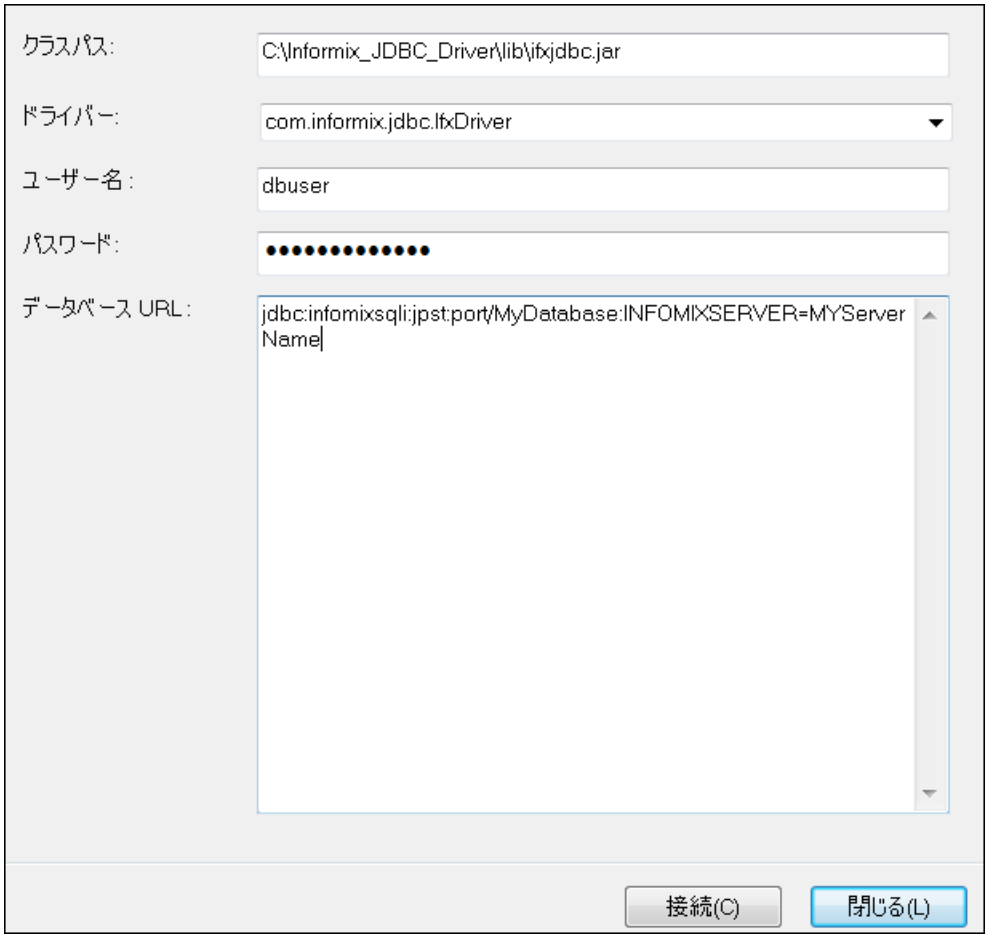
前提条件:

- JRE (Java Runtime Environment) または Java Development Kit (JDK) がインストールされている必要があります。これは Oracle JDK または Oracle OpenJDK などのオープンソースビルドであることができます。MapForce は次のロケーションから Java Virtual Machine (JVM) へのパスを次の順序で決定します: a) アプリケーションオプション内で設定したカスタム JVM パス (Java 設定を参照してください)。b) Windows レジストリ内で見つかった JVM パス。c) `JAVA_HOME` 環境変数。
- MapForce (32-ビット、64-ビット) のプラットフォームが JRE/JDK のプラットフォームに一致することを確認してください。
- (データベースへの接続を与える 1 つ、または、複数の .jar ファイル) JDBC ドライバーが使用中のオペレーティングシステムで使用可能である必要があります。この例では、IBM Informix JDBC ドライバーバージョン 3.70 が使用されています。ドライバーのインストールの説明に関しては、ドライバーに付随するドキュメント、または、「IBM Informix JDBC ドライバプログラマーズガイド」を参照してください)。
- データベース接続の詳細: ホスト、サーバーの名前、データベース、ポート、ユーザー名、および パスワード

(C) 2015-2021 Altova GmbH

JDBC を使用して IBM Informix に接続する

1. データベース接続ウィザードを開始します.
2. 「JDBC 接続」をクリックします。
3. 「クラスパス」の横にデータベースへの接続を与える .jar ファイルへのパスを入力します。必要であれば、セミコロンで区切られた .jar ファイルパスのリストを入力することもできます。このサンプルでは、必要とされる .jar ファイルは次のパスを使用して見つけることができます: **C:\Informix_JDBC_Driver\lib\ifxjdbc.jar**。.jar ファイルパスをオペレーティングシステムの CLASSPATH 環境変数に追加した場合、「クラスパス」テキストボックスを空白のままにすることができます。(次も参照: CLASSPATH の構成)。
4. 「ドライバー」ボックスから、**com.informix.jdbc.IfxDriver** を選択します。このエントリは、有効な .jar ファイルパスが、「クラスパス」テキストボックス内、または、オペレーティングシステムの CLASSPATH 環境変数内で検出された場合のみ使用することができます(前のステップを参照してください)。



5. データベースへのユーザー名とパスワードを対応するテキストボックスに入力します。
6. ハイライトされた値を、使用するデータベースサーバーに適用する値と置き換えて、データベース URL テキストボックス内にデータベースサーバーへの接続文字列を入力します。

```
jdbc:informix-sqli://hostName:port/databaseName:INFORMIXSERVER=myserver;
```

7. 「接続」をクリックします。

6.2.1.10.8 MariaDB (ODBC) への接続

このサンプルでは、ODBC を使用したMariaDB データベースへの接続方法について説明されています。

前提条件:

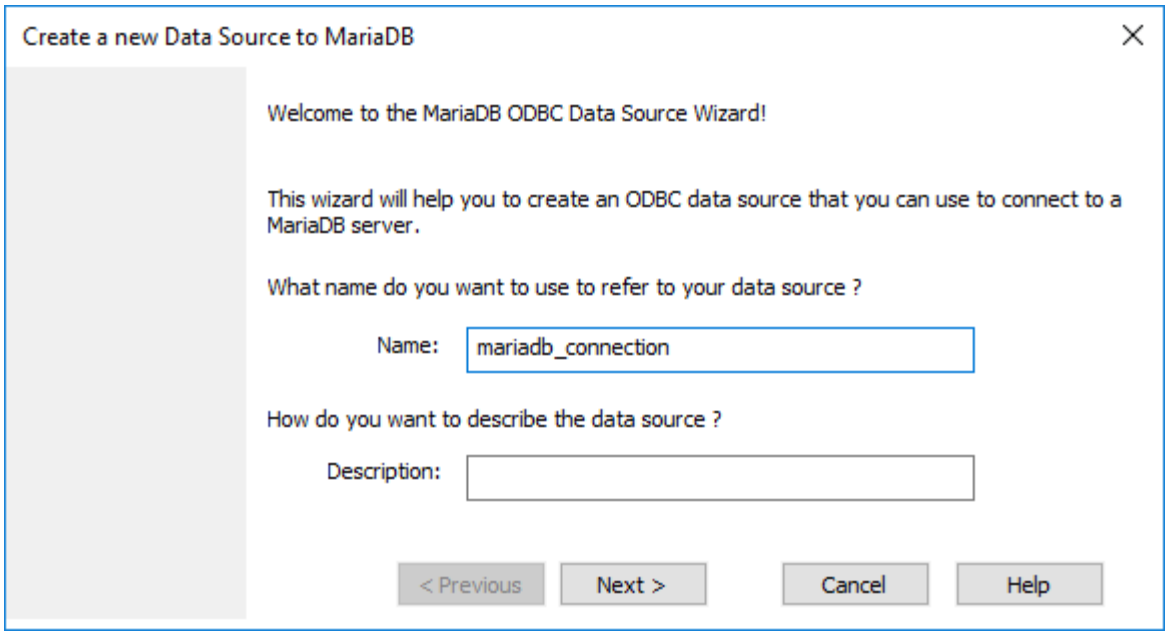
- MariaDB コネクタ/ODBC (https://downloads.mariadb.org/connector-odbc/) がインストールされている必要があります。
- 次のデータベース接続詳細が存在します: ホスト、データベース、ポート、ユーザー名、および、パスワード。

ODBC を使用したMariaDB への接続の方法:

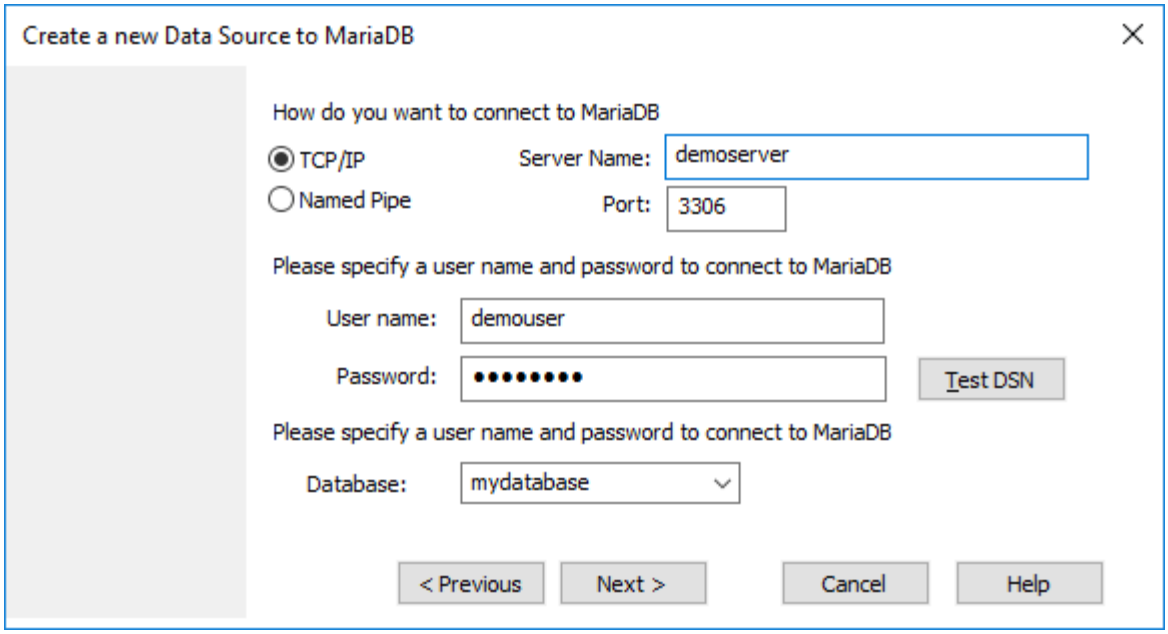
1. データベース接続ウィザードを開始します。
2. 「MariaDB (ODBC)」を選択して、「次へ」をクリックします。

3. 「ドライバーを使用して新規のデータソース名 (DSN) を作成する」を選択し、「MariaDB ODBC 3.0 Driver」を選択します。このようなドライバーがリスト内で使用できない場合、「ドライバーの編集」をクリックし、使用することのできるMariaDB ドライバーを選択します (リストには使用中のオペレーティングシステム上にインストールされているすべてのODBC ドライバーが含まれています)。
4. 「接続」をクリックします。

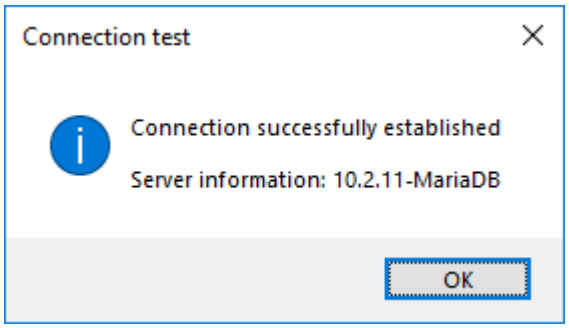
(C) 2015-2021 Altova GmbH



5. 名前を入力し、任意で、このODBC データソースを将来識別できる詳細を与えます。



6. データベース接続資格情報(TCP/IP サーバー、ユーザー、パスワード)を入力し、データベースを選択して、「Test DSN」をクリックします。接続に成功すると、メッセージが表示されます:



7. 「次へ」をクリックしてウィザードを完了します。他のパラメータが必要になる場合がありますが（例えば、安全な接続を使用して、MariaDB に接続する場合は、SSL 証明書が必要になります）。

メモ　データベースサーバーがリモートの場合、サーバー管理者が遠隔の接続を使用中のマシンの IP アドレスから受け入れるように構成する必要があります。

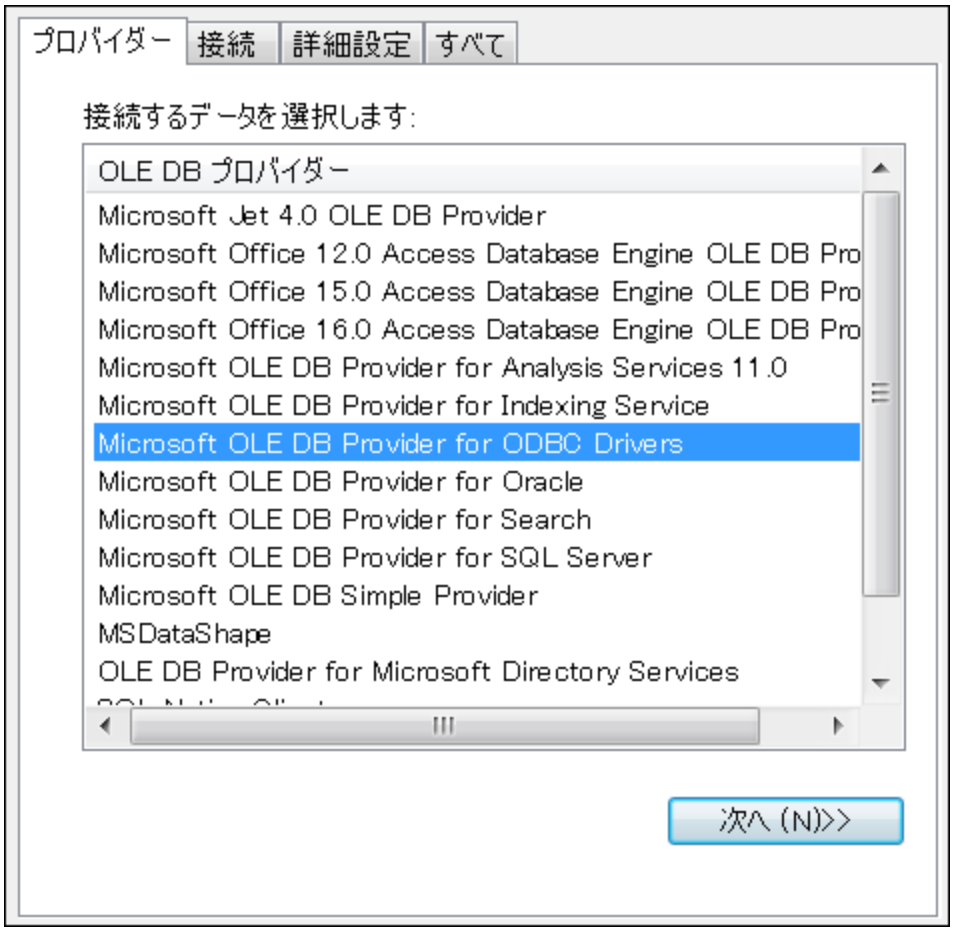
6.2.1.10.9 Microsoft Access (ADO) への接続

Microsoft Access データベースに接続するには、既存の Microsoft Access データベースに接続するで説明されているとおり、データベースファイルのためのウィザードとブラウザーの手順を踏みます。別の方法としては、このトピックで説明されているとおり、明確的に ADO 接続をセットアップすることです。このアプローチは、データベースがパスワードで保護されている場合、役に立ちます。

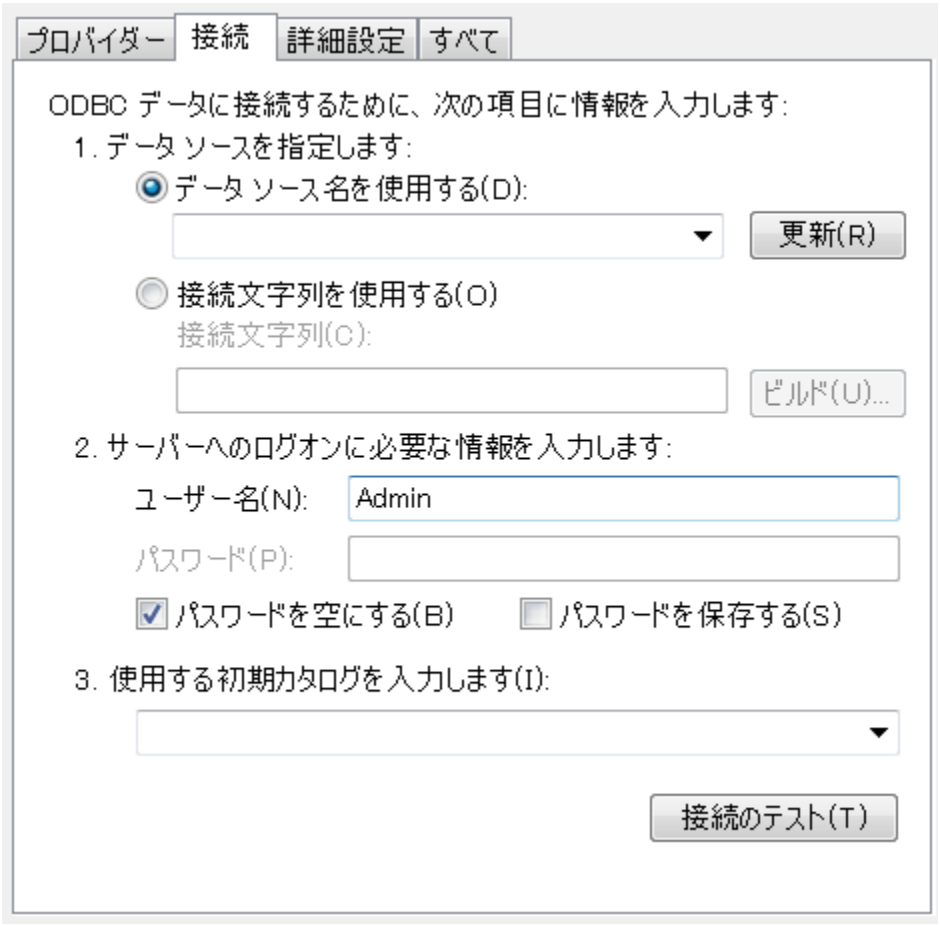
ODBC 接続を使用して、Microsoft Access に接続することもできますが、このシナリオには限度があるため、奨励されません。

パスワードで保護されている Microsoft Access データベースに接続する:

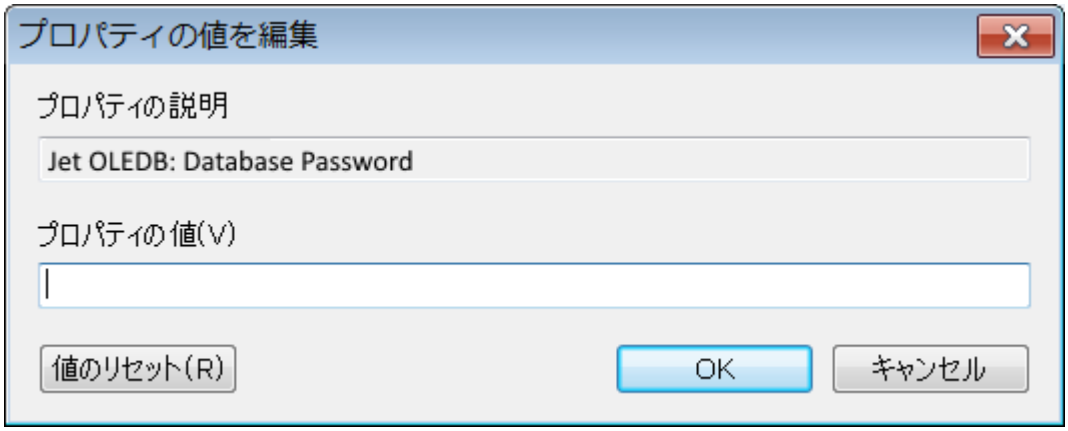
1. データベース接続ウィザードを開始します.
2. [ADO 接続] をクリックします。
3. [構築] をクリックします。



4. **[Microsoft Office 15.0 Access Database Engine OLE DB Provider]** を選択して、「次へ」をクリックします。



5. データソースボックス内でUNC フォーマットのMicrosoft Access ファイルへのパスを入力します。例えば、**myserver** がサーバーの名前であり **mynetworkshare** が共有するネットワークの名前である箇所での **\\myserver\mynetworkshare\Reports\Revenue.accdb**。
6. 全てタブの **Jet OLEDB:Database Password** パスワード プロパティをダブルクリックし、データベースパスワードをプロパティの値として入力します。



メモ また接続できない場合、ワーキンググループ情報ファイル(**System.MDW**) を検索して、ユーザープロフィールに適用できるようにします。**Jet OLEDB: System database** プロパティの値を **System.MDW** ファイルのパスに設定します。

(C) 2015-2021 Altova GmbH

6.2.1.10.10 Microsoft SQL Server (ADO) への接続

このサンプルでは、ADO を使用したSQL Server データベースへの接続方法について説明されています。 Https://docs.microsoft.com/en-us/sql/connect/oledb/download-oledb-driver-for-sql-server?view=sql-server-ver15 でダウンロード可能な奨励される **Microsoft OLE DB Driver for SQL Server (MSOLEDBSQL)**を使用する場合に適用される説明です。

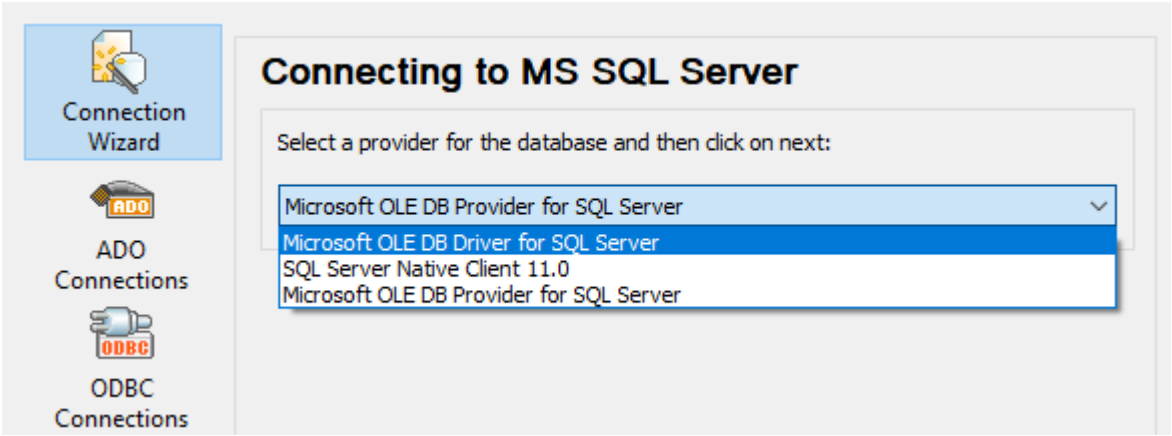
これらの説明を読む前に、ワークステーションに上記のプロバイダーがダウンロードされインストールされていることを確認してください。ADO プロバイダーはMapForce (32-ビット または64-ビット)のプラットフォームに一致する必要があることに注意してください。

SQL Server Native Client (SQLNCLI) または**Microsoft OLE DB Provider for SQL Server (SQLOLEDB)** などの他のADO プロバイダーを使用する場合、手順は同様ですが、これらのプロバイダーは奨励されていません。奨励されていないプロバイダーへの接続を成功させるには、SQL Server データリンクプロパティのセットアップで説明されている追加接続プロパティを設定する必要があります。

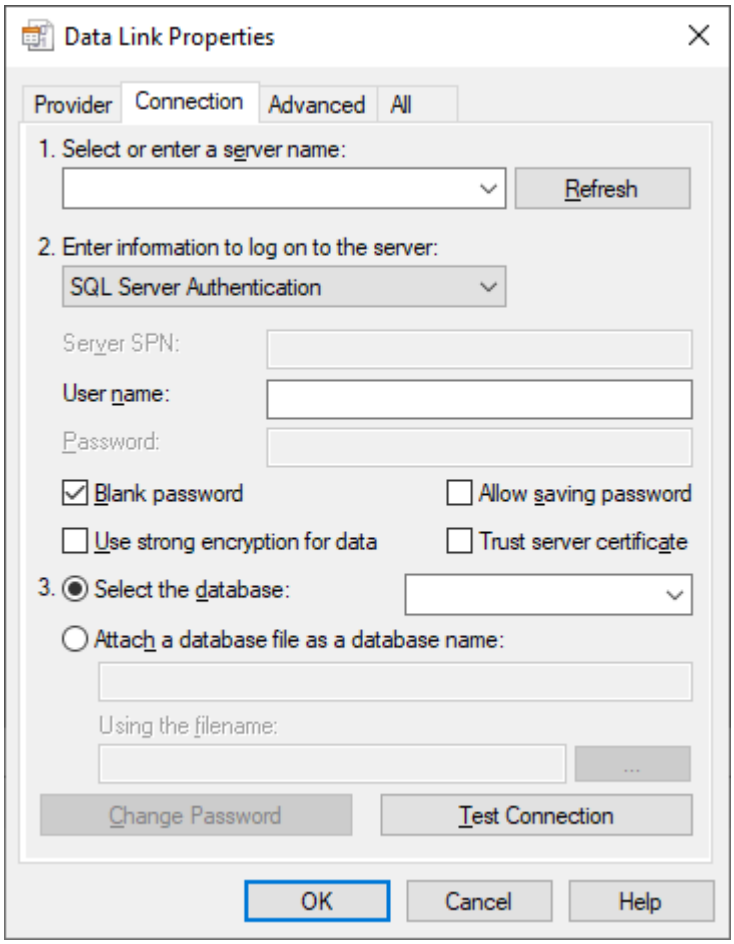
Microsoft OLE DB Provider for SQL Server (SQLOLEDB) は共通テーブル式 (CTE) とネストされたSELECT ステートメントなどの複雑なクエリのパラメーターバインディングに関する問題があることが知られています。

SQL サーバーに接続する方法:

1. データベース接続ウィザードを開始します。
2. **[Microsoft SQL Server (ADO)]** を選択して、[次へ] をクリックします。使用することのできるADO プロバイダーのリストが表示されます。このサンプルでは**Microsoft OLE DB Driver for SQL Server** が使用されています。リストに存在しない場合、上記の通りコンピューターに接続されていることを確認してください。

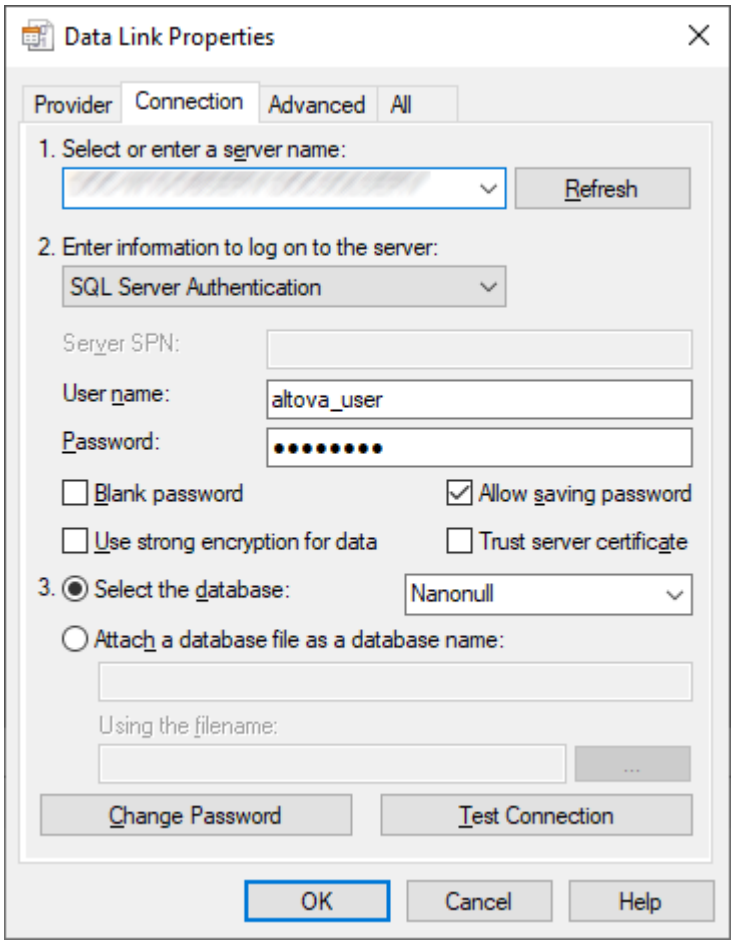


3. 「次へ」をクリックします。データリンクプロパティダイアログボックスが表示されます。



4. データベースサーバーの名前を選択、または入力します。例えば、**SQLSERV01**。名前を持つSQL Server *インスタンス*に接続する場合、サーバー名は**SQLSERV01\SOMEINSTANCE** になります。
5. データベースサーバーがW indows ドメイン上の認証されているユーザーの接続を許可している場合、「**W indows 認証**」を選択してください。それ以外の場合は、[**SQL Server 認証**]を選択し、[**空のパスワード**] チェックボックスをクリアし、対応するボックスにデータベース証明書を入力します。
6. **パスワードの保存の許可** チェックボックスを選択し、接続するデータベースを選択します（このサンプル内では、Nanonull です）。

(C) 2015-2021 Altova GmbH



7. この時点で接続をテストする場合は、[**接続のテスト**]をクリックします。これは任意の奨励されるステップです。
8. **[OK]**をクリックします。

6.2.1.10.11 Microsoft SQL Server (ODBC) への接続

このサンプルでは、ODBC を使用したSQL Server データベースへの接続方法について説明されています。

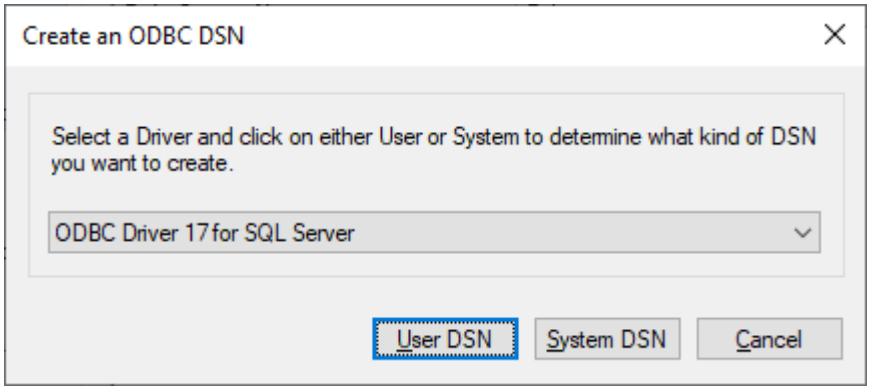
必要条件:

- **Microsoft ODBC Driver for SQL Server** をMicrosoft W eb サイトからダウンロードしてインストールします。https://docs.microsoft.com/en-us/SQL/connect/odbc/download-odbc-driver-for-sql-server を参照してください。このサンプルは**Microsoft ODBC Driver 17 for SQL Server** を**SQL Server 2016** データベースに接続するために使用しています。接続するSQL Server のバージョンにより異なるODBC ドライバーバージョンをダウンロードする必要がある場合があります。使用中のSQL Server によりサポートされるODBC ドライバーバージョンに関する情報は、システムの必要条件を参照してください。

ODBC を使用してのSQL Server への接続:

1. データベース接続ウィザードを開始します。

2. [ODBC 接続] をクリックします。
3. [ユーザーDSN] を選択して、(または、管理者の特権がある場合、 [システムDSN] を選択して、)[追加] をクリックします。
4. ドライバーをリストから選択します。インストール後にのみドライバーはリスト内に表示されることにご注意してください。

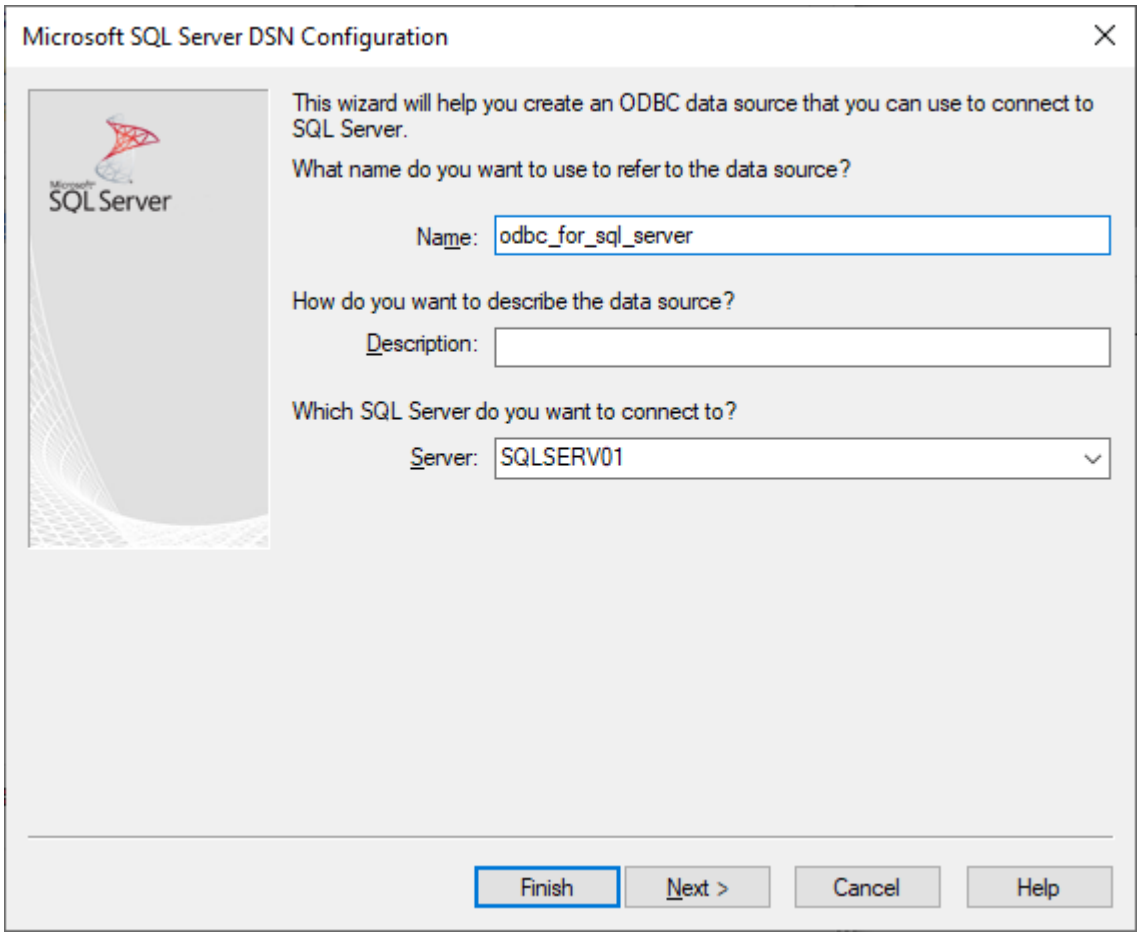


5. [ユーザーDSN] (またはシステムDSN を作成する場合) [システムDSN]をクリックします。

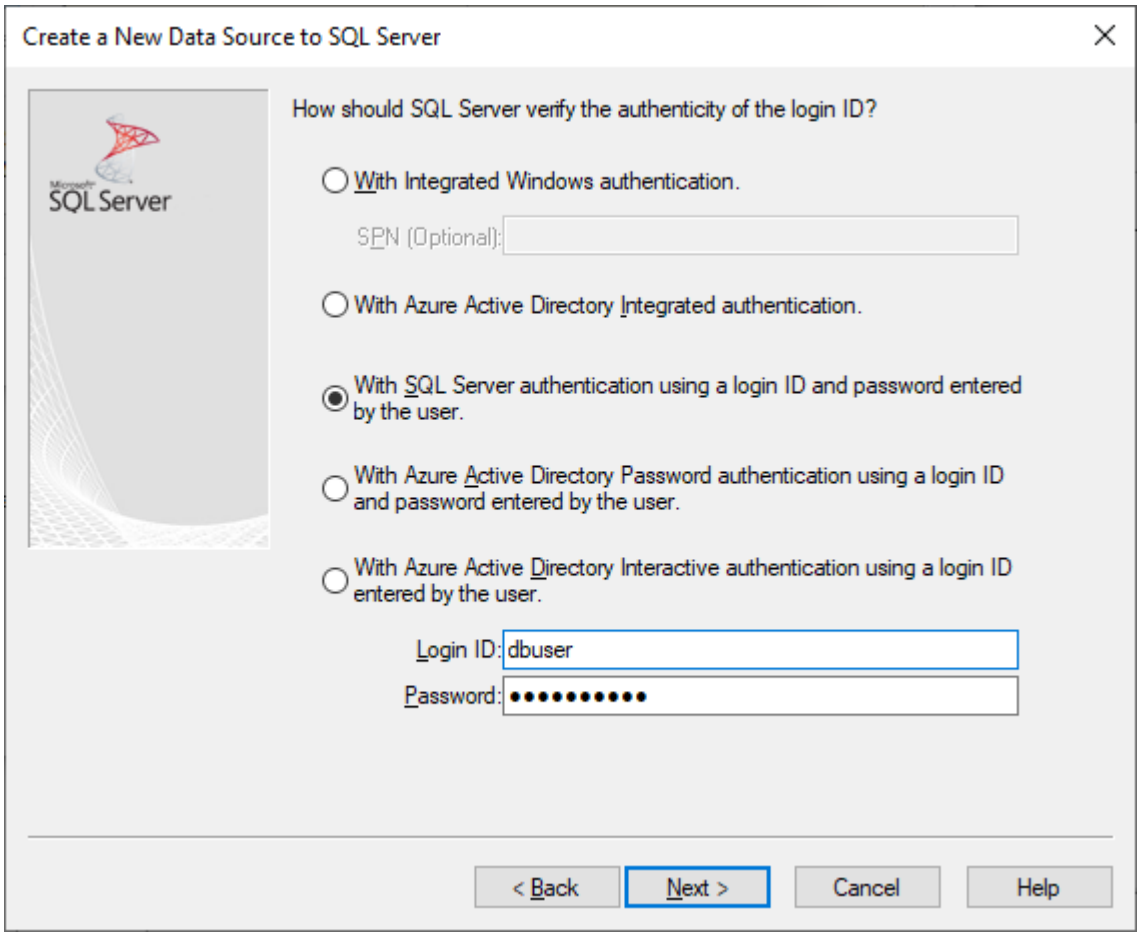
> MapForce が管理者として作動することを必要とするシステムDSN の作成方法このため**System DSN** を作成するには、ウィザードをキャンセルし、MapForce が管理者として作動していることを確認し、上記のステップを再度実行します。

6. 名前を入力して、任意でこの接続のIDの説明を入力し、接続するSQL Server (この場合 **SQLSERV01**)をリストから選択します。.

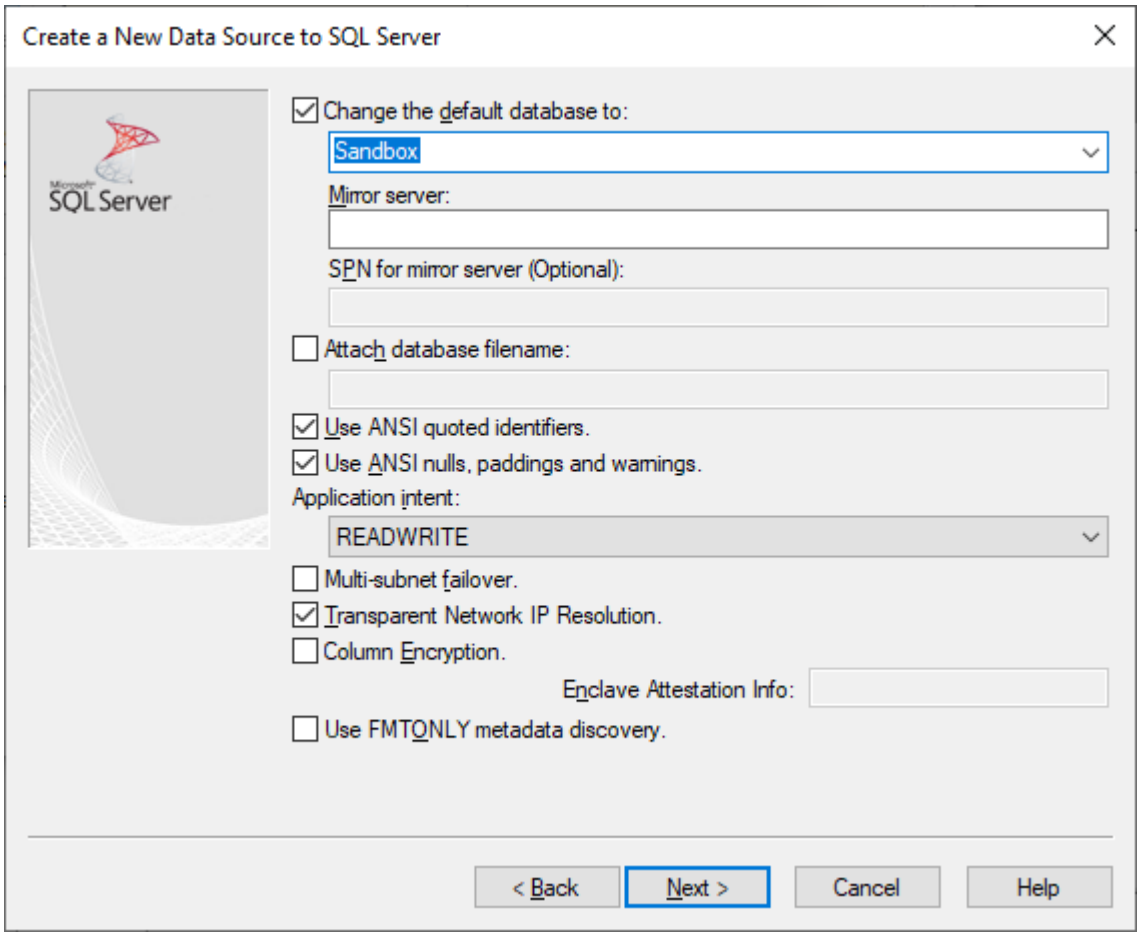
(C) 2015-2021 Altova GmbH



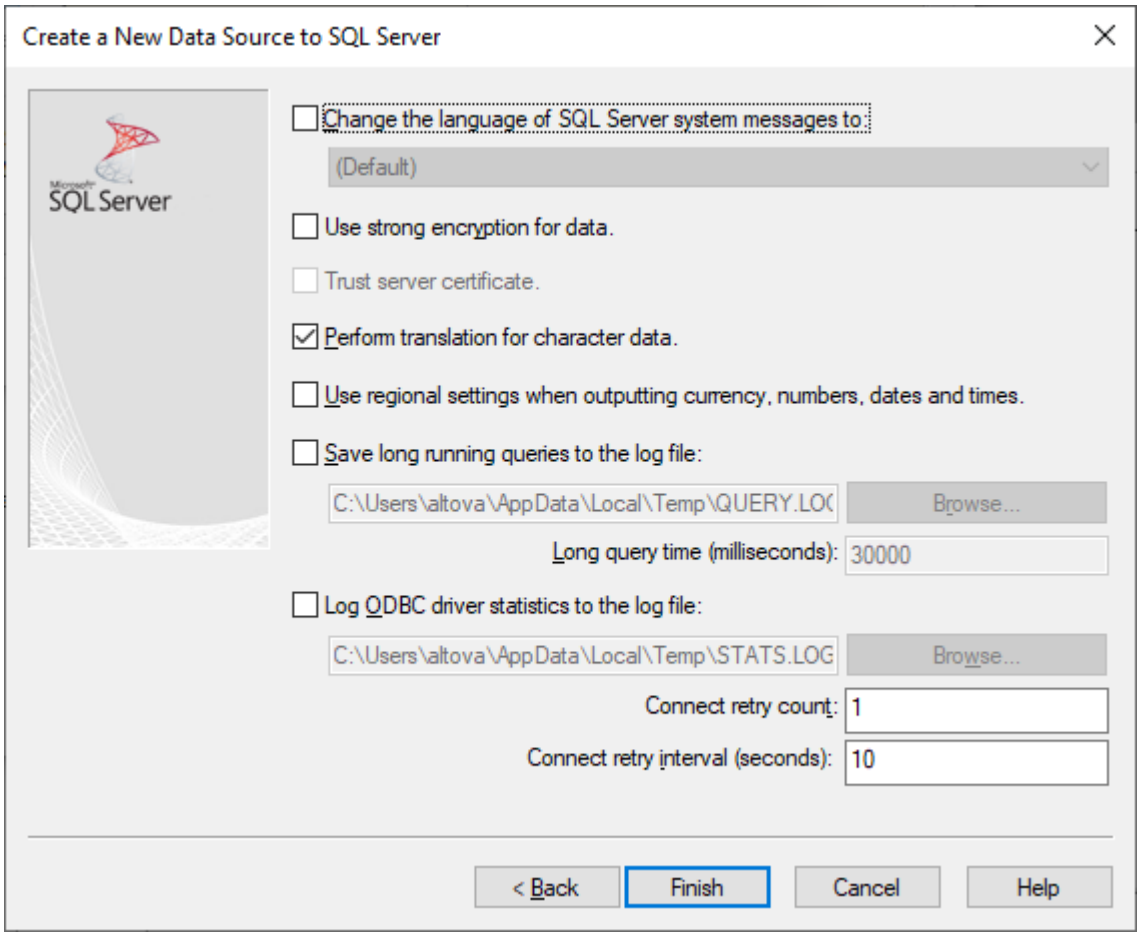
7. データベースサーバーがWindowsドメイン上の認証されているユーザーの接続を許可している場合、「**統合されたWindows認証**」を選択してください。それ以外の場合、適用可能なオプションから1つを選択してください。このサンプルは **SQL Server認証**...を使用しています。これはユーザー名とパスワードが関連するボックスに入力されることを必要としています。



8. **オプション**で **デフォルトのデータベースを変更** チェックボックスを選択して、接続するデータベースの名前を入力します（このサンプルでは**Sandbox**）。

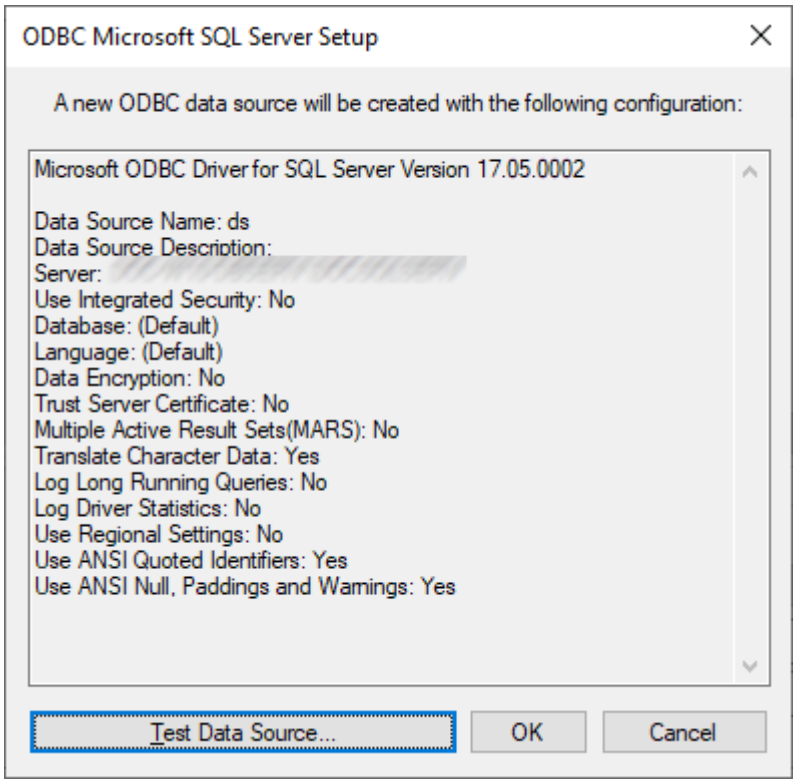


9. [次へ] をクリックして、任意で、この接続のための追加パラメーターを構成します。

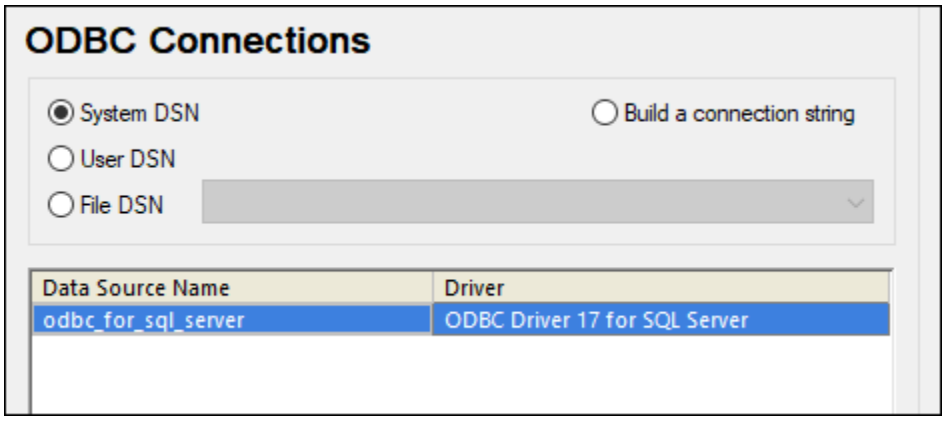


10. 「完了」をクリックします。接続の詳細をリストした確認ダイアログボックスが開かれます。

(C) 2015-2021 Altova GmbH



11. **[OK]**をクリックします。構成された通り ユーザー または システム データソースのリスト内でデータソースが表示されます。例:



6.2.1.10.12 MySQL (ODBC) への接続

このサンプルではODBC ドライバーを使用して、W indows マシンから MySQL データベースへ接続する方法について説明されています。W indows からは、MySQL ODBC ドライバーを入手することができませんので、個別にダウンロードおよびインストールする必要があります。このサンプルでは、MySQL コネクタ/ODBC 8.0 を使用しています。

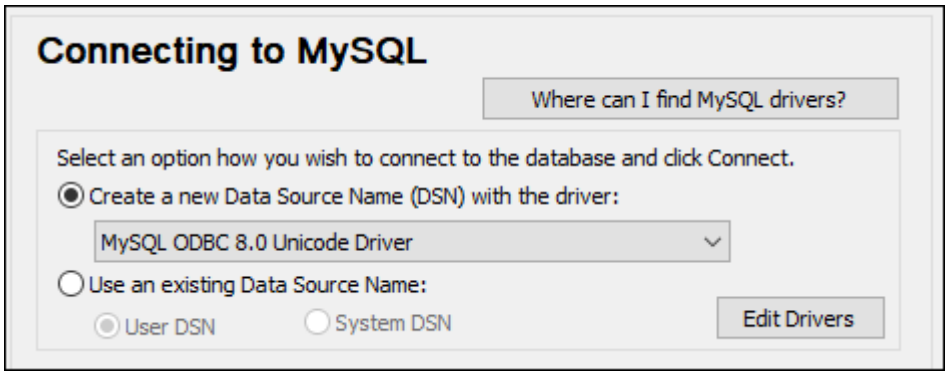
前提条件:

- MySQL ODBC ドライバーをオペレーティングシステム上にインストールする必要があります。使用するデータベースサーバーバージョンのために奨励されるドライバーバージョンのためのMySQLドキュメントを確認してください (https://dev.mysql.com/downloads/connector/odbc/ を確認してください)。
- データベース接続の詳細: ホスト、データベース、ユーザー名およびパスワード

64-ビットプラットフォームのためにMySQL Connector/ODBC をインストールしている場合、64-ビットプラットフォームのために MapForce をインストールしてください。

ODBC を介して MySQL に接続する方法:

1. データベース接続ウィザードを開始します。
2. **MySQL (ODBC)** を選択して、「次へ」をクリックします。

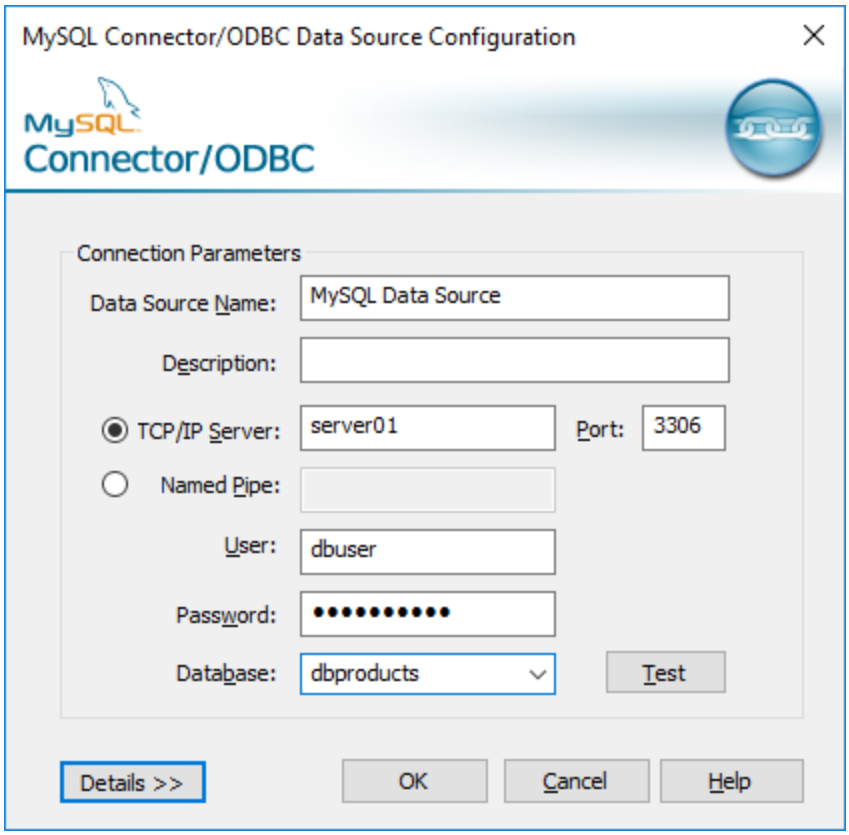


3. [ドライバーを使用して新しいデータソース名 (DSN) を作成] を選択して、MySQLドライバーを選択します。MySQLドライバーがリストにない場合は、[ドライバーを編集] をクリックして、使用可能なMySQLドライバーを選択します (使用中のオペレーティングシステムにインストールされているすべてのODBC ドライバーがリストされています)。

MapForce 64-ビットをインストールすると、64-ビット ODBC ドライバーがリスト内に表示されます。それ以外の場合、32-ビット ODBC ドライバーが表示されます。使用可能な ODBC ドライバーの確認も参照してください。

4. 「接続」をクリックします。

(C) 2015-2021 Altova GmbH



5. データソース名ボックスの中に、今後このODBC データソースを認識するために役に立つ詳細を入力します。
6. データベース接続資格情報（TCP/IP サーバー、ユーザー、パスワード）を入力して、データベースを選択し、「OK」をクリックします。

メモ　データベースサーバーがリモートの場合、サーバー管理者が使用中のコンピューターのIP アドレスからのリモート接続を承諾するように構成される必要があります。また、[詳細>>] をクリックすると使用することのできる追加パラメーターが表示されます。デフォルトの値を変更する前に、ドライバーのドキュメントを確認してください。

6.2.1.10.13 Oracle (JDBC) への接続

JDBC インターフェイスを使用して、クライアントマシンから、Oracle データベースサーバーへの接続の方法を説明します。接続は、Oracle W eb サイトから使用することのできる、**Oracle Instant Client Package (Basic)** を使用する、Java 接続として作成されます。接続の型の利点は、Oracle Instant Client Package により与えられる Java 環境と .jar ライブラリのみを必要とし、更に複雑なデータベースクライアントをインストールする手間を省くことができる点です。

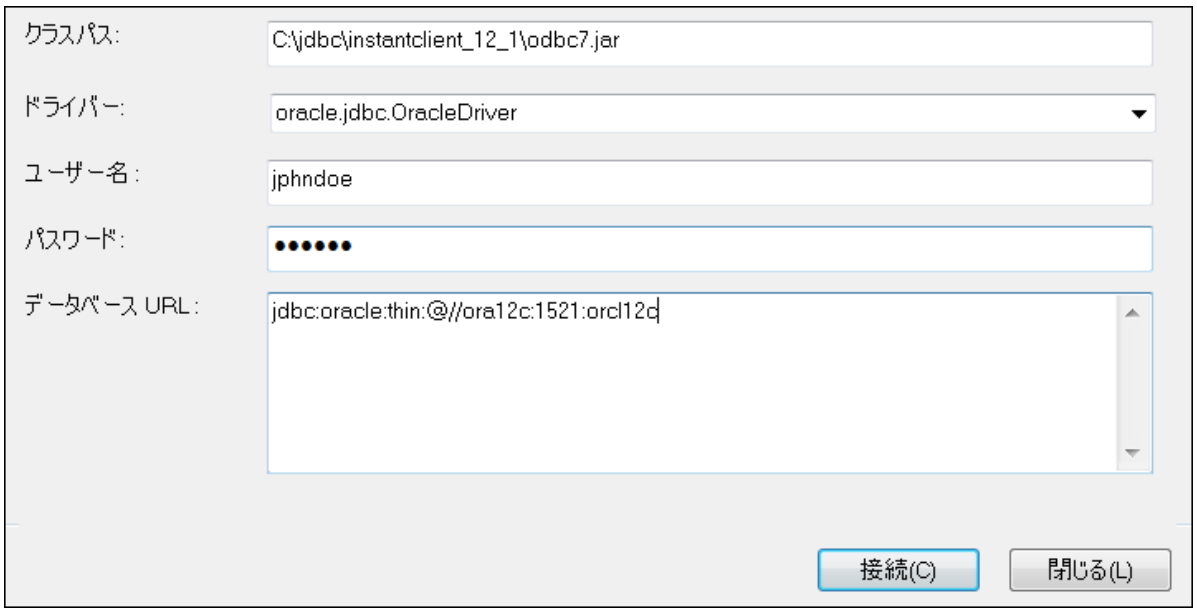
前提条件:

- JRE (Java Runtime Environment) または Java Development Kit (JDK) がインストールされている必要があります。これは Oracle JDK または Oracle OpenJDK などのオープンソースビルドであることができます。MapForce は次のロケーションから Java Virtual Machine (JVM) へのパスを次の順序で決定します: a) アプリケーション オプション内で設定したカスタム JVM パス Java 設定を参照してください。b) W indows レジストリ内で見つかった JVM パス。c) `JAVA_HOME` 環境変数。
- MapForce (32-ビット、64-ビット) のプラットフォームが JRE/JDK のプラットフォームに一致することを確認してください。

- **Oracle Instant Client Package (Basic)** が使用中のオペレーティングシステムで使用可能である必要があります。パッケージは、Oracle の公式 W eb サイトからダウンロードすることができます。このサンプルは、W indows 32-ビットのための Oracle Instant Client Package バージョン 12.1.0.2.0 、そして結果的に Oracle JDK 32-ビットを使用します。
- データベース接続の詳細があります: ホスト、データベース パス、または エイリアス、ユーザー名、および パスワード。

クライアント パッケージ を使用して Oracle に接続する

1. データベース接続ウィザードを開始します。
2. 「JDBC 接続」をクリックします。
3. 「クラス パス」の横にデータベースへの接続を与える .jar ファイルへの パスを入力します。必要であれば、セミコロンで区切られた .jar ファイル パスのリストを入力することもできます。このサンプルでは、必要とされる .jar ファイルは次の パスを使用して見つけることができます: **C:\jdbc\instantclient_12_1\ojdbc7.jar**。.jar ファイル パスをオペレーティングシステムの CLASSPATH 環境変数に追加した場合、「クラス パス」テキスト ボックスを空白のままにすることができます。(次も参照: CLASSPATH の構成)。
4. 「ドライバー」ボックスから、**oracle.jdbc.OracleDriver** または **oracle.jdbc.driver.OracleDriver** を選択します。このエントリは、有効な .jar ファイル パスが、「クラス パス」テキスト ボックス内、または、オペレーティングシステムの CLASSPATH 環境変数内で検出された場合のみ使用することができます (前のステップを参照してください)。
5. データベースへのユーザー名と パスワードを対応するテキスト ボックスに入力します。



6. ハイライトされた値を、使用するデータベース サーバーに適用する値と置き換えて、データベース URL テキスト ボックス内にデータベース サーバーへの接続文字列を入力します。

```
jdbc:oracle:thin:@//host:port:service
```

7. 「接続」をクリックします。

6.2.1.10.14 Oracle (ODBC) への接続

このサンプルでは、ローカル オペレーティング システム上にインストールされている Oracle データベース クライアントを介して、ネットワーク マシン上で MapForce から Oracle データベース サーバーへの接続のよく使用されている状況について説明されています。

(C) 2015-2021 Altova GmbH

このサンプルでは、MapForce 内のデータベース接続ウィザードを使用して、ODBC データソース（DSN）のセットアップについて説明されています。既にDSN が作成されている場合は Windows 内の**ODBC データソース管理者**から直接作成しプロンプトされたら選択する事も可能です。ODBC データソースの詳細に関しては、 ODBC 接続のセットアップを参照してください。

必要条件:

- （ODBC Oracle ドライバーを含む）Oracle データベースクライアントが使用中のオペレーティングシステムでインストール、および構成されていなければなりません。Oracle データベースクライアントのインストールおよび構成方法は Oracle ソフトウェアに付随するドキュメントを参照してください。
- **tnsnames.ora** ファイルのパスは Oracle ホームディレクトリがインストールされている場所によります。Oracle データベースクライアント 11.2.0 に関しては、デフォルト Oracle ホームディレクトリのパスは以下の通りです:

```
ORCL =
  (DESCRIPTION =
    (ADDRESS_LIST =
      (ADDRESS = (PROTOCOL = TCP)(HOST = server01)(PORT = 1521))
    )
    (CONNECT_DATA =
      (SID = orcl)
      (SERVER = DEDICATED)
    )
  )
```

tnsnames.ora ファイルのパスは Oracle ホームディレクトリがインストールされている場所によります。Oracle データベースクライアント 11.2.0 に関しては、デフォルト Oracle ホームディレクトリのパスは以下の通りです:

```
C:\app\username\product\11.2.0\client_1\network\admin\tnsnames.ora
```

接続の詳細を貼り付け保存する、または使用可能であれば Oracle *ネット構成アシスタントウィザード* を実行することにより **tnsnames.ora** ファイルに新規エントリを追加することができます。

ODBC を使用しての Oracle への接続:

1. データベース接続ウィザードを開始します.
2. **[Oracle (ODBC / JDBC)]** を選択して、「次へ」をクリックします。

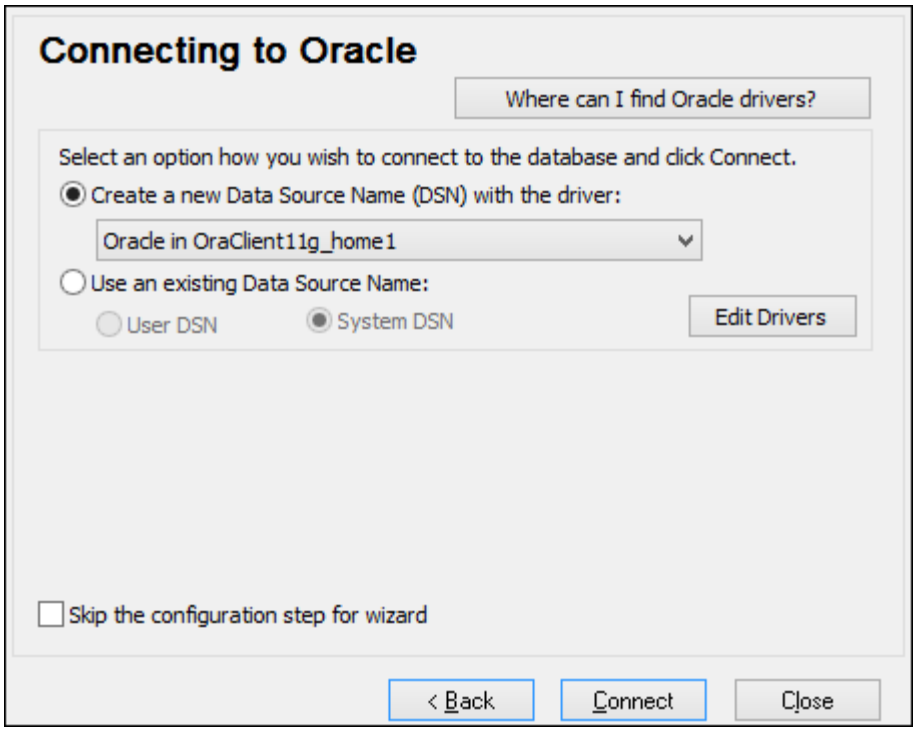
3. **[ODBC]** を選択します。

(C) 2015-2021 Altova GmbH

4. [ドライバーの編集] をクリックします。

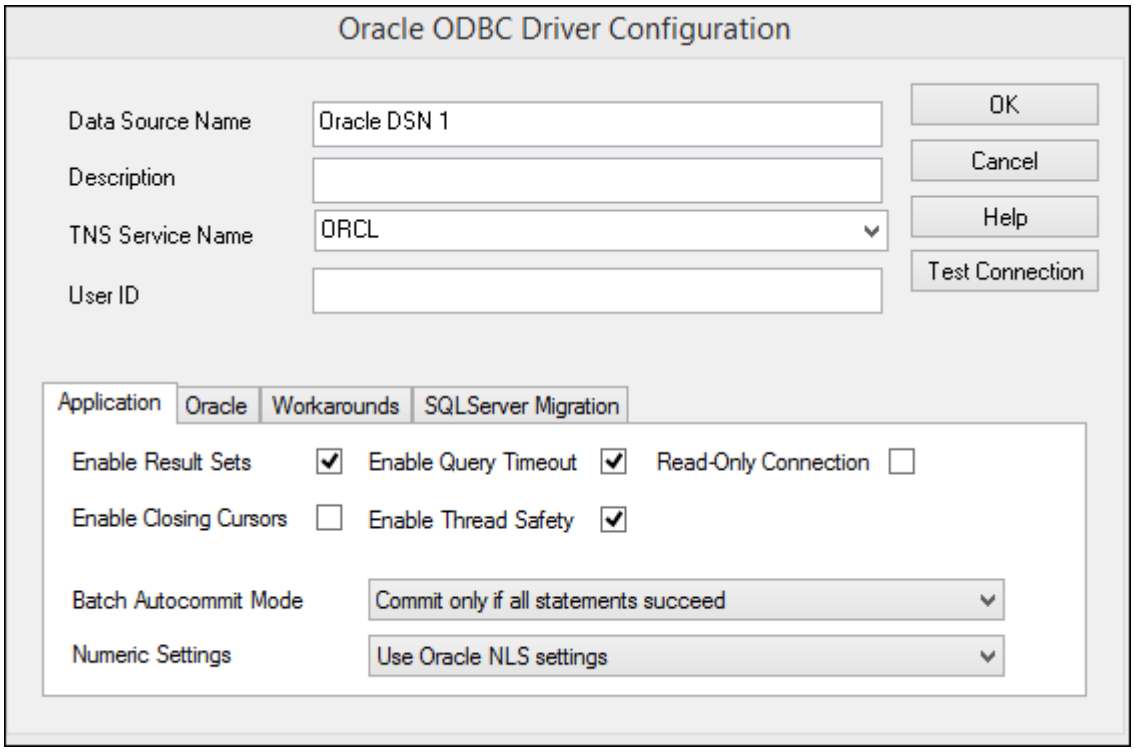
5. 使用するOracleドライバーを選択します(このサンプルでは**Oracle in OraClient11g_home1**です)。Oracle クライアントをインストール後システムで使用可能なOracleドライバーのリストを表示します。
6. [戻る] をクリックします。
7. **[新しいデータソース(DSN) をドライバーを使用して作成]** を選択して、ステップ4 で選択されたOracleドライバーを選択します。

(C) 2015-2021 Altova GmbH

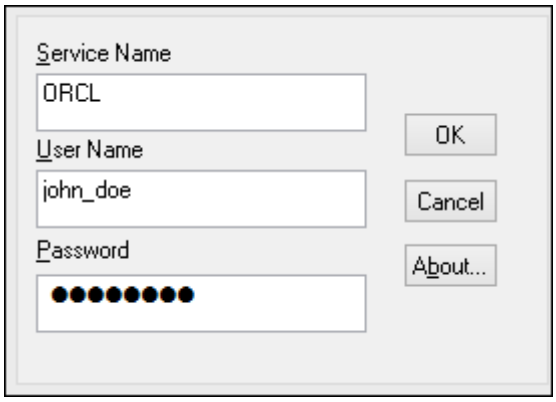


Microsoft ODBC for Oracle という名称のMicrosoftにより提供されるドライバーを使用することは避けてください。Microsoftは Oracle により提供される ODBC ドライバーを使用することを奨励しています。(以下参照: http://msdn.microsoft.com/en-us/library/ms714756%28v=vs.85%29.aspx)。

8. 「接続」をクリックします。



9. データソース名テキストボックスにデータソースを識別する名前を入力します（このサンプルの場合は**Oracle DSN 1** です）。
10. TNS サービス名ボックスでは、**tnsnames.ora** ファイルで定義されている通りの接続名を入力します。（prerequisites前提条件を参照してください）。このサンプルでは接続名 **ORCL** です。
11. 「OK」をクリックします。



12. データベースのユーザー名とパスワードを入力して、「OK」をクリックします。

6.2.1.10.15 PostgreSQL (ODBC) への接続

このトピックではODBC ドライバーを使用した、Windows マシンから PostgreSQL データベースへの接続のサンプル手順について説明されています。Windows からは、PostgreSQL ODBC ドライバーを入手することができませんので、個別にダウンロードおよびインストールする必要があります。このサンプルでは、公式 Web サイトからダウンロードされたpsqlODBC ドライバー（バージョン 11.0）を使用しています。（データベースドライバー概要も参照してください）。

(C) 2015-2021 Altova GmbH

メモ (ODBC ドライバー無しで直接 PostgreSQL データベースサーバーに接続するには、次を参照してください: PostgreSQL 接続のセットアップ。

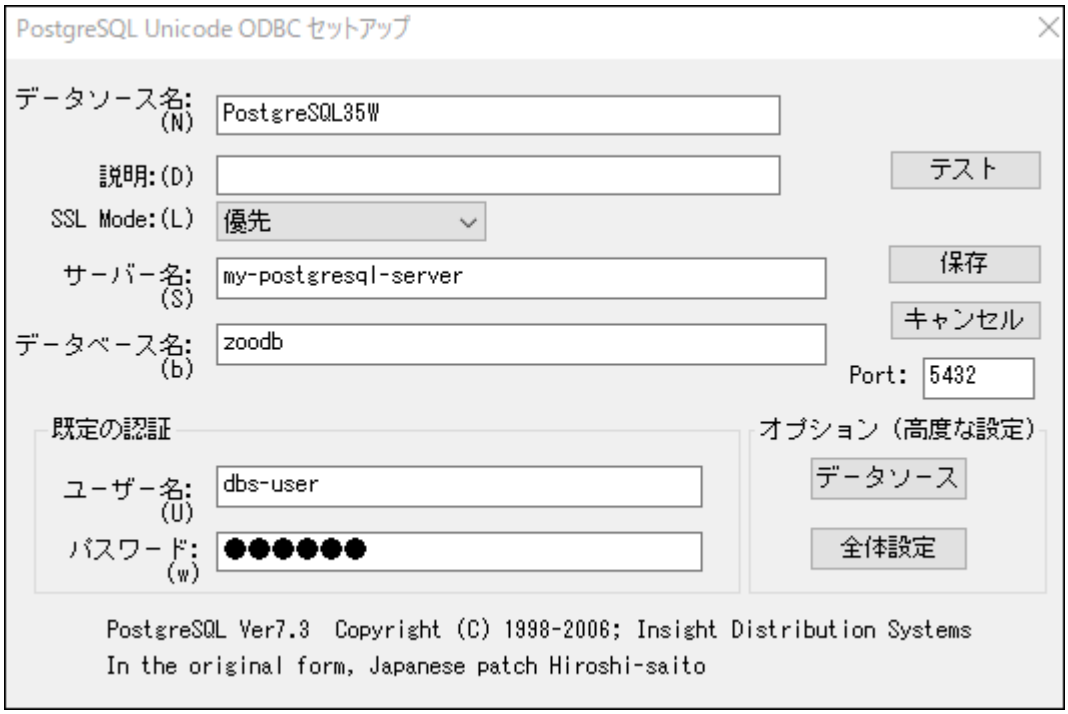
前提条件:

- *psqlODBC* ドライバーを使用中のオペレーティングシステムにインストールする必要があります。
- データベース接続の詳細: ホスト、データベース、ユーザー名およびパスワード

ODBC を使用した PostgreSQL への接続をセットアップする方法:

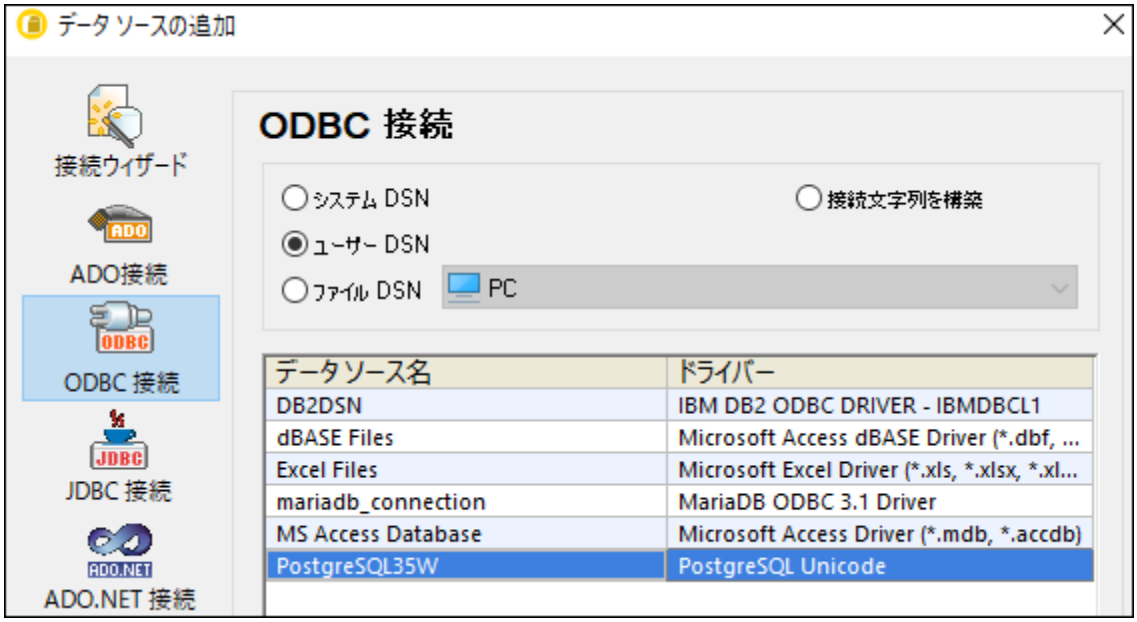
1. データベース接続ウィザードを開始します。
2. **PostgreSQL (ODBC)** をクリックします。
3. **ユーザー DSN** オプションを選択します。
4. 「**新規の DSN の作成**」 をクリックし、ドロップダウンリストからドライバーを選択します。PostgreSQL ドライバーがリスト内で使用できない場合、上記の必要条件で説明されているとおり、PostgreSQL ODBC ドライバーがオペレーティングシステム上にインストールされていることを確認してください。

5. 「**ユーザー DSN**」をクリックします。



6. データベース資格情報を入力し、「保存」をクリックします（これらはデータベースオーナーにより提供されています）。

接続はODBC 接続のリスト内で使用できるようになります。データベースに接続するには、接続をダブルクリックし、「接続」をクリックします。



6.2.1.10.16 Progress OpenEdge (JDBC) への接続

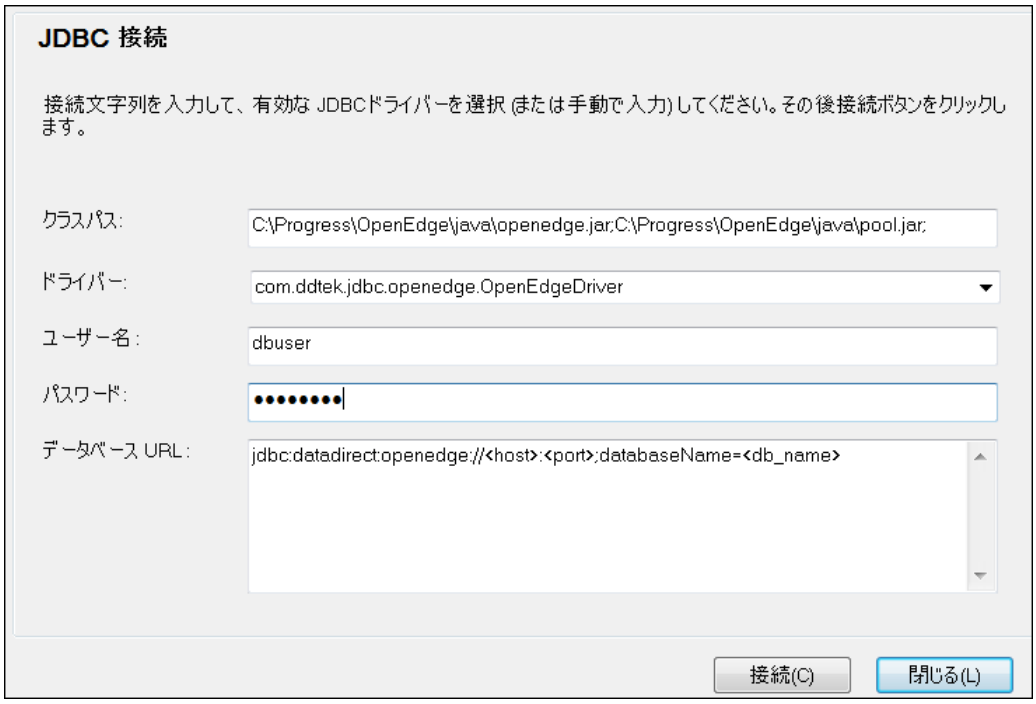
JDBC を使用して Progress OpenEdge 11.6 データベースサーバーへの接続するためのサンプルの手順が説明されています。

前提条件:

- JRE (Java Runtime Environment) または Java Development Kit (JDK) がインストールされている必要があります。これは Oracle JDK または Oracle OpenJDK などのオープンソースビルドであることができます。MapForce は次のロケーションから Java Virtual Machine (JVM) へのパスを次の順序で決定します: a) アプリケーション オプション内で設定したカスタム JVM パス Java 設定を参照してください。b) Windows レジストリ内で見つかった JVM パス。c) `JAVA_HOME` 環境変数。
- MapForce (32-ビット、64-ビット) のプラットフォームが JRE/JDK のプラットフォームに一致することを確認してください。
- オペレーティングシステムの `PATH` 環境変数は、JRE または JDK インストールの `bin` ディレクトリへのパスを含んでいる必要があります。例、`C:\Program Files (x86)\Java\jre1.8.0_51\bin`.
- Progress OpenEdge JDBC ドライバーが使用中のオペレーティングシステムで使用可能である必要があります。このサンプルでは、JDBC 接続は、**C:\Progress\OpenEdge\java** に OpenEdge SDK のインストールの一部として存在する **openedge.jar** と **pool.jar** ドライバーコンポーネントファイルにより与えられています。.
- データベース接続の詳細があります: ホスト、データベース パス、またはエイリアス、ユーザー名、および パスワード。

JDBC を使用して OpenEdge へ接続する:

1. データベース接続ウィザードを開始します.
2. 「**JDBC 接続**」をクリックします。
3. 「**クラス パス**」の横にデータベースへの接続を与える .jar ファイルへのパスを入力します。必要であれば、セミコロンで区切られた .jar ファイル パスのリストを入力することもできます。このサンプルでは、必要とされる .jar ファイルは次のパスを使用して見つけることができます`C:\Progress\OpenEdge\java\openedge.jar;C:\Progress\OpenEdge\java\pool.jar;`。.jar ファイル パスをオペレーティングシステムの CLASSPATH 環境変数に追加した場合、「**クラス パス**」テキスト ボックスを空白のままにすることができます。(次も参照: CLASSPATH の構成)。
4. 「**ドライバー**」ボックスから、**com.ddtek.jdbc.openedge.OpenEdgeDriver** を選択します。このエントリは、有効な .jar ファイル パスが、「**クラス パス**」テキスト ボックス内、または、オペレーティングシステムの CLASSPATH 環境変数内で検出された場合のみ使用することができます (前のステップを参照してください)。



5. データベースへのユーザー名とパスワードを対応するテキストボックスに入力します。
6. ハイライトされた値を、使用するデータベースサーバーに適用する値と置き換えて、データベースURL テキストボックス内にデータベースサーバーへの接続文字列を入力します。

```
jdbc:datadirect:openedge://host:port;databaseName=db_name
```

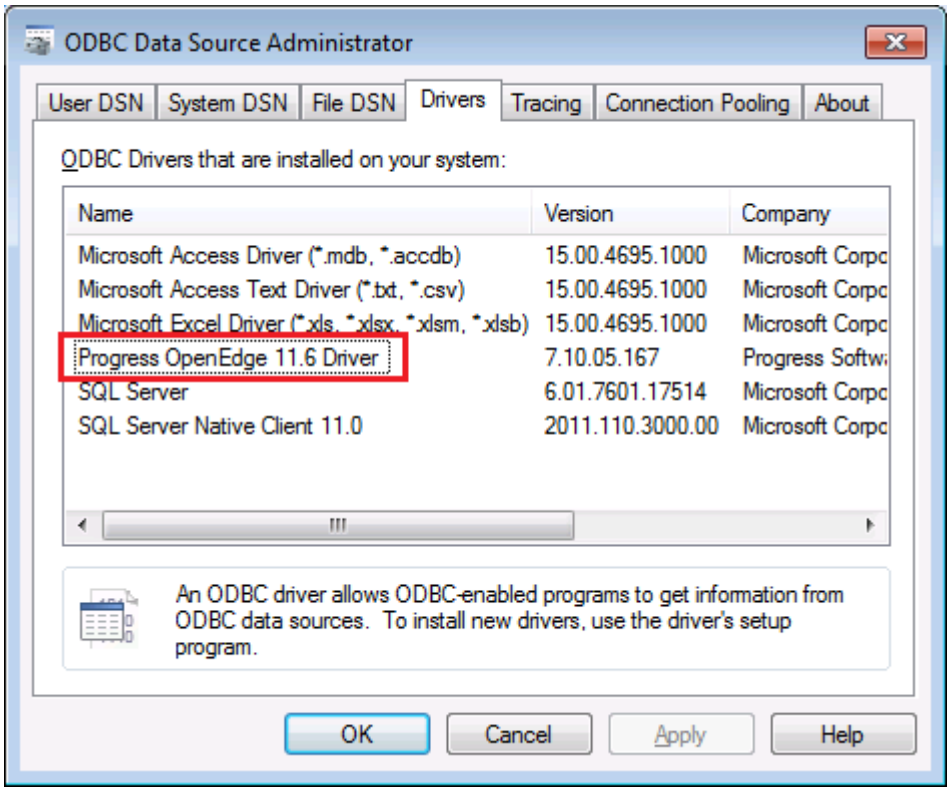
7. 「接続」をクリックします。

6.2.1.10.17 Progress OpenEdge (ODBC) への接続

このサンプルではProgress OpenEdge 11.6 ODBC ドライバーを使用して、Progress OpenEdge データベースサーバーへの接続の簡単な手順について説明されています。

前提条件:

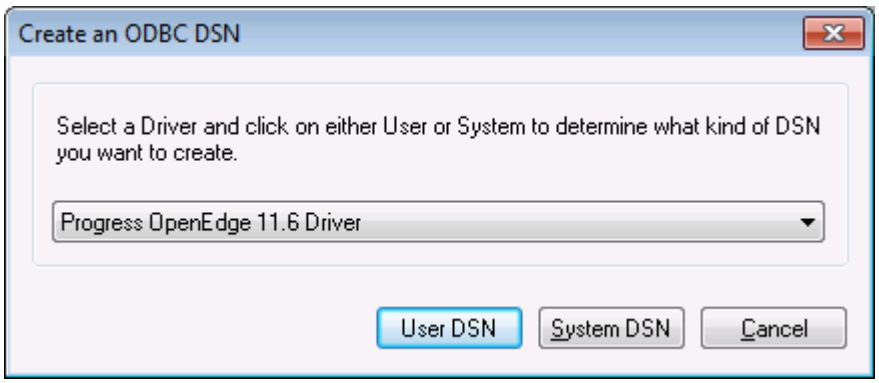
- *Progress OpenEdge* ドライバーのための *ODBC* コネクタが使用中のオペレーティングシステムにインストールされている必要があります。Progress OpenEdge ODBC ドライバーは、ベンダーのW eb サイトからダウンロードすることができます(次も参照: データベースドライバーの概要)。MapForce の32 ビットバージョンを作動する場合は、32 ビット ドライバーをダウンロードし、64 ビットバージョンを作動する場合は、64 ビット ドライバーをダウンロードしてください。インストールの後、ODBC ドライバーが使用中のマシンで使用することができるかを確認してください(次も参照: 使用することのできる ODBC ドライバーを確認する)。



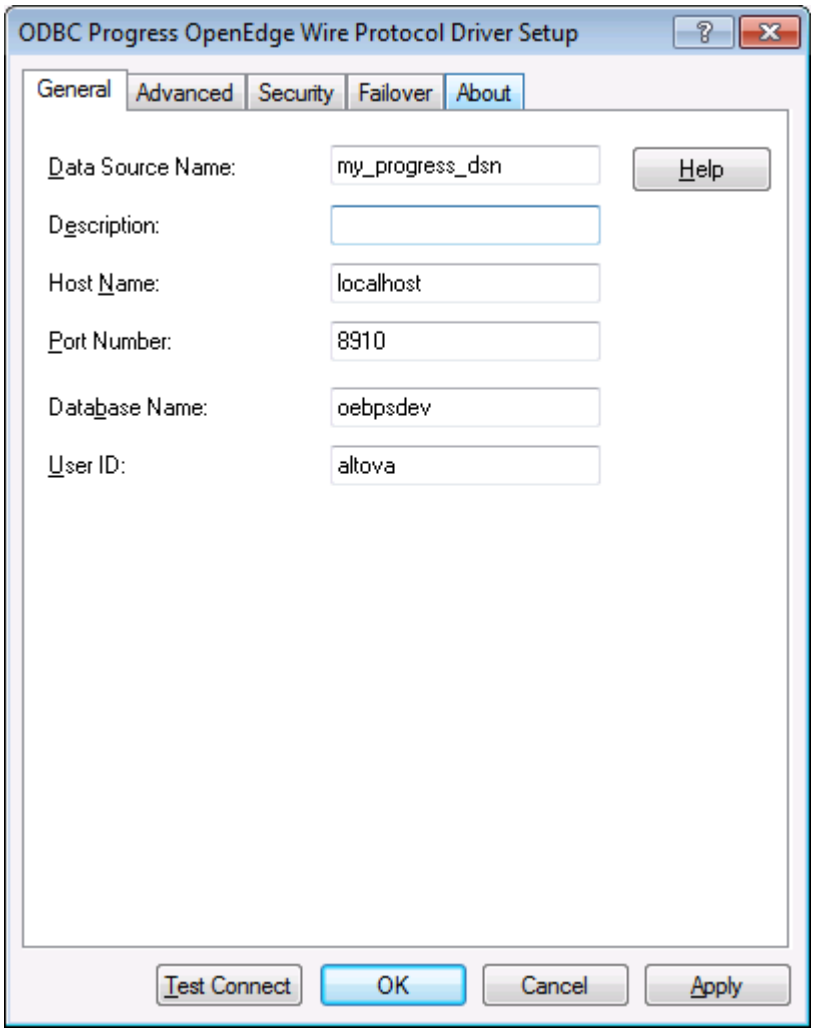
- データベース接続の詳細があります: ホスト名、ポート番号、データベース名、ユーザーID、および パスワード。

ODBC を使用して Progress OpenEdge に接続する:

1. データベース接続ウィザードを開始します.
2. 「ODBC 接続」をクリックします。
3. 「ユーザーDSN」をクリックします。(または、次の手順が類似している「システムDSN」、または、「ファイルDSN」をクリックします)。
4. 「追加」 をクリックします。
5. 「Progress OpenEdge ドライバー」をリストから選択し、「ユーザーDSN」(または、適用できる場合は、「システムDSN」)をクリックします。

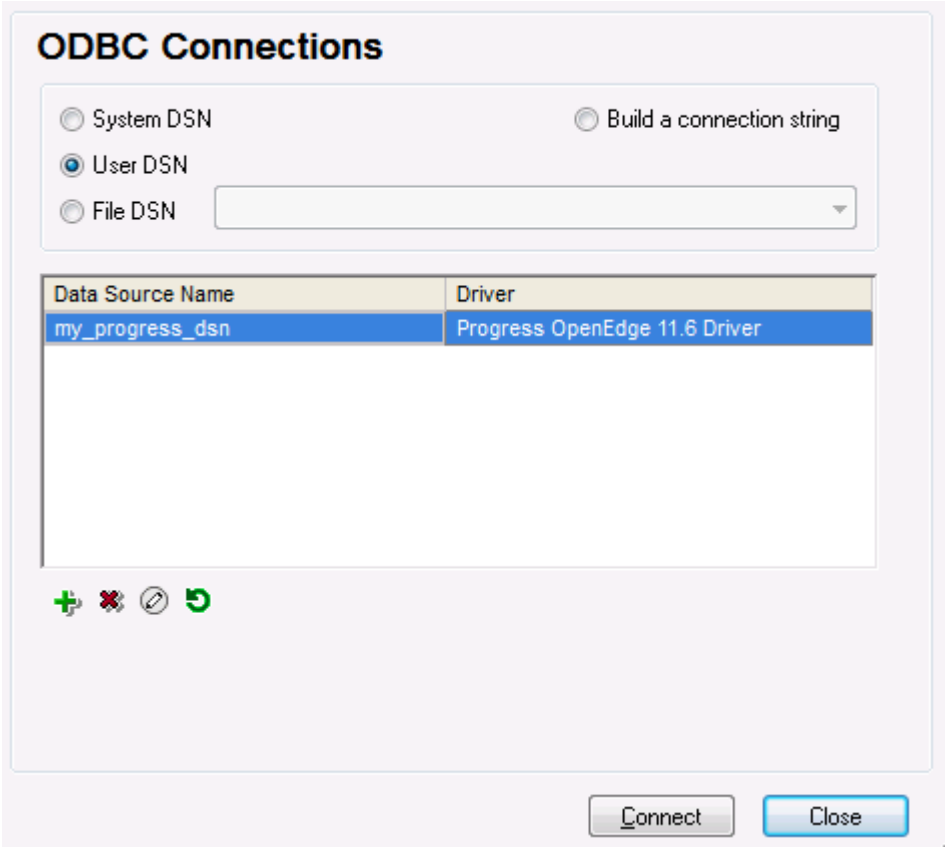


6. データベース接続の資格情報（データベース、サーバー、ポート、ユーザー名、パスワード）を入力し、「OK」をクリックします。入力されたデータを保存する前に、接続を検証するには、「**接続をテストする**」をクリックします。



7. 「OK」をクリックします。新規データソースがODBC データソースのリストに表示されます。

(C) 2015-2021 Altova GmbH



8. 「接続」をクリックします。

6.2.1.10.18 Sybase (JDBC) への接続

このサンプルでは、JDBC を使用して Sybase データベースサーバーに接続する手順について説明されています。

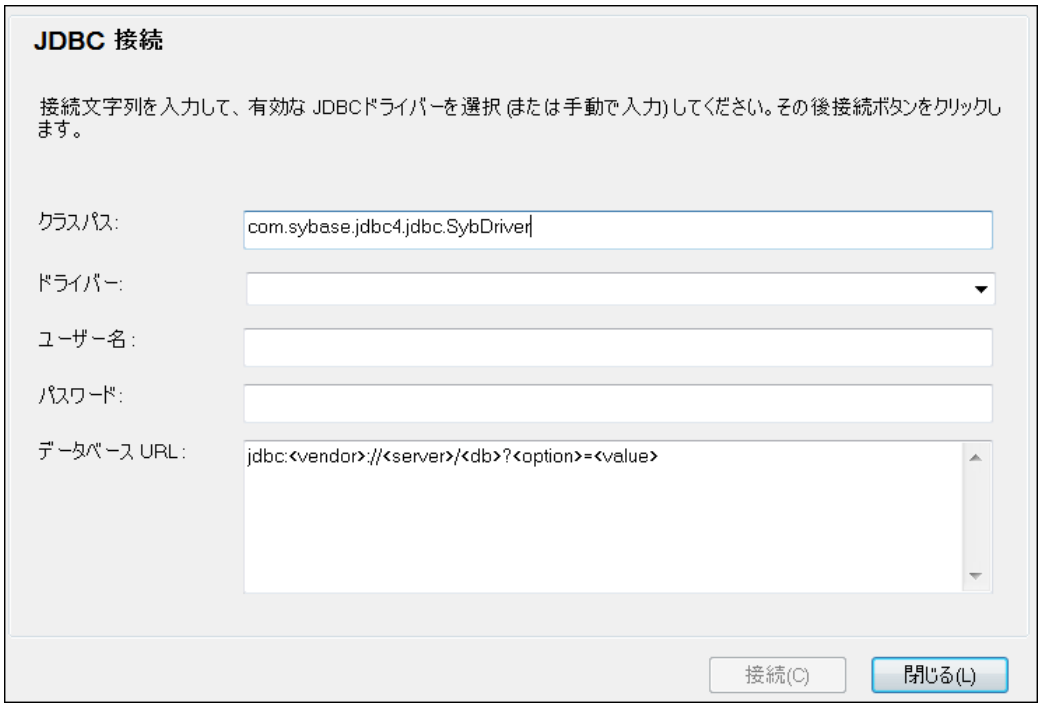
前提条件:

- JRE (Java Runtime Environment) または Java Development Kit (JDK) がインストールされている必要があります。これは Oracle JDK または Oracle OpenJDK などのオープンソースビルドであることができます。MapForce は次のロケーションから Java Virtual Machine (JVM) へのパスを次の順序で決定します: a) アプリケーションオプション内で設定したカスタム JVM パス Java 設定を参照してください。b) Windows レジストリ内で見つかった JVM パス。c) `JAVA_HOME` 環境変数。
- MapForce (32-ビット、64-ビット) のプラットフォームが JRE/JDK のプラットフォームに一致することを確認してください。
- Sybase *jConnect* コンポーネントがオペレーティングシステムにインストールされている必要があります (このサンプルでは *jConnect 7.0* が使用されており、*Sybase Adaptive Server Enterprise PC Client* インストールの一部としてインストールされています)。データベースクライアントのインストール命令に関しては、Sybase ドキュメントを参照してください。
- 次のデータベース接続詳細が存在します: ホスト、データベース、ポート、ユーザー名、および、パスワード。

JDBC を使用して Sybase へ接続する:

1. データベース接続ウィザードを開始します。

2. 「JDBC 接続」をクリックします。
3. 「クラス パス」の横にデータベースへの接続を与える .jar ファイルへのパスを入力します。必要であれば、セミコロンで区切られた .jar ファイル パスのリストを入力することもできます。このサンプルでは、必要とされる .jar ファイルは次のパスを使用して見つけることができます: **C:\sybase\jConnect-7_0\classes\jconn4.jar**。.jar ファイル パスをオペレーティングシステムの CLASSPATH 環境変数に追加した場合、「クラス パス」テキスト ボックスを空白のままにすることができます。(次も参照: CLASSPATH の構成)。
4. 「ドライバー」ボックスから、 を選択します。「ドライバー」ボックスから、**com.sybase.jdbc4.jdbc.SybDriver** を選択します。このエントリは、有効な .jar ファイル パスが、「クラス パス」テキスト ボックス内、または、オペレーティングシステムの CLASSPATH 環境変数内で検出された場合のみ使用することができます (前のステップを参照してください)。



5. データベースのユーザー名とパスワードを対応するテキスト ボックスに入力します。
6. ハイライトされた値をデータベース サーバーに適用することのできる値と置き換えて、データベース URL テキスト ボックス内のデータベース サーバーの接続文字列を入力します。

```
jdbc:sybase:Tds:hostName:port/databaseName
```

7. 「接続」をクリックします。

6.2.1.10.19 Teradata (JDBC) への接続

このサンプルでは、JDBC を使用した Teradata データベースへの接続方法について説明されています。

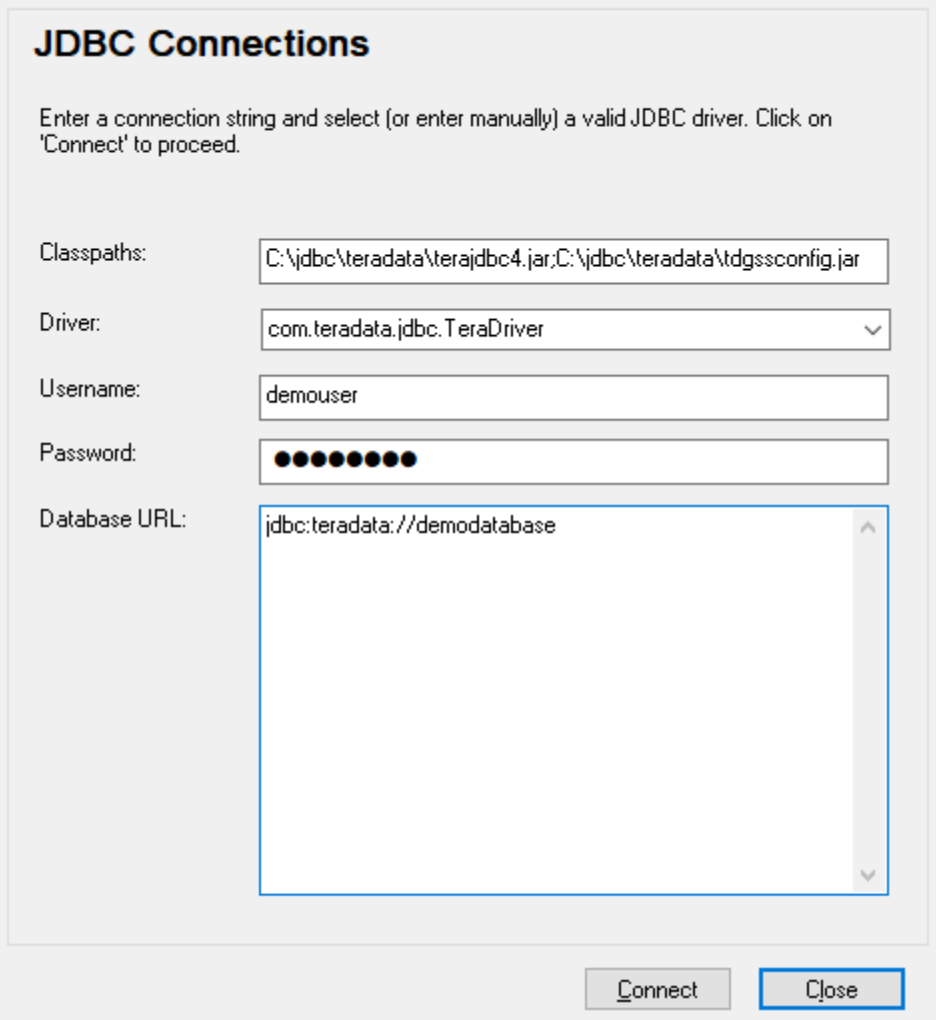
必要条件:

- JRE (Java Runtime Environment) または Java Development Kit (JDK) がインストールされている必要があります。これは Oracle JDK または Oracle OpenJDK などのオープンソース ビルドであることができます。MapForce は次のロケーションから Java Virtual Machine (JVM) へのパスを次の順序で決定します: a) アプリケーション オプション内で設定したカスタム JVM パス (Java 設定を参照してください)。b) Windows レジストリ内で見つかった JVM パス。c) JAVA_HOME 環境変数。

- MapForce (32-ビット、64-ビット) のプラットフォームが JRE/JDK のプラットフォームに一致することを確認してください。
- JDBC ドライバー (データベースへの接続を提供する 1 つまたは複数の .jar ファイル) がオペレーティングシステム上で使用できる必要があります。このサンプルでは、Teradata JDBC ドライバー 16.20.00.02 が使用されます。詳細に関しては、次を参照してください: http://downloads.teradata.com/download/connectivity/jdbc-driver。
- 次のデータベース接続詳細が存在します: ホスト、データベース、ポート、ユーザー名、および、パスワード。

JDBC を使用した Teradata への接続の方法:

1. データベース接続ウィザードを開始します。
2. 「JDBC 接続」をクリックします。
3. 「クラスパス」の横に、データベースへの接続を与える .jar ファイルへのパスを入力します。必要な場合、セミコロンで区切られた .jar ファイルパスのリストを入力することができます。このサンプルでは、.jar ファイルは、次のパスの場所に存在します: **C:\jdbc\teradata**。オペレーティングシステムの CLASSPATH 環境変数に .jar ファイルパスが追加されている場合、「クラスパス」テキストボックス空のままにすることができます (次も参照してください: CLASSPATH の構成)。
4. 「ドライバー」ボックス内で、**com.teradata.jdbc.TeraDriver** を選択します。有効な .jar ファイルパスが「クラスパス」テキストボックスで、または、オペレーティングシステムの CLASSPATH 環境変数内に存在すると、使用することができます (前のステップを参照してください)。



5. ユーザー名とパスワードを対応するテキストボックスのデータベース内に入力します。

6. ハイライトされた値をデータベースサーバーと適用することのできる値を置換え、データベースURL テキスト内にデータベースサーバーに接続文字列を入力します。

```
jdbc:teradata://databaseServerName
```

7. 「接続」をクリックします。

6.2.1.10.20 Teradata (ODBC) への接続

このサンプルでは、ODBC を使用したSQL Teradata データベースへの接続方法について説明されています。

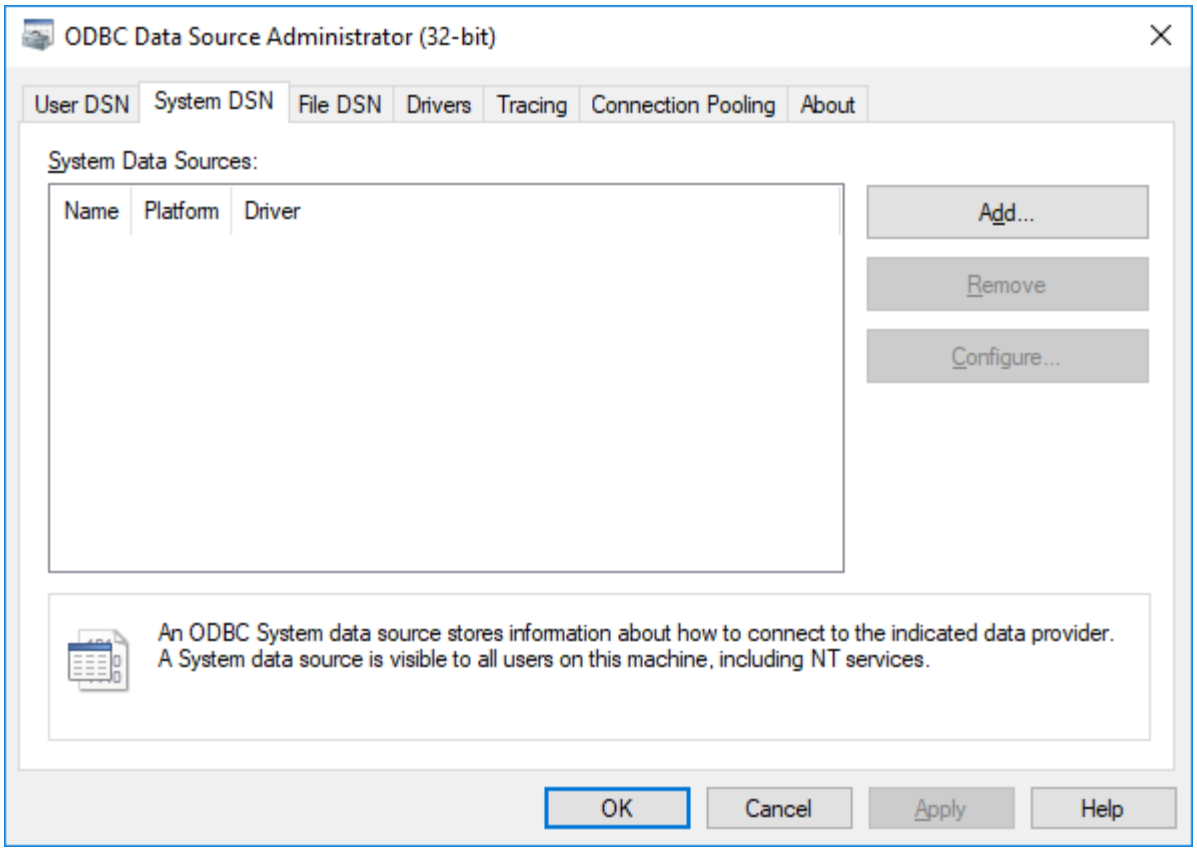
前提条件:

- Teradata ODBC ドライバーがインストールされる必要があります(参照: http://downloads.teradata.com/download/connectivity/odbc-driver/windows。このサンプルは、W indows バージョン16.20.00 のためにODBC ドライバーを使用しています。
- 次のデータベース接続詳細が存在します: ホスト、ユーザー名、および、パスワード。

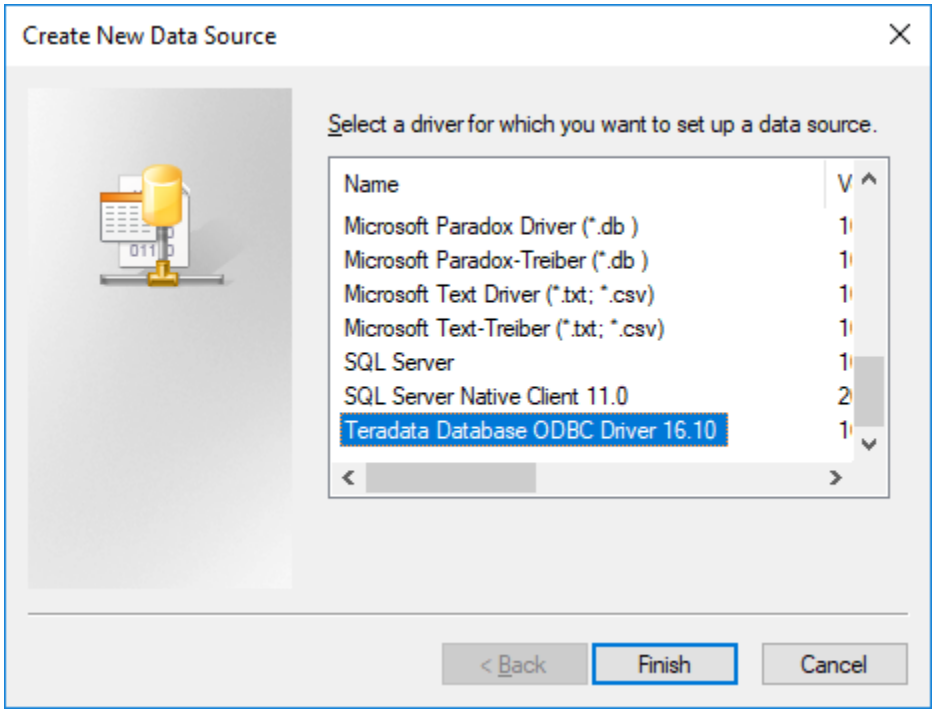
ODBC を使用した Teradata への接続の方法:

1. **Windows** キーを押して、「ODBC」を入力し、「ODBC データソースのセットアップ(32-ビット)」を提案されるリストから選択します。64-ビット ODBC ドライバーが存在する場合、「ODBC データソースのセットアップ(64-ビット)」を選択し、次のステップ内で64-ビット MapForce を使用します。

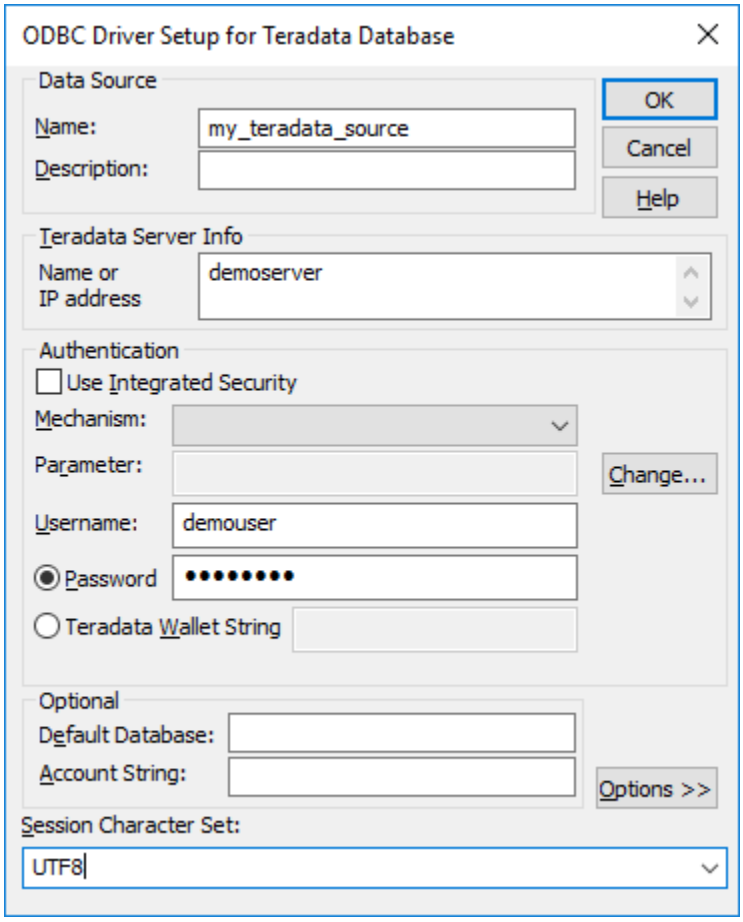
(C) 2015-2021 Altova GmbH



2. 「System DSN」タブをクリックして、「追加」をクリックします。

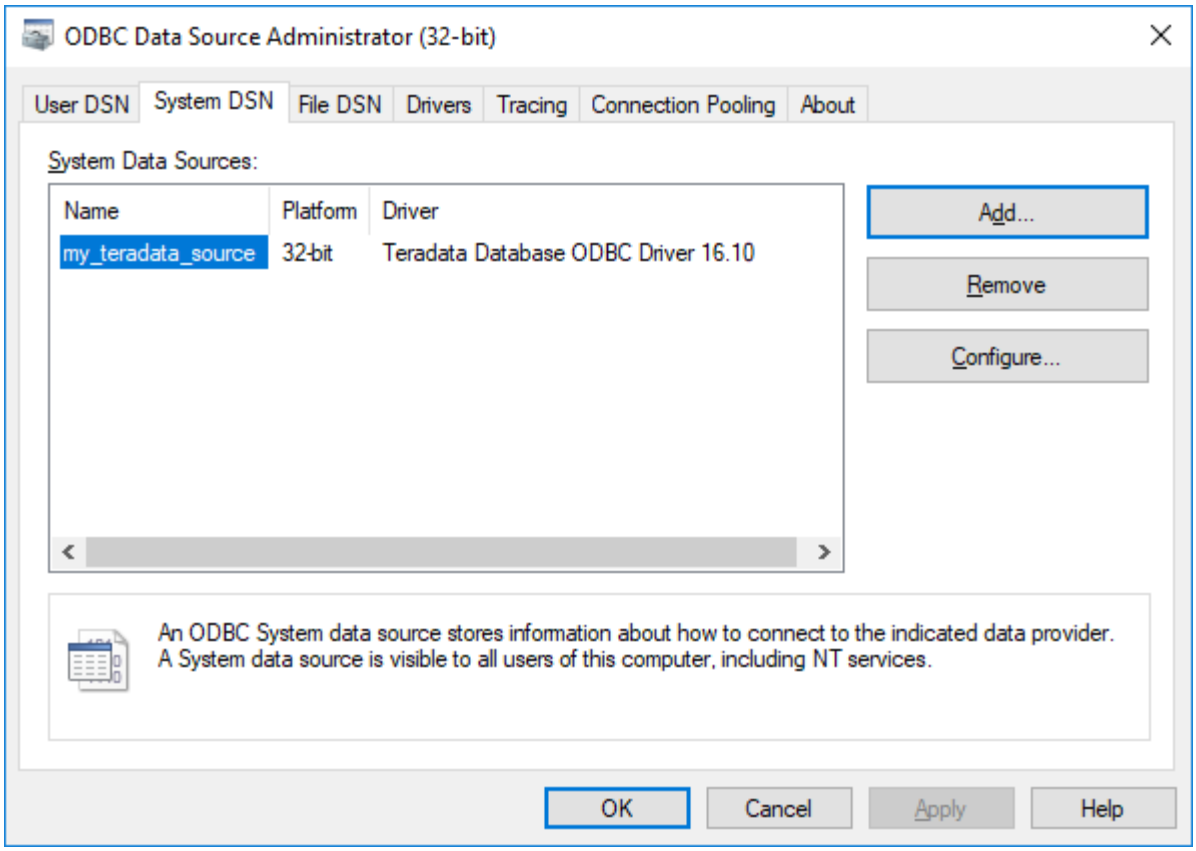


3. 「Teradata データベース ODBC ドライバー」を選択して、「終了」をクリックします。

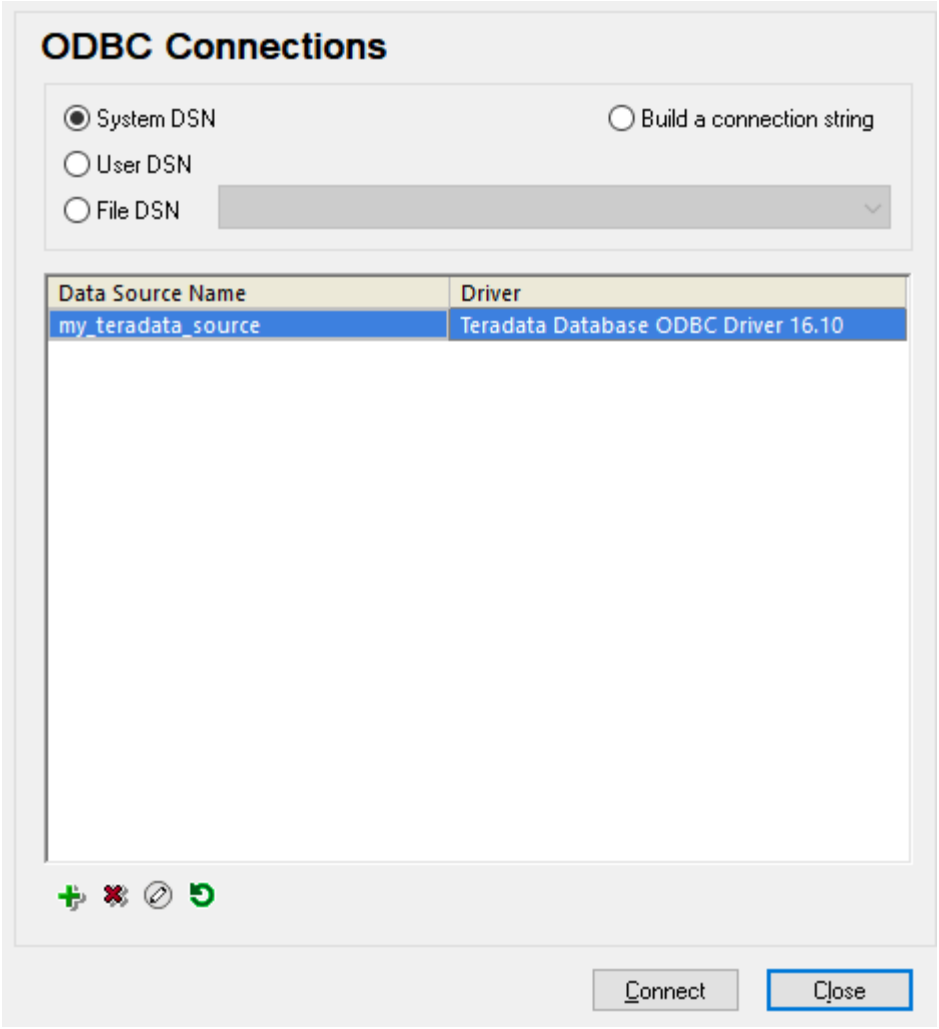


4. 名前を入力し、任意で、このODBC データソースを将来識別できる詳細を与えます。また、データベース接続資格情報（データベース サーバー、ユーザー、パスワード）を入力し、オプションでデータベースを選択します。
5. 「OK」をクリックします。データソースがリスト内に表示されます。

(C) 2015-2021 Altova GmbH



6. MapForce を作動し、データベース接続ウィザードを開始します。
7. 「ODBC 接続」をクリックします。



8. 「**System DSN**」をクリックし、前に作成されているデータソースを選択し、「**接続**」をクリックします。

メモ 次のエラーが発生した場合: 「ドライバーが無効として返されました（または返しませんでした） SQL_ DRIVER_ ODBC_ VER: 03.80」「ドライバーが無効と返されました」。ODBC クライアントへのパスが使用中のシステムのPATH 環境変数内に存在することを確認してください（例えば、このロケーションにインストールされている場合、**C:\Program Files\Teradata\Client\16.10\bin** ）。このパスが不足している場合、手動で追加します。

6.2.2 データベースマッピングの概要

このセクションはMapForce 内出データベースと作業する方法に関する概要です。 接続の確立に成功した後、どのようにしてデータベースと作業するかについて説明されています（次を参照: データベースへの接続 ）。これには、マッピングに表示されるデータベースオブジェクト、データベースリレーションシップの処理、マッピングプロセスに適用することのできるデータベースの構成 などが含まれます。データベースコンポーネントから、またはデータベースコンポーネントにデータをマッピングする際の特定の目的を達成する方法もこのセクションの例に含まれています。

(C) 2015-2021 Altova GmbH

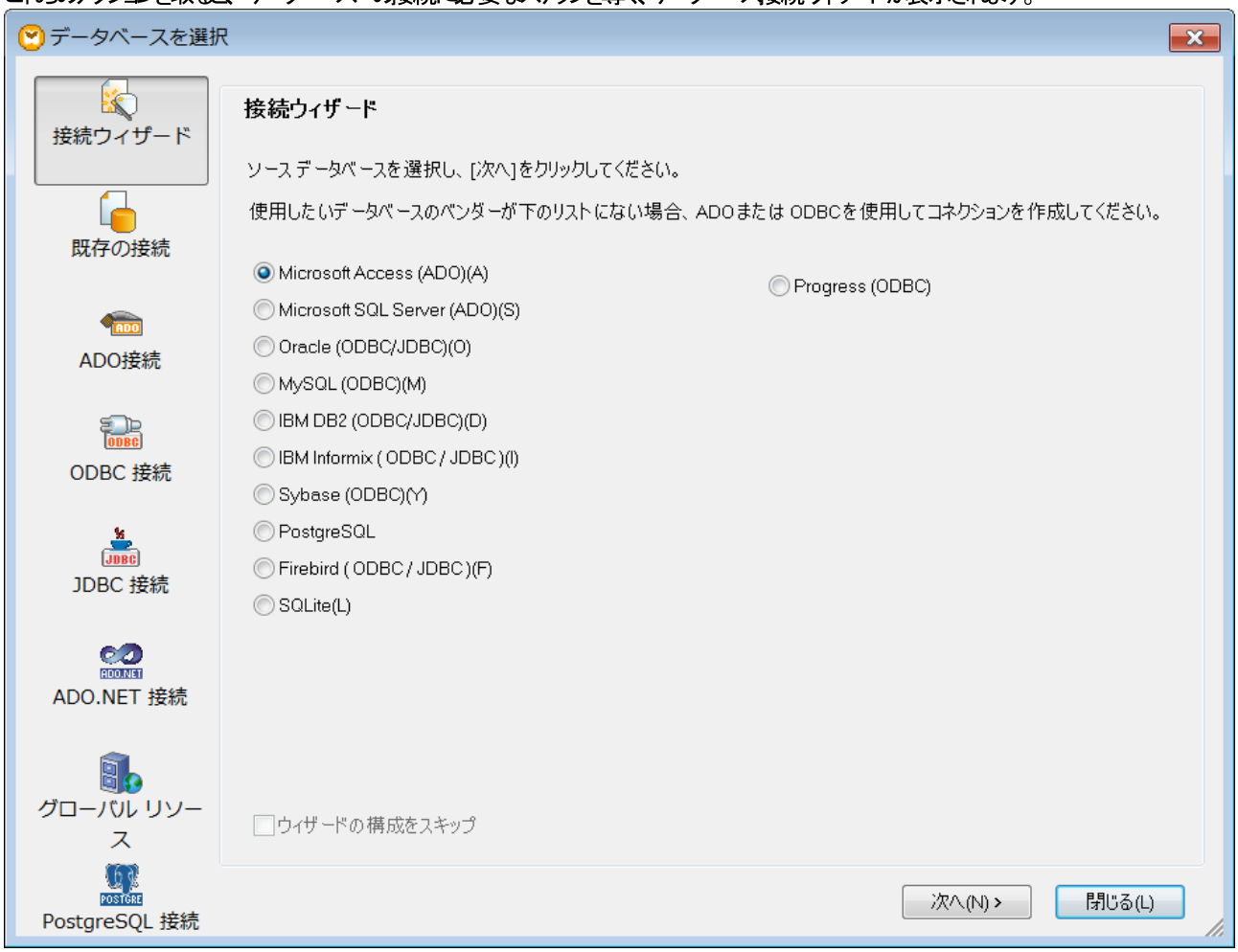
6.2.2.1 マッピングにデータベースを追加する

マッピングにデータベースを追加する前に、データベースマッピングがサポートされている変換言語が選択されていることを確認してください。これは、BUILT-IN 変換言語、または次の言語であることができます: C++、C#、Java (次も参照してください: 変換言語の選択)。マッピングを FlowForce Server にデプロイ、または MapForce Server で実行、またはバルク転送とストアドプロシージャなどの機能を使用する場合、BUILT-IN が変換言語として選択される必要があります。

希望する変換言語が選択されると、データベースをマッピングに次の方法で追加することができます:

- 「挿入」メニューから「データベース」をクリックします。
- 「データベースの挿入」( ) ツールバーボタンをクリックします。

これらのアクションを取ると、データベースへの接続に必要なステップを導く、データベース接続ウィザードが表示されます。



データベース接続ウィザード

メモ 高度なシナリオの一部では、データベースに変数として追加することができます (次を参照: 変数の使用)。データベース構成を変数として追加する場合は、上記と同じデータベース接続ウィザードが表示されます。

このウィザードを使用して MapForce によりサポートされているデータベースへの接続の方法に関しては、以下を参照してください: データベースへ接続 。

- Sybase への接続(JDBC)

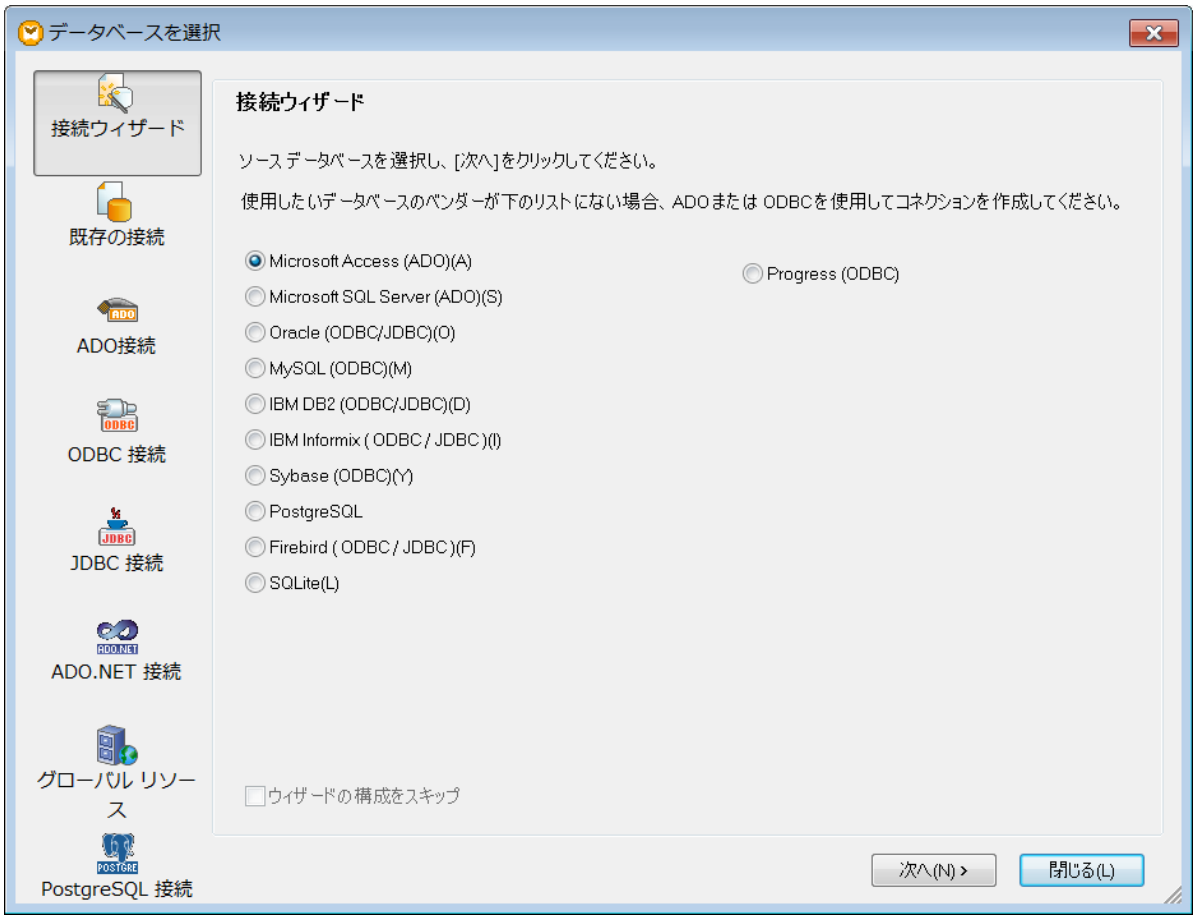
データベースへの接続に成功すると、マッピングに表示されるデータベースオブジェクトを選択するようにプロンプトされます。詳細に関しては、以下を参照してください: データベースオブジェクトの追加、編集、および削除。

6.2.2.2 例: "altova.mdb" データベースをマッピングに追加する

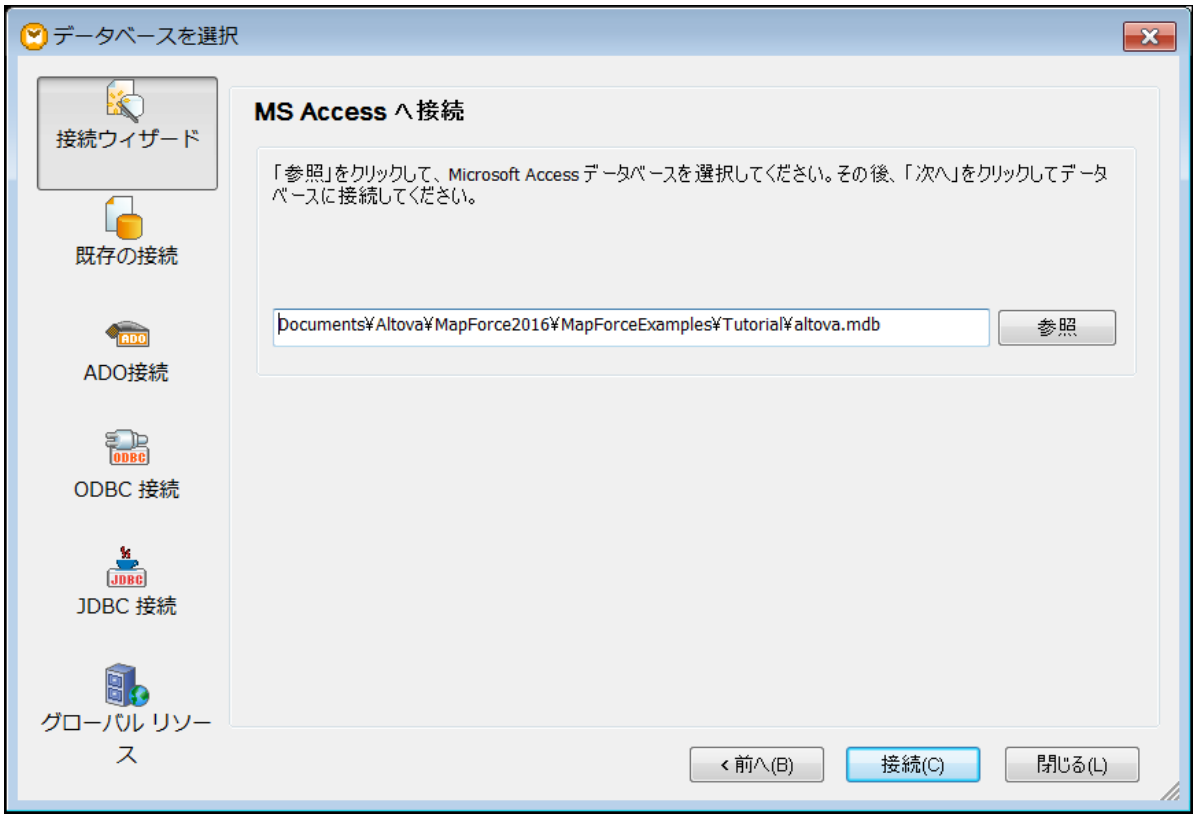
この例は、Microsoft Access データベースをマッピングに追加する方法を説明します。サンプルのデータベースは、**altova.mdb** と呼ばれ、<**マイドキュメント**>¥Altova¥MapForce2021¥MapForceExamples¥フォルダー内にあります。**altova.mdb** データベースは、このドキュメント内で説明されているデータベースに関連したアクションと概念をサポートします。

altova.mdb データベースをマッピングに追加する:

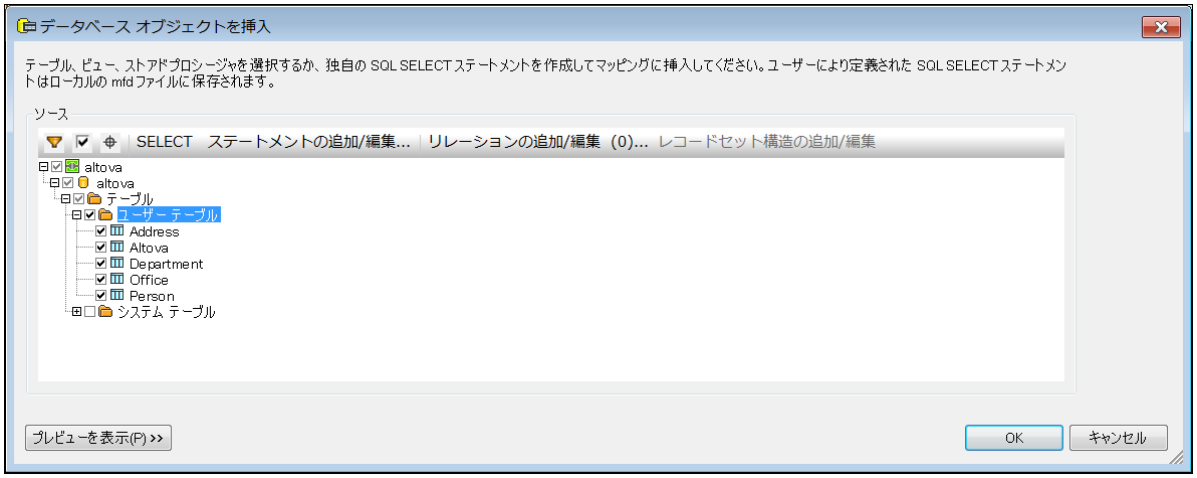
1. 「**挿入**」メニューから「**データベース**」をクリックします。または、「**データベースの挿入**」( ) ツールボタンをクリックします。



2. **Microsoft Access (ADO)**をクリックして、「次へ」をクリックします。



3. <マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥フォルダー内の**altova.mdb** ファイルを参照し、「**接続**」をクリックします。
4. データベースオブジェクトを選択するようにプロンプトされると、「**ユーザーテーブル**」を選択します。

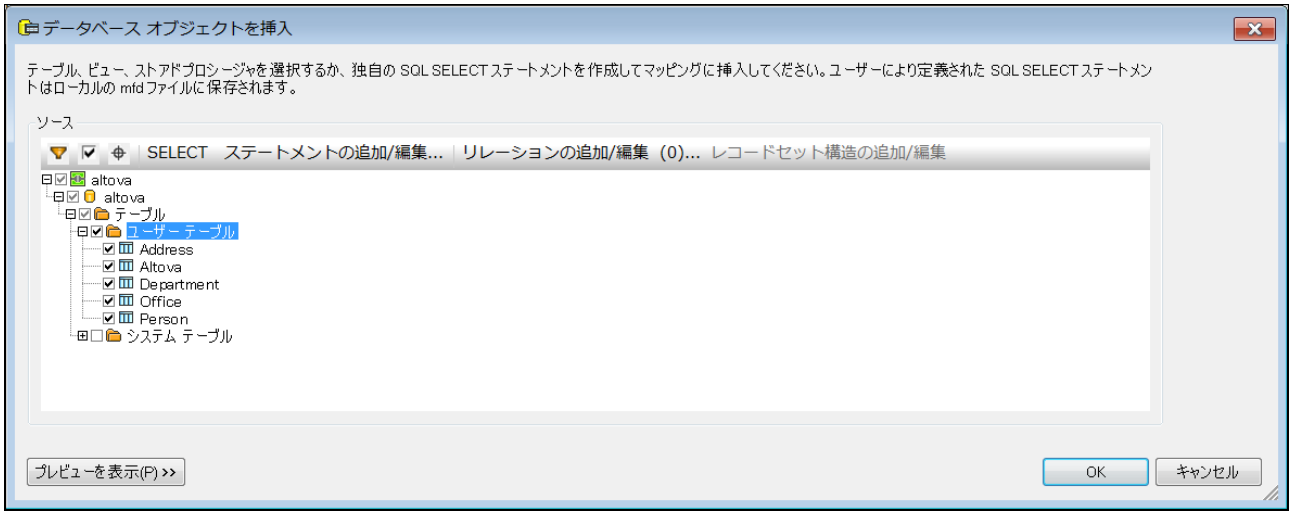


6.2.2.3 データベースオブジェクトを追加、編集、削除する

データベースの一部は、(スキーマ、テーブル、ビューなどの) 多数のオブジェクトをもつことができます。このトピックは、マッピングの目的に必要なデータベースオブジェクトのみをマッピングする方法について説明します。下では、サンプル Access データベースが使用されています。データベース型のための手順と類似しています。

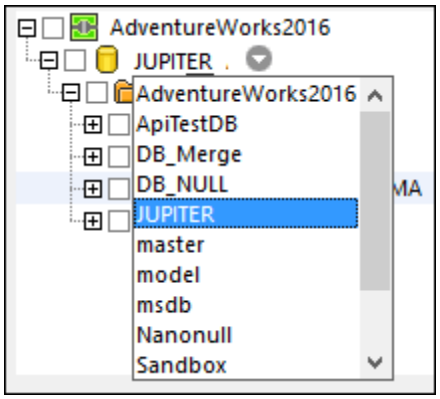
1. 「挿入」メニューから、「データベース」をクリックします。
2. 「接続ウィザード」をクリックして、**Microsoft Access (ADO)** をクリックします。
3. 「次へ」をクリックし、<マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥フォルダー内の **altova.mdb** を参照します。

マッピングに含まれるデータベースオブジェクトを選択することのできるアクション ダイアログボックスが表示されます。



構造内のトップノード はデータベース接続を表します。その後の構造はデータベースの種類により異なります。例えば、Oracle とIBM DB2 データベースには接続ノードの下にスキーマノード が存在しますが他のデータベースの種類にはカタログ(データベース) ノードが存在します。太線はデフォルトのカタログ(データベース) 、またはスキーマを示しています。

データベースユーザーアカウントがサーバー上に複数のデータベースまたはスキーマを持つ場合、 アイコンをクリックして必要なものに切り替えることができます。下のサンプルは SQL Server の例を表示しています。構造は他のデータベースの型では異なる場合があります。

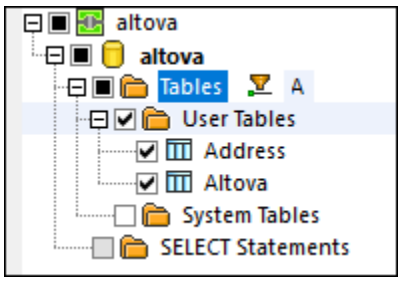


データベースオブジェクトを含むには、(例えば、テーブル) マッピングの横のチェックボックスをクリックしてください。この例の目的のために、「ユーザーテーブル」の横のチェックボックスをクリックします。

「オブジェクト ロケーター」ボタン( ) を使用して特有の データベースアイテムを検索することができます。ダイアログボックスの下の部分に表示される、コンボボックス内から特定のオブジェクトを選択(または、名前を入力)します。

(C) 2015-2021 Altova GmbH

「フィルター」ボタン（ ）によりオブジェクトを名前別にフィルターすることができます。1度フィルターボタンをクリックすると、フィルターアイコンがフィルタリングをサポートするオブジェクトの横に表示されます（この例では、"Tables"）。 フィルターアイコンをクリックして、オブジェクト名が検索テキストと終わる、同等、または、含むかを選択してください。フィルターの横に検索テキストを入力することができます（この場合、"A"）:



「チェックされたオブジェクトのみ表示」ボタン（ ）は、チェックボックスがアクティブな箇所のアイテムを表示します。

「SELECT ステートメントの追加/編集 」ボタンを使用して、現在のデータベースのためにカスタムSELECT ステートメントを追加、または編集することができます。このようなステートメントにより返されたデータは、この後マッピングソースとして使用することができます。 詳細に関しては、次を参照してください: 仮想テーブルのSQL SELECT ステートメント。

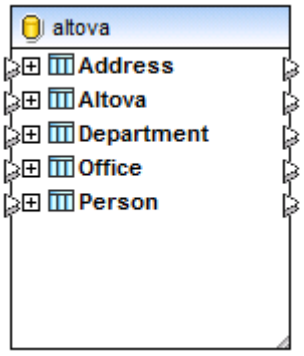
「リレーションの追加/編集 」ボタンを使用して、既存のものに加え、データベース内のフィールド間のローカルな主キーと外部キーリレーションシップを定義することができます。詳細に関しては、次を参照してください: ローカルリレーションシップの定義。

「レコードセット構造の追加/編集 」ボタンは、ストアドプロシージャをサポートするデータベースに適用されます。ストアド プロシージャ がデータベース ツリーから現在選択されている場合のみ有効化することができます。詳細に関しては、次を参照してください: ストアド プロシージャ。

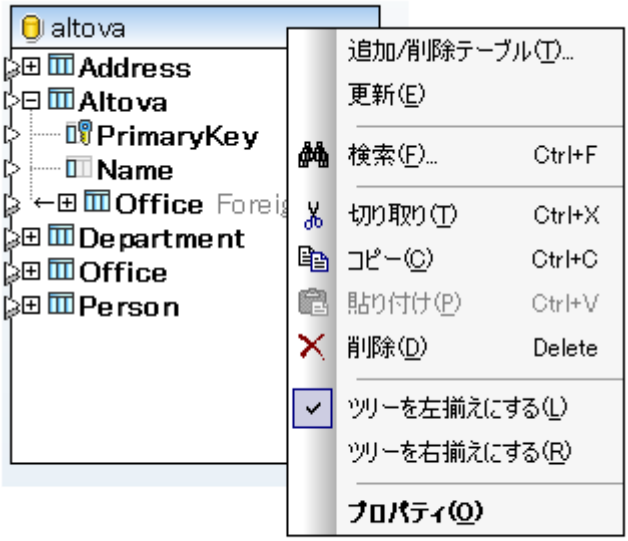
「プレビューの表示」ボタンにより、現在選択されているテーブルから、またはビューから素早くデータをプレビューすることができます。データベースブラウザーを使用して、マッピング処理とは別にデータベースを参照し、クエリすることもできます。詳細に関しては、次を参照してください: データベースの参照とクエリ。

「テーブル名からスキームを削除する」チェックボックスがでオタベースのために有効化されます。この機能は後に異なるデータベースに切り替える際に役に立ちます。詳細に関してはデータベースとスキーマの切り替えを参照してください。

マッピングへデータベースオブジェクトを追加する準備が整うと、「OK」をクリックします。選択されたテーブル、ビュー、のみがデータベースコンポーネント上に表示され、マッピング接続を MapForce の標準の方法で描くことができます。



データベースオブジェクトを変更するには、コンポーネントを右クリックして、「データベースオブジェクトの追加/削除/編集」を選択します。



6.2.2.4 データベースとスキーマの切り替え

新規のデータベース接続を使用するようにマッピング上のデータベースコンポーネントを変更するには、データベースコンポーネント設定 を開き、「変更」をクリックし、新規のデータベースへ接続するためにウィザードのステップに従います。データベースオブジェクトのみを変更する場合、データベースコンポーネントのタイトルバーを右クリックして、「データベースオブジェクトの追加/編集/削除」をコンテキストメニューから選択します。データベースオブジェクトの追加、編集、および削除も参照してください。
デフォルトではデータベースの変更はすべての既存のマッピング接続をデータベース接続に、またはは、データベース接続から再マップしないことに注意してください。このトピック内で説明される準備ステップに従うとデータベースの型の多数はデータベースの変更後にマッピング接続を保持することができます。

データベースオブジェクトの追加、編集、および削除 内で説明されている通りデータベースオブジェクトを選択すると、任意でデータベースオブジェクト名がデフォルトのスキーマに対して相対的に考慮されるかを選択することができます。データベースのコンテキストでは、「スキーマ」は通常パーミッションの割り当てなどセキュリティの目的のためのデータベースオブジェクトのロジカルなグループ分類です。スキーマはデータベースの型により若干異なる振る舞いを取ります。

MapForce 内では、マッピングを異なるデータベースに後に変更する予定がある場合データベースオブジェクト名をスキーマに対して相対的に扱うことは需要です（例えば、生産データベースに変更する場合など）。これはサーバー上でデータベーススキーマに名前が付けられており、マッピングを必要に応じて更新する場合に役に立ちます。スキーマがマッピングデザイン時に使用されたものと同じ構造を持つ場合、マッピング接続を手動で変更することなく切り替えることができます。上記のとおりオブジェクト名がデフォルトのスキーマに対して相対的になっている場合のみこれは可能です。

以下の点に注意してください：

- デフォルトのスキーマに対して相対的なオブジェクト名の使用はスキーマをサポートするデータベース型のために使用することができます: IBM DB2、IBM Informix, IBM Db2 for i (iSeries)、Oracle、PostgreSQL、Progress OpenEdge、SQL Server とSybase。
- この機能はMapForce バージョン2020 からサポートされています。
- ローカルリレーションシップ または 仮想テーブルとしてのSELECT ステートメント. がデータベースコンポーネントに含まれる場合相対的な名前を使用することはできません。

(C) 2015-2021 Altova GmbH

データベースオブジェクト名をデフォルトのスキーマに対して相対的に扱う方法:

1. マッピング上のデータベースコンポーネントのタイトルバーを右クリックし、「**データベースオブジェクトの追加/削除/編集**」をコンテキストメニューから選択します。
2. デフォルトのスキーマに属する1つまたは複数のオブジェクト、または *デフォルトのカタログ (データベース)* と*スキーマ*を選択します。デフォルトのデータベースとスキーマは太字で表示されています。下のサンプルでは、デフォルトのカタログは「Sandbox」で、デフォルトのスキーマは「user」です。この構造はSQL Server データベースに固有で、他のデータベース型とは異なる可能性があります。表示される構造に関しての詳細は、データベースオブジェクトの追加、編集、および削除を参照してください。

3. 「**デフォルトのスキーマに対して相対的なオブジェクト名を使用する**」チェックボックスを選択します。データベースが相対的なオブジェクト名をサポートしない場合チェックボックスは灰色で表示されることにご注意してください。

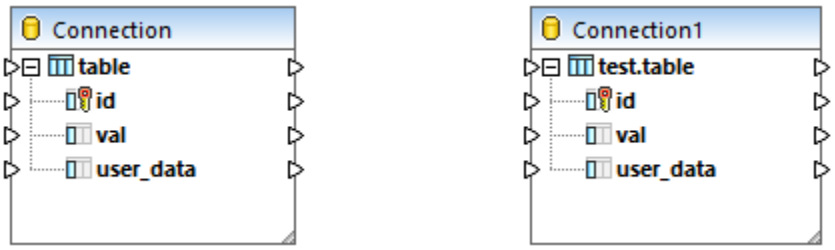
マッピング上で必要なオブジェクトがデフォルトではなく異なるスキーマ内に存在する場合以下の代替が存在します:

- 必要とされるデフォルトのスキーマへのアクセスを持つ他のデータベースユーザーとして接続します。
- 必要とされる特権を有することが前提で既存のデータベースユーザーのデフォルトのスキーマを変更するためにデータベースサーバーを再構成します。

SQL Server を例として、データベースの管理者は現在のカタログが「サンドボックス」でユーザーとスキーマが既存であることを想定して以下に示されるステートメントに類似したステートメントを使用してデータベースユーザーのデフォルトのスキーマを変更することができます。

```
USE [Sandbox]
GO
ALTER USER [test_user] WITH DEFAULT_SCHEMA=[test_schema]
GO
```

デフォルトのスキーマに相対的なデータベースオブジェクトはマッピング上で相対的な名前と共に表示されています。次の2つのデータベースコンポーネント間の差分を考慮してください:



左側のコンポーネントでは、「**デフォルトのスキーマに相対的なオブジェクト名を使用する**」チェックボックスが選択されており、このためにテーブルは相対的な名前と共に表示されています。右側のコンポーネントでは、チェックボックスは選択されておらず、テーブルにはスキーマ名が含まれています。

データベースオブジェクトが相対的な方法で参照されている場合、*マッピング接続を失うことなしに*マッピングを新規のデータベースまたはスキーマに以下のように変更することができます:

- データベースコンポーネント設定を開き、「**変更**」をクリックします。ウィザードステップに従い、新規のデータベースに新しいユーザーとして接続します。オブジェクトを相対的な名前で修飾した場合、新規のデータベースは同じ構造を持ちます。そして、マッピング上のすべての接続は自動的に更新されます。具体的には、*新規のデータベースユーザーのデフォルトカタログとスキーマにマッチします*。
- 切り替えを再帰的なベースで行う必要がある場合、データベース接続をグローバルリソースとして定義することが奨励されます。例えば、グローバルリソースは2つの構成を持つことができます: 開発データベースのためのデフォルトの構成と生産のための構成を持つことができます。両方のデータベースが同じ構成を持つことを前提とし、ドロップダウンリストから必要とされる構成を変更してデータベース間を切り替えることができます:

  このシナリオの詳細に関しては例: データベースの切り替えを参照してください。

データベースオブジェクトが切り替え後赤い色で表示されます。これは新規のデータベーススキーマではこれらのデータベースオブジェクトが存在しないことを指します。

コード生成

「**デフォルトのスキーマに対して相対的にオブジェクト名を使用する**」チェックボックスは生成されたC#、C++、またはJavaプログラムコードに影響を与えます。このチェックボックスが選択されていると、全てのデータベースレファレンスが生成されたコード内で相対的になります。次のサンプルはチェックボックスが選択されていない生成された(C#) SQLステートメントのサンプルです。テーブル名にスキーマ名が含まれていることに注意してください。

```
var3_NewStatement = (Altova.Db.Dbs.NewStatement(closure.var1_Connection_Instance, 1,
"SELECT [id], [val], [user_data] FROM [user].[table]"));
```

次はチェックボックスが選択された生成されたSQLステートメント(C#)のサンプルです:

```
var3_NewStatement = (Altova.Db.Dbs.NewStatement(closure.var1_Connection_Instance, 1,
"SELECT [id], [val], [user_data] FROM [table]"));
```

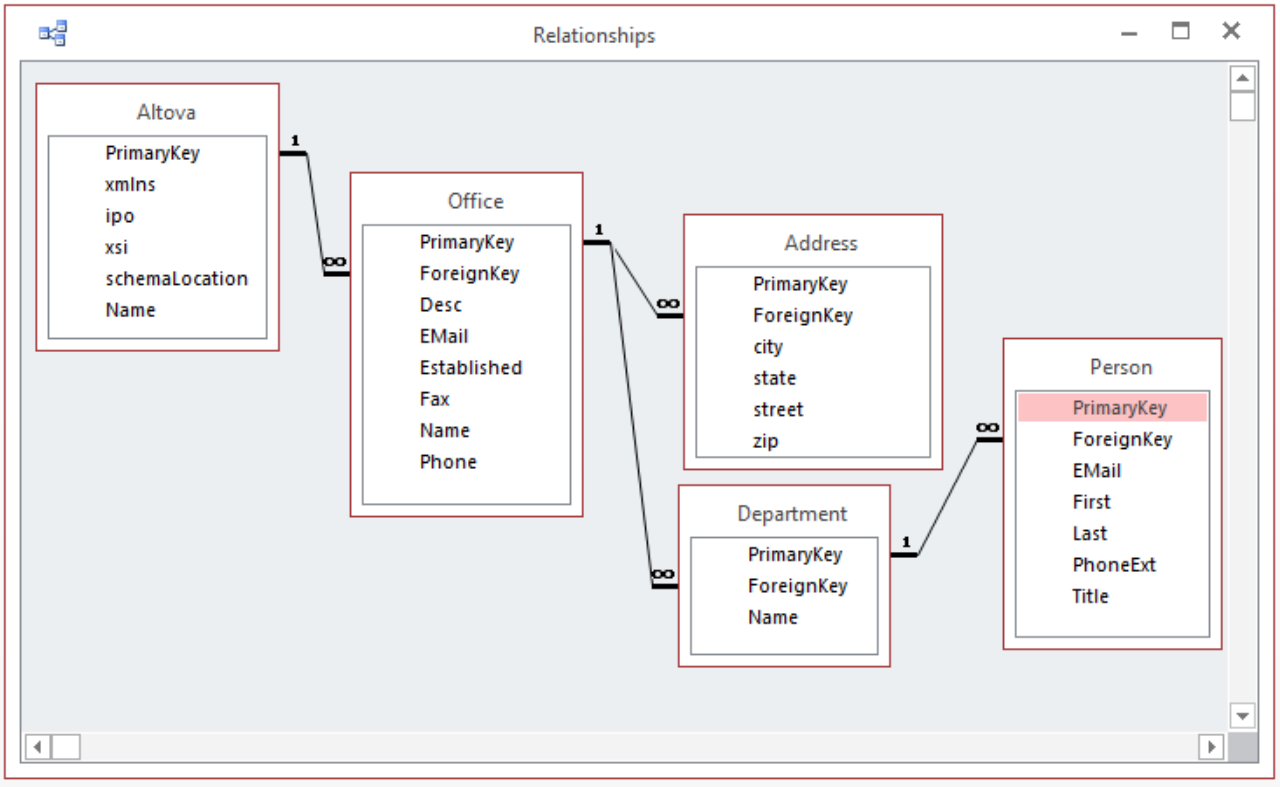

(C) 2015-2021 Altova GmbH

メモ　スキーマ名を削除するためにレガシーな方法が存在します。「テーブル名からスキーマ名を削除する」チェックボックスを データベースコンポーネント設定をから選択します。レガシーなアプローチは下位互換性のみのためにサポートされますが、回避されるべきです。

6.2.2.5 データベースリレーションシップの処理

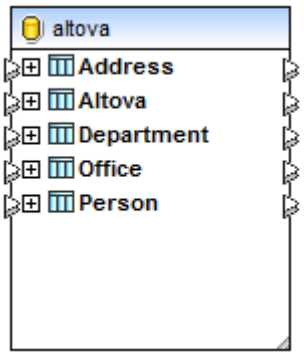
リレーショナルデータベースは、その名前が示すように、テーブル間にリレーションシップを持ちます。フォルダー <**マイドキュメント>\Altova\MapForce2021\MapForceExamples\Tutorial** にある **altova.mdb** データベースを例にしてみると、複数のリレーションシップが存在することに気がつきます:

- サンプルの会社（テーブル「Altova」に対応する）1つまたは複数のオフィスから構成されています（例えば、1つは、東京に、もう1つは名古屋に存在するとします）。データベース用語では、「Altova」と「Office」テーブル間に1つ対複数のすなわち、「Altova」テーブル内の各「PrimaryKey」レコードのために、「Office」テーブル内で複数の「ForeignKey」レコードが存在することができます。「ForeignKey」値が、「Altova」内の「PrimaryKey」値と対応する「Office」の箇所は、「Altova」のオフィスとして考えされます。
- 各オフィスは、1つまたは複数の部署（例えば、「Marketing」、「IT」、「Development」）から構成されています。「Office」と「Department」テーブル間に1つと複数のリレーションシップが存在します。
- 最後に、各部署は、複数の社員から構成されています。ですから、1つ対複数のリレーションシップは、「Department」と「Person」テーブル間のリレーションシップです。



altova.mdb データベース内のテーブルリレーションシップ（Microsoft Access "リレーションシップ" ビュー）

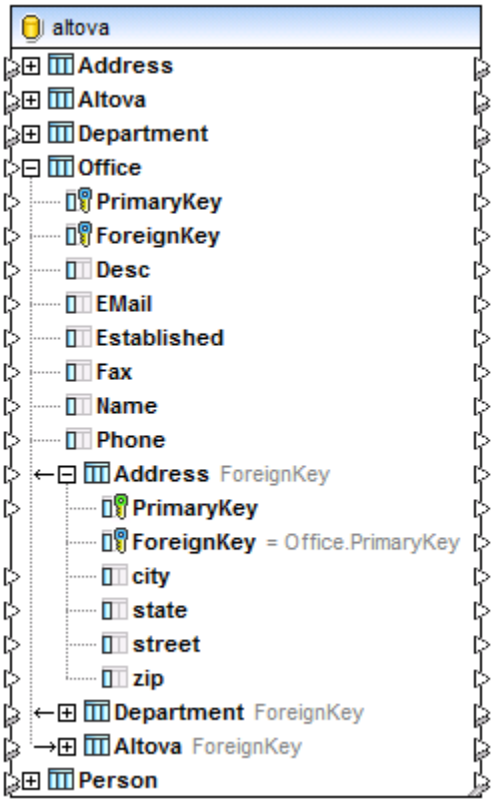
データベーステーブル間のリレーションシップは、マッピングの目的にとって重要です。MapForce は、データベースをマッピングに追加する際に、このようなデータベースリレーションシップの個々の変更を記録します。これにより、データをデータベースにマッピングする際、またはデータベースにデータをマッピングする際、データベースリレーションシップを保持することができます。この概念をより良く理解するために、（**挿入 | データベース** メニューコマンドを使用して）**altova.mdb** データベースをマッピングに追加します。「ルート」テーブルの下のテーブルを呼び出します:



「ルート」テーブル

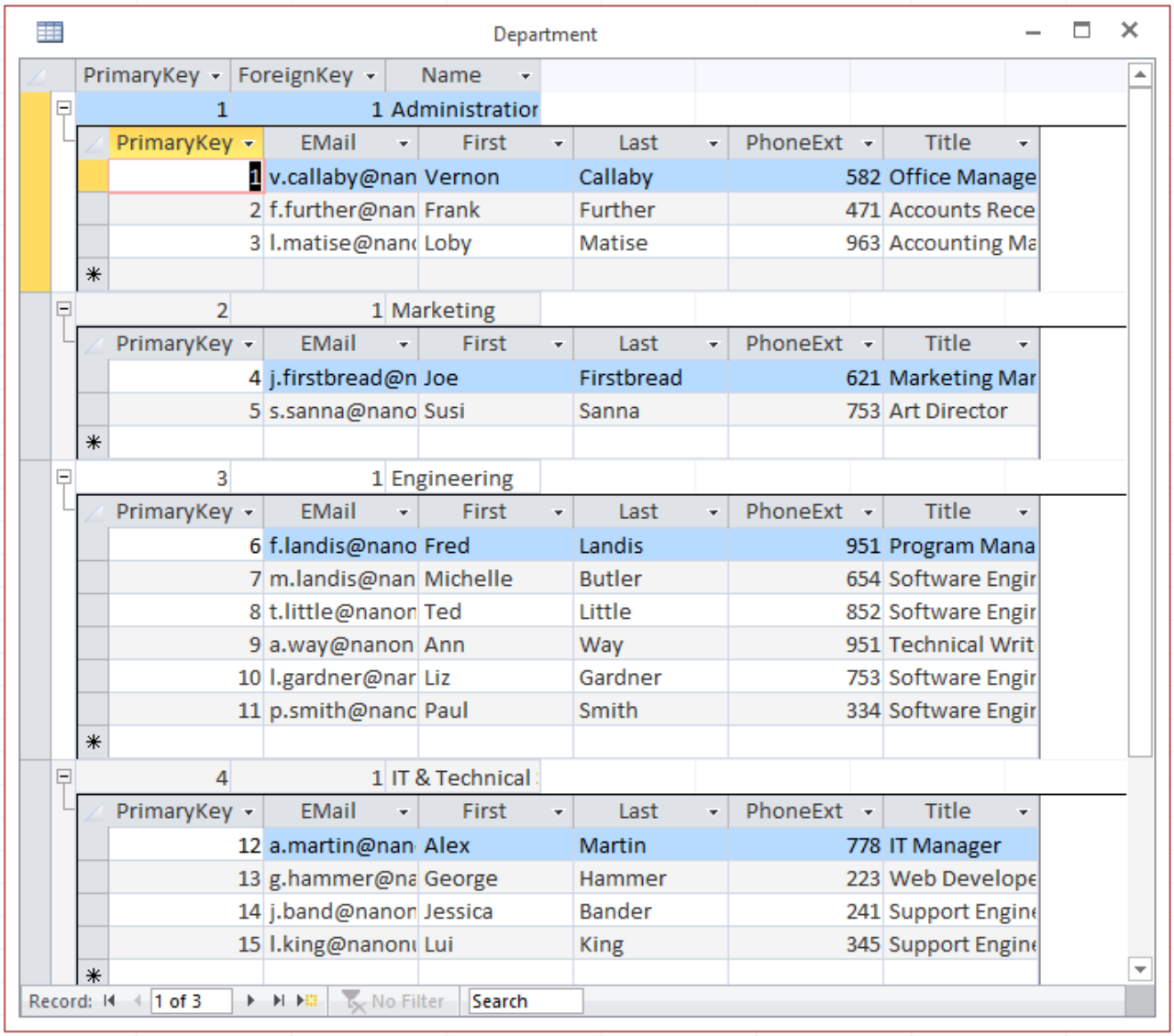
「ルート」テーブルを展開すると、ツリービュー内に存在する、すべての関連するテーブルが表示されます。例えば、**Office** テーブルを展開すると、関連したテーブルの階層が表示されます:

- テーブルの前の左矢印 (←) は、このテーブルが子テーブルであることを表示しています。例えば、**Address** は、**Office** の子です。**Department** も **Office** の子であり、また、**Address** の兄弟テーブルであり、同じインデントのレベルに存在します。表示されているように、マッピング上のリレーションシップは、上の「リレーションシップ」ダイアグラムに対応しています。
- テーブルの前の右矢印 (→) は、このテーブルが親テーブルであることを表示しています。例えば、**Altova** は、**Office** の親です。



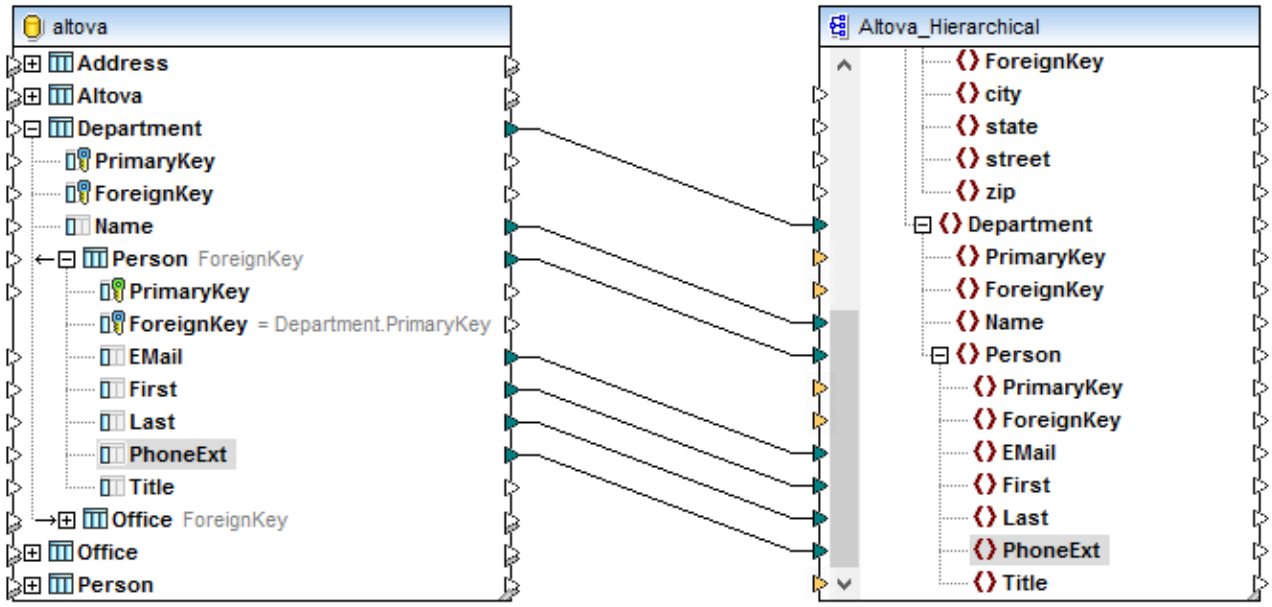
MapForce 内のテーブル リレーションシップ (altova.mdb データベース)

テーブルの階層の表示は、マッピングがデータベースから読み取る、またはデータベースへ書き込む際に既存のデータベースリレーションシップを保存する手助けをします。例えば、**Person** テーブルから全てのレコードを取得し、XML ファイルに各個人を部署にグループ別に入力するとします。具体的には、XML ファイルは、この例で使用されている **altova.mdb** データベースのように、各個人を部署にリンクします:



上記のように、「Administration」部署は、3人の社員、「Marketing」部署には、2人の社員、「Engineering」部署には6人の社員が所属しています。

この データベースからデータをマッピングする場合、各社員が正確な部署に割り当てられる場合、「ルート」テーブルとして、Department を使用することは重要で、Department の子である Person テーブルからマップします：

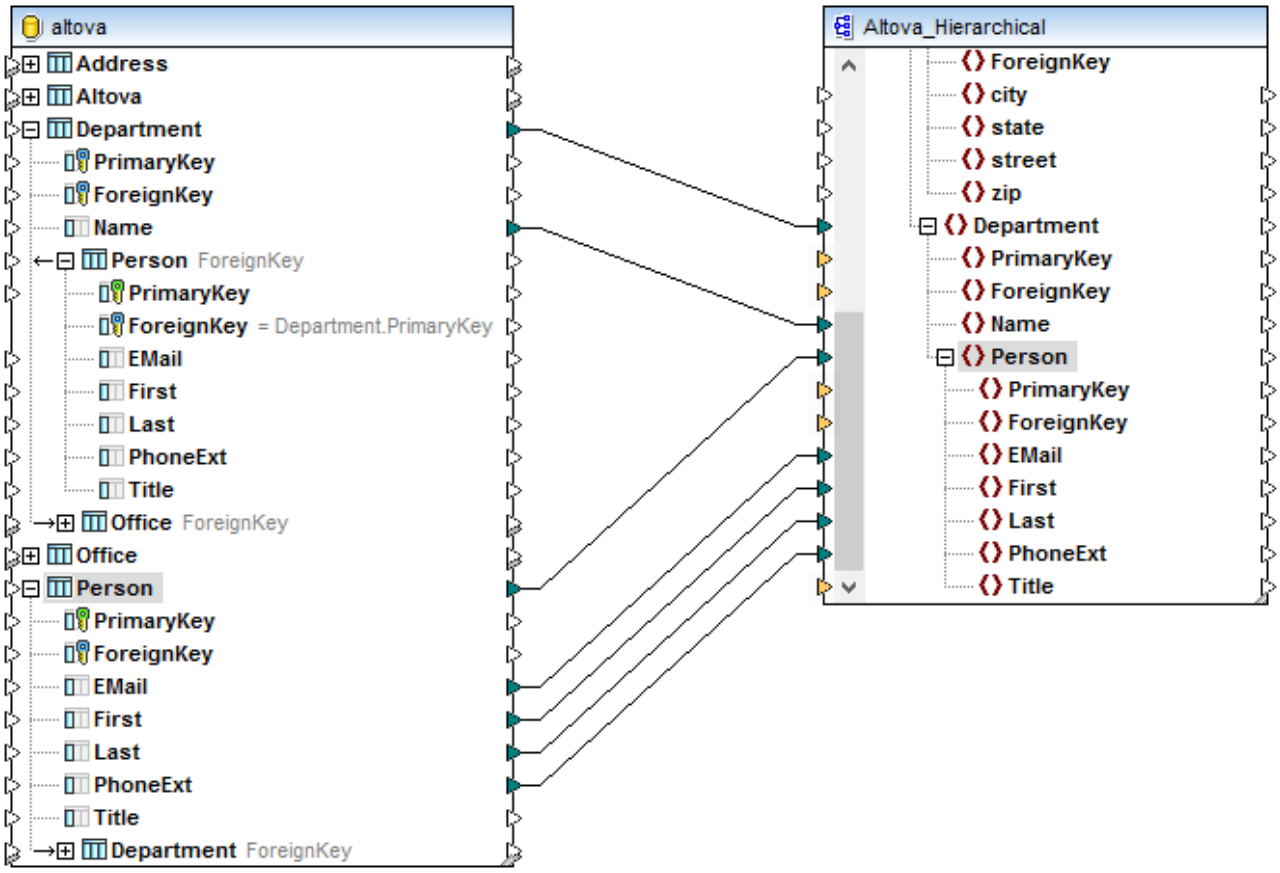


上のマッピングは、〈マイ ドキュメント〉¥Altova¥MapForce2021¥MapForceExamples¥ フォルダー内の DB_Altova_Hierarchical.mfd ファイルに変更を加えたマッピングです。マッピングをプレビューする際には、各個人の結果は、意図とされる振る舞いであるとおり、部署別にグループ化されます。「Administration」には 3 人、「Marketing」には 2 人、「Engineering」には6 人、などの結果が得られます。

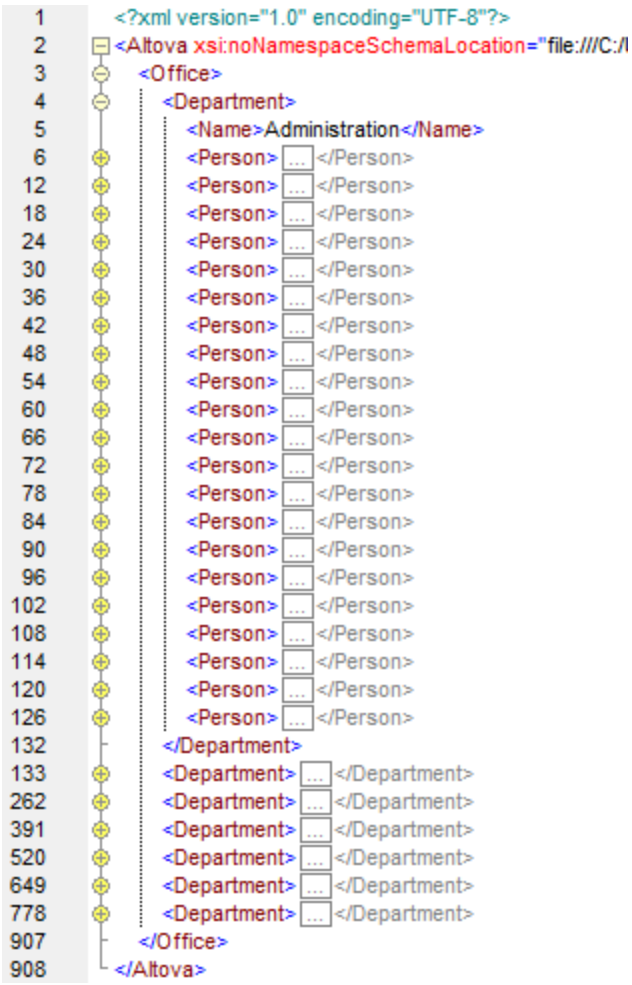
```
<?xml version="1.0" encoding="UTF-8"?>
<Altova xsi:noNamespaceSchemaLocation="file:///C:
  <Office>
    <Department>
      <Name>Administration</Name>
      <Person> ... </Person>
      <Person> ... </Person>
      <Person> ... </Person>
    </Department>
    <Department>
      <Name>Marketing</Name>
      <Person> ... </Person>
      <Person> ... </Person>
    </Department>
    <Department>
      <Name>Engineering</Name>
      <Person> ... </Person>
      <Person> ... </Person>
      <Person> ... </Person>
      <Person> ... </Person>
      <Person> ... </Person>
      <Person> ... </Person>
    </Department>
    <Department> ... </Department>
    <Department> ... </Department>
    <Department> ... </Department>
    <Department> ... </Department>
  </Office>
</Altova>
```

若干修正の加えられた下のマッピングを確認してください。**Department** と **Person** の両方に接続が意図的に描かれているため両方が「ルート」テーブルとなっています。

(C) 2015-2021 Altova GmbH



マッピングをプレビューすると、（ソースの部署に関係なく）すべての個人は、各ターゲット部署の下にグループ別に表示されています。これは意図とされた振る舞いではありません。つまり、つまり、「Administration」に21人の社員が存在し、「Marketing」に21人の社員が存在し、「Engineering」に21人の社員が存在するなどという結果を指します。

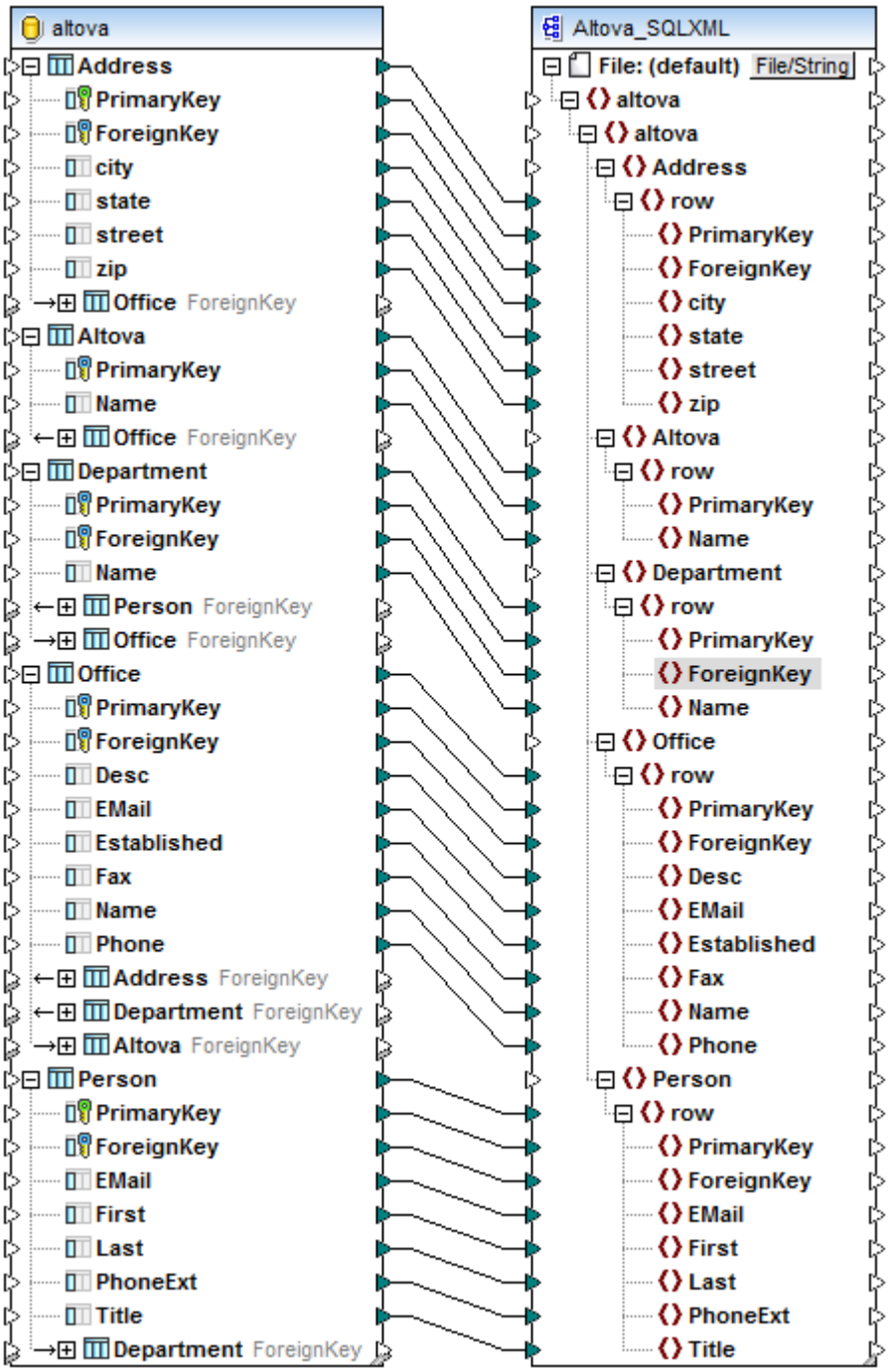


第2番目の例では、データベースリレーションシップは、接続の方法のため無視されています。

ですから、データベースリレーションシップを保持する場合、リレーションシップを保持する子テーブルを含むことを、または同じ「ルート」テーブルから接続が描かれていることを確認してください。これは、ソースとターゲットデータベースと同じように作動します。リレーションシップを保持するデータベースマッピングの例に関しては <マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥フォルダー内にある **DB_Altova_Hierarchical.mfd** と **Altova_Hierarchical_DB.mfd** ファイルを参照してください。(次も参照してください: データを複数にリンクされているテーブルに挿入する)。

データベースリレーションシップを保持しない場合もあります。例えば、**altova.mdb** データベースからの全てのデータを、SQL/XML 仕様 (Structured Query Language (SQL) 仕様のパート 14 に準ずる) フラットな XML ファイルにエクスポートすると想定します。この種類のマッピングは、<マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥フォルダ内の **DB_Altova_SQLXML.mfd** サンプルにより説明されています。マッピングの目的は、フラットな XML ファイルとしてデータベースデータを取得することです。ターゲット SQL/XML スキーマは、XMLSpy で「**変換 | DB 構造から XML スキーマを作成する**」メニューコマンドを使用して生成することができます。

(C) 2015-2021 Altova GmbH



DB_Altova_SQLXML.mfd

上記のように、各データベーステーブルには、対応する要素がターゲット XML 内に存在します。マッピングの結果をプレビューすると、実際のデータベースの行は、ターゲット内の「行」要素に書き込まれていることが確認されます。

```
<?xml version="1.0" encoding="UTF-8"?>
<altova xsi:noNamespaceSchemaLocation="file:///
  <altova>
    <Address>
      <row> ... </row>
      <row> ... </row>
    </Address>
    <Altova>
      <row> ... </row>
    </Altova>
    <Department>
      <row> ... </row>
      <row> ... </row>
      <row> ... </row>
      <row> ... </row>
      <row> ... </row>
      <row> ... </row>
      <row> ... </row>
    </Department>
    <Office>
      <row> ... </row>
      <row> ... </row>
    </Office>
    <Person>
      <row> ... </row>
      <row> ... </row>
      <row> ... </row>
      <row> ... </row>
      <row> ... </row>
      <row> ... </row>
      <row> ... </row>
      <row> ... </row>
      <row> ... </row>
      <row> ... </row>
      <row> ... </row>
      <row> ... </row>
      <row> ... </row>
      <row> ... </row>
      <row> ... </row>
      <row> ... </row>
      <row> ... </row>
      <row> ... </row>
      <row> ... </row>
      <row> ... </row>
      <row> ... </row>
    </Person>
  </altova>
</altova>
```

XML 出力が示すように、XML 要素間には、階層は存在しません。これはフラットな SQL/XML 構造です。複数の「ルート」テーブルからのデータが内部でマップされたために、データベースリレーションシップは、無視されます。

6.2.2.6 ローカルリレーションシップの定義

データベーステーブルが明示的にリレーションシップを定義していない場合、MapForce 内でそのようなリレーションシップをローカルに定義することができます。特に、MapForce 内から、異なるテーブルの列の間の主キーと外部キーリレーションシップをデータベースに影響を与えることなく作成することができます。すべてのデータベース列を主キーまたは外部キーを作成するために使用することができます。また、データベー

(C) 2015-2021 Altova GmbH

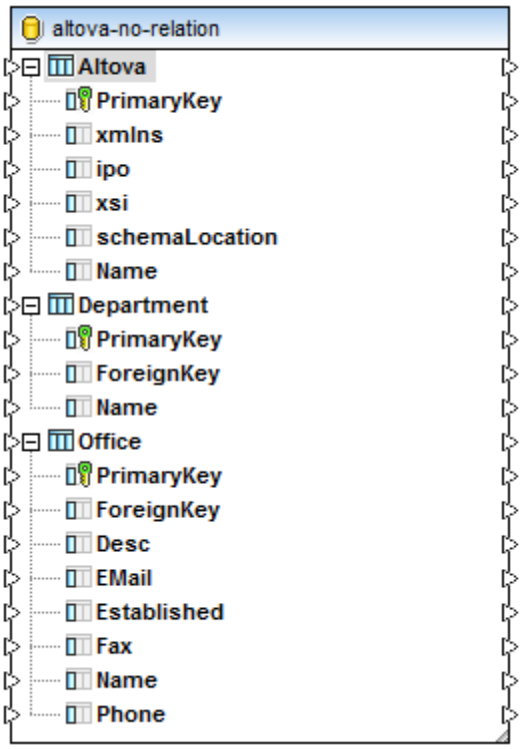
スキーマに既存のリレーションに加え、新規のリレーションを素早く作成することができます。ローカルで定義されたリレーションシップはマッピングと共に保存されます。

これらの素早く作成されたリレーションシップは、MapForce 内では**ローカルリレーション**と呼ばれます。次のテーブルはローカルリレーションを定義することのできるすべての可能なフィールドをリストしています（例えば、ストアドプロシージャの出力をデータベース列にマッピングするなど）。リレーションシップ内の一部のフィールドは、同じ、または互換性のあるデータ型を持つ必要があります。

主/ユニークキー	外部キー
• データベーステーブル、またはビューの列 • ストアドプロシージャの出力パラメーターまたは戻り値。ストアドプロシージャを参照してください。 • ストアドプロシージャにより返されたレコードセットの列* • ユーザー定義 SELECT ステートメントの列。仮想テーブルとしてのSQL SELECT ステートメント を参照してください。	• データベーステーブル、またはビューの列 • ストアドプロシージャの入力パラメーター • ユーザー定義 SELECT ステートメントの入力パラメーター

**（パラメータを持たない）データソースまたは、（入力と出力パラメーターを持つ）関数としてストアドプロシージャが呼び出された場合適用することができます。セクションのためにレコードセットが使用できるようになるには、レコードセットを取得するためにストアドプロシージャを一度実行する必要があります。*

この例で使用されている、**altova-no-relation.mdb** データベースは、MapForce により与えられる **altova.mdb** データベースの簡素化されたバージョンです。Person ならびに Address テーブルや、関連するテーブルが Microsoft Access。**altova-no-relation** ツリーにあるテーブルは、全て「ルート」レベルのテーブルで、子テーブルはありません。各テーブルのエンテンツは、そのテーブルに含まれるものだけに限定されています。

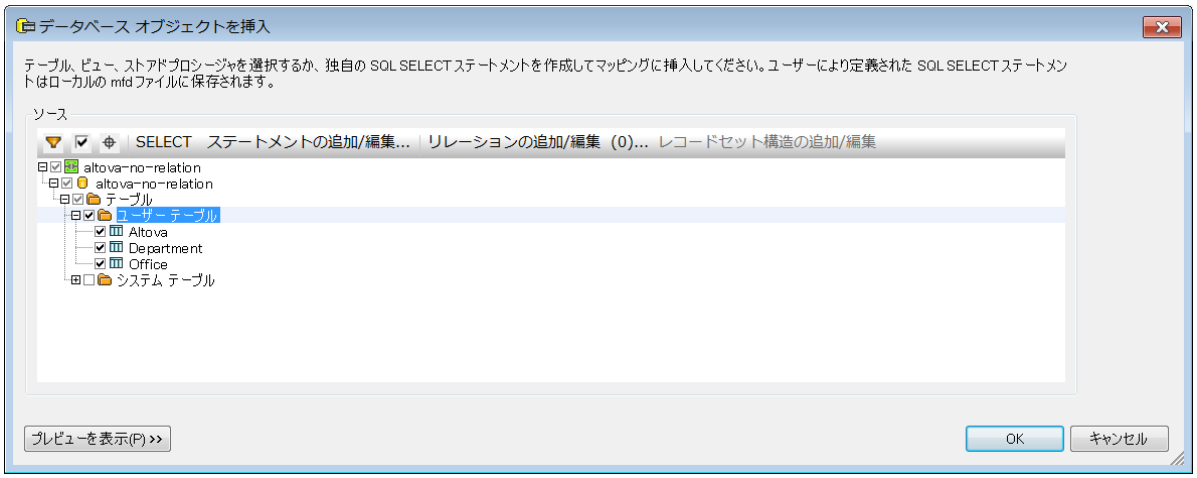


明示的なリレーションシップを持たないデータベース構造

この例の目的は、「Altova」のオフィスを表示し、各オフィス内の部署を表示することです。**altova-no-relation.mdb** 内では、主キーと外部キーリレーションシップは、上記のように、明示的には存在しません。このリレーションシップは、論理的に（明示的に）存在し、マッピングの目的を完成するために ローカルで再作成されます。

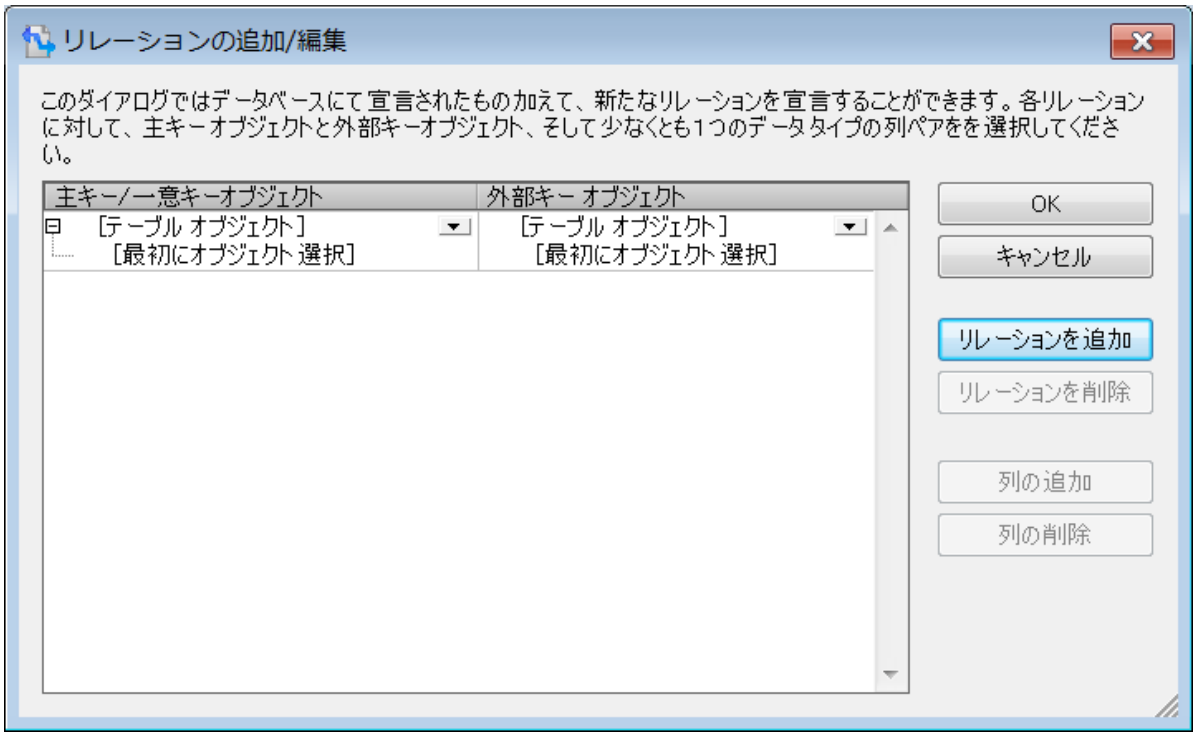
ローカルリレーションは、データベースをマッピングに追加中に定義することができます、または既存の データベース コンポーネントを右クリックして、または コンテキスト メニューから「**データベース オブジェクトの追加/削除/編集**」を選択して定義することができます。

1. 「**挿入**」メニューから、「**データベース**」をクリックします。
2. 「接続ウィザード」内の **Microsoft Access (ADO)** をクリックして、「**次へ**」をクリックします。
3. <**マイ ドキュメント**>¥Altova¥MapForce2021¥MapForceExamples¥Tutorial¥ フォルダー内にある **altova-no-relation.mdb** データベースを参照し、「**接続**」をクリックします。
4. **ユーザーテーブル** チェックボックスを選択します。

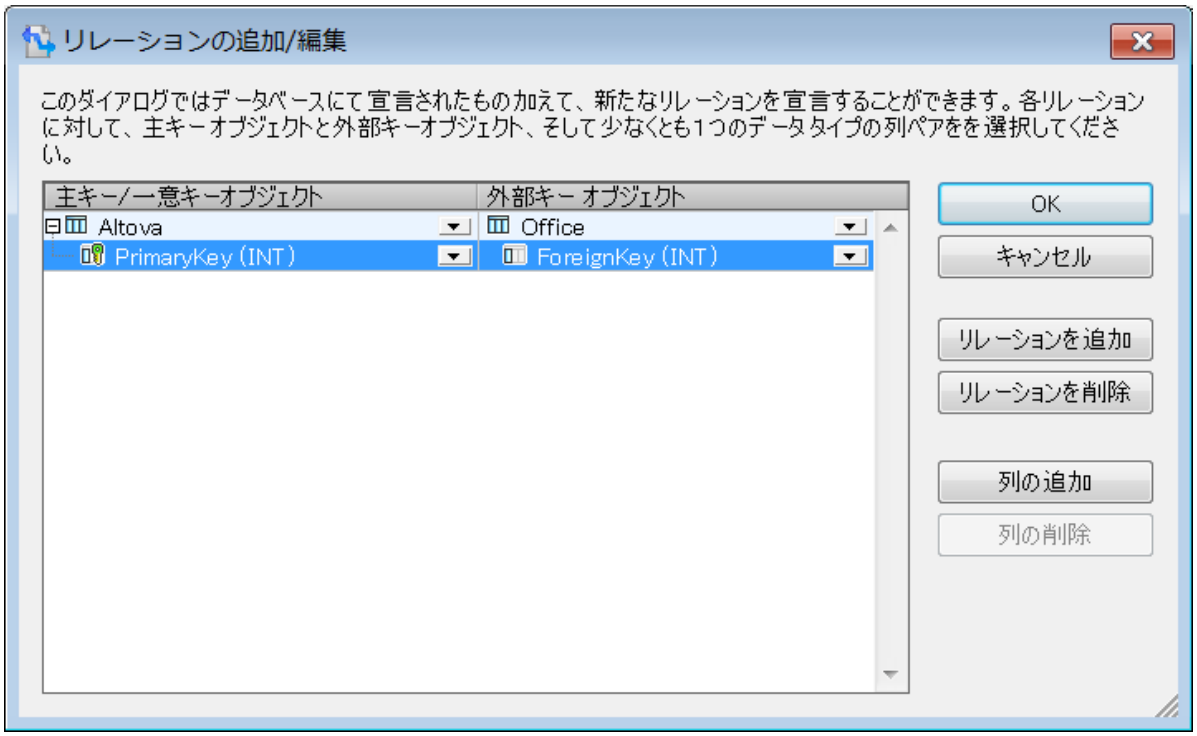


5. アイコンバーにある**リレーション**を追加/編集ボタンをクリックします。
6. **リレーション**を追加/編集ダイアログボックスが開かれますので、**リレーション**を追加ボタンをクリックします。

(C) 2015-2021 Altova GmbH

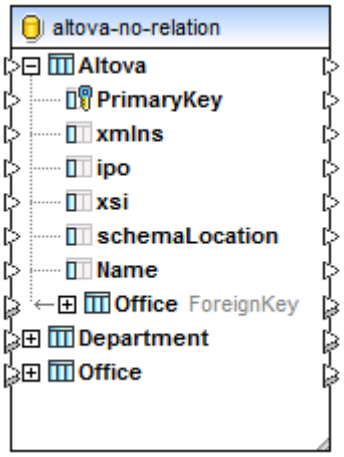


7. 2 つのコンボボックスによりテーブルまたはデータベースオブジェクト間のリレーションを選択することができます。左側のコンボボックスが主キー/ユニークキーのテーブルで、右側が外部キーのテーブルになります。2 つのドロップダウンリストにより、リレーションを作成するテーブル、またはデータベースオブジェクトを選択することができます。「主/一意のキー」を保管する左側のリストは、オブジェクトを指定し、右側のリストは外部キーオブジェクトを指定します。主キー/一意キーオブジェクトが MapForce における親オブジェクトとなり、外部キーオブジェクトがデータベースコンポーネントの子として表示されます。(次も参照してください: データベースリレーションシップの処理)。

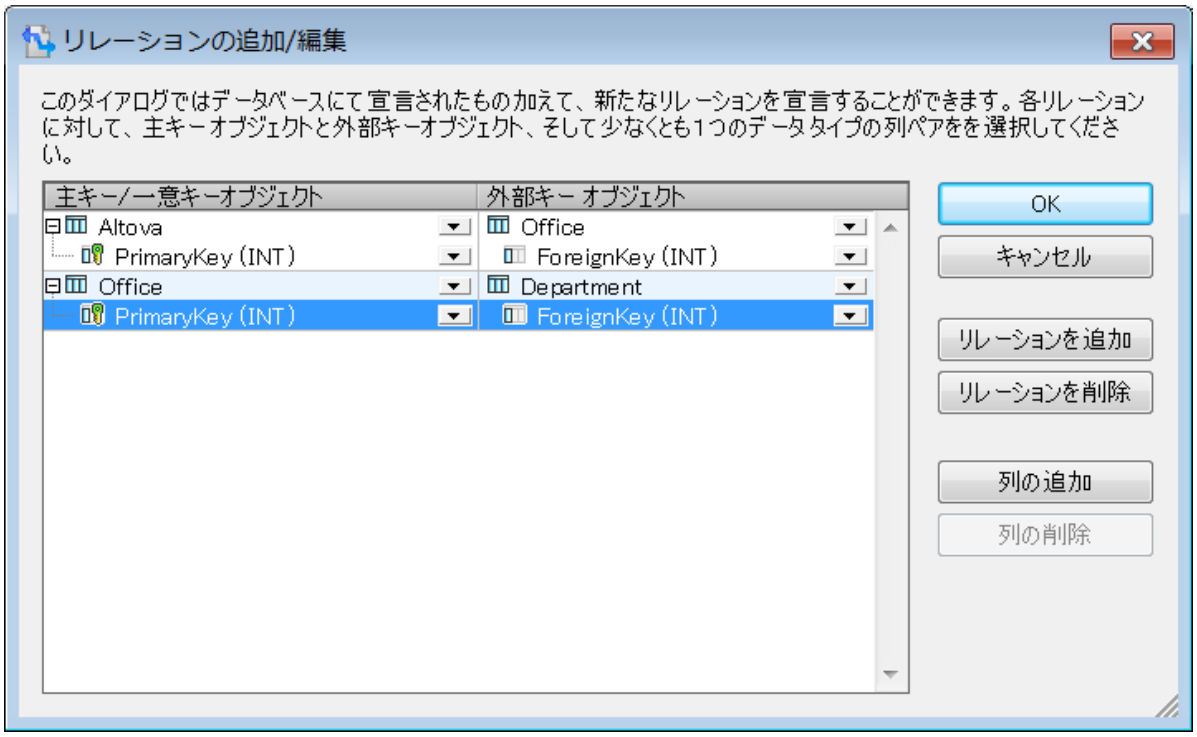


8. 「OK」ボタンをクリックしてローカルリレーションの定義を完了し、「挿入」ボタンをクリックして、データベースをマッピングエリアへ挿入します。

この段階では、「Altova」テーブルの「PrimaryKey」列と「Office」テーブルの「ForeignKey」列の間にローカルリレーションシップを作成します。下に表示されるとおり、「Altova」ルートテーブルは、「Office」テーブルの親になりました。具体的には、Office テーブルは、Altova テーブルの下の、自身の展開することのできるアイコンを持つ、関連するテーブルとして表示されています。



しかしながら、マッピングの目的はまだ達成されていません。マッピングの目的を完成するために、下に表示されるとおり、**Office** と **Department** テーブル間のリレーションシップを作成するメソッドと同じメソッドを使用します。

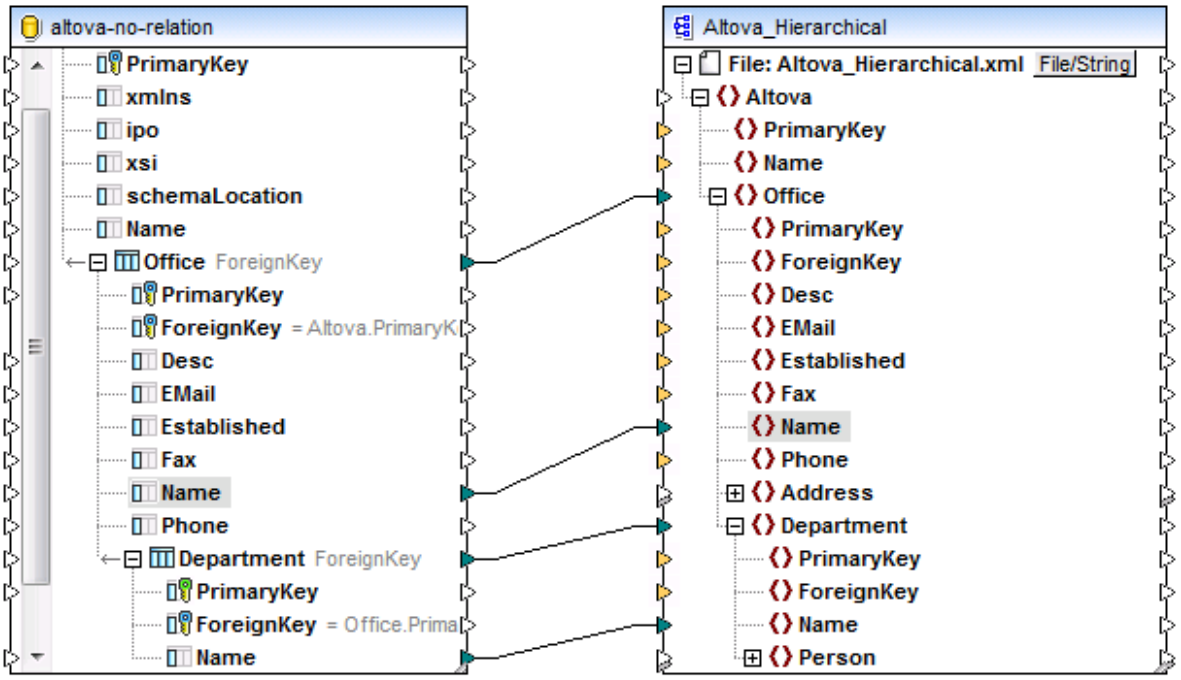


「リレーションの追加/編集」ダイアログボックスを再度開くには、データベースコンポーネントを右クリックし、「データベース オブジェクトの追加/削除/編集」コンテキストメニューから選択します。

最後に、ターゲットスキーマをマッピングに以下のように追加します:

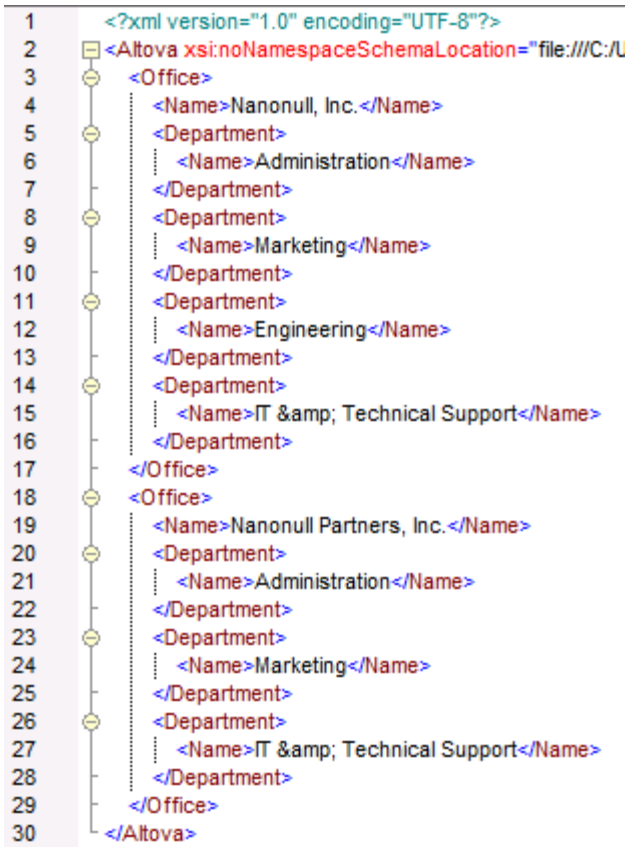
1. 「挿入」メニューから、「XML スキーマ/ファイルの挿入」をクリックします。
2. <マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥フォルダー内で使用することのできる **Altova_Hierarchical.xsd** ファイルを参照します。サンプルXML ファイルを与えるようにプロンプトされると、「スキップ」をクリックします。ルート要素を選択するようにプロンプトされると、「Altova」を選択します。

ターゲット XML 内のテーブル間のリレーションシップを保持するために、全ての接続は、同じ「ルート」テーブルから、階層的(この場合、「Altova」)に描かれています。詳細に関しては、次を参照してください: データベースリレーションシップの処理。



マッピング上記のように、マッピングを定義し、「出力」タブをクリックして、結果をプレビューします。マッピングの結果は、このマッピングの目的のとおり、それぞれのオフィスの下にネストされた部署の要素を表示しています。

(C) 2015-2021 Altova GmbH



6.2.2.7 データベースを変更するマッピングの実行

(例えば、レコードの挿入、更新、削除など) マッピングがデータベースのデータを変更すると、変更は、マッピングを実行したエンジンによりデータベースに適用されます。マッピングを実行するエンジンは、MapForce、(スタンドアロン または FlowForce Server 管理下での) MapForce Server、または C++、C#、または Java のために生成されたコードの実行環境などです。

(「出力」タブをクリックして、) MapForce 内で直接マッピングの結果をプレビューすると、更新スクリプトが表示されます。スクリプトは、このアクションを明示的に実行するまで、データベースに対して実行されません。これはプレビューのためのみに使用することができます。

```
/*
The following SQL statements are only for preview and may not be executed in another SQL query tool!
To execute these statements use function "Run SQL-script" from menu "Output".
Connect to database using the following connection-string:
Data Source=C:/Users/altova/Documents/Altova/MapForce2016/MapForceExamples/AltovaTarget.mdb;Provider=Microsoft.Jet.OLEDB.4.0
*/

-- begin transaction

DELETE FROM [Address]

DELETE FROM [Person]

DELETE FROM [Department]

DELETE FROM [Office]

DELETE FROM [Altova]

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

INSERT INTO [Altova] ([Name], [PrimaryKey]) VALUES ('Organization Chart', '%PrimaryKey1%')
```

データベースを変更するマッピングの出力プレビュー(Altova_ Hierarchical_ DB.mfd)

スクリプトは疑似-SQL ステートメントを情報提供の目的で表示しています。実行されようとしているステートメントを直接反映していません。例えば、("Update if... Insert Rest" などの) 複数のアクションがテーブルに対して定義されている場合、プレビュー内では、2番目のアクションは条件付きで実行されるため最初のアクションのみが表示されています。

疑似-SQL スクリプトは上記の実行エンジン以外 SQL ツールを使用してデータベースに手動で適用されることはできません。スクリプトは外部 SQL エディターにより理解不可能な値のフォーマットを含んでいる可能性があります。

マッピングへの変更をデータベースへ直接 MapForce から適用する場合、「出力」メニュー内で使用することのできる「SQL-スクリプトの実行」コマンドをクリックします。このアクションは、実際にデータベースを変更し、効果はすぐに適用されます。

(スタンドアロン または FlowForce Server 管理下で) MapForce Server でマッピングが実行される場合、データベースへの変更はすぐに適用されます。生成されたコード内でも同様が発生します。コードをコンパイルし実行した際に、データベースへの変更は適用されます(例えば、Visual Studio 内で「実行」コマンドをクリックしてなど)。

MapForce のインストールには、<マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥フォルダー内に存在する、複数のサンプルデータベース(Microsoft Access または SQLite ファイル) が含まれます。MapForce から「SQL-スクリプトの実行」コマンドを使用して、MapForce により与えられるサンプルのデータベースに対してデータベースへの変更を適用することが奨励されません。これは、サンプルが使用できなくなる可能性があるからです。オリジナルのデータへの上書きを回避するために、<マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥フォルダー全体のバックアップをファイルを更新する前に作成することが最善策です。

MapForce 以外の実行環境内でのマッピングの実行に関する情報は、以下を参照してください:

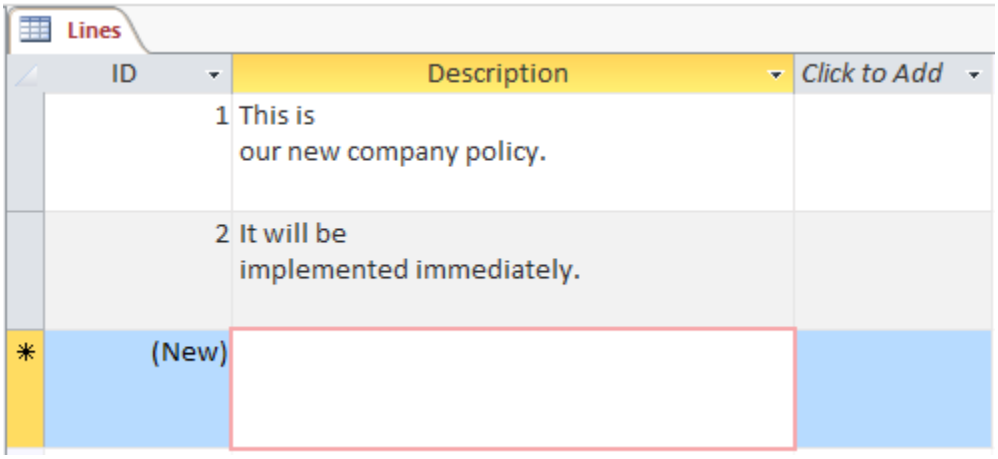
- FlowForce Server にマッピングをデプロイする
- MapForce Server 実行可能ファイルにマッピングをコンパイルする
- コード ジェネレーター

(C) 2015-2021 Altova GmbH

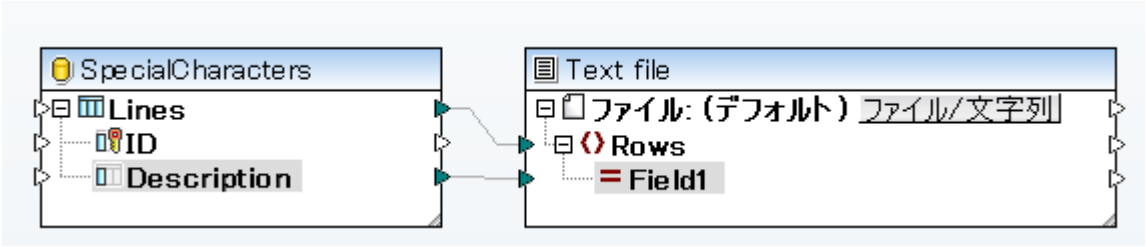
6.2.2.8 特殊文字の使用

データベースデータを変換する際に、データソースから、Enter キー/改行（CR/LF）文字などの、固有の特別文字を削除する必要がある可能性があります。ソース。これは、MapForce ライブラリ関数`char-from-code` を使用して行うことができます。

テーブル「Lines」は、2つの列（「ID」と「Description」）から構成されています。Microsoft Access データベースが以下に表示されています。



目的は、CSV ファイルの詳細（ラインごとの詳細）を抽出することです。ですから、この目的を達成するためのマッピングは以下に類似します:

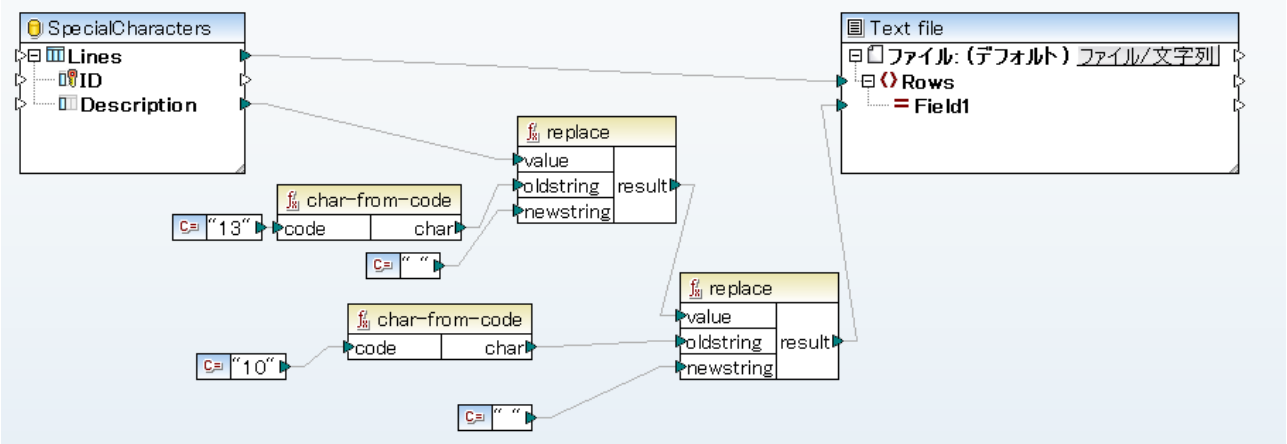


しかしながら、Access 内の各「Description」行は、意図する結果ではない、改行を含むCR/LF 文字により区切られた複数のラインを含んでいます:

```
1    "This is
2    our new company policy."
3    "It will be
4    implemented immediately."
5
```

この問題を解決するために、マッピングにMapForce ビルトイン ライブラリから`char-from-code` と`replace` 関数 を追加します（マッピングに関数を追加するも参照してください）。上記で説明されている文字が発生すると、スペース文字で置き換えられるように、すべての詳細が処理されなくてはなりません。

Unicode チャート（http://www.unicode.org/charts/）では、LF とCR 文字 は、**hex 0A | dec 10** と**hex 0D | dec 13** 文字にそれぞれ対応します。ですから、マッピングは、更に`replace` 関数を使用した処理を行うために、小数のUnicode 値 13を文字列に変換するように変更されなければなりません。



マッピングをこの時点でプレビューすると、各データベースフィールド内のCR/LF 文字 がスペースと置き換えられていることにご注目してください。

```
1    This is  our new company policy.
2    It will be  implemented immediately.
3
```

6.2.2.9 Null 値の処理

マッピングランタイムでデータベースフィールドがnull 値であるかをチェックするには、`is-null` と`is-not-null` MapForce ライブラリ関数を使用します。MapForce から、テーブル内にnull フィールド が存在するかを確認するには、データベース ブラウザーを使用してクエリします(次を参照: データベースの参照とクエリ)。

データベースフィールドをnull に設定するために、 `set-null` 関数を使用します。

データベースのnull 値を文字列と置き換えるには、 `substitute-null` 関数を使用します。この機能を説明する **DB_ApplicationList.mfd** は、<マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥フォルダー内にあります。

データベースを更新するマッピング内のNULL 値の比較に関しては、データベーステーブルアクション内のNull 値の処理を参照してください。

XML ドキュメントから、マッピングをデータベースにマップする、またはデータベースからXML ドキュメントにデータをマッピングする際のnull 値の処理に関する情報は、次を参照してください:Nil の値 / Nillable。

6.2.2.10 連続する、および、一意の値の生成

データを挿入し、データベースを更新するには、マップされたソースからの入力がないデータベースフィールドのために シーケンシャル、または一意の値を作成する必要がある場合があります。このような場合には、次のビルトイン MapForce ライブラリ関数を使用します:

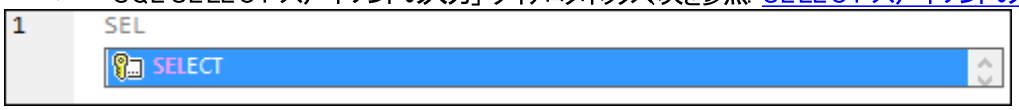
- `auto-number` ("core | generator functions" ライブラリ内で使用することができます)。この関数は、数値フィールドのための主キーの値を生成するために通常使用されます。
- `create-guid` ("lang | generator functions" ライブラリ内で使用することができます)。この関数は、 特定のフィールドのためのグローバルに一意の識別子(16 進でエンコードされた文字列として) を作成するために使用されます。

データベースにより生成された値を使用してデータベースフィールドのための値を書き込むこともできます。このオプションは、データベース テーブル アクション ダイアログボックスで使用することができ、主キーを生成する際にとても役にたちます(次を参照: データベーステーブルアクション設定) 。

6.2.2.11 SQL 自動補完の提案

特定のコンテキスト内にSQL ステートメントを入力すると、MapForce は、テキストのエントリを自動的に提案する場合があります。自動補完を次のコンテキストで使用することができます:

- SQL Editor (次を参照: データベースの参照とクエリ)
- 「データベーステーブルアクション」ダイアログボックス内の「カスタムSQL」テキストボックス(次を参照: データベーステーブルアクションの設定)
- 「SQL SELECT ステートメントの入力」ダイアログボックス(次を参照: SELECT ステートメントの作成)



自動補完

「上へ」と「下へ」キーボードキーを使用して、提案のリスト内を移動します。提案されたエントリを選択するには、クリック、または「Enter」を押します。

自動補完の提案の無効化:

1. 「ツール」メニューから「オプション」をクリックします(または**Ctrl+Alt+O**を押します)。
2. 「データベース」から、「SQL エディター」をクリックします。
3. 「入力ヘルパー」から、「自動的に開く」チェックボックスをクリアします。

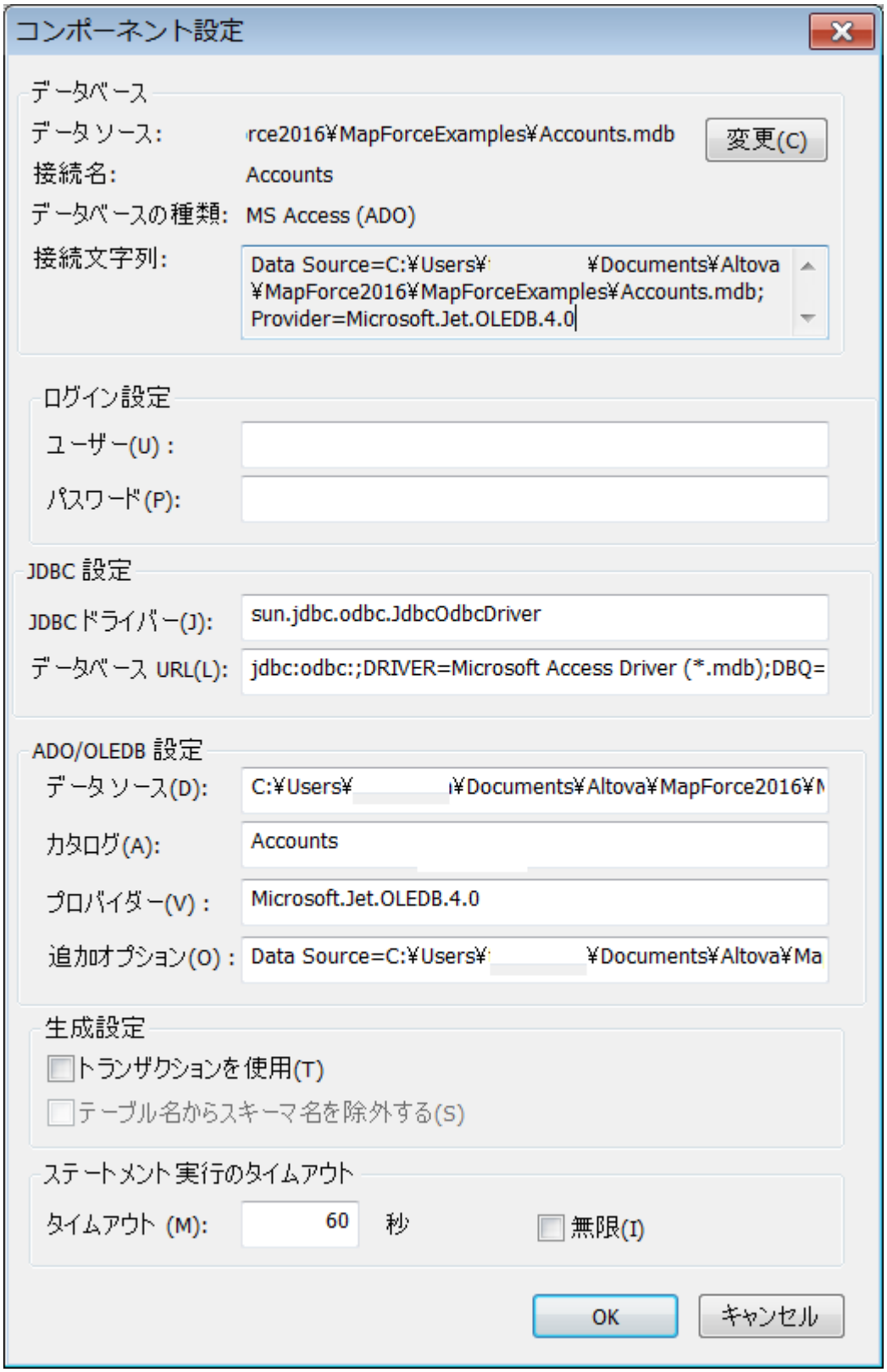
必要に応じて、自動補完の提案を手動で呼び出す:

- **Ctrl+Space** を押します。

6.2.2.12 データベースコンポーネントの設定

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

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



データベースコンポーネント設定ダイアログボックス

(C) 2015-2021 Altova GmbH

使用することのできる設定は、以下のとおりです。

データベース

このグループは、データベース接続情報を表示します。「**変更**」をクリックして異なるデータベースを選択し、既存のデータベースコンポーネント内のデータベースオブジェクトを際定義します。同じ名前のテーブルへの接続は保持されます。データベースコンポーネントを右クリックして、「**追加/削除/データベースオブジェクトの編集**」を選択することにより、コンポーネント内のテーブルを変更することができます。

データソース	現在のデータベースの名前を指定します。ファイルベースのデータベースでは、これはファイルシステム上のパスです。 この設定を使用して相対的または絶対的なパスを使用してファイルベースデータベースがマッピングに追加されるかを決定します。相対的なパスの場合、「データソース」にはパスが含まれています。絶対的なパスにはデータベースファイル名のみが含まれています。
接続名	接続の名前を指定します。MapForce によりこの名前は自動的に生成されます。通常、データソース名と同じ名前ですが、Altova グローバルリソースに接続されている場合、いわゆる「エイリアス」内である場合もあります。マッピング上で同じ接続名を持つ複数のデータベースコンポーネントが存在する場合、名前は"<connection1>"、"<connection2>"などの書式を取ります。
データベースの種類	データベースの型を指定します。
接続文字列	現在のデータベース接続文字列を表示します。この読み取り専用のフィールドは、データベース接続の作成時または変更時に提供される情報を基に生成されます。

ログイン設定

ログイン設定はすべてのコード生成ターゲットとビルドイン実行エンジンのために使用されます。

ユーザー	データベースに接続するためのユーザー名を変更することができます。データベースが接続のためにユーザー名を必要とする場合必須です。
パスワード	データベースに接続するためのパスワードを変更することができます。データベースが接続のためにパスワードを必要とする場合必須です。

JDBC-固有の設定

JDBC 接続が含まれている場合、これらの設定がデータベースに接続するために使用され、生成された Java コード、または MapForce Server により実行されます。

メモ マッピングが Linux または macOS マシン上で実行される場合、ADO、ADO.NET、および ODBC 接続は JDBC（および JDBC 設定が適用されます）に変換されます。次を参照してください：異なる実行環境内でのデータベースマッピング.

JDBC ドライバー	データベースコンポーネントのための現在アクティブなドライバーを表示します。デフォルトのドライバーは、データベースコンポーネントを定義する際に自動的に入力されます。必要に応じてこの項目に入力されるドライバーを変更することができます。データベース URL フィールド内のエントリの構文が選択する特定のドライバーに従っていることを確認してください。
データベース URL	現在選択されている データベースの URL です。このエントリは JDBC ドライバーフィールド内に入力されている特定の JDBC ドライバー構文にしたがっていること

	を確認してください。

ADO/OLEDB-固有の設定

マッピングにADO 接続が含まれる場合、これらの設定がデータベースに接続するために使用されます。生成されたC# またはC++ コード、またはW indows 上で作動するMapForce Server により実行されます。次を参照してください: 多種の実行環境内でのデータベースマッピング。**データソースとカタログ**設定は、ビルトインの実行エンジンでは使用されていません。

データソース	ADO データソースの名前を表示します。
カタログ	ADOカタログの名前を表示します。
プロバイダー	データベースコンポーネントのための現在アクティブなプロバイダーを表示します。
追加オプション	追加のデータベースオプションを表示します。

生成設定

すべてのコード生成ターゲットとビルドイン実行エンジンに生成の設定は適用することができます。

トランザクションの使用	データベースをターゲットとして使用する場合、トランザクション処理 を有効化します。エラーが発生すると、ダイアログボックスが開かれ、続行のオプションを選択できます。トランザクション処理は、このオプションを選択すると、すべてのデータベースコンポーネントのテーブルのために有効化されます。
テーブル名からスキーマ名を削除する	柔軟性のためにテーブル名だけを残し、データベース スキーマ名を生成されたコードから削除することが可能になります。 このオプションにより生成された、MapForce SQL Select ステートメント のみに対して動作することにご注意してください。ユーザー定義 SQL-ステートメントは、仮想テーブルを作成する際には、変更されません。

ステートメント実行のためのタイムアウト

データベースがターゲットコンポーネントとして使用される場合、実行タイムアウトがサーバーの使用状況、交通、長期的に実行されているトリガー、および他の要素により発生する場合があります。この設定により、データベースへの接続が閉じられる前のタイムアウトの時間を定義することができます。設定は、データベースデータをクエリする際、C#、Java、およびC++ コード内で実行されます。

タイムアウト	実行エンジンが、データベースステートメントの実行を中断する前にデータベースのレスポンスを待つ期間を秒数で定義します。実行タイムアウトのデフォルト設定は60秒です。
タイムアウトなし	有効化されると、このオプションは実行エンジンにタイムアウトしないように命令します。

メモ ステートメントの実行のためのタイムアウトはSQLite データベースに使用することができません。

データベーストランザクションの扱い

トランザクションの使用	データベースターゲットとしてデータベースを使用する場合トランザクションの処理を有効化します。トランザクションの処理はこのオプションを選択するとデータベースコンポーネントのすべてのテーブルのために有効化されます。詳細に関してはデータベース例外の扱いを参照してください。

トレース

トレースレベル	ターゲットデータベースコンポーネントに適用可能です。トレーシングが有効化されていると、データベースに対して行われるアクションがトレースファイル内でログされます。詳細に関してはデータベーストレースとエラーログを参照してください。
トレースファイル	データベーストレース情報が書き込まれるファイルを指定します。トレースファイルは XML フォーマットです。トレースファイル構成を参照してください。このパスは絶対的または相対的で、「MFD に対して相対的なすべてのファイルを保存する」ファイルチェックボックスにより影響されます。

MFD ファイルに相対的にすべてのファイルパスを保存する方法

オプションが有効化されている場合、MapForce はコンポーネント設定ダイアログ上に表示されているファイルパスを MapForce Design (.mfd) ファイルの場所に対して相対的に保存します。異なるオペレーティングシステム上の MapForce Server マッピングを使用して実行する場合、相対的なパスを使用してください。コンポーネント上の相対的なパスの使用を参照してください。

6.2.3 データをデータベースにマップする

このセクションは、MapForce によりサポートされるマッピングソースからデータを転送する例と方法について説明しています(例えば、XML ファイル) をターゲットデータベースに転送するなど。以下のテーブルは、使用することのできるオプションのまとめをリストしています。

希望するアクション	このトピックを参照する
マッピングにより提供されたデータを基にして、ターゲットデータベーステーブルにデータを挿入します。	• データをテーブルに挿入する
主キーの値の作成方法を管理します。	• データをテーブルに挿入する • データを複数にリンクされているテーブルに挿入する
「準備中」SQL ステートメントをテーブルがマッピングにより変更される前に実行します(例えば、テーブル、またはカスタム SQL ステートメント内の全てのレコードを削除するなど。)。	• データを複数にリンクされているテーブルに挿入する • データベーステーブルアクション設定
外部キーによりリンクされているテーブル内の階層的なリレーションシップを保持します。	• データを複数にリンクされているテーブルに挿入する
条件付きでテーブルを更新します。	• テーブルの更新
条件に従い、レコードをデータベースにマージします(レコードを更新して、他のレコードを同じテーブルに挿入します)。	• 「以下の条件で更新し、残りを挿入」アクション • MERGE ステートメント

希望するアクション	このトピックを参照する
外部キーリレーションシップを使用して他のテーブルにリンクされているテーブルを更新する際に、データベースの整合性を保持します。	• 親テーブルを更新する際の子テーブルのためのオプション
同じテーブルに対して複数のアクションを定義します（例えば、条件が満たされるとレコードは削除され、それ以外の場合、新規のレコードが挿入されます）...	• 「以下の条件で削除...」アクション • 「以下の条件で無視...」アクション
マッピングの実行に失敗した場合、データの整合性を保持します。	• トランザクションロールバックの使用
データベーステーブルに複数のレコードをバルク挿入します（複数の INSERT ステートメントを1つのクエリにまとめます）。	• バルク転送（MapForce Server）

6.2.3.1 データをテーブルに挿入する

マッピングは、他のデータベースを含む MapForce によりサポートされるソースコンポーネントからデータベーステーブルにデータを挿入することができます。新規に挿入されたレコードの主キーがどのように作成されるかを柔軟的に構成することができます。例えば、主キーは、データベースにより生成されたマッピングから取得、またはデータベーステーブル内の既存のキーの値に基づき計算することができます。

この例は、新規のレコードを XML ファイルから既存の データベーステーブルに挿入する方法を説明しています。主キーの生成の構成方法についても説明されています。この例は以下のファイルを使用します:

- **altova-cmpy.xml** – データベースに挿入されるソースデータを含んでいます。
- **Altova_Hierarchical.xsd** – 上記のインスタンスファイルを検証するために使用されるスキーマです。
- **altova.mdb** – 更新されるターゲットデータベースです。

全てのファイルは＜マイドキュメント＞¥Altova¥MapForce2021¥MapForceExamples¥フォルダー内にあります。下の説明では、簡素化のために、これらのファイルへの完全なパスは省略されています。

> この例のマッピングは、サンプルデータベースファイルを変更します。元のデータベースをバックアップし、下記のステップを開始する前に、新規のコピーを起動することが奨励されます。することが奨励されます。バックアップを作成することにより、元の例が上書きされず、下記と同じ結果を得ることができます。詳細に関しては、次を参照してください: データベースを変更するマッピングの実行。

マッピングの目的は、です。 **altova.mdb** データベースの「Altova」テーブル内に、新規のレコードとして **Altova-cmpy.xml** 内の会社を入力することです。ソース XML ファイルを開くと、「Microtech OrgChart」と呼ばれる会社が1つのみが含まれていることに気がつきます。ですから、マッピングは、という名前「Microtech OrgChart」の新規のレコードを「Altova」テーブルに追加しなければなりません。また、新規の主キーが生成される必要があります。

マッピングの目的を達成するために、以下のステップを行います。

ステップ 1: ソース XML コンポーネントを挿入する

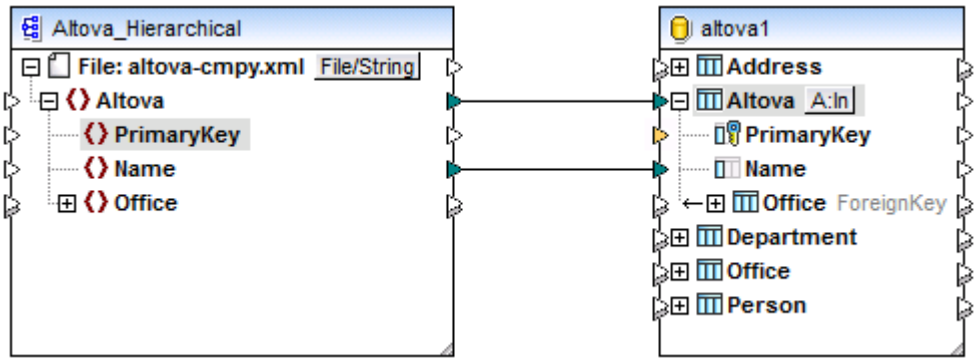
- 「挿入」メニューから、「XML スキーマ/ファイル」をクリックして、**Altova_Hierarchical.xsd** を参照します。インスタンスファイルを与えるように問われると、**Altova-cmpy.xml** を参照してください。

ステップ 2: ターゲットデータベースを挿入する

- 「挿入」メニューから、「データベース」をクリックして、**altova.mdb** に接続するためにウィザードのステップを行ってください。（次を参照してください: 例: "altova.mdb" データベースをマッピングに追加する。

ステップ 3: 接続を描く

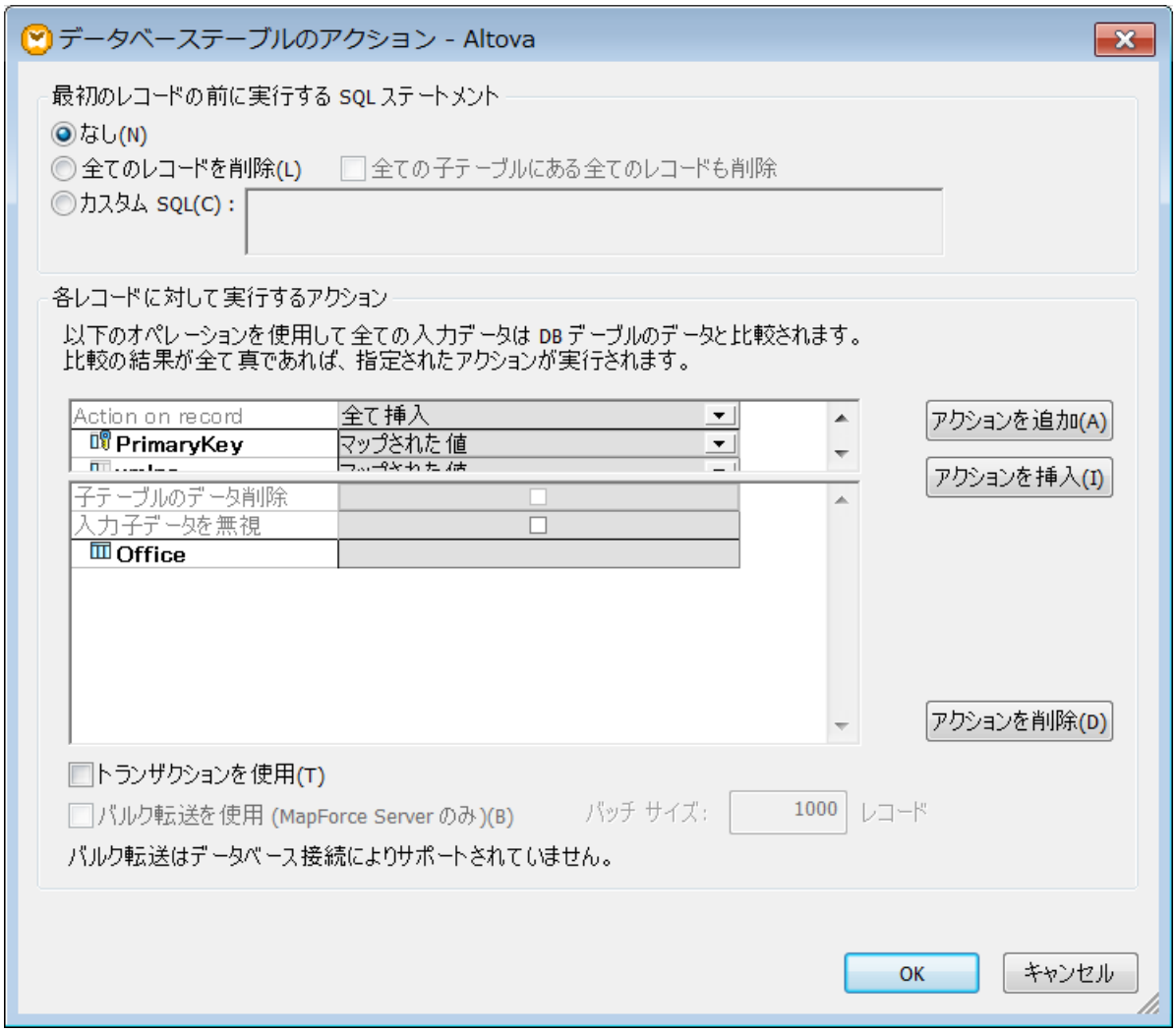
- 下に表示されるとおりマッピングの接続を描きます。



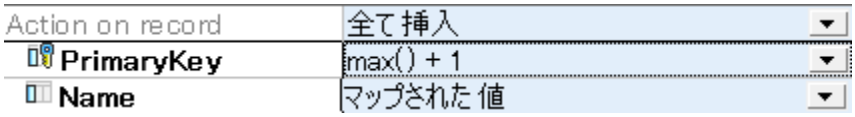
メモ 希望しない接続が自動的に子孫アイテムに作成される場合は、「一致する子を自動的に接続する」が有効化されています。この場合、最後のアクションを元に戻すには、メニューオプション「編集 | 元に戻す」を選択します。自動接続オプションを無効化するには、「接続 | 一致する子を自動的に接続する」を選択します。

ステップ 4: 挿入アクションを構成する

1. ターゲットコンポーネント上に、**アクション: 挿入** (A:In) ボタンが表示されていることにご注意してください。このボタンは、マッピングからの接続がある各テーブルに表示されます (この場合は、「Altova」テーブル)。このボタンをクリックして、実行されるデータベースアクションの詳細を構成してください (この場合は、入力アクション)。データベース テーブル アクション ダイアログボックス が表示されます。



2. データベーステーブルアクションダイアログボックス内で、「全て挿入」から「PrimaryKey」の横の、**max() + 1** オプションを選択します。



このリスト内で使用することのできるオプションは以下を意味します:

オプション	説明
マップされた値	ソースデータをデータベースフィールドに直接マップすることができ、全てのデータベースフィールドのための通常の設定です。リレーションを定義することにより、ストアド プロシージャを使用して、キーの値を与えることもできます。次を参照してください: ストアドプロシージャを使用した主キーの生成 。
max() + 1	データベース内の既存のキーをベースにして、キーの値を生成します。例えば、テーブルに主キー 1、2、および 3 を持つ 3 つのレコードが存在する場合、**max() + 1** は 4 になり

	ます。 この例では、「Altova」テーブルには、主キー 1 を持つレコードは 1 つしか存在せず、**max() + 1** は、期待される新規の主キーの値である 2 です。
DB により生成された	データベースは、キーの値を生成するために「Identity 関数」 を使用します。

「Name」の横のオプション マップされた値はこの列がマッピングから値を直接取得することを示しています。テーブル アクション ダイアログボックスで使用することのできる他のオプションについての説明は、次を参照してください: データベーステーブルアクション設定。

ステップ 5: マッピングをプレビューし、データベースを更新する

マッピングをプレビューするために「出力」タブをクリックします。データベースに対して実行されるアクションを含む、SQL スクリプトが生成されます。スクリプトは、まだ、データベースを変更していません。これはまだプレビューです。

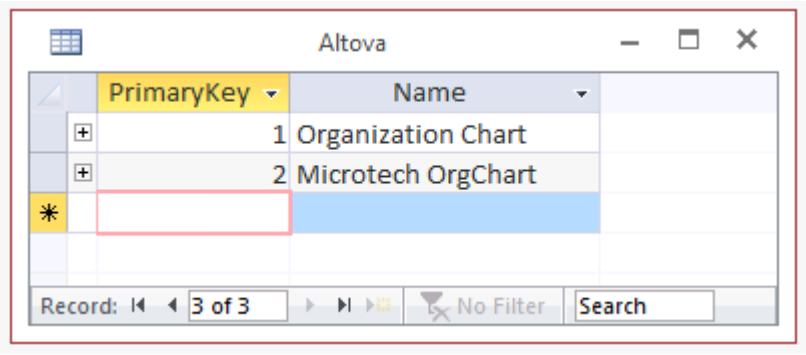
```
1   /*
2   以下の SQL ステートメントは、プレビューのためだけに生成されており、他の SQL クエリー ツールで使用することはできません!
3   これらのステートメントを実行するには、[出力] メニューから [SQL スクリプトの実行] 機能を使用ください。
4   以下の接続時文字列によりデータベースへ接続することができます:
5   Data Source=C:\Users\t.a.yasaka\Documents\Altova\MapForce2019\MapForceExamples/altova.mdb;Password=•••••;Provider=Micr
6   */
7
8   SELECT IIF(MAX([Altova].[PrimaryKey]) IS NULL,0,MAX([Altova].[PrimaryKey]))+1 AS "PrimaryKey" FROM [Altova]
9   -->>> %PrimaryKey1%
10
11  INSERT INTO [Altova] ([Name], [PrimaryKey]) VALUES ('Microtech OrgChart', '%PrimaryKey1%')
12
```

データベースに対してスクリプトを実行するには以下を行います:

- 「出力」メニューから、「SQL-スクリプトの実行」をクリックします。

メモ MapForce から直接 SQL スクリプトを実行することは、データベースを更新する方法のひとつです。次も参照してください: データベースを変更するマッピングの実行。

結果を確認するには、DatabaseSpy または Access 内で **altova.mdb** データベースを開きます。新規の「Microtech OrgChart」レコードが「Altova」テーブルに新規の主キー 2 と共に、追加されていることに注意してください。このレコードためのデータは、入力 XML インスタンスに基づいたものです。



データベース テーブルにデータを挿入するマッピングの作成が完了しました。現在のテーブルと依存する子テーブルの両方にデータを挿入する例に関しては、次を参照してください: データを複数の関連するテーブルに挿入する。

6.2.3.2 データを複数にリンクされているテーブルに挿入する

データベーステーブルは、「親」テーブルである可能性があります。つまり、このテーブルはデータベース内の他のテーブルに外部キーリレーションシップを使用して参照されている可能性があります。このような場合、マッピングに親テーブルだけでなく、依存する子テーブルにもレコードを挿入するように構成することができます。例えば、新規の"company"レコードをデータベーステーブルに挿入する場合、レコードをこの会社に関連したオフィス、オフィスの部署、社員などのためにも挿入することができます。

この例は、データベースリレーションシップを保持しつつ、複数のテーブルにデータを挿入する方法について説明しています。この例は前の例であるデータをテーブルに挿入するを若干高度にしたバージョンです:

- **Altova_Hierarchical.mfd** – 実際のマッピングファイルです。
- **Altova_Hierarchical.xml** – データベースに挿入されるソースデータを含んでいます。
- **Altova_Hierarchical.xsd** – 上記のインスタンスファイルを検証するために使用されるスキーマです。
- **AltovaTarget.mdb** – 更新されるターゲットデータベースです。

全てのファイルは<マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥フォルダー内にあります。下の説明では、簡素化のために、これらのファイルへの完全なパスは省略されています。

> この例のマッピングは、サンプルデータベースファイルを変更します。元のデータベースをバックアップし、下記のステップを開始する前に、新規のコピーを起動することが奨励されます。することが奨励されます。バックアップを作成することにより、元の例が上書きされず、下記と同じ結果を得ることができます。詳細に関しては、次を参照してください:データベースを変更するマッピングの実行。

マッピングの目的は、ソースからのデータXMLファイルを持つターゲットデータベース(**AltovaTarget.mdb**)内のデータを置き換えることです。XMLファイル構成は、データベース内のテーブルの階層構造に大体対応します。これは、以下のように構成されている組織図です: 一番上の要素は、2つのオフィスと含む会社で、各オフィスには部署が存在し、各部署には社員が存在します。同じ階層は、「Altova」テーブルが会社に対応する**AltovaTarget.mdb** 内にも存在します。このテーブルは、外部キーリレーションシップを介して、「Office」テーブル内のレコードにリンクされています。同様に、「Office」は、「Department」にリンクされており、「Department」は、「Person」にリンクされています。**AltovaTarget.mdb** データベースのリレーションシップの図を確認するには、Access のリレーションシップビューでファイルを開きます(次も参照してください: データベースリレーションシップの扱い)。

ステップ 1: ソース XML コンポーネントを挿入する

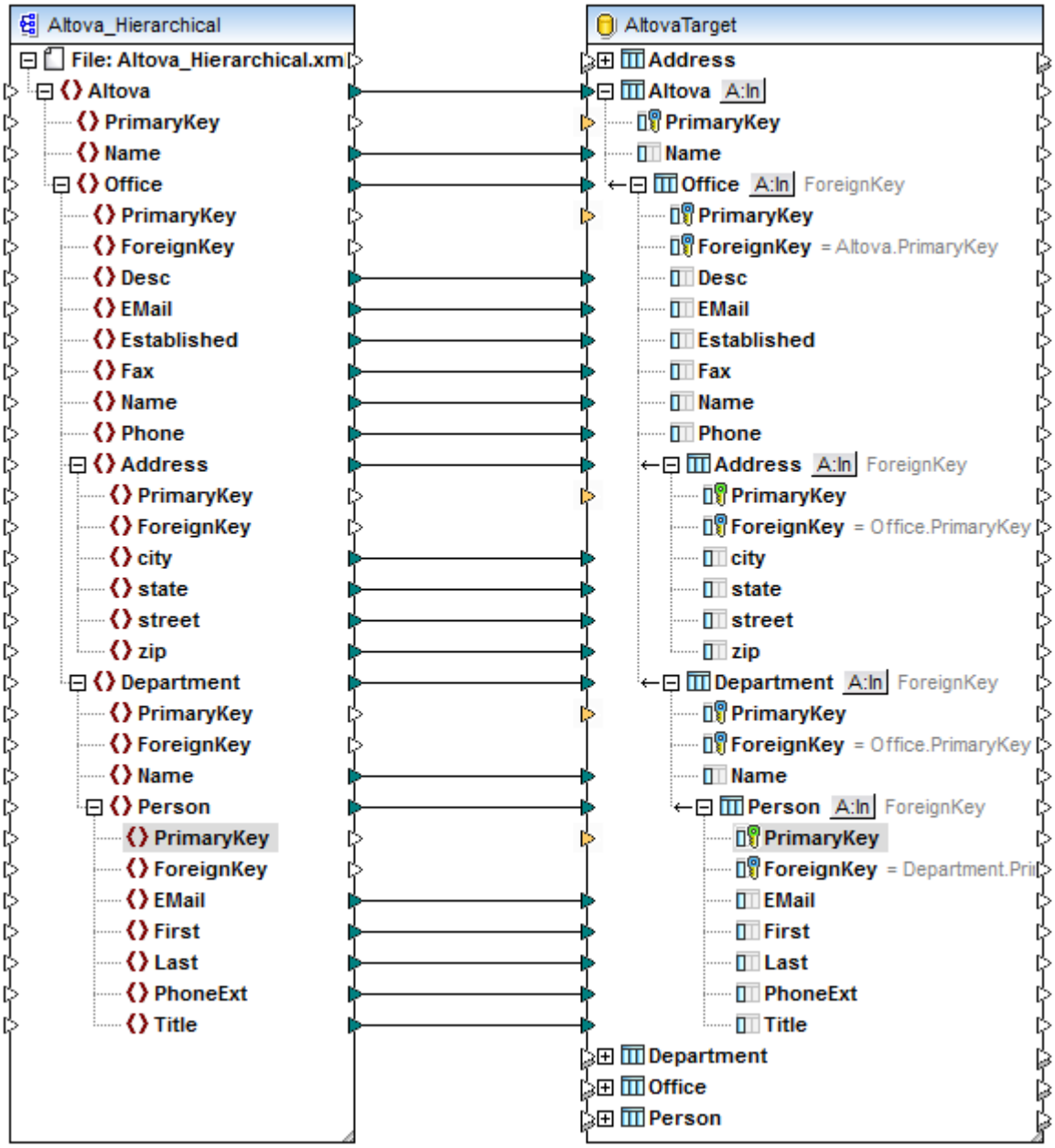
- 「挿入」メニューから、「XML スキーマ/ファイル」をクリックして、**Altova_Hierarchical.xsd** を参照します。インスタンスファイルを与えるように問われると、**Altova_Hierarchical.xml** を参照してください。

ステップ 2: ターゲットデータベースを挿入する

- 「挿入」メニューから、「データベース」をクリックして、**AltovaTarget.mdb** に接続するためにウィザードのステップを行ってください。データベースへの接続方法は、altova.mdb への接続方法と同じです。(次を参照してください: 例: "altova.mdb" データベースをマッピングに追加する)。

ステップ 3: 接続を描く

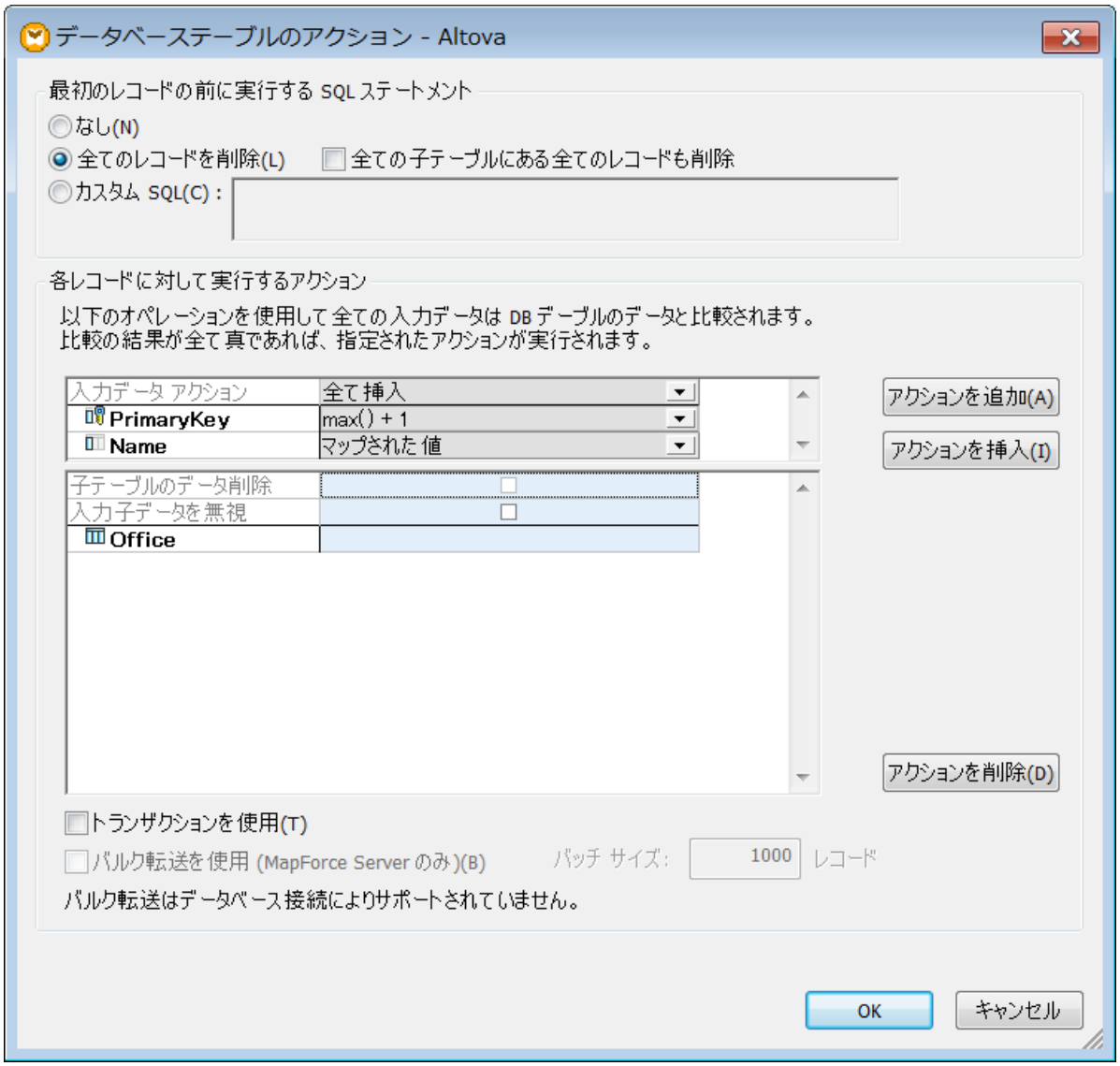
- 下に表示されるとおりマッピングの接続を描きます。主キーと外部キーがマップされていないことに注意してください。下に表示されるように素早く生成されます。



メモ 希望しない接続が自動的に子孫アイテムに作成される場合は、「一致する子を自動的に接続する」が有効化されています。この場合、最後のアクションを元に戻すには、メニューオプション「編集 | 元に戻す」を選択します。自動接続オプションを無効化するには、「接続 | 一致する子を自動的に接続する」を選択します。

ステップ 4: 挿入アクションを構成する

1. ターゲットコンポーネント上で、「Altova」テーブルの横の**アクション: 挿入**（ A:In ）ボタンをクリックし、下に示されるとおり、主キーの**max() + 1** 設定を構成します。この設定に関しては、前の例で更に詳しく説明されています。次を参照してください: データをテーブルに挿入する。



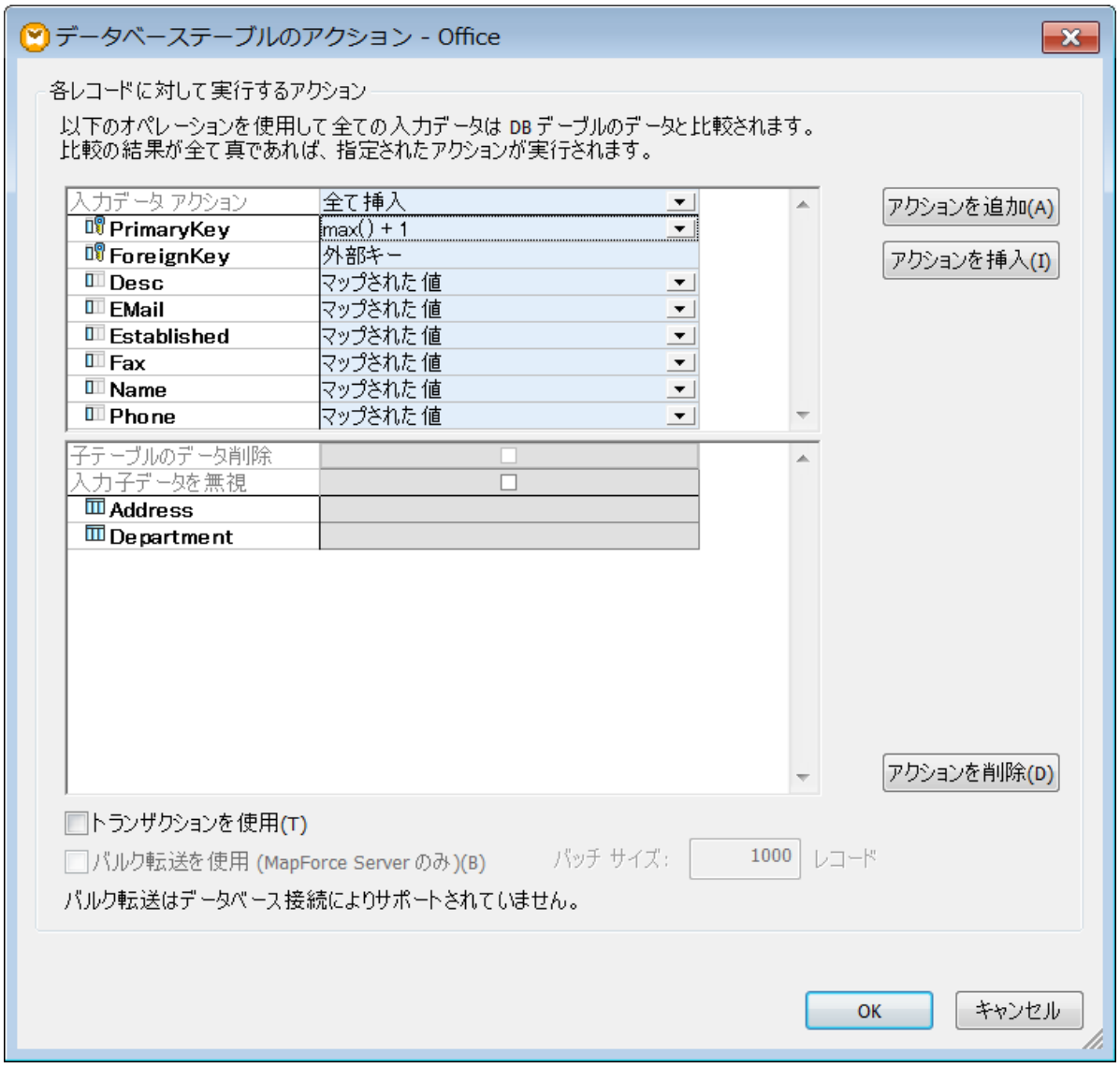
また、「**全てのレコードの削除**」オプションが有効化されていることご注意してください。これは、この例で希望される振る舞いどおり、新規のレコードが挿入される前に、テーブルからの既存のレコード全てをクリアします。このオプションを無効化すると、（新規の主キーを持つ）新規のレコードは、既存のレコードに加え、マッピングを実行する都度、既存のレコードに加え、データベースに追加されます。

この例のスコープのために、「**子テーブルから全てのレコードの削除**」オプションも有効化されています。このオプションにより、「Altova」テーブルからレコードが削除されるだけではなく、外部キーリレーションシップを使用して「Altova」テーブルにリンクされているすべてのテーブル内のレコードも削除します。子テーブルに子が存在する場合、はこれらの子テーブルも削除され、依存関係ツリー内の最後のテーブルまで削除されます。ルート「Altova」テーブルからレコードを削除することは、データベースの整合性の違反であり、マッピングの実行は、失敗します。

データベース テーブル アクション ダイアログボックスで使用することのできる他のオプションに関しての詳細は、次を参照してください：[データベーステーブルアクション設定](#)。

2. ダイアログボックスを閉じるために「OK」をクリックします。マッピングエリアで、ボタンの外観が DEL, A:In に変更されたことに注意してください。これは、"Delete" ステートメントが「挿入」アクションの前に実行されることが構成されていることを示します。
3. 「Office」テーブルの横の A:In ボタンをクリックし、主キーの **max() + 1** 設定を構成します。

(C) 2015-2021 Altova GmbH



4. 「Office」から降順し、各テーブルのためにステップ 3 を行います。具体的には、「Address」、「Department」、および「Person」です。これらすべてのテーブルがルート「Altova」テーブルの直接の子孫であることを確認してください。「ルート」テーブルの詳細とその重要性に関しては、次を参照してください: データベースリレーションシップの扱い。

ステップ 5: マッピングをプレビューし、データベースを更新する

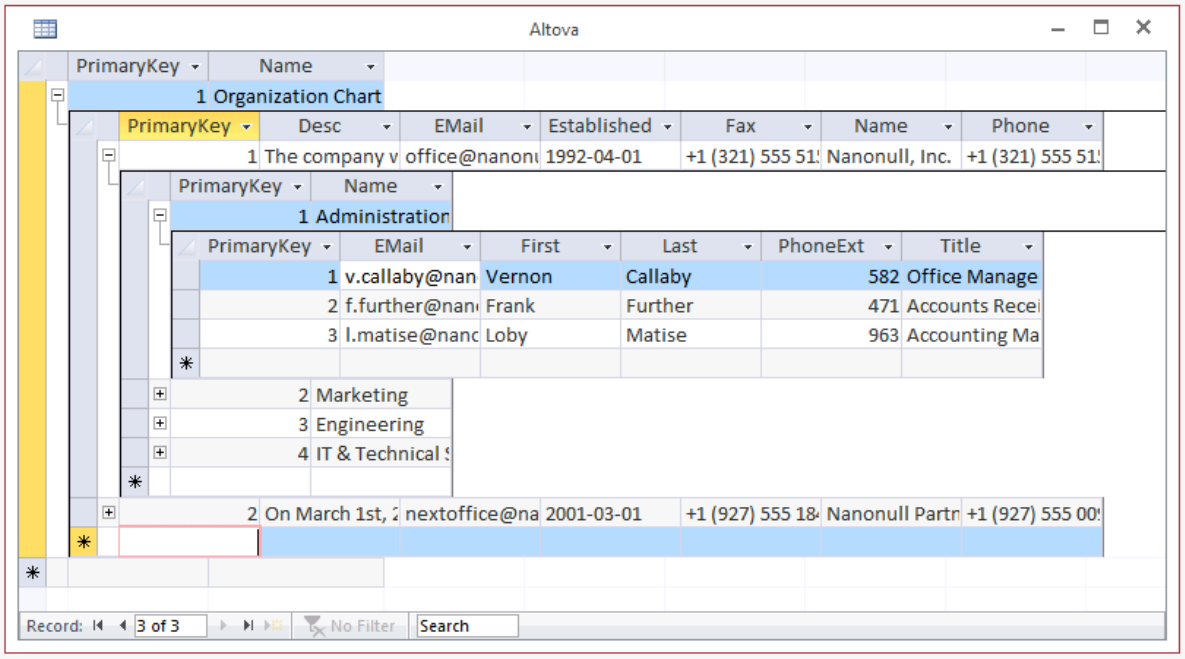
マッピングをプレビューするために「出力」タブをクリックします。データベースに対して実行されるアクションを含む、SQL スクリプトが生成されます。スクリプトは、まだ、データベースを変更していません。これはまだプレビューです。

データベースに対してスクリプトを実行するには以下を行います:

- 「出力」メニューから、「SQL-スクリプトの実行」をクリックします。

メモ MapForce から直接 SQL スクリプトを実行することは、データベースを更新する方法のひとつです。次も参照してください: データベースを変更するマッピングの実行。

結果を確認するには、Microsoft Access で「Altova」テーブルを開き、「Altova」テーブルから「Person」テーブルに、XML ファイルからのリレーションシップがどのようにデータベースに伝達されているかを確認してください。



テーブルの整合性のリレーションシップを保持しつつ、複数の データベース テーブルへデータを挿入するマッピングの作成が完了しました。

6.2.3.3 テーブルの更新

この例は、XML ソースからデータが与えられる既存の データベース テーブルの更新方法について説明しています。この例は以下のファイルを使用します:

- **altova-cmpy.xml** – データベースに挿入されるソースデータを含んでいます。
- **Altova_Hierarchical.xsd** – 上記のインスタンスファイルを検証するために使用されるスキーマです。
- **altova.mdb** – 更新されるターゲット データベースです。

全てのファイルは<マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥フォルダー内にあります。下の説明では、簡素化のために、これらのファイルへの完全なパスは省略されています。

> この例のマッピングは、サンプルデータベースファイルを変更します。元のデータベースをバックアップし、下記のステップを開始する前に、新規のコピーを起動することが奨励されます。することが奨励されます。バックアップを作成することにより、元の例が上書きされず、下記と同じ結果を得ることができます。詳細に関しては、次を参照してください:データベースを変更するマッピングの実行。

マッピングの目的は、XML ドキュメントからの「Person」インスタンスを持つ「Person」テーブル内のレコードを更新することです。XML ファイル内の各 個人は、`PrimaryKey` 子要素を持ちます。「Person」テーブル内の各個人には、`PrimaryKey` 列が存在します。データベース内に`PrimaryKey` を持つ個人とXML ファイル内での個人の`PrimaryKey` が対応するレコードのみが更新される必要があります。

マッピングの目的を達成するために、以下のステップを行います。

ステップ 1: ソース XML コンポーネントを挿入する

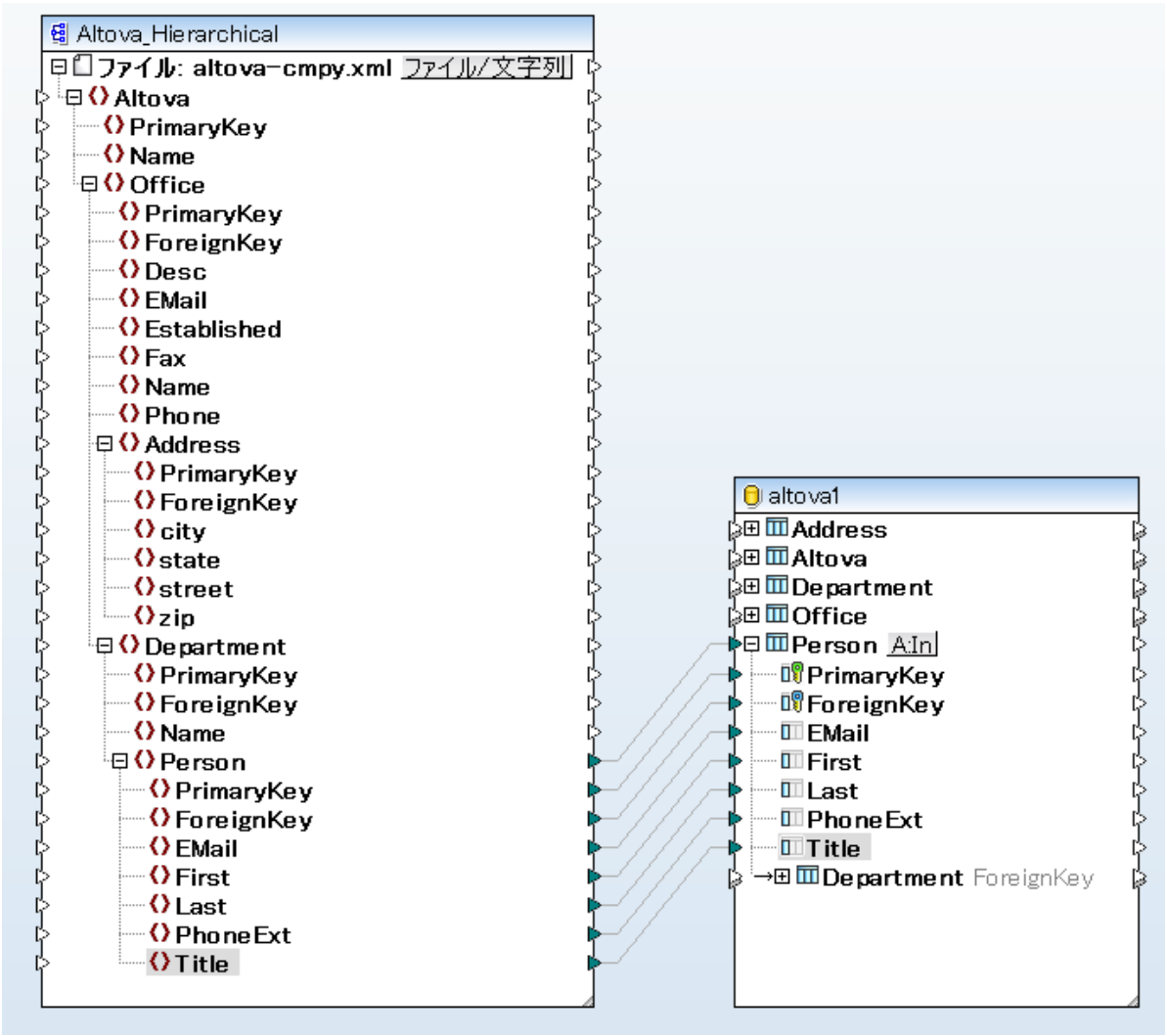
- 「挿入」メニューから、「XML スキーマ/ファイル」をクリックして、**Altova_Hierarchical.xsd** を参照します。インスタンスファイルを与えるように問われると、**altova-cmpy.xml** を参照してください。

ステップ 2: ターゲットデータベースを挿入する

- 「挿入」メニューから、「データベース」をクリックして、 **altova.mdb** に接続するためにウィザードのステップを行ってください。(次を参照してください: 例: "altova.mdb" データベースをマッピングに追加する)。

ステップ 3: 接続を描く

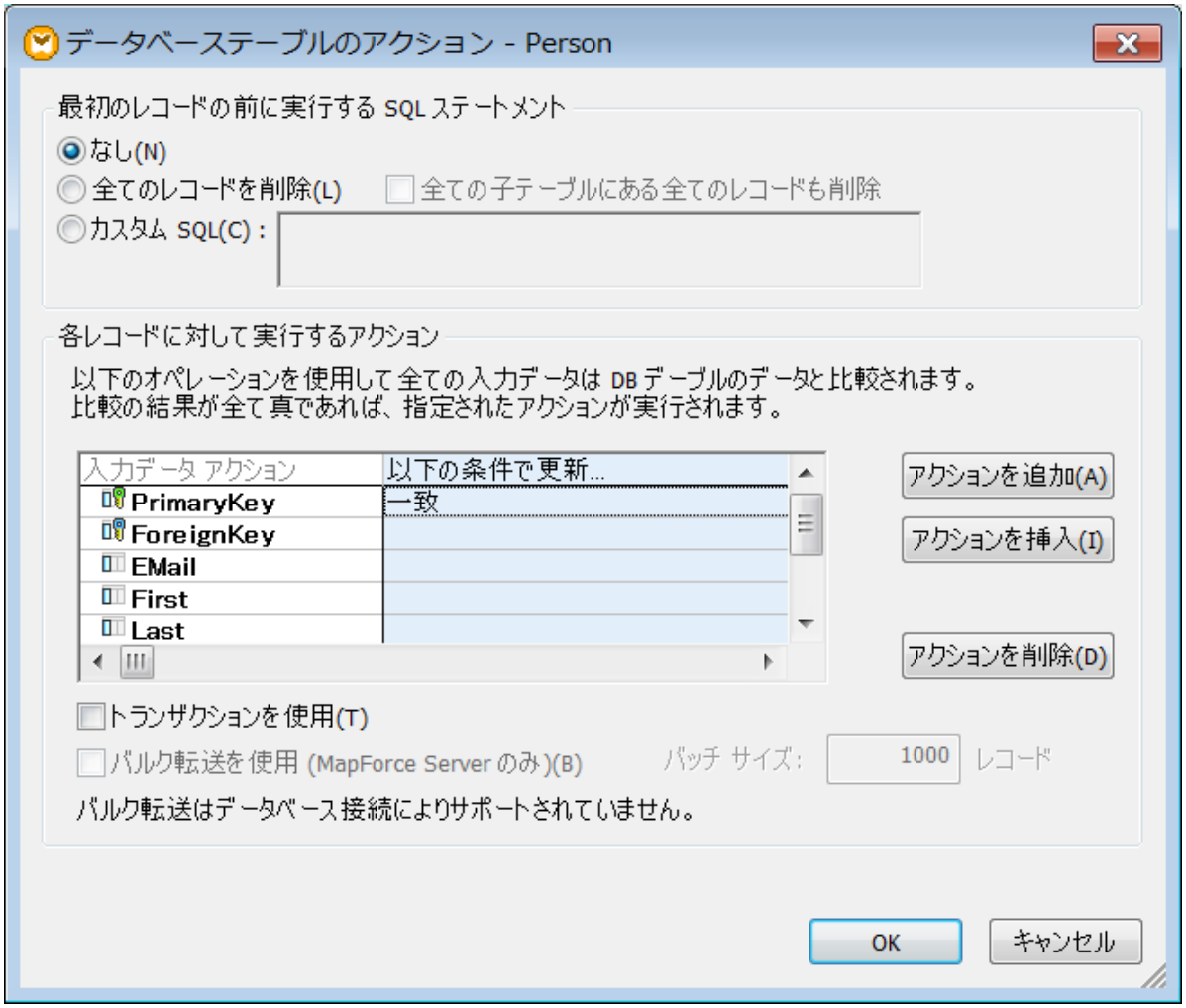
- 下に表示されるとおりマッピングの接続を描きます。



ステップ 4: 更新アクションの構成

1. ターゲットコンポーネントで、「Person」テーブルの横の**アクション: 挿入** (A:In) ボタンをクリックします。

2. 「**レコードに対するアクション**」の横の「**以下の条件で更新する...**」、を選択します。これにより、データベーステーブルアクションを、条件付きの更新アクションに変更します。つまり、条件が満たされる場合のみ、現在のレコードは更新されます（次のステップを参照してください）。
3. 「PrimaryKey」値 の横にあるオプションを、以下に表示されるように「equal」に選択します。これは更新の条件を定義します。つまり、データベースの記録は、「PrimaryKey」値 が、マッピングから来る「PrimaryKey」値に等しい場合のみ、更新されます。



この例では、等値演算子 が「PrimaryKey」フィールドに適用されています。これはデータベースを更新する場合、よくあるシナリオです。型のフィールドに対して定義することのできる定義は、必ずしも主キーではありません。例えば、「最初」と「最後」のフィールドの横のオプションに「equal」を選択することにより、 ソースXML 内で、最初と最後の名前が同じであるレコードのみを更新します。

任意で、**NULL equal** チェックボックスを選択します。このサンプルでは、ソースXML内の**PrimaryKey** フィールド、またはデータベース内の**PrimaryKey** フィールドにnull 値が含まれていないため、このチェックボックスには関連性はありません。しかしながら、データにnull 値が含まれており、このような値を等価として扱うには、場合、このチェックボックスをチェックする必要があります。それ以外の場合は、希望しない結果が生成される場合があります。詳細に関しては、データベーステーブルアクション内のNull 値の処理を参照してください。

4. ダイアログボックスを閉じるために「OK」をクリックします。マッピングに戻り、アクション: 挿入 A:In ボタン がアクション: 更新 (A:Up) ボタンに変更されたことに注目してください。これはこのテーブルのために更新アクションが構成されていることを示しています。

ステップ 5: マッピングをプレビューし、データベースを更新する

マッピングをプレビューするために「出力」タブをクリックします。データベースに対して実行されるアクションを含む、SQL スクリプトが生成されます。スクリプトは、まだ、データベースを変更していません。これはまだプレビューです。

```
1   /*
2   The following SQL statements were executed during "Generate output" function.
3   Every single result is written right to the "-->>>" string.
4   These statements are only for preview and may not be executed in another SQL query tool!
5   The database was connected using the following connection-string:
6   Data Source=\\viepfs06\Documentation\Public\ExampleFiles\EN\MapForce\DB Update
    if\altova.mdb;Provider=Microsoft.Jet.OLEDB.4.0
7   */
8
9   UPDATE [Person] SET [ForeignKey] = 1, [EMail] = 'A.Aldrich@microtech.com', [First] = 'Albert', [Last] = 'Aldrich',
    [PhoneExt] = 582, [Title] = 'Manager' WHERE ([Person].[PrimaryKey] = 1)
10  -->>> OK. 1 row(s).
11
12  UPDATE [Person] SET [ForeignKey] = 1, [EMail] = 'b.bander@microtech.com', [First] = 'Bert', [Last] = 'Bander',
    [PhoneExt] = 471, [Title] = 'Accounts Receivable' WHERE ([Person].[PrimaryKey] = 2)
13  -->>> OK. 1 row(s).
14
15  UPDATE [Person] SET [ForeignKey] = 1, [EMail] = 'c.clovis@microtech.com', [First] = 'Clive', [Last] = 'Clovis',
    [PhoneExt] = 963, [Title] = 'Accounting Manager' WHERE ([Person].[PrimaryKey] = 3)
16  -->>> OK. 1 row(s).
17
18  UPDATE [Person] SET [ForeignKey] = 2, [EMail] = 'd.Durnell@microtech.com', [First] = 'Dave', [Last] = 'Durnell',
    [PhoneExt] = 621, [Title] = 'Marketing Manager Europe' WHERE ([Person].[PrimaryKey] = 4)
19  -->>> OK. 1 row(s).
```

データベースに対してスクリプトを実行するには以下を行います:

- 「出力」メニューから、「SQL-スクリプトの実行」をクリックします。

メモ MapForce から直接 SQL スクリプトを実行することは、データベースを更新する方法のひとつです。次も参照してください: データベースを変更するマッピングの実行。

6.2.3.4 「以下の条件で更新し、残りを挿入」アクション

既存のレコードを更新するだけでなく、同じデータベーステーブルに新規のレコードを挿入する必要がある場合があります。このような場合には、MapForce では、「以下の条件で更新し、残りを挿入」アクション使用することができ、これは以下のように動作します:

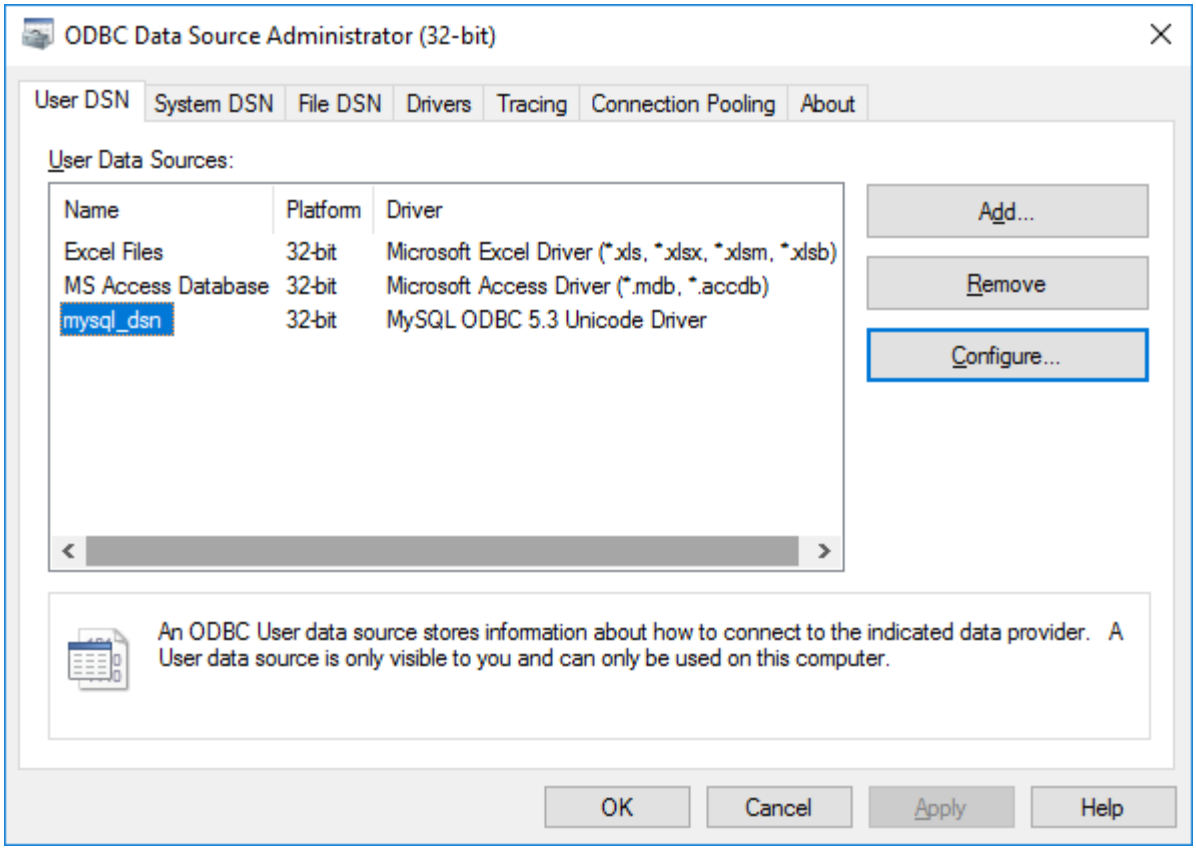
- 「**以下の条件で更新する**」条件が true の場合、既存のデータベースの記録は、マッピングからのデータにより更新されます。
- 「**以下の条件で更新する**」条件が false の場合、また、「**残りを挿入**」条件が存在する場合、新規のレコードが挿入されます。
- ソースファイル内に対応するエントリが存在しないデータベース内にレコードが存在する場合、これらのレコードは変更されません。

MySQL / MariaDB ODBC メモ

ターゲットデータベースが ODBC を使用した MySQL または MariaDB の場合、オプションは「影響された行の代わりに一致した行を返す」が MySQL ODBC コネクタの**カーソル結果** タブで有効化されている必要があります。または、データベース接続ウィザードを使用して接続文字列を手動で 入力する場合 `Option=2` を接続文字列に追加します: `Dsn=mydsn;Option=2;`

MySQL ODBC コネクタからこのオプションを有効化する方法:

1. **Windows** キーを押して、"ODBC"を入力します。
2. (インストールされた MySQL ODBC コネクタにより 32-ビット、または 64-ビットの) ODBC データソース管理者を実行します
3. MapForce マッピングにより使用される データソース名 (DSN) をクリックし、「構成」をクリックします。



4. 使用することのできる高度なオプションを使用するために「 Details >> 」をクリックします。
5. **Cursors/Results** タブをクリックし、チェックボックス「Return matched rows instead of affected rows」を選択します。

サンプル

次の例は、XML ソースからのデータをデータベース テーブルに更新と挿入してマージする方法を説明しています。この例は以下のファイルを使用します:

- **altova-cmpy-extra.xml** – データベースに挿入されるソースデータを含んでいます。
- **Altova_Hierarchical.xsd** – 上記のインスタンスファイルを検証するために使用されるスキーマです。
- **altova.mdb** – 更新されるターゲット データベースです。

全てのファイルはフォルダー <マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥ 内にあります。下の説明では、簡素化のために、これらのファイルへの完全なパスは省略されています。

この例のマッピングは、サンプルデータベースファイルを変更します。元のデータベースをバックアップし、下記のステップを開始する前に、新規のコピーを起動することが奨励されます。することが奨励されます。バックアップを作成することにより、元の例が上書きされず、下記と同じ結果を得ることができます。詳細に関しては、次を参照してください: データベースを変更するマッピングの実行。

マッピングの目的は、ターゲット「Person」テーブルにソースXMLドキュメントからのすべてのレコードをマージすることです。具体的には、ソースXMLデータ内の各レコードのために、マッピングは以下を行う必要があります:

- XMLファイル内の個人の`PrimaryKey`が、データベース内の個人の`PrimaryKey`に対応する場合、レコードは更新されます。
- 上記の条件を満たさないPersonテーブル内の既存のレコードは影響を受けません。
- XMLファイル内の個人の`PrimaryKey`が、ターゲットデータベーステーブルに一致するものが無い場合、新規のレコードをデータベーステーブルに追加してください。

マッピングの目的を達成するために、以下のステップを行います。

ステップ1: ソースXMLコンポーネントを挿入する

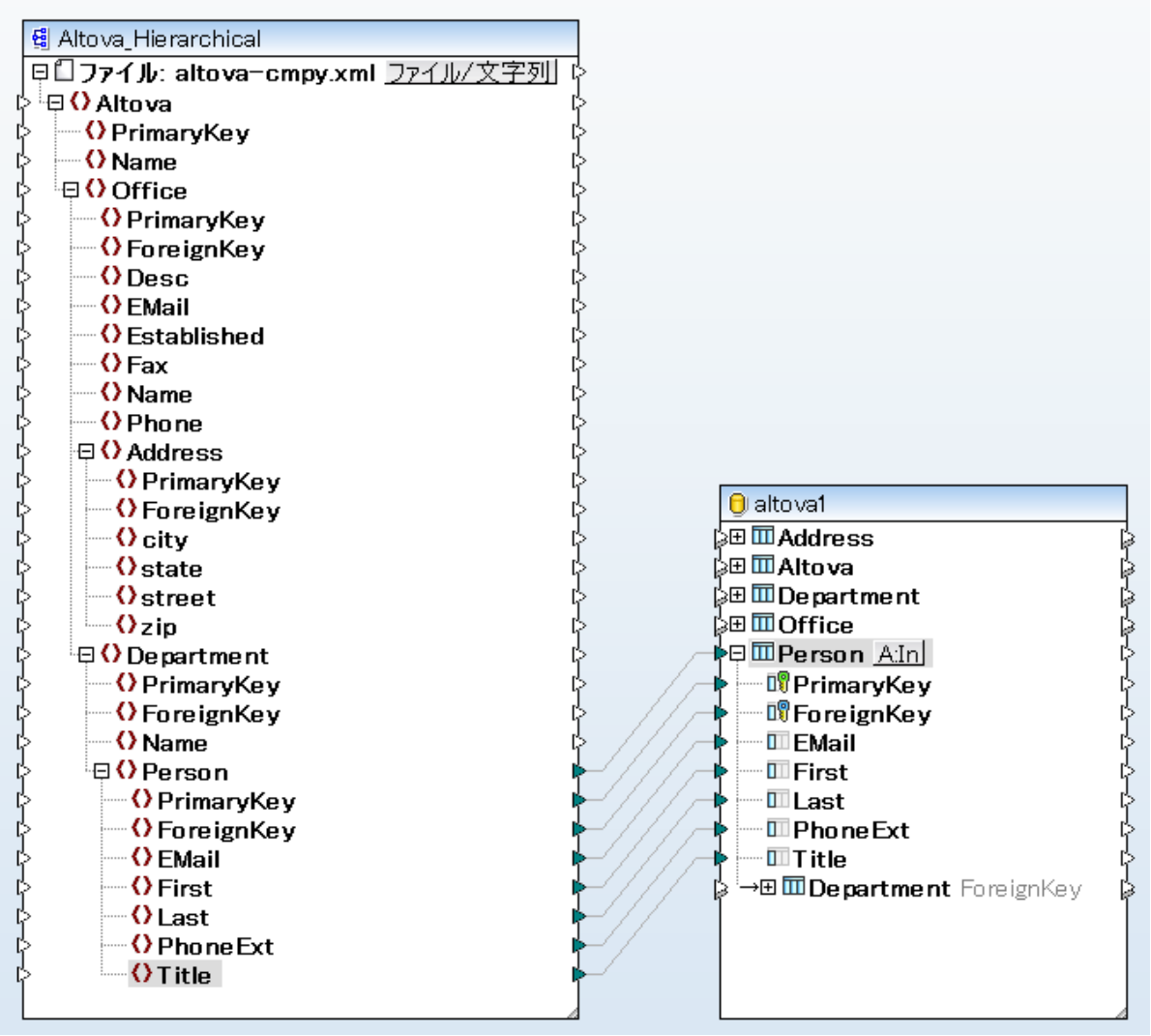
- 「挿入」メニューから、「XMLスキーマ/ファイル」をクリックして、**Altova_Hierarchical.xsd**を参照します。インスタンスファイルを与えるように問われると、**altova-cmpy-extra.xml**を参照してください。

ステップ2: ターゲットデータベースを挿入する

- 「挿入」メニューから、「データベース」をクリックして、**altova.mdb**に接続するためにウィザードのステップを行ってください。(次を参照してください: 例: "altova.mdb" データベースをマッピングに追加する)。

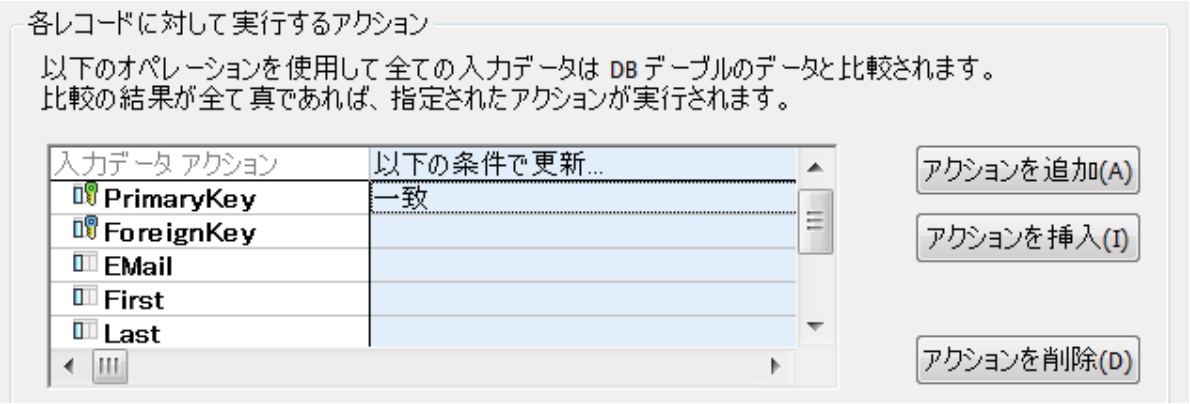
ステップ3: 接続を描く

- 下に表示されるとおりマッピングの接続を描きます。

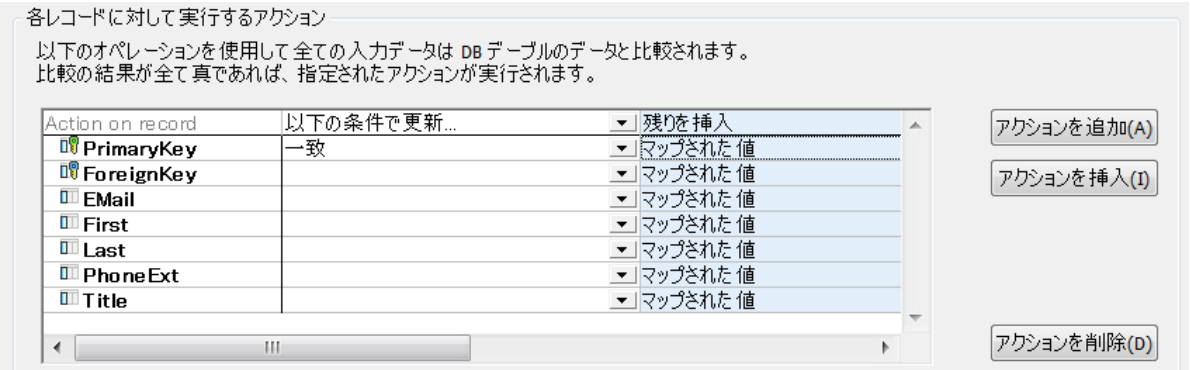


ステップ 4: 「以下の条件で更新し、残りを挿入」アクションを構成する

1. ターゲットコンポーネントで、「Person」テーブルの横の**アクション: 挿入** (A:In) ボタンをクリックします。
2. 「**レコードに対するアクション**」の横の、「**以下の条件で更新する..**」を選択します。これはデータベーステーブルアクションを条件付きの更新アクションに変更します。つまり、条件が満たされる場合のみ、現在のレコードは更新されます (次のステップを参照してください)。
3. 「PrimaryKey」値 の横にあるオプションを、以下に表示されるように「equal」に選択します。つまり、データベースの記録は、「PrimaryKey」 値が、マッピングから与えられるの「PrimaryKey」 値に等しい場合のみ更新されます。



4. 任意で、**NULL equal** チェックボックスを選択します。このサンプルでは、ソースXML内の**PrimaryKey** フィールド、またはデータベース内の**PrimaryKey** フィールドにnull 値が含まれていないため、このチェックボックスには関連性はありません。しかしながら、データにnull 値が含まれており、このような値を等価として扱うには、場合、このチェックボックスをチェックする必要があります。それ以外の場合は、希望しない結果が生成される場合があります。詳細に関しては、データベーステーブルアクション内のNull 値の処理を参照してください。
5. 「**アクションの追加**」をクリックします。これは、新規のアクションを既存の「**以下の条件で更新する**」アクションの右に追加します。新規のアクションを「**残りを挿入**」として構成します:



上のイメージでは、データベーステーブルアクションは、マッピングの目的に従い構成されています。つまり、「**以下の条件で更新する**」... 条件が満たされる場合のみレコードは更新されます。それ以外の場合、挿入されます。オプション「マップされた値」は、マッピングからの値がレコードの全てのフィールドを作成するために使用されることを指定します。

(この例では必要ありませんが)2つ以上のアクションを同じデータベーステーブルに対して定義することもできます。マッピングランタイムでは、アクションは左から右へ実行されます。最後の「**挿入**」アクションは最後と考えられ、このアクションの後のアクションは無視されます。

ダイアログボックスの「**アクションの追加**」ボタンは、選択されたアクションの後に新規にアクションを追加します。「**アクションの挿入**」は、新規のアクションを選択されたアクションの**前**に追加します。既存のアクションを削除するには、アクションをクリックし、を「**アクションの削除**」クリックします。

6. ダイアログボックスを閉じるために「OK」をクリックします。マッピングでは、**アクション: 挿入** A:In ボタンが、**アクション: 更新; 挿入** (A:Up,In) これは、更新と挿入アクションが、このテーブルのために構成されることを示します。

ステップ 5: マッピングをプレビューし、データベースを更新する

マッピングをプレビューするために「出力」タブをクリックします。データベースに対して実行されるアクションを含む、SQL スクリプトが生成されます。スクリプトは、まだ、データベースを変更していません。これはまだプレビューです。

```
1   /*
2   The following SQL statements are only for preview and may not be executed in another SQL query tool!
3   To execute these statements use function "Run SQL-script" from menu "Output".
4   Connect to database using the following connection-string:
5   Data Source=\\viepfs06\Documentation\Public\ExampleFiles\EN\MapForce\DB Update if... Insert
    Rest\altova.mdb;Provider=Microsoft.Jet.OLEDB.4.0
6   */
7
8   UPDATE [Person] SET [ForeignKey] = 1, [EMail] = 'A.Aldrich@microtech.com', [First] = 'Albert', [Last] = 'Aldrich',
    [PhoneExt] = 582, [Title] = 'Manager' WHERE ([Person].[PrimaryKey] = 1)
9
10  UPDATE [Person] SET [ForeignKey] = 1, [EMail] = 'b.bander@microtech.com', [First] = 'Bert', [Last] = 'Bander',
    [PhoneExt] = 471, [Title] = 'Accounts Receivable' WHERE ([Person].[PrimaryKey] = 2)
11
12  UPDATE [Person] SET [ForeignKey] = 1, [EMail] = 'c.clovis@microtech.com', [First] = 'Clive', [Last] = 'Clovis',
    [PhoneExt] = 963, [Title] = 'Accounting Manager' WHERE ([Person].[PrimaryKey] = 3)
13
14  UPDATE [Person] SET [ForeignKey] = 1, [EMail] = 'c.Cicada@microtech.com', [First] = 'Camilla ', [Last] = 'Cicada',
    [PhoneExt] = 765, [Title] = 'HR' WHERE ([Person].[PrimaryKey] = 30)
```

データベース更新前の SQL スクリプト（部分的なビュー）

INSERT ステートメントがプレビューのスクリプト内で表示されていないことに気がついているかもしれません。これは、レコードは条件付きで挿入され、INSERT ステートメントは（マッピングが実行される前は未知である）「以下の条件で更新する」アクションの結果に依存するため、通常の振る舞いです。

メモ　　特定のデータベース型のために、MapForce は、MERGE ステートメントを UPDATE ステートメントの代わりに作成します。詳細に関しては、次を参照してください：MERGE ステートメント.

データベースに対してスクリプトを実行するには以下を行います:

- 「出力」メニューから、「SQL-スクリプトの実行」をクリックします。

マッピングが実行され、スクリプトがデータベースに適用されました。「出力」タブ内で INSERT ステートメントが表示されていることに注意してください。

(C) 2015-2021 Altova GmbH

```
1   /*
2   The following SQL statements were executed during "Generate output" function.
3   Every single result is written right to the "-->>>" string.
4   These statements are only for preview and may not be executed in another SQL query tool!
5   The database was connected using the following connection-string:
6   Data Source=\\viepfs06\Documentation\Public\ExampleFiles\EN\MapForce\DB Update if... Insert
    Rest\altova.mdb;Provider=Microsoft.Jet.OLEDB.4.0
7   */
8
9   UPDATE [Person] SET [ForeignKey] = 1, [EMail] = 'A.Aldrich@microtech.com', [First] = 'Albert', [Last] = 'Aldrich',
    [PhoneExt] = 582, [Title] = 'Manager' WHERE ([Person].[PrimaryKey] = 1)
10  -->>> OK. 1 row(s).
11
12  UPDATE [Person] SET [ForeignKey] = 1, [EMail] = 'b.bander@microtech.com', [First] = 'Bert', [Last] = 'Bander', [PhoneExt]
     = 471, [Title] = 'Accounts Receivable' WHERE ([Person].[PrimaryKey] = 2)
13  -->>> OK. 1 row(s).
14
15  UPDATE [Person] SET [ForeignKey] = 1, [EMail] = 'c.clovis@microtech.com', [First] = 'Clive', [Last] = 'Clovis', [PhoneExt] =
     963, [Title] = 'Accounting Manager' WHERE ([Person].[PrimaryKey] = 3)
16  -->>> OK. 1 row(s).
17
18  UPDATE [Person] SET [ForeignKey] = 1, [EMail] = 'c.Cicada@microtech.com', [First] = 'Camilla ', [Last] = 'Cicada',
    [PhoneExt] = 765, [Title] = 'HR' WHERE ([Person].[PrimaryKey] = 30)
19  -->>> OK. 0 row(s).
20
21  INSERT INTO [Person] ([PrimaryKey], [ForeignKey], [EMail], [First], [Last], [PhoneExt], [Title]) VALUES (30, 1,
    'c.Cicada@microtech.com', 'Camilla ', 'Cicada', 765, 'HR')
22  -->>> OK. 1 row(s).
```

データベース更新後のSQL スクリプト（部分的なビュー）

メモ MapForce から直接 SQL スクリプトを実行することは、データベースを更新する方法のひとつです。次も参照してください： データベースを変更するマッピングの実行。

MapForce 内のDB クエリタブ内の「Person」テーブルを開くと（次を参照してください： データベースの参照とクエリ）、マッピングの結果は以下の通りです:

- 対応する主キーin XML ファイル内で、対応する主キーを持つすべてのデータベースの記録は更新されました。例は、主キー1、2、3、4 、および5 を持つレコードです。
- XML ファイル内に対応するキーをもたない全てのデータベースの記録は、影響を受けず、変更されていません。例は、主キー6、7、8、および9 を持つレコードです。
- 新規のレコードが（データベース内でキーが存在していない箇所である）「Person」テーブルに挿入されました。例は、主キー30 と31 を持つレコードです。

	PrimaryKey	ForeignKey	EMail	First	Last	PhoneExt	Title
1	1	1	A.Aldrich@microtech.com	Albert	Aldrich	582	Manager
2	2	1	b.bander@microtech.com	Bert	Bander	471	Accounts Receivable
3	3	1	c.clovis@microtech.com	Clive	Clovis	963	Accounting Manager
4	4	2	d.Durnell@microtech.com	Dave	Durnelll	621	Marketing Manager Europe
5	5	2	e.ellas@microtech.com	Eve	Ellas	753	Art Director
6	6	3	f.landis@nanonull.com	Fred	Landis	951	Program Manager
7	7	3	m.landis@nanonull.com	Michelle	Butler	654	Software Engineer
8	8	3	t.little@nanonull.com	Ted	Little	852	Software Engineer
9	9	3	a.way@nanonull.com	Ann	Way	951	Technical Writer
10	10	3	l.gardner@nanonull.com	Liz	Gardner	753	Software Engineer
11	11	3	p.smith@nanonull.com	Paul	Smith	334	Software Engineer
12	12	4	a.martin@nanonull.com	Alex	Martin	778	IT Manager
13	13	4	g.hammer@nanonull.com	George	Hammer	223	Web Developer
14	14	4	n.newbury@microtech.com	Nira	Newbury	241	Support Engineer
15	15	4	o.origone@microtech.com	Olanda	Origone	345	Support Engineer
16	16	5	s.meier@microtech.com	Paul	Ponetti	114	Office Manager
17	17	5	t.bone@microtech.com	Quang	Qui	331	Accounts Receivable
18	18	6	m.nafta@microtech.com	Rita	Rotter	122	PR & Marketing Manager US
19	19	7	v.bass@microtech.com	Silla	Semua	716	IT Manager
20	20	7	c.franken@microtech.com	Tom	Tinker	147	Support Engineer
21	21	7	m.redgreen@microtech.com	Uri	Ugoner	152	Support Engineer
22	30	1	c.Cicada@microtech.com	Camilla	Cicada	765	HR
23	31	1	c.corrigan@microtech.com	Carol	Corrigan	629	Admin

データベースを更新後の「Person」テーブル

6.2.3.5 MERGE ステートメント

データベーステーブルを更新し、データを挿入する特定のマッピングに関しては、MapForce は、データベースに対して実行される MERGE ステートメントをマッピングランタイムも生成する可能性があります。(次も参照してください: 「以下の条件で更新し、残りを挿入」アクション)。実行エンジンが MapForce である必要は必ずしもありません。次を参照してください: データベースを変更するマッピングの実行。

MERGE ステートメント は、次のデータベース型によりサポートされています:

- SQL Server 2008 および以降
- Oracle
- DB2
- Firebird

INSERT と UPDATE ステートメントを1つにまとめるため、MERGE ステートメントは、データベースサーバーの呼び出しの回数を減らします。また、MERGE ステートメントの場合、整合性は、データベースによりチェックされます。データベース型を検出すると、MapForce は、自動的に MERGE ステートメントを作成します。MapForce による MERGE ステートメントの生成を手動で影響することはできません。

(C) 2015-2021 Altova GmbH

マッピングがランタイムにデータベースMERGE ステートメントを実行するかを確認します（INSERT とUPDATE ステートメントの組み合わせの適用に対して）:

1. 「以下の条件で更新する...」と「残りを挿入」アクションを使用するマッピングを作成します。例えば、次を参照してください：「以下の条件で更新し、残りを挿入」アクション。
2. マッピングをプレビューするには、「出力」タブをクリックします。

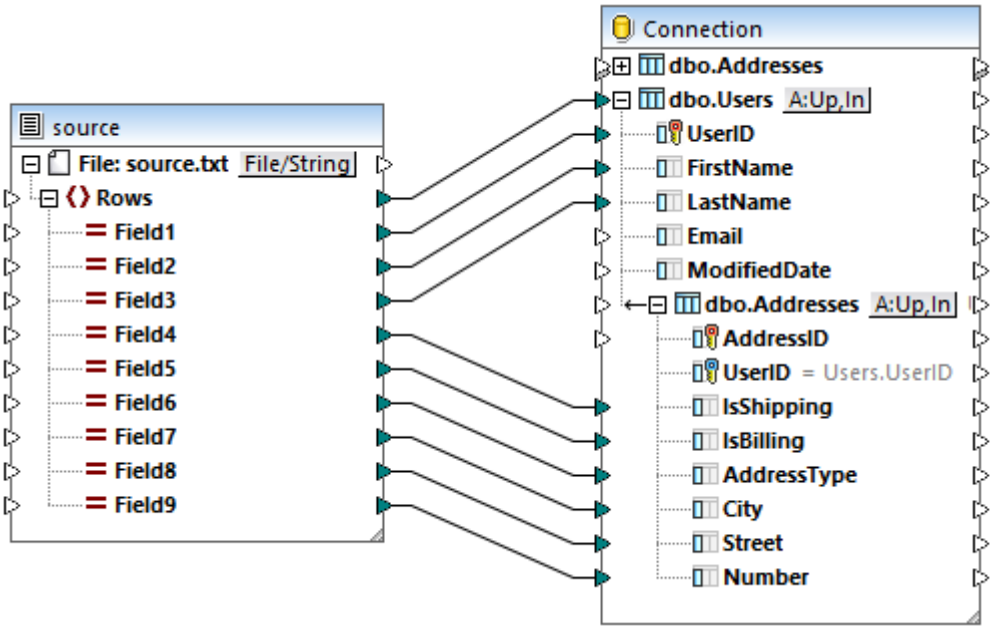
MERGE がデータベース型によりサポートされている場合、生成されたSQL スクリプトは、MERGE ステートメントを含みます。例:

```
1   /*
2   The following SQL statements are only for preview and may not be executed in another SQL query tool!
3   To execute these statements use function "Run SQL-script" from menu "Output".
4   Connect to database using the following connection-string:
5   Data Source=                              ;Initial
    Catalog=DB MERGE;MultipleActiveResultSets=True;Password=•••••;Persist Security Info=True;User
    ID=
6   */
7
8   SET QUOTED_IDENTIFIER ON
9
10  UPDATE [dbo].[Users] SET [FirstName] = (CAST('Despine' AS nvarchar(50))), [LastName] =
    (CAST('Buttler' AS nvarchar(50))) WHERE ([dbo].[Users].[UserID] = 1)
11
12  SELECT [UserID] FROM [dbo].[Users] WHERE ([UserID] = 1)
13  -->>> %UserID1%
14
15  DELETE FROM [dbo].[Addresses] WHERE EXISTS(SELECT * FROM [dbo].[Users] WHERE [dbo].[Users].[UserID] =
    [dbo].[Addresses].[UserID] AND ([dbo].[Users].[UserID] = '%UserID1%'))
16
17  MERGE INTO [dbo].[Addresses] AS T USING ( VALUES ( '%UserID1%', 1, 1, (CAST('Home' AS nvarchar(20))),
    (CAST('Louisville' AS nvarchar(50))), (CAST('Elm Street' AS nvarchar(50))), (CAST('12' AS
    nvarchar(20))) ) ) AS S ( [UserID], [IsShipping], [IsBilling], [AddressType], [City], [Street],
    [Number] ) ON ( (S.[UserID] = T.[UserID]) ) WHEN MATCHED THEN UPDATE SET [IsShipping] = S.
    [IsShipping], [IsBilling] = S.[IsBilling], [AddressType] = S.[AddressType], [City] = S.[City],
    [Street] = S.[Street], [Number] = S.[Number] WHEN NOT MATCHED THEN INSERT ( [UserID], [IsShipping],
    [IsBilling], [AddressType], [City], [Street], [Number] ) VALUES ( S.[UserID], S.[IsShipping], S.
    [IsBilling], S.[AddressType], S.[City], S.[Street], S.[Number] );
18

Mapping | DB Query | Output
```

データベース型がMERGE をサポートしない場合、生成されたSQL スクリプトは、UPDATE ステートメントのみを含みます。「以下の条件で更新する...」条件が満たされている場合のみ、実行されるため、（マッピングの実行前には未知である）INSERT ステートメントは、プレビューでは表示されません。

メモ

- MapForce はサポートされるデータベースからを検知すると、MERGE ステートメントを自動的に作成します。MapForce がMERGE ステートメントを作成するか否かを手動で影響することはできません。
- 親子リレーションシップを持つ複数のテーブルを更新する場合、マージは「リーフ」テーブルのためにのみ作成されます。「リーフ」テーブルはマップされている最も深い子テーブルです。例えば、下のマッピングでは、「以下の条件で更新する...」と「残りを挿入」アクションがUsers テーブルとAddresses テーブル両方のために定義されています。しかしながら、MERGE ステートメントはリーフテーブルAddresses のためにのみ生成されます。親テーブルUsers は、MERGE の代わりにUPDATE またはINSERT ステートメントを取得します。



- ステートメントに関しては、「バルク転送」オプション（次を参照してください: バルク転送 (MapForce Server)）は、ODBC と JDBC データベース接続のためにのみによりサポートされています。

6.2.3.6 親テーブルを更新する際の子テーブルのためのオプション

マッピングは、「親」テーブルであるテーブルを更新する際、（つまり、テーブルに対して外部キーリレーションシップ持っている場合）、ソースデータとターゲットテーブル内で、どのように依存するレコードが扱われるかを構成することができます。例えば、**altova.mdb** データベース内の「Department」テーブルを更新するとします。すべての個人 は、外部キーを使用して、department に、リンクされています。「Person」テーブルに対しても、（挿入、更新、または削除などの）アクションを取る必要があるかもしれません。これを行うことは、データベースの整合性を保持し、マッピングエラーを回避する助けとなります。

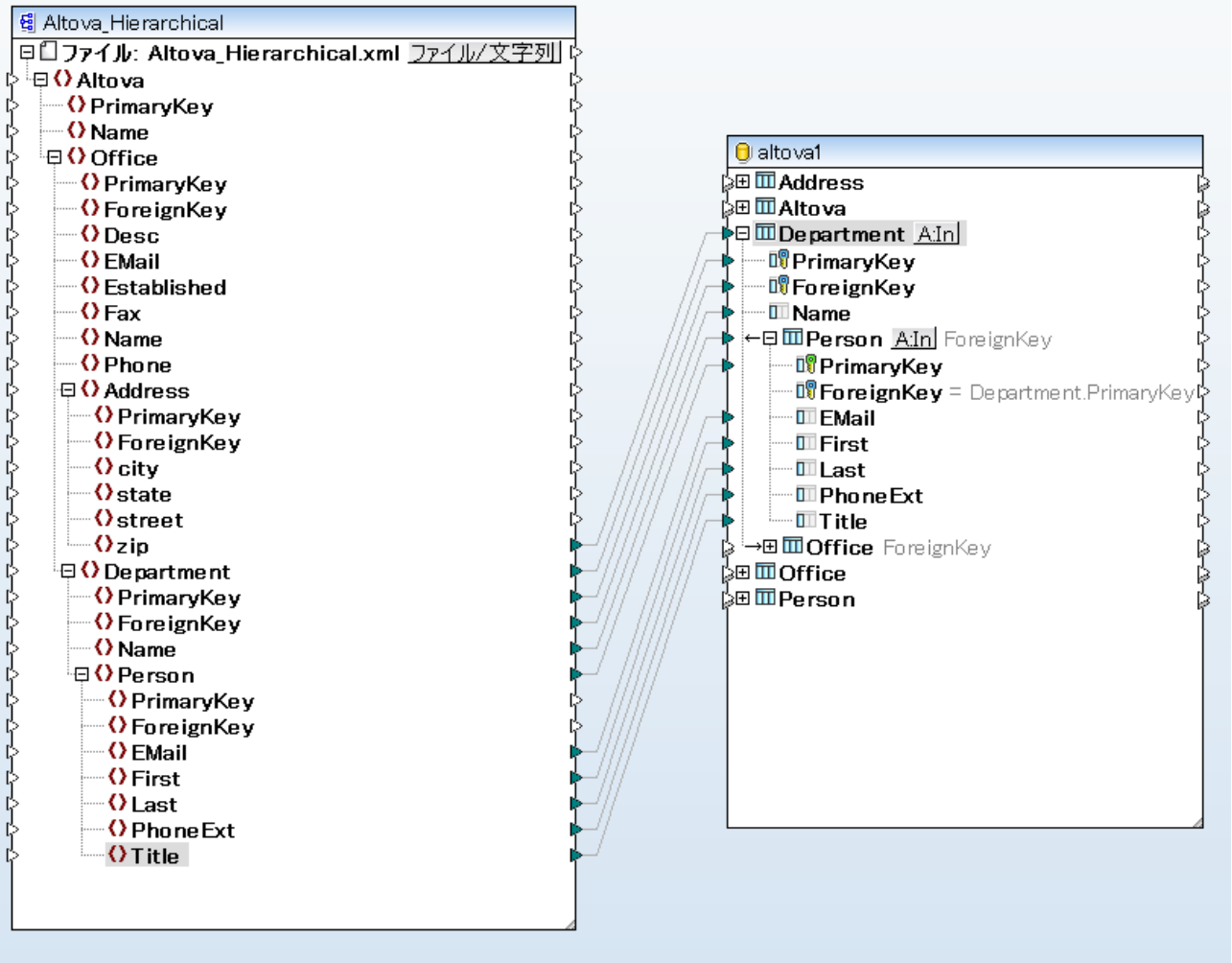
このトピックは、親「Department」テーブルを更新した際に、「Person」テーブルに対して使用することのできるオプションについて説明しています。以下のサンプルファイルが使用されています:

- **altova-cmpy-extra.xml** – データベースに挿入されるソースデータを含んでいます。
- **Altova_Hierarchical.xsd** – 上記のインスタンスファイルを検証するために使用されるスキーマです。
- **altova.mdb** – 更新されるターゲットデータベースです。

全てのファイルは＜マイドキュメント＞¥Altova¥MapForce2021¥MapForceExamples¥フォルダー内にあります。下の説明では、簡素化のために、これらのファイルへの完全なパスは省略されています。

> この例のマッピングは、サンプルデータベースファイルを変更します。元のデータベースをバックアップし、下記のステップを開始する前に、新規のコピーを起動することが奨励されます。することが奨励されます。バックアップを作成することにより、元の例が上書きされず、下記と同じ結果を得ることができます。詳細に関しては、次を参照してください: データベースを変更するマッピングの実行。

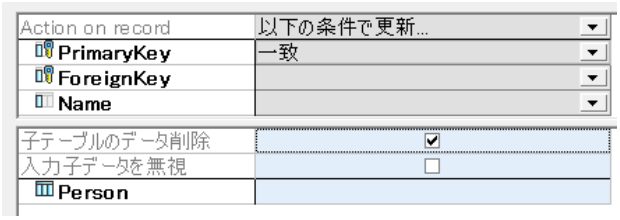
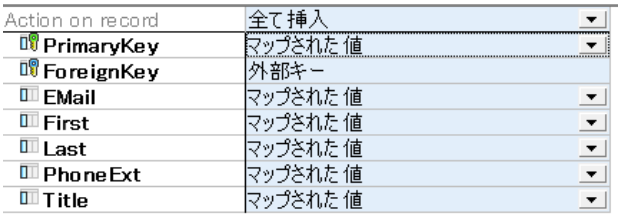
最初に、ソース XML スキーマとインスタンスとターゲットデータベースをマッピングに追加します（次を参照: 例: "altova.mdb" データベースをマッピングに追加する）。「以下の条件で更新し、残りを挿入」アクションと同じステップを踏んでください。次に、以下に表示されるようにマッピングの接続を描きます:



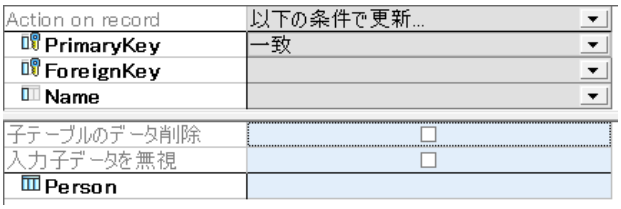
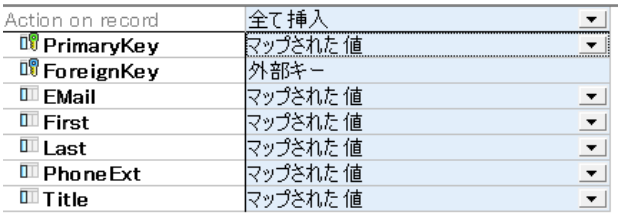
上記のように、マッピングは、ターゲットデータベース内の「Department」テーブルを更新します。「Department」テーブルは、「ルート」テーブルとして選択されています。ルートテーブルに関する情報と、その重要性に関する詳細は、次を参照してください： データベースリレーションシップの処理。子「Person」テーブルに対して取られるアクションは、このトピックの主題です。

次のテーブルは、多種の構成オプションと対応するマッピングの結果を表しています。これらのオプションは、親「Department」テーブルと子「Person」テーブルのデータベーステーブルアクションダイアログボックスから選択することができます。

構成 A

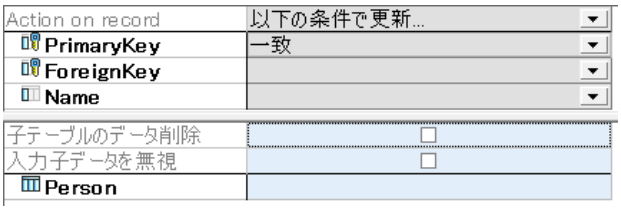
設定	マッピングの結果
 *「Department」テーブル*  *「Person」テーブル*	• ソースXML 内の `PrimaryKey` がデータベーステーブル内の `PrimaryKey` に対応する箇所で、Department レコードを更新します。 • 入力 XML ファイル内に対応するレコードを持たない（このようなキーはソースに存在しません）既存のDepartment レコードを更新しません。 • データベース内で既存ではないすべてのPerson レコードを、入力 XML インスタンスから挿入します。 • 「以下の条件で更新する...」条件を満たす Department レコードの子データ(Person レコード) を削除します。

構成 B

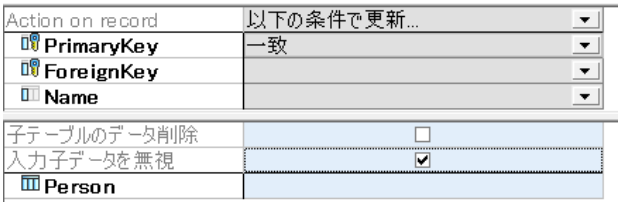
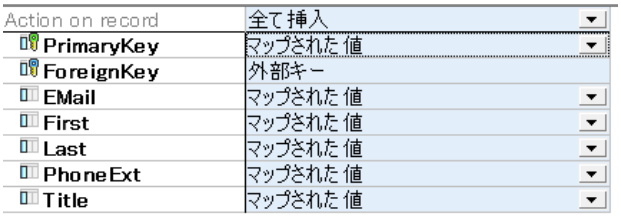
設定	マッピングの結果
 *「Department」テーブル*  *「Person」テーブル*	SQL 実行エラーと共にマッピングは失敗します。理由は、マッピングが、新規のPerson レコードを既存のPerson レコードと同じ主キーを使用して挿入しようと試みたからです。入力 XML からのレコードを、データベース内に既存のレコードに加えて追加する場合は、次を参照してください。

(C) 2015-2021 Altova GmbH

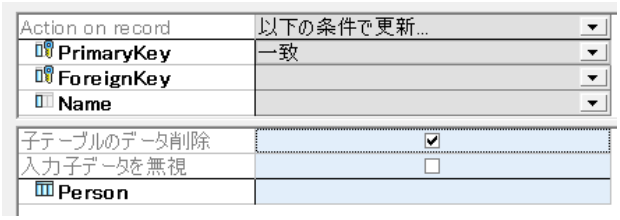
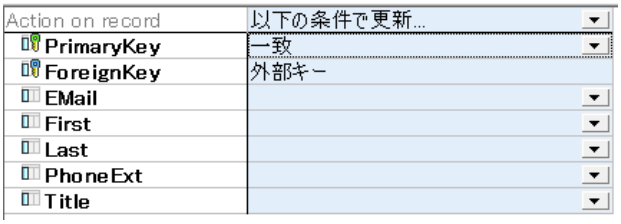
構成 C

設定	マッピングの結果
 *「Department」テーブル* *「Person」テーブル*	• ソースXML 内の`PrimaryKey` がデータベーステーブル内の`PrimaryKey` に対応する箇所で、Department レコードを更新します。 • 入力 XML ファイル内に対応するレコードを持たない（このようなキーはソースに存在しません）既存のDepartment レコードを更新しません。 • 新規のPerson レコード（生成された主キーと共に）は、既存のもの加え、Person テーブルに挿入されます。

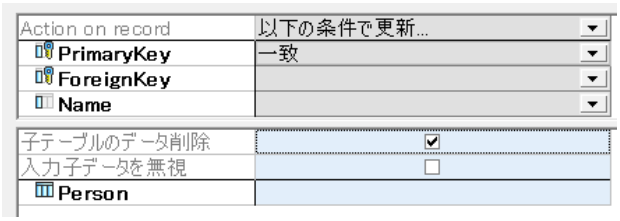
構成 D

設定	マッピングの結果
 *「Department」テーブル*  *「Person」テーブル*	• ソースXML 内の`PrimaryKey` がデータベーステーブル内の`PrimaryKey` に対応する箇所で、Department レコードを更新します。 • 入力 XML ファイル内に対応するレコードを持たない（このようなキーはソースに存在しません）既存のDepartment レコードを更新しません。 • オプション **入力子データを無視する** が親 Department テーブルのために有効化されているため、Person テーブルにレコードは挿入されません。

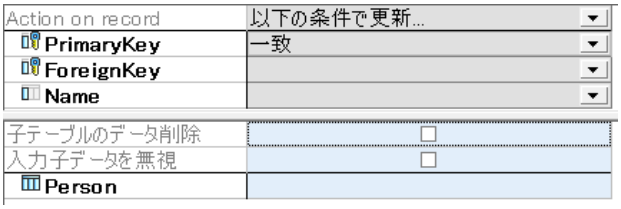
構成 E

設定	マッピングの結果
 *「Department」テーブル*  *「Person」テーブル*	• ソースXML 内の`PrimaryKey` がデータベーステーブル内の `PrimaryKey` に対応する箇所で、Department レコードを更新します。 • 入力 XML ファイル内に対応するレコードを持たない（このようなキーはソースに存在しません）既存のDepartment レコードを更新しません。 • ソースXML 内に対応する、`PrimaryKey` をもつ Department にリンクされているレコードを全て削除します。理由は、親 Department テーブルのために「**子テーブル内のデータの削除**」オプションが有効化されているからです。 • 「**以下の条件で更新する...**」条件を満たさなかった部署にリンクされているPerson レコードは、データベースに残されます。 • Person テーブル内のレコードは更新されていません。

構成 F

設定	マッピングの結果
 *「Department」テーブル* *「Person」テーブル*	• ソースXML 内の`PrimaryKey` がデータベーステーブル内の `PrimaryKey` に対応する箇所で、Department レコードを更新します。 • 入力 XML ファイル内に対応するレコードを持たない（このようなキーはソースに存在しません）既存のDepartment レコードを更新しません。 • ソースXML 内に対応する、`PrimaryKey` をもつ Department にリンクされているレコードを全て削除します。この理由は、「**子テーブル内のデータの削除**」オプションは、親 Department テーブルのために有効化されているからです。 • 「**以下の条件で更新する...**」条件を満たさなかった Department にリンクされているPerson レコードは、データベースに残されます。

構成 G

設定	マッピングの結果
 「Department」テーブル 「Person」テーブル	• ソースXML 内の `PrimaryKey` がデータベーステーブル内の `PrimaryKey` に対応する箇所で、Department レコードを更新します。 • 入力 XML ファイル内に対応するレコードを持たない（このようなキーはソースに存在しません）既存の Department レコードを更新しません。 • 次の2つの条件を満たす、すべての Person レコード を削除します: a. 対応するソースXML 内に対応する `PrimaryKey` が存在する Department にリンクされている Person レコード。 b. 対応するソースXML 内に対応する `PrimaryKey` が存在する Person レコード

6.2.3.7 「以下の条件で削除... 」アクション

テーブルアクション「**以下の条件で削除...**」(**Delete if..**) は、データベーステーブルからデータを条件付きで削除する場合に使用されます。データベーステーブル アクション ダイアログボックスからこのアクションを定義することができます。（次を参照してください: [データベーステーブルアクション設定)。](#) 例えば、ソースからのデータ XML を ターゲット データベースにマッピングする際、ソース XML 内の特定の条件が、ターゲット データベースの記録（通常、主キーの値）内のフィールドに対して同じかをチェックするために、「**以下の条件で削除...**」条件を構成することができます。「**以下の条件で削除...**」条件が true の場合、（つまり、2つのフィールドが等しい場合）データベースの記録は、マッピングが実行されると削除されます。

メモ 「**以下の条件で削除...**」テーブルアクションをデータベース テーブル アクション ダイアログボックス内で使用することのできる「**子テーブル内のデータの削除**」オプションと混合しないように注意してください。「**以下の条件で削除...**」テーブルアクションは、アクションが定義されているテーブルのみに影響を与え、他のテーブルには影響はありません。

この例は、データベース テーブルからデータを条件付きで削除する方法、また削除条件の結果をベースにして同じデータベーステーブルにレコードを挿入する方法を説明します。

この例は以下のファイルを使用します:

- **altova-cmpy-extra.xml** – データベースに挿入されるソースデータを含んでいます。
- **Altova_Hierarchical.xsd** – 上記のインスタンスファイルを検証するために使用されるスキーマです。
- **altova.mdb** – 更新されるターゲット データベースです。

全てのファイルは＜**マイドキュメント**＞¥Altova¥MapForce2021¥MapForceExamples¥フォルダー内にあります。下の説明では、簡素化のために、これらのファイルへの完全なパスは省略されています。

この例のマッピングは、サンプルデータベースファイルを変更します。元のデータベースをバックアップし、下記のステップを開始する前に、新

規のコピーを起動することが奨励されます。することが奨励されます。バックアップを作成することにより、元の例が上書きされず、下記と同じ結果を得ることができます。詳細に関しては、次を参照してください: データベースを変更するマッピングの実行。

このマッピングの目的は以下の通りです:

- 同じ `PrimaryKey` を持つ、person レコードが、ソースXML とターゲット Person テーブル内両方に存在する場合、これらはPerson テーブルから削除されなければなりません。
- ソースXML からの全てのレコードは、Person テーブルに挿入される必要があります。

マッピングの目的を達成するために、以下のステップを行います。

ステップ 1: ソース XML コンポーネントを挿入する

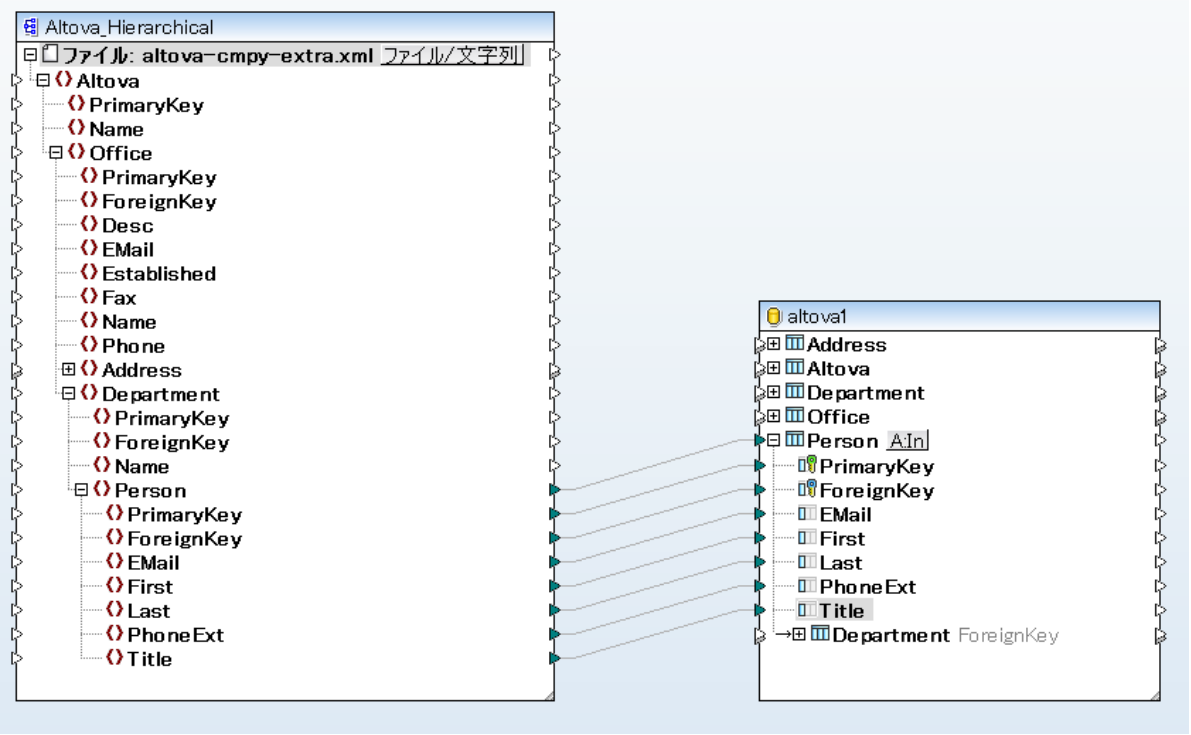
- 「挿入」メニューから、「XML スキーマ/ファイル」をクリックして、**Altova_Hierarchical.xsd** を参照します。インスタンスファイルを与えるように問われると、**altova-cmpy-extra.xml** を参照してください。

ステップ 2: ターゲットデータベースを挿入する

- 「挿入」メニューから、「データベース」をクリックして、 **altova.mdb** に接続するためにウィザードのステップを行ってください。(次を参照してください: 例: "altova.mdb" データベースをマッピングに追加する)。

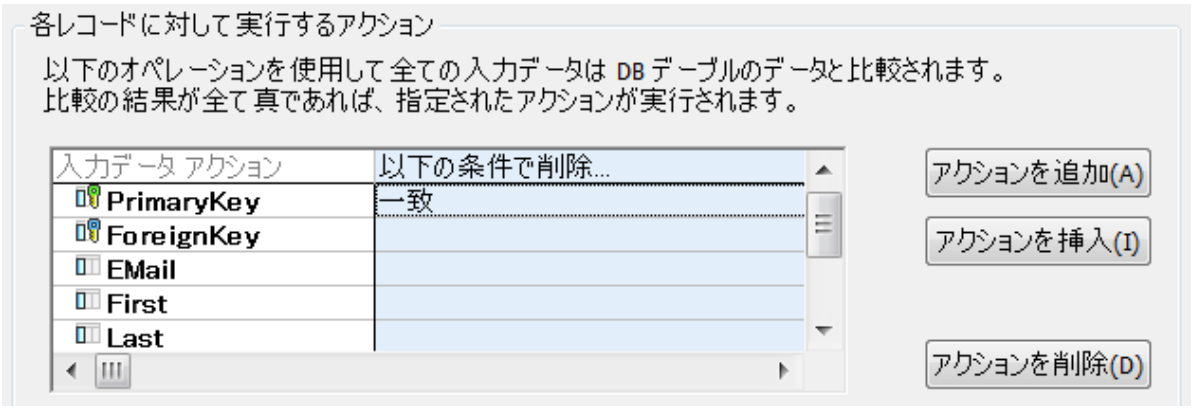
ステップ 3: 接続を描く

- 下に表示されるとおりマッピングの接続を描きます。

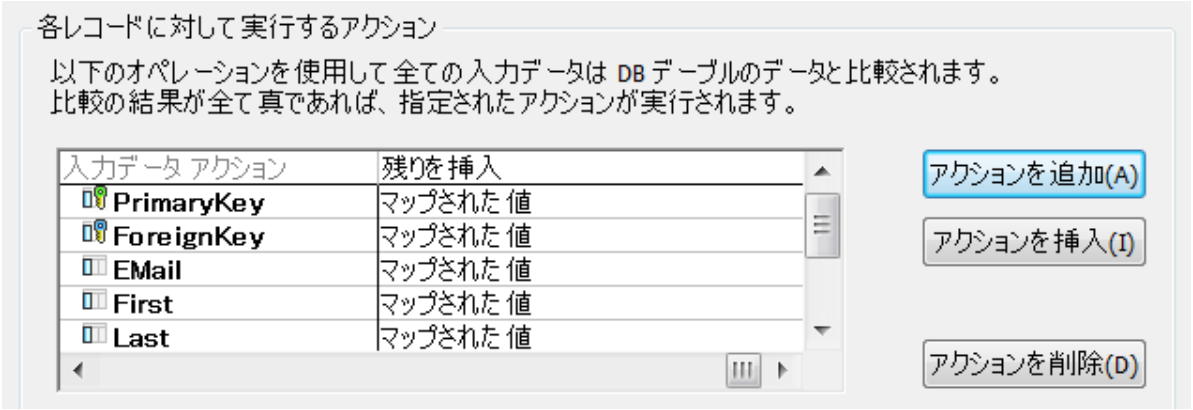


ステップ 4: 「以下の条件で削除…」「残りを挿入」" アクションを構成する

1. ターゲットコンポーネントで、「Person」テーブルの横の**アクション: 挿入** (A:In) ボタンをクリックします。
2. 「**レコードに対するアクション**」の横の、「**以下の条件で削除…**」を選択します。これは、データベーステーブルアクションを 条件月の「アクションの削除」に変更します。つまり、条件が満たされる場合のみ、現在のレコードは削除されます (次のステップを参照してください)。
3. 「PrimaryKey」の横の、値 「**一致**」を選択します。これにより、更新条件を定義します。つまり、下に表示されるとおり、データベースの記録は、「PrimaryKey」値が、マッピングより与えられる「PrimaryKey」値に等しい場合、削除されます。



4. 任意で、**NULL equal** チェックボックスを選択します。このサンプルでは、ソース XML 内の **PrimaryKey** フィールド、またはデータベース内の **PrimaryKey** フィールドに null 値が含まれていないため、このチェックボックスには関連性はありません。しかしながら、データに null 値が含まれており、このような値を等価として扱うには、場合、このチェックボックスをチェックする必要があります。それ以外の場合は、希望しない結果が生成される場合があります。詳細に関しては、データベーステーブルアクション内の Null 値の処理を参照してください。
5. 「**アクションの追加**」をクリックします。これは、新規のアクションを既存の「**以下の条件で削除…**」アクションの右に追加します。新規のアクションを「**残りを挿入**」として構成します:



上のイメージでは、データベーステーブルアクションは、マッピングの目的に従い構成されています。つまり、「**以下の条件で削除…**」(**Delete If..**) 条件が満たされる場合のみ、レコードは削除されます。それ以外の場合、挿入されます。オプション「マップされた値」は、マッピングからの値がレコードの全てのフィールドを作成するために使用されることを指定します。

6. ダイアログボックスを閉じるために「OK」をクリックします。マッピングの背景で、**アクション: 挿入** A:In ボタンが、**アクション: 削除; 挿入** (A:De,In) ボタンに変更されていることに注意してください。これは、削除と挿入アクションが、このテーブルのために構成されることを示します。

ステップ 5: マッピングをプレビューし、データベースを更新する

マッピングをプレビューするために「出力」タブをクリックします。データベースに対して実行されるアクションを含む、SQL スクリプトが生成されます。このスクリプトはプレビューのためであり、データベースをまだ変更していません。データベースに対してスクリプトを実行するには以下を行います:

- 「出力」メニューから、「SQL-スクリプトの実行」をクリックします。

メモ MapForce から直接 SQL スクリプトを実行することは、データベースを更新する方法のひとつです。次も参照してください: データベースを変更するマッピングの実行。

MapForce 内のDB クエリタブ内の「Person」テーブルを開くと(次を参照してください: データベースの参照とクエリ)、マッピングの結果は以下の通りです:

- XML ファイル内に対応する主キーを持つすべてのデータベースの記録は、削除されました。例は、主キー1、2、3、4、および 5 を持つレコードです。
- XML ファイル内に対応するキーをもたない全てのデータベースの記録は、影響を受けず、変更されていません。例は、主キー6、7、8、9、10、11、12、および13 を持つレコードです。
- 新規のレコードが(データベース内でキーが存在していない箇所である)「Person」テーブルに挿入されました。例は、主キー30 と31 を持つレコードです。

	PrimaryKey	ForeignKey	EMail	First	Last	PhoneExt	Title
1	6	3	f.landis@nanonull.com	Fred	Landis	951	Program Manager
2	7	3	m.landis@nanonull.com	Michelle	Butler	654	Software Engineer
3	8	3	t.little@nanonull.com	Ted	Little	852	Software Engineer
4	9	3	a.way@nanonull.com	Ann	Way	951	Technical Writer
5	10	3	l.gardner@nanonull.com	Liz	Gardner	753	Software Engineer
6	11	3	p.smith@nanonull.com	Paul	Smith	334	Software Engineer
7	12	4	a.martin@nanonull.com	Alex	Martin	778	IT Manager
8	13	4	g.hammer@nanonull.com	George	Hammer	223	Web Developer
9	30	1	c.Cicada@microtech.com	Camilla	Cicada	765	HR
10	31	1	c.corrigan@microtech.com	Carol	Corrigan	629	Admin

データベースを更新後の「Person」テーブル

6.2.3.8 「以下の条件で無視...」アクション

テーブルアクション「**以下の条件で無視...**」は、定義された条件に従い、データベース内の特定のレコードが更新されることを回避するために使用されます。「**以下の条件で無視...**」アクションは、(「**残りを挿入**」アクションなどの) 他のデータベーステーブルアクションと組み合わせて使用される場合のみ、意味があります。例えば、ソース XML からのデータをターゲットデータベースにマッピングする場合、「**以下の条件で無視...**」条件を構成し、ソース XML 内の特定のフィールドがターゲットデータベースの記録内の(通常、主キーの値である)フィールドと等価であるかをチェックするように構成することができます。マッピングが実行される際、「**以下の条件で無視...**」条件が true の場合、(つまり、2つのフィールドが等しい場合)、データベースの記録は無視され、次に定義されたアクション(例えば、「**残りを挿入**」) は、実行されます。

この例は、「**以下の条件で無視...**」条件の結果に基づいたデータベーステーブルにレコードを挿入する方法について説明します。この例は、次のファイルを使用します:

(C) 2015-2021 Altova GmbH

- **altova-cmpy-extra.xml** – データベースに挿入されるソースデータを含んでいます。
- **Altova_Hierarchical.xsd** – 上記のインスタンスファイルを検証するために使用されるスキーマです。
- **altova.mdb** – 更新されるターゲットデータベースです。

全てのファイルは<マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥フォルダー内にあります。下の説明では、簡素化のために、これらのファイルへの完全なパスは省略されています。

この例のマッピングは、サンプルデータベースファイルを変更します。元のデータベースをバックアップし、下記のステップを開始する前に、新規のコピーを起動することが奨励されます。することが奨励されます。バックアップを作成することにより、元の例が上書きされず、下記と同じ結果を得ることができます。詳細に関しては、次を参照してください: データベースを変更するマッピングの実行。

このマッピングの目的は以下の通りです:

- ソースXML とターゲット Person テーブル内の両方で、同じ `PrimaryKey` を持つ個人のレコードが存在する場合、 アクションは取られません。(つまり、無視されます)。
- 個人のレコードがPerson テーブル内に存在する上記の条件を満たさない場合、アクションは取られません。
- Person テーブル内に対応する(主キーが無い)ソースXML からのレコードは、新規として扱われ、新規の主キーと共に Person テーブルに挿入されます。

マッピングの目的を達成するために、以下のステップを行います。

ステップ 1: ソース XML コンポーネントを挿入する

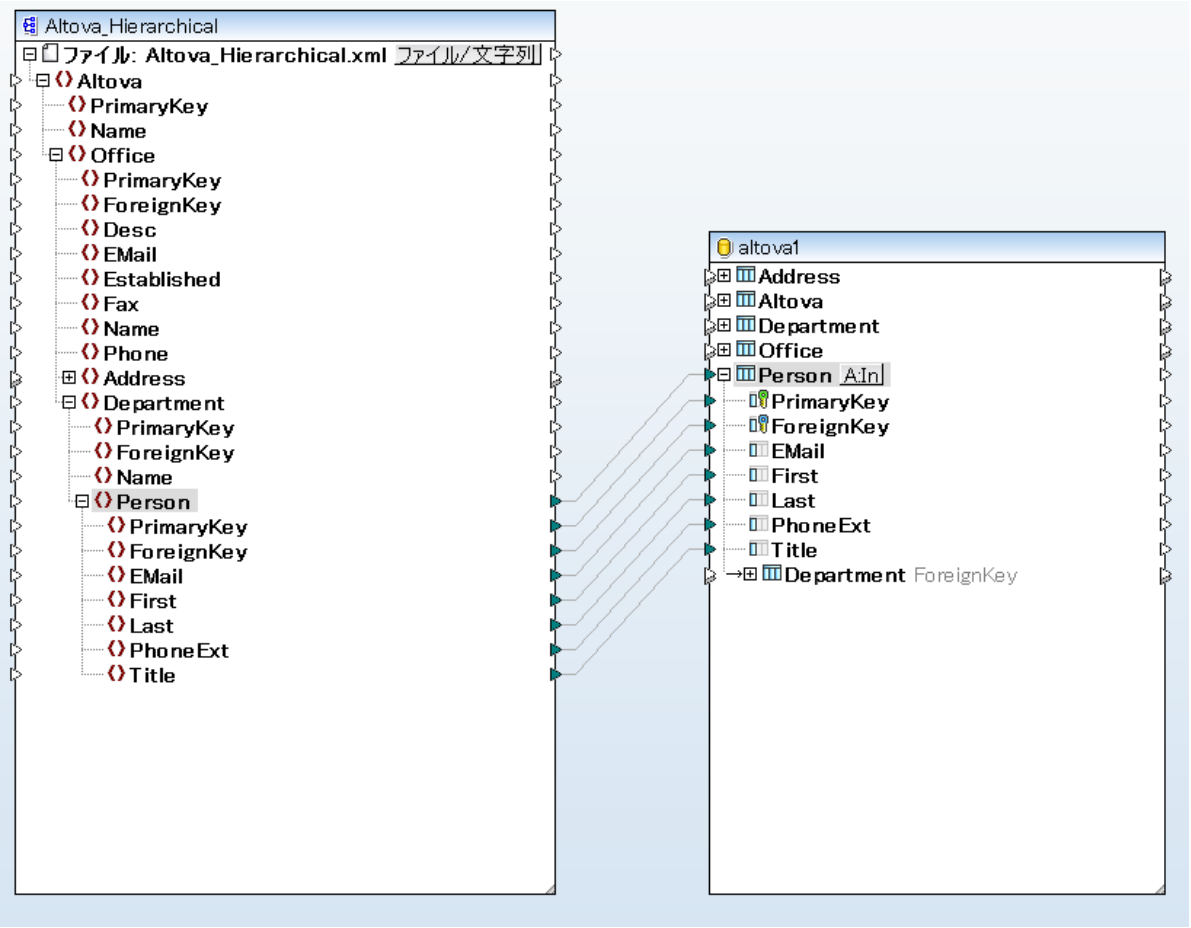
- 「挿入」メニューから、「XML スキーマ/ファイル」をクリックして、**Altova_Hierarchical.xsd** を参照します。インスタンスファイルを与えるように問われると、**altova-cmpy-extra.xml** を参照してください。

ステップ 2: ターゲットデータベースを挿入する

- 「挿入」メニューから、「データベース」をクリックして、**altova.mdb** に接続するためにウィザードのステップを行ってください。(次を参照してください: 例: "altova.mdb" データベースをマッピングに追加する)。

ステップ 3: 接続を描く

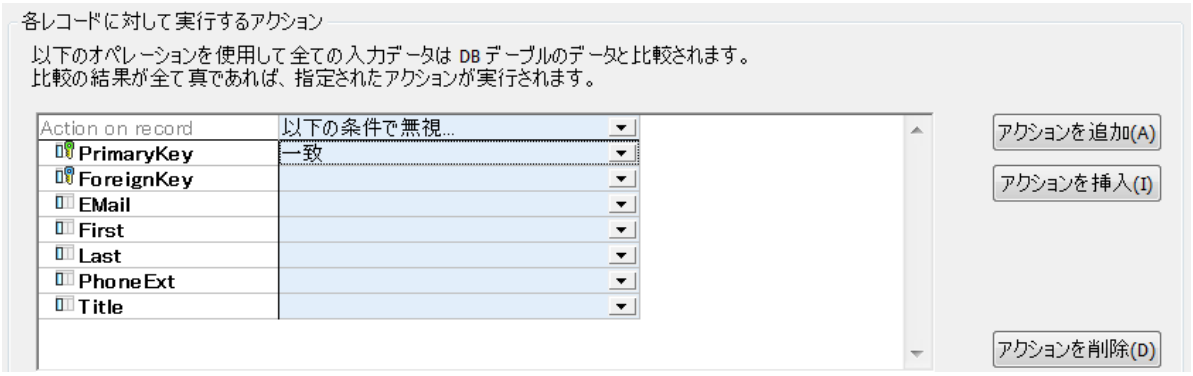
- 下に表示されるとおりマッピングの接続を描きます。



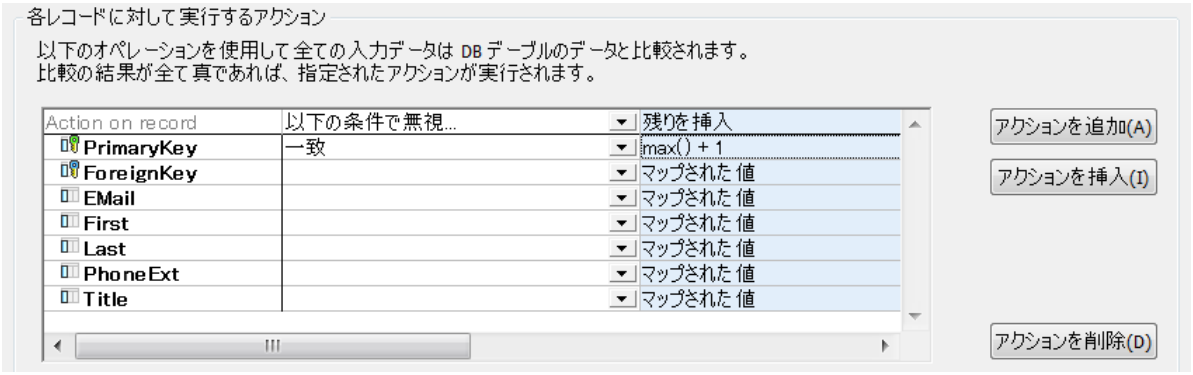
ステップ 4: "「以下の条件で無視…残りを挿入」" アクションを構成する

1. ターゲットコンポーネントで、「Person」テーブルの横の**アクション: 挿入** (A:In) ボタンをクリックします。
2. **「レコードに対するアクション」**の横の、**「以下の条件で無視…」**を選択します。これは、条件付きの無視アクションへのデータベーステーブルアクションを変更します。つまり、条件が満たされる場合のみ、現在のレコードは無視されます（次のステップを参照してください）。
3. **「PrimaryKey」**値の横から下に表示されるとおり、**equal** を選択します。これは無視の条件を定義します。つまり、データベースの記録は、`PrimaryKey` 値がマッピングから与えられる `PrimaryKey` の値と同じ場合、無視されます。

(C) 2015-2021 Altova GmbH



4. 任意で、**NULL equal** チェックボックスを選択します。このサンプルでは、ソースXML内の**PrimaryKey** フィールド、またはデータベース内の**PrimaryKey** フィールドにnull 値が含まれていないため、このチェックボックスには関連性はありません。しかしながら、データにnull 値が含まれており、このような値を等価として扱うには、場合、このチェックボックスをチェックする必要があります。それ以外の場合は、希望しない結果が生成される場合があります。詳細に関しては、データベーステーブルアクション内のNull 値の処理を参照してください。
5. 「**アクションの追加**」をクリックします。これは、新規のアクションを既存の「**以下の条件で無視...**」アクションの右に追加します。主キーが**max() + 1** に設定され、下に表示されるとおり、新規のアクションが「**残りを挿入**」として構成されます：



上のイメージでは、データベーステーブルアクションは、マッピングの目的に従い構成されています。つまり、「**以下の条件で無視...**」条件が満たされると、レコードはスキップされます。それ以外の場合、挿入されます。オプション「マップされた値」は、マッピングからの値がレコードの全てのフィールドを作成するために使用されることを指定します。オプション**max() + 1** は、レコードのために一意の、新規の主キーの値を生成します。

6. ダイアログボックスを閉じるために「**OK**」をクリックします。マッピングでは、**アクション: 挿入** A:In ボタンは、**アクション: 無視; 挿入** ( ) ボタンに変更されます。これは、アクションの無視と挿入がこのテーブルのために構成されていることを示しています。

ステップ 5: マッピングをプレビューし、データベースを更新する

マッピングをプレビューするために「**出力**」タブをクリックします。データベースに対して実行されるアクションを含む、SQL スクリプトが生成されます。このスクリプトはプレビューのためであり、データベースをまだ変更していません。データベースに対してスクリプトを実行するには以下を行います:

- 「**出力**」メニューから、「**SQL-スクリプトの実行**」をクリックします。

メモ MapForce から直接 SQL スクリプトを実行することは、データベースを更新する方法のひとつです。次も参照してください: データベースを変更するマッピングの実行。

MapForce 内のDB クエリタブ内の「Person」テーブルを開くと（次を参照してください： データベースの参照とクエリ）、マッピングの結果は以下の通りです:

- XML ファイル内に対応する主キーを持つ全てのデータベースの記録は、「以下の条件で無視...」条件を満たし、影響を受けず、変更されていません。例は、主キー1、2、3、4、および5 を持つレコードです。
- XML ファイル内で対応するキーを持たない全てのデータベースの記録は、「以下の条件で無視...」条件を満たしませんでしたが、影響を受けず変更されていません。例は、主キー6、7、8、9、10、11、12、および13 を持つレコードです。
- 新規のレコードが（データベース内でキーが存在していない箇所である）「Person」テーブルに挿入されました。例は、主キー30 と31 をソースXML ファイル内に持つレコードです。これらは、それぞれ新規の主キー22 と23 と共にデータベースに挿入されます。

	PrimaryKey	ForeignKey	EMail	First	Last	PhoneExt	Title
1	1	1	v.callaby@nanonull.com	Vernon	Callaby	582	Office Manager
2	2	1	f.further@nanonull.com	Frank	Further	471	Accounts Receivable
3	3	1	l.matise@nanonull.com	Loby	Matise	963	Accounting Manager
4	4	2	j.firstbread@nanonull.com	Joe	Firstbread	621	Marketing Manager Europe
5	5	2	s.sanna@nanonull.com	Susi	Sanna	753	Art Director
6	6	3	f.landis@nanonull.com	Fred	Landis	951	Program Manager
7	7	3	m.landis@nanonull.com	Michelle	Butler	654	Software Engineer
8	8	3	t.little@nanonull.com	Ted	Little	852	Software Engineer
9	9	3	a.way@nanonull.com	Ann	Way	951	Technical Writer
10	10	3	l.gardner@nanonull.com	Liz	Gardner	753	Software Engineer
11	11	3	p.smith@nanonull.com	Paul	Smith	334	Software Engineer
12	12	4	a.martin@nanonull.com	Alex	Martin	778	IT Manager
13	13	4	g.hammer@nanonull.com	George	Hammer	223	Web Developer
14	14	4	j.band@nanonull.com	Jessica	Bander	241	Support Engineer
15	15	4	l.king@nanonull.com	Lui	King	345	Support Engineer
16	16	5	s.meier@nanonull.com	Steve	Meier	114	Office Manager
17	17	5	t.bone@nanonull.com	Theo	Bone	331	Accounts Receivable
18	18	6	m.nafta@nanonull.com	Max	Nafta	122	PR & Marketing Manager US
19	19	7	v.bass@nanonull.com	Valentin	Bass	716	IT Manager
20	20	7	c.franken@nanonull.com	Carl	Franken	147	Support Engineer
21	21	7	m.redgreen@nanonull.com	Mark	Redgreen	152	Support Engineer
22	22	1	c.Cicada@microtech.com	Camilla	Cicada	765	HR
23	23	1	c.corrigan@microtech.com	Carol	Corrigan	629	Admin

データベースの更新後の「Person」テーブル

6.2.3.9 トランザクションロールバックの使用

データベースを変更するマッピングを実行する場合、MapForce（またはMapForce Server、またはMapForceにより生成されたプログラム）ではデータベースに関連したエラーが発生する場合があります。例えば、データベースアカウントには、特定のデータベースアクションを実行するだけの特権がない、または無効なデータを入力、または更新することを回避するデータベース制約を実行する特権がない場合があります。後者は、列が必須で、マッピングが値を適用しない場合に発生する場合があります。

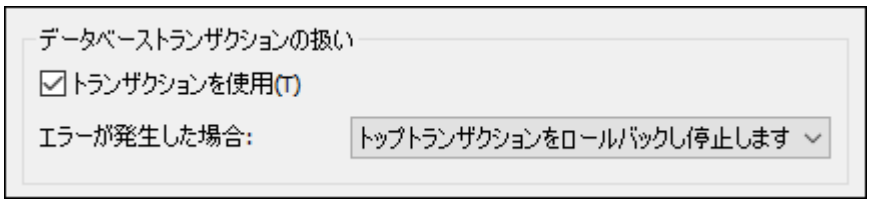
(C) 2015-2021 Altova GmbH

データベース例外が発生した場合にロールバックを可能にするには、マッピングがデータベーストランザクションを使用するように構成します。下で説明されるとおり、データベースコンポーネントレベルとテーブルアクションレベルでトランザクションを有効化することができます。

データベースコンポーネントレベルでのトランザクション

データベースコンポーネントレベルでトランザクションを有効化するには、以下を行います:

1. データベースコンポーネントを右クリックし、コンテキストメニューから「**プロパティ**」を選択します(代わりに、データベースコンポーネントのタイトルバーをダブルクリックします)。これによりデータベースコンポーネント設定 ダイアログボックスが開かれます。



2. 「**トランザクションの使用**」チェックボックスを選択します。

これを行うことにより、データベースコンポーネントにより行われるすべての変更がエラーを発生した場合ロールバックされるトランザクション内に囲まれます。マッピングの実行中にエラーが発生すると、出力はドロップダウンリストから選択されたオプションにより異なります:

- **トップトランザクションをロールバックし停止** – データベースコンポーネントにより行われるすべての変更を囲むトランザクションはロールバックされ、マッピングは停止されます。
- **トップトランザクションをロールバックし継続** – 上記同様ですが、マッピングはロールバック後も継続されます。(例えば、存在する場合、2番目のターゲットコンポーネントを処理する場合など)。

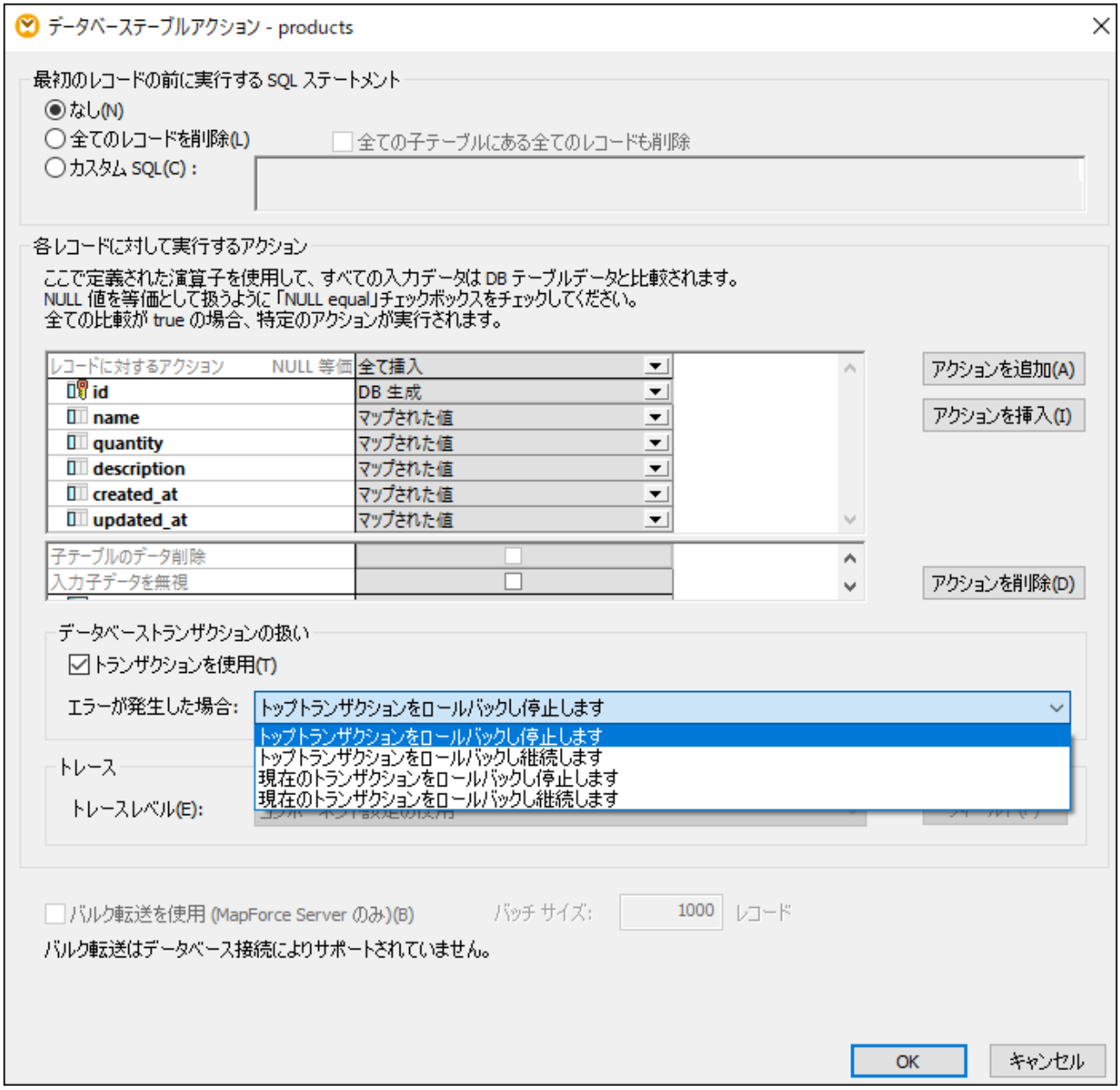
マッピングの一部は複数のデータベースコンポーネントを含んでいる可能性があります。これらは、同じデータベース接続を使用する、またはまったく異なるデータベースへの接続を使用している場合があります。このようなマッピングの出力は、データベースエラーが発生する場合、実行エンジンによりとなります:

- マッピングがMapForce で実行される場合マッピングの実行時、1つのターゲットコンポーネントが実行されます。「**プレビュー**」 ボタンが有効化されている個所のコンポーネントです。そのコンポーネント内でデータベースエラーが発生し、「**トランザクションの使用**」チェックボックスが有効化されていると、コンポーネントによる全ての変更がロールバックされます。この場合、データベースエラーが発生すると、エラーが発生する箇所でロールバックが発生し、マッピングは停止されるか、**エラーが発生する箇所で**ドロップダウンリストで選択されたとおり、次のターゲットコンポーネントの実行を継続します。

データベースコンポーネント設定 ダイアログボックスで使用することのできるすべての設定に関数の参照に関しては、データベースコンポーネント設定 を参照してください。

テーブルアクションレベルでのトランザクション

トランザクション at テーブルアクション レベルでのトランザクションは、個別の(挿入、更新などの)テーブルアクションを含みます。このセットアップでは、レコード上で挿入、または更新アクションが不足する場合、現在失敗したアクションのみをロールバックし、他のレコードの処理を続行することができます。テーブルアクションレベルでトランザクションを実行するには、データベーステーブルアクション ダイアログボックス「**トランザクションの使用**」チェックボックスから選択します。(次を参照: データベーステーブルアクション設定).



使用することのできるオプションは以下の通りです:

- **トップトランザクションをロールバックし停止** – 親子リレーションシップを持つマッピングが複数のテーブルを更新する場合、親テーブルのためだけではなく、子テーブルのためにトランザクションを有効化することができます。トランザクションが複数のレベルで有効化することができるため、「トップ」は階層で更に高いレベルを参照することができます。これは、そのレベルでのトランザクションが有効化されている場合、データベースコンポーネントレベルからのトランザクションです。それ以外の場合、コンポーネントには複数のネストされたテーブルアクションが存在することが仮定され、一番上のテーブルアクションで定義されたトランザクションが取られます。データを複数にリンクされているテーブルに挿入する で説明されるとおり、このオプションを理解するには、コンポーネントが複数のテーブルの階層を更新することができることを思い出してください。例えば、コンポーネントには、個人のレコードを挿入する親「挿入」アクション、および、個別のテーブルに各個人のレコードを挿入する1つまたは複数のネストされた「挿入」アクションが存在します。両方の挿入アクションのためにトランザクションを有効化することができ、「トップ」は階層レベルの最も高いレベルを指しています(このサンプルでは、個人がアクションを挿入しています)。このため、アドレスを挿入する際に例外が発生すると、これは(このサンプルでは、個人の

(C) 2015-2021 Altova GmbH

テーブルレベルである)トップのトランザクションをロールバックし、実行は停止されます。すなわち、個人のアドレスを挿入できない場合、個人のレコードは挿入されません。

- **トップトランザクションをロールバックし継続する** – 上記同様ですが、ロールバックの後もマッピングは継続されます(例えば、存在する場合、他のターゲットコンポーネントを処理するために継続されます)。
- **現在のトランザクションをロールバックし停止する** – 例外が発生すると、このロールバックは、現在のトランザクション内で発生した変更のみをロールバックします。現在のトランザクション外で発生した変更は、コミットされます。
- **現在のトランザクションをロールバックし継続する** – 上記同様ですが、マッピングはロールバック後も継続されます。

ストアドプロシージャレベルトランザクション

ストアドプロシージャのためのデータターゲットと呼ばれるデータベーストランザクションを有効化することができます。例えば、ターゲットコンポーネント内のストアドプロシージャを参照してください。(入力パラメーターを持たない)データソースとして呼び出される、または関数のように呼び出される(入力と出力を持つ)ストアドプロシージャのためにはトランザクションはサポートされていません。.

ロールバック

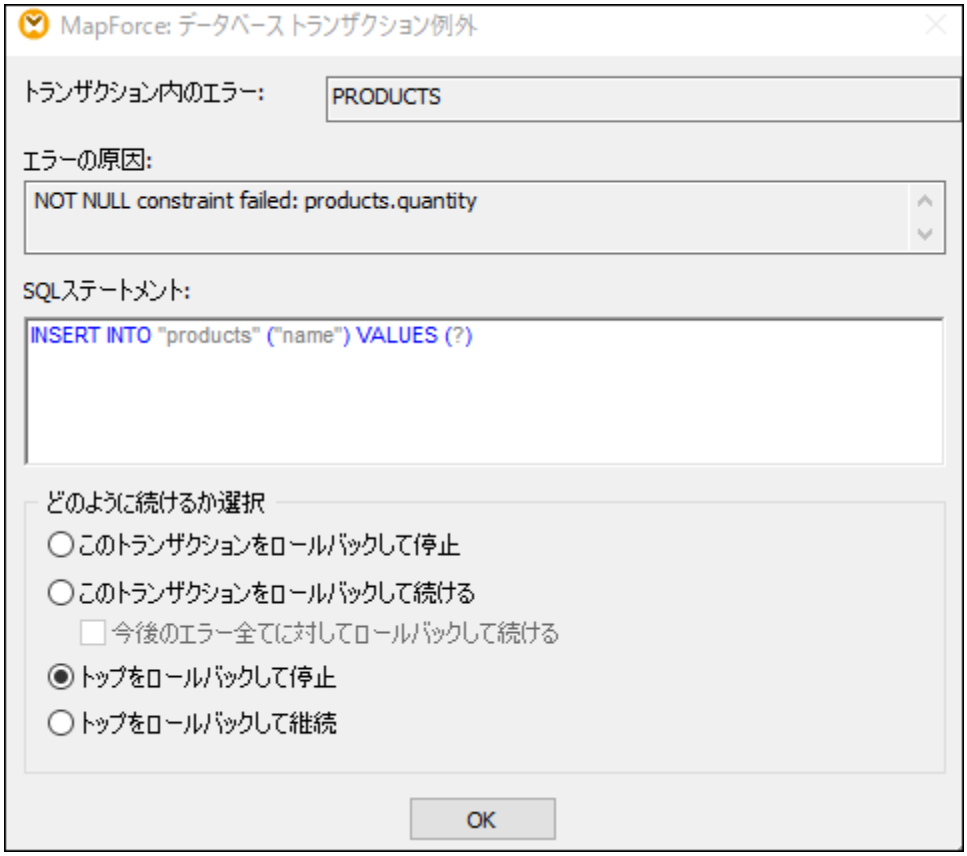
トランザクションがマッピングランタイムに発生するかを確認するには、「**出力**」タブをクリックして、MapForce により生成された疑似-SQL codeコードを確認します。下のイメージは、コンポーネントレベルとテーブルアクションレベルでトランザクションが有効化されているマッピングのプレビューを示しています。下に示されているとおり、メインのトランザクションは各オペレーションのために発生します(コメント `--begin transaction` と`--commit transaction` により識別されます)。また、複数の小さいトランザクションは各入力オペレーションのために発生します(`SAVEPOINT` コマンドにより識別されます)。

```
 7
 8    -- begin transaction
 9
10    PRAGMA foreign_keys = ON;
11
12    SAVEPOINT PRODUCTS
13
14    INSERT INTO "products" ("name") VALUES ('Product A')
15
16    SAVEPOINT PRODUCTS
17
18    INSERT INTO "products" ("name") VALUES ('Product B')
19
20    SAVEPOINT PRODUCTS
21
22    INSERT INTO "products" ("name") VALUES ('Product C')
23
24    -- commit transaction
25
```

マッピング | DB クエリ | 出力

マッピングがMapForce Server またはMapForce により生成されたプログラムにより実行されると、上記のとおり、マッピングのデザイン中に構成されたオプションに基づきロールバックが実行されます。例えば、オプション「**トップトランザクションをロールバックし停止**」を選択すると、他の挿入が成功した場合でも、失敗した入力オペレーションにより全ての変更がロールバックされます。しかしながら、「**現在のトランザクションをロールバックし継続する**」を選択すると、失敗した挿入が最も近い保存ポイントにロールバックされ、マッピングは他のレコードを挿入しようと試みます。

メニューコマンド「**出力 | Run SQL-スクリプトの実行**」を使用してMapForce 内でトランザクションが有効化されたマッピングを直接実行すると、ダイアログボックスはデータベースの例外が発生したかを通知します。例:



以降の振る舞いを変更することができます:

- **このトランザクションをロールバックし停止する** – これは、現在のトランザクションに含まれる変更のみをロールバックし、マッピングの実行を停止します。現在のトランザクションの外部で前に行われた変更はコミットされます。
- **このトランザクションをロールバックし継続する** – 上記同様ですが、マッピングロールバック後も実行されます。チェックボックス **ロールバックして全てのエラーのために継続する** を選択して、更にエラーが発生した場合、ダイアログボックスを抑制します。
- **すべてをロールバックし停止する** – このデータベース呼び出し内のすべてのトランザクションは、ロールバックされ、マッピングの実行は停止されます。

6.2.3.10 バルク挿入 (MapForce Server)

「**バルク転送**」コマンドによりMapForce コンポーネント (TXT、CSV、DAT など) からデータベーステーブルに高速にデータを挿入することができます。このオプションを使用することにより、1つのステートメントを実行することにより、挿入プロセスを劇的にスピードアップすることができます。

「**バルク転送を使用**」オプションは、マッピングのデザイン時に下に示されるとおり、MapForce 内で有効化することができます。このオプションが有効化することのできるマッピングは、MapForce 内で実行することができますが、バルク挿入はこの時点では適用することはできません。データの実際のバルク転送はMapForce Server によりマッピングが実行された時に起こります。

バルク転送は以下の条件が満たされた場合にサポートされます

- マッピング変換言語がBUILT-IN に設定されている場合。詳細に関しては、次を参照してください: 変換言語の選択。

- マッピングがスタンドアロン、またはFlowForce Server の管理下で MapForce Server により作動している場合。これは、マッピングが.mfx フォーマットにコンパイルされるか、またはFlowForce Server にデプロイされる必要があることを意味します。詳細に関しては、次を参照してください：MapForce Server 実行可能ファイルにマッピングをコンパイルするとFlowForce Server にマッピングをデプロイする。
- MapForce Server ライセンスは、マルチコアマシン上の"単一スレッドの実行" だけに制限されません。つまり、チェックボックス Altova LicenseServer の"サーバー管理" タブ内の「単一スレッドの実行に制限する」が無効化されている必要があります。
- データベースアクションは「全て挿入する」です。次も参照してください：データベーステーブルアクション設定。
- データがバルクロードされるテーブルは、"リーフテーブル" である必要があります。すなわち、データベースの一番下の階層である必要があります。関連したテーブル、ビュー、またはマッピング内のテーブルを参照するストアドプロシージャは存在してはなりません。
- データベースドライバーはW HERE 条件でのバルク挿入をサポートします。

次のテーブルは使用されるデータベースの種類とドライバーにより異なるバルク挿入へのサポートの概要を示しています。

	ADO	ODBC	JDBC	ADO.NET	ネイティブ
Access	いいえ	いいえ	n/a	n/a	n/a
DB2	いいえ	はい	はい	はい	n/a
Firebird	n/a	はい	はい	いいえ	n/a
Informix	いいえ	はい	はい	はい	n/a
iSeries	いいえ	はい	はい	はい	n/a
MariaDB	いいえ	はい	はい	はい	n/a
MySQL	n/a	はい*	はい	はい	n/a
Oracle	いいえ	はい	はい	はい	n/a
PostgreSQL	n/a	はい	はい	n/a	はい
Progress	n/a	はい	はい	n/a	n/a
SQL Server	はい	はい	はい	はい	n/a
SQLite	n/a	n/a	n/a	n/a	いいえ
Sybase	いいえ	はい	はい	n/a	n/a
Teradata	n/a	はい	はい	n/a	n/a

** MySQL バージョン 5 または以降が必要とされます。*

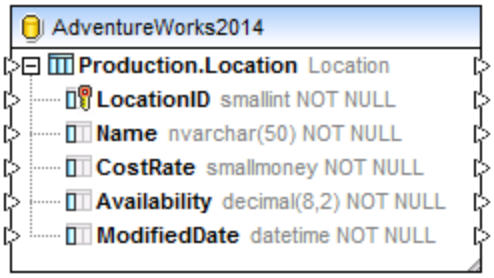
この例は、サンプル**source.txt** ファイルからデータをターゲットデータベースにバルクロードするマッピングを作成する方法について説明しています。例は、SQL Server 2014 とAdventureW orks 2014 データベースを使用しています。後者は、CodePlex W eb サイトからダウンロードすることができます（https://msftdbprodsamples.codeplex.com/）。

```
Location A,15.3,39
Location B,46,34
Location C,56.33,0
Location D,0,399
Location E,0,97.43
```

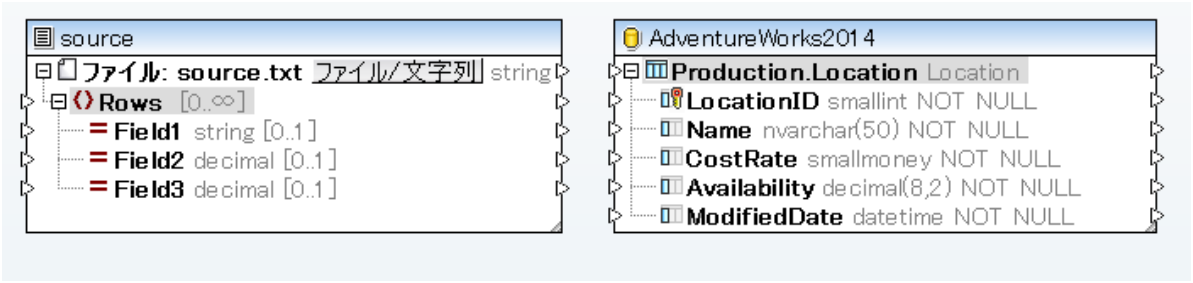
source.txt

バルク転送の定義:

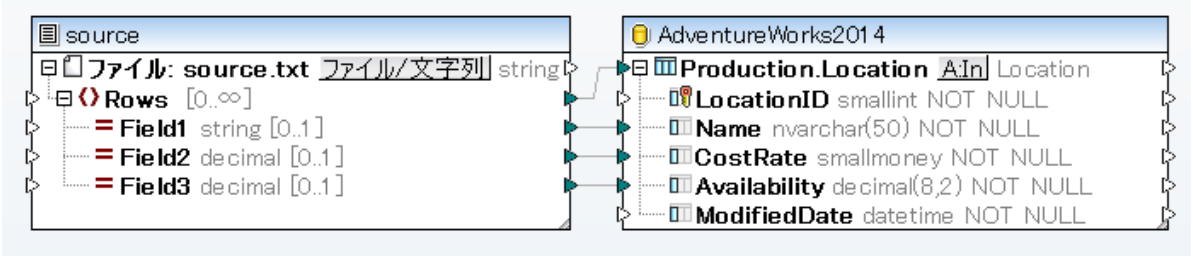
1. 内蔵のターゲット出力アイコン が有効化されていることを確認してください。
2. AdventureW orks 2014 データベースに接続し、"Production.Location" テーブルをマッピングに追加します。詳細に関しては、次を参照してください: データベースをマッピングに追加する。



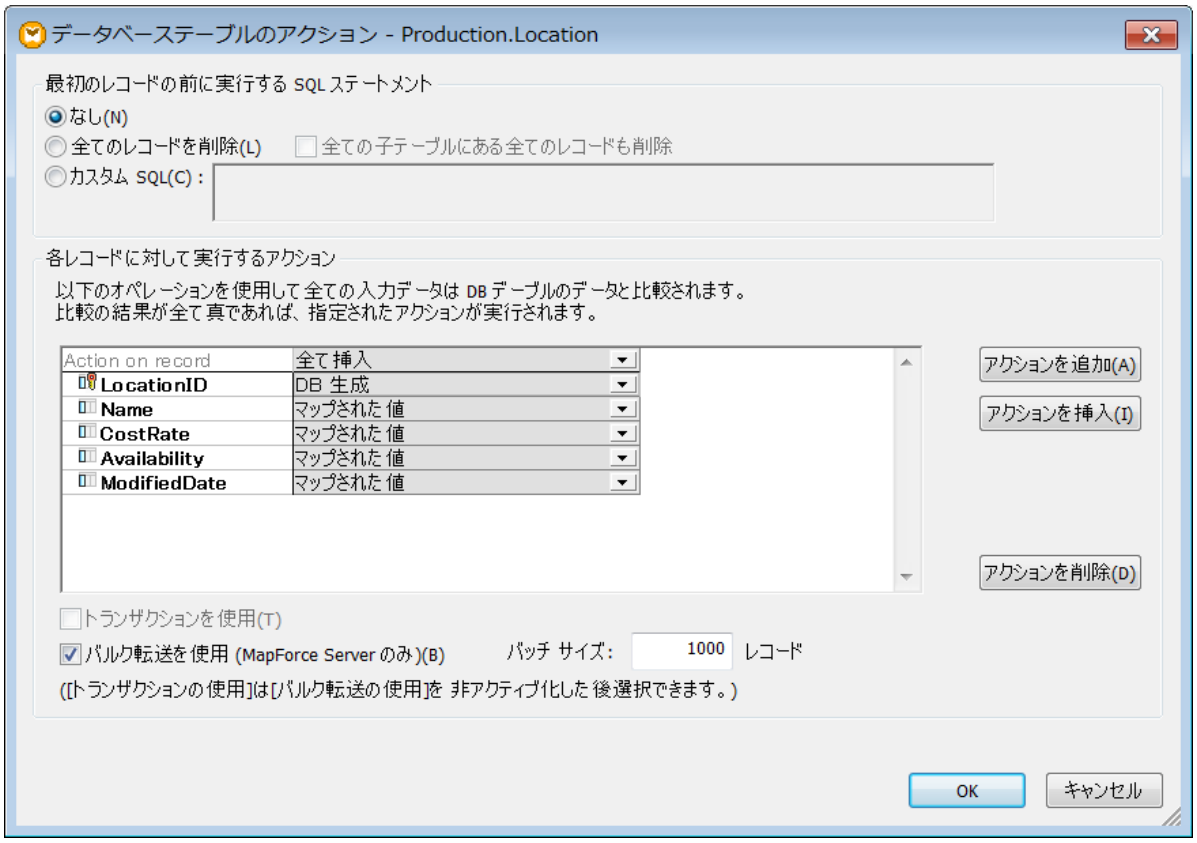
1. 「挿入」メニューから、「テキスト」をクリックして、(上に表示されている **source.txt** サンプル等の) ソーステキストファイル をマッピングに追加します。詳細に関しては、次を参照してください: CSV とテキストファイル。ソースとターゲット コンポーネントのデータ型と互換性があることを確認してください。「**データ型の表示**」( ) ツールボタンが有効化されている場合、データ型はコンポーネント上で表示されます。



2. 下に表示されるとおりマッピングの接続を描きます。**データベースアクション** ボタン A:In は、テーブル名の右に表示されていることに注意してください。



3. **データベースアクション** ボタン (A:In) をクリックして、"バルク転送の使用" チェックボックスを有効化します。「OK」をクリックして確認します。「**バルク転送の使用**」チェックボックスを選択して、「OK」をクリックして確認します。



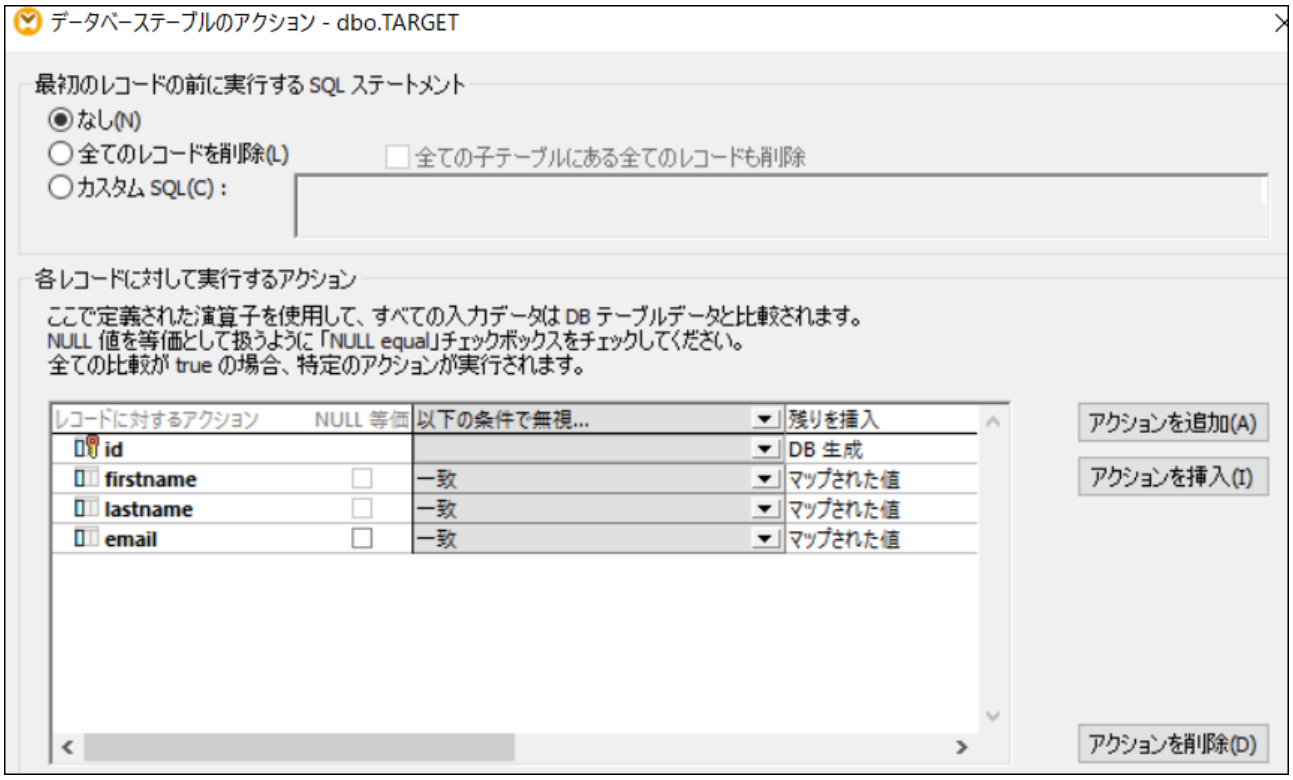
上のダイアログボックス内で、「レコードに対するアクション」が「全て挿入」に設定されていることを確認してください。「**バッチサイズ**」フィールドは、各アクションに挿入されるレコードの数量を定義します。

メモ 「**バルク転送の使用**」オプションが有効化されていると、「**トランザクションの使用**」オプションは無効化されます。また、「**トランザクションの使用**」オプションが有効化されていると、「**バルク転送の使用**」オプションは無効化されます。トランザクション処理を有効化するには、「**バルク転送の使用**」チェックボックスのチェックを解除します。

バルク挿入が有効化された次のステップは、MapForce Server 内でのマッピングの実行です(スタンドアロン、または FlowForce Server の管理下で)。詳細に関しては、次を参照してください:MapForce Server 実行可能ファイルにマッピングをコンパイルする と FlowForce Server にマッピングをデプロイする。

6.2.3.11 データベーステーブルアクション内の Null 値の処理

「Ignore If」、「Update If」、「Delete If」などのテーブルアクションを使用してマッピングがターゲットデータベースを更新する場合、MapForce はソースデータをターゲットデータに対して比較し、結果として内部データベース更新クエリを生成します。(これらの内部クエリは、MapForce の**出力**ペイン内のプレビューのために使用することができます次を参照してください: データベースを変更するマッピングの実行)。生成されたクエリは、「データベーステーブルアクション」ダイアログボックスから定義された比較条件を反映しています。



データベーステーブルアクション ダイアログボックス

ソースまたはターゲットデータにnullable フィールドが含まれている場合、ソース内のnull 値をターゲットデータ内の値と比較する方法が2つ存在します:

1. null 値を等価として扱う
2. null 値を等価として扱わない

デフォルトでは、NULL-対応は、無効化されています。条件が満たされており、NULL-対応が無効化されている場合、ターゲットデータベーステーブルが期待通り更新されないインスタンスが存在する場合があります(例えば、必要以上の行が挿入、または更新されます)。NULL 値 がデータの比較を影響し、希望しない結果を生成するために このような状況が発生します。このような状況を回避するために、「データベーステーブルアクション」ダイアログボックスから各 Null 許容 フィールド(下のイメージ内 email)の横のチェックボックスを選択します。

マッピング内のNULL 対応を良く理解するために、以下の条件がtrue の場合、NULL データの比較が起こるサンプルを分析してみましょう:

1. 「データベーステーブルアクション」ダイアログボックスは、「Ignore if」、「 Update if」、「Delete if」アクションを含んでいます。
2. これらのアクションは、NULL 値を含んでいる可能性のあるレコードに対して行われます。
3. ソーステーブル内のNULL 値は、ターゲットテーブル内のNULL 値の等価として扱われる必要があります。

デフォルトでは、NULL-対応は、無効化されています。条件が満たされており、NULL-対応が無効化されている場合、ターゲットデータベーステーブルが期待通り更新されないインスタンスが存在する場合があります(例えば、必要以上の行が挿入、または更新されます)。NULL 値 がデータの比較を影響し、希望しない結果を生成するために このような状況が発生します。このような状況を回避するために、「データベーステーブルアクション」ダイアログボックスから各 Null 許容 フィールド(下のイメージ内 email)の横のチェックボックスを選択します。Null 許容であるフィールドのためにのみ、また、少なくとも1つのテーブルアクションが「等価」または「等価(大文字と小文字を区別しない)」場合のみ、チェックボックスを選択することができます。

(C) 2015-2021 Altova GmbH

サンプル

マッピング内のNULL 対応を良く理解するために、NULL データの比較が起こるサンプルを分析してみましょう。このサンプルは、Microsoft SQL Server データベースを使用します。しかしながら、サポートされる他のデータベースの型にも適用することができます。任意で、Microsoft SQL Server を使用する場合、次のデータベーススクリプトを実行してこのサンプル内で使用されているテーブルとデータを作成することもできます: **<マイドキュメント>\Altova\MapForce2021\MapForceExamples\Tutorial\CreateNullableFields.sql**.

簡素化のために、データベーステーブルは以下のように説明されています。両方のテーブルは、個人のデータを保管しており、同じ列が存在しています。更に、列 `email` は、両方のテーブル内のnull データを含むことができます。

```
+----+-----------+-----------+--------------------------+
| id | firstname | lastname  | email                    |
+----+-----------+-----------+--------------------------+
| 1  | Toby      | Hughey    | t.hughey@nanonull.com    |
| 2  | Mia       | Dahill    | NULL                     |
| 3  | Fred      | Weinstein | f.weinstein@nanonull.com |
+----+-----------+-----------+--------------------------+
```

SOURCE テーブル

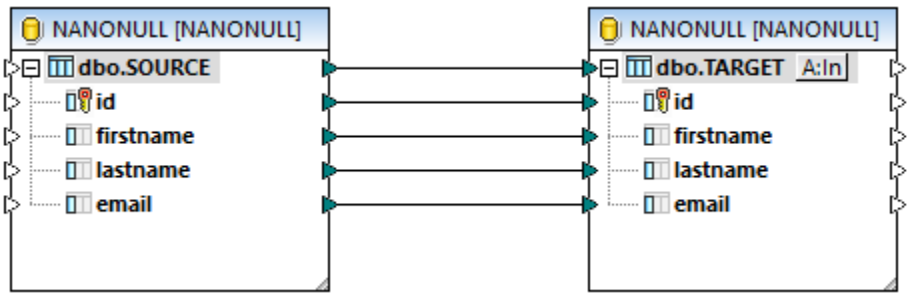
```
+----+-----------+-----------+--------------------------+
| id | firstname | lastname  | email                    |
+----+-----------+-----------+--------------------------+
| 1  | Mia       | Dahill    | NULL                     |
| 2  | Fred      | Weinstein | f.weinstein@nanonull.com |
+----+-----------+-----------+--------------------------+
```

TARGET テーブル

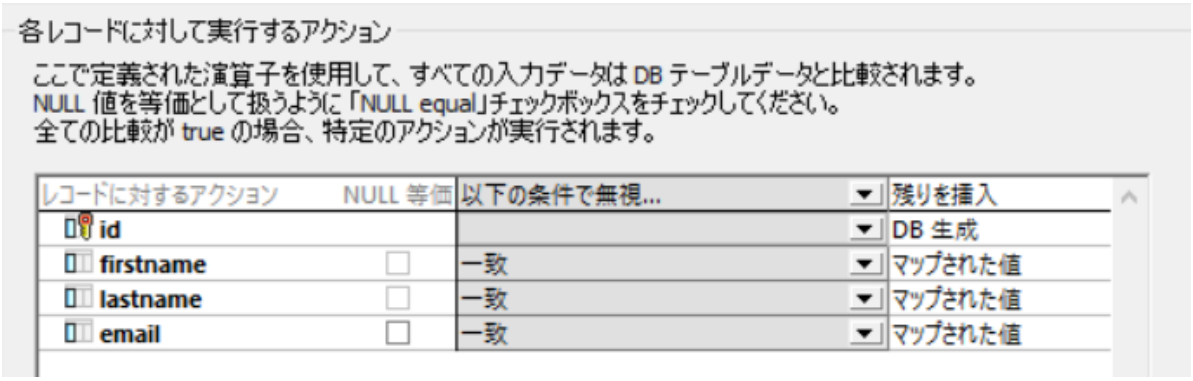
`SOURCE` テーブルからのデータを `TARGET` テーブルにマージすることが目的と仮定します。新規のレコードのみが `TARGET` テーブルに挿入されます(このサンプルでは、「Tobie Hughey」)。両方のテーブルに存在するレコード(「Mia Dahill」と「Fred W einstein」) は、無視されます。

タスクは以下のように達成することができます。

1. 「挿入」メニューから「データベース」を選択します。ウィザードステップに従い、データベースに接続します。(次を参照: データベースに接続)。データベースオブジェクトを追加するようにプロンプトされると、テーブル `SOURCE` を選択します。
2. 「挿入」メニューから「データベース」を選択します。データベースに再接続し、テーブル `TARGET` をマッピングに追加します。
3. ソースとターゲットコンポーネント間のマッピング接続を描きます。



1. **Action:Insert** A:In ボタンをクリックして、以下のようにデータベーステーブルアクションを構成します:



上記のとおり、「Ignore if.. Insert Rest」アクションの組み合わせが定義されます。この構成は、各レコードに対してマッピングが以下をチェックします:

- ソース内の`firstname` は、ターゲット内の`firstname` と等しいです。そして、
- ソース内の`lastname` は、ターゲット内の`lastname` と等しいです。そして
- ソース内の`email` は、ターゲット内の`email` と等しいです。

上のすべての条件が満たされる場合、(条件に応じて) レコードは無視されます。それ以外の場合、新規のレコードは、ターゲットテーブルに挿入されます。新規のレコードの`id` は、データベースにより生成されますが、他のフィールド (`firstname`、`lastname`、`email`) は、ソースからマップされた値を使用して表示されます。
`email` の横のチェックボックスは、このフィールドのためのNULL-対応の比較を有効化、または無効化することができます。`email` はNull の値を含むことができるため、このチェックボックスを選択する必要があります。(すなわち「Mia Dahill」にはNULL 電子メールアドレスが存在するからです)。チェックボックスの役割を確認するには、データベースを2 度更新してください: 最初はチェックボックスを選択し更新し、2 度目はチェックボックスのチェックが解除された状態で更新してください。

データベースを更新するには、**出力** タブをクリックして、メニューコマンド「**出力 | SQL-スクリプトの実行**」を実行します。

チェックボックスが選択されている場合、MapForce はNULL フィールドを等価として扱うように明示的に指定されています。ですから、レコード「Mia Dahill」はターゲットテーブルに挿入されません。これは期待されている結果です。

チェックボックスが選択されていない場合、希望する結果ではないレコード「Mia Dahill」が(このレコードは既存にもかかわらず)ターゲットテーブルに挿入されます。MapForce にNULL 値を等価として扱うように明示的な指定が与えられていないからです 。データベースに次のクエリを実行すると同様のシチュエーションが発生する可能性があります (NULL 値 が「=」 演算子と比較され、NULL 対応ではないためこのクエリは、レコードを抽出しません):

```
SELECT firstname, lastname, email FROM TARGET WHERE firstname = 'Mia' AND lastname =
'Dahill' AND email = NULL;
```

NULL に対応するには、上のクエリ以下のように書かれる必要があります:

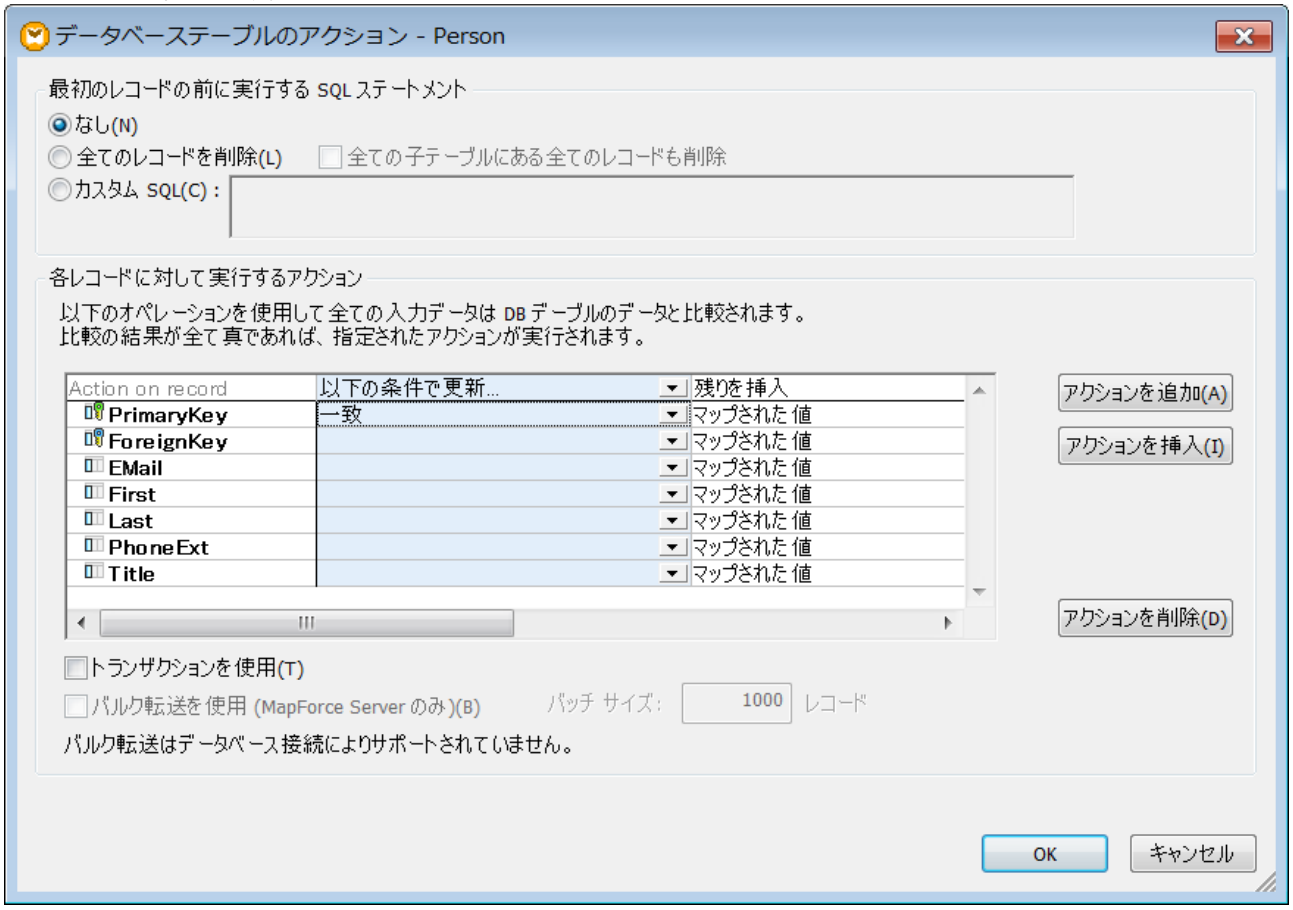
```
SELECT firstname, lastname, email FROM TARGET WHERE firstname = 'Mia' AND lastname =
'Dahill' AND email IS NULL;
```

メモ 上のクエリは、説明のためで、MapForce により生成される内部クエリの実際の構文を反映していません。(異なるデータベースベンダーはNULL 比較を異なるアプローチで処理するため) NULL 対応が有効化されると、MapForce は、データベース型に応じて生成されたクエリの構文を適応します。

(C) 2015-2021 Altova GmbH

6.2.3.12 データベーステーブルアクション設定

データベーステーブルにマッピング接続を作成する都度、データベースアクション ボタンが影響を受けるテーブルの横に表示されます。このボタンをクリックすると、データベースの挿入、更新、アクションの削除、とその他のオプションを構成することのできる、データベース テーブル アクション ダイアログボックスが開かれます。



データベース テーブル アクション ダイアログボックス

以下は、データベース テーブル アクション ダイアログボックスで使用することのできる設定の説明です。

最初のレコードの前に実行する SQL ステートメント

このグループボックスでは、最初のレコードがデータベースから取得される前に実行する SQL ステートメントを指定することができます。以下にあるラジオボタンから選択を行ってください:

- **無し**（デフォルトの設定）: アクションは取られることはありません。
- **全てのレコードを削除**: 「各レコードに対して実行するアクション」グループボックスにて定義されたテーブルアクションを実行する前に、選択されたテーブルにある全てのレコードを削除します。**全ての子テーブルにある全てのレコードも削除**チェックボックスを有効にすることで、選択されたテーブルにある子テーブル内にあるデータを取り除くことができます。例えば、次を参照してください: データを複数にリンクされているテーブルに挿入する
- **カスタム SQL**: テーブル全体を対象としたカスタム SQL ステートメントを記述することができます。

各レコードのために実行するアクション

設定は、マッピングが実行されるときにこのテーブルに対して取られるデータベースアクションを指定します。テーブルアクションを管理するには、「**アクションの追加**」をクリックし、「**アクションの挿入**」または「**アクションの削除**」ボタン. 複数のアクションを必要であれば定義することができます（例えば、「残りを挿入」アクションが後に続く「以下の条件で更新...」アクション。

定義されたテーブルアクションは、左から右に処理されます。上の例では、「以下の条件で更新...」アクションが最初に処理されます。更新の条件 が満たされない場合、次のアクションが処理されます（この例では、「残りを挿入」アクション）。以下の点に留意してください：

- 1つのアクションのすべての定義された条件が実行されるテーブルアクションのために満たされなければなりません。条件が全て満たされる場合、これらすべてのフィールドは、マッピング上でソースとターゲットアイテムの間の接続が存在する場所で更新されます。（条件が一致するアクションの右にある）その後のテーブルアクションは、そのレコードのために無視されます。
- 定義された条件が満たされない場合、テーブルアクションはスキップされ、次のアクションが（右に向かって）処理されます。
- 条件が満たされない場合、テーブルアクションは発生しません。

> 列の条件が入力アクションのために存在しないため、「全て挿入」または「残りを挿入」アクションの後に定義されているアクションは、実行されません。この場合、その後のテーブルアクション列が削除されることを通知するダイアログボックスが表示されます。

「NULL Equal」列内のレコードの横にあるチェックボックスにより、適用できる場合は、MapForce に明示的に、全ての列がNULL 値を持ちそのように扱われることを命令することができます（次を参照してください: データベーステーブルアクション内のNull の扱い）。

他のテーブルへの外部キーリレーションシップを持つマッピングを更新する場合、次のオプションを使用することができます:

子テーブル内のデータの削除	親テーブルのために「以下の条件で更新...」アクションを選択すると、このオプションは、役に立ちます。を選択すると、ソースファイル内のレコードの数がターゲットデータベース内のレコードの数と異なる場合、必要になる可能性があり、データベースを同期化（子テーブル内で孤立下データがない状態にしておくとよいかもしれません。以下も参照してください：親テーブルを更新する際の子テーブルのためのオプション.
入力子データを無視する	テーブルの子テーブルレコードに影響を及ぼすこと無く、ターゲット親テーブルを更新するためにこのオプションを使用することができます。以下も参照してください：親テーブルを更新する際の子テーブルのためのオプション.

アクションの多種の組み合わせを説明している例に関しては、以下を参照してください：

- データをテーブルに挿入する
- データを複数にリンクされているテーブルに挿入する
- テーブルの更新
- 親テーブルを更新する際の子テーブルのためのオプション
- 「以下の条件で更新し、残りを挿入」アクション
- 「以下の条件で削除...」アクション
- 「以下の条件で無視...」アクション

トランザクションの使用

この特定のテーブルアクションのためのデータベーストランザクション を有効化します。詳細に関しては、次を参照してください：トランザクションロールバックの使用。

(C) 2015-2021 Altova GmbH

バルク転送の使用

(複数のINSERT ステートメントを1つのクエリとしての)バルク転送を有効化します。マッピングがMapForce Server により実行されている場合およびデータベースアクションが「全て挿入」の場合、バルク転送はサポートされています。詳細に関しては、バルク挿入 (MapForce Server) を参照してください。

6.2.3.13 例: XML から SQLite へのデータマッピング

このサンプルでは、XML ファイルからデータを読み取り、SQLite データベースにデータを書き込むMapForce マッピングの作成に必要なステップについて説明されています。このサンプルにはサンプルマッピングデザイン(.mfd) ファイルが付属しています。このサンプルを開始する前に、サンプルファイルを確認するには、次のパスを使用してファイルを開いてください: **<マイドキュメント>\Altova\MapForce2021\MapForceExamples\Tutorial\XMLtoSQLite.mfd**。

XML ファイルからデータを SQLite データベースに挿入することがこのサンプルの目的です。サンプルのゴールを達成するために、データを書き込むことの出来る SQLite データベースが必要になります。このサンプルの目的を達成するために、データを書き込むことの出来る空のデータベースが必要になります。上で説明されているとおり、Altova DatabaseSpy または、公式な SQLite W eb サイトからダウンロード数rことの出来るコマンドラインシェルを使用して SQLite データベースを作成探索します。

SQLite データベースの作成:

(Altova MissionKit の一部、または スタンドアロンとして) DatabaseSpyis が使用中のコンピューターにインストールされている場合、新規 SQLite データベースを次のように作成することができます:

1. DatabaseSpy を起動します。
2. 「ファイル」メニューから、「データベースの接続の作成」をクリックします。
3. 「接続ウィザード」をクリックし、「SQLite」をクリックします。
4. 「新規 SQLite データベースの作成」をクリックし、**c:\sqlite\articles.sqlite** をパスとして入力し、「接続」をクリックします。
5. データソース名を設定するようにプロンプトされると、デフォルトの名前のままにします。
6. 新規 SQL エディターを開き(**Ctrl+N**) データベースに対して次のクエリを実行します:

```
create table articles (number smallint, name varchar(10), singleprice real);
```

それ以外の場合、下のステップに従いデータベースを作成してください:

1. SQLite ダウンロードページからW indows のための、SQLite コマンドラインシェルをダウンロードしてください。(http://www.sqlite.org/download.html)、そして、使用中のローカルマシンのディレクトリで .zip アーカイブを解凍してください(このサンプルのためには、**c:\sqlite** を使用してください)。
2. **c:\sqlite\sqlite3.exe** を実行して、次のステートメントを入力します:

```
create table articles (number smallint, name varchar(10), singleprice real);
```

これによりテーブル`articles` がインメモリデータベース内に作成されます。テーブル`articles` は3つの列により構成されています: *number*、*name*、および*singleprice*。この列の目的はXML ソーススキーマ内で定義された同じ名前を持つ要素からのデータを補完するためです。各列は、その列内に保管されるデータにふさわしいデータ方と共に宣言されています。

3. コマンドを実行します:

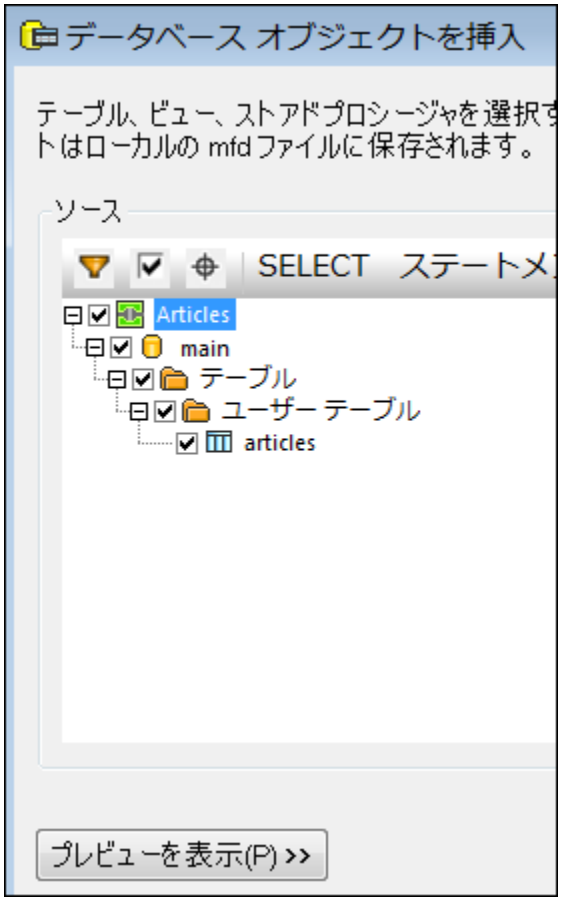
```
.save articles.sqlite
```

これによりインメモリのデータベースが現在の作業パスに保存されます: **c:\sqlite\articles.sqlite**。このパスを後のステップで参照する必要があることに注意してください。

このサンプルで必要とされる SQLite データベースが作成されました。

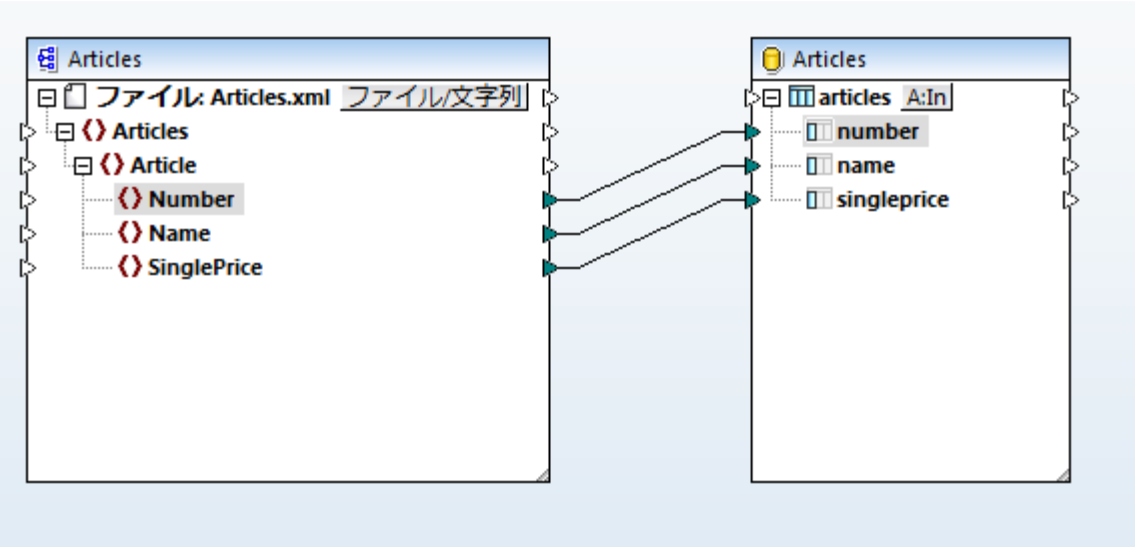
SQLite マッピングデザインに XML を作成する方法:

1. MapForce を起動し、(メニューコマンド「**出力 | ビルトイン実行エンジン**」を使用して)変換言語が BUILT-IN であることを確認してください。
2. **<Documents>\Altova\MapForce2021\MapForceExamples\Tutorial** フォルダー内にあるファイル **Articles.xml** を(「**挿入 | XML スキーマ/ファイルの挿入**」を使用して)マッピングエリアに追加します。
3. 前のステップ(メニューコマンド「**挿入 | データベースの挿入**」)で作成されたデータベース articles.sqlite をマッピングエリアに追加し、SQLite を選択します。
4. 「**接続**」をクリックします。データベースオブジェクトを選択するようにプロンプトされると、`articles` テーブルを選択します。

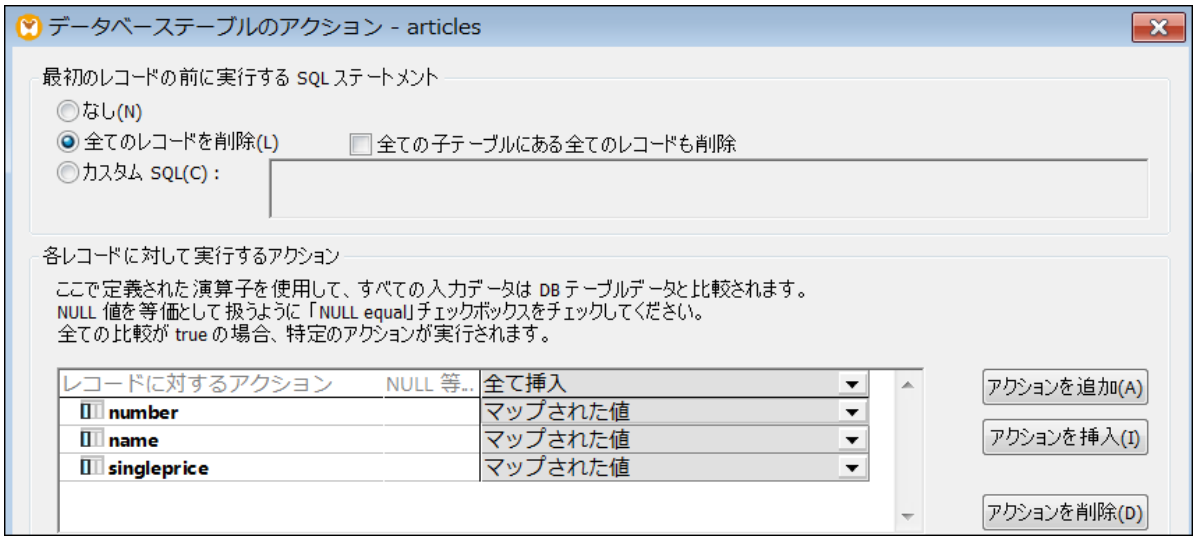


5. 下に示されるように接続を描きます:

(C) 2015-2021 Altova GmbH



6. データベースコンポーネント上の**A:In** ボタンをクリックし、「全てのレコードを削除する」オプションを選択します。これにより、マッピング実行される都度、重複を回避するために全ての既存の行が最初に削除されます。



7. メインマッピングウィンドウの「出力」タブをクリックします。MapForce は、内蔵の実行エンジンを使用してマッピングを実行し、出力ウィンドウ内にSQL クエリを表示します。

```
/*
The following SQL statements are only for preview and may not be executed in another SQL query tool!
To execute these statements use function "Run SQL-script" from menu "Output".
Connect to database using the following connection-string:
C:/sqlite/articles.sqlite
*/

PRAGMA foreign_keys = ON;

DELETE FROM "articles"

INSERT INTO "articles" ("number", "name", "singleprice") VALUES (1, 'T-Shirt', 25)

INSERT INTO "articles" ("number", "name", "singleprice") VALUES (2, 'Socks', 2.3)

INSERT INTO "articles" ("number", "name", "singleprice") VALUES (3, 'Pants', 34)

INSERT INTO "articles" ("number", "name", "singleprice") VALUES (4, 'Jacket', 57.5)

```

8. （メニューコマンド「出力 | SQL-スクリプトの実行」）SQL スクリプトを実行し、データベースを作成します。MapForce がランタイムエラーに遭遇しない場合、レコードは SQLite データベースに挿入されます。

```
/*
The following SQL statements were executed during "Generate output" function.
Every single result is written right to the "-->>>" string.
These statements are only for preview and may not be executed in another SQL query tool!
The database was connected using the following connection-string:
C:/sqlite/articles.sqlite
*/

PRAGMA foreign_keys = ON;
-->>> OK. 0 row(s).

DELETE FROM "articles"
-->>> OK. 0 row(s).

INSERT INTO "articles" ("number", "name", "singleprice") VALUES (1, 'T-Shirt', 25)
-->>> OK. 1 row(s).

INSERT INTO "articles" ("number", "name", "singleprice") VALUES (2, 'Socks', 2.3)
-->>> OK. 1 row(s).

INSERT INTO "articles" ("number", "name", "singleprice") VALUES (3, 'Pants', 34)
-->>> OK. 1 row(s).

INSERT INTO "articles" ("number", "name", "singleprice") VALUES (4, 'Jacket', 57.5)
-->>> OK. 1 row(s).

```

SQLite データベースにデータが正確に挿入されているかチェックする方法:

1. MapForce の「DB クエリ」タブをクリックします。
2. 「DB クエリ」ペインの上にあるドロップダウンリストから **articles** エントリを選択します。
3. SQL エディター内に次のクエリを入力します:

```
select * from articles;
```


(C) 2015-2021 Altova GmbH

または、次のステップをおこなってください:

1. ファイル **c:\sqlite\sqlite3.exe** を実行し、コマンドを使用してデータベースを開きます:

```
.open articles.sqlite
```

2. 次の Select ステートメントを実行します:

```
select * from articles;
```

テーブルデータの選択のアプローチに関わらず、(MapForce または SQLite シェル)、クエリは4つの行を返します。これは、このサンプルの目的であるソース XML ファイル内のレコードの数に対応しています。

6.2.4 データベースのデータのジョイン

データベースからデータを読み取るマッピング内では、データベースオブジェクト テーブルまたはビューなどを、マッピングにジョインコンポーネントを追加することによりジョインすることができます。例えば、データがリレーショナルデータベース内に保管される外部キーリレーションシップによりバウンドされた2つ、または複数のテーブルからのデータを組み合わせることができます。INNER JOIN (または適用可能な場合 LEFT JOIN) オペレーションを使用して2つまたは複数のテーブルがジョインされた箇所にデータベース SQL クエリを実行する結果と同じ結果になります。

ジョインコンポーネントに接続されているデータの型により、ジョインの操作は、標準 (非-SQL) モード、またはSQL モード内で行うことができます。非-SQL モード内のジョインは、マッピングエンジンにより実行され、SQL モード内のジョインは、マッピングがデータを読み取るデータベースにより実行されます。

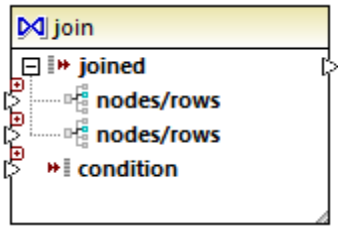
非-SQL モード内のジョインは、コンポーネント型を入力としてサポートするため柔軟性に富んでいます (例えば、ジョインは異なるデータベースからのテーブル間、またはXML 構造とデータベーステーブル間であることができます)。非-SQL ジョインのサンプルに関しては例: XML 構造のジョイン内で詳しく説明されています。。その一方、非-SQL ジョインの実行は、マッピングエンジンにより更にメモリを必要とする実行となります。(これは、比較の総数がジョインされた構造のクロスジョイン、またはデカルト積を表すためです)。通常、データを中心としたマッピング内では、この過程はとても速く、影響が小さいです。しかしながら、ジョインしたデータソースが大量の記録により構成される場合、マッピングは実行するために時間を要します。マッピングが多数のレコードを処理する場合、マッピング実行をスピードアップするジョイン最適化に特化した機能を含む MapForce Server Advanced Edition にライセンス供与することを考慮してください。

SQL モード内のジョインは、使用することのできる (テーブルまたはビューなどの) データベースオブジェクトを入力として受け入れます。しかしながら、データベースによりネイティブにより良いマッピングを行うため、柔軟性のある 非-SQL ジョインとしては受け入れません。詳細に関しては、SQL モード内のジョインを参照してください。

メモ ジョインコンポーネントの使用のみがデータベーステーブル、またはビューをジョインする唯一の方法ではありません。データベースへのジョインの適用はSQL SELECT ステートメントを使用して行われます データベースデータの選択 (SQL SELECT) を参照してください。SQL SELECT ステートメントのジョインコンポーネント主要な違いは、前者は手動で作成され、そのために柔軟性があります。しかしながら、SQL ステートメントを手動で作成を希望しない場合、ジョインコンポーネントは簡単な代替方法です。

ジョインコンポーネントを追加する方法:

1. マッピング 変換言語を BUILT-IN に設定します (これを行うには ツールバーボタンをクリックします、または [出力 | ビルトイン実行エンジン] メニューコマンドを使用します)。
2. [挿入] メニューから [ジョイン] をクリックします。または [ジョイン] ツールバーボタンをクリックします。ジョインコンポーネントがマッピングに表示されます。デフォルトでは2つの構造からデータを受け入れるため、2つの `nodes/rows` 入力が存在します。必要であれば [**入力の追加**] ( ) ボタンをクリックして、ジョインに新しい入力を追加することができます。3つ以上の構造をジョインするを参照してください。



3. ジョインコンポーネントの`nodes/rows` アイテムとジョインされる構造を選択します。
4. ジョイン(または複数の条件)のための条件を追加してください。これを行うには、ジョインコンポーネントを右クリックしプロパティを選択します。ジョインの条件は、関数のブール値の結果をジョインコンポーネントの`condition` アイテムに接続することにより、マッピングから直接追加することができます。データベースがジョインされる特定の場合、ジョインの条件は自動的にMapForce により作成されます。詳細に関しては、ジョインの条件の追加を参照してください。

メモ

- ジョインコンポーネントは、BUILT-IN 実行エンジン内でのみサポートされています。C#、C++、またはJava 内でのコードの生成はサポートされていません。
- 構造がサポートされていない、またはジョインのための入力ソースによりサポートされて以内場合、MapForce マッピング上直接ヒントを表示するか、またはマッピングを検証する際に、メッセージウィンドウ内にヒントを表示します。(マッピングの検証参を照してください)。
- ジョインコンポーネントをインラインユーザー定義関数の結果または入力パラメーターに接続しないでください。このような接続が存在すると、マッピングの検証中に検証エラーが発生します。
- 使用することのできるデータベースコンポーネント(テーブルまたはビューなどの)を直接ジョインコンポーネントに接続すると、SQL モード SQL ボタンが自動的にジョインコンポーネントの右上に表示されます。有効化されると、このボタンは、ジョイン操作に適用することのできる特別な SQL 機能を提供します(SQL モード内のジョインを参照してください)。
- ジョインされたアイテムの出力を他のジョインコンポーネントに接続することはできません。しかしながら、ジョインの結果の一部を他のジョインに接続することは可能です。

6.2.4.1 SQL モード内のジョイン

使用することのできるデータベースコンポーネント(テーブルまたはビューなどの)を直接ジョインコンポーネントに接続すると、SQL モード SQL ボタンがジョインコンポーネントの右上に表示されます。SQL モードが有効化されていると、ジョインオペレーションは、マッピングがデータを読み込む箇所でデータベースにより行われます。すなわち、MapForce は、ジョインに使用される全てのテーブルからのデータを選択し結合するために適切な SQL 構文を使用して、内部でデータベースにクエリを送信します。重要な点は、SQL を書く必要がない点です。下のサンプルで示されている通りジョインコンポーネントがマッピング上でデザインされた方法をベースにして必要なくクエリが生成されます。

SQL モードを使用可能にするためには、以下の条件が満たされている必要があります:

1. ジョインされる両方のオブジェクト(テーブルまたはビュー)は、同じデータベースから与えられなければなりません。
2. ジョインされる両方のオブジェクトは、同じ MapForce コンポーネントから与えなければなりません。(コンポーネント内でデータベースオブジェクトを追加および削除する方法は以下のとおりです: データベースコンポーネントを右クリックして、コンテキストメニューから**追加/削除/編集** データベースオブジェクトを選択します)。
3. ジョインの条件(または条件)は、マッピング上ではなく、コンポーネント プロパティからのみ定義されます(ジョインコンポーネントのヘッダーを右クリックして、**プロパティ** を選択します)(ジョインの条件の追加を参照してください)。

メモ SQL モード内でデータベーステーブルがジョインされると、MapForce は、テーブル間で検出された外部キーリレーションシップをベースにして、ジョインの条件を自動的に作成します。自動的なジョインの条件が起こるには、データベーステーブルがMapForce コンポーネント上で親子関係内に存在する必要があります(すなわち、1つのテーブルが「親」または「子」で、他のテーブルがコンポーネント上に存在する必要があります)。例: SQL モード内のテーブルのジョインを参照してください。

4. すべてのデータベーステーブルは、現在のターゲットコンテキスト内にまだ存在することはできません。ジョインの結果がターゲットコンポーネント内で使用される場合、ジョインされたテーブルは直接または間接的にターゲットの親ノードに接続されている場合があります。マッピングの実行方法に関しては[マッピングルールと戦略]を参照してください。

ジョインコンポーネントの右上にある **SQL** (SQL) ボタンを使用して、SQL モードを閲覧または管理することができます:

SQL SQL モードが無効化されています。(ジョインは MapForce (または適用可能な場合 MapForce Server)により実行されます。

SQL SQL モードが有効化されています (ジョインはデータベースにより実行されます)。

SQL ボタンが不足している場合、SQL モードは意味を成さないことを示します 、または ジョインされたデータのためにサポートされていないことを意味します。

特定のケースの場合、SQL モードは明示的に無効化されている必要があります SQL)、例:

- マッピングがジョインコンポーネントプロパティの外側のジョイン条件を必要な場合 (すなわち、マッピング上で定義されている条件がジョインコンポーネントの `condition` アイテムに接続されている場合)。
- 異なるデータベースからのテーブルをジョインする場合。異なるデータベースからテーブルをジョインする場合、標準 (非-SQL) ジョインを使用します。

ジョインモードの変換

SQL モード SQL 内でコンポーネントをジョインする場合、データベーステーブルをジョインまたは以下の方法の 1 つでビューすることができます:

- INNER JOIN – 両方の入力セット内の条件を満たすレコードのみがジョインコンポーネントにより返されます。
- LEFT OUTER JOIN – [左端]のテーブル空のすべてのレコードを含むジョインコンポーネント (MapForce 内では、これはジョインコンポーネントの一番上のテーブルです) とジョイン条件を満たす後にジョインされたテーブルからのレコード。

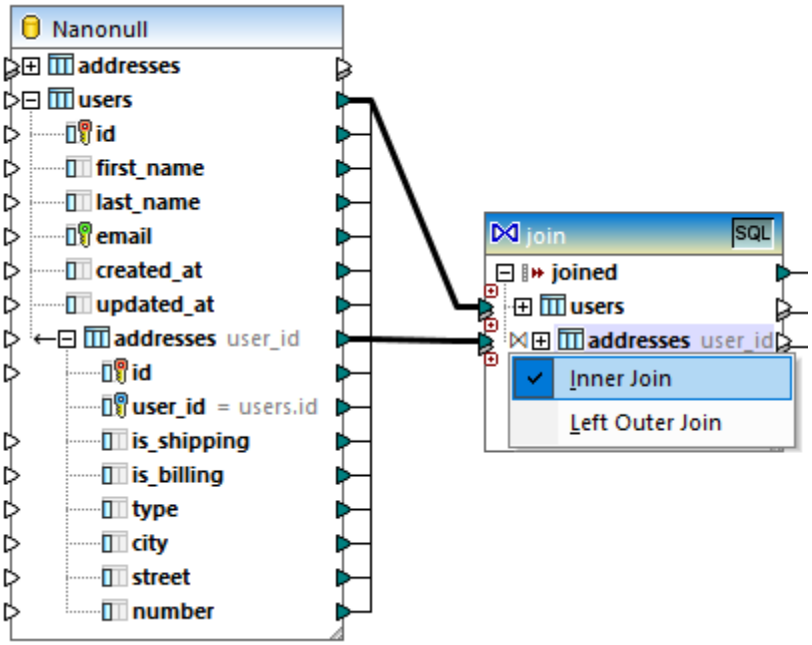
テーブルのジョインモードをビュー、またはジョインコンポーネント上でビューするには ジョインされたテーブルまたはビューの前に表示されるアイコンに着目してください。以下のアイコンの一つが最初のものを除いてジョインされたテーブルまたはビューのために表示されます:

- 内部結合 ⋈
- 左結合 ⋈

ジョインに関する詳細のヒントを表示するには、アイコンにカーソルをポイントします:

ジョインモードを変更するには、次のいずれかを行ってください:

- 各ジョインされたテーブルまたはビューの前の **内部結合** ⋈ または **左結合** ⋈ アイコンをクリックし **内部結合** または **左外側結合** をコンテキストメニューから選択します。
- ジョインコンポーネントの2番目 (または3番目、4番目)のジョインされたテーブルまたはビューを右クリックし **型のジョイン | 内部結合** または **型の結合 | 左外側結合** をコンテキストメニューから選択します。



以下の点に注意してください：

- LEFT OUTER JOIN にジョインモードを変更すると、上のテーブルまたはビューはジョインの「左側」を表します。
- ジョインモードの変更はINNER JOIN またはLEFT JOIN はデータベース内のSQL クエリに影響する同様にジョインコンポーネントにより返されたデータに影響を与えます。

エイリアス名

ジョインされたデータベーステーブル、またはビューに同一のフィールド名が含まれていることが頻繁にあります。例えば、SQL モードが有効化されると、このようなアイテムは、キーワード「AS」によりプレフィックスされたコンポーネント上に表示されます。例えば、2つのジョインされたテーブルが「id」フィールドを含む場合、このフィールドは、最初にジョインされたテーブル上で「id」として表示されます。そして、2番目にジョインされたテーブルでは「id AS id2」として表示されます。ジョインされたテーブルは、例えば、同じテーブルがテーブル内でジョインされた場合、エイリアス名を生成することができます。

マッピング上で後に参照する場合エイリアス フィールドまたはテーブル名は重要です。例えば、ジョインコンポーネントの出力がSQL W HERE/ORDER コンポーネントに接続されている場合などです。目的を達成するには、ジョインコンポーネントの出力を、SQL W HERE とORDER BY 句を入力するSQL W HERE/ORDER コンポーネントに接続することができます。

W HERE 句からフィールドを参照するには、フィールド名が後に続き、ピリオド（.）文字が後に続く、テーブル名を書き込みます。テーブルエイリアスを参照するには、ジョインコンポーネントに表示される順のエイリアス名を使用します。ORDER BY 句内では、同じテクニック（`table.field`）、または、（「AS」の後に表示される名前である）エイリアスフィールド名を書き込みます。

SQL W HERE/ORDER 句を使用するマッピングのサンプルに関しては、サンプル: SQL モード内のジョインテーブルを参照してください。

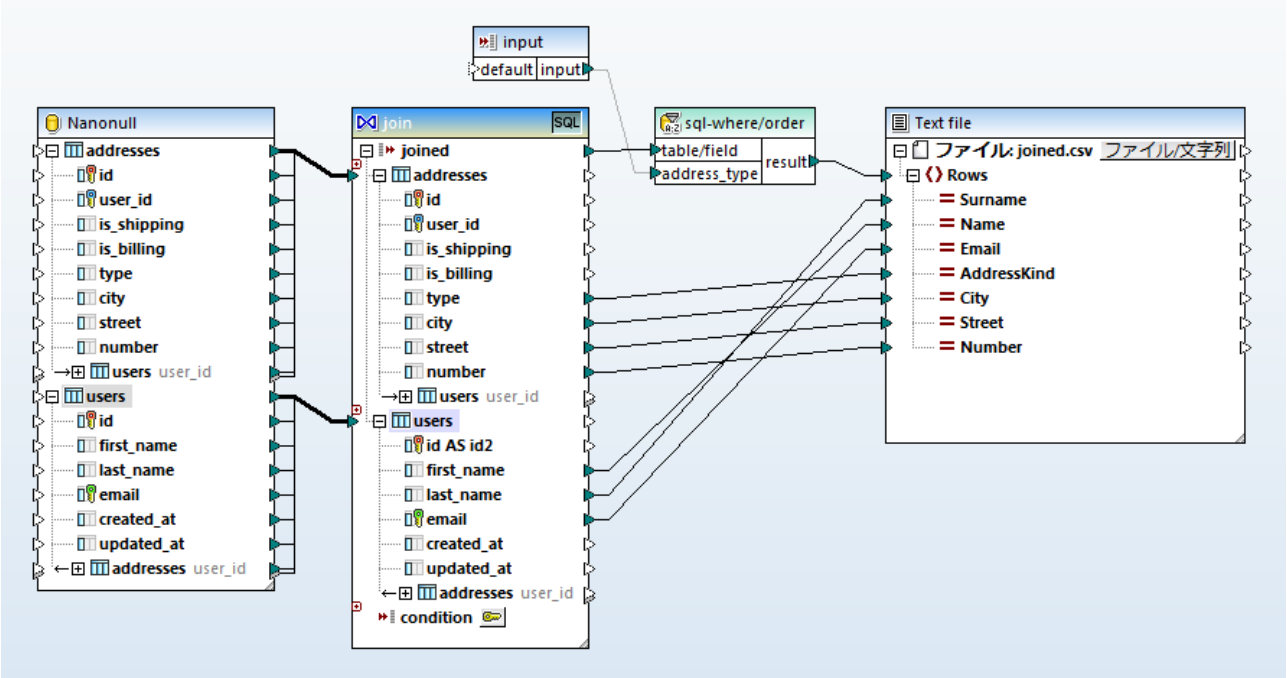
メモ データベーステーブルとジョインコンポーネント間でSQL W HERE/ORDER コンポーネントは許可されていません。ジョインコンポーネントの（前ではなく）後に追加することができます。SQL W HERE/ORDER コンポーネントの詳細に関しては、データベースデータのフィルターと並べ替え（SQL W HERE/ORDER）を参照してください。

(C) 2015-2021 Altova GmbH

6.2.4.2 例: SQL モード内でテーブルをジョインする

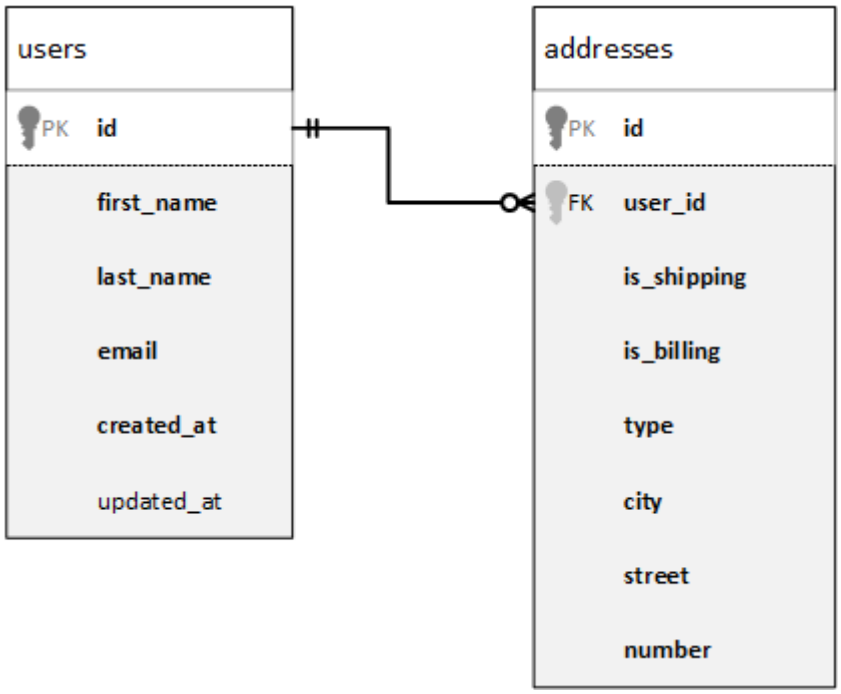
このサンプルは、MapForce ジョイン コンポーネントを使用した、データベーステーブルからのデータをジョインする方法について説明しています。ジョインの操作は、SQL モード内のジョインで説明されているように、SQL モード内で実行されます。3つ、または以上のテーブルワークを同様にジョインする場合も同様に行います。例: CSV レポートを複数のテーブルから作成するを参照してください。

このサンプルは、マッピングサンプルが伴い、次のパスで検索することができます: **<マイドキュメント>\Altova\MapForce2021\MapForceExamples\Tutorial\JoinDatabaseTables.mfd**.



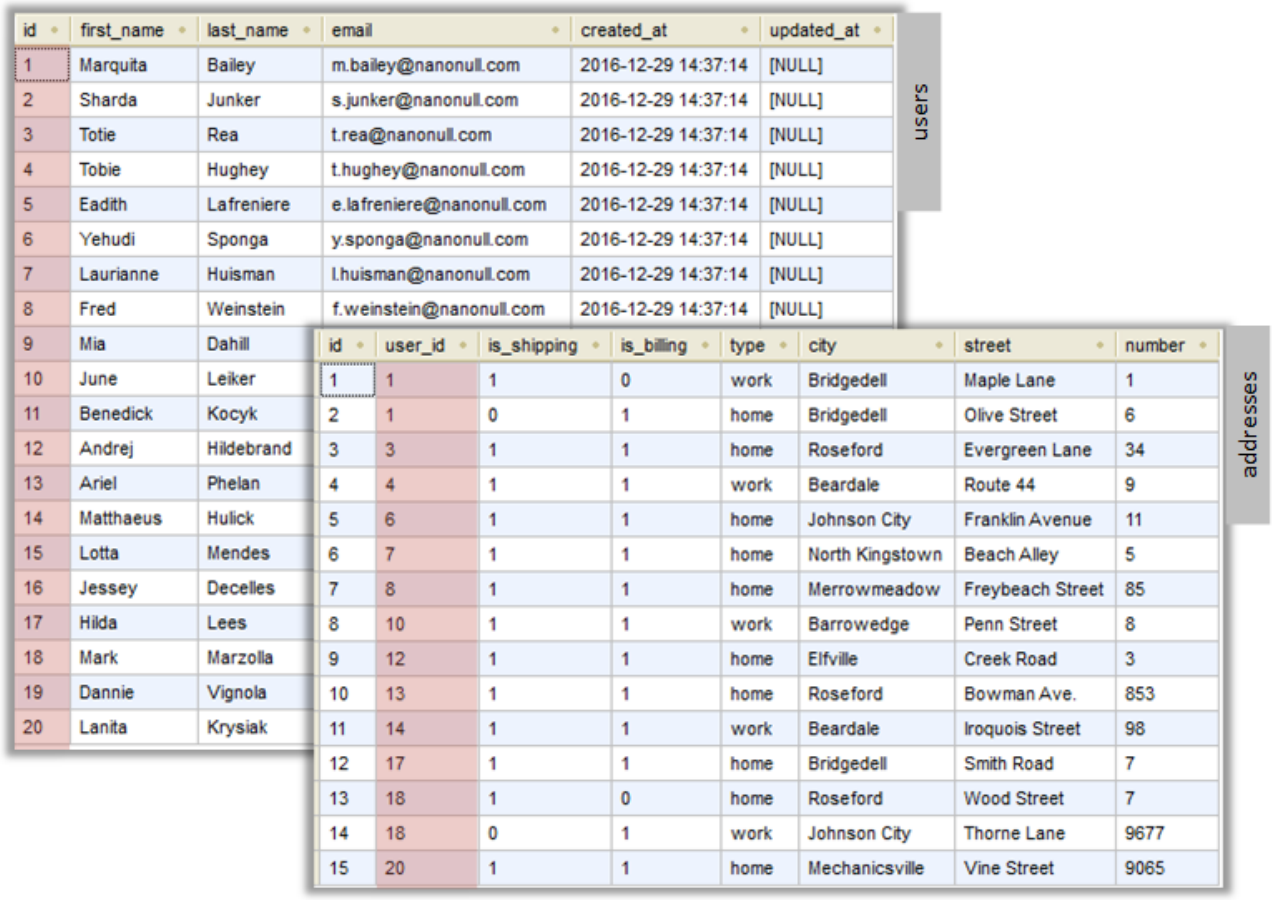
JoinDatabaseTables.mfd

上のマッピングの目的は、2つのソースデータベーステーブルからのデータを単一のターゲット CSV ファイルに結合することです。下のデータベースダイアグラムに示されているとおり、最初のテーブル(`users`)は個人の住所を保管し、2 番目のテーブル(`addresses`)は個人の名前と電子メールアドレスを保管します。2つのテーブルは、共有のフィールドによりリンクされています(`users` 内の`id` が`addresses` 内の`user_id` に対応しています)。データベース用語では、この種類おリレーションは[外部キーリレーションシップ]と呼ばれています。



簡素化のために、下のイメージは両方のテーブルの実際のデータを表示しています。

(C) 2015-2021 Altova GmbH



users

id	first_name	last_name	email	created_at	updated_at
1	Marquita	Bailey	m.bailey@nanonull.com	2016-12-29 14:37:14	[NULL]
2	Sharda	Junker	s.junker@nanonull.com	2016-12-29 14:37:14	[NULL]
3	Totie	Rea	t.rea@nanonull.com	2016-12-29 14:37:14	[NULL]
4	Tobie	Hughey	t.hughey@nanonull.com	2016-12-29 14:37:14	[NULL]
5	Eadith	Lafreniere	e.lafreniere@nanonull.com	2016-12-29 14:37:14	[NULL]
6	Yehudi	Sponga	y.sponga@nanonull.com	2016-12-29 14:37:14	[NULL]
7	Laurianne	Huisman	l.huisman@nanonull.com	2016-12-29 14:37:14	[NULL]
8	Fred	Weinstein	f.weinstein@nanonull.com	2016-12-29 14:37:14	[NULL]
9	Mia	Dahill			
10	June	Leiker			
11	Benedick	Kocyk			
12	Andrej	Hildebrand			
13	Ariel	Phelan			
14	Matthaeus	Hulick			
15	Lotta	Mendes			
16	Jessey	Decelles			
17	Hilda	Lees			
18	Mark	Marzolla			
19	Dannie	Vignola			
20	Lanita	Krysiak			

addresses

id	user_id	is_shipping	is_billing	type	city	street	number
1	1	1	0	work	Bridgedell	Maple Lane	1
2	1	0	1	home	Bridgedell	Olive Street	6
3	3	1	1	home	Roseford	Evergreen Lane	34
4	4	1	1	work	Beardale	Route 44	9
5	6	1	1	home	Johnson City	Franklin Avenue	11
6	7	1	1	home	North Kingstown	Beach Alley	5
7	8	1	1	home	Merrowmeadow	Freybeach Street	85
8	10	1	1	work	Barrowedge	Penn Street	8
9	12	1	1	home	Elfville	Creek Road	3
10	13	1	1	home	Roseford	Bowman Ave.	853
11	14	1	1	work	Beardale	Iroquois Street	98
12	17	1	1	home	Bridgedell	Smith Road	7
13	18	1	0	home	Roseford	Wood Street	7
14	18	0	1	work	Johnson City	Thorne Lane	9677
15	20	1	1	home	Mechanicsville	Vine Street	9065

`users` テーブル内の各ユーザーはゼロの値、または `addresses` テーブル内に複数の住所を有することができます。例えば、ユーザーは、「home」の種類に属する1つの住所、または2つの住所（「home」と「work」の2種類の住所）、または住所なしを選ぶことができます。

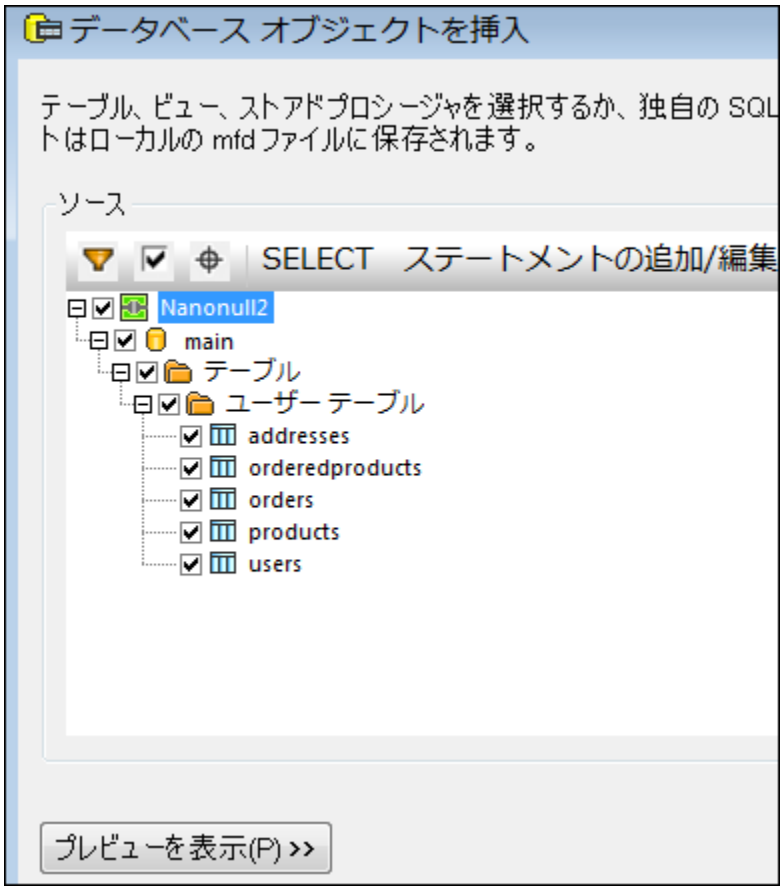
マッピングの目的は住所テーブル内に住所が少なくとも1つ記入されている全てのユーザーのデータ（名、姓、電子メール、住所など）を抽出することです。また、特定の種類の住所を簡単に抽出することもできます（例えば、自宅の住所、または勤務先の住所のみを抽出することができます）。（「home」、または「work」）などの抽出する住所は、マッピングにパラメーターとして与えられます。抽出された個人のレコードは姓によりアルファベット順に並べ替えられます。

下で説明されているとおり、ジョインコンポーネントを使用して、マッピングの必要条件が満たされます。

メモ ジョインコンポーネントの使用のみがデータベーステーブル、またはビューをジョインする唯一の方法ではありません。データベースへのジョインの適用はSQL SELECT ステートメントを使用して行われます [データベースデータの選択（SQL SELECT）](#) を参照してください。SQL SELECT ステートメントのジョインコンポーネント主要な違いは、前者は手動で作成され、そのために柔軟性があります。しかしながら、SQL ステートメントを手動で作成を希望しない場合、ジョインコンポーネントは簡単な代替方法です。

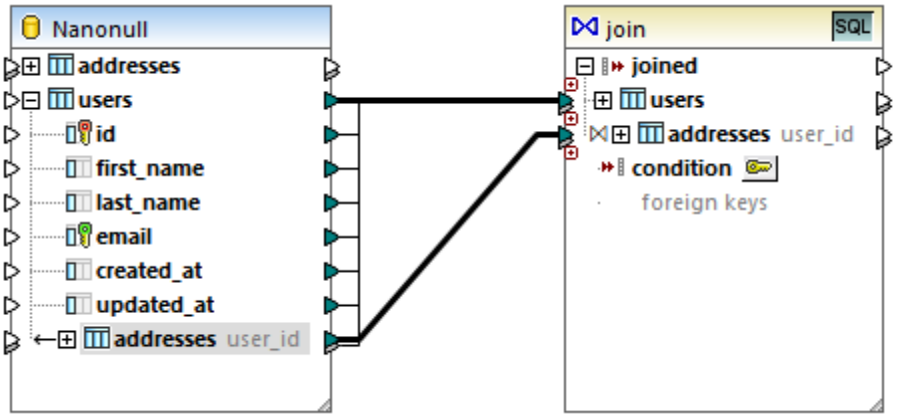
ステップ 1: ソースデータベースを追加する

1. [挿入] メニューから、[データベース]をクリックしください。（または、「**データベースの挿入**」 ツールバーボタンをクリックします。）
2. [SQLite] をデータベースの型として選択し、[次へ]をクリックします。
3. フォルダー内で使用することのできる **Nanonull.sqlite** ファイルを参照します。<**マイドキュメント >\Altova\MapForce2021\MapForceExamples\Tutorial**>, を選択し[**クリック**]をクリックします。
4. プロンプトされると、 `addresses` と `users` テーブルを選択します。



ステップ 2: ジョインコンポーネントを追加する

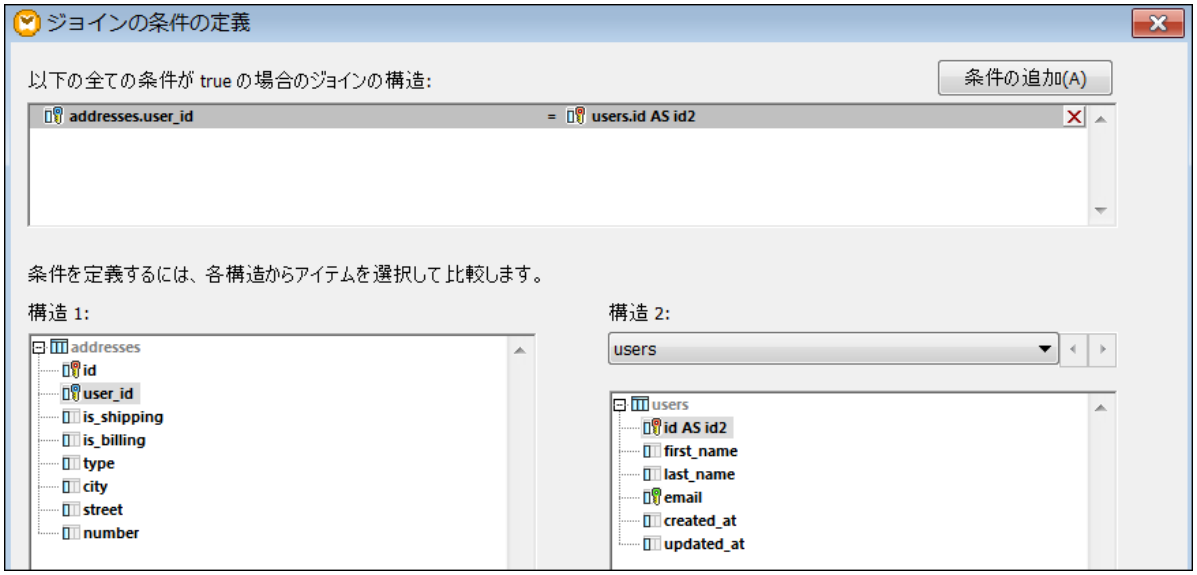
1. [挿入] メニューから [ジョイン] をクリックします。(または [ジョイン] ツールバーボタンをクリックします)。
2. `users` テーブルからジョインコンポーネントの最初の入力に接続線を描きます。
3. `users` テーブルを展開し、`addresses` テーブル (`users` の子) からジョインコンポーネントの2番目の入力に接続を描きます。
   ボタンにより必要に応じて更にテーブルを追加することができます。しかしながら、このサンプルでは、2つのテーブルのみがジョインされます。



(C) 2015-2021 Altova GmbH

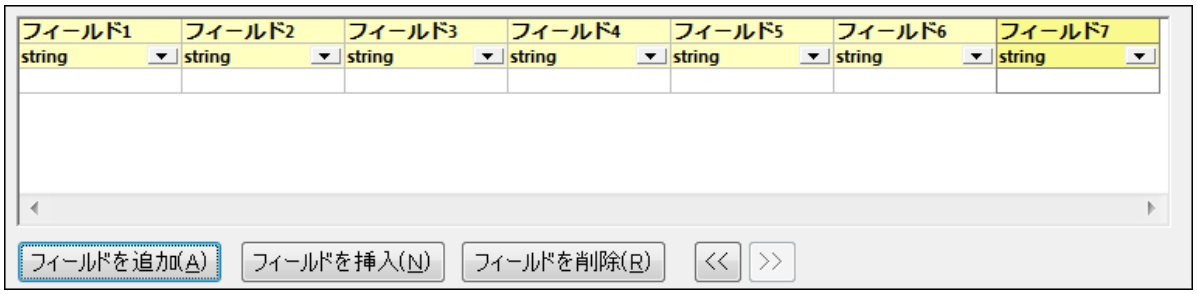
メモ　addresses テーブル(users の子ではない) から直接接続を追加することができます。しかしながら、この場合、ジョイン条件の追加で説明されているとおり、ジョインの条件は手動で定義されなければなりません。このサンプルの目的のために、上に示されるように接続を作成してください。これにより、必要なジョインの条件が自動的に作成されます。

4. ジョインコンポーネント上で使用することのできる[ジョインの条件の定義] ボタンをクリックします。ジョインの条件が自動的に作成される事に注意してください(users.id = addresses.user_id)。

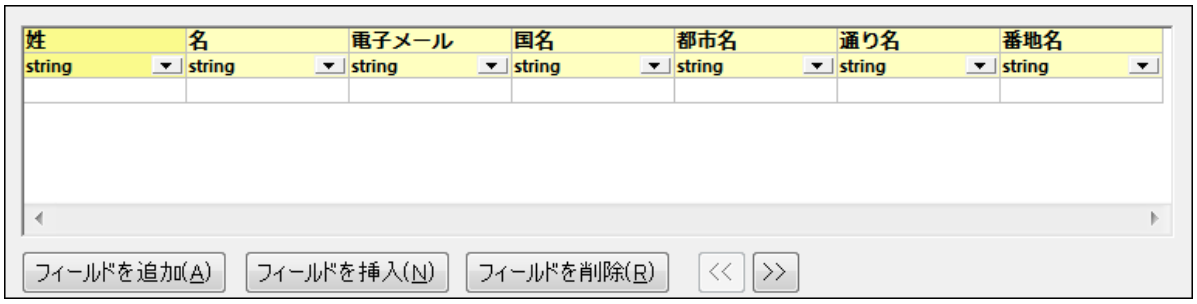


ステップ 3: ターゲット CSV コンポーネントの追加

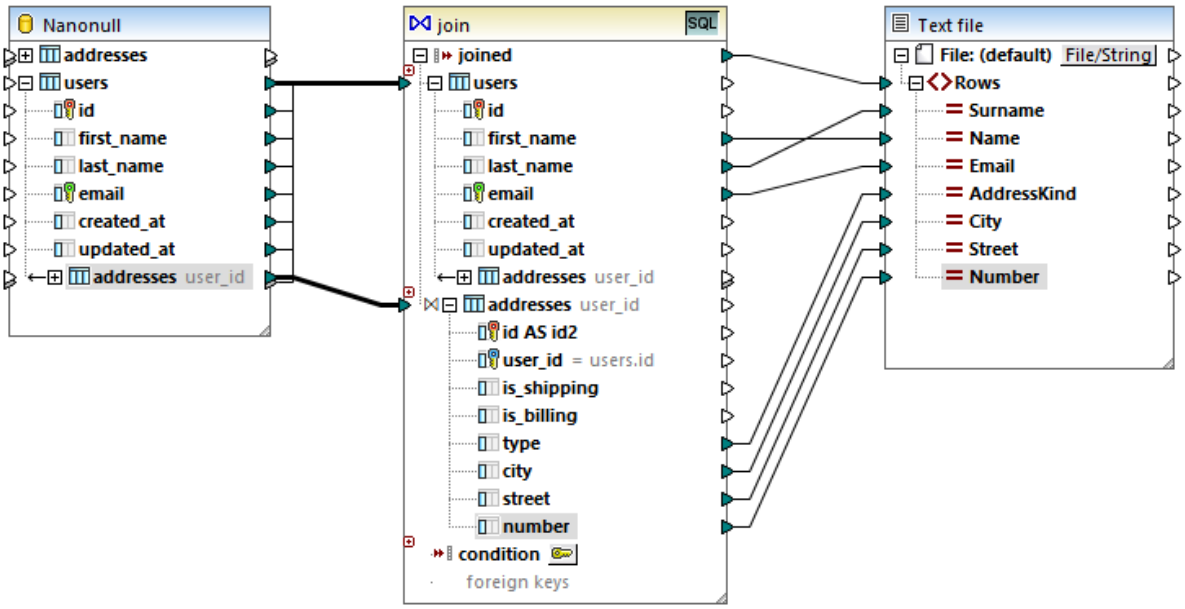
1. 「挿入」メニューから「テキストファイル」をクリックします。(または、[テキストファイルの挿入] ツールバーボタンをクリックします。)
2. テキストの処理モードを選択されるように問われると、標準 CSVのために簡単処理を使用する... を選択します。。
3. [フィールドの追加] を数回クリックして、7つのCSV フィールドを作成します。他の設定をそのままにします。



4. 各フィールドのタイトルのセルをダブルクリックして、詳細を説明する名前を与えます(これによりマッピングを簡単に読み込むことができます)。



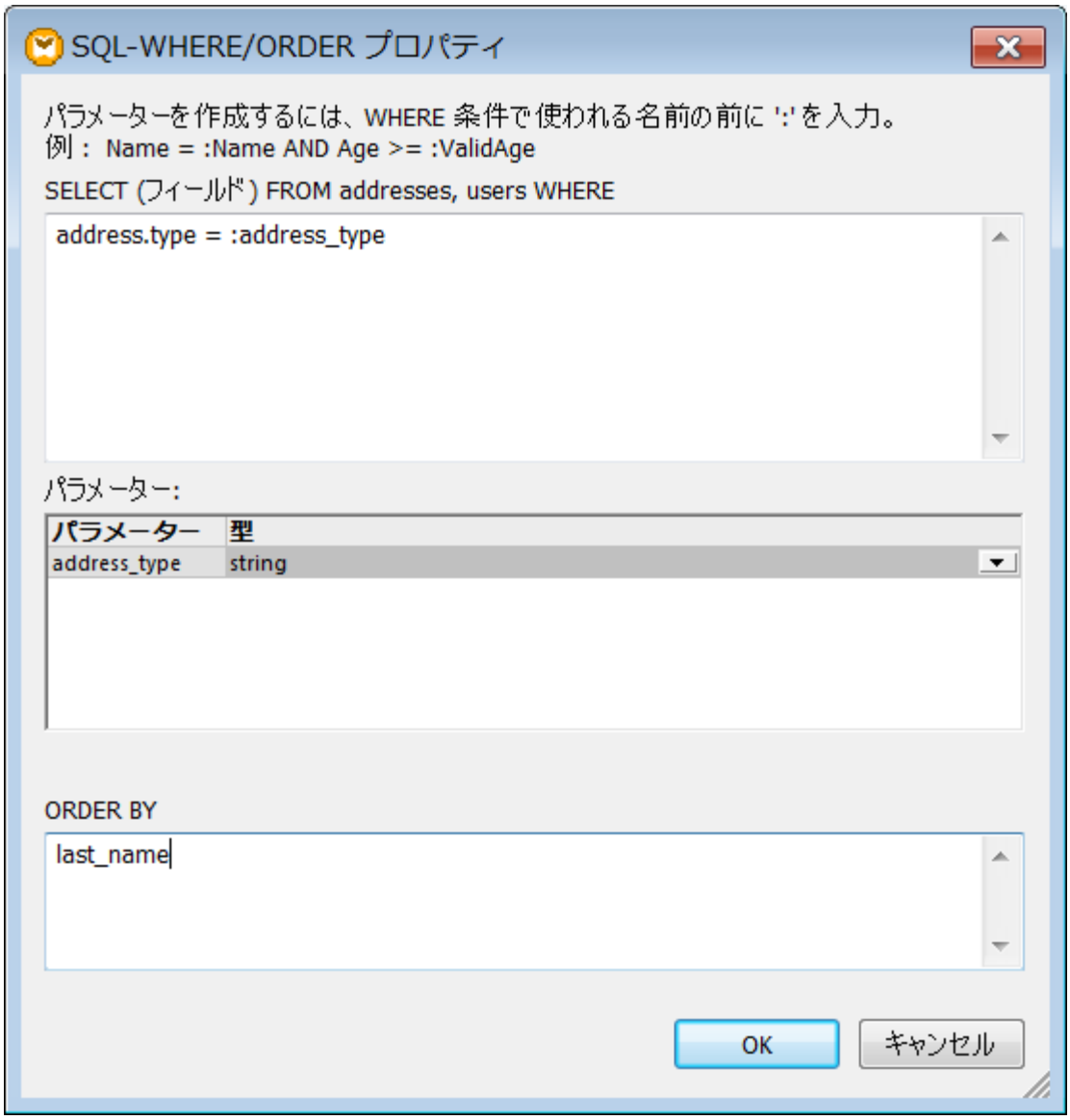
5. ジョインコンポーネント とCSV コンポーネント間の接続を下に示されるように描きます。ジョインコンポーネントの `joined` アイテムとターゲットコンポーネントの `Rows` 間の接続はジョイン条件を満たす数量のレコードを(行) をターゲット内に作成することを意味します。



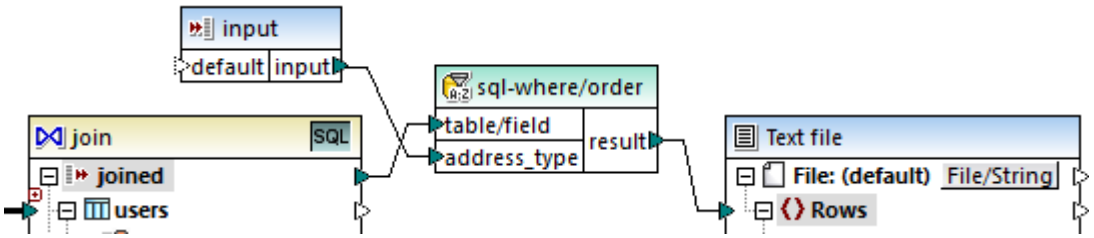
ステップ 4: SQL WHERE/ORDER 条件と入力パラメーターを入力する

1. ジョインコンポーネントの `joined` アイテムとターゲットコンポーネントの `Rows` アイテム間の接続を右クリックし、**[SQL-WHERE/ORDER を挿入する]** を選択します。
2. WHERE とORDER BY 句 を下に示されるように入力します。

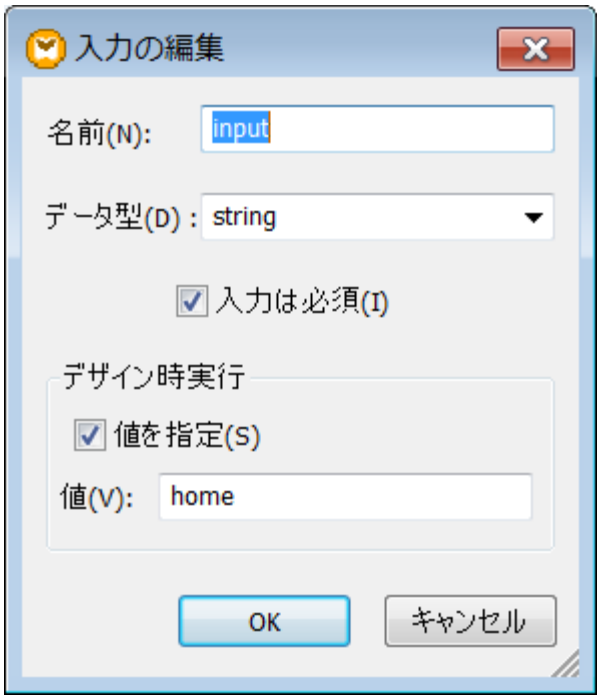
(C) 2015-2021 Altova GmbH



3. マッピング上で、（[挿入 | 入力の挿入] メニューコマンドを使用して）入力コンポーネントを追加し、前のステップで作成された `address_type` パラメーターに出力を接続します。



4. 入力コンポーネントをダブルクリックして、下に示されるように構成します。MapForce 内でマッピング出力をプレビューするためにデザイン時刻 値(この場合、「home」)が必要とされます。職場の住所を抽出するためにプレビューを行うには、この値を [work] と置き換えてください。



マッピングの説明

ステップ 2 は定義されているジョインの条件 `users.id = addresses.user_id` を満たすレコードのみがターゲットにコピーされることを確認します。2 つのテーブルは外部キーリレーションシップにより バインドされており、マッピング接続は状況に応じて描かれているためジョインの条件は自動的に追加されています。テーブルリレーションシップの詳細に関しては、データベースリレーションシップの処理 を参照してください。このサンプルは既存のテーブルリレーションシップを活用しているため、ジョイン条件を手動で定義する必要がありません。ジョイン条件の定義を手動で行う方法に関するサンプルは 例: CSV レポートを複数のテーブルから作成するを参照してください。

2 つのソーステーブルは、同じデータベースおよび同じコンポーネントからであり、このジョインは SQL SQL モード を活用することができます。SQL モードが有効化されているため、ジョインの操作は MapForce* ではなく、データベースにより行われます。SELECT ステートメントが代わりに作成され、データベースに送信されます。ジョイン (INNER JOIN) の型はジョインコンポーネント上の **addresses** テーブルの前にある **内部結合** ⋈ アイコンにより表示されています。ジョインモードの変更で説明される通りジョインの型を LEFT OUTER JOIN ⟕, に変更することもできます。ジョインモードの変更はこのサンプルの出力に影響を与えないことにご注意してください。

ステップ 4 で追加された SQL WHERE/ORDER コンポーネントにより、レコードセットを (自宅、または勤務先の住所を抽出するなど) フィルターし並べ替えることができます。WHERE 句が型 `string` の パラメーター **`:address_type`** を作成したことにご注意してください。を作成することにご注意してください。パラメーターを使用することにより、マッピングから住所の種類 (自宅、または勤務先) を与えることができます。SQL WHERE/ORDER の詳細に関しては、 データベースデータのフィルターと並べ替え (SQL WHERE/ORDER) を参照してください。

最後に、入力コンポーネントによりマッピングの実行中に実際の パラメーター値を提供することができるようになります。MapForce 外でマッピングが実行される場合、(例えば、異なるマシン上で MapForce Server によりマッピングが実行される場合) 上記のデザイン時刻が無視されるように、コマンドライン パラメーターとしてマッピングのランタイムに入力が与えられる必要があります。詳細に関しては、マッピングに パラメーターを与えるを参照してください。

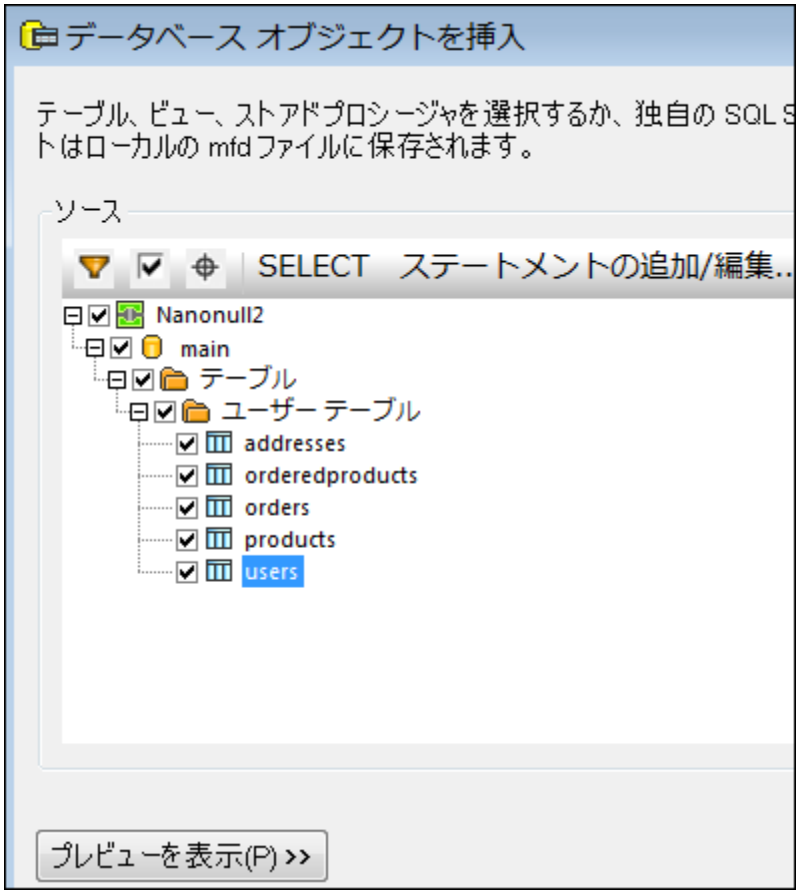
(C) 2015-2021 Altova GmbH

6.2.4.3 例: 複数のテーブルから CSV レポートを作成する

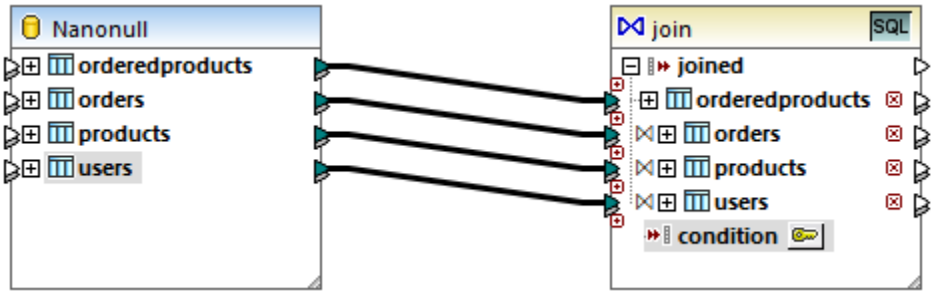
このサンプルは、CSV フォーマットの単一のレポートにデータを抽出する目的のための複数のデータベーステーブルのジョインの方法について説明しています。このサンプルで使用されているデータベースは、**Nanonull.sqlite** で、以下のパスで見つけることができます: <**マイドキュメント**>¥Altova¥MapForce2021¥MapForceExamples¥Tutorial¥。このデータベースは、(注文、製品、および住所が含まれる) 架空の会社の情報を含んでいます。リレーショナルデータベースでは通常、情報が正規化され、複数のテーブルに分割されます。例えば、`users` テーブルには、(姓、名、および電子メールアドレスがふくまれる) ユーザーの個人の情報が保管されます。データベースは、ユーザーにより注文された製品の情報が2つの異なるテーブルに含まれます: (注文の一意のID 、および注文の時刻を含む) `orders` および (注文された製品のリスト、および、個数が含まれる) `orderedproducts`。更に、製品の名前は`products` という異なるテーブルに保管されます。

このサンプルの目的は、特定の製品を誰が、何個注文したかを明確にするため、異なるテーブルより抽出されるデータをベースにしたレポートの生成です。この目的を達成するには以下の手順を行います:

1. **挿入** メニューから、**データベース**をクリックしてください。
2. データベース型を選択するようにプロンプトされると、「SQLite」をクリックして、「次へ」をクリックします。
3. 上記の**Nanonull.sqlite** データベースを参照し、「**接続**」をクリックします。
4. プロンプトされると、テーブル`orderedproducts`、`orders`、`products` および`users` を選択して、「OK」をクリックします。



5. ジョインコンポーネントをマッピングに追加し、4つのnodes/rows アイテムを「**入力の追加**」( ) ボタンをクリックして作成します。
6. データベースからの4つのテーブルをジョインコンポーネントの対応する入力アイテムに接続します。

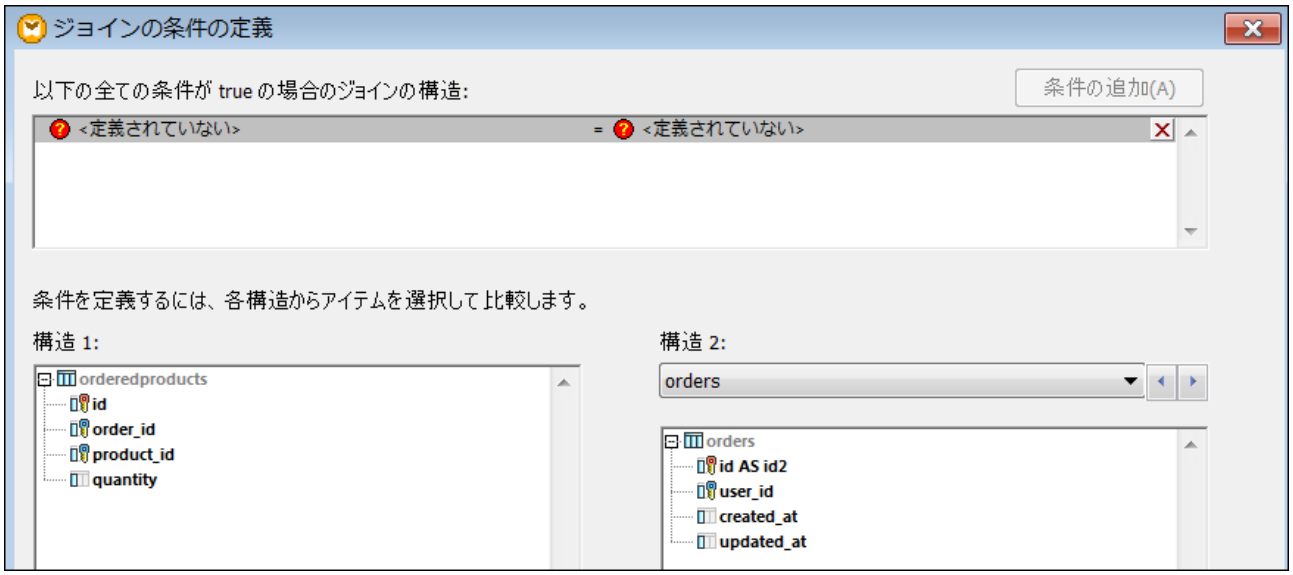


メモ 別のシナリオでは、全てのジョインされたテーブルが同じ「ルート」テーブルの下でネストされるようにジョインコンポーネントにテーブル `orderedproducts` を接続することができ、また、(同じレベルではなく、下のレベルに存在する)テーブル `orders` などを接続することができます。データベースリレーションシップの扱いも参照してください。マッピングの結果は、テーブルをジョインした結果と同じになります。違いとしては、このサンプルではジョイン条件は手動で作成されており、他のシナリオでは、ジョインの条件が MapForce により自動的に作成されています。ジョイン条件の定義を手動で行わずテーブルをジョインするサンプルに関しては例：SQL モード内のジョインテーブルを参照してください。同じ「ルート」テーブルの下でジョインされたテーブルの他のマッピングは、以下のパスで検索することができます: **<マイドキュメント>\Altova\MapForce2021\MapForceExamples\DB_Denormalize.mfd**。

このサンプルではジョインコンポーネントに接続されているテーブルの順序は以下のとおりです:

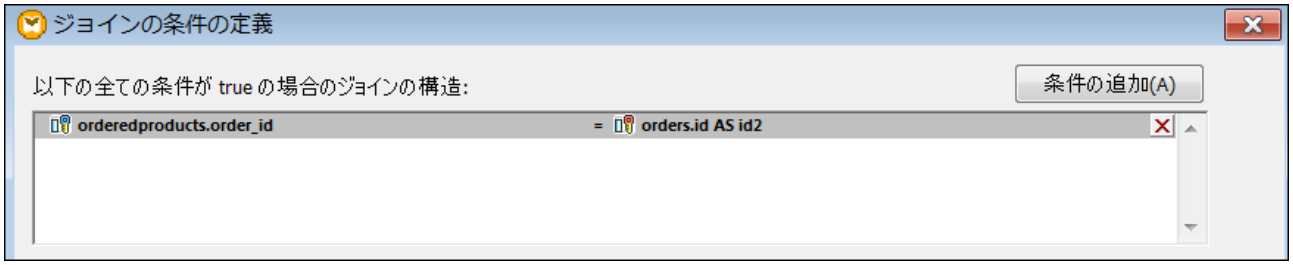
1. `orderedproducts`
2. `orders`
3. `products`
4. `users`

この順序は、「ジョインの条件の定義」Define Join Condition ( ) ボタンをクリックした時に、どのように対応する構造が「ジョインの条件の定義」ダイアログボックス上で表示されるかに影響を及ぼします。すなわち、最初のテーブル(`orderedproducts`) はデフォルトで構造 1の下に表示され、そのすぐ後のテーブル(`orders`) は、構造 2の下に表示されます。

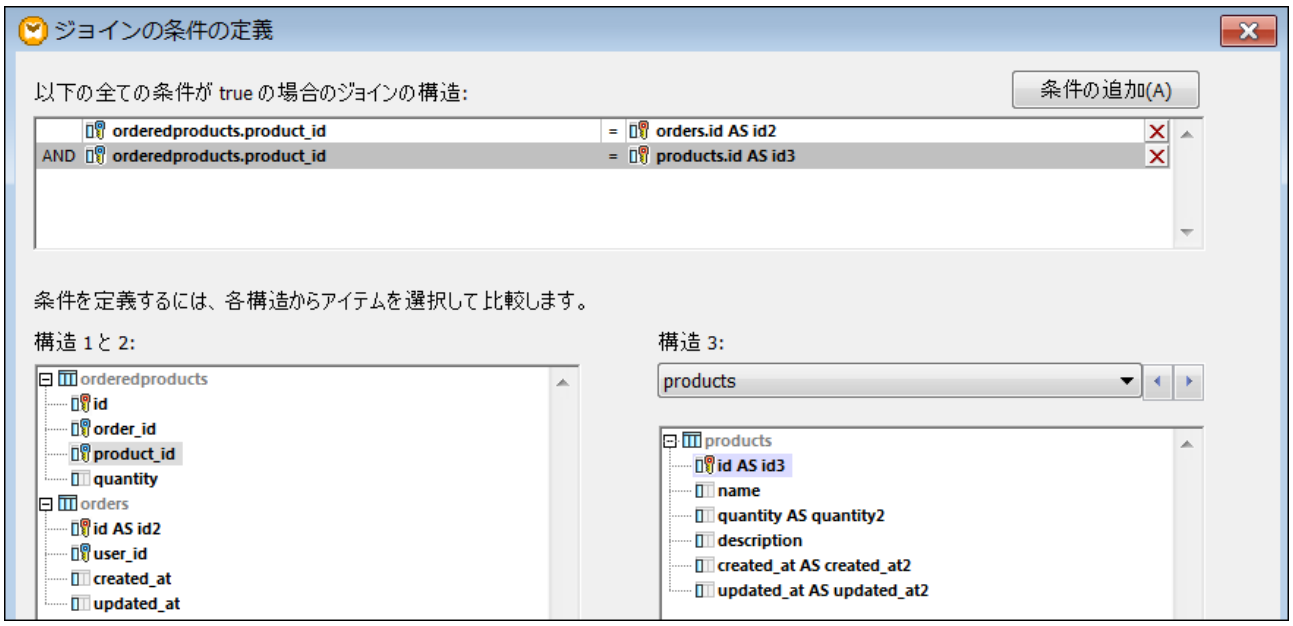


最初のジョインの条件を定義するには、左側のペイン内の `order_id` アイテムをクリックし、右側のペイン内の `id` アイテムをクリックします。`orderedproducts.order_id` と `orders.id` が組み合わされます:

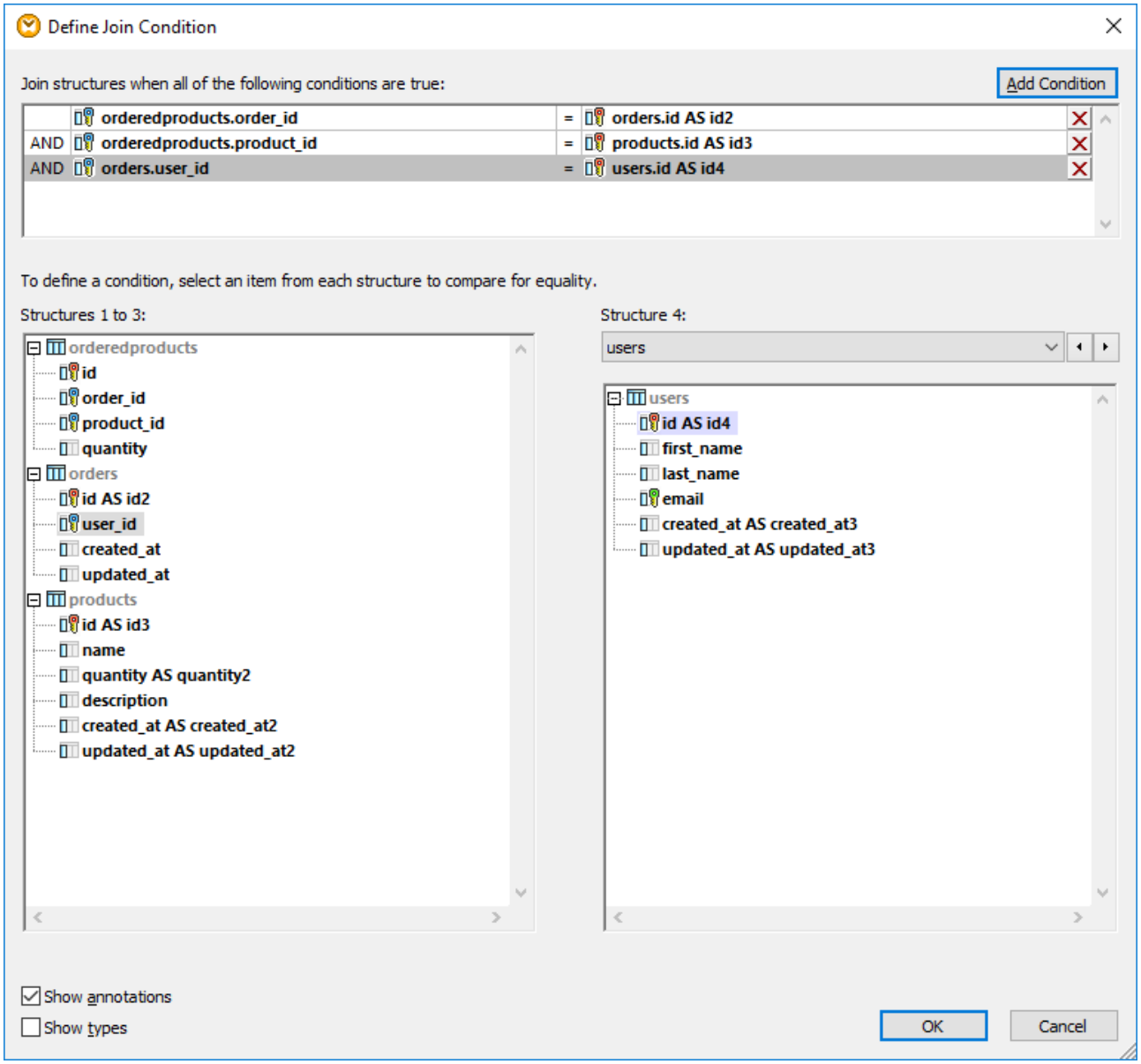
(C) 2015-2021 Altova GmbH



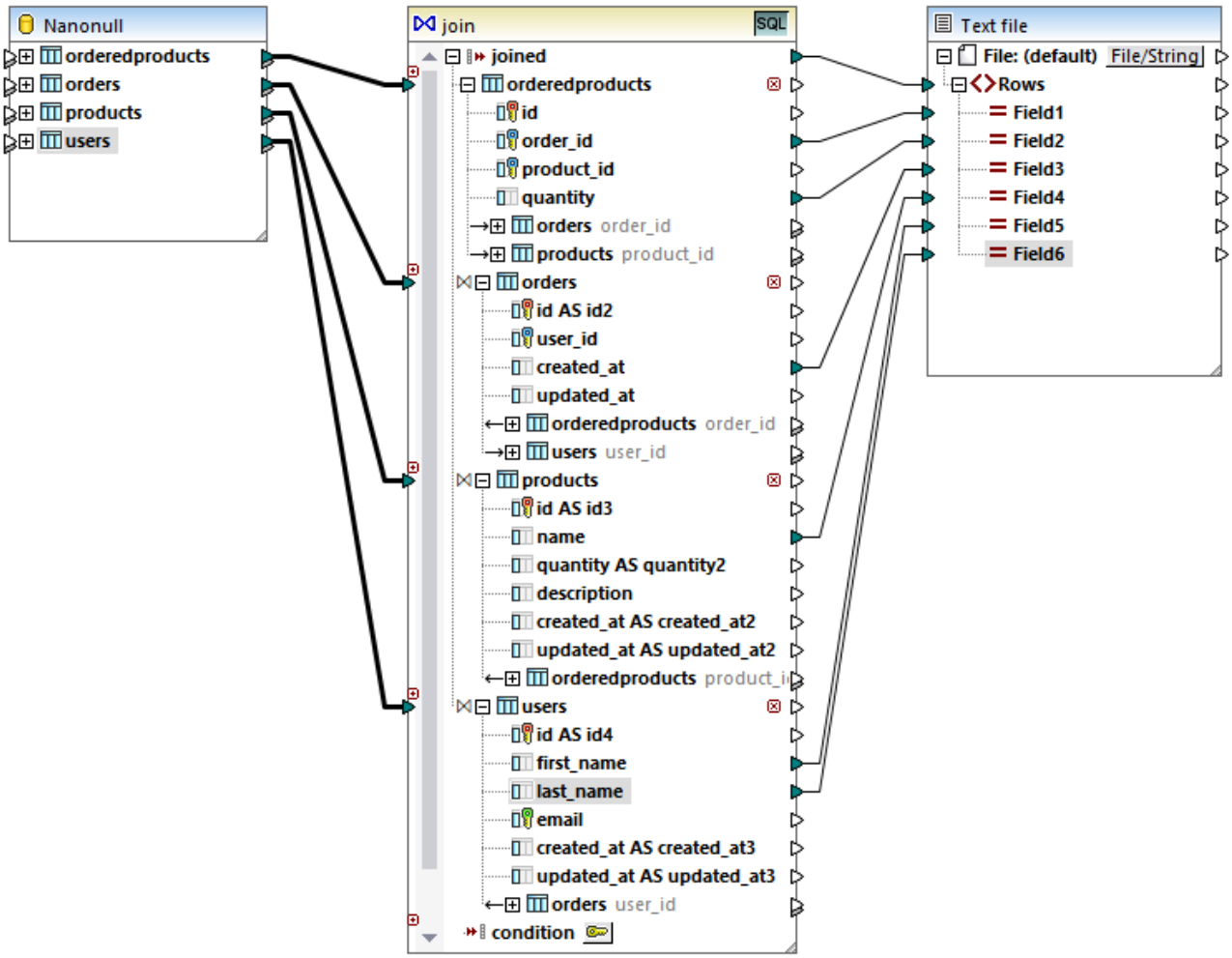
この時点まででは、2つのテーブルがジョインされました。3番目のテーブルを含むジョインの条件を定義するには、右側のペインの上で使用することのできるドロップダウンリストから希望するテーブルを選択します。左側のペインは、この場合、ジョインコンポーネント上でジョインの*前に*発生する全てのテーブルが表示されています。例えば、例えば、(これらのテーブルは、ジョインコンポーネント上で`products` の前に発生するため) `products` を右側で選択すると、左側は、`orderedproducts` と`orders` を表示します。テーブル`products` のフィールドを前のテーブルとジョインすることができます(この場合、`orderedproducts.product_id` と`products.id`)。



4番目のテーブル(`users`) をジョインするには、`users` テーブルをドロップダウンリストから選択します。フィールド `orders.user_id` と `users.id` を組み合わせることができます。



必要とされるジョインの条件がすべて定義されると、ジョインコンポーネントのアイテムをターゲットコンポーネントにマップすることができます。マッピングを完了するには、CSV コンポーネントを追加し、ジョインコンポーネントからのアイテムを下に示されるようにターゲット CSV コンポーネントに接続します（CSV とテキストファイルを参照してください）：



上で説明されているマッピングは、ジョインに含まれる4つのテーブルからコンパイルされた(CSV フォーマット の) レポートを作成します:

- 注文のID (`orderedproducts` テーブルから取得)
- 注文されたアイテムの個数 (`orderedproducts` テーブルから取得)
- 注文の時刻 (`orders` テーブルから取得)
- 注文された製品の名前 (`products` テーブルから取得)
- 製品を注文したユーザーの氏名 (`users` テーブルから取得)

このサンプル内のすべてのテーブルはINNER JOIN モードを使用してジョインされています。LEFT OUTER JOIN へジョインモードを変更するための詳細は、ジョインモードの変更を参照してください。

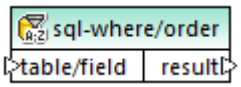
6.2.5 データベースデータのフィルターと並べ替え (SQL WHERE/ORDER)

データベースデータをフィルターし、並べ替える場合、SQL W HERE/ORDER コンポーネントを使用します。これにより、データをフィルターするSQL W HERE 句をMapForce グラフィカルユーザーインターフェイスから入力することができます。または、特定のデータベースフィールドによるレコードのセットを昇順および降順に並べ替える場合、ORDER BY 句を指定することもできます。

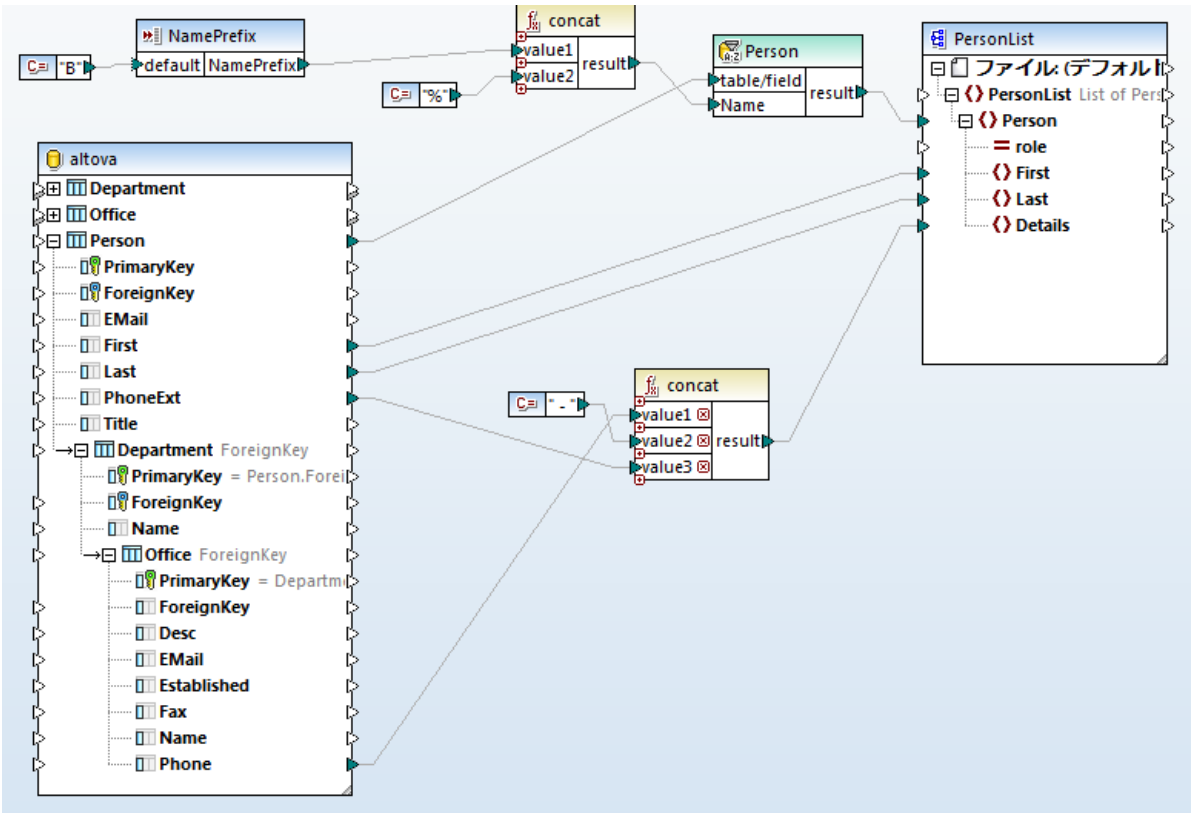
SQL WHERE/ORDER コンポーネントはデータベースマッピングコンポーネントに接続されている必要があります。ジョインされたセット、またはレコードをフィルターする必要がある場合、SQL WHERE/ORDER をジョインコンポーネントに接続することもできます(次を参照してください: データベースのデータのジョイン)。

SQL WHERE/ORDER コンポーネントをマッピングに追加する

1. 「挿入」メニューから **SQL WHERE/ORDER** をクリックします。デフォルトでは、SQL WHERE/ORDER コンポーネントは次の構造を持ちます:

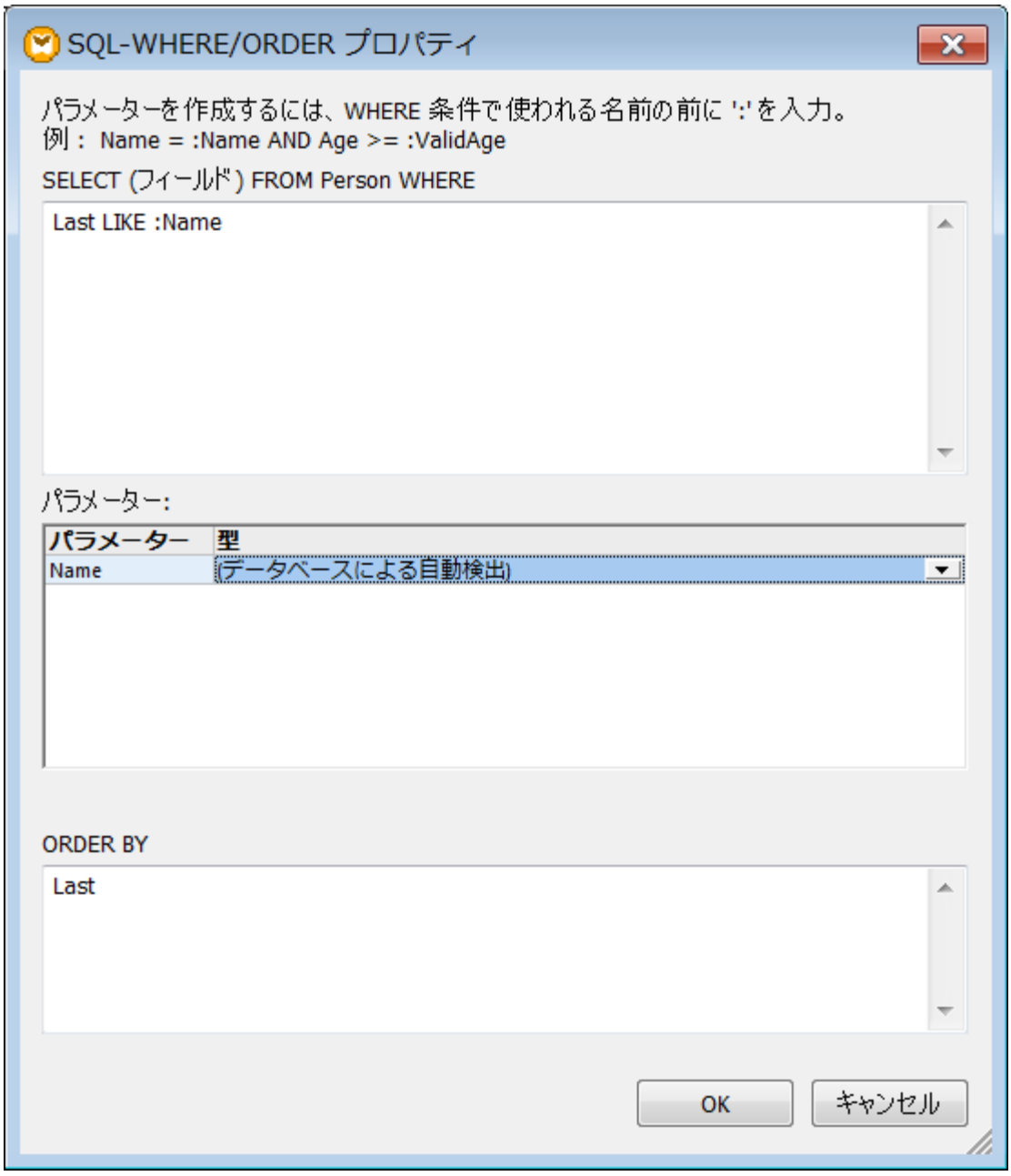


2. ソースデータベーステーブルを接続、またはSQL WHERE/ORDER の `table/field` アイテムのフィールドに接続します。例えば、<マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥フォルダーからマッピング **DB_PhoneList.mfd** を開きます。このマッピングでは、SQL WHERE/ORDER は、ソーステーブル「Person」から、「B」文字から始まる姓を持つ全てのレコードをフィルターします。



3. SQL WHERE/ORDER コンポーネントのヘッダーをダブルクリックします。(または、右クリックしてコンテキストメニューからプロパティを選択します)。これにより、「SQL WHERE/ORDER プロパティ」ダイアログボックスが開かれます。

(C) 2015-2021 Altova GmbH



4. 上のテキストボックスにSQL WHERE 句を入力します。または、ORDER BY 句を入力します。上のイメージは **DB_PhoneList.mfd** マッピング内で定義されているWHERE とORDER BY 句を示しています（この設定については、下で詳しく説明されています）。例えば、次を参照してください：WHERE とORDER BY 句の作成。

SQL WHERE/ORDER にパラメーターを与える

マッピング **DB_PhoneList.mfd** 内で使用されているSQL WHERE/ORDER コンポーネントは、WHERE 句を次のように定義しています:

```
Last LIKE :Name
```

「Last」は、接続されたテーブル内のデータベースフィールド名を参照しています。「LIKE」は、SQL 演算子です。「:Name」は、マッピング上に「Name」という名前のパラメーターを作成します。

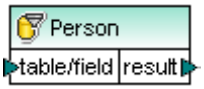
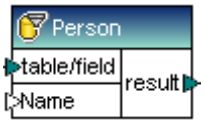
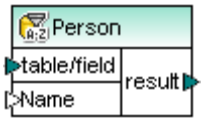
SQL WHERE/ORDER コンポーネント内のパラメーターは任意です。マッピングから WHERE 句に値をパスする際に役に立ちます。パラメーター無しでは、上の WHERE 句は以下のように書くことができます:

```
Last LIKE "B%"
```

これは「B」から始まる姓を持つ全ての個人を抽出します。クエリに更に柔軟性を与えるため、「B% 」の代わりにパラメーターを追加します。これによりマッピングから他の文字を与えることができるようになります(例えば、「C」、そして、定数、またはマッピング入力パラメーターを変更するだけで、「C」から始まる姓を持つ個人を抽出できるようになります)。

SQL WHERE/ORDER コンポーネントの表示

SQL WHERE/ORDER コンポーネントの重要な点は設定内の定義により外観が異なる点です。このようにして SQL WHERE/ORDER コンポーネントの振る舞いを直接マッピングから確認することができます。例:

	WHERE 句が定義済みです。
	定義済みのパラメーターが存在する WHERE 句。パラメーター「Name」「table/field」アイテムの下に表示されています。
	定義済みのパラメーターが存在する WHERE 句。更に、ORDER BY 句が定義済みです。並べ替えは、A-Z 並べ替えアイコンにより表示されています。

SQL WHERE/ORDER ヘッダーにマウスのカーソルをポイントすると、定義されている異なる句を表示するヒントが表示されます。

6.2.5.1 WHERE と ORDER BY 句の作成

SQL WHERE/ORDER コンポーネントがマッピングに追加されると、コンポーネントに接続されるデータをフィルターする方法を定義することのできる WHERE 条件(句)が必要になります。WHERE 条件は、MapForce の「SQL WHERE/ORDER プロパティ」ダイアログボックスに入力される必要があります。

MapForce から WHERE 条件を作成することは、MapForce 外で同じ SQL 句を作成することと類似しています。対応するデータベースの SQL 言語に適応される構文を使用してください。例えば、演算子、ワイルドカード、サブセレクト、または集計関数を使用することもできます。マッピングからパスすることのできるパラメーターを作成し、後にパラメーター名が続くセミコロン文字(:)を入力します。

メモ WHERE 句の作成が終わると、「OK」をクリックします。MapForce は最後の SQL ステートメントの整合性を検証します。構文エラーが存在するとダイアログボックスがプロンプトを表示します。

以下の演算子を WHERE コンポーネントにて使用することができます:

(C) 2015-2021 Altova GmbH

演算子	説明
=	等価
<>	異なる
<	未満
>	を超える
>=	以上
<=	以下
IN	指定した値をカラムから検索
LIKE	指定したパターンを検索
BETWEEN	指定した範囲を検索

% (パーセント) ワイルドカードを使用して、パターン内の文字数を示します。例えば、 lastname という名前のフィールドから「r」で終わる記録を全て抽出します。

```
lastname = "%r"
```

XML データベースデータの保管とクエリをサポートするデータベース(例えば、IBM DB2、Oracle、SQL Server) をクエリする場合、その特定のデータベースに適用することのできる XML 関数をキーワードに使用することができます。例:

```
xmlexists('$c/Client/Address[zip>"55116"]' passing USER.CLIENTS.CONTACTINFO AS
"c")
```

次も参照してください: 例: IBM DB2 XML 型列からデータを抽出する。

任意で特定のフィールドにより抽出されたレコードセットを並べ替えるには、「SQL WHERE/ORDER プロパティ」ダイアログボックスの対応するテキストボックスに ORDER BY 句 を追加します。複数のフィールドを並べ替えるには、フィールド名 をコンマで区切ります。並べ替えの順序を変更するには、 ASC と DESC キーワードを使用します。例えば、次の ORDER BY 句は、 lastname と firstname により降順に順序付けられたレコードを抽出します:

```
lastname, firstname DESC
```

サンプル 1

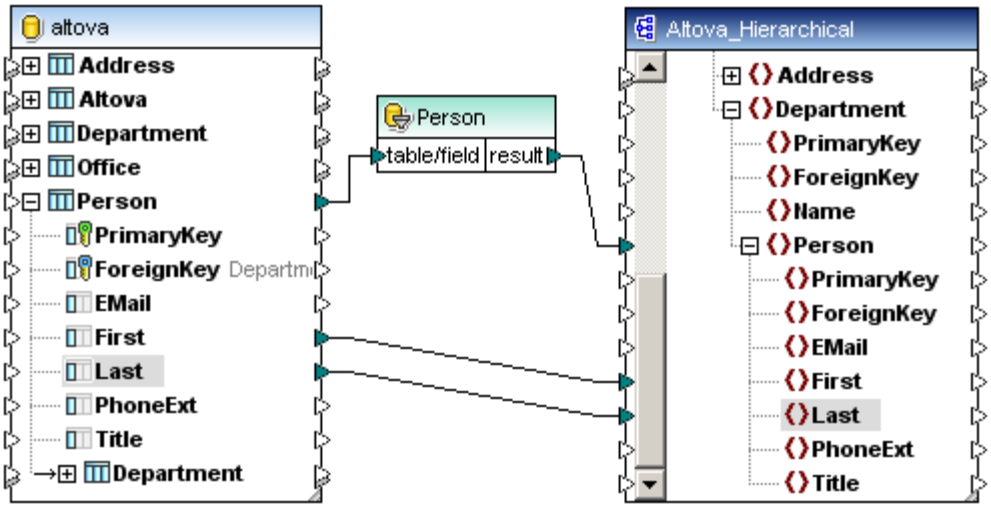
次の WHERE 条件は **altova.mdb** データベースコンポーネントの Person テーブルに添付されています。これは First と Last が文字「C」より大きいレコードを抽出します。すなわち、「Callaby」以降の名前をすべて抽出します。SELECT 文により First ならびに Last 内にあるコンテンツが C という文字より大きなレコードが取得されます。**altova.mdb** サンプルファイルの Person テーブルにある Last を例にすると、Callaby 以降のレコードが取得されます。

```
First > "C" AND Last > "C"
```

どのように接続が作成されているかに注目してください:

- 問い合わせを行いたいテーブル(この場合は「Person」)から **table/field** パラメーターへの接続を作成します。

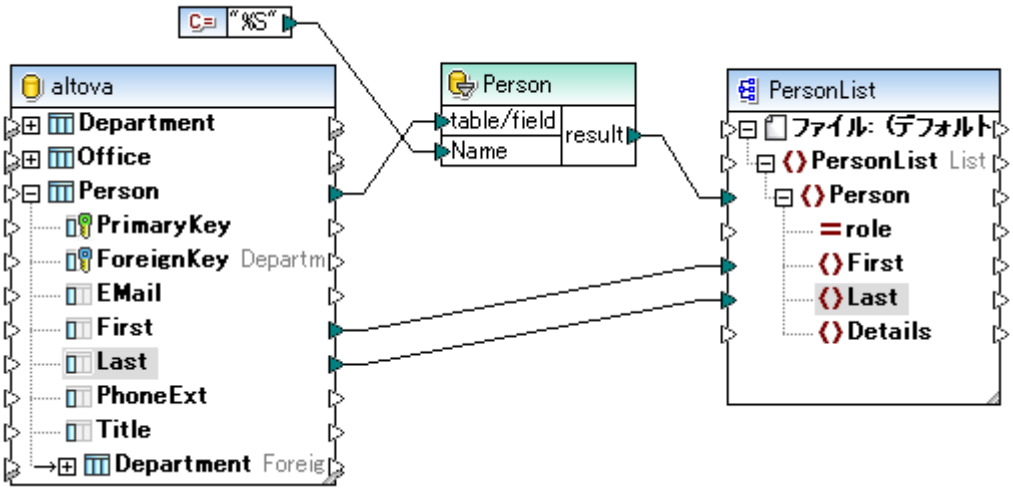
- **result** パラメーターを、問い合わせ/フィルタリングを行うフィールドの「親」アイテム(この場合はPerson アイテム) へ接続します。First ならびにLast のフィールドがターゲットコンポーネントに接続され、マッピングの結果に表示されます。



サンプル2

このW HERE ステートメントにより、SQL W HERE/ORDER コンポーネントの パラメーターとして **Name** が作成されます。この場合、「Last」 カラムにて パターンの検索が行われます。ワイルドカードの% により、任意の数の文字が指定されます。

```
Last LIKE :Name
```

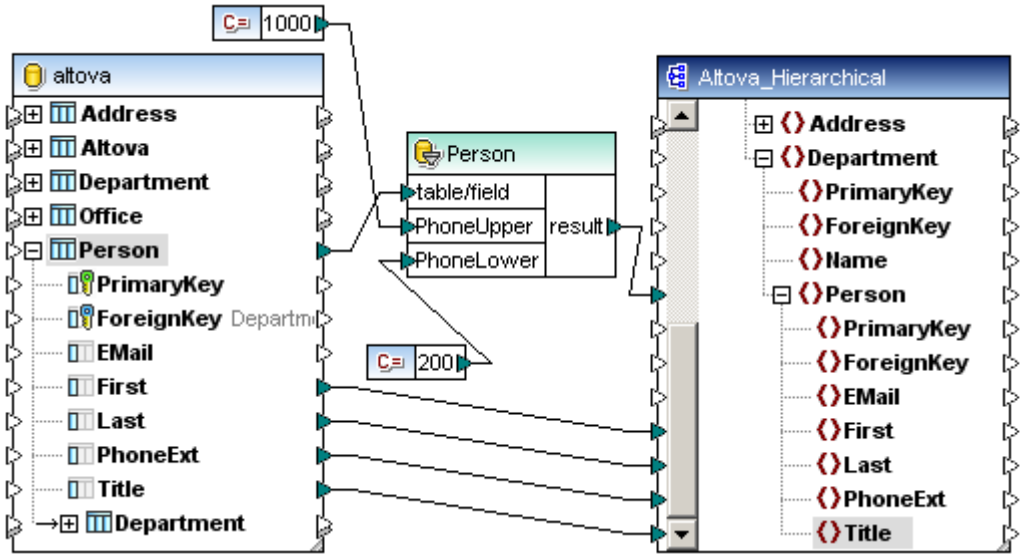


定数 コンポーネント %S は**Name** パラメーターの値を与えます。ワイルドカード % は文字数を表示します。これにより、マッピングが列「Last」 内の パターン(「S」 で終わる名前)を検索します。

サンプル3

このWHERE 句により、PhoneUpper とPhoneLower という2 つのパラメーターが作成され、PhoneExt の値が比較されます。上限の値と下限の値は、以下のマッピングに示される通り、定数コンポーネントにより与えられます。

```
PhoneExt < :PhoneUpper and PhoneExt > :PhoneLower
```



この例にあるWHERE 句では、BETWEEN 演算子を使用することもできます

```
PhoneExt BETWEEN :PhoneUpper and :PhoneLower
```

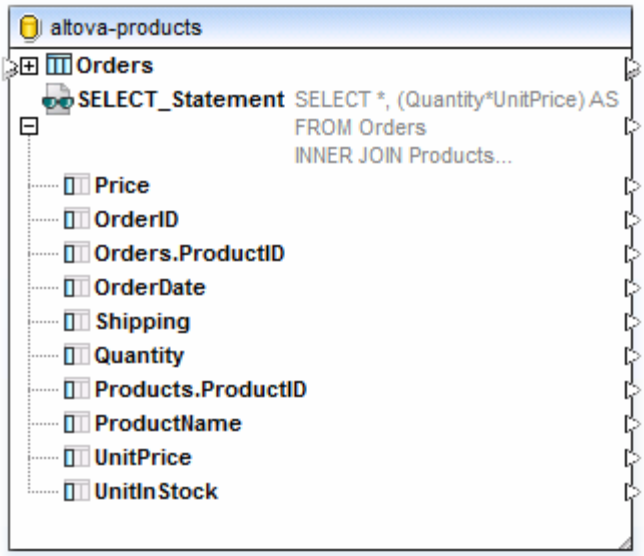
6.2.6 仮想テーブルとしての SQL SELECT ステートメント

MapForce ではデータベースコンポーネントにてパラメーターを伴うSQL SELECT ステートメントの作成がサポートされています。SELECT ステートメントはテーブル型の構造をしており、SELECT ステートメントにより生成された結果のフィールドが含まれます。このような構造は(通常のデータベーステーブルやビューと同様に)マッピングデータソースとして使用することができます。

- SELECT ステートメントにてINNER/OUTER **JOIN** が使用されている場合、全てのテーブルのフィールドがコンポーネントには収められます。
- (SQL の"AS" キーワードにより指定された)相関名を伴う表現もコンポーネント内のマッピングアイテムとして表示されます。
- SELECT ステートメントは既存のデータベーステーブルをMapForce へ挿入する際に作成されます。
- SELECT ステートメントには、SQL WHERE/ORDER コンポーネントで使用される構文によるパラメーターを含めることができます。

> 入力パラメータを含まない限り、SQL SELECT ステートメントはC++、C#、Java、およびBUILT-IN 言語内でサポートされます。入力パラメーターを持つSQL SELECT ステートメントはBUILT-IN 変換言語内でのみサポートされます。

SELECT がデータベースコンポーネントに追加されると、返されるフィールドはマッピングのために使用することができます。例:



SELECT ステートメントはコンポーネント内部に表示されます。ステートメントにより表示される結果の行数は、メニューコマンド「**ツール | オプション**」を選択し、全般タブにて指定することができます。

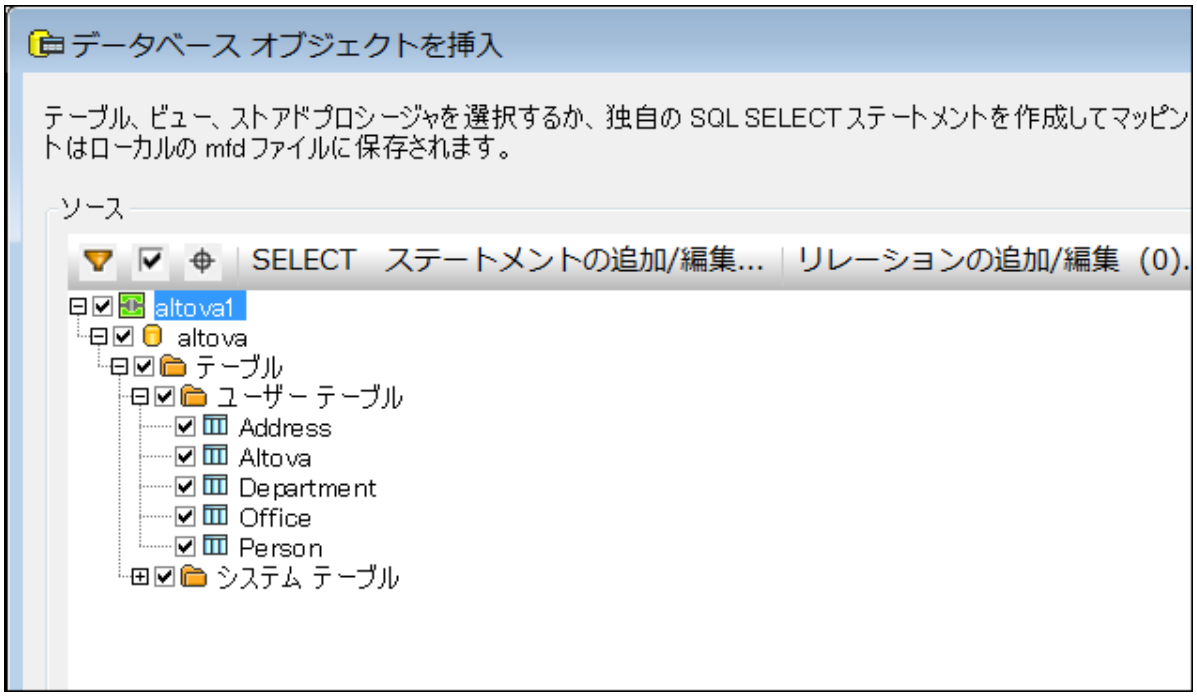
6.2.6.1 SELECT ステートメントの作成

データベースコンポーネントを含むマッピングで SELECT ステートメントを作成することができます。マッピングがデータベースをまだ含んでいない場合、データベースを先に追加します（データベースへ接続するを参照）。このサンプルのスコープのために、メニューコマンド「**挿入 | データベースを挿入**」を選択して、<**マイドキュメント**>¥Altova¥MapForce2021¥MapForceExamples¥Tutorial¥ フォルダー内で使用することのできる **altova-products.mdb** ファイルにウィザードに従い接続します。

SELECT ステートメントの作成：

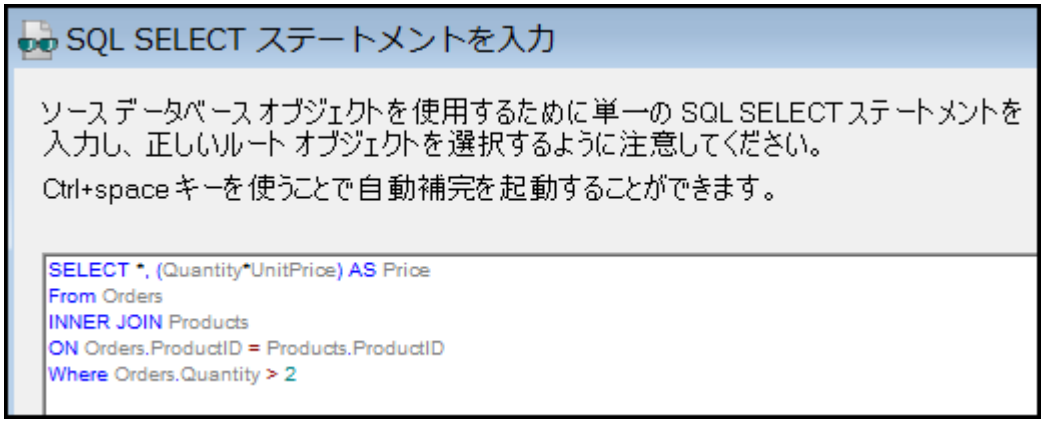
1. データベースコンポーネントのタイトルを右クリックして、「**追加/削除/データベースオブジェクトの編集**」を選択します。（代わりに、データベースコンポーネントを選択してメニューコマンド「**コンポーネント | 追加/削除/データベースオブジェクトの編集**」を選択します。）

(C) 2015-2021 Altova GmbH



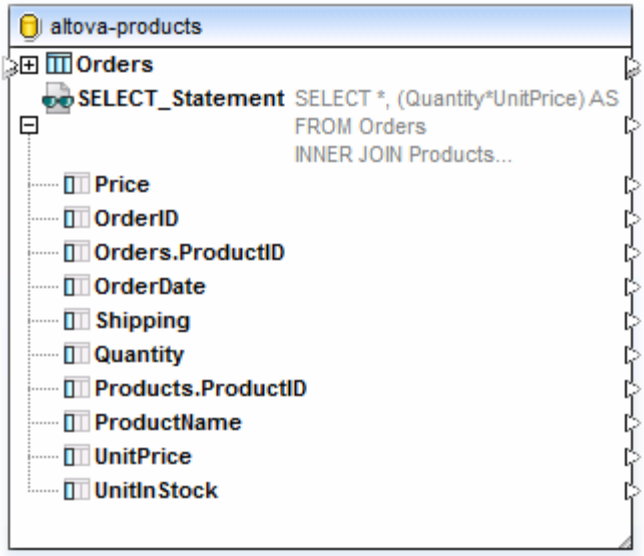
2. 以下を行います:
   - 既存のテーブルからSELECT ステートメントを生成するには、テーブルを右クリックして、コンテキストメニューから、**SQL ステートメントの生成と追加** を選択します。生成されたステートメントは後に編集することができます。
   - カスタムSELECT ステートメントを作成するには、 **SELECT ステートメントの追加/編集** ボタンをクリックします。
3. 必要に応じて、ステートメントを編集または作成します。例: 下のSELECT ステートメントは、**<マイドキュメント>\Altova\MapForce2021\MapForceExamples\Tutorial** フォルダー内の**altova-products.mdb** ファイルにより有効です。 **Price** フィールドは、**Quantity** と**UnitPrice** の2つのフィールドで構成されており、関連付けられた名前(`AS Price` として宣言されます。

```
SELECT *, (Quantity*UnitPrice) AS Price
From Orders
INNER JOIN Products
ON Orders.ProductID = Products.ProductID
Where Orders.Quantity > 2
```



4. 「SELECT ステートメントの追加」をクリックします。SELECT ステートメントは、テーブル、ビュー、プロシージャと同様にデータベースオブジェクトとして、表示されます。

5. 「OK」をクリックします。データベースコンポーネントにSELECT ステートメントが表示され、SELECT クエリにより返されたフィールドのデータをマップすることができます。



重要な注意点:

- SELECT ステートメント内のすべての計算された式は、マップすることのできるアイテムとして(このサンプルでは、"AS Price" のような)一意の関連付けられた名前を持つ必要があります。
- JDBC を使用して、Oracle またはIBM DB2 データベースに接続する場合、`SELECT` ステートメントでは最後のセミコロンは存在しません。

SELECT ステートメントを削除する:

1. データベースコンポーネントのタイトルを右クリックして、「**追加/削除/データベースオブジェクトの編集**」を選択します。
2. 削除するSELECT ステートメントを右クリックして、「**SELECT ステートメントの削除**」 を選択します。

6.2.6.2 例:パラメーター付きの SELECT

このサンプルは、 Microsoft Access データベースからデータを読み込みCSV ファイルに書き込むMapForce マッピングの作成方法について説明します。特に、この例のマッピングは、パラメーターを持つカスタムデータベースSELECT クエリを使用しています。SELECT ステートメントは複数のテーブルからのデータを 結合します。そして、更なる処理のために結果はマッピングへ与えられます。

このサンプルは、以下のパスで見つけることのできるマッピングデザイン(.mfd) が伴います: **<マイドキュメント>\Altova\MapForce2021\MapForceExamples\Tutorial\select-component.mfd**。このサンプルファイルを開き、最初に分析する、または以下のステップを最初から行います。

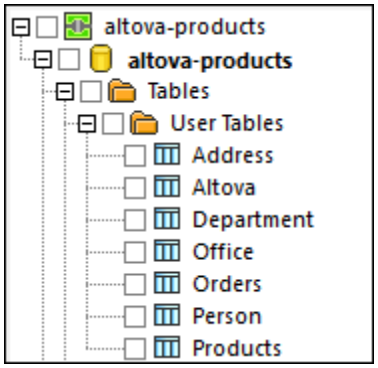
このサンプルはMicrosoft Access データベースを使用しますが、その他のデータベース方でも同じ方法で作動します。データベースへ接続するを参照してください。

目的は、以下のとおりです:

1. データベースから、オーダーされるアイテムの数が、カスタムの値を超えるオーダーのみを選択する必要があります。このカスタムの値は、マッピングへのパラメーターとして与えられます。この目的を達成するために、入力パラメーターを持つカスタムデータベース SELECT ステートメントを作成します。
2. Access データベース内では、日時フォーマットは、`YYYY-MM-DD HH-MI-SS` です。CSV ファイル内では、時刻の部分が省かれるため、フォーマットは `YYYY-MM-DD` になります。この目的を達成するために、MapForce 内で使用することのできる、date-from-datetime 関数を使用します。
3. 結果 CSV ファイルは **OrdersReport.csv** という名前を持つ必要があります。

ステップ 1: SELECT 構造を追加する

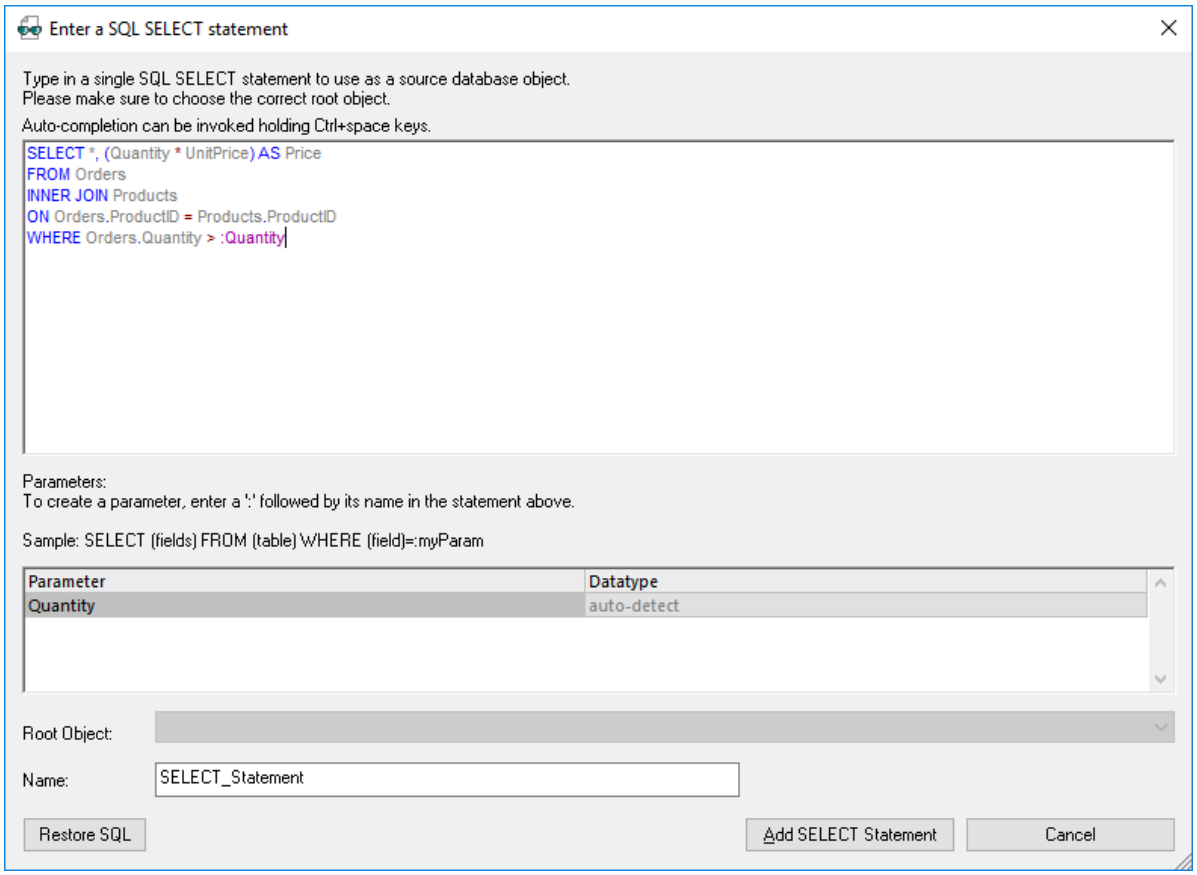
1. メニューから**データベースの挿入**をクリックします。
2. **Microsoft Access (ADO)** を選択して、**<マイドキュメント>\Altova\MapForce2021\MapForceExamples\Tutorial** フォルダー内で使用することのできる **altova-products.mdb** ファイルに接続するウィザードステップに従います。



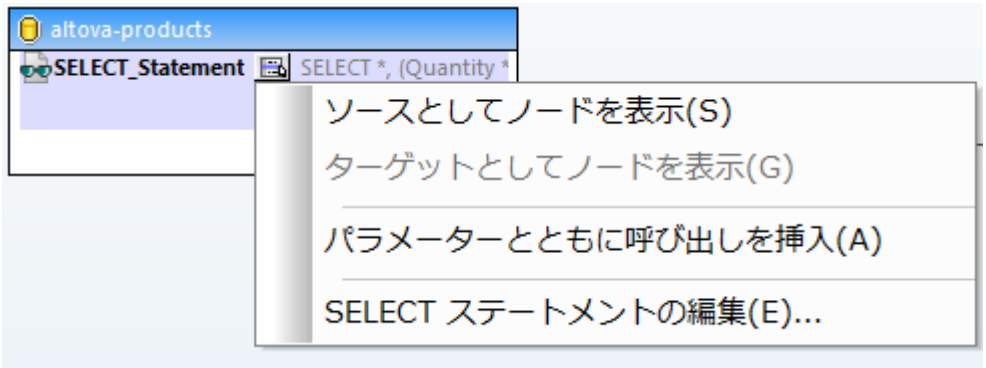
3. 「**データベースオブジェクトの挿入**」ダイアログボックスから「**SELECT ステートメントの追加/編集**」をクリックし、以下のクエリを挿入します:

```
SELECT *, (Quantity * UnitPrice) AS Price
FROM Orders
INNER JOIN Products
ON Orders.ProductID = Products.ProductID
WHERE Orders.Quantity > :Quantity
```

このクエリ Orders と Products テーブルの結合を使用し、すべてのフィールド (`*`), と計算された値 (`AS Price`) 。クエリは以下も指定します : WHERE 句内の **Quantity** パラメーター

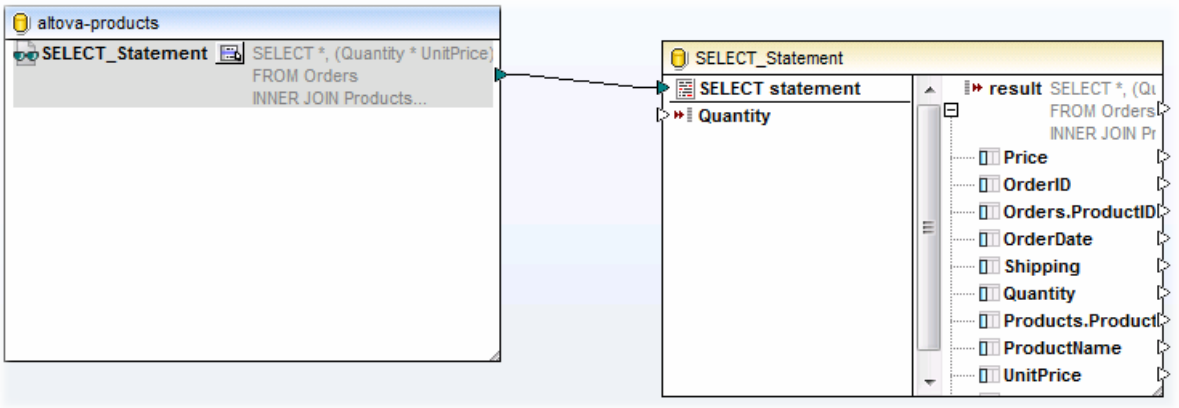


4. 「SELECT ステートメントの追加」をクリックします。
5. 「OK」をクリックします。**altova-products** コンポーネントがマッピングエリアに追加されます。
6. **altova-products** コンポーネント から、**パラメータ**と共に呼び出しを挿入 を選択します。



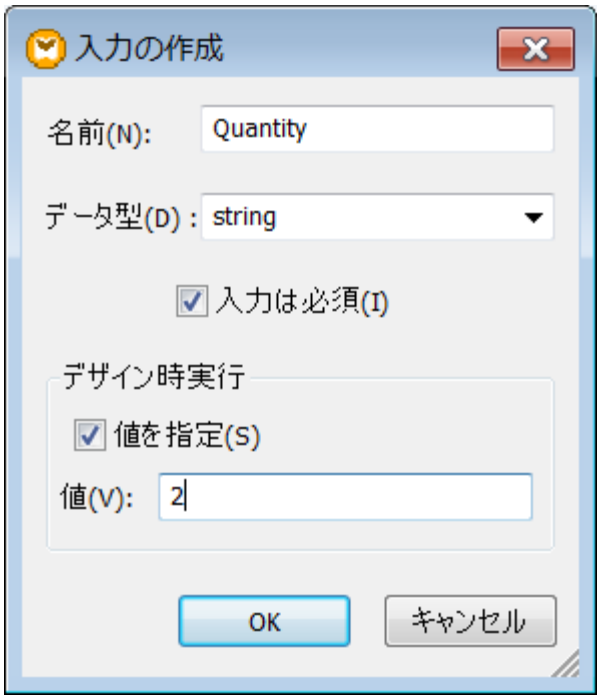
新規の構造（SELECT_ステートメント）をマッピングで使用することができるようになります。これは2 つのパートに分割されます: 左のパートは入力コネクタを、右のパートは出力コネクタを提供します。左のパートは前に定義された**Quantity** パラメーターを含んでいることに注意してください。

(C) 2015-2021 Altova GmbH

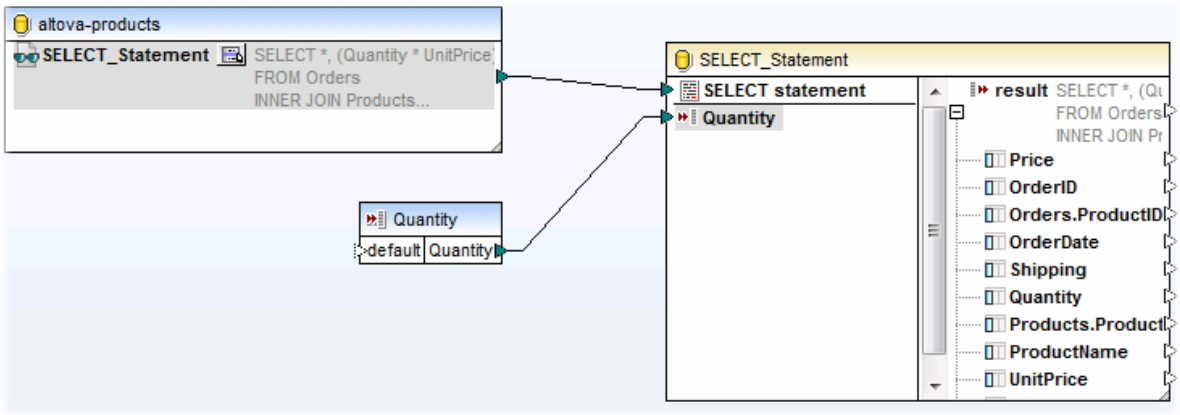


ステップ 2: 入力パラメーターの追加

1. 「挿入」メニューから「入力の挿入」をクリックします。
2. 型 "Quantity" を名前とします。
3. **Design-time Execution** 内に、デザイン中にマッピングの実行のために使用されるパラメーター値を入力します。(このサンプルでは、"2") 詳細に関しては、マッピングにパラメーターを与えるを参照してください。



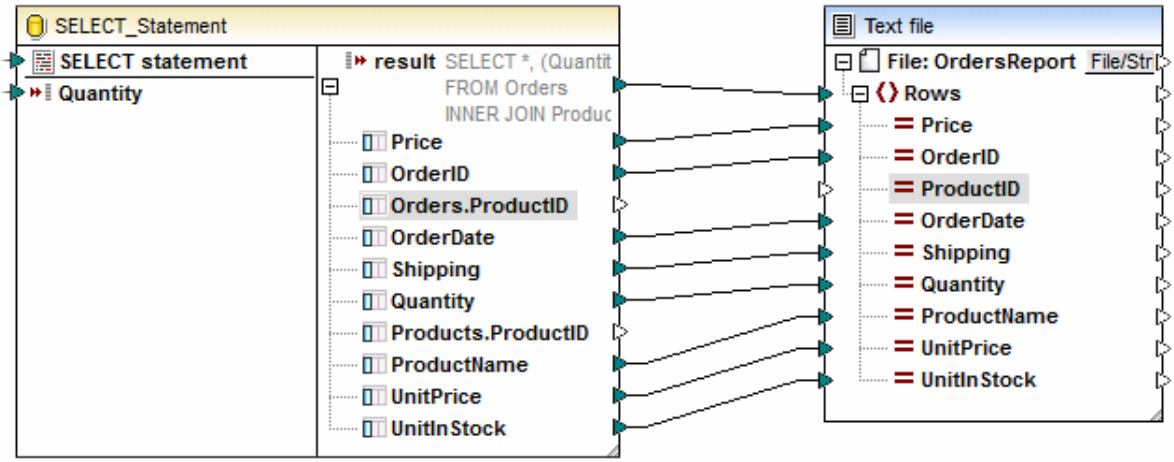
入力パラメーターをデータベースの呼び出しに接続します。構造は、以下のとおりです。



ステップ 3: ターゲット CSV コンポーネントを追加する

1. 「挿入」メニューから「テキストファイル」をクリックします。
2. 「標準 CSVのための単純な処理を使用する..」を選択し、「続ける」をクリックします。
3. コンポーネント設定ダイアログボックスから、「フィールドの追加」をクリックし、9つの新しいフィールドを追加します。データベースフィールドの名前と同じ名前をCSV フィールドに与えることが奨励されています。これは、マッピング接続を作成する際に時間の節約になります。これらの設定に関しての詳しい情報は、CSV オプションの設定を参照してください。

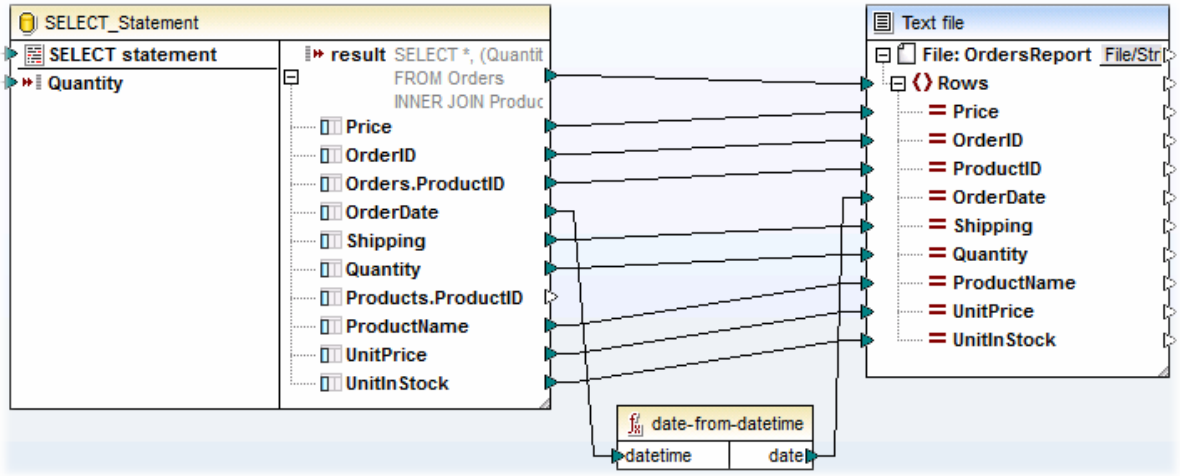
4. SELECT 構造の**result** ノードとCSV コンポーネントの**Rows** ノード間の接続を作成します。



CSV コンポーネントのフィールドの多くは、データベースと同じ名前を持ち、マッピング接続は、**result** を **Rows** に接続すると自動的に作成されます。これが発生すると、「**接続**」メニューを選択して、「**マッチする子を自動的に接続する**」オプションが有効化されていることを確認してください。 SELECT 構造内でこの名前を持つフィールドがないため、手動で接続したマッピングアイテムは **ProductID** のみです。

ステップ 4: 日付の変換

ライブラリウィンドウ内で、`date-from-datetime` 関数を検索し、マッピングエリアにドラッグします。入力と出力下で表示されるように接続します。



ステップ 5: 出力ファイルの名前の設定

OrdersReport.csv の出力ファイルの名前を設定するには、CSV コンポーネント、をダブルクリックし、出力ファイルボックス内に値を入力します。**出力ファイル** テキストボックス内に値を挿入することのできるコンポーネント設定ダイアログボックスが開かれます。

6.2.7 XML データから / データベースフィールドへのマッピング

MapForce を使用して、XML を保管するデータベースフィールド（列）からデータをマップ、またはデータベースフィールドへデータをマップすることができます。データベースフィールド（列）に保管される XML データが、抽出され、MapForce によりサポートされる他の構造に書き込まれること、またはその逆が可能になります。データを以下のようにマップすることができます:

1. XML 型専用の"へ" または "から" フィールド（例えば、SQL Server 内の `Xml` 、Oracle 内の `XMLType` ）。繰り返しまたは、または XML へから書き込む専用の XML フィールドは、XML をサポートするネイティブのデータベースに適用することができます(IBM DB2、Oracle、および SQL Server など)。
2. XML コンテンツを保管する"へ" または "から" テキストフィールド（例えば、`Text`、`Varchar`）。これはテキストフィールドが XML ドキュメントを保管するために十分な長さがあるデータベースに適用することができます。

いずれにせよ、有効な XML スキーマは、マップするデータのデータベース列、またはデータベース列からのために存在する必要があります。データベース列が XML を保存する場合、MapForce は、（データベースがサポートする場合）XML スキーマを直接データベースから，割り当てる、または外部ファイルからスキーマを選択するオプションを与えます。各データベース列につき、1 つの XML スキーマを割り当てることができます。スキーマに複数のルート要素が存在する場合、スキーマの単一のルート要素を選択することができます。

XML がデータベース内で文字列フィールドとして保存されている場合、XML ドキュメントの文字エンコードは文字列フィールドをベースとしています。データベースフィールドがテキストを Unicode として保存していない場合は、文字の一部を表示できない場合があります。

データベースの一部では、XML フィールドのためのXML エンコード(このエンコードはデータベースの文字のセットと異なる文字のセットを有する場合があります)をサポートします。データベースによりサポートされている場合はXML ドキュメント エンコード宣言はXML フィールド内で宣言されているものと解釈されます。データベースにより与えられたXML エンコードサポートに関する詳細は、それぞれのドキュメントを参照してください。

6.2.7.1 XML スキーマをデータベースフィールドに割り当てる

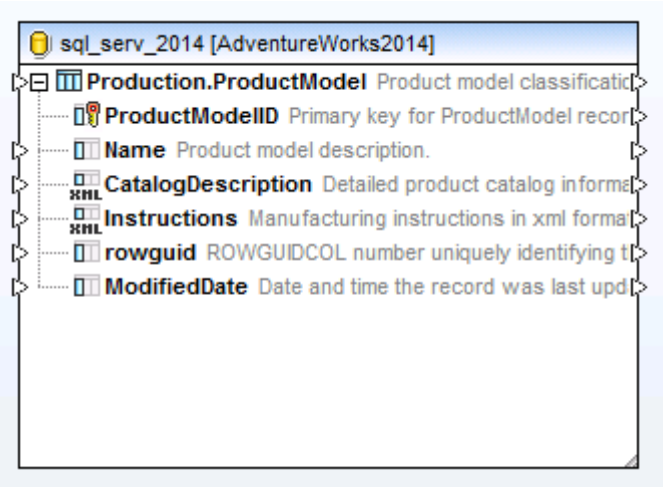
このトピックはデータベース内でXML 型として定義されているフィールドにスキーマを割り当てる方法を説明します。下の説明は、SQL Server 2014 とAdventure W orks 2014 データベースを使用しています。後者は、GitHub (https://github.com/Microsoft/sql-server-samples/releases/tag/adventureworks) のAdventureW orks サンプルからダウンロードすることができます。XML フィールドからの、またはXML フィールドへのデータのマッピングは、XML フィールドをサポートする他のデータベースの型と同様に動作します。

Adventure W orks 2014 データベースをマッピングコンポーネントとして追加する:

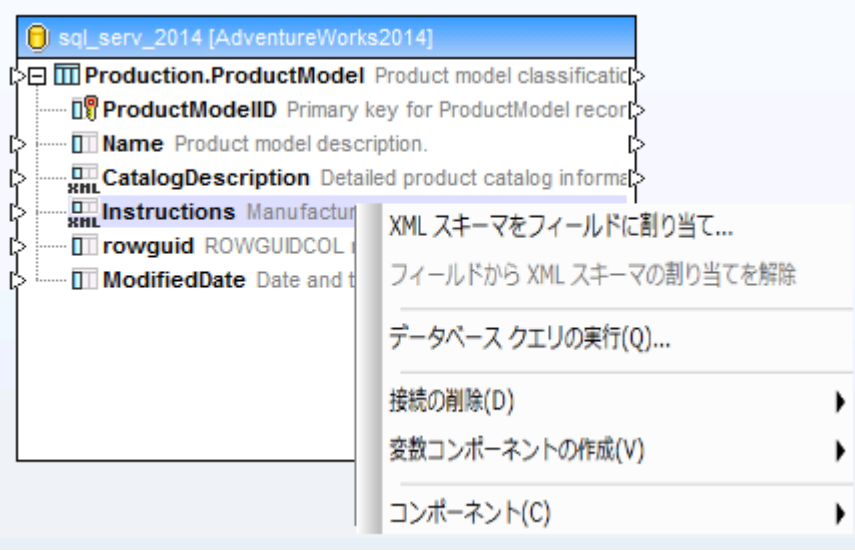
1. **挿入** メニューから **データベースをクリック**し、接続ウィザードを優先順序(ADO またはODBC)に従い進めます。更に詳しい情報に関しては、Microsoft SQL Server (ADO) に接続 とMicrosoft SQL Server (ODBC) に接続 を参照してください。
   メモ **SQL Server Native Client** ドライバーを使用する場合、**統合されたセキュリティ** プロパティをスペース文字に設定する必要があります(SQL Server Data Link プロパティのセットアップを参照)。
2. 「**データベースオブジェクトの挿入**」 ダイアログボックスから、**Production** スキーマを展開し、**ProductModel** テーブルを選択します。

3. 「OK」をクリックします。

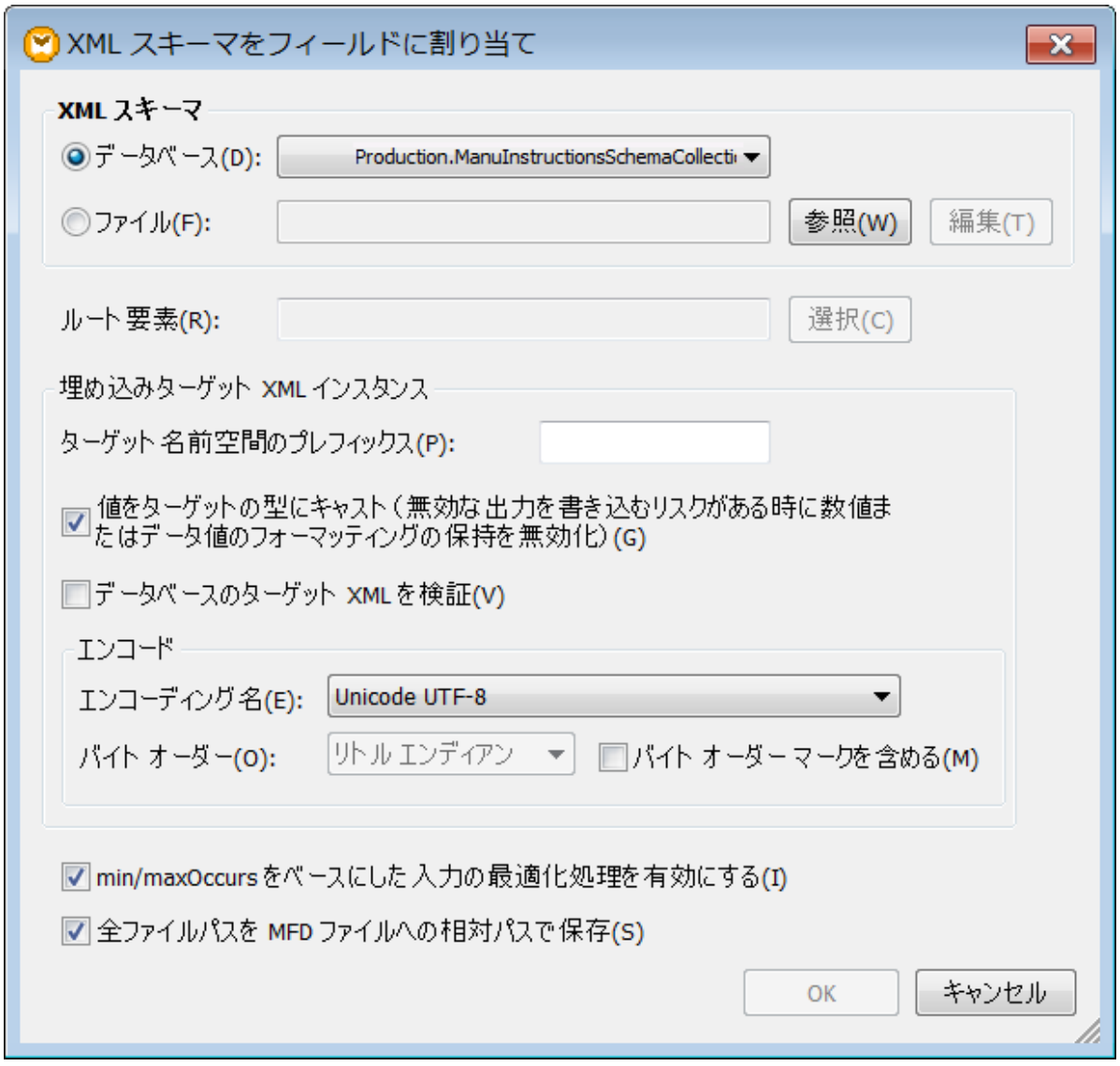
データベーステーブルがマッピングエリアに追加されました。このテーブルには次の2 つのXML 型のフィールドがあります:
CatalogDescription と**Instructions**:



マッピング上にXML フィールド の構造が表示されるようにするには、フィールドコンテンツのXML スキーマ が必要になります。**Instructions** フィールドを右クリックして、コンテキストメニューから、XML スキーマをフィールドに割り当てるを選択します。

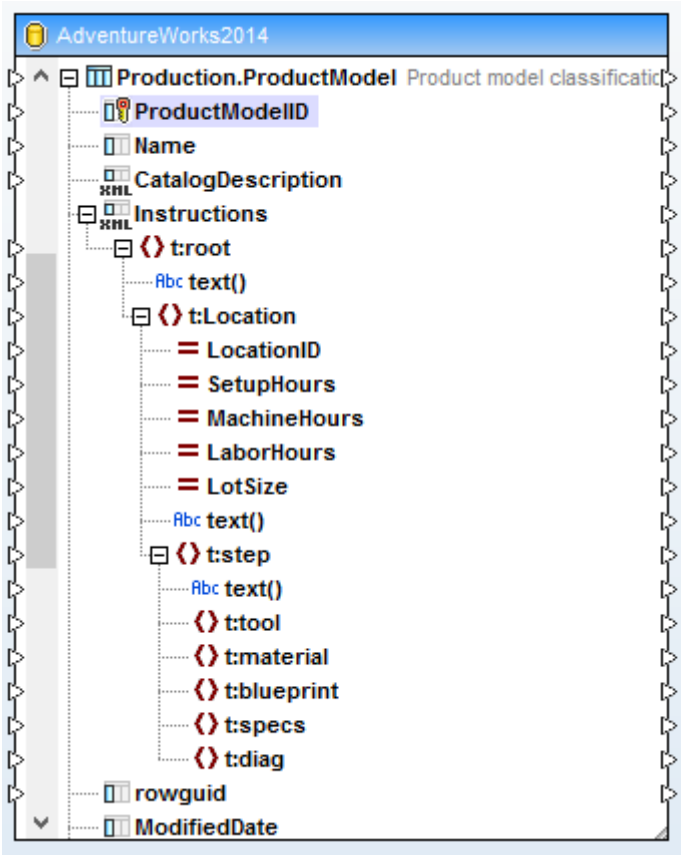


この特定の例の中では、**Instructions** フィールドに直接データベースからスキーマを割り当てます。これを行うには、データベース オプションの横の**Production.ManuInstructionsSchemaCollection** アイテムを選択し、「OK」をクリックします。



XML フィールドの構造がコンポーネント上に表示されます。このフィールドへ、またはこのフィールドへ接続を描く(およびマップを作成する)ことができます。

(C) 2015-2021 Altova GmbH



6.2.7.2 例:XML データを SQLite フィールドに書き込む

この例は、複数のXML ファイルからデータを読み込み、 SQLite データベースに書き込むMapForce マッピングを作成する必要な手順について説明します。マッピングの目的は、それぞれのソースXML ファイルのために、 SQLite データベース内に新規のデータベースを作成することです。それぞれのレコードは、TEXT フィールドとしてXML ドキュメントを保管します。

使用されているサンプルは、以下のパスで見つけることができます: **<マイドキュメント>\Altova\MapForce2021\MapForceExamples\Tutorial** ファイル名は、以下のとおりです:

マッピングデザインファイル	• **XmlToSqliteField.mfd**
ソースXML ファイル	• **bookentry1.xml** • **bookentry2.xml** • **bookentry3.xml**
検証に使用されるXML スキーマ	• **books.xsd**
ターゲット SQLite データベース	• **library.sqlite**

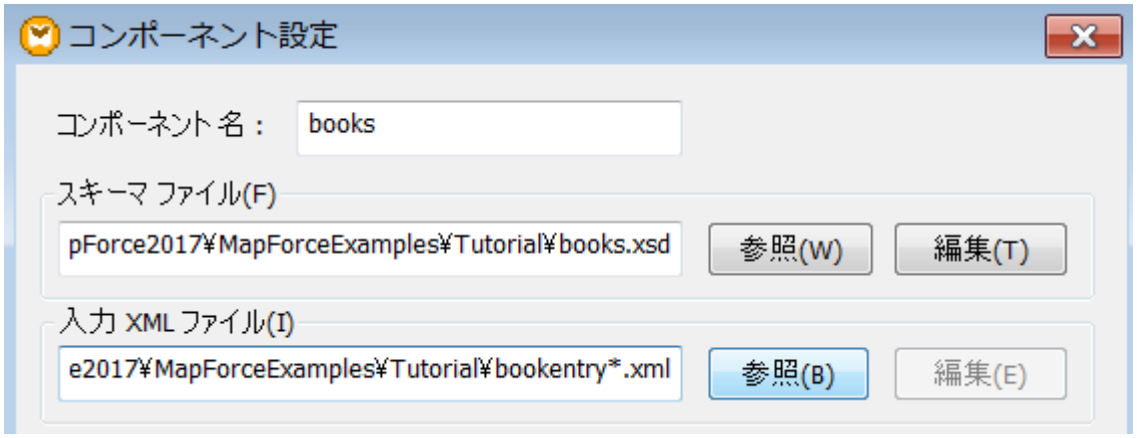
マッピングのゴールを達成するには、以下のステップを踏みます:

1. XML コンポーネントを追加し、複数のファイルから読み込むように構成します。

2. SQLite データベースコンポーネントを追加し、XML スキーマをターゲット TEXT フィールドに割り当てます。
3. マッピング接続の作成とデータベース「アクションの挿入」を構成します。

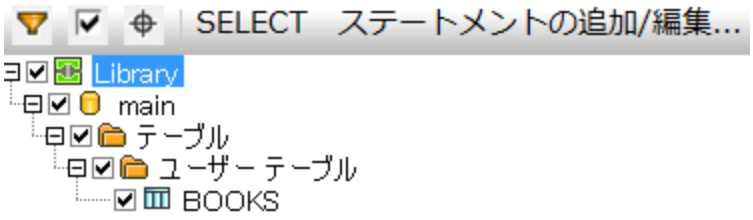
ステップ 1: XML コンポーネントの追加

1. **挿入** メニューから **XML スキーマ/ファイル**をクリックし、**<マイドキュメント>\Altova\MapForce2021\MapForceExamples\Tutorial** ディレクトリ内の**books.xsd** スキーマを参照します。XML ファイルを提供するようにプロンプトされると、「**スキップ**」をクリックします。ルート要素を選択するようにプロンプトされると、**Books** を選択します。
2. コンポーネントヘッダーをダブルクリックして、**入力 XML ファイル**ボックス内に**bookentry*.xml** を入力します。これにより、MapForce にソースディレクトリ内の、"bookentry-" から名前が始まるすべての XML ファイルから読み込むように命令します。この技術に関する詳細については、複数の入力または出力ファイルを動的に処理するを参照してください。



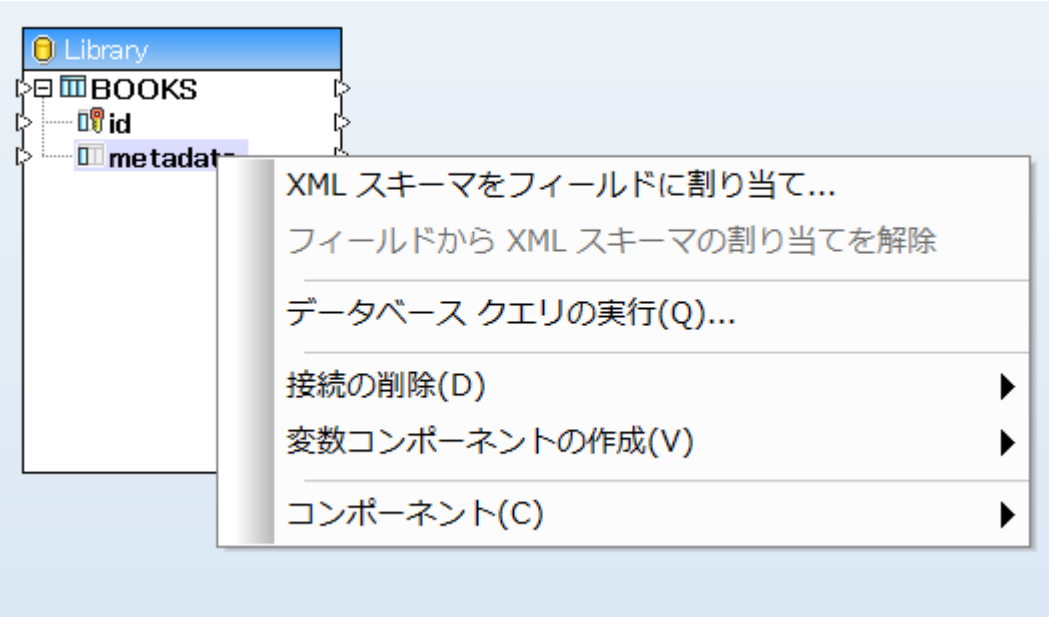
ステップ 2: SQLite コンポーネントを追加する

挿入 メニューから、**データベース**をクリックし、ウィザードに従い、**<マイドキュメント>\Altova\MapForce2021\MapForceExamples\Tutorial** ディレクトリの**Library.sqlite** データベースファイルに接続します（既存の SQLite データベースに接続するも参照してください。）。データベースオブジェクトを選択するように問われると、`BOOKS` テーブルを選択します。

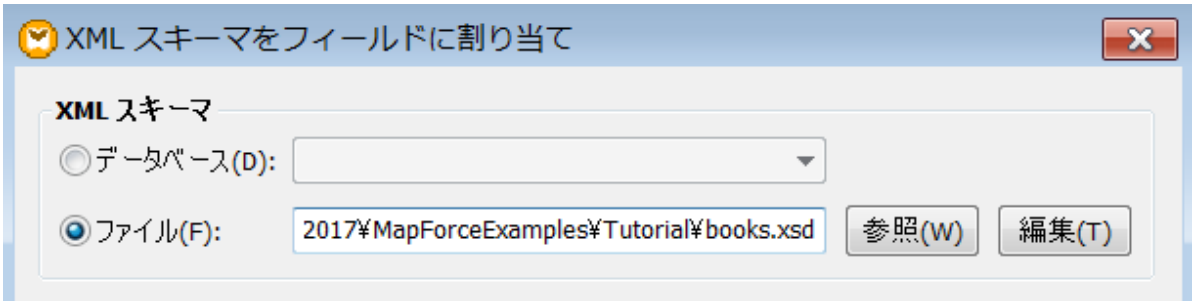


XML コンテンツが書き込まれるデータベースフィールドは、`metadata` と呼ばれます。このフィールドに XML スキーマを割り当てるには、右クリックして、コンテキストメニューから **XML スキーマをフィールドに割り当てる**を選択します。

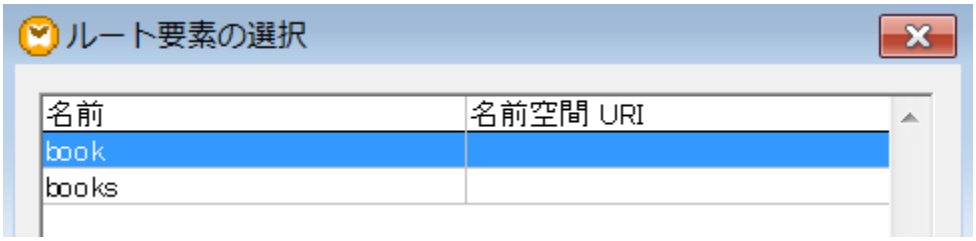
(C) 2015-2021 Altova GmbH



このチュートリアルで、`etadata` フィールドに割り当てられているスキーマは、ソース XML ファイルを検証するために使用されているものと同じです。「参照」 をクリックし、<マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥Tutorial¥ ディレクトリから **books.xsd** スキーマを選択します:

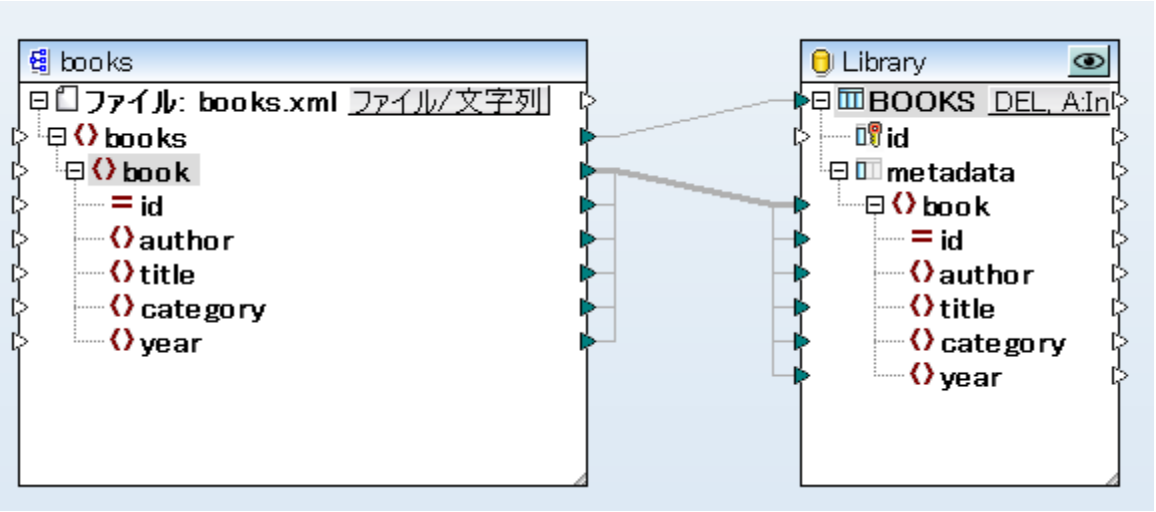


books.xsd スキーマには、次のグローバル宣言を持つ2つの要素があります: `book` と `books`。この例では、`book` をデータベースに書き込まれる XML のルート要素として設定します。選択をクリックして、`book` をルート要素として選択します。



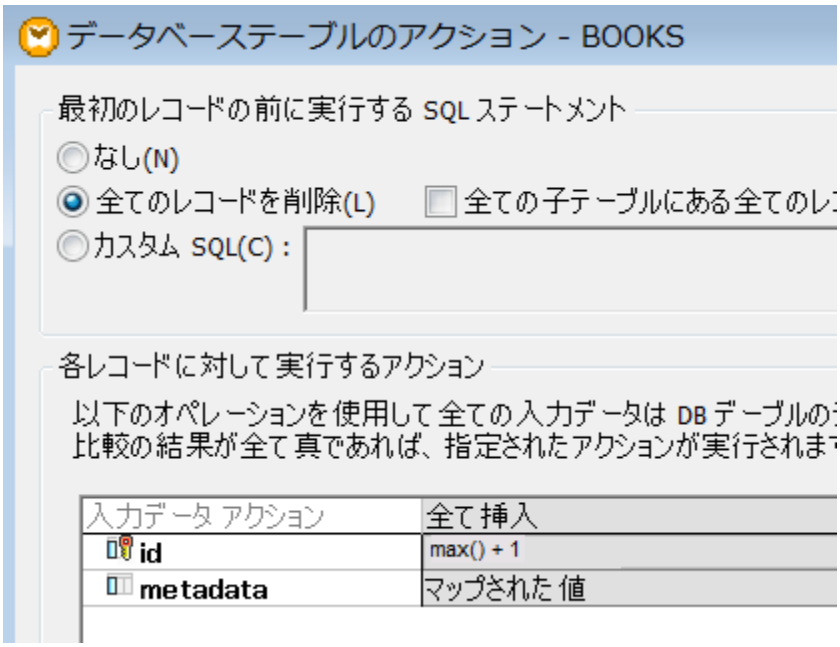
ステップ 3: マッピング接続の作成とデータベース「アクションの挿入」の構成

マッピング接続の作成 は、以下の通りです:



上に示されるように、ソースとターゲットが同じスキーマと子要素の名前を使用するため、 book から book への接続は、"すべてをコピー" 接続です。このような接続に関する詳細については、 すべてコピー接続を参照してください。

一番上の接続 (books から BOOKS) は、ソース内のそれぞれの book 要素を反復し、新規のレコードを BOOKS テーブルに書き込みます。データベースコンポーネントの A:In ボタンをクリックして、データベースが、下で表示されるように設定を更新するように設定してください。



すべてのレコードを削除 オプションは、レコードを挿入する前に BOOKS テーブルのすべてのコンテンツを削除するように MapForce に命令します。

すべて挿入 アクションは、データベース INSERT クエリが行われるように指定します。フィールド metadata は、マッピングから提供される一方、フィールド id は、データベース自身から生成されます。

実行する前にマッピングが保存されていることを確認してください。

(C) 2015-2021 Altova GmbH

マッピングを実行し、生成された出力を確認するには、**出力** タブをクリックします。このアクションはデータベースをすぐに更新しないことに注意してください。生成されたデータベーススクリプトを実行する準備が整うと、メニューコマンド **出力| SQL スクリプトの実行** を選択します。（または ツールバーボタンをクリックします）。

6.2.7.3 例: IBM DB2 XML 型列からデータを抽出する

このサンプルはXML 型のIBM DB2 データベース列からのデータの抽出方法、および、ターゲット CSV ファイルへの書き込みの方法について説明しています。また、条件に応じてXML コンテンツを抽出するするために、SQL 内に埋め込まれたXQuery ステートメントの使用方法についても説明されています。このサンプルは、テーブルを作成し、テーブルを表示する許可が存在する、IBM DB2 データベースへのアクセスが必要です。

XML データが実際に含まれるデータベースを準備します。使用中のデータベースに固有のデータベース管理ツール、またはMapForce で直接おこなうことができます。以下のステップを行ってください：

1. 新規マッピングを作成し、**DBQuery** タブをクリックします。
2. 「**クイック接続**」（ ）クリックして、ウィザードステップに従い、新規のデータベース接続を作成します（次を参照してください：データベース接続のサンプル）。
3. SQL エディターに次のテキストを貼り付けます。このSQL クエリは、`ARTICLES` という名前のデータベーステーブルを作成し、データと共に表示します。

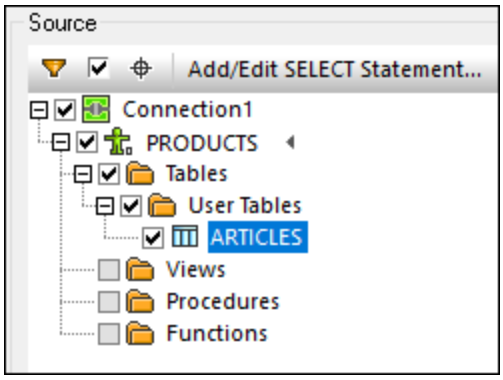
```
-- Create the table
CREATE TABLE
    ARTICLES (
        id INTEGER NOT NULL,
        article XML ) ;
-- Populate the table
INSERT INTO ARTICLES VALUES
   (1, '<Article>
      <Number>1</Number>
      <Name>T-Shirt</Name>
      <SinglePrice>25</SinglePrice>
   </Article>'),
(2, '<Article>
      <Number>2</Number>
      <Name>Socks</Name>
      <SinglePrice>230</SinglePrice>
   </Article>'),
(3, '<Article>
      <Number>3</Number>
      <Name>Pants</Name>
      <SinglePrice>34</SinglePrice>
   </Article>'),
 (4, '<Article>
      <Number>4</Number>
      <Name>Jacket</Name>
      <SinglePrice>5750</SinglePrice>
   </Article>');
```

4. 「**実行**」（ ）ボタンをクリックします。クエリの実行の結果がクエリの結果ウィンドウに表示されます。クエリの実行に成功すると、4つの行が新規に作成されたテーブルに追加されます。

次に、条件に応じて作成された`ARTICLES` テーブルからXML データを抽出するマッピングを作成します。目的は、`ARTICLES` 列から100 よりも高額な商品のみを抽出することです。

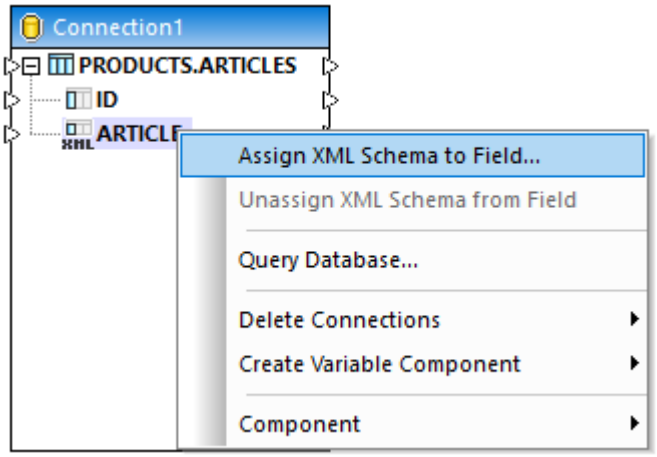
ステップ 1: データベースを追加する

1. マッピングペインに切り替えるために「マッピング」タブをクリックします。
2. 「挿入」メニューから「データベース」をクリックし、ウィザードステップに従い、データベースに接続します。
3. データベースオブジェクトを選択するようにプロンプトされると、以前に作成されている ARTICLES テーブルを選択します。



ステップ 2: フィールドに XML 型スキーマを割り当てる

1. コンポーネントの ARTICLE アイテムを右クリックし、フィールドに XML 型スキーマを割り当てる をコンテキストメニューから選択します。

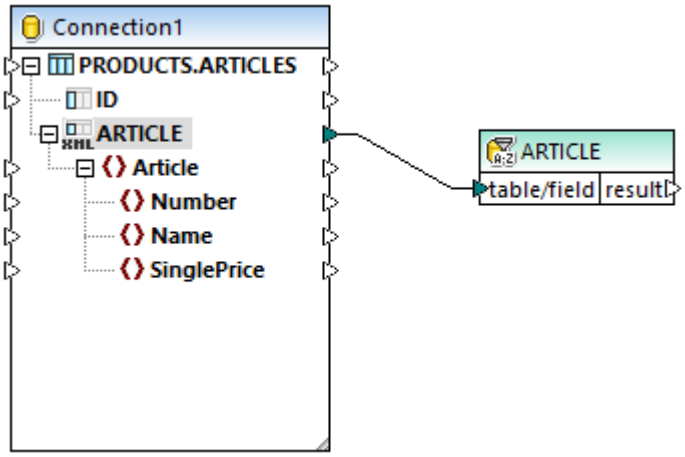


2. 「ファイル」を選択して、次のスキーマを参照します: **<マイドキュメント >\Altova\MapForce2021\MapForceExamples\Tutorial\DB2xsd.xsd**。

ステップ 3: SQL WHERE/ORDER コンポーネントを追加する

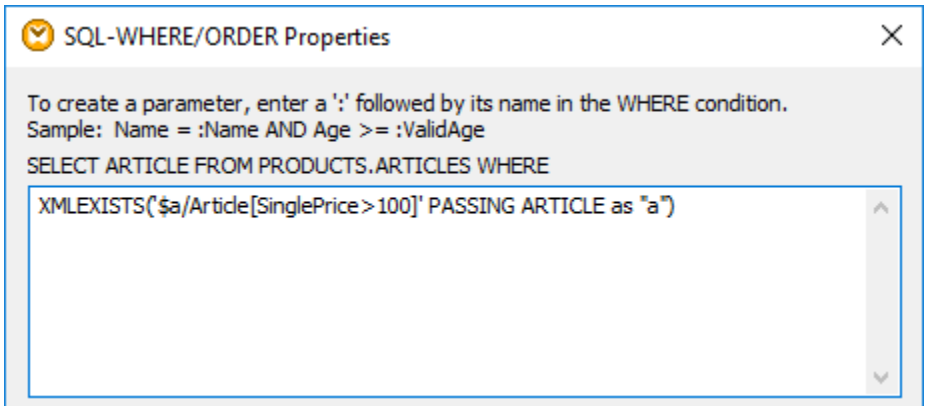
1. 「挿入」メニューから**SQL WHERE/ORDER** をクリックします。
2. ARTICLE XML 型の列 SQL WHERE/ORDER の入力に接続します。

(C) 2015-2021 Altova GmbH



3. SQL WHERE/ORDER プロパティダイアログボックス内に次のテキストを入力します:

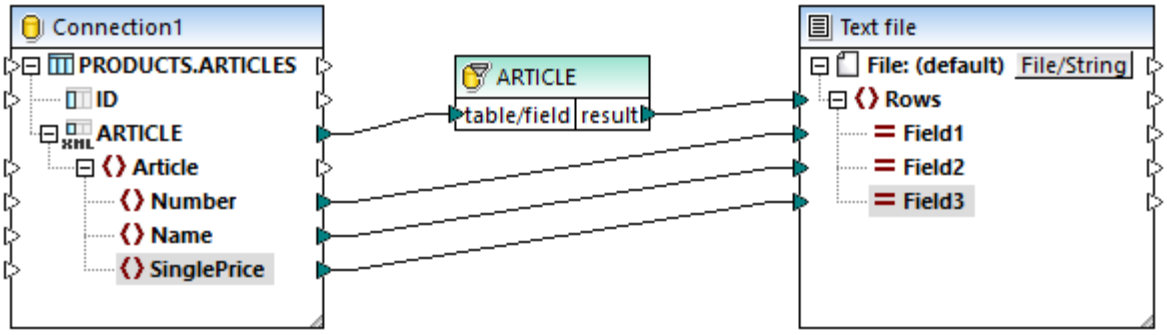
```
XMLEXISTS('$a/Article[SinglePrice>100]' PASSING ARTICLE as "a")
```



上のテキストは、SQL クエリの"WHERE" の部分を示しています。マッピングのランタイムでは、ダイアログボックス上で表示されている"SELECT" の部分と結合されています。このステートメントは、`XMLEXISTS` 関数とIBM DB2 データベースに固有の構文を使用しています。

ステップ 4: ターゲット CSV ファイルを追加する

1. 「挿入」メニューから「テキストファイル」をクリックします。
2. プロンプトされると、「標準 CSV のために簡単な処理を使用する...」、「継続する」を選択します。
3. 「フィールドの追加」3回クリックして、商品番号、名前、価格をそれぞれ保管する3つのフィールドを追加します。他の設定を保留します。
4. 下に示されるようにマッピング接続を描きます。



マッピングの結果を、出力 タブ。をクリックして、プレビューすることができます。予想通り、出力内に100 より先高額の価格の商品のみが出力に表示されます。

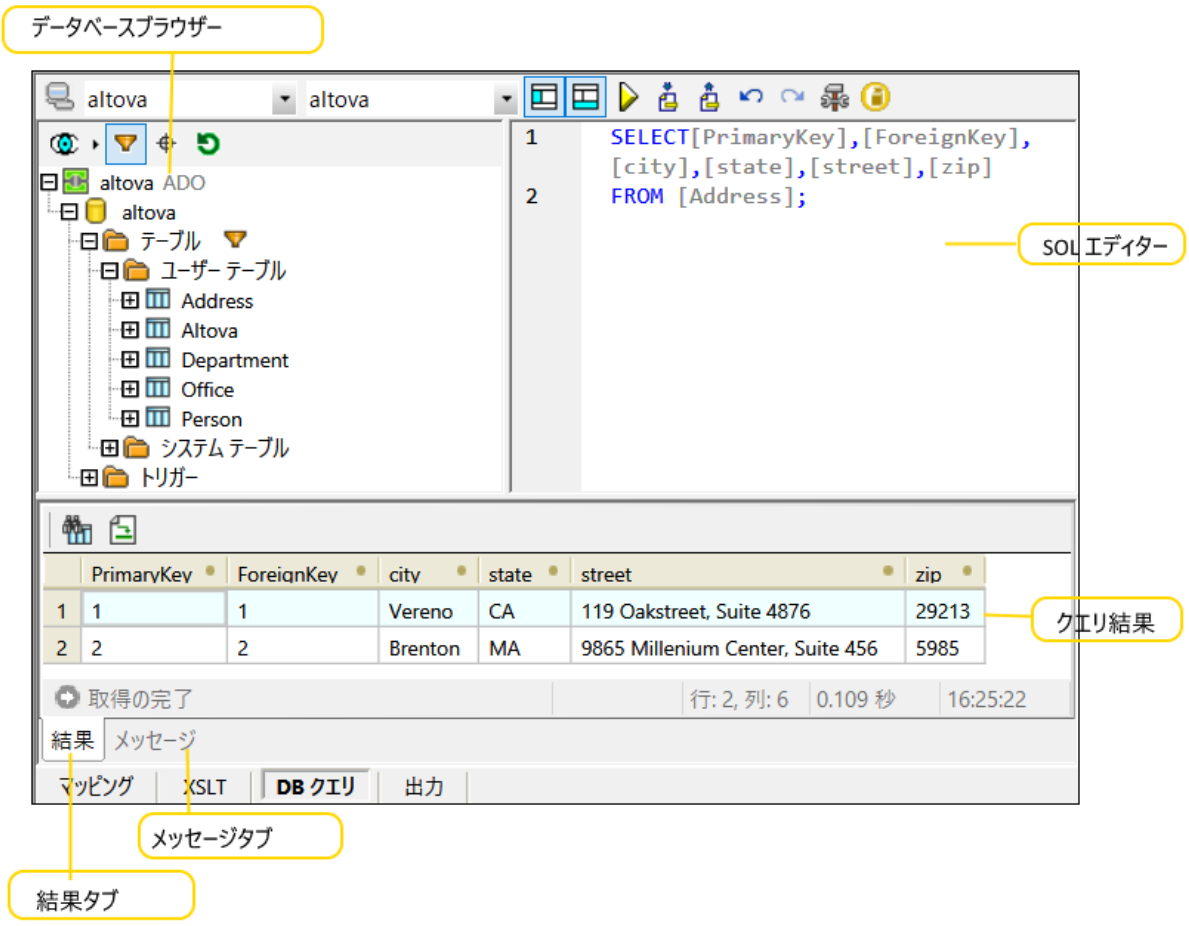
```
1       2,Socks,230
2       4,Jacket,5750
```

6.2.8 データベースの参照とクエリ

MapForce にはマッピング処理から独立しデータベースをクエリすることのできる専用のデータベースクエリ(またはDB クエリ)ペインが搭載されています。このような直接的クエリは、マッピング *.mfd ファイルには保存されず、 MapForce から直接参照またはデータベースのコンテンツを変更することのできる便利な方法が与えられます。

個別のDB クエリ ペインは、現在アクティブなマッピングのそれぞれに存在します。DB クエリ ペイン内で異なるデータベースへの複数のアクティブな接続を作成することができます。DB クエリ ペインから作成された接続は、マッピングの一部ではなく、グローバルリソースとして定義しない限り、MapForce を閉じた後には保管されません。

(C) 2015-2021 Altova GmbH



このタブのウィンドウは以下のパーツから構成されます:

- **データベースブラウザー** にはデータベーステーブルに対する接続情報が表示されます。
- **SQL エディター** ではSQL クエリを作成することができます。
- **結果タブ** にはクエリの結果がテーブル形式で表示されます。
- **メッセージタブ** には警告ならびにエラーメッセージが表示されます。

データベースクエリウィンドウの上部には接続コントロールが用意されており、作業用データベースや接続ならびにデータベーススキーマを定義することができます。

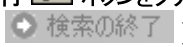
6.2.8.1 データベースの選択と接続

クエリするそれぞれのデータベースに対して、接続を作成する必要があります。マッピングが既にデータベースコンポーネントを含む場合、**DB クエリ** ペインの上のエリアにある既存のデータベース接続(デフォルトでは接続はオフライン)を選択することができ、データベースオブジェクトを探索し、クエリを実行することができます。

マッピングがデータベースコンポーネントを含まない場合、または新規のデータベースに接続する場合は、**クイック接続** ( ) をクリックし、新規のデータベース接続を作成するウィザードに従います(サンプルを参照) 。接続の内の1つがグローバルリソースとして定義されている場合、グローバルリソースから既存のデータベース接続を選択することができます (グローバルリソースを参照) 。

データベースに接続すると、次のメソッドを使用してデータベースクエリを作成することができます:

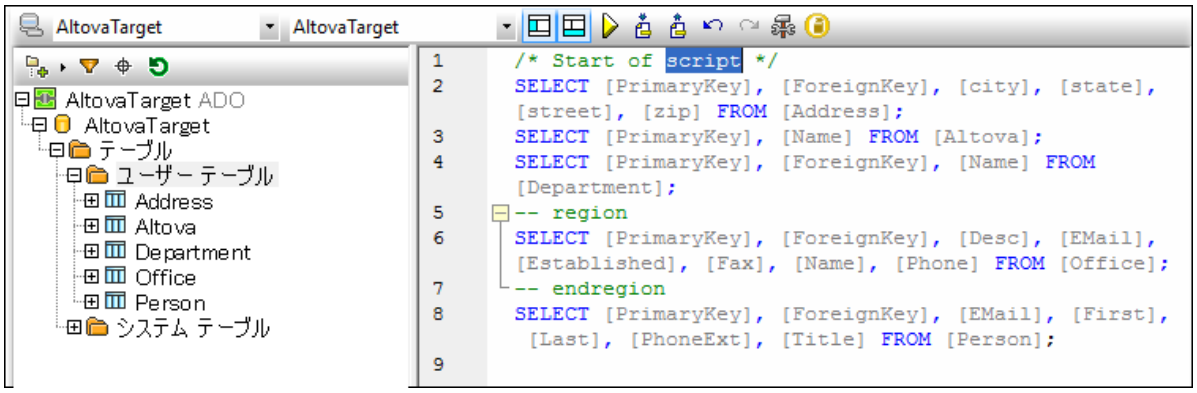
- SQL クエリを SQL エディターペインから既存の SQL ファイルインポートする。
- クエリを SQL エディターペイン内で作成する。
- データベースブラウザーペイン内のオブジェクトを右クリックして、クエリを生成します。(通常、SELECT)。

SQL エディターペイン内でクエリを実行する準備が整ったら、実行 ボタンをクリックします。データベースデータはテーブルの様式で結果タブに表示されます。ステータスバーは検索の終了したメッセージ()を、およびクエリの結果に関連する適切な情報を表示します。

検索の終了メッセージが表示されると、検索結果を検索、並べ替え、クリップボードにコピーすることができます(クエリの結果のコピー、並べ替え、および検索 を参照)。

6.2.8.2 SQL ステートメントの作成と編集

SQL エディターは、SQL ステートメントを作成または実行するために使用されます。既存の SQL スクリプトからロードされ自動的に生成された、または手動で作成されたすべての SQL ステートメントが表示されます。SQL エディターは自動補完、領域、またはラインブロックコメントをサポートします(次を参照してください: 自動補完)。



SQL エディターツールバーで、以下のアイコンを使用することができます:

	ブラウザーの切り替え: ブラウザーペインの表示/非表示を選択します。
	結果の切り替え: 結果ペインの表示/非表示を選択します。
	クエリの実行 (F5): このボタンをクリックすることで、現在選択されている SQL ステートメントが実行されます。複数のステートメントが表示されていて、そのうちどれも選択されていない場合、全てのステートメントが実行されます。
	元に戻す: SQL ウィンドウで行われた変更を、任意の回数だけ元に戻すことができます。
	やり直し: それまで元に戻した変更を再度入力することができます。元に戻すコマンドと併用することで、履歴内にある変更を確認することができます。
	SQL ファイルのインポート: SQL ファイルを SQL エディターにて開き、実行することができます。

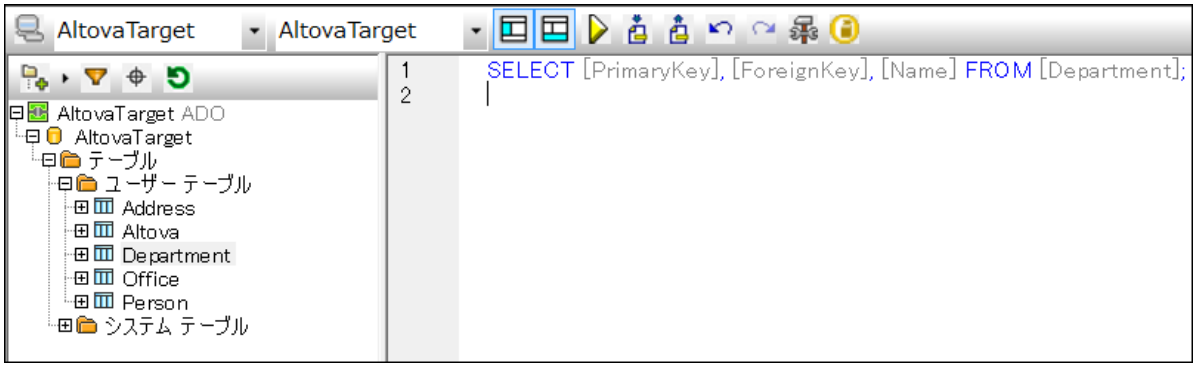
	SQL ファイルのエクスポート: SQL クエリを保存して、将来も使用できるようにします。
	現在の SQL スクリプトを DatabaseSpy で開く: DatabaseSpy を起動し、SQL エディターウィンドウに表示されているスクリプトを開きます。
	オプション: オプションダイアログボックスを開き、MapForce 全般ならびに SQL エディターに関する設定を変更することができます。

6.2.8.2.1 SQL ステートメントの生成

SQL ステートメントスクリプトからロードされるはデータベースブラウザーから自動的に生成されることができ、また手動で入力することもできます。

データベースブラウザーから SQL SELECT ステートメントを生成するには、以下を行います:

- データベースブラウザー内の(テーブルまたはビューなどの) データベースオブジェクトまたはフォルダーをクリックし、データベースオブジェクトをブラウザーペインから SQL エディターへドラッグします。



- データベースブラウザーからデータベースオブジェクトを右クリックして、「**SQL エディターで表示 | 選択**」を選択します。ブラウザーペインに表示されているデータベースオブジェクトを右クリックして、コンテキストメニューからオプションを選択

SQL ステートメントを手動で作成する:

1. SQL エディター内に SQL ステートメントを入力します。自動補完が自動的に生じる場合は、ステートメントの入力中にドロップダウンリストが表示されます。

2. カーソルの上と下キーを使用して、提案ヒントを選択し、ハイライトされたオプションを挿入するために「Enter」を押します(以下も参照してください: SQL 自動補完の提案)。

6.2.8.2.2 SQL ステートメントの実行

SQL エディターに表示される SQL ステートメントは、データベースに対して使用することができ、即時結果を確認することができます。SQL クエリと影響を受けた行は「DB クエリ」ペインの内「メッセージ」ペインに表示されます。

複数の SQL ステートメントが SQL エディターに表示される場合、選択されたステートメントのみが実行されます。個々の SQL ステートメント を以下のように選択することができます:

- 左マウスボタンをクリックして、カーソルを特定のステートメントにドラッグします。
- SQL エディター内のライン番号をクリックします。
- 特定のステートメントをトリプルクリックします。

SQL ステートメントの実行:

1. SQL エディターにて SQL ステートメントを入力または選択します（ SQL ステートメントの生成を参照 ）。
2. 「クエリの実行」 ボタンをクリックします。

6.2.8.2.3 SQL スクリプトのインポートとエクスポート

SQL エディターウィンドウに表示されている SQL を保存することで、スクリプトを将来再利用することができます。

SQL エディターペインに表示されている内容をファイルへ保存する:

- 「SQL ファイルのエクスポート」アイコン をクリックして、保存する SQL スクリプトの名前を入力します。

過去に保存した SQL ファイルを開く:

- 「SQL ファイルのインポート」アイコン をクリックして、SQL ウィンドウに表示する SQL ファイルを選択します。

6.2.8.2.4 SQL コメントの追加と削除

SQL エディターでは、ステートメントやステートメントの一部、またはステートメントのグループをコメントアウトすることができます。コメントアウトされたステートメントは SQL スクリプトの実行時にスキップされます。

テキストの領域でコメントアウトを行う:

1. ステートメントまたはステートメントの一部を選択します。
2. 選択されたステートメントを右クリックし、「ブロックコメントを挿入/削除」を選択します。

(C) 2015-2021 Altova GmbH

ラインでコメントアウトを行う:

- コメントアウトを行う位置を右クリックして、「**行コメントを挿入/削除**」を選択します。カーソルがある位置から、その行の最後までがコメントアウトされます。

ブロックコメントまたはラインコメントを削除する:

1. コメントアウトされた箇所を選択します。ラインコメントの削除を行う場合、コメントの前に挿入されているコメントマーク(—) だけを選択するだけでも同様の効果を持ちます。
2. 右クリックして、「**ブロック(または行)コメントを挿入/削除**」を選択します。

6.2.8.2.5 ブックマークを使用する

ブックマークを使用することで、スクリプト内にあるアイテムをマークすることができます。

ブックマークを追加する:

- ブックマークを行いたい行を右クリックして、コンテキストメニューからブックマークの挿入/削除を選択します。

```
1
2     SELECT [PrimaryKey], [ForeignKey], [city], [state],
      [street], [zip] FROM [Address];
3     SELECT [PrimaryKey], [Name] FROM [Altova];
4     SELECT [PrimaryKey], [ForeignKey], [Name] FROM
      [Department];
5     SELECT [PrimaryKey], [ForeignKey], [Desc], [EMail],
      [Established], [Fax], [Name], [Phone] FROM [Office];
6     SELECT [PrimaryKey], [ForeignKey], [EMail], [First],
      [Last], [PhoneExt], [Title] FROM [Person];
7
```

ブックマークアイコン がブックマーク余白に表示されます。

ブックマークを削除する:

- ブックマークを削除したい行をクリックして、コンテキストメニューからブックマークの「**挿入/削除**」を選択します。

ブックマーク間のナビゲーションを行う:

- 右クリックにて表示されるコンテキストメニューから、「**次のブックマークへ移動**」を選択することで次のブックマークへカーソルを移動します。
- 右クリックにて表示されるコンテキストメニューから、「**前のブックマークへ移動**」を選択することで前のブックマークへカーソルを移動します。

全てのブックマークを削除する:

- 右クリックして「**全てのブックマークを削除**」を選択します。

6.2.8.2.6 領域の挿入

領域とはテキストの選択範囲のことで、ユニット単位でSQL スクリプトのマーキングや宣言を行うことができます。領域は展開ならびに縮退することができ、SQL スクリプトの一部を隠すことができます。ある領域を、別の領域内部にネストすることもできます。

領域を挿入すると、選択されたテキストの上部に－region コメントが挿入され、展開/縮退アイコンが表示されます。

メモ －region コメントの後に領域の説明をしたテキストを挿入することで、領域の名前を変更することができます。"region" という名前を変更しないでください（例: －region DB2query）。

領域を作成する:

1. 領域として宣言したいステートメントをSQL エディターにて選択します。
2. 右クリックして表示されるコンテキストメニューから**領域の挿入**を選択します。マークされた箇所が領域になり、展開ならびに縮退を行うことができます。

```
1
2     SELECT [PrimaryKey], [ForeignKey], [city], [state],
      [street], [zip] FROM [Address];
3     -- region
4     SELECT [PrimaryKey], [Name] FROM [Altova];
5     SELECT [PrimaryKey], [ForeignKey], [Name] FROM
      [Department];
6     -- endregion
7     SELECT [PrimaryKey], [ForeignKey], [Desc], [EMail],
      [Established], [Fax], [Name], [Phone] FROM [Office];
8     SELECT [PrimaryKey], [ForeignKey], [EMail], [First],
      [Last], [PhoneExt], [Title] FROM [Person];
9
```

3. + または－ボックスをクリックして、領域の展開または縮退を行うことができます。

領域を削除する:

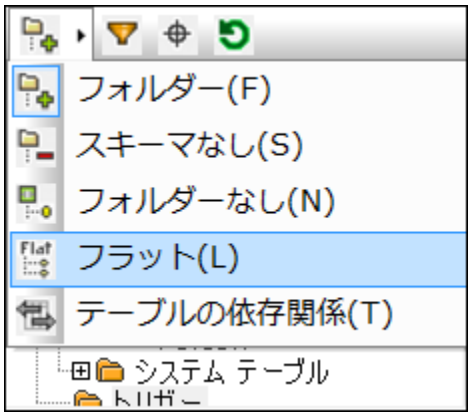
- －region ならびに－endregion コメントを削除します。

6.2.8.3 データベースオブジェクトの参照

1つまたは複数のデータベースに接続している場合、「**データベースブラウザー**」ペインは、各データベース内のテーブル、ビュー、プロシージャ、などの詳細を含むオブジェクトの概要を表示します。XML をサポートするデータベースでは、「**データベースブラウザー**」で個別のフォルダー内の登録されたXML スキーマも追加して表示します。

(C) 2015-2021 Altova GmbH

データベースオブジェクトをナビゲートするには、「**データベースブラウザー**」ペインに複数の定義済みのデータベース表示レイアウトがあります。データベースブラウザーの上の部分で、定義済みのレイアウトを使用することができます。



レイアウトを選択するには、**フォルダーのレイアウト**（ ）ドロップダウンボタンをクリックして、リストからエントリを選択します。このボタンは選択されたレイアウトを変更することに注意してください。

- **フォルダー** レイアウトは、データベースオブジェクトをオブジェクト型をベースにしたフォルダーを階層構造ツリーに整理します。これはデフォルトの設定です。
- **スキーマなし** レイアウトは、フォルダーレイアウトに類似していますが、データベーススキーマは存在しません。ですから、テーブルはデータベーススキーマ別に整理されません。
- **フォルダーなし** レイアウトはデータベースオブジェクトをフォルダーを使用せずに階層構造に表示します。
- **フラット** レイアウトは、データベースオブジェクトを最初の階層構造で型別に表示します。例えば、対応するテーブルに列が含まれるのではなく、すべての列が個別の列フォルダーに表示されます。
- **テーブルの依存関係** レイアウトは、テーブルを他のテーブルとの関係に基づき整理します。外部キーを持つテーブル、他のテーブルとの関係を持たないテーブルのためのカテゴリーが存在します。

レイアウトナビゲーションに加え、「**データベースブラウザー**」以下のタスクのために使用することができます:

- SQL ステートメントの生成（ SQL ステートメントの生成 を参照）。
- 表示されるデータベースオブジェクトのフィルターおよび検索（ データベースオブジェクトのフィルターおよび検索を参照 ）。
- "システム" と "ユーザー" テーブルにテーブルを並び替える
- アクティブなデータソースのルートオブジェクトの更新

ユーザーとシステムテーブルにテーブルを並び替える:

- 「**データベースブラウザー**」内の"テーブル" フォルダーを右クリックして、**ユーザーとシステムテーブルの並べ替え**を選択します。

  メモ この関数は、次のレイアウトが選択されていると使用することができます: **フォルダー**、**スキーマなし**、または **フラット**。

アクティブなデータソースのルートオブジェクトの更新:

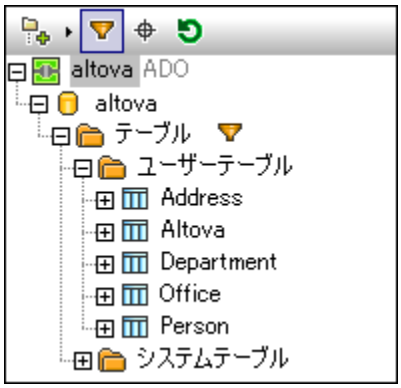
- 「**データベースブラウザー**」の上にある、**更新**（ ）をクリックします。

6.2.8.3.1 データベースオブジェクトのフィルタリングと検索

「**データベースブラウザー**」に表示されるデータベースオブジェクト（スキーマ、テーブル、ビューなど）を名前または名前の一部によりフィルターすることができます。オブジェクトは文字を入力中にフィルターされます。フィルターはデフォルトで大文字と小文字を区別します。フィルターは「No Folders」レイアウトを選択している場合サポートされません。

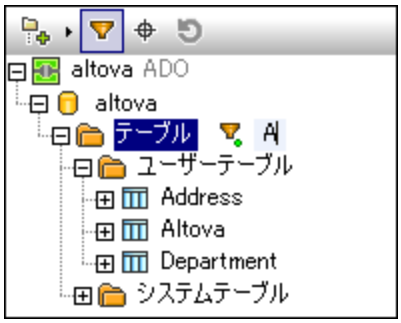
データベースオブジェクトのフィルタリング

1. ツールバーから「**フォルダーコンテンツのフィルターアイコン**」（ ）をクリックします。現在のレイアウトで表示されているフォルダーの隣にフィルターアイコンが表示されます。



2. フィルタリングを行うフォルダーの隣にあるフィルターアイコンをクリックして、ポップアップメニューからフィルタリングオプション（例えば「含む」）を選択します。

3. 空のフィールドがフィルターアイコンの隣に表示されます。検索を行う文字列（例：″G″）を入力します。入力中に検索結果が表示されます。

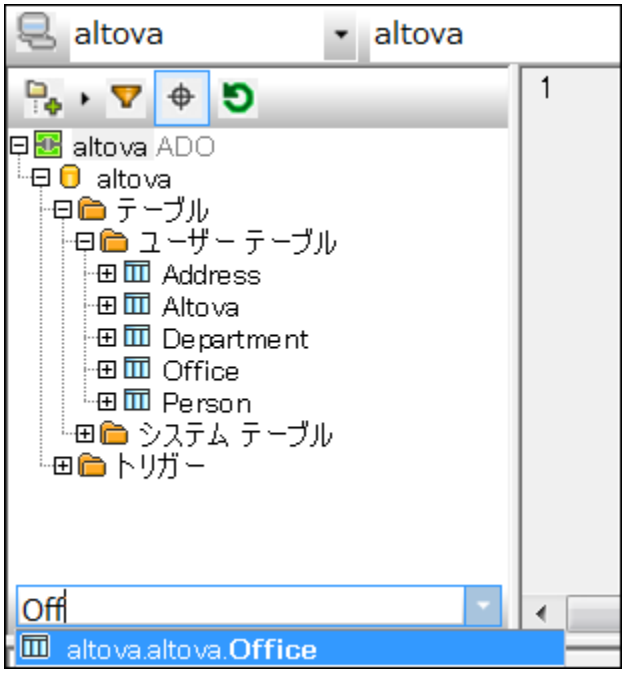


(C) 2015-2021 Altova GmbH

データベースオブジェクトの検索

名前によりデータベースアイテムの検索を行うには、フィルター機能の他にも**オブジェクトロケーター**を使用することができます。

1. ブラウザーペインにて「**オブジェクトロケーター**」アイコン ( ) をクリックします。
2. ドロップダウンリストがブラウザーペイン下部に表示されます。検索を行う文字列を入力します(例: "Off")。



3. 「**データベースブラウザー**」内のリストからオブジェクトをクリックします。

6.2.8.3.2 データベースブラウザー内のコンテキストオプション

データベースブラウザー 内で使用することのできるコンテキストメニューオプションは選択されたオブジェクトにより異なります。例えば

- 「ルート」オブジェクトを右クリックすると、データベースを「**更新**」することができます。
- フォルダーを右クリックすると常に同じオプションが表示されます:「**展開 | 兄弟 | 子**」と「**折りたたみ | 兄弟 | 子**」。
- データベースオブジェクトを右クリックすると、下で説明される「**SQL エディターで表示**」コマンドとサブメニューアイテムが表示されます。

複数のデータベースオブジェクトを選択する場合、「**Shift + クリック**」または「**Ctrl + クリック**」を押します。

メモ SQL ステートメントの構文は、使用するデータベースにより異なります。下の文法は Microsoft SQL Server 2014 に適用することができます。

ルートオブジェクトのために「**SQL エディターで表示**」コンテキストメニューの次のオプションを使用することができます:

- **CREATE**: 選択されたデータベースのルートオブジェクトのために CREATE ステートメントを作成します。例えば
  `CREATE DATABASE [MYDB]`
- **DROP**: 選択されたデータベースのルートオブジェクトのために DROP ステートメントを作成します。例えば
  `DROP DATABASE [MYDB]`

テーブルとビューのために「SQL エディターで表示」コンテキストメニューの次のオプションを使用することができます:

- **SELECT**: ソーステーブルの全ての列からデータを抽出し SELECT ステートメントを作成します。例えば
  `SELECT [DepartmentID], [Name], [GroupName], [ModifiedDate] FROM [MYDB].[HumanResources].[Department]`
- **Name**: 選択されたテーブルの名前を返します。
- **Path**: テーブルのフルパスを以下のフォーマットで返します: ￥
  `DataSourceName.DatabaseName.SchemaName.TableName.`

複数のテーブルを選択した場合、名前またはパスはコンマで区切られ、異なるラインに表示されます。

列のために「SQL エディターで表示」コンテキストメニューの次のオプションを使用することができます:

- **SELECT**: 親テーブルの選択された列からデータを取得する SELECT ステートメントを作成します。例えば
  `SELECT [DepartmentID] FROM [MYDB].[HumanResources].[Department]`
- **Name**: 選択された列の名前を返します。
- **Path**: 列のフルパスを以下のフォーマットで返します:
  `DataSourceName.DatabaseName.SchemaName.TableName.ColumnName.`

複数の列を選択した場合、名前またはパスはコンマで区切られ、異なるラインに表示されます。

制約のために「SQL エディターで表示」コンテキストメニューの次のオプションを使用することができます::

- **Name**: 選択された制約の名前を返します。
- **Path**: 制約のフルパスを以下のフォーマットで返します:
  `DataSourceName.DatabaseName.SchemaName.TableName.ConstraintName.`

複数の制約を選択した場合、 名前またはパスはコンマで区切られ、異なるラインに表示されます。

次のオプションを使用することができます。「SQL エディターで表示」コンテキストメニューからインデックス:

- **Name**: 選択されたインデックスの名前を返します。
- **Path**: インデックスのフルパスを以下のフォーマットで返します:
  `DataSourceName.DatabaseName.SchemaName.TableName.IndexName.`

複数のインデックスを選択した場合、名前またはパスはコンマで区切られ、異なるラインに表示されます。

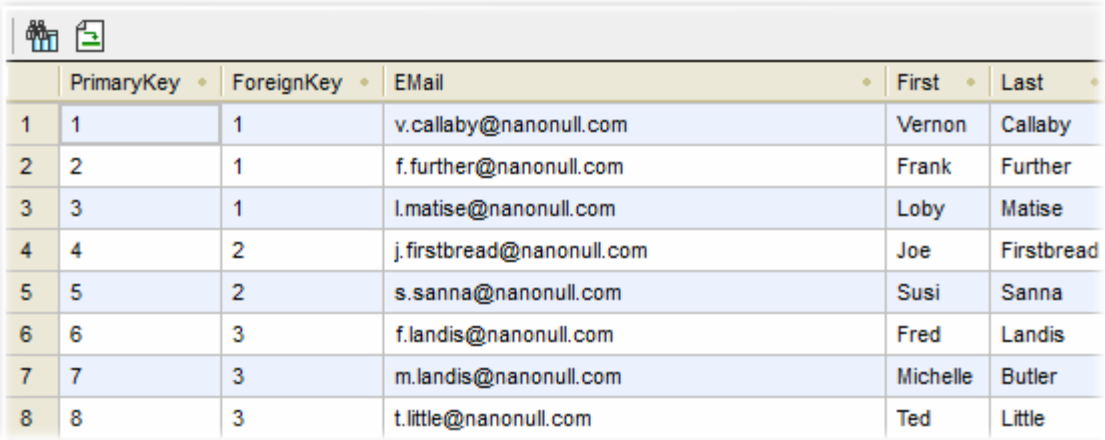
データベースが XML スキーマをサポートする場合、 "XML スキーマ" フォルダーに表示される全てのスキーマのために次のオプションを使用することができます:

- **XMLSpy 内のビュー**: XMLSpy 内でデータベーススキーマを開きます。
- **XML スキーマの管理**: 新規データベース XML スキーマを追加し、既存のデータベース XML スキーマを削除することのできるダイアログボックスを開きます。

(C) 2015-2021 Altova GmbH

6.2.8.4 クエリの結果のコピー、並べ替え、および検索

DB クエリ ペインの結果タブは、データベースクエリの結果として抽出されたレコードセットを表示します。



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

ツールバーにより、結果とSQL ステートメントの間をナビゲートすることができ、クエリの結果内の検索を促進します。

	検索： 結果ウィンドウにある文字列を検索します。F3 キーを押下することで検索を継続することができます。
	ステートメントへ移動: SQL エディターウィンドウへ移動し、その結果を得るために使用されたSQL ステートメントのグループがハイライトされます。

クエリの結果からセルを選択する:

- 列ヘッダーをクリックして、全体の列を選択します。
- 列全体を選択するために、列番号をクリックします。
- 個別のセルをクリックします。複数の選択を行うことができる「Ctrl」 キーを押します。列 またはセルがXML データを含む場合、このデータもコピーすることができます。

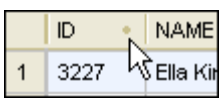
  メモ コンテキストメニュー「**選択 | 行 | 列 | すべて**」もデータの選択に使用する子ができます。

クリップボードに選択されたセルをコピーする:

- 右クリックして、コンテキストメニューから「**選択されたセルをコピーする**」を選択します。

データの並べ替え:

- 並べ替えるために列を右クリックし、「**並べ替え | 昇順 または 降順**」を選択します。
- 列のヘッダー内の並べ替えアイコンをクリックします。



	ID	NAME
1	3227	Ella Kir

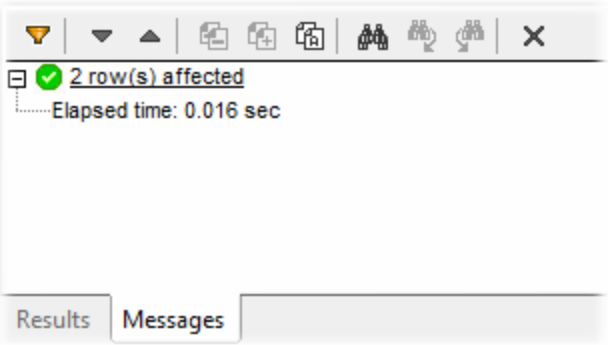
データは並べ替えられた列のコンテンツにより並べ替えられます。

デフォルトの並べ替え順序の復元:

- テーブル内を右クリックし、コンテキストメニューから「**並べ替え | デフォルトの復元**」を選択します。

6.2.8.5 実行されたクエリの状態を確認する

DB クエリ ペインのメッセージ タブは、最後に実行されたSQL ステートメントに関する特定の情報を提供し、また、エラーまたは警告メッセージを報告します。



メッセージ タブのカスタマイズされた異なるフィルターを使用するか、または、「**次へ**」および「**前へ**」ボタンを使用して、データを行から行へと参照することができます。上のボタンは、メッセージをナビゲート、クリップボードへコピー、メッセージの一部を非表示するために使用されます。これらのオプションは、**メッセージ**タブ内を右クリックすると表示されるコンテキストメニュー内でも使用することができます。

	フィルター: 個別のメッセージのパート（**概要**、**成功**、**警告**、**エラー**）を開いた箇所からポップアップメニューを表示のために開きます。「自動挿入」は、SQL ステートメント、またはSQL 構造がSQL エディターにより自動的に挿入されると挿入されるメッセージを指します。「進行状況」メッセージは、データベース接続の結果、およびSQL 解析とデータ構造のロードの結果もを報告します。 「**全てチェックする**」または「**チェックを全て解除する**」をポップアップメニューから選択し、マウスをクリックするだけでこれらのオプションを全て選択する、または選択しないことができます。
	次へ: 次のメッセージに移動してハイライトします。
	前へ: 前のメッセージに移動してハイライトします。
	選択されたメッセージをクリップボードにコピーします。
	選択されたメッセージとその子をクリップボードにコピーします。
	すべてのメッセージをクリップボードにコピーします。
	検索: **検索** ダイアログボックスを開きます。
	前を検索: **検索** ダイアログボックス内で指定された文字列の前の発生まで移動します。
	次を検索: **検索** ダイアログボックス内で指定された文字列の次の発生まで移動します。

(C) 2015-2021 Altova GmbH

	クリア: SQL エディターウィンドウのメッセージタブからすべてのメッセージを削除します。

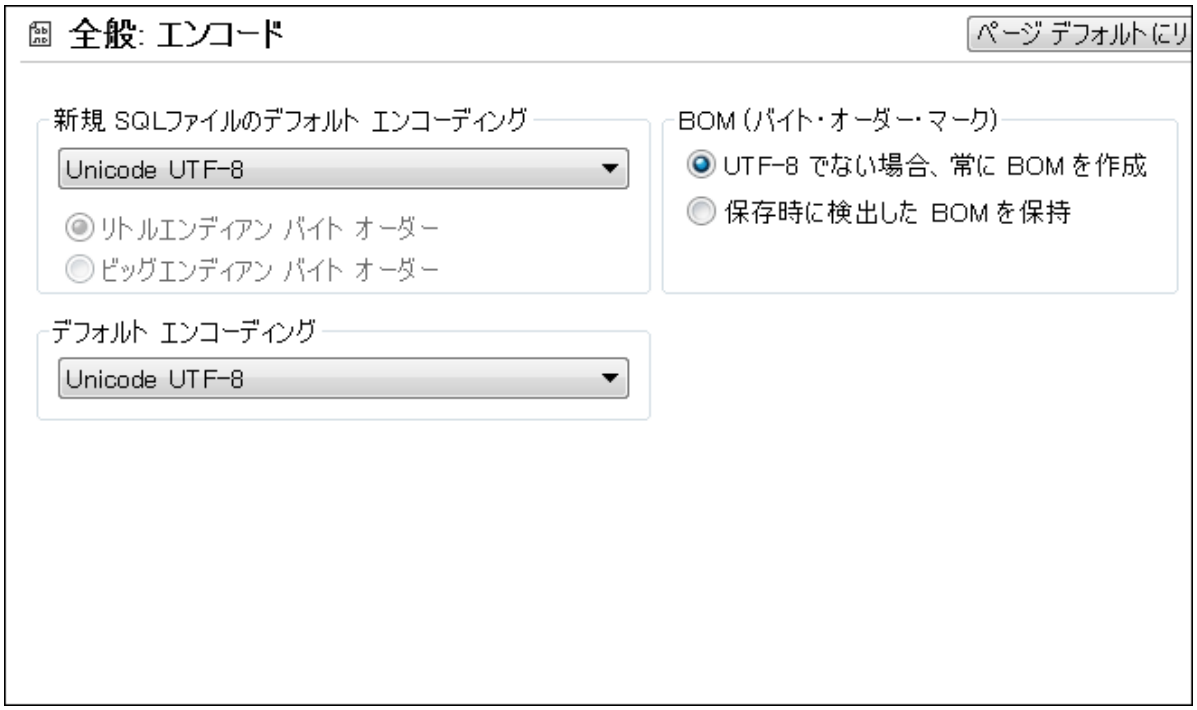
6.2.8.6 データベースクエリ設定

このセクションでは、入力されたSQL ステートメント、またはロードされたSQL エディターに適用することのできるその他の設定の構成についての情報と、クエリが実行された後に表示されるクエリの結果について説明されています。

6.2.8.6.1 SQL ファイルエンコードの設定

作成されたSQL ファイルまたはSQL エディターで開かれたファイルのためのエンコードオプションの指定方法は、以下の通りです:

1. 「DB クエリ」タブをクリックします。
2. ペインの上にある、「オプション」 をクリックして、「全般 | エンコード」をクリックします。



新規のSQL ファイルのためのデフォルトのエンコーディング

新規ファイルで使用するデフォルトのエンコーディングを指定することで、新たに作成されるドキュメントには、そのエンコーディング指定が追加されます。2 または4 バイトのエンコーディング(UTF-16、UCS-2、またはUCS-4)が選択された場合、リトルエンディアンまたはビッグエンディアンを選択することができます。

既存のファイルで使用されているエンコーディングは、勿論そのまま保持されます。

未知のエンコードを使用して SQL ファイルを開く

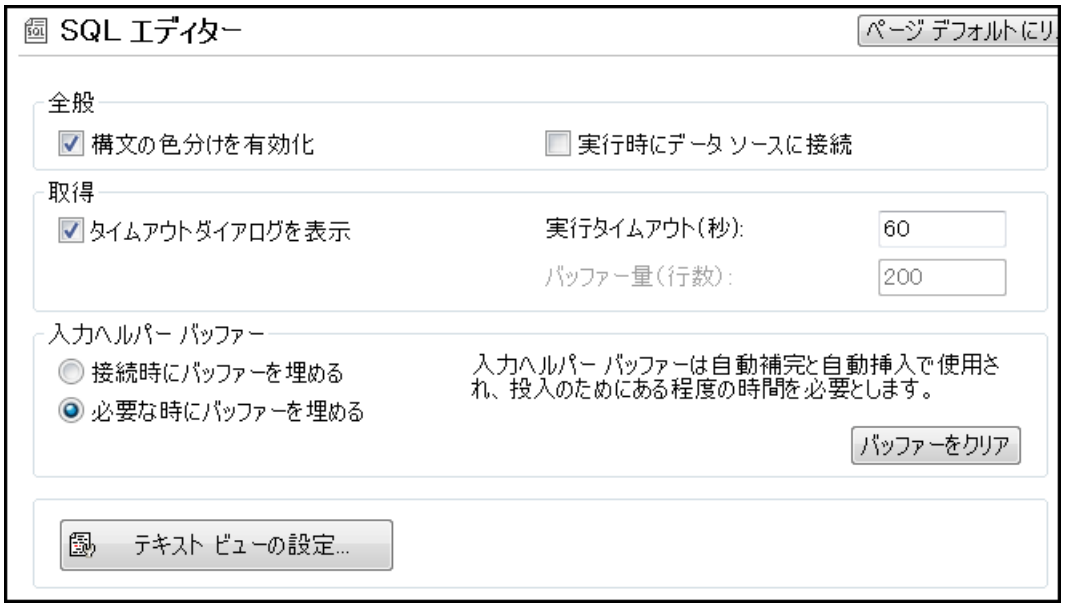
エンコードの指定がない、またはエンコードを検出することのできない SQL ファイルを開く場合使用するエンコードを選択することができます。

メモ エンコード指定のない SQL ファイルは、UTF-8 エンコードで保存されます。

6.2.8.6.2 SQL エディターの一般的な設定

SQL エディターに適用することのできる全般的な設定を以下のように変更することができます:

1. 「DB クエリ」タブをクリックします。
2. ペインの上にある、「**オプション**」 をクリックして、「SQL エディター」をクリックします。



全般

「**構文の色分けを有効化**」により、SQL 構文における要素が異なる色により表示されるようになります。

「**タイムアウト ダイアログを表示**」チェックボックスにより、SQL ファイルが実行される際ならびにデータソースが接続されて以内場合、対応するデータソースへの接続が自動的に行われます。

取得

SQL の実行が許可される最大時間(実行タイムアウト)を秒単位で定義します。

「**タイムアウト ダイアログを表示**」チェックボックスを有効にすることで、タイムアウトが発生した際にタイムアウトの設定を変更することができるようになります。

入力ヘルパー バッファー

SQL ステートメントの入力中の自動補完の提案を有効化するには、「**自動的に開く**」チェックボックスを選択します(次も参照する 自動補完)。

(C) 2015-2021 Altova GmbH

自動補完のための入力ヘルパーバッファーをデータソースに接続する際、または最初に使用する際に入力することができます。バッファーの入力には時間がかかる可能性があります。「バッファーのクリア」ボタンを使用して、バッファーをリセットすることができます。

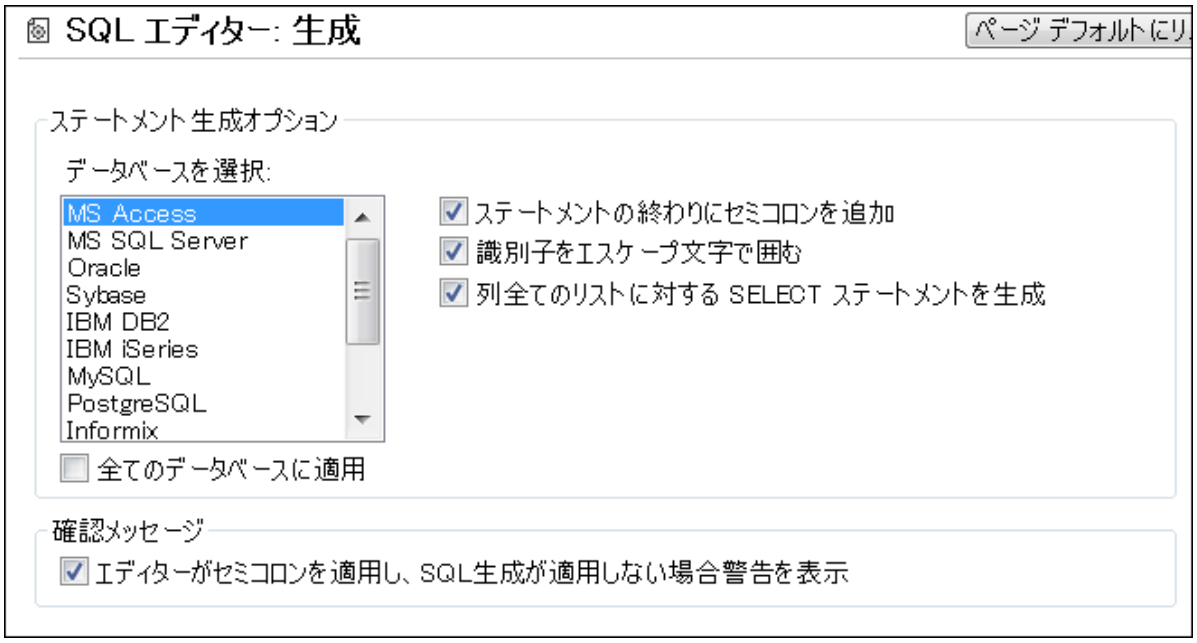
テキストビュー設定

余白やタブ、視覚的な補助やテキストビューナビゲーションのホットキーといったテキストビューの設定を定義することができます。

6.2.8.6.3 SQL ステートメント生成の設定

SQL ステートメントの生成構文を複数のデータベースの型のために指定することができます:

1. 「DB クエリ」タブをクリックします。
2. ペインの上にある、「オプション」 をクリックして、「SQL エディター」をクリックします。



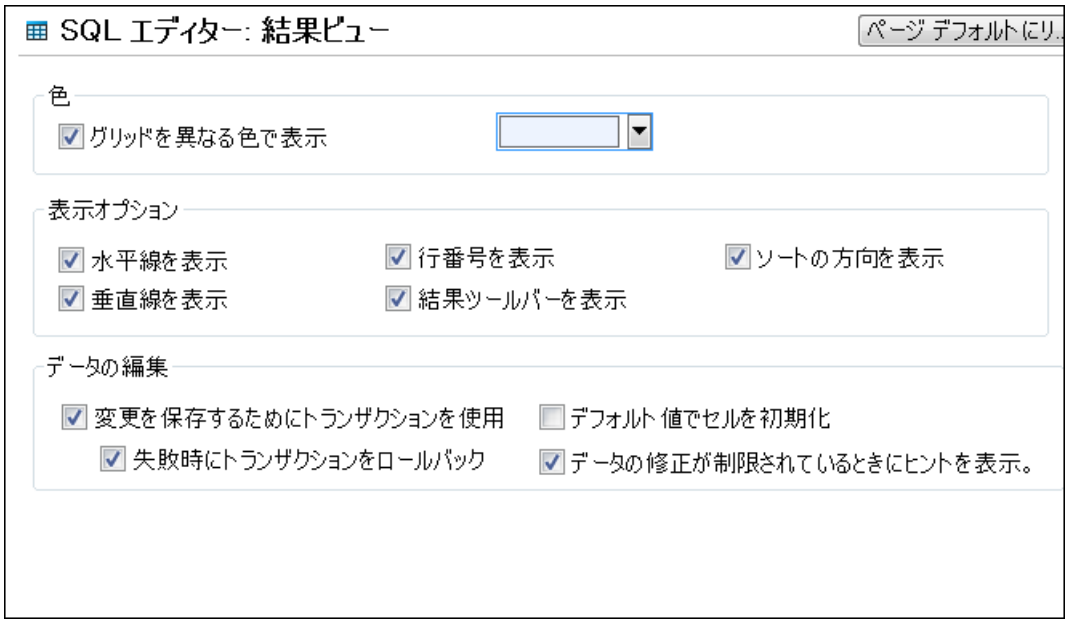
特定のデータベースのための構文の優先順序を定義するには、リストから選択して、右の3つのチェックボックスを有効化または無効化します。

すべての データベースのための一意の構文を定義するには、全てのデータベースに適用を選択して、右の3つのチェックボックスを有効化または無効化します。全てのデータベースのための共通の設定は、JDBC 接続を使用したOracle とIBM DB2 とiSeries データベース内のデータを編集する際、不都合が生じる可能性があることにご注意してください。

6.2.8.6.4 クエリの結果ビューの設定

DB クエリペインの結果タブの外観の構成の方法 は、以下の通りです:

1. DB クエリ タブをクリックします。
2. ペインの上の、オプション をクリックし、「SQL エディター | 結果ビュー」をクリックします。



色を変更してグリッドを表示するを選択して 結果タブを単純なグリッド表示にするか、または色を使った行の表示を行うか指定することができます。

表示オプショングループでは水平ならびに垂直のグリッド線、行番号や**結果ツールバー**などの表示を指定することができます。対応するチェックボックスを有効/無効にすることで、これらの表示/非表示を切り替えることができます。

データの編集グループでは、トランザクションに関する設定や、デフォルトの値でセルを埋めるのか、またヒントの表示を行うかといった設定を行うことができます。

6.2.8.6.5 SQL エディターフォントの設定

SQL エディターに表示されるSQL ステートメントの色とフォントの設定を構成する方法は、以下の通りです:

1. 「DB クエリ」タブをクリックします。
2. ペインの上にある、「**オプション**」 をクリックして、「**SQL エディター | フォント**」をクリックします。

(C) 2015-2021 Altova GmbH

フォント設定リストボックスに表示されているフォント設定はSQL ステートメントの要素になります。SQL エディターに表示されるフォントフェイスやスタイル、サイズ、また色の指定を行うことができます。同一のフォントサイズが全てのテキストに対して使用されます。

個々のテキストタイプごとに指定することのできる設定はスタイルだけです。この機能により構文による色分けが可能になります。「**ページデフォルトにリセット**」ボタンをクリックすることで、オリジナルの設定に復元することができます。

6.2.9 ストアドプロシージャ

ストアドプロシージャとはデータベースサーバー上で動作するプログラムのことで、クライアントアプリケーションから呼び出されることになります。通常これらのプログラムはSQL を拡張した方言により記述され、データベースによっては、Java や.NET CLR といったプログラミング言語による実装をサポートしているものもあります。

ストアドプロシージャの典型的な使用方法には、データベースのクエリと呼び出し元のクライアントにデータを返す、または入力パラメーターの追加検証の後にデータベースを変更するなどが挙げられます。ストアドプロシージャは電子メールの送信などデータベース外で他のアクションを実行することができます。

ストアドプロシージャは、ゼロ、または1つ以上の入力と出力パラメーターをデフォルトの戻り値に加え持つことができます。この結果、MapForce 内では、ストアドプロシージャを複数の方法で呼び出すことができます:

- マッピング上のソースコンポーネントとしてデータを抽出するためにストアドプロシージャを呼び出します。入力パラメーターを取らないプロシージャのためにのみ適用することができます。マッピングが実行されると、プロシージャが呼び出され、レコードセット、または出力パラメーターが返されます。レコードセット、または出力パラメーター、または両方をMapForce によりサポートされるデータ型にマップすることができます。例えば、ストアドプロシージャとデータソースを参照してください。
- パラメーターを持つ関数としてストアドプロシージャを呼び出します。この場合、マッピングから必要とされるすべての入力パラメーターを提供し、MapForce によりサポートされる他のターゲットに返されたレコードセット、または出力パラメーターをマップすることができます。例えば、入力と出力を持つストアドプロシージャを参照してください。
- マッピング上のターゲットコンポーネントとしてストアドプロシージャを呼び出します。一般的なユースケースはデータベースを変更するためにパラメーターを持つストアドプロシージャを呼び出す場合です(例えば、レコードの挿入など)。このアプローチはストアドプロシージャから出力を呼び出す必要がない場合に適しています。また、このアプローチによりエラーが発生した場合ロールバックすることのできるデータベーストランザクション内でストアドプロシージャを実行することができます。 例えば、ターゲットコンポーネント内のストアドプロシージャを参照してください。

(最初に挿入して、更新するなど)特定の順序でストアドプロシージャを呼び出す、またはデータベーステーブル上でアクションを実行する必要がある場合があります。例えば、ストアドプロシージャの出力パラメーターを他のストアドプロシージャにパスする必要がある場合があります。または、ストアドプロシージャにより返されたデータをテーブルからのデータと組み合わせる必要がある場合があります。基になるデータベースがテーブル間の主/外部キーリレーションシップを強制しない場合でも、MapForce 内で定義されているローカルリレーションを使用してこのようなアクションは可能になります。詳細に関しては、ストアドプロシージャとローカルリレーションを参照してください。

メモ MapForce がどのようにストアドプロシージャを実行するか示すために、このチャプターは Microsoft SQL Server 2016 と「AdventureW orks 2016」データベースを使用しています。後者は https://github.com/Microsoft/sql-server-samples/releases/tag/adventureworks からダウンロードすることができます。

サポートに関するメモ

- BUILT-IN 実行エンジン内でのみストアドプロシージャを使用することができます。C++、C#、または Java 内でのコード生成はサポートされていません。
- ユーザー定義型、カーソル型、変数型、多種の"エキゾチック" データベース固有のデータ型(配列、形状、CLR 型) 通常は入力、または出力パラメーター型としてサポートされません。
- プロシージャと関数のオーバーロード(同じ名前と異なるパラメーターを持つルーチンの複数の定義)はサポートされません。
- データベースによっては入力パラメーターにデフォルト値を定義することもできますが、この機能は現在サポートされていません。マッピングにて入力パラメーターへの接続が行われなかったとしても、デフォルト値が使用されることはありません。
- 複数のレコードを返すストアドプロシージャは、ドライバーとデータベース API (ODBC/ADO/ADO.NET/JDBC) の組み合わせによってはサポートされます。固定されたカラム構造を持つ同じ数のレコードセットを返すプロシージャだけがサポートされます。
- 可能な場合、データベースのベンダーにより管理されているネイティブのドライバーの最新のバージョンを使用することが奨励されています。ODBC から ADO へのブリッジ、ODBC から JDBC へのブリッジなどのブリッジドライバーの使用を避けてください。
- データターゲットとして呼び出されるストアドプロシージャのためのデータベーストランザクションを任意で有効化することができます。ターゲットコンポーネント内のストアドプロシージャを参照してください。(入力パラメーターを持たない) データソースとして呼び出されるストアドプロシージャ、または関数同様に呼び出される(入力と出力を持つ)関数のためにはトランザクションはサポートされていません。

次のテーブルリストはデータベース固有のサポートメモをリストしています。

データベース	サポートメモ
Access	• Microsoft Access データベース内のストアドプロシージャには制限された機能のみを持ち、MapForce 内ではサポートされません。
DB2	• MapForce にてサポートされる対象: ストアドプロシージャ、スカラー関数、テーブル値関数。 • MapForce 内で使用されるデータベース API により読み取ることができないため、DB2 ストアドプロシージャからの戻り値はサポートされていません。 • 値関数(RETURNS ROW) はサポートされません。 • エラーと計画を読み取る際に、確認されている JDBC ドライバーの問題を避けるために、少なくとも"IBM_DB2 9.7 Fix Pack 3a" をインストールすることが奨励されます。不足する結果セット行を引き起こす ADO プロバイダーに関する問題をフィックスすることができます。
Firebird	• MapForce にてサポートされる対象: ストアドプロシージャ、テーブル値関数
Informix	• MapForce にてサポートされる対象: ストアドプロシージャ、テーブル値関数。
MariaDB	• MapForce ストアドプロシージャ、スカラー関数によりサポートされています。
MySQL	• MapForce にてサポートされる対象: ストアドプロシージャ、スカラー関数 • MySQL バージョン 5.5. 以降は、ストアドプロシージャに対する完全なサポートを含みます。これ以前のバージョンを使用する場合、MapForce の機能性は制限されます。
Oracle	• MapForce にてサポートされる対象: ストアドプロシージャ、スカラー関数、テーブル値関数。

(C) 2015-2021 Altova GmbH

データベース	サポートメモ
	• **Microsoft OLE DB Provider for Oracle** の代わりにネイティブな Oracle ドライバーの使用が奨励されます。 • Oracle では REF CURSOR 型の出力パラメーターを使用することで結果セットが返されます。この仕様はストアドプロシージャに対して MapForce でサポートされるものの、関数に対してはサポートされません。このため、Oracle ストアドプロシージャに対して常に固定された数と名前が使用されることになります。
PostgreSQL	• MapForce にてサポートされる対象: スカラー関数、row-valued 関数、テーブル値関数。 • PostgreSQL では、関数にて定義された任意の出力パラメーターにより、結果セットのカラムが記述されます。この情報は MapForce にて自動的に使用され、情報取得のための実行やレコードセットの入力を手動で行う必要はありません。refcursor 型のパラメーターはサポートされていません。
Progress OpenEdge	• MapForce にてサポートされる対象: ストアドプロシージャ。
SQL Server	• MapForce にてサポートされる対象: ストアドプロシージャ、スカラー関数、テーブル値関数。 • **Microsoft OLE DB Provider for SQL Server** の代わりに最新の **SQL Server Native Client** ドライバーを使用することが奨励されます。 • SQL Server 2008 にて導入されたいくつかのデータ型 (`datetime2`, `datetimeoffset`) に対して、ADO API は限られたサポートしか提供していません。ADO と SQL Server ネイティブクライアントを使用するなかで、新たな期間型にてデータの切り捨て問題に直面した場合、接続文字列に `DataTypeCompatibility=80` という値を加えるか、ODBC を使用するようにしてください。(`datetime2`, `datetimeoffset`)。 • SQL Server プロシージャには暗黙的な型 `int null` の戻り値パラメーターが含まれており、これらの値をマッピングに使用することもできます。プロシージャにて RETURN ステートメントが省略された場合、戻り値は 0 となります。
SQLite	• SQLite はストアドプロシージャを使用しません。
Teradata	• MapForce にてサポートされる対象: ストアドプロシージャ、マクロ。 • スカラー関数、集計関数、およびテーブル関数はサポートされません。 • 既知の問題: Teradata ODBC ドライバーがプロシージャコールの後、出力パラメーターの値を作成することを拒否する問題。

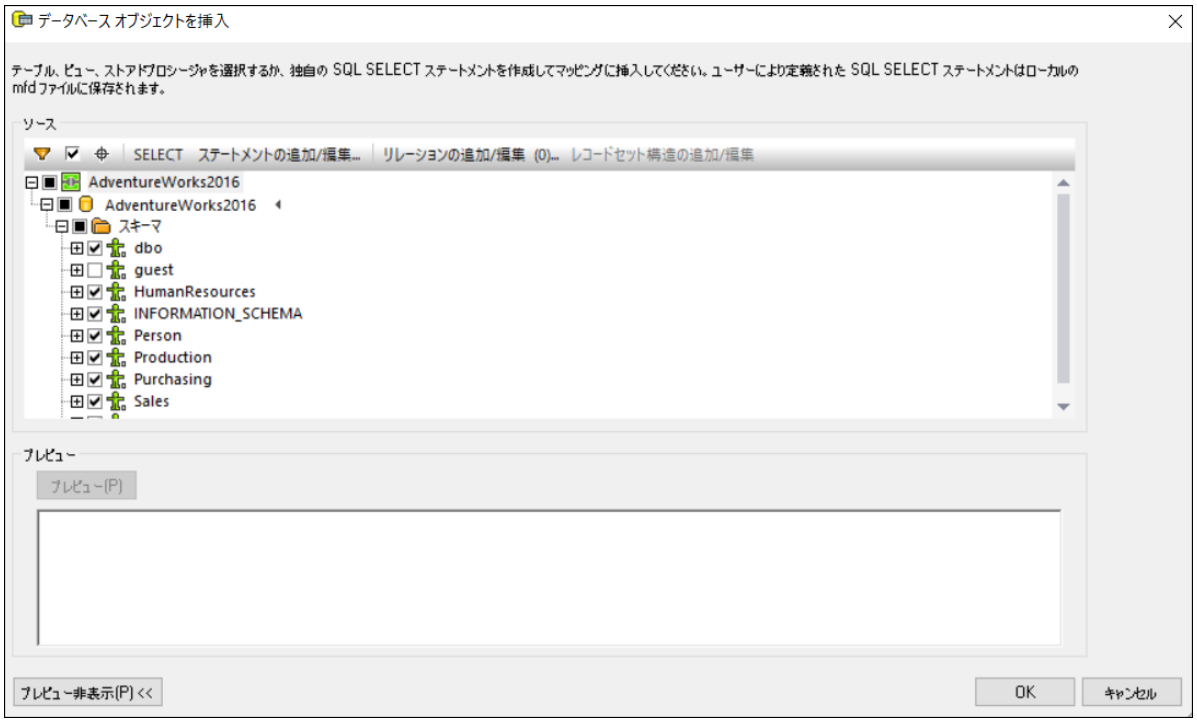
6.2.9.1 マッピングにストアドプロシージャを追加する方法

マッピングエリア内で、ストアドプロシージャは属するデータベースコンポーネントの一部として表示されます。ストアドプロシージャをデータベースコンポーネント上で可視化するには、マッピングにデータベースコンポーネントを追加する際に、下に示されるように、明示的に選択する必要があります。このサンプルでは、SQL Server 上で作動する「AdventureW orks」データベースを接続します。命令は他のデータベースの種類と類似しています。

Oracle データベースの場合、ストアドプロシージャはスタンドアロン、または Oracle パッケージの一部として保管されます。マッピングに両方のカテゴリを追加することができます。パッケージに属するストアドプロシージャまたは関数は下に表示される「データベースオブジェクト」ダイアログボックス対応するパッケージ名の下に表示されます。

マッピングにストアドプロシージャを追加する方法:

1. 以下を行います:
   - 「挿入」メニューで、「データベース」をクリックします。
   - 「データベースの挿入」( ) ツールバーボタンをクリックします。
2. 「データベースオブジェクトの挿入」ダイアログボックスにたどり着くまでデータベースウィザードのステップに従ってください。各データベース型に適応することのできる詳細に関しては、データ接続のサンプルを参照してください。



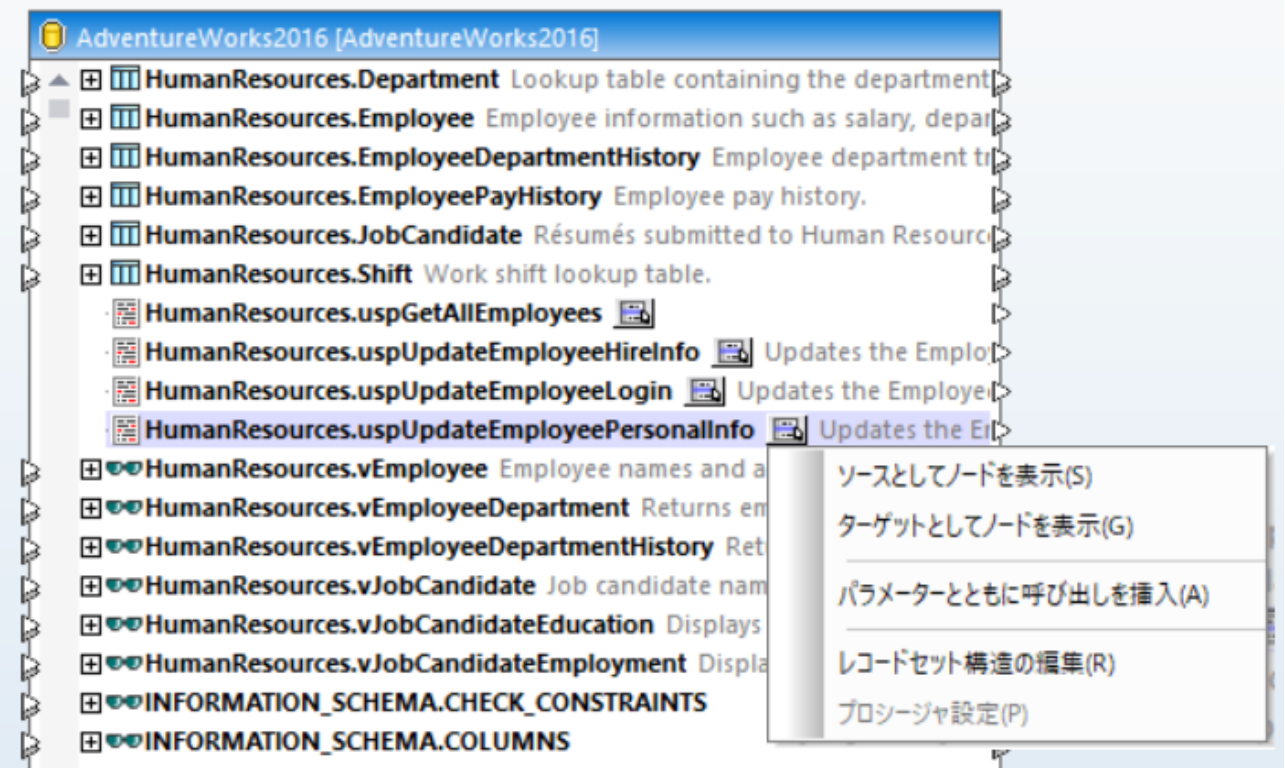
3. マッピング上で表示されるデータベースオブジェクトの横のチェックボックスを選択し、「OK」を選択します。このサンプルでは、全てのテーブル、ビュー、および「HumanResources」スキーマ内で使用することのできるストアドプロシージャが選択されています。

メモ

- データベースコンポーネントのタイトルバーを右クリックし、、をコンテキストメニューから「データベースオブジェクトを追加/削除/編集」を選択して、選択済みのオブジェクトを後に変更することもできます。
- データベースユーザーアカウントは、データベース内のストアドプロシージャをビューして実行する権利が必要です。

データベースコンポーネントがマッピングに追加されます。 アイコンによりストアドプロシージャが識別されていることに注意してください。更に、テーブル、ビュー、プロシージャはデータベースコンポーネント内で、アルファベット順に並べ替えられていることに注意してください。

(C) 2015-2021 Altova GmbH



各ストアドプロシージャの横の「**コンテキストメニューの表示**」 ボタンによりストアドプロシージャがどのように呼び出されるか、または他のプロシージャに関連した設定を構成することができます:

オプション	使用法
ソースとしてノードを表示	データベースからデータを抽出し、MapForce によりサポートされる（XML、テキスト、EDI などの）他のコンポーネントにマップするために、*パラメーター無しの*ストアドプロシージャを呼び出す場合このオプションを選択します。例えば、ストアドプロシージャとデータソースを参照してください。
ノードをターゲットとして表示	ストアドプロシージャの出力を必要としない個所でデータベースを変更、または他の特定のアクションを実行するストアドプロシージャを呼び出す場合、このオプションを選択します。例えば、ターゲットコンポーネント内のストアドプロシージャを参照してください。
パラメーター付きの呼び出しの挿入	MapForce によりサポートされる他のコンポーネントに返されたデータをマップするために、*パラメーターがある*ストアドプロシージャを呼び出す場合、このオプションを選択します。例えば、入力と出力パラメータを持つストアドプロシージャを参照してください。
レコードセット構造の編集	レコードセットを返すストアドプロシージャのために適用することができます。MapForce が返されたレコードセットの構造を決定し、マッピングを表示するストアドプロシージャを実行するために、このオプションを選択します。または、デザインタイムにストアドプロシージャを実行しない場合、レコードセット構造を手動で定義することができます。
プロシージャの設定	（データベースを更新する）「ターゲット」として構成されたストアドプロシージャのためにのみ適用することができます。プロシージャを呼び出す前にカスタム SQL クエリを実行、またはデータベーストランザクションを有効化するなど追加プロシージャ設定を構成するためにこのオプションを選択します。

6.2.9.2 データソースとしてのストアドプロシージャ

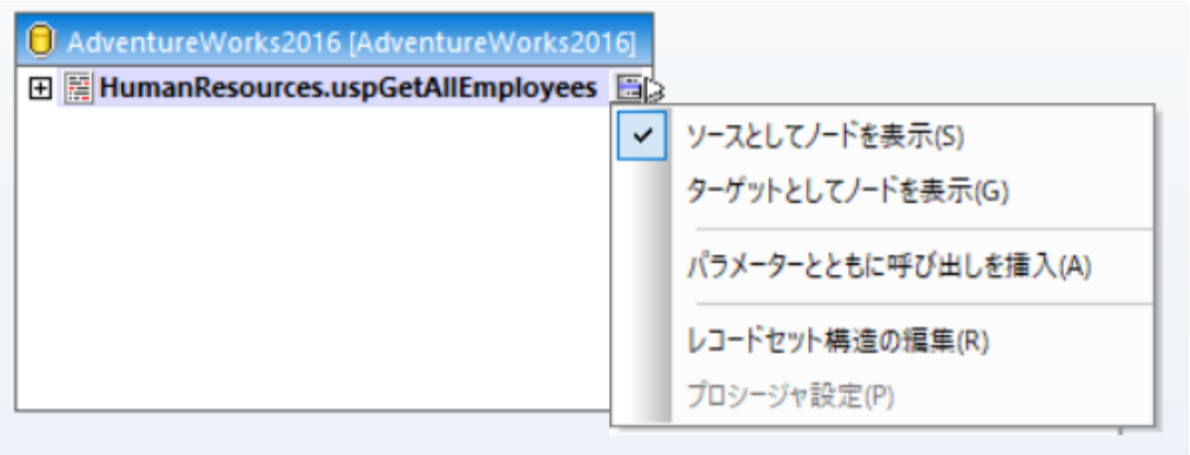
このサンプルは入力パラメーターを取らない、データベースからデータを抽出するだけのプロシージャを呼び出す方法を示しています。このシナリオでは、ストアドプロシージャはマッピングへのソースコンポーネントとしての役割を果たし、MapForce によりサポートされる他のターゲットコンポーネントにより抽出されるデータをマップすることができます。入力パラメーターを持つストアドプロシージャを呼び出す必要がある場合、入力と出力を持つストアドプロシージャを参照してください。

「AdventureW orks」データベース内にデモのストアドプロシージャを最初に作成しましょう。これを行うには、下のスクリプトをデータベースに対して実行します。**Microsoft SQL Server Management Studio** のクエリウィンドウ、またはDatabaseSpy (https://www.altova.com/ja/databasespy) などのツールからこれを行うことができます。どちらの場合でも、データベースユーザーアカウントがストアドプロシージャを作成するパーミッションを有していることを確認してください。

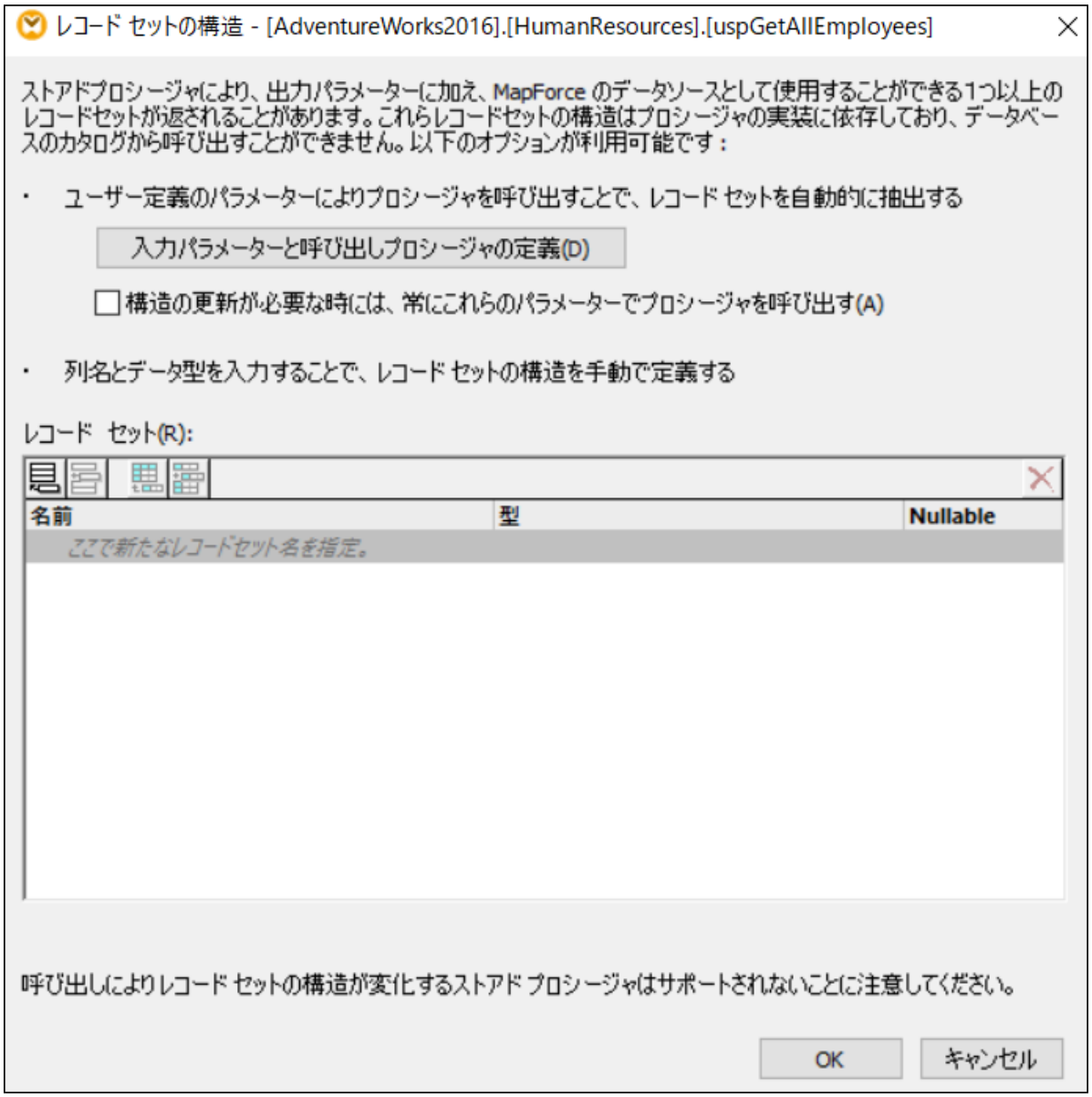
```
CREATE PROCEDURE HumanResources.uspGetAllEmployees
AS
    SELECT LastName, FirstName, JobTitle, Department
    FROM HumanResources.vEmployeeDepartment
```

上のストアドプロシージャは **vEmployeeDepartment** ビューから従業員の情報を返します。次のステップはこのプロシージャから返されたデータを消費するマッピングを作成する方法を表示しています。

1. MapForce から「AdventureW orks」データベースに接続し、マッピングにストアドプロシージャを追加する方法で説明されているとおり、ストアドプロシージャをマッピングに追加します。データベースユーザーアカウントにストアドプロシージャをビューし実行するパーミッションが存在することを確認してください。
2. ストアドプロシージャの横の「**コンテキストメニューの表示**」ボタンをクリックし、「**ソースとしてノードを表示**」を選択します。

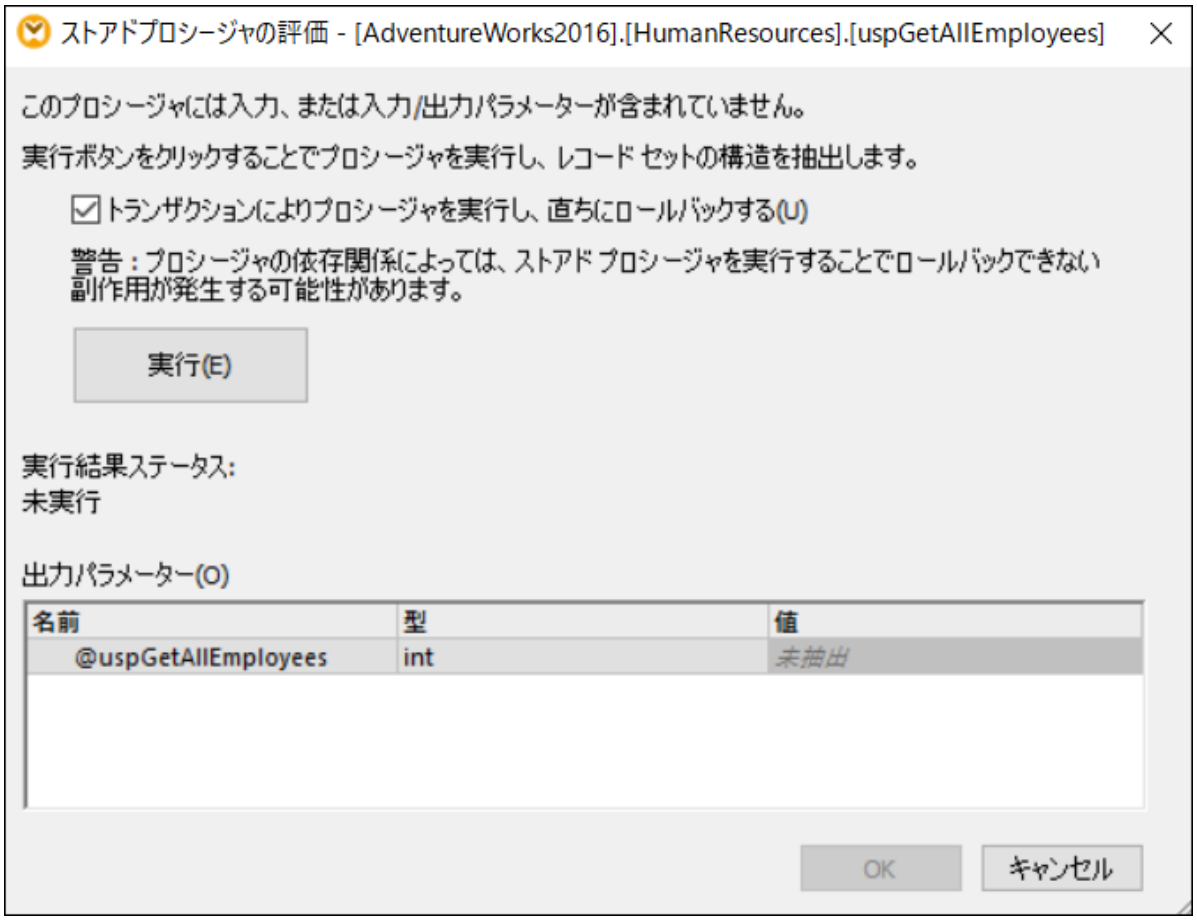


3. 「**コンテキストメニューの表示**」ボタンをもう一度クリックし、「**レコードセット構造の編集**」を選択します。「レコードセットの構造」ダイアログボックスが表示されます。

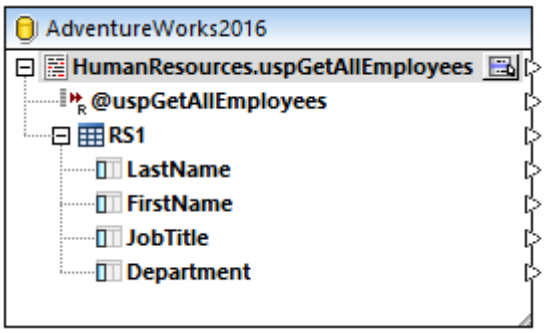


デザイン時にストアドプロシージャを呼び出すとプロシージャの実装により副作用が発生する可能性があります。デザイン時にストアドプロシージャを実行する場合、下記で説明されているとおり、「実行」をクリックしないでください。代わりに、レコードセットを手動で関連する列に追加し、「レコードセット構造」ダイアログボックス内で期待されるレコードセットを定義します。「レコードセット構造」ダイアログボックス内の「**レコードセットの追加**」または「**列の追加**」ボタンを使用します。

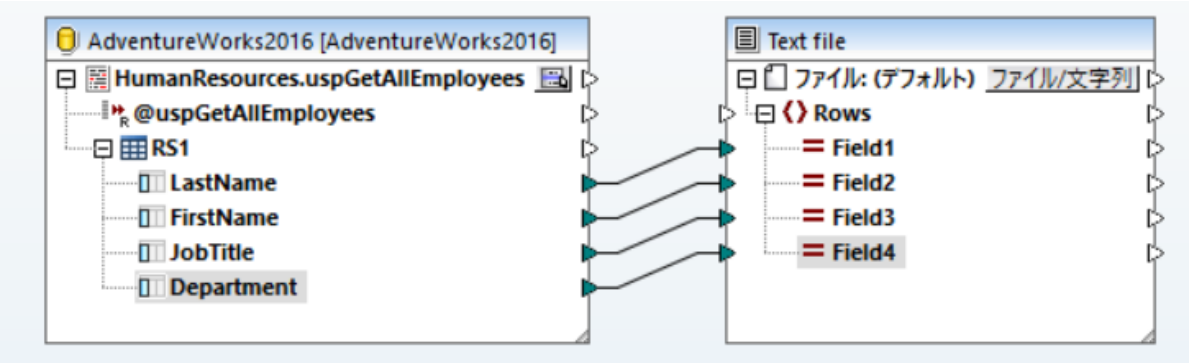
4. 「**入力パラメータと呼び出しプロシージャの定義**」をクリックし、「OK」をクリックします。「ストアドプロシージャの評価」ダイアログボックスが表示されます。



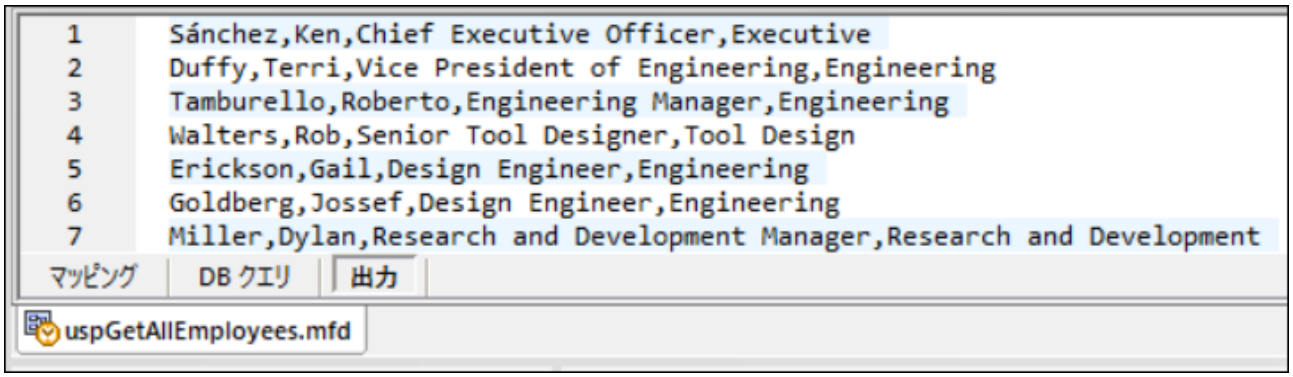
5. 「実行」をクリックし、「OK」をクリックします。レコードセット構造（"RS1"）が「レコードセットの構造」ダイアログボックスとマッピング上に表示されます。



6. この時点では、抽出されたデータが書き込まれるターゲットコンポーネントを追加することができます。このサンプルでは、データはCSV ファイルに書き込まれます。「挿入」メニューから「テキストファイル」をクリックし、データはCSV コンポーネントをマッピングに追加します。詳細に関しては、CSV とテキストファイルを参照してください。



マッピングをプレビューすることができます。「出力」ボタンをクリックして、「出力」ペイン内でマッピングの結果を確認します。例:



6.2.9.3 入力と出力を持つストアドプロシージャ

このサンプルは、入力パラメータを取り、データベースから出力を取得するプロシージャを呼び出す方法を示しています。このシナリオでは、ストアドプロシージャはWeb サービス、または関数と同様に呼び出されています。MapForce によりサポートされる他のターゲットコンポーネントにより抽出されたデータをマップすることができます。

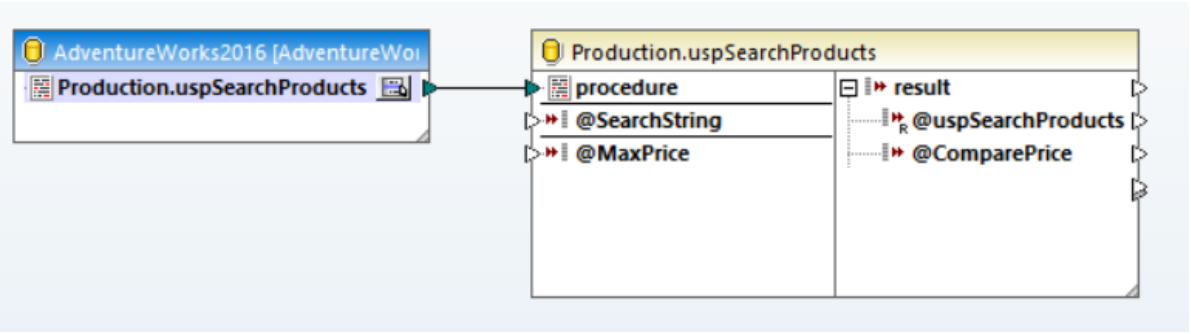
「AdventureWorks」データベース内にデモのストアドプロシージャを最初に作成しましょう。これを行うには、下のスクリプトをデータベースに対して実行します。**Microsoft SQL Server Management Studio** のクエリウィンドウ、またはDatabaseSpy (https://www.altova.com/ja/databasespy) などのツールからこれを行うことができます。どちらの場合でも、データベースユーザーアカウントがストアドプロシージャを作成するパーミッションを有していることを確認してください。

```
CREATE PROCEDURE Production.uspSearchProducts
    @SearchString nvarchar(50)
    ,@MaxPrice money
    ,@ComparePrice money OUTPUT
AS
BEGIN
    SET NOCOUNT ON
    SELECT pr.[Name], pr.ListPrice FROM [Production].[Product] pr
    WHERE pr.[Name] like @SearchString AND  pr.ListPrice < @MaxPrice

    SET @ComparePrice = @MaxPrice
    RETURN @ComparePrice
END
```

上のストアドプロシージャは製品の情報を含むレコードセットを抽出します。入力として2つのパラメーターを取ります: 製品名 (`@SearchString`) を持つ文字列、および、製品の価格 (`@MaxPrice`) の最高額。レコードセットとデフォルトの戻りパラメータに加え、出力パラメーター (`@ComparePrice`) も抽出します。

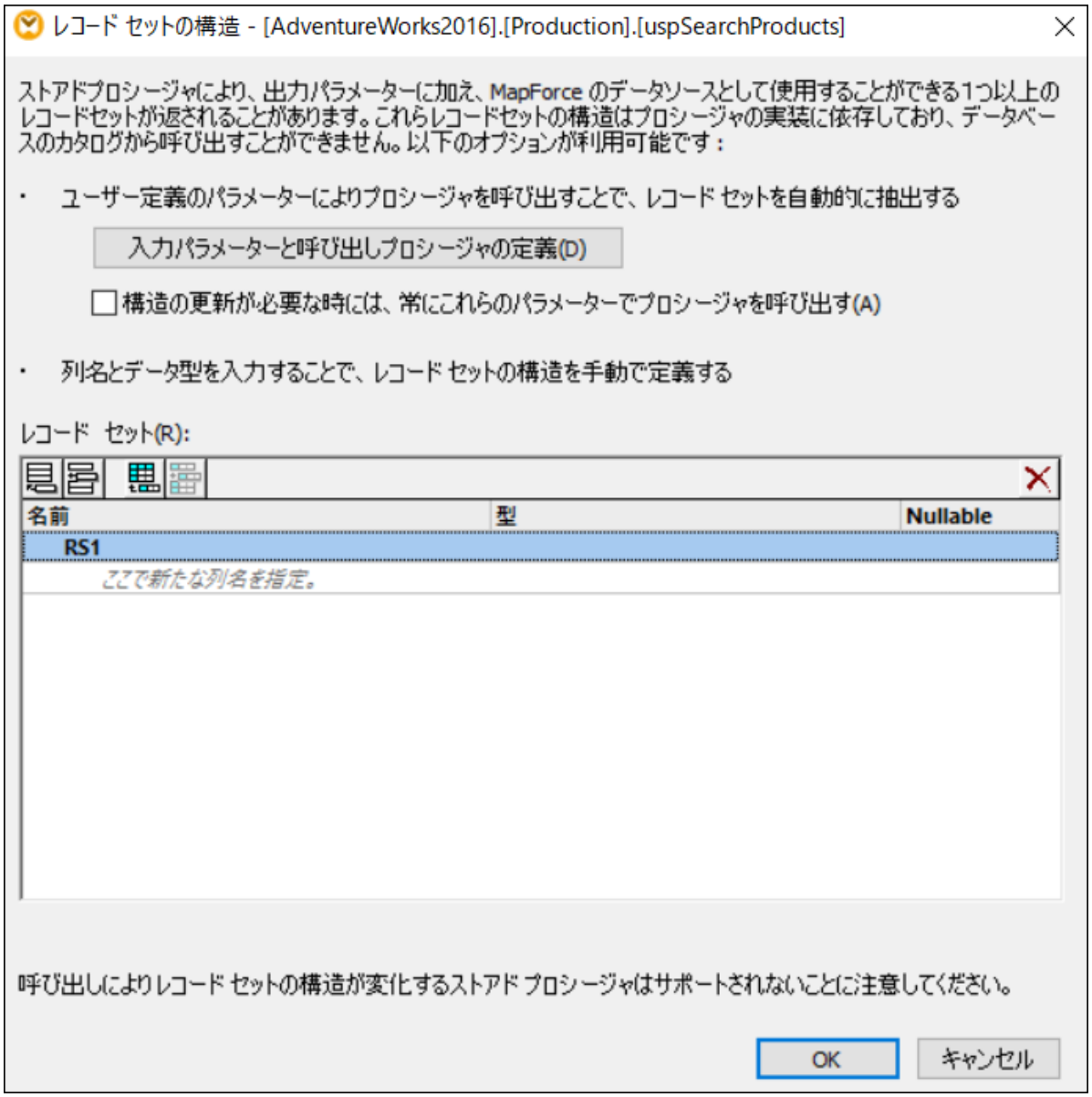
次のステップは、このプロシージャにより返されたデータを使用するマッピングの作成方法を表示します。

1. 「AdventureW orks」データベースを MapForce から接続し、マッピングにストアドプロシージャを追加する方法で説明されているとおり、ストアドプロシージャをマッピングに追加します。データベースユーザーアカウントがストアドプロシージャをビューして実行するパーミッションが存在することを確認してください。
2. ストアドプロシージャの横の「**コンテキストメニューの表示**」ボタンをクリックし、「**パラメーター付きの呼び出しの挿入**」を選択します。ストアドプロシージャが左側に入力パラメーターを表示し、右側は戻り値と出力パラメーターを含むマッピングの個別のコンポーネント内に表示されます。



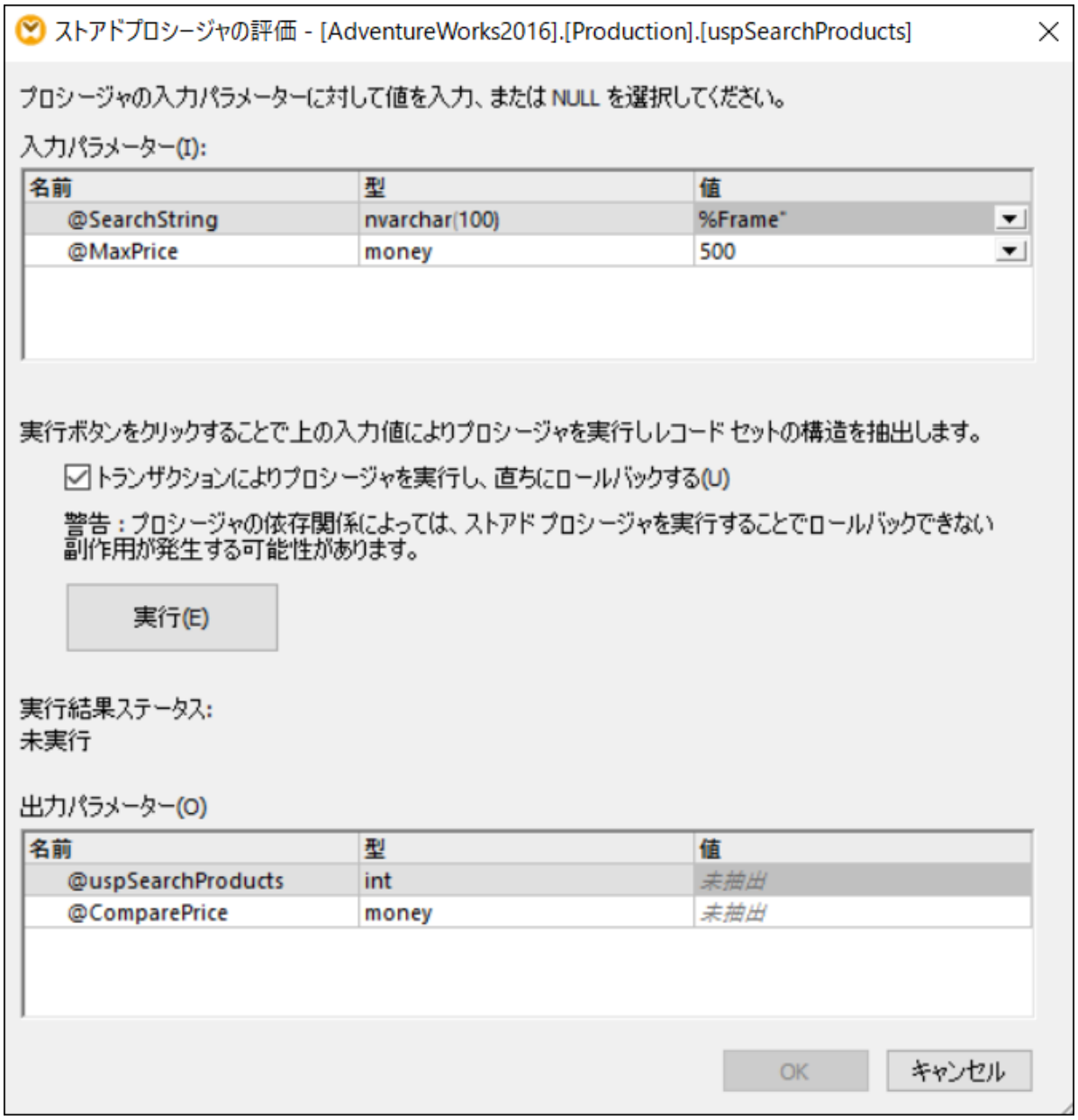
3. 「**コンテキストメニューの表示**」ボタンをもう一度クリックし、「**レコードセット構造の編集**」を選択します。これは、プロシージャにより返されたレコードセットの構造に関する MapForce 情報を提供するために必要です。「レコードセットの構造」ダイアログボックスが表示されます。

(C) 2015-2021 Altova GmbH



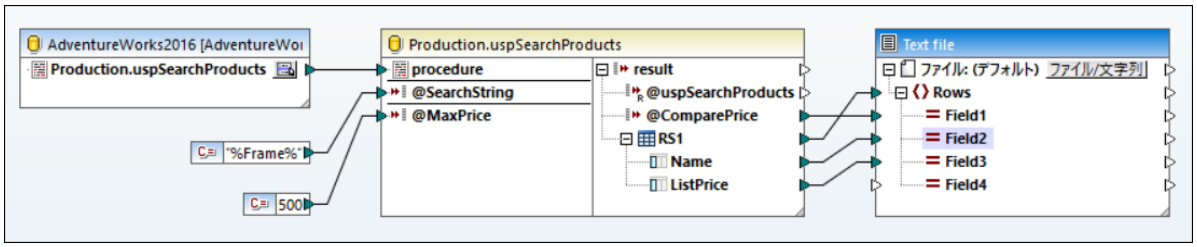
デザイン時にストアドプロシージャを呼び出すとプロシージャの実装により副作用が発生する可能性があります。デザイン時にストアドプロシージャを実行する場合、下記で説明されているとおり、「実行」をクリックしないでください。代わりに、レコードセットを手動で関連する列に追加し、「レコードセット構造」ダイアログボックス内で期待されるレコードセットを定義します。「レコードセット構造」ダイアログボックス内の「**レコードセットの追加**」または「**列の追加**」ボタンを使用します。

4. 「**入力パラメータと呼び出しプロシージャの定義**」をクリックします。「ストアドプロシージャの評価」ダイアログボックスが表示されます。

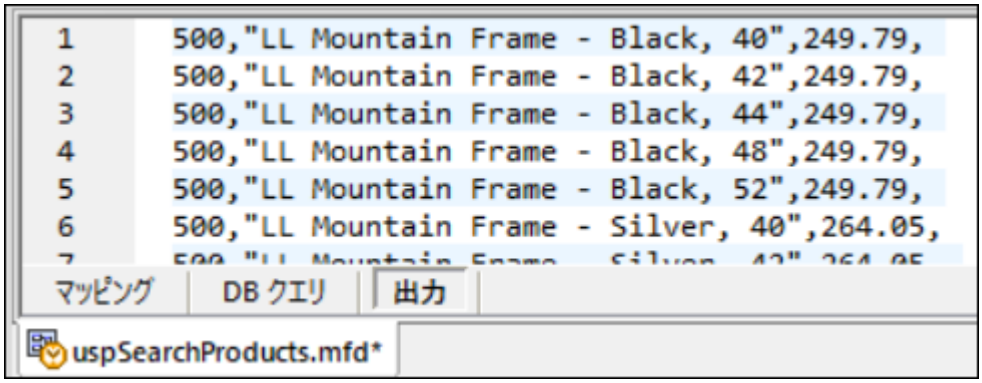


5. 上記の通りパラメータの値を入力し、「実行」をクリックします。
6. 「OK」をクリックします。レコードセット構造（"RS1"）が「レコードセットの構造」ダイアログボックスとマッピング上に表示されます。
7. この時点では、抽出されたデータが書き込まれるターゲットコンポーネントを追加することができます。このサンプルでは、データはCSV ファイルに書き込まれます。「挿入」メニューから「テキストファイル」をクリックし、データはCSV コンポーネントをマッピングに追加します。CSV とテキストファイルを参照してください。次に、下に示されるようにマッピング接続を描きます。プロシージャの入力パラメータが定数により提供されていることにご注意してください。詳細に関しては、マッピングに定数を追加するを参照してください。

(C) 2015-2021 Altova GmbH



マッピングをプレビューすることができます。「出力」ボタンをクリックして、「出力」ペイン内でマッピングの結果を確認します。例:



6.2.9.4 ターゲットコンポーネント内のストアドプロシージャ

このサンプルは、入力パラメータを取るプロシージャの呼び出しとデータベースの更新の方法について説明しています。プロシージャをこの方法で呼び出すことにより、トランザクションを有効化し、エラーの場合アクションをロールバック、またはプロシージャの呼び出し前にカスタムSQL ステートメントの実行を可能にすることができます。このシナリオは、ストアドプロシージャがMapForce 内のターゲットコンポーネントのように振る舞い、返される出力に興味がない場合を想定しています。パラメーターのパスの方法と、ストアドプロシージャにより返されたデータをマップする方法に関しては、 入力と出力を持つストアドプロシージャを参照してください。

「AdventureW orks」データベース内にデモのストアドプロシージャを最初に作成しましょう。これを行うには、下のスクリプトをデータベースに対して実行します。**Microsoft SQL Server Management Studio** のクエリウィンドウ、またはDatabaseSpy (https://www.altova.com/ja/databasespy) などのツールからこれを行うことができます。どちらの場合でも、データベースユーザーアカウントがストアドプロシージャを作成するパーミッションを有していることを確認してください。

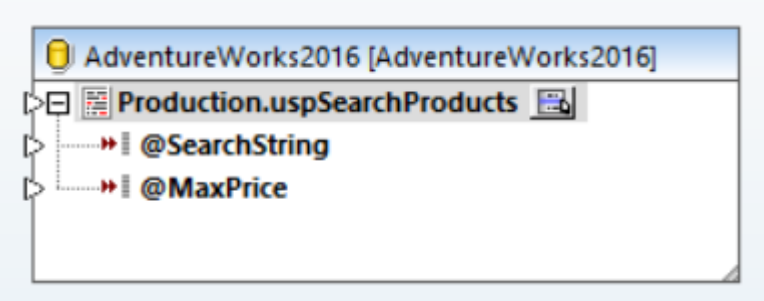
```
CREATE PROCEDURE Production.uspAddProductModel
   @ModelName nvarchar(50)
   ,@Inst xml
AS
BEGIN
INSERT INTO [Production].[ProductModel]
           ([Name]
           ,[Instructions]
           ,[rowguid]
           ,[ModifiedDate])
     VALUES
           (@ModelName
           ,@Inst
           ,NEWID()
```

```
            ,GETDATE())
END
```

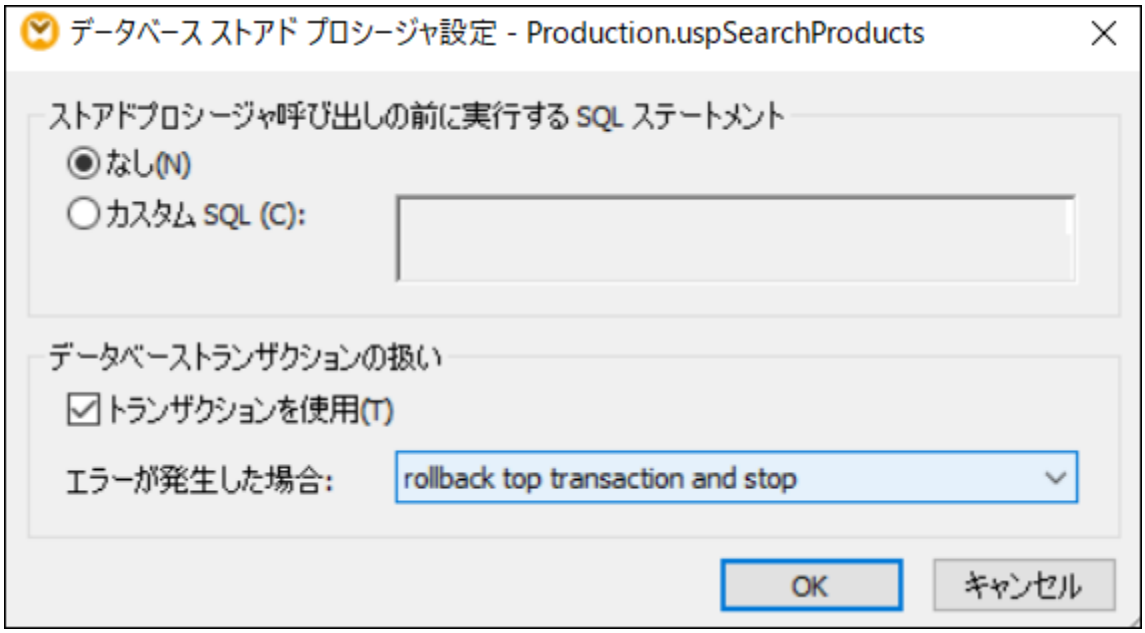
上のストアドプロシージャ入力として2つのプロシージャ(@ModelName, @Inst)を取り、データベースにより生成されたデータと共に、AdventureWorks データベースに ProductModel テーブルに対応する値を挿入します。

次のステップは、このプロシージャにより返されるデータを使用するマッピングの作成方法について示しています。

1. 「AdventureWorks」データベースを MapForce から接続し、マッピングにストアドプロシージャを追加する方法で説明されるとおり、ストアドプロシージャをマッピングに追加します。データベースユーザーアカウントがストアドプロシージャをビューして実行するパーミッションが存在することを確認してください。
2. ストアドプロシージャの横の「コンテキストメニューの表示」ボタンをクリックし、「ノードをターゲットとして表示」を選択します。左側に入力パラメーターがリストされているマッピング上でストアドプロシージャがターゲットコンポーネントとして表示されます。



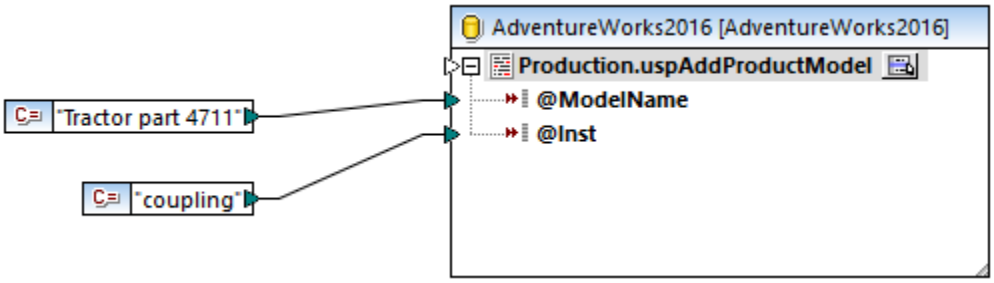
3. 「コンテキストメニューの表示」ボタンをもう一度クリックし、「プロシージャの設定」を選択します。この任意のステップにより、ロールバック可能なトランザクションの内部でストアドプロシージャを実行することができます。プロシージャが呼び出される前にカスタム SQL ステートメントの実行を追加することができます。
4. 「トランザクションの使用」チェックボックスを選択します。



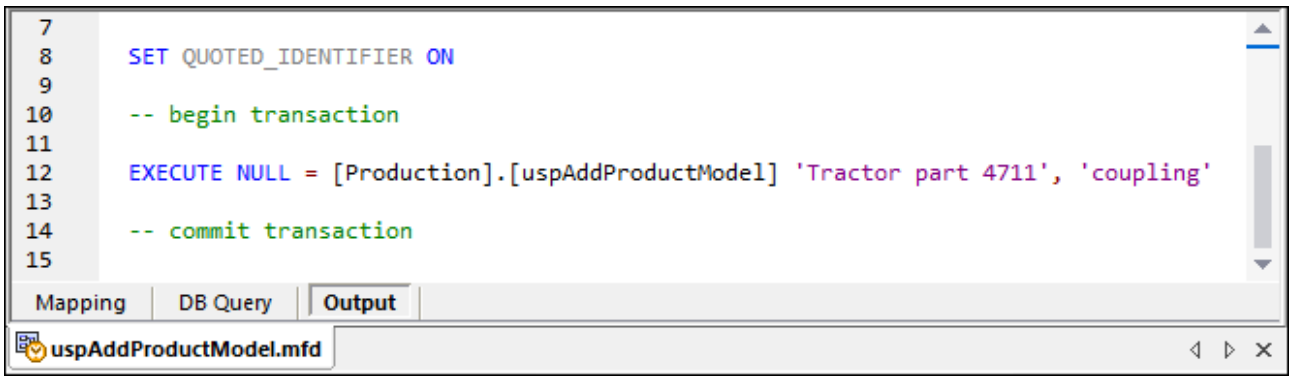
メモ このサンプルでは、データベースのトレースがデータベースコンポーネントのレベルで無効化されており、トレースが実行されません。しかしながら、必要な場合、ストアドプロシージャのためにデータベースのトレースを有効化することができます。詳細に関しては、データベースのトレースとエラーのログを参照してください。

(C) 2015-2021 Altova GmbH

5. データベースに挿入されるデータを提供するソースコンポーネントを追加します。このサンプルでは、ソースデータが定数により提供されますが、MapForce によりサポートされる他のソースコンポーネントは入力としての役割を果たします。詳細に関しては、マッピングに定数を追加するを参照してください。



このマッピングはデータベースを更新し、他のマッピングのように出力を直接プレビューしません。代わりに、「出力」ボタンをクリックして、データベースがどのように変更されるかについてのヒントを含む疑似 SQL を表示します。トランザクションを有効化すると、これらはコメントにより示されます。



「出力」ペイン内に表示される疑似 SQL は、実際のトランザクションコマンドを表示せず、(コメントとして) ヒントのみを表示します。実際の SQL コマンドは基になるデータベース API に送信されます。

データベースに対してマッピングを実行するには、以下を行います:

- 「出力」メニューから「SQL スクリプトの実行」をクリックします。
- 「SQL スクリプトの実行」 ツールバーボタンをクリックします。

ストアドプロシージャと入力の重複

同じストアドプロシージャにマッピング上の複数のソースからのデータをマップする場合、複数の入力を受け入れるようにストアドプロシージャを複製します。これを行うには、コンポーネント上のストアドプロシージャを右クリックし、コンテキストメニューから「入力の複製を追加する」を選択します。入力の複製を参照してください。マッピング実行される個所で、このようなストアドプロシージャの複製はそれぞれの複製された入力のために一度呼び出されます。

各パラメータは動的な値(または「nullable」)であることができるため、「入力の複製を追加する」はストアドプロシージャのパラメーターのために無効化されていることに注意してください。

6.2.9.5 ストアドプロシージャとローカルリレーション

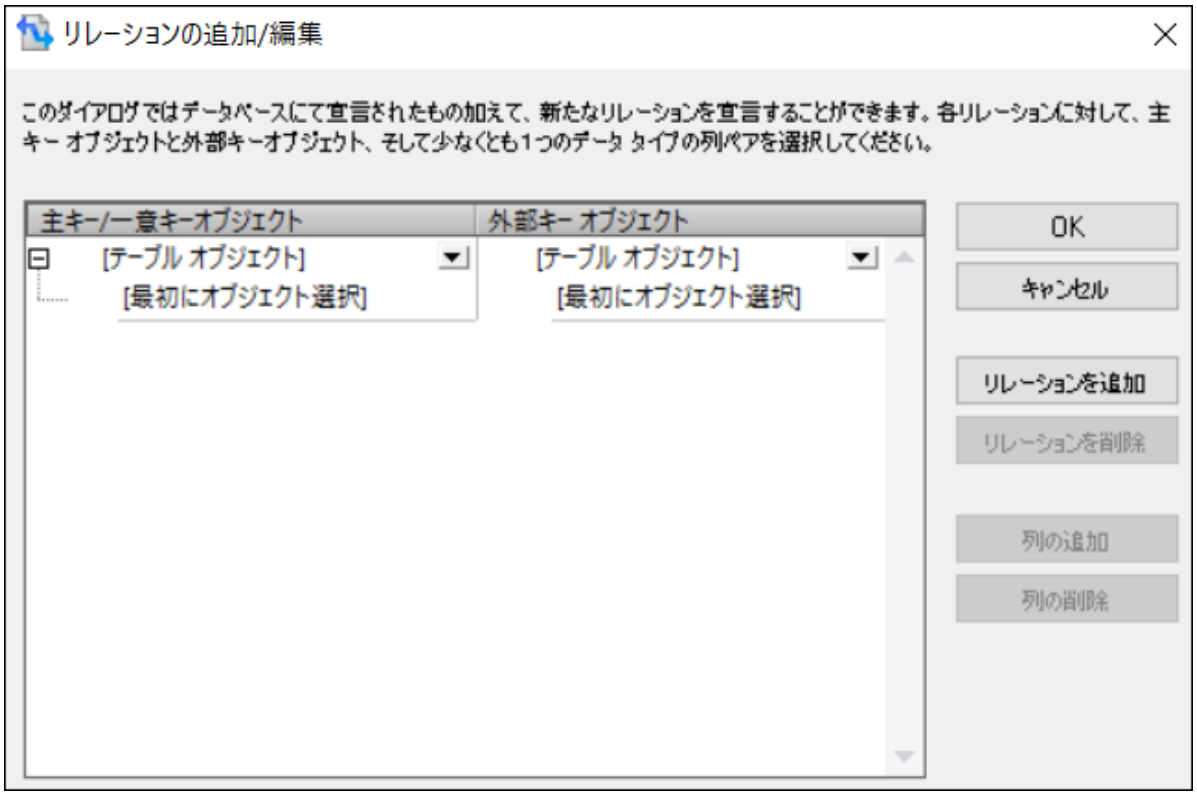
ローカルリレーションは、基になるデータベースを変更することなく MapForce 内で作成することのできるデータベースフィールド間のロジカルなリレーションシップです。ローカルリレーションシップの定義を参照してください。ローカルリレーションをデータベースフィールドのためだけではなく、ストアドプロシージャのためにソースとターゲットコンポーネント内で定義することができます。

ソースコンポーネント内で、ローカルリレーションにより関連したオブジェクトからデータを簡単に読み取れることができます。例えば、データベーステーブルからID を読み取り、関連した情報を取得するために、ID のそれぞれからストアドプロシージャを呼び出します。

ターゲットコンポーネント内で、ローカルリレーションにより複数の関連したプロシージャが呼び出される階層の順序を定義することができます。例えば、ID 値を作成するストアドプロシージャを最初に呼び出すことができ、テーブルに関連した情報を入力するものを呼び出すことができます。また、ストアドプロシージャとローカルリレーション内のテーブルをミックスすることもできます。例えば、他のプロシージャの呼び出しの代わりに直接関したテーブルに挿入することもできます。キーを生成するためにストアドプロシージャを使用する方法を参照してください。

ローカルリレーションの作成方法:

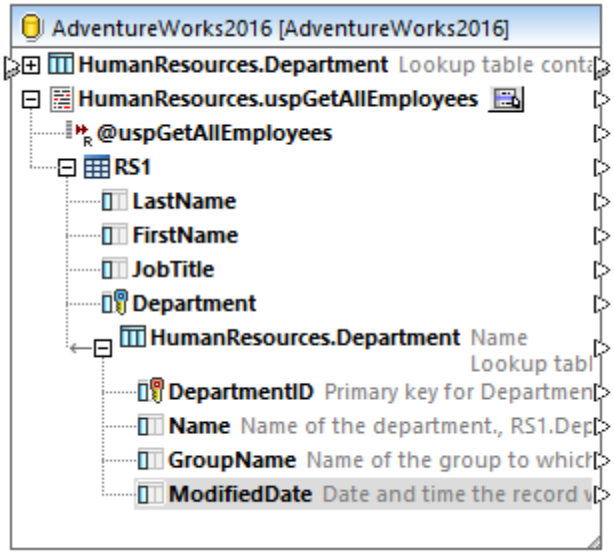
1. データベースコンポーネントのタイトルバーを右クリックし、「**データベースオブジェクトを追加/削除/編集**」をコンテキストメニューから選択します。「データベースオブジェクトを追加/削除/編集」ダイアログボックスが開かれます。
2. 「**リレーションの追加/編集**」をクリックします。
3. 「**リレーションの追加**」をクリックして、リレーションシップを作成するオブジェクトを選択します。



上記のとおり、「**主/一意のキー**」オブジェクトと**外部キー**オブジェクトによりローカルリレーションは構成されています。親-子 リレーションシップと同様と考えて下さい。マッピングコンポーネント上では、主/一意のキーがある(テーブル、ビュー、プロシージャ、などの)オブジェクトは親として示され、外部キーのあるオブジェクトは下にネストされ表示されます。例えば、下記のデータベースコンポーネント内では、ローカルリレーションはレコード列(**RS1.Department**)とテーブル列(**Department.Name**)との間で定義されています。この結果、**Department** テ

(C) 2015-2021 Altova GmbH

ーブルはマッピング上のストアドプロシージャの子として表示されます。このサンプルの詳細は、ソースコンポーネント内のローカルリレーションで詳しく説明されています。



次のテーブルはローカルリレーションを定義することのできるすべての可能なフィールドをリストしています（例えば、ストアドプロシージャの出力をデータベース列にマッピングするなど）。リレーションシップ内の一部のフィールドは、同じ、または互換性のあるデータ型を持つ必要があります。

主/ユニークキー	外部キー
• データベーステーブル、またはビューの列 • ストアドプロシージャの出力パラメーターまたは戻り値。ストアドプロシージャを参照してください。 • ストアドプロシージャにより返されたレコードセットの列* • ユーザー定義 SELECT ステートメントの列。仮想テーブルとしてのSQL SELECT ステートメントを参照してください。	• データベーステーブル、またはビューの列 • ストアドプロシージャの入力パラメーター • ユーザー定義 SELECT ステートメントの入力パラメーター

*（パラメータを持たない）データソースまたは、（入力と出力パラメーターを持つ）関数としてストアドプロシージャが呼び出された場合適用することができます。セクションのためにレコードセットが使用できるようになるには、レコードセットを取得するためにストアドプロシージャを一度実行する必要があります。

6.2.9.6 ソースコンポーネント内のローカルリレーション

このサンプルは、ローカルリレーションシップを使用して、ストアドプロシージャにより返されたデータを同じデータベース内のテーブルからのデータと組み合わせる方法について説明しています。

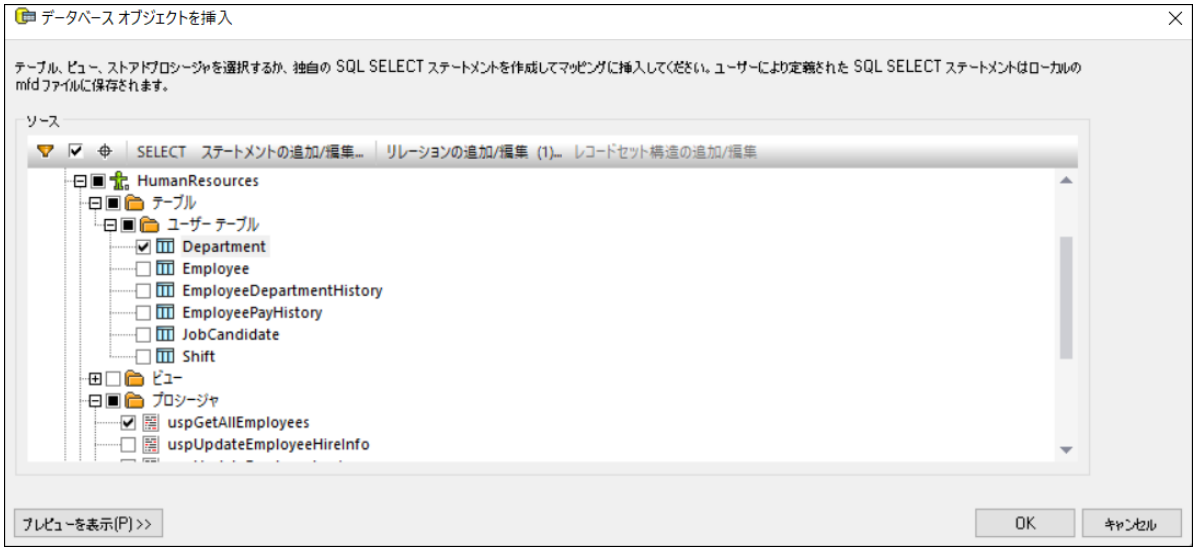
前のサンプルで既に行っていない場合、次のスクリプトを実行して、「AdventureW orks」データベース内にデモのストアドプロシージャを作成します。これをクエリウィンドウMicrosoft SQL Server Management Studio のクエリウィンドウ、またはMapForce の「DB クエリ」ペインから行うことができます。データベースの参照とクエリを参照してください。どちらの場合でも、データベースユーザーアカウントがストアドプロシージャを作成するパーミッションを持っていることを確認してください。

```
CREATE PROCEDURE HumanResources.uspGetAllEmployees
AS
```

```
SELECT LastName, FirstName, JobTitle, Department
FROM HumanResources.vEmployeeDepartment
```

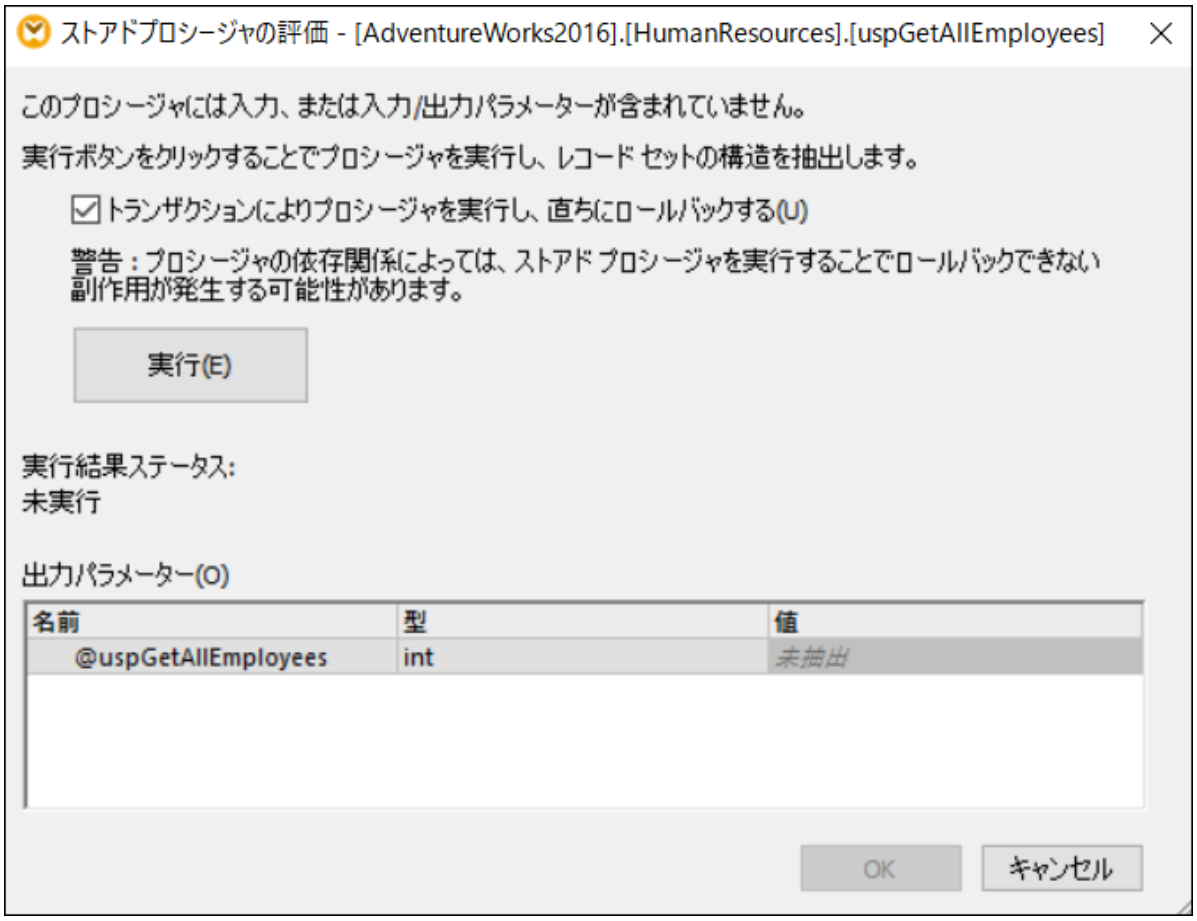
上のストアドプロシージャは、**vEmployeeDepartment** ビューから従業員の情報を返します。次のステップはこのプロシージャにより返されるデータを使用するマッピングの作成方法について説明しています。

1. マッピングにストアドプロシージャを追加する方法で説明されているとおり MapForce から「AdventureW orks」データベースに接続します。データベースユーザーアカウントがストアドプロシージャをビューして実行するパーミッションが存在することを確認してください。
2. データベースオブジェクトを選択するようにプロンプトされると、**Department** テーブルと **uspGetAllEmployees** ストアドプロシージャを選択します。

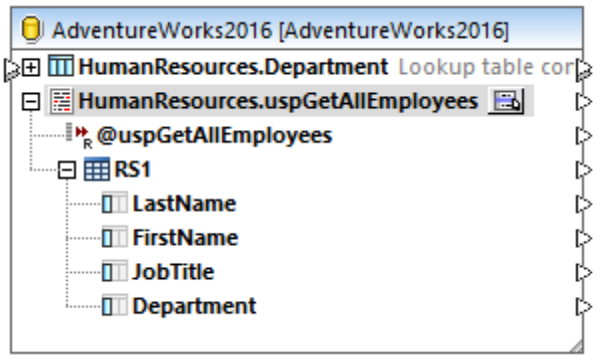


3. ストアドプロシージャの横の「**コンテキストメニューの表示**」ボタンをクリックし、「**ソースとしてノードを表示**」を選択します。
4. 「**コンテキストメニューの表示**」ボタンをもう一度クリックし、「**レコードセット構造の編集**」を選択します。「レコードセットの構造」ダイアログボックスが表示されます。
5. 「**入力パラメータと呼び出しプロシージャの定義**」をクリックし、「OK」をクリックします。「ストアドプロシージャの評価」ダイアログボックスが表示されます。

(C) 2015-2021 Altova GmbH



6. 「実行」をクリックし、「OK」をクリックします。レコードセット構造（"RS1"）が「レコードセットの構造」ダイアログボックスとマッピング上に表示されます。

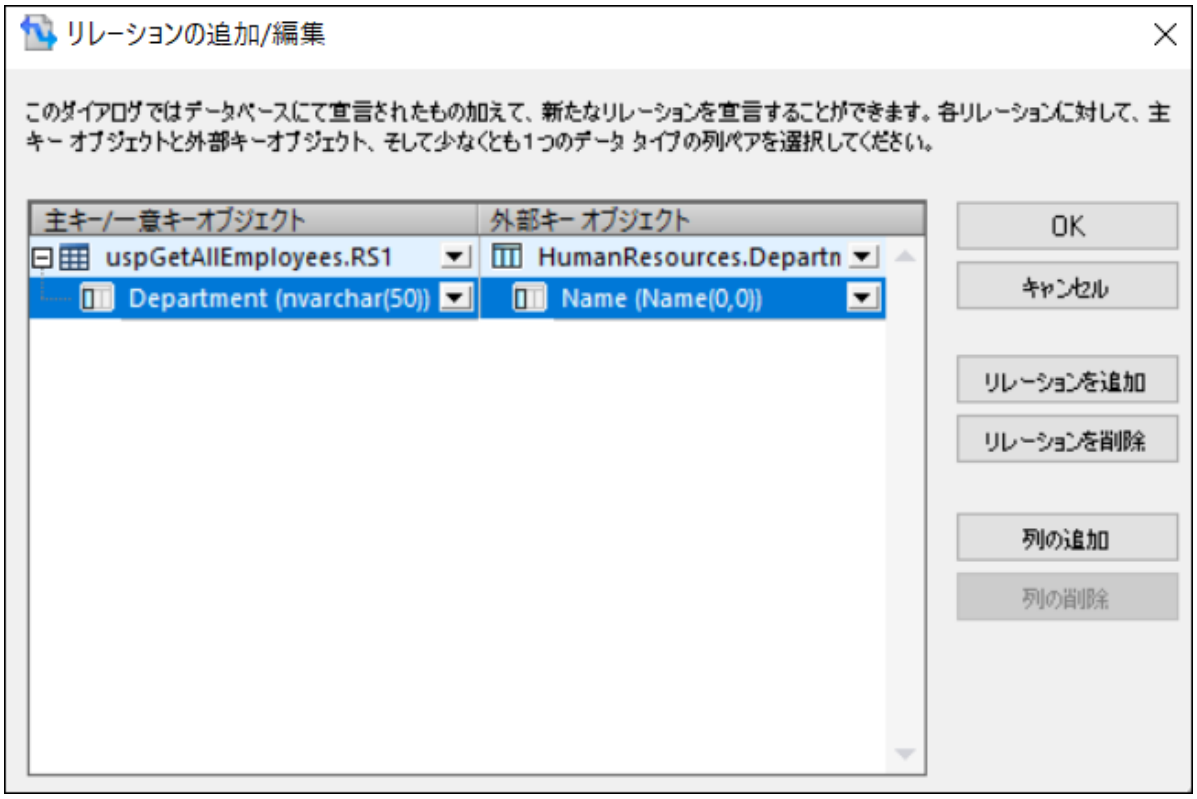


「ローカルリレーションシップの定義」

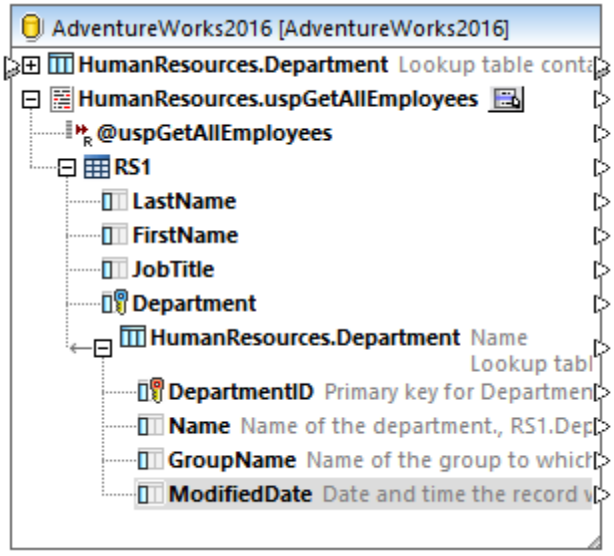
返されたレコードセットの**Department** 列と**Department** テーブルの**Name** 列の間のローカルリレーションシップを定義します。

1. データベースコンポーネントのタイトルバーを右クリックし、「**データベースオブジェクトを追加/削除/編集**」をコンテキストメニューから選択します。

2. 「リレーションの追加/編集」をクリックし、「リレーションの追加」をクリックします。以下に示されるようにリレーションシップを定義します。



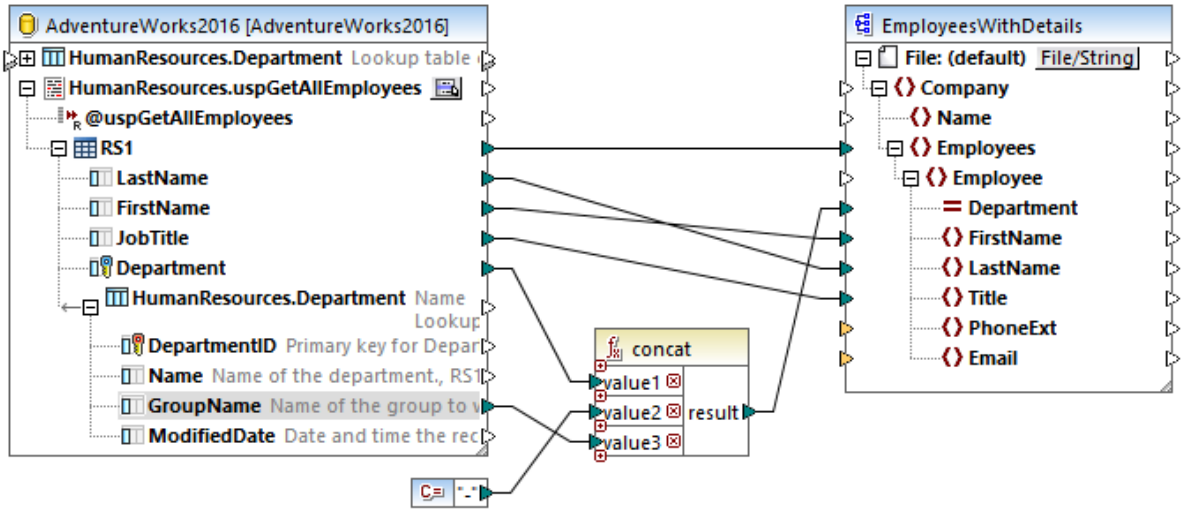
3. 「OK」をクリックして、ダイアログボックスを閉じます。**Department** テーブルは **RS1** レコードセットの子になったことに注意してください。



マッピングを完了する方法

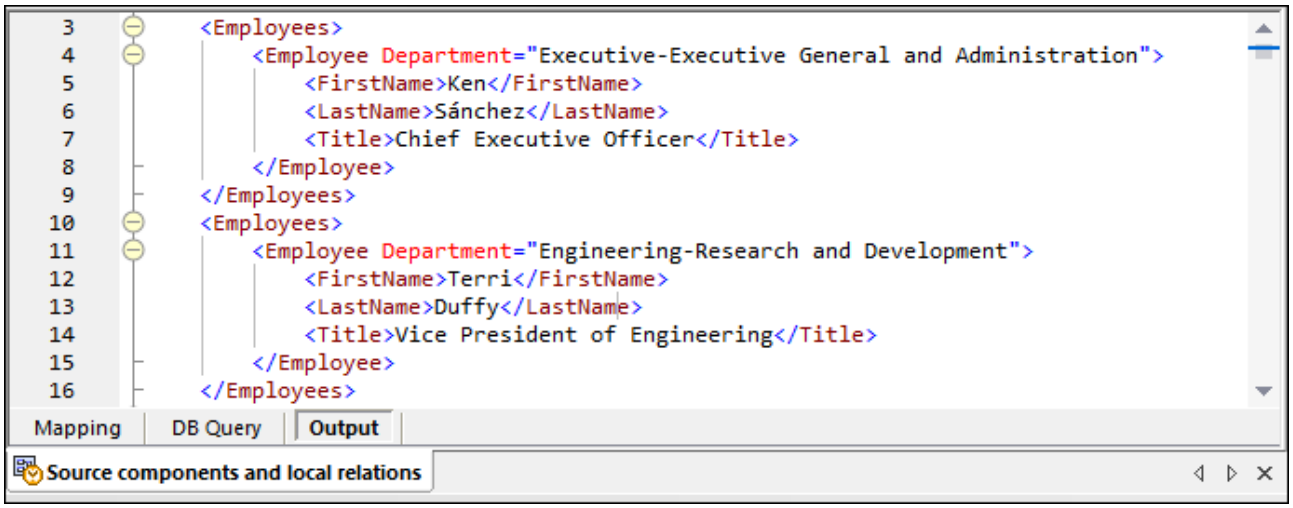
今しがた作成されたリレーションシップにより、テーブルからのデータと結合されたレコードセットからのデータをマップすることができるようになりました。サンプルのスコープのために、データを次のようにターゲット XML ファイルに書き込むことにしましょう:

1. 「挿入」メニューから「XML スキーマ/ファイル」をクリックし、次のファイルを選択します: **<マイドキュメント>\Altova\MapForce2021\MapForceExamples\EmployeesWithDetails.xsd**。
2. XML インスタンスファイルを提供するようにプロンプトされると、「スキップ」をクリックします。
3. 下に示されるようにマッピングの接続を描きます。



上で示されるマッピングはデータベースからターゲット XML ファイルにデータを書き込みます。ソースデータはストアドプロシージャから抽出されたデータと直接テーブルから抽出されたデータの組み合わせです。マッピングは、`concat` 関数を使用して、ダッシュ文字、とグループ名が後に続く部署名を含む文字列を作成します。

マッピングをプレビューするには、「出力」ボタンをクリックして、「出力」ペイン内でマッピングの結果を確認します。例:



6.2.9.7 キーを生成するためにストアドプロシージャを使用する方法

ローカルリレーションを使用して、ストアドプロシージャにより生成されたキー（ID）の一部を挿入する方法について説明されています。

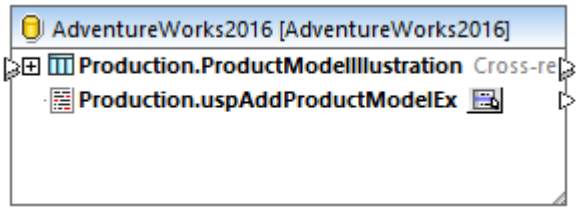
「AdventureW orks」データベース内にデモのストアドプロシージャを最初に作成しましょう。これを行うには、下のスクリプトをデータベースに対して実行します。**Microsoft SQL Server Management Studio** のクエリウィンドウ、またはDatabaseSpy (https://www.altova.com/ja/databasespy) などのツールからこれを行うことができます。どちらの場合でも、データベースユーザーアカウントがストアドプロシージャを作成するパーミッションを有していることを確認してください。

```
CREATE PROCEDURE Production.uspAddProductModelEx
  @ModelName nvarchar(50)
  ,@Inst xml
  ,@ProductModelID int OUTPUT
AS
BEGIN
INSERT INTO [Production].[ProductModel]
           ([Name]
           ,[Instructions]
           ,[rowguid]
           ,[ModifiedDate])
     VALUES
           (@ModelName
           ,@Inst
           ,NEWID()
           ,GETDATE())
   SELECT @ProductModelID = SCOPE_IDENTITY()
END
```

上のストアドプロシージャは、入力として2つのプロシージャ（`@ModelName`, `@Inst`）を取り、`INSERT` オペレーションを **ProductModel** テーブル内で実行します。生成された `@ProductModelID` を出力パラメータを返します。必要条件はストアドプロシージャにより返された `@ProductModelID` を **ProductModelIllustration** テーブル内に挿入することです。

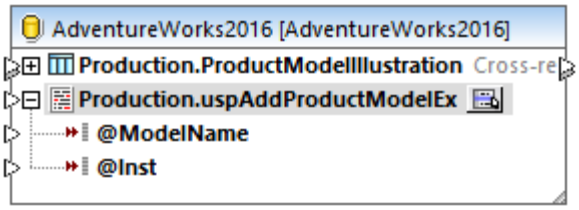
次のステップは必要条件を満たすマッピングの作成方法を示しています。

1. マッピングにストアドプロシージャを追加する方法で説明されているとおりMapForce から「AdventureW orks」データベースに接続します。データベースユーザーアカウントがストアドプロシージャをビューして実行するパーミッションが存在することを確認してください。
2. データベースオブジェクトを選択するようにプロンプトされると、**ProductModelIllustration** テーブルと **uspAddProductModelEx** ストアドプロシージャを選択します。

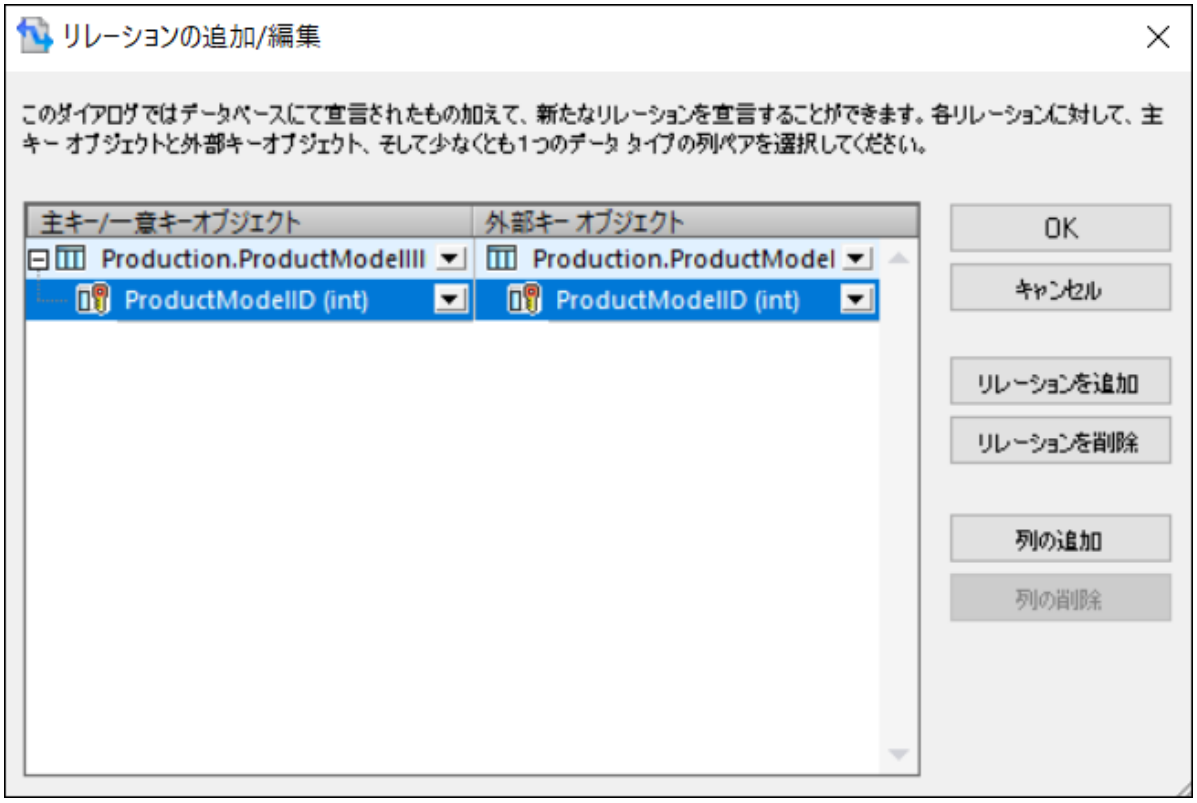


3. ストアドプロシージャの横の「コンテキストメニューの表示」ボタンをクリックし、「ノードをターゲットとして表示」を選択します。左側に入力パラメーターがリストされているマッピング上でストアドプロシージャがターゲットコンポーネントとして表示されます。

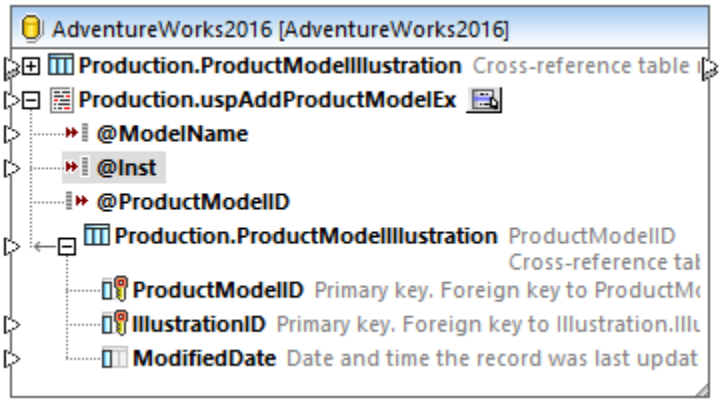
(C) 2015-2021 Altova GmbH



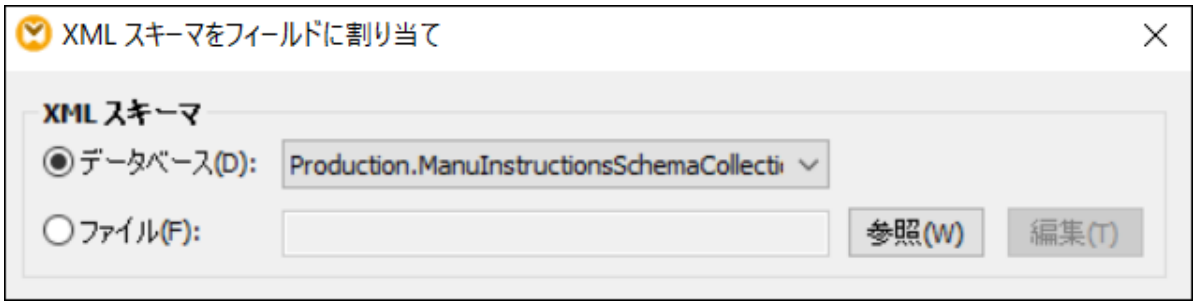
4. 任意で、トランザクション内でストアドプロシージャを実行するには、「**コンテキストメニューの表示**」ボタンをもう一度クリックし、「**プロシージャの設定**」を選択し、「**トランザクションの使用**」チェックボックスを選択します。ストアドプロシージャのためにトランザクションを定義することは、同じトランザクション中にキーの取得とレコードの挿入が同時に発生することを保証することができます。
5. データベースコンポーネントのタイトルバーを右クリックし、「**データベースオブジェクトを追加/削除/編集**」をコンテキストメニューから選択します。
6. 「**リレーションの追加/編集**」をクリックし、「**リレーションの追加**」をクリックします。以下に示されるようにリレーションシップを定義します。



7. 「OK」をクリックして、ダイアログボックスを閉じます。**ProductModelIllustration** テーブルはストアドプロシージャの子として表示されていることにご注意してください。ストアドプロシージャ出力パラメーター(`@ProductModelID`)はローカルリレーション内で使用されるインジケーターに表示されます。入力、または出力コネクタを持ちません。

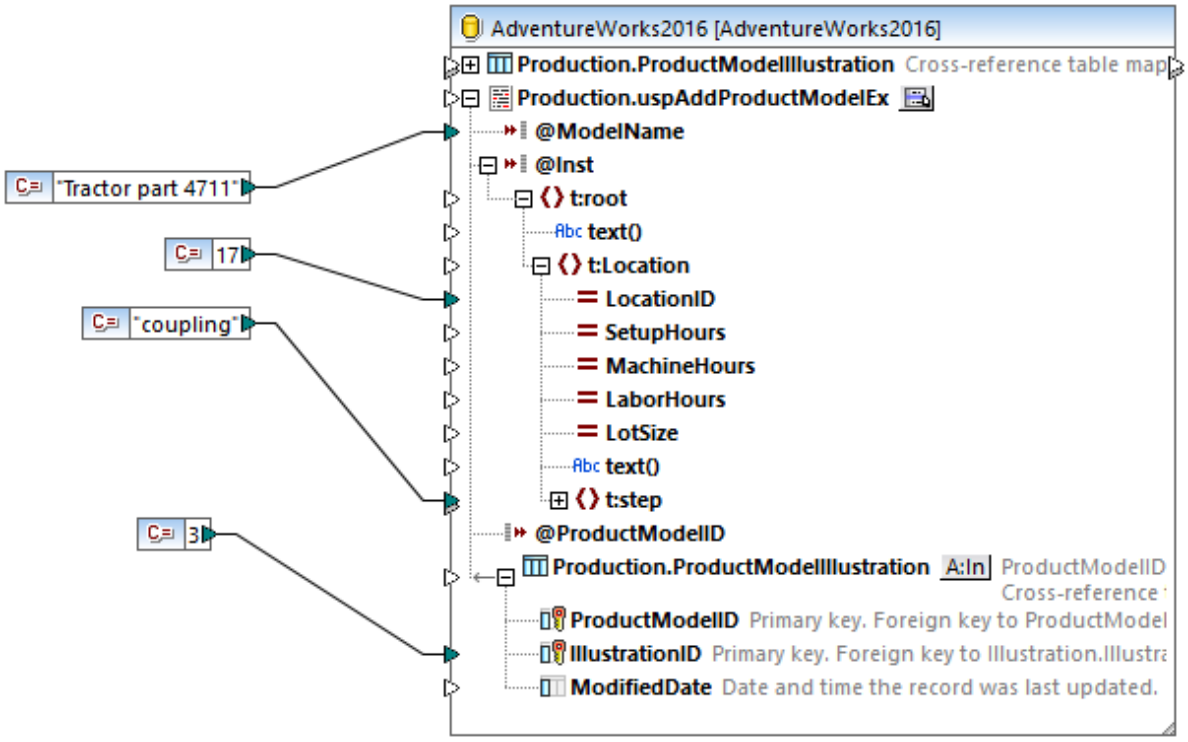


8. このサンプルでは、`@Inst` パラメーターはXML の型です。コンポーネント上の`@Inst` パラメーターを右クリックし、「XML スキーマをフィールドに割り当てる」をコンテキストメニューから選択します。**Production.ManuNstructionsSchemaCollection** スキーマをデータベースから選択します。ルート要素を選択するようにプロンプトされると、デフォルトの値をそのままにし、「OK」をクリックします。データベースXML フィールドへのデータのマッピングの詳細に関しては、XML データから / データベースフィールドへのマッピングを参照してください。



9. データベースに挿入されるデータを提供するソースコンポーネントを追加します。このサンプルでは、ソースデータが定数により提供されますが、MapForce によりサポートされる他のソースコンポーネントは入力としての役割を果たします。定数の詳細に関しては、マッピングに定数を追加するを参照してください。

(C) 2015-2021 Altova GmbH



このマッピングはデータベースを更新し、他のマッピングのように出力を直接プレビューしません。代わりに、「出力」ボタンをクリックして、データベースがどのように変更されるかについてのヒントを含む疑似 SQL を表示します。トランザクションを有効化すると、これらはコメントにより示されるように発生します。

```
10    -- begin transaction
11
12    NULL = {{{[Production].[uspAddProductModelEx]}}} ( 'Tractor part 4711','<root
      xmlns="http://schemas.microsoft.com/sqlserver/2004/07/adventure-works/
      ProductModelManuInstructions"><Location LocationID="17"><step>coupling</
      step></Location></root>',NULL )
13    -->>> %@uspAddProductModelEx1%
14    -->>> %@ProductModelID1%
15
16    INSERT INTO [Production].[ProductModelIllustration] ([ProductModelID],
      [IllustrationID]) VALUES ('%@ProductModelID1%', 11)
17
18    -- commit transaction
19
```

Mapping | DB Query | Output

uspAddProductModelEx

「出力」ペイン内に表示される疑似 SQL は、実際のトランザクションコマンドを表示せず、(コメントとして) ヒントのみを表示します。実際の SQL コマンドは基になるデータベース API に送信されます。

データベースに対してマッピングを実行するには、以下を行います:

- 「出力」メニューから「SQL スクリプトの実行」をクリックします。
- 「SQL スクリプトの実行」 ツールバーボタンをクリックします。

6.2.10 トランザクションロールバックの処理

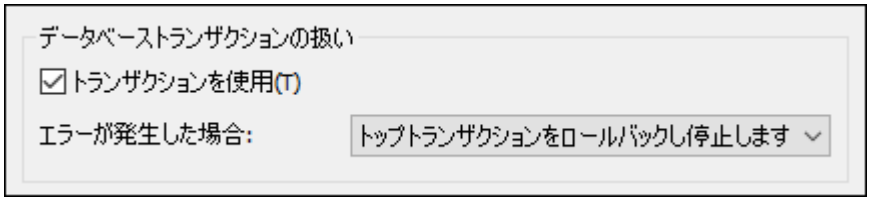
データベースを変更するマッピングを実行する場合、MapForce (またはMapForce Server、またはMapForceにより生成されたプログラム) ではデータベースに関連したエラーが発生する場合があります。例えば、データベースアカウントには、特定のデータベースアクションを実行するだけの特権がない、または無効なデータを入力、または更新することを回避するデータベース制約を実行する特権がない場合があります。後者は、列が必須で、マッピングが値を適用しない場合に発生する場合があります。

データベース例外が発生した場合にロールバックを可能にするには、マッピングがデータベーストランザクションを使用するように構成します。下で説明されるとおり、データベースコンポーネントレベルとテーブルアクションレベルでトランザクションを有効化することができます。

データベースコンポーネントレベルでのトランザクション

データベースコンポーネントレベルでトランザクションを有効化するには、以下を行います:

1. データベースコンポーネントを右クリックし、コンテキストメニューから「プロパティ」を選択します (代わりに、データベースコンポーネントのタイトルバーをダブルクリックします)。これによりデータベースコンポーネント設定 ダイアログボックスが開かれます。



2. 「トランザクションの使用」チェックボックスを選択します。

これを行うことにより、データベースコンポーネントにより行われるすべての変更がエラーを発生した場合ロールバックされるトランザクション内に囲まれます。マッピングの実行中にエラーが発生すると、出力はドロップダウンリストから選択されたオプションにより異なります:

- **トップトランザクションをロールバックし停止** – データベースコンポーネントにより行われるすべての変更を囲むトランザクションはロールバックされ、マッピングは停止されます。
- **トップトランザクションをロールバックし継続** – 上記同様ですが、マッピングはロールバック後も継続されます。(例えば、存在する場合、2番目のターゲットコンポーネントを処理する場合など)。

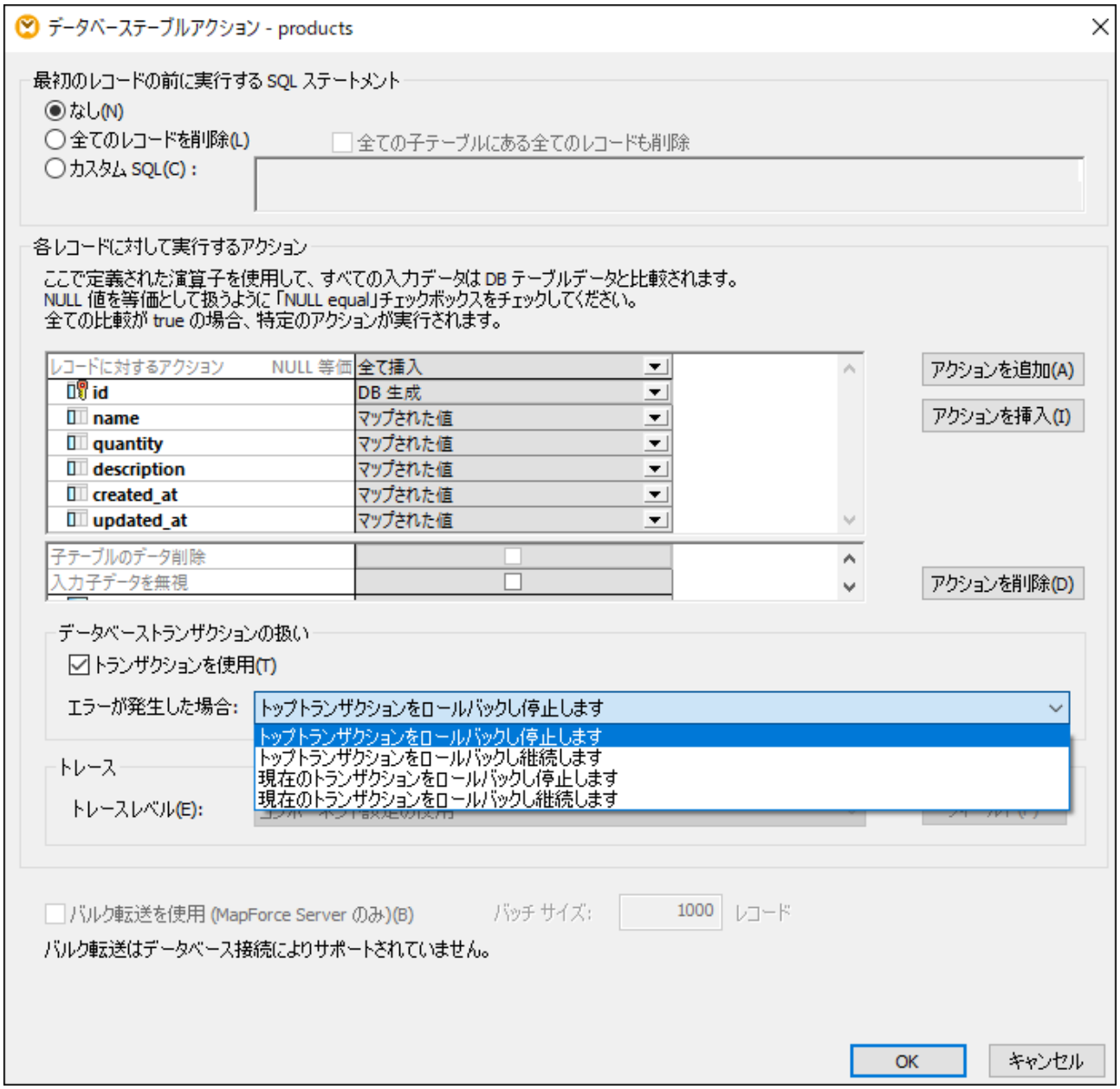
マッピングの一部は複数のデータベースコンポーネントを含んでいる可能性があります。これらは、同じデータベース接続を使用する、またはまったく異なるデータベースへの接続を使用している場合があります。このようなマッピングの出力は、データベースエラーが発生する場合、実行エンジンによりとなります:

- マッピングがMapForce で実行される場合マッピングの実行時、1つのターゲットコンポーネントが実行されます。「プレビュー」 ボタンが有効化されている個所のコンポーネントです。そのコンポーネント内でデータベースエラーが発生し、「トランザクションの使用」チェックボックスが有効化されていると、コンポーネントによる全ての変更がロールバックされます。この場合、データベースエラーが発生すると、エラーが発生する箇所でロールバックが発生し、マッピングは停止されるか、エラーが発生する箇所でドロップダウンリストで選択されたとおり、次のターゲットコンポーネントの実行を継続します。

データベースコンポーネント設定 ダイアログボックスで使用することのできるすべての設定に関数の参照に関しては、データベースコンポーネント設定 を参照してください。

テーブルアクションレベルでのトランザクション

トランザクション at テーブルアクション レベルでのトランザクションは、個別の挿入、更新などのテーブルアクションを含みます。このセットアップでは、レコード上で挿入、または更新アクションが不足する場合、現在失敗したアクションのみをロールバックし、他のレコードの処理を続行することができます。テーブルアクション レベルでトランザクションを実行するには、データベーステーブルアクション ダイアログボックス「**トランザクションの使用**」チェックボックスから選択します。（次を参照: データベーステーブルアクション設定).



使用することのできるオプションは以下の通りです:

- **トップトランザクションをロールバックし停止** – 親子リレーションシップを持つマッピングが複数のテーブルを更新する場合、親テーブルのためだけではなく、子テーブルのためにトランザクションを有効化することができます。トランザクションが複数のレベルで有効化することができるため、「トップ」は階層で更に高いレベルを参照することができます。これは、そのレベルでのトランザクションが有効化

されている場合、データベースコンポーネントレベルからのトランザクションです。それ以外の場合、コンポーネントには複数のネストされたテーブルアクションが存在することが仮定され、一番上のテーブルアクションで定義されたトランザクションが取られます。データを複数にリンクされているテーブルに挿入する で説明されるとおり、このオプションを理解するには、コンポーネントが複数のテーブルの階層を更新することができることを思い出してください。例えば、コンポーネントには、個人のレコードを挿入する親「挿入」アクション、および、個別のテーブルに各個人のレコードを挿入する1つまたは複数のネストされた「挿入」アクションが存在します。両方の挿入アクションのためにトランザクションを有効化することができ、「トップ」は階層レベルの最も高いレベルを指しています（このサンプルでは、個人がアクションを挿入しています）。このため、アドレスを挿入する際に例外が発生すると、これはこのサンプルでは、個人のテーブルレベルである)トップのトランザクションをロールバックし、実行は停止されます。すなわち、個人のアドレスを挿入できない場合、個人のレコードは挿入されません。

- **トップトランザクションをロールバックし継続する** – 上記同様ですが、ロールバックの後もマッピングは継続されます（例えば、存在する場合、他のターゲットコンポーネントを処理するために継続されます）。
- **現在のトランザクションをロールバックし停止する** – 例外が発生すると、このロールバックは、現在のトランザクション内で発生した変更のみをロールバックします。現在のトランザクション外で発生した変更は、コミットされます。
- **現在のトランザクションをロールバックし継続する** – 上記同様ですが、マッピングはロールバック後も継続されます。

ストアドプロシージャレベルトランザクション

ストアドプロシージャのためのデータターゲットと呼ばれるデータベーストランザクションを有効化することができます。例えば、 ターゲットコンポーネント内のストアドプロシージャ を参照してください。（入力パラメーターを持たない）データソースとして呼び出される、または関数のように呼び出される（入力と出力を持つ）ストアドプロシージャのためにはトランザクションはサポートされていません。.

ロールバック

トランザクションがマッピングランタイムに発生するかを確認するには、「出力」タブをクリックして、MapForce により生成された疑似-SQL codeコードを確認します。下のイメージは、コンポーネントレベルとテーブルアクションレベルでトランザクションが有効化されているマッピングのプレビューを示しています。下に示されているとおり、メインのトランザクションは各オペレーションのために発生します（コメント `--begin transaction` と`--commit transaction` により識別されます）。また、複数の小さいトランザクションは各入力オペレーションのために発生します（`SAVEPOINT` コマンドにより識別されます）。

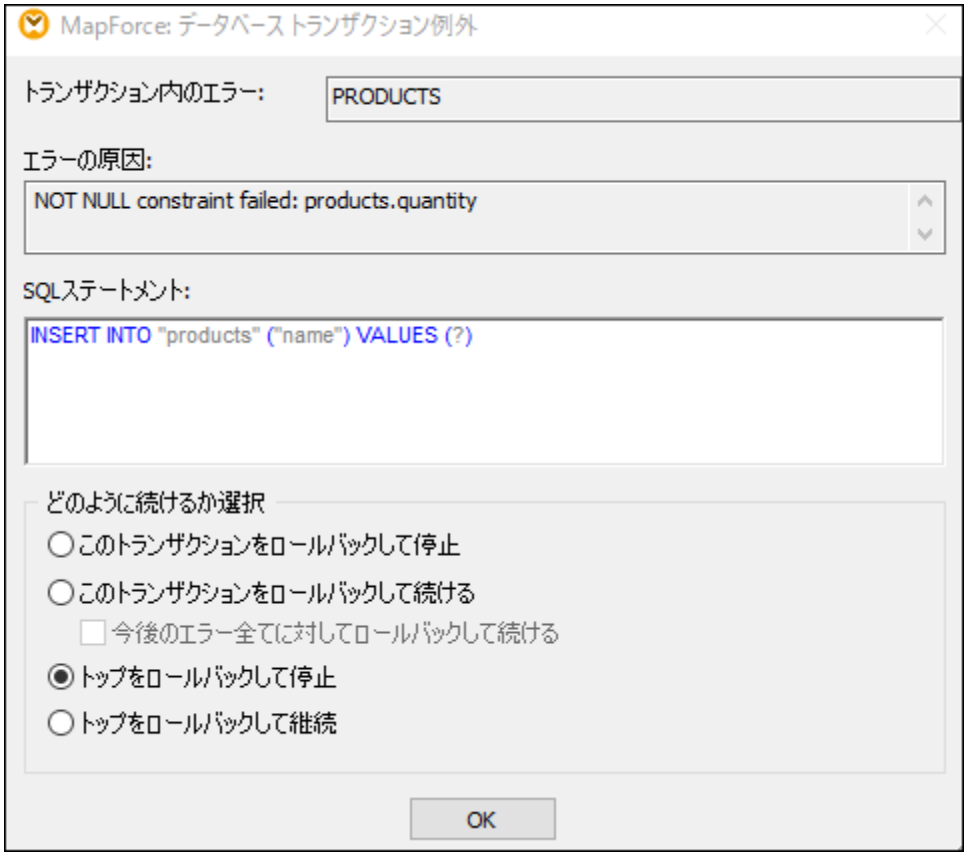
```
 7
 8    -- begin transaction
 9
10    PRAGMA foreign_keys = ON;
11
12    SAVEPOINT PRODUCTS
13
14    INSERT INTO "products" ("name") VALUES ('Product A')
15
16    SAVEPOINT PRODUCTS
17
18    INSERT INTO "products" ("name") VALUES ('Product B')
19
20    SAVEPOINT PRODUCTS
21
22    INSERT INTO "products" ("name") VALUES ('Product C')
23
24    -- commit transaction
25
```

マッピング | DB クエリ | 出力

マッピングがMapForce Server またはMapForce により生成されたプログラムにより実行されると、上記のとおり、マッピングのデザイン中に構成されたオプションに基づきロールバックが実行されます。例えば、オプション「トップトランザクションをロールバックし停止」を選択すると、他の挿入が成功した場合でも、失敗した入力オペレーションにより全ての変更がロールバックされます。しかしながら、「現在のトランザク

ションをロールバックし継続する」を選択すると、失敗した挿入が最も近い保存ポイントにロールバックされ、マッピングは他のレコードを挿入しようと試みます。

メニューコマンド「出力 | Run SQL-スクリプトの実行」を使用して MapForce 内でトランザクションが有効化されたマッピングを直接実行すると、ダイアログボックスはデータベースの例外が発生したかを通知します。例:



以降の振る舞いを変更することができます:

- **このトランザクションをロールバックし停止する** – これは、現在のトランザクションに含まれる変更のみをロールバックし、マッピングの実行を停止します。現在のトランザクションの外部で前に行われた変更はコミットされます。
- **このトランザクションをロールバックし継続する** – 上記同様ですが、マッピングロールバック後も実行されます。チェックボックス **ロールバックして全てのエラーのために継続する** を選択して、更にエラーが発生した場合、ダイアログボックスを抑制します。
- **すべてをロールバックし停止する** – このデータベース呼び出し内のすべてのトランザクションは、ロールバックされ、マッピングの実行は停止されます。

6.2.10.1 例: トランザクションロールバック

このサンプルは、データベースを変更するマッピングを実行する際にデータベースの例外を処理する可能な方法について説明しています。このサンプル内で使用されているデータベースはユーザーレコード (`users` テーブル) とアドレス (`addresses` テーブル) を保管します。各ユーザーはゼロ、1つ、または2つのアドレスを持つことができます (例えば、自宅の住所と職場の住所)。具体的には、アドレス内の各アドレスには、`users` テーブル内の `id` フィールドを指す `user_id` フィールドが含まれています。`users` テーブルはこのため `addresses` のための「親」テーブルです。下のリスティングは両方のテーブルのための SQL 作成スクリプトを示しています (構文が SQLite データベースに適用できることにご注意してください):

```
CREATE TABLE
       users (id INTEGER NOT NULL PRIMARY KEY,
       first_name TEXT NOT NULL,
       last_name TEXT NOT NULL,
       email TEXT UNIQUE NOT NULL);

CREATE TABLE
       addresses (id INTEGER NOT NULL PRIMARY KEY,
       user_id INTEGER NOT NULL,
       is_shipping INTEGER,
       is_billing INTEGER,
       type TEXT NOT NULL,
       city TEXT NOT NULL,
       street TEXT NOT NULL,
       number INTEGER NOT NULL,
       FOREIGN KEY (user_id) REFERENCES users (id) ) ;
```

ソーステーブル

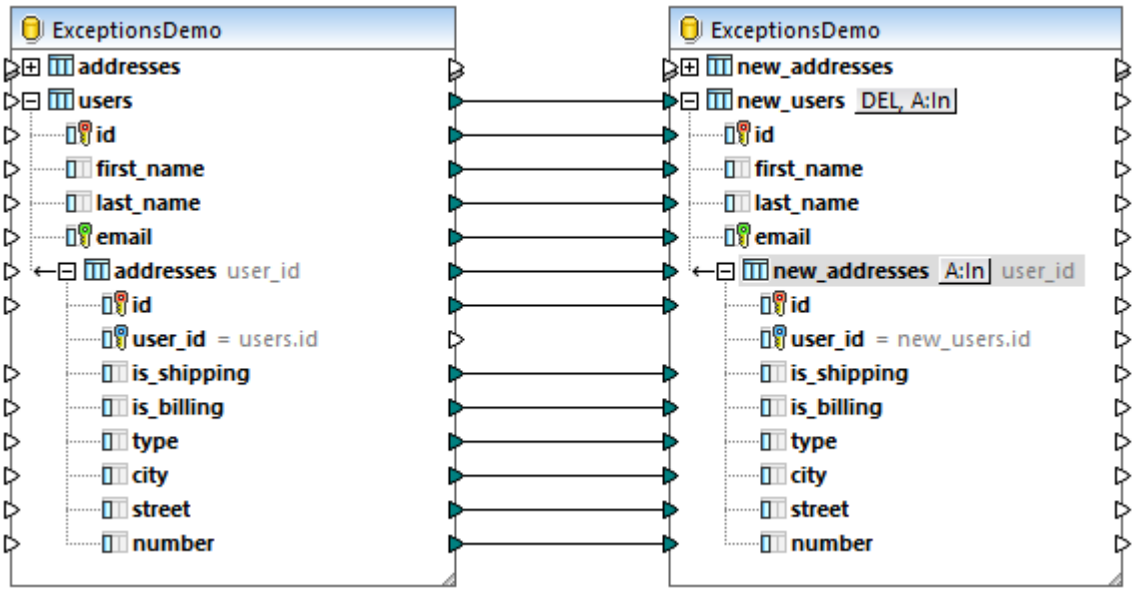
必要条件は users と addresses テーブルからのすべてのデータを新規のテーブルすなわち new_users と new_addresses テーブルにコピーします。これらのテーブルは users と addresses テーブルに類似しています。唯一の違いはテーブル new_addresses は下のコードリスト内でハイライトされている通り列 is_shipping と is_billing のために値を許可しません。これは、値が発生すると例外が発生することを意味するためこれは重要です。

```
CREATE TABLE
       new_users (id INTEGER NOT NULL PRIMARY KEY,
       first_name TEXT NOT NULL,
       last_name TEXT NOT NULL,
       email TEXT UNIQUE NOT NULL);

CREATE TABLE
       new_addresses (id INTEGER NOT NULL PRIMARY KEY,
       user_id INTEGER NOT NULL,
       is_shipping INTEGER NOT NULL,
       is_billing INTEGER NOT NULL,
       type TEXT NOT NULL,
       city TEXT NOT NULL,
       street TEXT NOT NULL,
       number INTEGER NOT NULL,
       FOREIGN KEY (user_id) REFERENCES new_users (id) ) ;
```

ターゲットテーブル

古いテーブルから新しいテーブルに全てのデータをコピーするマッピングは以下に示されています。このマッピングを次のパスで見つけることができます: <マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥Tutorial¥**DatabaseExceptions.mfd**。

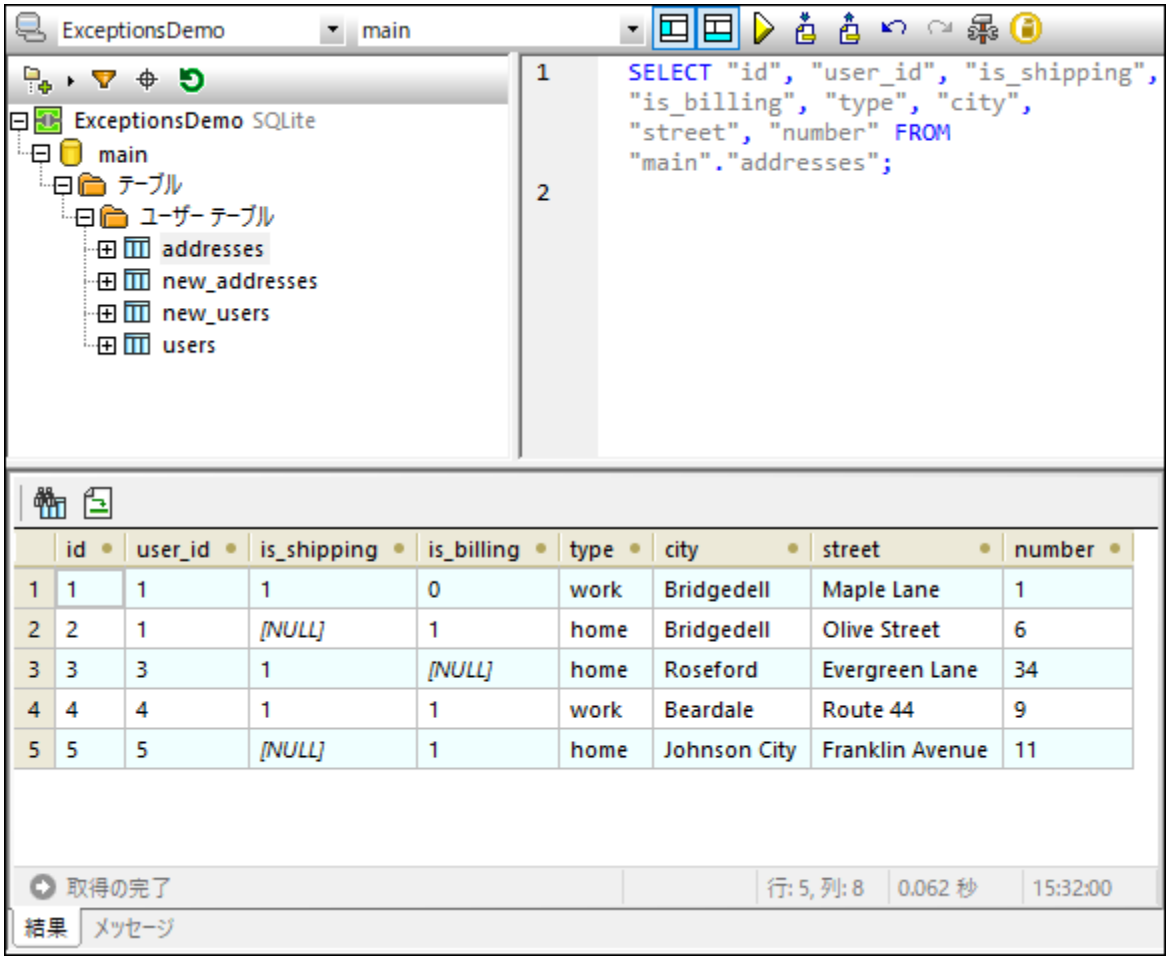
(C) 2015-2021 Altova GmbH



上で表示されている通り、このマッピングはデータをそのまま同じデータベースに属するソースデータベースコンポーネントからターゲットデータベースコンポーネントにコピーします。唯一の違いは、ソースとターゲットテーブルが異なる点です。ターゲットコンポーネントのタイトルバーをダブルクリックすると、各入力アクションの前に全てのレコードセットが削除されていることにご注目してください。これにより、ターゲットテーブルが各挿入の前に空であることを保証することができます。

上記の通り、ソース `addresses` テーブルにnull 値が含まれていない場合、例外が発生します。以下の通りテーブルにnull 値が含まれていないかをチェックすることができます:

1. 「DB クエリ」タブをクリックします。
2. `addresses` テーブルの横のボタンを右クリックし、「SQL エディター | SELECT」内で表示を選択します。
3. 「クエリの実行」 をクリックします。



上で表示されている通り、ソース`addresses`テーブル内に複数の`NULL`フィールドが存在します。この点を考慮して、マッピングを実行する際に、例外を扱うための複数のオプションを使用することができます。例:

- A) 例外が発生すると、全ての変更がロールバックされます。すなわち、ソースデータに無効なレコードが含まれているとレコードが挿入されません。
- B) 例外が発生すると、例外が発生する箇所でレコードをスキップし、有効なレコードを挿入し続けます。

シナリオ A: 例外の場合に全ての変更をロールバックする

必要条件: 例外が発生するとデータベースへ加えられた全ての変更をロールバックします。この目的を達成するためにマッピングを構成する方法は以下のとおりです:

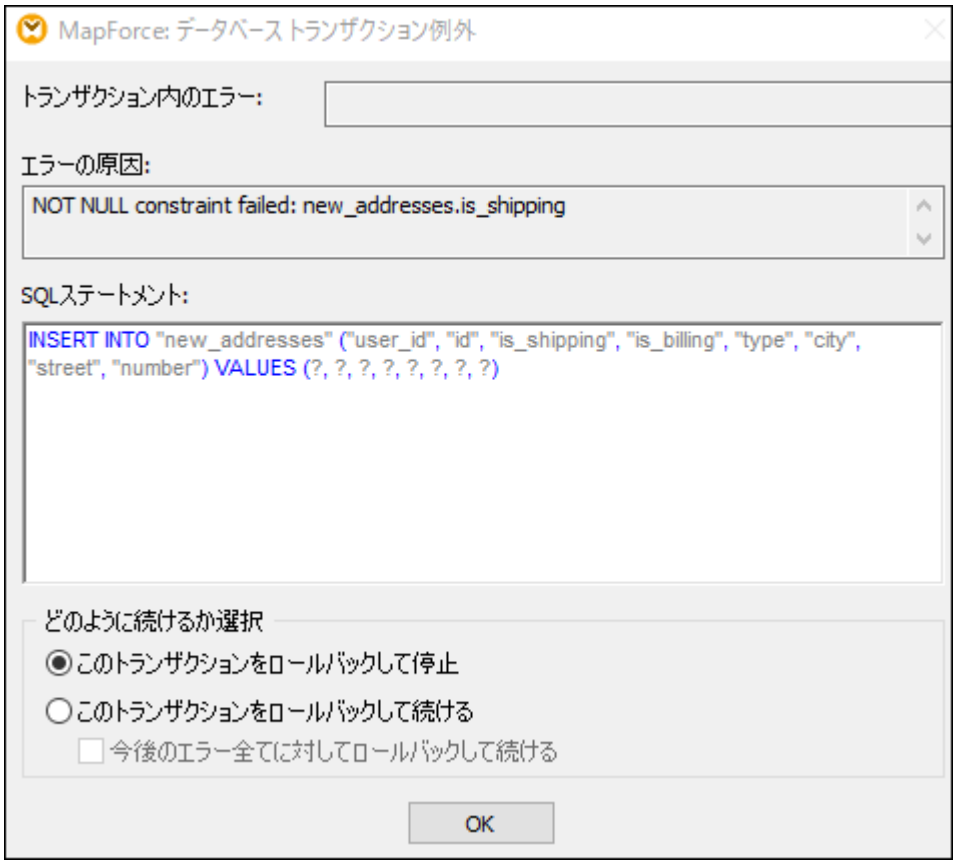
1. ターゲットデータベースコンポーネントのタイトルバーを右クリックし、コンテキストメニューから「**プロパティ**」を選択します。または、タイトルバーをダブルクリックします。これにより、データベースコンポーネント設定 ダイアログボックスが開かれます。
2. 「**トランザクションの使用**」チェックボックスを選択し、「**トップトランザクションをロールバックして停止します**」を選択します。

マッピングが上記のように構成されていると、例外の発生は、データベースコンポーネントレベルでのトップレベルのトランザクションのロールバックを引き起こします。これらの設定を使用してマッピングを実行するには以下を行います:

1. 「**出力**」タブをクリックします。

(C) 2015-2021 Altova GmbH

2. 出力メニューから、「SQL-Script の実行」をクリックします。ソースアドレステーブル内の null 値のため、このステージでは、マッピングは例外に遭遇します。データベーストランザクション例外 ダイアログボックスが表示されます。



3. デフォルトの選択済みのオプションを変更せず、「OK」をクリックします。

結果は以下のようになります:

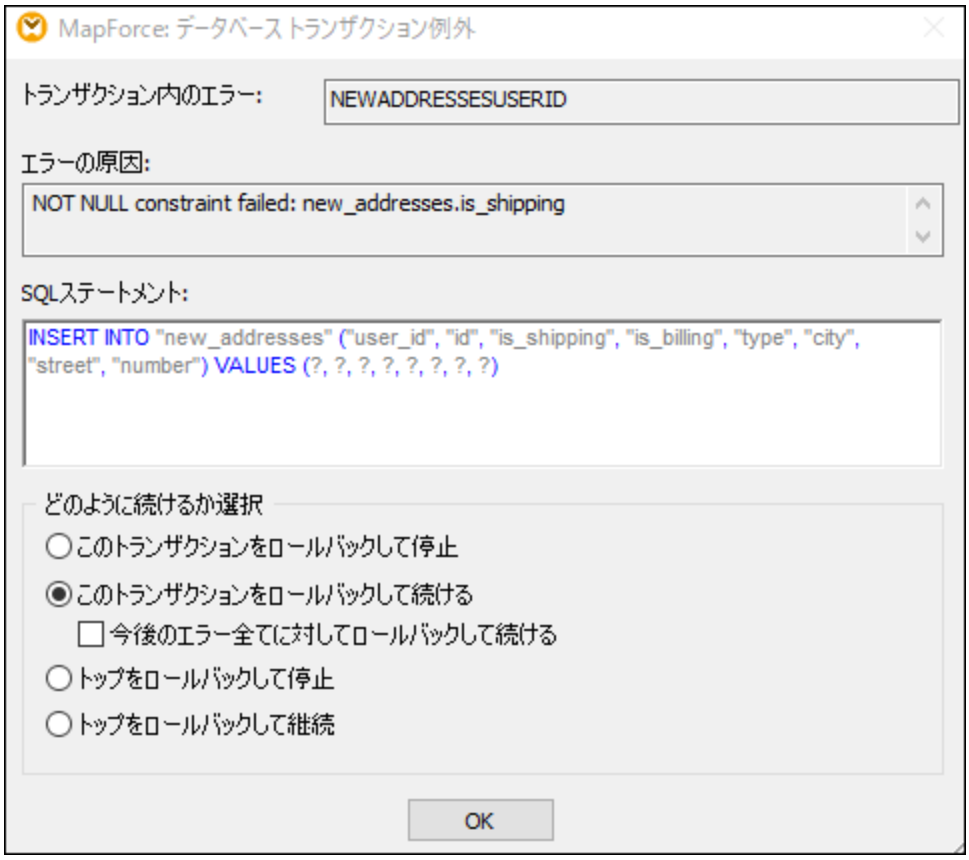
- 全ての変更がロールバックされます。
- `new_users` テーブル内にレコードは挿入されません。
- `new_addresses` テーブル内にレコードは挿入されません。

シナリオ B) 「現在のトランザクションをロールバックし、継続する」を実行する

必要条件: 例外を生成するレコードをスキップし、有効なレコードを挿入し続ける。この目的を達成するためにマッピングを構成する方法は以下のとおりです:

1. `new_users` テーブルの横の DEL, A:In ボタンをクリックし、「**トランザクションの使用**」チェックボックスを選択し、「**現在のトランザクションをロールバックし、継続する**」を選択します。
2. `new_addresses` テーブルの横の A:In ボタンをクリックし、「**トランザクションの使用**」チェックボックスを選択し、「**現在のトランザクションをロールバックし、継続する**」を選択します。

マッピングをこれらの設定(データベーストランザクション例外 ダイアログボックス)で実行します。以下に示されるようにオプションを選択して、現在のトランザクションをロールバックし、マッピングの実行を継続します:



結果は以下のようになります:

- エラーを伴うトランザクションはすべてロールバックされます。
- (データベース例外によりユーザーレコードが生成されていないため) 5 人中5人のユーザーが挿入されます
- (3 つのアドレスが値を含み例外が生成されているため) 5 つのアドレス中2 つのアドレスが挿入されます

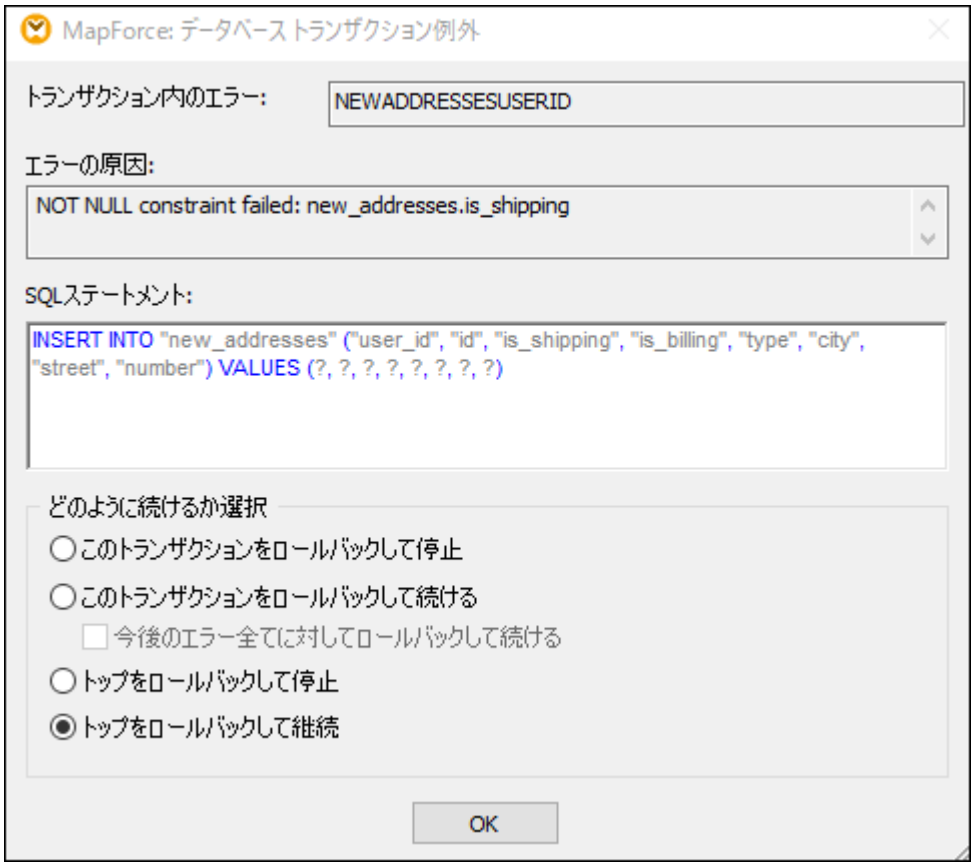
シナリオ C) 「トップトランザクションをロールバックし、継続する」を実行する

必要条件: アドレスの挿入に失敗した場合、親ユーザーレコードも挿入されるべきではありません。これを達成するためにマッピングを以下のように構成します:

1. データベースコンポーネントのタイトルバーをダブルクリックし、「**トランザクションの使用**」チェックボックスをクリアします。
2. `new_users` テーブルの横の DEL, A:In ボタンをクリックし、「**トランザクションの使用**」チェックボックスを選択し、「**現在のトランザクションをロールバックし、継続する**」を選択します。
3. `new_addresses` テーブルの横の A:In ボタンをクリックし、「**トランザクションの使用**」チェックボックスを選択し、「**トップトランザクションをロールバックし、継続する**」を選択します。

この設定でマッピングを実行すると、データベーストランザクション例外ダイアログボックスが表示されます。以下に示されるようにオプションを選択し、プロンプトされる回数とおり選択を繰り返します:

(C) 2015-2021 Altova GmbH



結果は以下のようになります:

- 2 人のユーザーが挿入されます（Sharda Junker、Tobie Hughey）
- 1 つのアドレスが挿入されます

説明: ユーザー「Sharda Junker」はアドレスを持たない唯一のユーザーです。このレコードのために例外は発生しません。2番目のユーザーである「Tobie Hughey」はこのアドレスレベルで例外が発生していない唯一のユーザーであるため挿入されています。他のユーザーレコードは、例外が発生しトランザクションがロールバックされているアドレスを持っているため挿入されていません。

nulls を持たないため、例外をスローしない 2 つのアドレスが存在します。これらは `id=1` と `id=4`. を持つアドレスです。最初のアドレスのための挿入トランザクションは、しかしながら、ユーザートランザクションがロールバックされる必要があったため、ロールバックされています。このため、 `id=4` を持つアドレスのみが挿入されています。

同じ結果を以下のように達成することができます:

1. `new_users` テーブルの横の DEL, A:In ボタンをクリックし、「**トランザクションの使用**」チェックボックスを選択し、「**現在のトランザクションをロールバックし、継続する**」を選択します。
2. `new_addresses` テーブルの横の A:In ボタンをクリックし、「**トランザクションの使用**」チェックボックスをクリアします。

MapForce Server を使用してマッピングを実行する方法

ライセンス供与済みの MapForce Server を使用する場合、コマンドライン、Linux、macOS、または Windows マシン上でマッピングを以下の通り実行することができます:

1. メニューコマンド「ファイル | MapForce Server 拡張ファイルにコンパイルする」,を使用して MapForce Server 拡張ファイル(.mfx) へマッピングをコンパイルします。 MapForce Server 拡張ファイルにマッピングをコンパイルするも参照してください。
2. .mfx ファイルと **ExceptionsDemo.sqlite** データベースをサーバーマシン上のディレクトリにコピーします。これを「作業ディレクトリ」と呼ぶことにします。
3. コマンドプロンプトを作業ディレクトリに変更し、以下のコマンドで MapForce Server を実行します:

```
mapforceserver run DatabaseExceptions.mfx
```

メモ

- **mapforceserver** は使用中のオペレーティングシステムのための MapForce Server 実行可能ファイルへのパスです。.mfx ファイルへのパスを適用できるように変更します。例えば、Linux マシンでは、パスは以下のとおりです : `/opt/Altova/MapForceServer2021/mapforceserver`.

サーバーの実行の場合、オンデマンド、または再帰的に API 呼び出し、または FlowForce Server ジョブとしてマッピングを実行することができます。詳細に関しては、MapForce Server を使用して自動化するを参照してください。.

6.2.11 データベースのトレースとエラーのログ

マッピングがデータをデータベースに書き込む場合、オプションでデータベーストレースとエラーログを有効化することができます。マッピングの作動時データベースへのマッピングにより全ての変更をトラックする場合、役に立ちます。トレースがオンの場合、データベース挿入、または更新アクション、またはエラーが XML ファイル内でログされている場合、これらを後に分析または、更に自動化された方法で処理することができます。XML 以外の書式でログファイル作成する場合、任意でデータを MapForce によりサポートされるコンポーネントの種類からマップすることができます(例、テキストファイル、または他のデータベース)。

メモ マップがトレースとログをサポートするには、マッピング言語が C#、C++、または Java に設定されている必要があります。

データベースのトレースを構成することは可能です。構成により、すべてのメッセージ、またはエラーのみをトレースし、エラーのみを、またはトレースを完全に無効化することもできます。更に、次のレベルでトレースを使用することができます:

- データベースコンポーネントレベル。このレベルでは、そのデータベースコンポーネントのためにトレースを効果的にオンとオフに切り替えることができます。これは複数のターゲットデータベースコンポーネントを持つマッピングにとり役に立ちます。そして、一部のためにのみトレースが必要となります。
- テーブルまたはストアドプロシージャレベル。データベースコンポーネント上では、各テーブル、またはストアドプロシージャのためにトレースを行うかを決めることができます。有効化されると、このレベルでのトレースは "Insert All"、"Update If"、"Ignore If"、"Delete If" などのテーブルアクションのためにこのイベントを含みます。ストアドプロシージャに関連したイベントがトレースされます。
- データベースフィールドレベル。デフォルトで、全てのフィールドがトレースされますが、特定のフィールドをトレースから除外、または エラー時にのみトレースファイルに含まれるように指定することができます。

重要な点は、上記の3つのレベルは階層で、トレース設定は上から下に伝達されます。このため、データベースコンポーネントレベルでトレースを無効化します。テーブルまたはストアドプロシージャレベルでは有効化することは無理です。同様に、テーブルレベルでトレースを無効化することができますが、データベースフィールド(列) レベルでは、設定することはできません。同様がトレースレベルを狭める際に適用することができます。例、データベースコンポーネントレベルでのエラーにのみトレースを制限する場合、テーブルまたはストアドプロシージャレベルでフルトレースを使用することは不可能です。

データベースコンポーネントレベルでのトレース

データベースコンポーネントレベルでトレースを有効化する方法:

1. データベースコンポーネントを右クリックし、「プロパティ」をコンテキストメニューから選択します(または、データベースコンポーネントのタイトルバーをダブルクリックします)。これにより、データベースコンポーネント設定 ダイアログボックスが開かれます。

2. 希望するトレースレベル(**常に**、**エラー**、**無効化済み**) を選択します。「**エラー**」を選択すると、データベースエラーがマッピングの実行時に発生した場合にトレースファイルを作成します。
3. 「**ファイルのトレース**」の横にトレースのすべての詳細が書き込まれるXMLファイルの名前、またはパスを書き込みます(例、**Log.xml**)。フルパスの代わりにファイル名を入力すると、トレースファイルがマッピングが作動する同じディレクトリ内に作成されます。

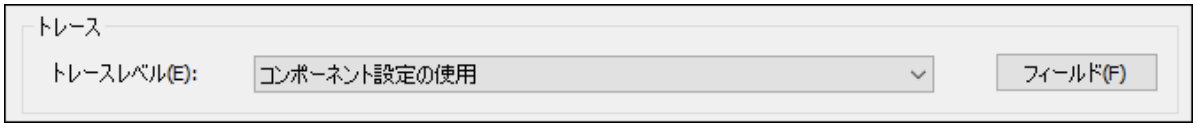
トラッキングを有効化した後、データベースコンポーネントはマッピング上の構造を変更します。具体的には、データベース構造に加え、階層内のトップアイテムがトレースファイルであるアイテムの新規のノードの階層が表示されます。詳細に関しては、ファイル構造のトレースを参照してください。.

データベースコンポーネントレベルでのトレースの有効化により、マッピング接続の存在するそのコンポーネント上のすべてのテーブルまたはストアドプロシージャのために有効化されます。必要であれば、下記の通りテーブルまたはストアドプロシージャレベルでトレースをトラックすることができます。マッピング上では役割を果たさないコンポーネント上に表示されているテーブル、またはストアドプロシージャはトレースされません(すなわち、マッピング接続は存在しません)。

テーブルまたはストアドプロシージャレベルでのトレース

テーブルまたはストアドプロシージャレベルでのトレースの有効化:

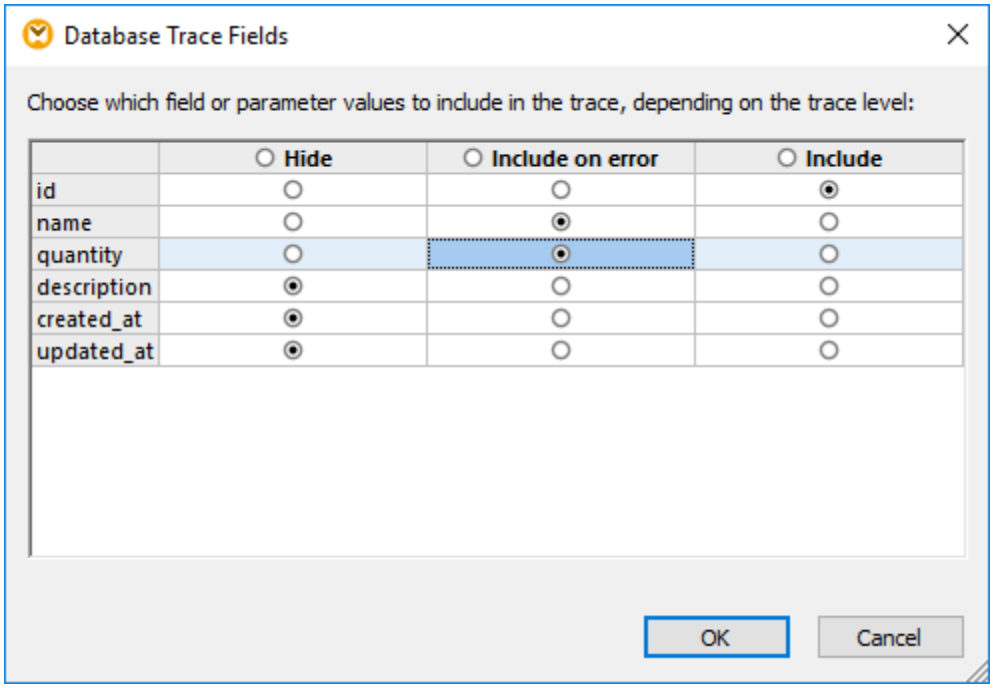
1. データベースコンポーネントレベル出のトレースレベルが**常に**または**エラー**に設定されていることを確認してください(上記参照)。
2. 以下の1つを行います:
   - トレースを有効化するテーブルの横のテーブルアクションボタン(例、) をクリックします。
   - ストアドプロシージャの場合、「**コンテキストメニューの表示**」 ボタンをクリックし、「**プロシージャの設定**」をコンテキストメニューから選択します。



3. トレースレベルを選択します。「**コンポーネント設定の使用**」オプションはコンポーネントレベルで定義された同じ設定を継承します。「**エラーに制限**」オプションはトレースをエラーイベントに制限します。「**常に無効化**」はこのテーブル、またはストアドプロシージャではトレースが発生しないことを意味します。

データベースフィールドレベル出のトレース

データベースコンポーネントレベルとテーブルまたはストアドプロシージャレベル出のトレースを有効化すると、全てのフィールド(データベース列)はデフォルトでトレースレポートに含まれます。任意で、データベースフィールド、またはストアドプロシージャパラメータートレースを制限することができます。これを行うには、テーブルアクション、またはデータベースプロシージャ設定ダイアログボックス上の**フィールド**ボタンを押します。



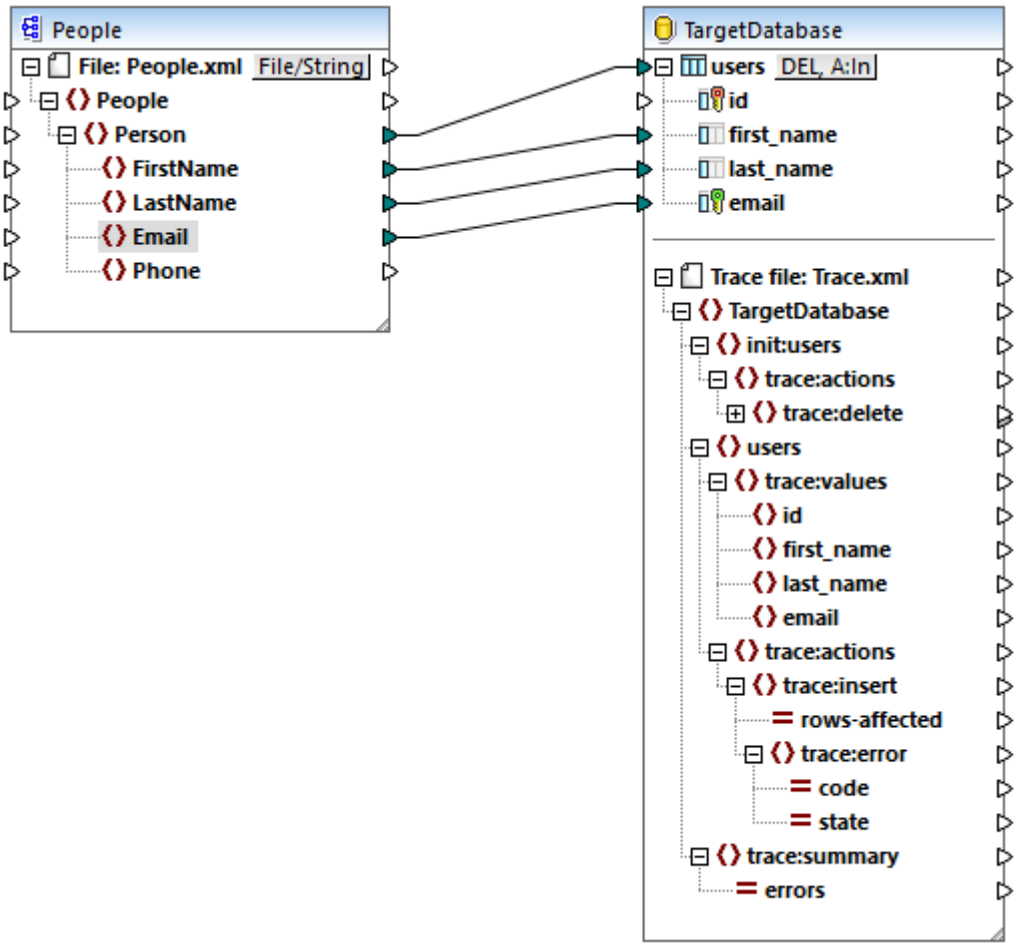
このダイアログボックス上の選択は、ファイル構造のトレースに影響を与えます。 例、隠すフィールドは、トレースファイル上で表示されません。これを行う必要があるかもしれません。例えばログファイルのサイズを減らす場合などが挙げられます。

複数のフィールドを同時に隠す、または含む場合、ヘッダー内にそれぞれのオプションをクリックしてください。

6.2.11.1 ファイル構造のトレース

データベースコンポーネントのためにトレースが有効化されている場合、トレース構造がコンポーネントの下の部分で使用できるようになります。例:

(C) 2015-2021 Altova GmbH



上記の通り、トレース構造のトップノードはマッピングが実行される際に書き込まれるトレースファイルの名前を示しています。トレース構造はマッピング内で発生するデータベーステーブル、またはストアドプロシージャをモデルとしています。このサンプルでは、データベースの名前は "NanonullTarget" ですので、トレース構造内に同様のアイテムが存在します。

"NanonullTarget" の子アイテムは "products" データベーステーブルに対応する **products** です。このサンプルでは、データベースコンポーネントにはテーブル "products" が 1 つのみ存在します。しかしながら、多くのテーブルのコンポーネント内では、トレース構造はコンポーネント上に存在する各テーブルを含みます。テーブル、またはストアドプロシージャをデータベースコンポーネントに追加、または削除すると、トレース構造が対応して更新されます。命令に関しては、データベースオブジェクトの追加、編集、および、削除を参照してください。.

このサンプルでは存在しませんが、**trace:init** 構造も表示される場合があります。これは、マッピングがメインの挿入、更新、または他のアクションの前に初期アクションが構成されている場合に発生します。例、新規のテーブルを挿入する前にすべての既存のレコードを削除する場合、**trace:init** 構造が存在します。（削除、またはカスタム SQL クエリなどの）初期アクションのトレース情報が収集されます。

trace:values 構造は、データベースのテーブルのすべての列を表示します（この場合、「products」テーブルの列）。ストアドプロシージャのために、この構造はストアドプロシージャのパラメーターを表示します。デフォルトで、全ての列はトレースに設定されており、上記の通りどの列がトレースされるかを構成することができます。

trace:actions アイテムは、どのアクションがデータベーステーブル内で現在設定されているかを表示します。このサンプルでは、データベース更新アクション A:Up が発生するように設定されており、**trace:update** アイテムを使用することができます。データベース挿入アクション A:In を実行するようにコンポーネントを変更すると、トレース構造は他のアクションのために **trace:insert** に変更されます。ストアドプロシージャの場合、**trace:execute** アクションが代わりに表示されます。

各トレースアクションには対応するデータベースアクションにより影響を受ける行数を指定する **rows-affected** 属性が存在します。

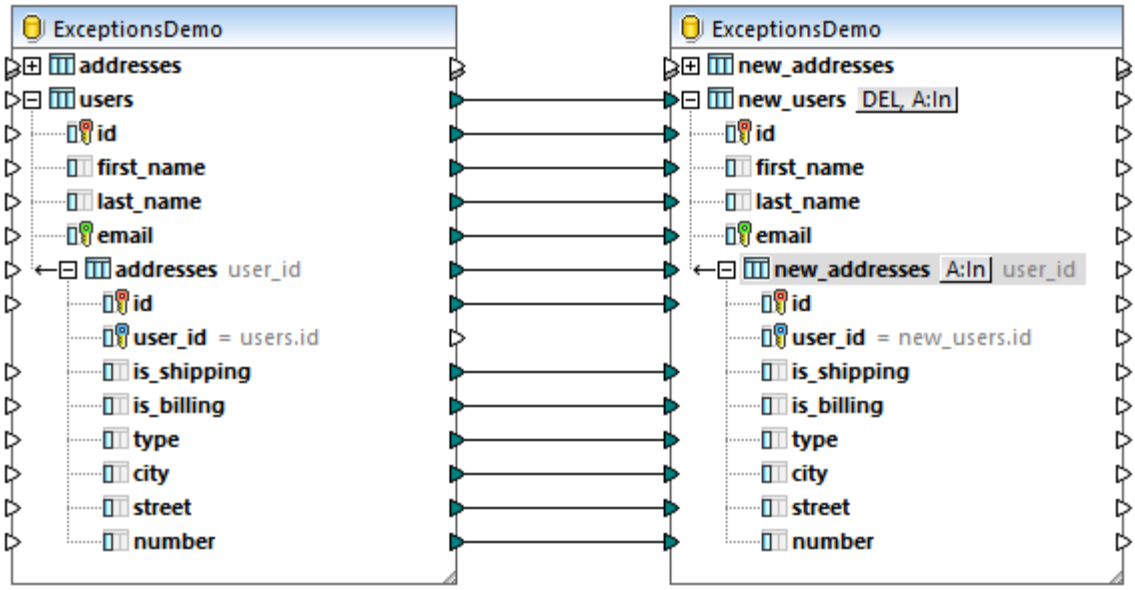
trace:error アイテムはマッピング実行中にデータベースエラーが発生した場合に作成されます。以下の2つの属性が存在します: **code** と **state**。エラーのテキストと属性の値はデータベースドライバーにより提供され、各データベースの種類により異なります。

trace:summary アイテムには、発生したエラーの数量をレポートする **errors** 属性が含まれます。エラーが発生しない場合、属性の値は **0** です。

6.2.11.2 例: データベースエラーのログ

このサンプルは、SQLite データベースにデータを挿入するためにデータベーストレースとエラーログの構成方法について説明されています。命令は MapForce によりサポートされる他のデータベース型、および、("update"、"ignore"、"delete" などの) 他のデータベースアクション型に類似しています。

次のパスでこのサンプルで参照されているデモマッピングを見つけることができます: **<マイドキュメント>\Altova\MapForce2021\MapForceExamples\Tutorial\DatabaseExceptions.mfd**。このマッピングは、データベースの2つのテーブルからのデータを同じデータベースのテーブルの新規のペアに挿入し、例: トランザクションロールバック内で更に詳しく説明されています。データベース内のレコードの一部は例外を生成するために故意的に null に設定されています。



DatabaseExceptions.mfd

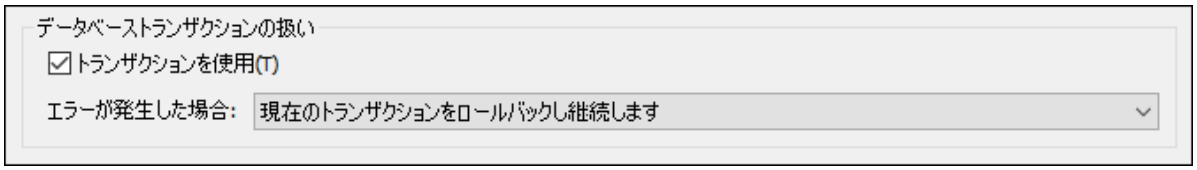
このサンプルでは、ビジネスの必要条件は生成されるエラー数とエラーが発生する箇所を決定するために、ターゲットデータベースコンポーネントのためにトレースを有効化することです。下のステップはこの目的を達成する方法を説明しています。

ステップ 1: トランザクションロールバックの有効化

このデモマッピングでは、データベースコンポーネントレベルでのトランザクションロールバックを使用するために構成されています。これは、最初の例外が発生すると、マッピングは実行を停止し、最初の状態まで全てのデータをロールバックします。トランザクションロールバックオプションを変更し、マッピングの実行がレコードの一部のために例外が発生しても継続するようにします:

1. `new_users` テーブルの横の DEL, A:In ボタンをクリックし、「**トランザクションの使用**」チェックボックスを選択し、「**現在のトランザクションをロールバックし継続する**」を選択します。

(C) 2015-2021 Altova GmbH



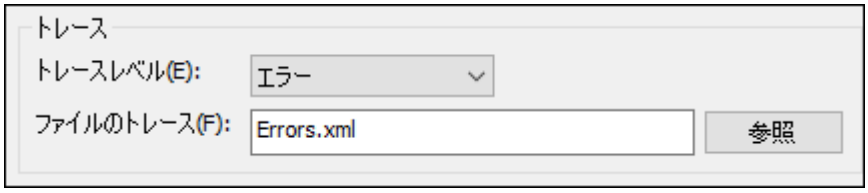
2. `new_addresses` テーブルの横の A:In ボタンをクリックし、「**トランザクションを使用**」チェックボックスを選択し、「**現在のトランザクションをロールバックし継続する**」を選択します。

トランザクションロールバックを完全に無効化することができます。しかしながら、これを行うとエラーが最初に発生するとマッピングの実行は停止され、変更をロールバックすることはできません。ですから奨励されません。この場合トレースとログは、最初に発生した例外の時点の実行のみを反映します。トランザクションロールバックに関する詳細は[データベース例外の扱い](#)を参照してください。

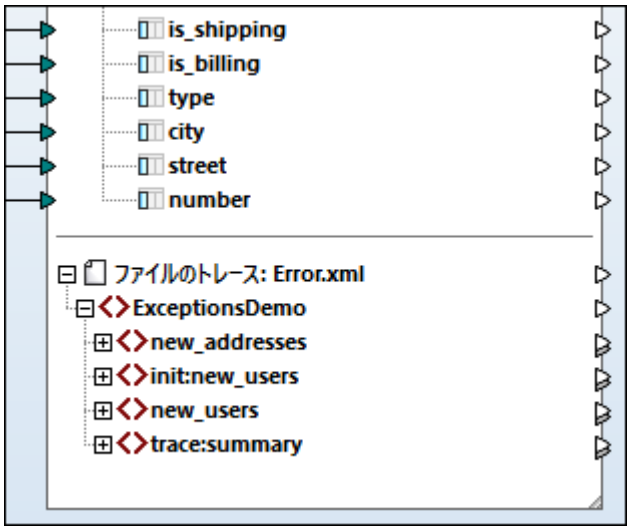
ステップ 2: エラーログの構成

トレースを有効化するには、以下を行います:

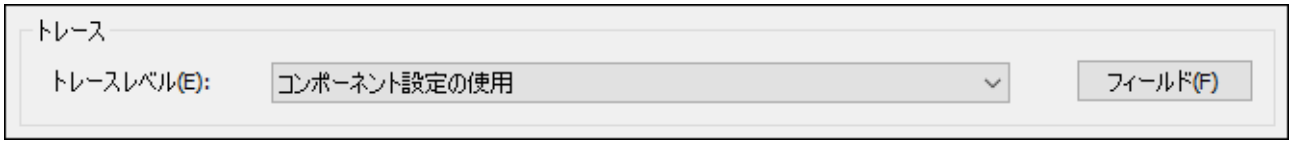
1. データベースコンポーネントのタイトルバーを右クリックし、「**プロパティ**」をコンテキストメニューから選択します(または、データベースコンポーネントのタイトルバーをダブルクリックします)。これによりデータベースコンポーネント設定ダイアログボックスが開かれます。
2. 「**エラー**」をトレースレベルとして選択します。
3. 「**トレースファイル**」の横に、トレースファイルの名前を入力します。(この場合 **Errors.xml**)。他のマシン、またはオペレーティングシステム上の MapForce Server でマッピングを実行する場合は、絶対パスを入力しないでください。



ターゲットデータベースコンポーネントは構造を変更し、トレースファイル構造を使用して追加コンパートメントを表示します:



データベースコンポーネントレベルでのトレースが有効化されています。今後、トレース設定はこのデータベースコンポーネント上のテーブルアクションにより継承されます。このサンプルでは、`new_users` と `new_addresses` テーブルの挿入アクションにより継承されます。これらのテーブルのそれぞれのために、アクションボタン A:In をクリックすると、設定はこの事実を表示します:



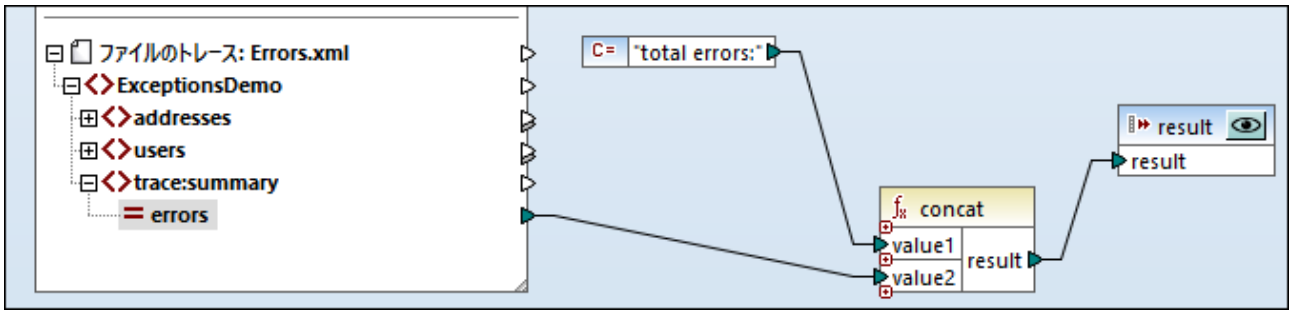
任意で、**フィールド** ボタンをクリックして、どのフィールドがトレースファイル内に含まれるか、非表示にされるかを構成します。このサンプルでは、全てのフィールドが含まれています。

ステップ 3: マッピングの出力の構成方法

このステップは、他のコンポーネントの種類(この場合、単純型出力コンポーネント) へトレースファイルからの値をマップする方法を示しています。このステップは任意です。エラーログファイルは他のコンポーネントにトレースファイルをマップしない場合でも作成されます。このサンプルでは、マッピングを明示的に構成して、出力内に発生したエラーの総数を返すように構成します:

1. 「**挿入 | 出力の挿入**」メニューコマンドを選択して、単純型出力コンポーネントを追加します。単純型入力コンポーネントに関する詳細は、マッピングから文字列の値を返す方法を参照してください。
2. 「**挿入 | 定数**」メニューコマンドを選択して、定数を追加します。定数は、マッピング出力テキストを明確にするために静的なテキスト「エラーの総数: 」を提供します。
3. `concat` 関数をライブラリからマッピングにドラッグします。ビルトイン関数に関する詳細は、マッピングにビルトイン関数を追加する方法を参照してください。

下に示される通り、`concat` 関数は、定数のテキストとトレースファイルにより与えられる **errors** 値を結合します。



上記の構成の通り、MapForce Server を使用してマッピングが実行されると、出力はテキストを表示します: **value** が実行されると、ランタイムに遭遇したエラーの数量を示す "エラーの総数: {value}"。

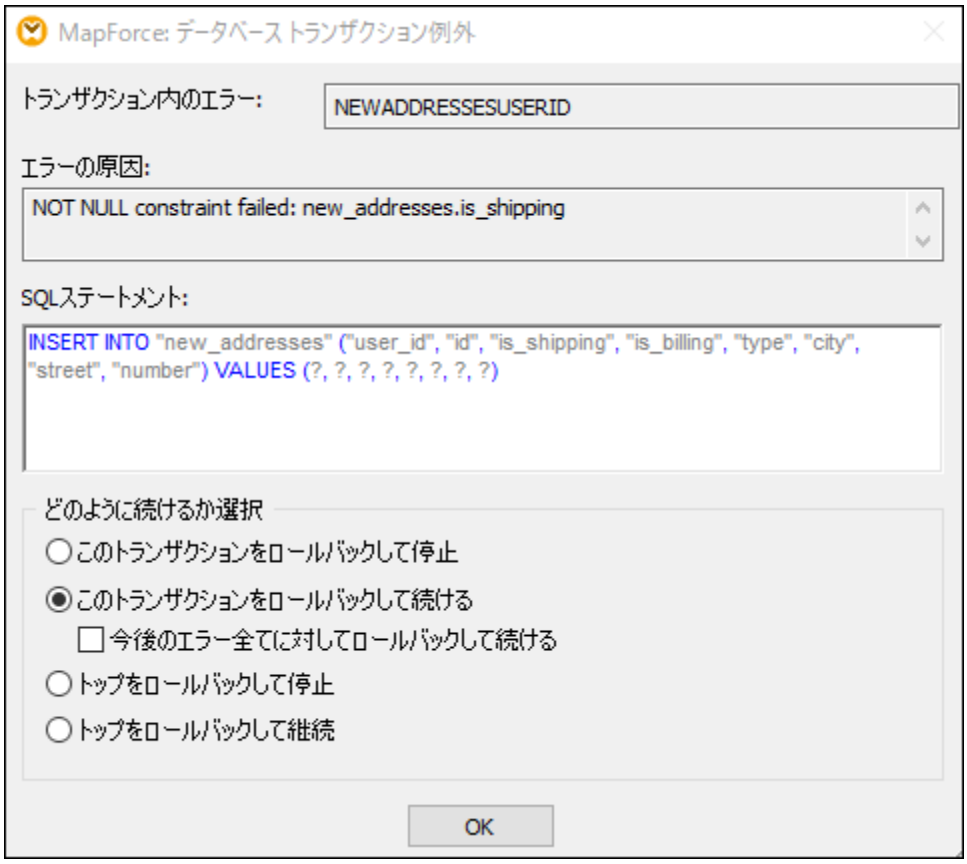
この時点では、「結果」コンポーネント上の**プレビュー** ボタンを押して、MapForce 内でマッピングを実行することは不可能で、エラーを引き起こします。単純な出力はトレースファイルからデータを読み取るため、トレースファイルが最初に存在することが重要です。次のステップはトレースファイルの作成方法を示しています。

ステップ 4: マッピングの実行

MapForce を使用してマッピングを実行する方法:

1. データベースコンポーネント上の「**プレビュー**」 ボタンをクリックします。
2. 「**出力**」タブをクリックします。
3. 出力メニュー上で、「**SQL-スクリプトの実行**」をクリックします。この時点では、マッピングで最初の例外が発生します。これは期待されるものです。データベーストランザクション例外ダイアログボックスが表示されます。

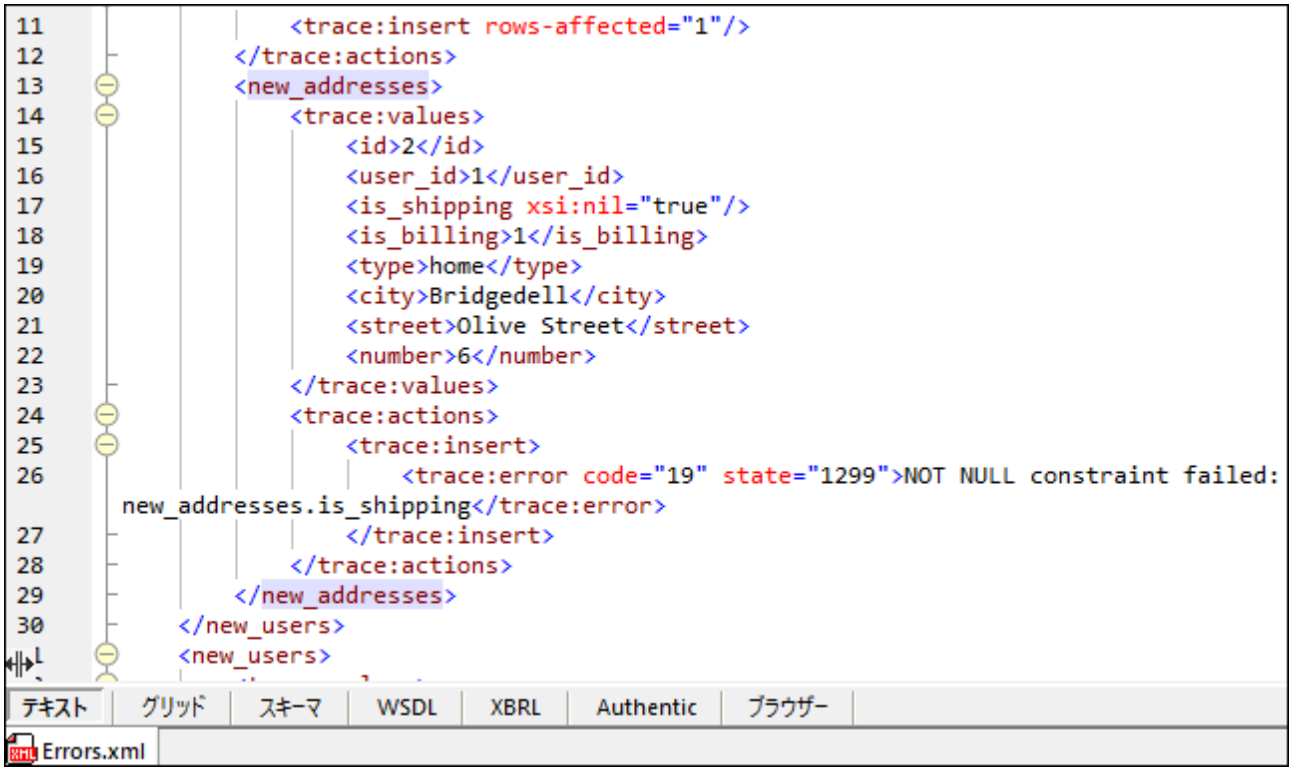
(C) 2015-2021 Altova GmbH



4. 上記の通りロールバックオプションを選択し、「OK」をクリックします。

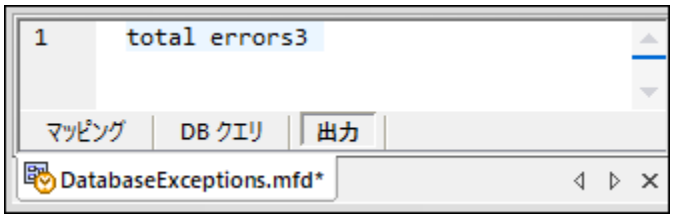
メモ MapForce Server によりマッピングが実行されると、ダイアログボックスは表示される、MapForce 内で構成されたロールバック設定に従い、エラーのトランザクションが自動的にロールバックされます。

このサンプルでは、マッピングの実行が完了すると、失敗したすべてのトランザクションがロールバックされ、成功した実行のみがデータベースにコミットされます。発生したエラーは、**Errors.xml** ファイルに書き込まれます。以下にこの実行の一部が表示されています:



ログファイルが生成されたので、MapForce 実行内のエラーの総数を返す出力をプレビューすることができます:

1. マッピングに戻り、「結果」コンポーネント上の「プレビュー」 ボタンをクリックします。
2. 「出力」タブをクリックして、結果を確認します。



マッピングのこの部分は、以前に作成されたログファイルからのデータを読み取ります。ログファイルが存在しない場合、エラーが生成されます。

(C) 2015-2021 Altova GmbH

6.3 CSV とテキストファイル

MapForce は、CSV (コンマで区切られた値) とFLF (固定長フィールド) テキストファイルなどのテキストベースのファイルから、またはへの、データのマッピングのためのサポートを含みます。例: 以下のようなデータ変換を作成することができます:

- XML スキーマとフラットファイルフォーマット
- データベースとフラットファイルフォーマット
- UN/EDIFACT ならびにANSI X12 と、フラットファイルまたはデータベース

CSV の場合、ファイルはコンマだけでなく、タブ、セミコロン、スペースまたは他のカスタム化された値などの区切り記号を含むことができます。

CSV とFLF ファイルの他に、MapForce FlexText (このモジュールはMapForce Enterprise Edition で使用することができます) を使用すると、複雑またはカスタム化された構造のためのテキストファイルから、またはへのマッピングが可能になります。FlexText を使用すると、("FlexText テンプレート"を使用して) 他のフォーマットにマッピングするために カスタムテキストデータを定義することが可能になります。詳細は MapForce FlexText を参照ください。

ファイルテキストから、またはテキストファイルへのデータのマッピングは以下の言語でサポートされています: Java、C#、C++、または BUILT-IN。

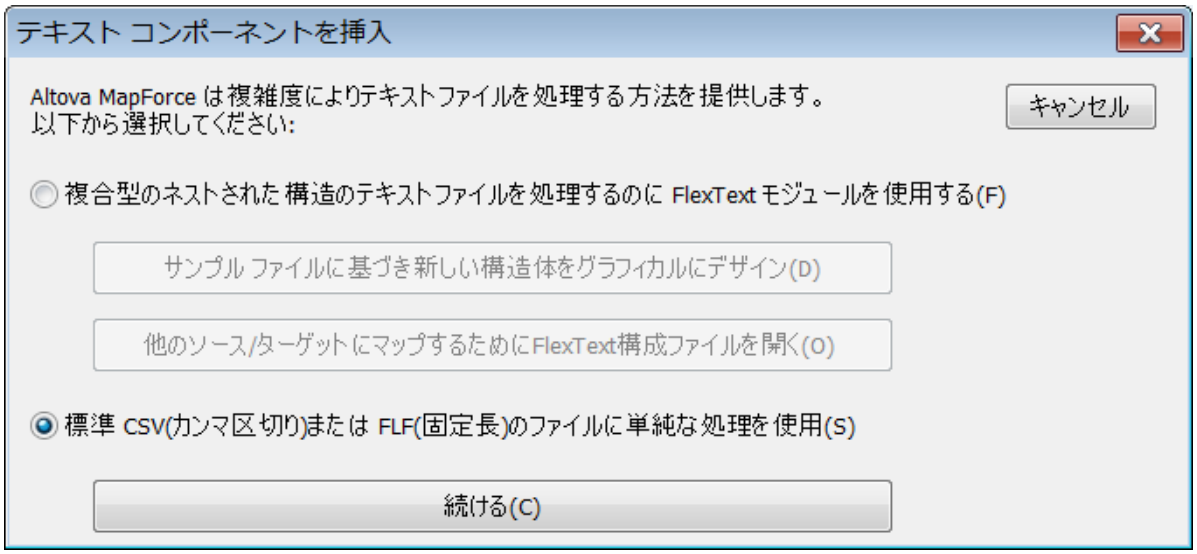
マッピングされたフラットなファイルデータ生成/保存するには2 つの方法があります:

- 出力タブをクリックすることで、内蔵の実行エンジンを使ったプレビューが生成され、「出力 | 出力ファイルの保存」メニューオプションを選択するか、 アイコンをクリックすることで結果を保存できます。
- メニューオプションの「ファイル | コードの生成 | Java、C#、または C++」を選択し、生成されたコードをコンパイル、実行します。

6.3.1 例: CSV ファイルから XML へのマッピング

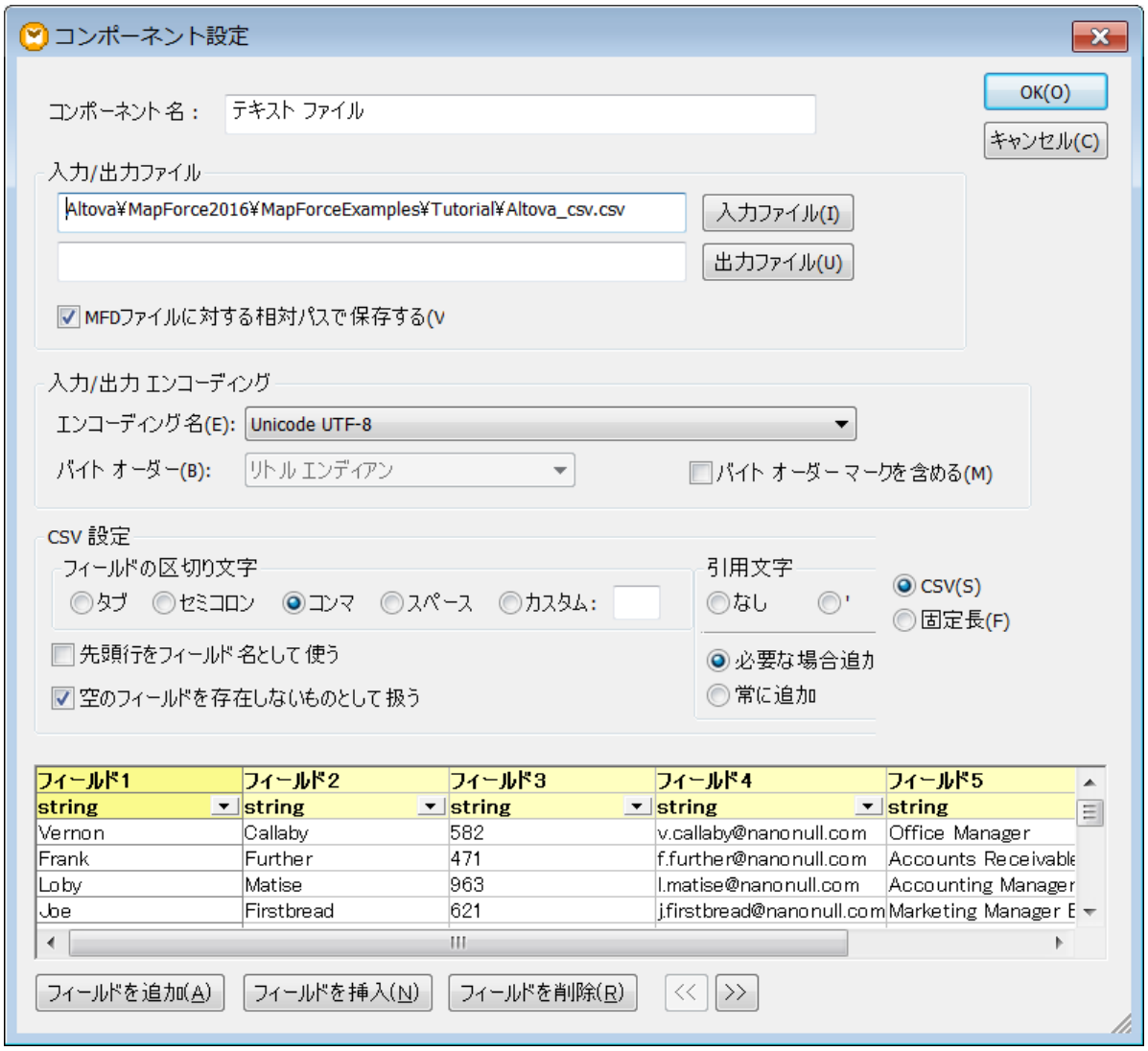
この例ではMFCompany.xsd スキーマファイルをベースに、単純な CSV ファイルを XML ファイルへマッピングします。以下の例で使用されるファイルは全て <マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥Tutorial¥ フォルダーに収められています。

1. ツールバーにて Java、C#、C++ または BUILT-IN のアイコンが選択されていることを確認します。
2. メニューオプションの「挿入 | テキストファイル」を選択するか、テキストファイルを挿入アイコン ( ) をクリックします。「テキストコンポーネントを挿入」ダイアログボックスが表示されます。

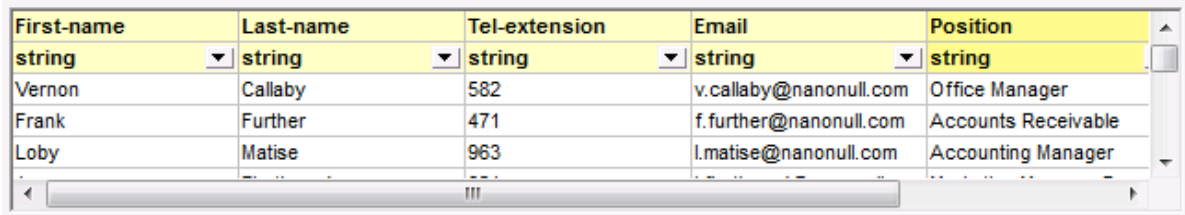


3. 標準 CSV または固定長のファイル... のラジオボタンをクリックして、続けるボタンをクリックします。
4. コンポーネント設定ダイアログボックスから、入力ファイルをクリックして、**Altova_csv.csv** ファイルを参照します。ファイルのコンテンツは、ダイアログボックスの下の部分に表示されます。プレビューモードの場合最初の20列のみが表示されることにご注意してください。

(C) 2015-2021 Altova GmbH



5. フィールド 1 のヘッダーをクリックして、テキストの内容を（例えばFirst-name に）編集します。他のフィールドも同様の方法で編集します: フィールド 2 => Last-name、フィールド 3 => Tel-extension、フィールド 4 => Email、フィールド 5 => Position。ヒント: タブ キーを押下することで、フィールド間を移動することができます: ヘッダー1 、ヘッダー2 など。

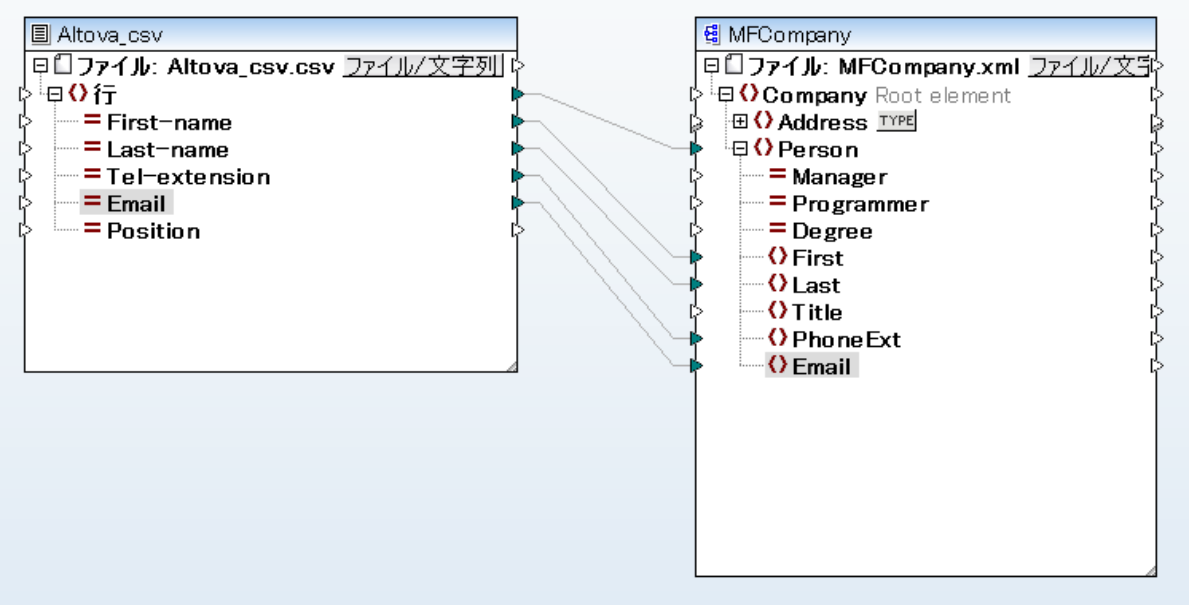


First-name	Last-name	Tel-extension	Email	Position
string	string	string	string	string
Vernon	Callaby	582	v.callaby@nanonull.com	Office Manager
Frank	Further	471	f.further@nanonull.com	Accounts Receivable
Loby	Matise	963	l.matise@nanonull.com	Accounting Manager

6. 「OK」ボタンをクリックします。
7. （プロンプトにて表示される）コンポーネント名を変更ボタンをクリックすることで、コンポーネント名がセットされます。CSV コンポーネントがマッピングに挿入されます。
8. メニューオプションから「挿入 | XML スキーマ/ファイル」を選択し、**MFCompany.xsd** を選択します。
9. スキップをクリックし、ルート要素として Company を選択します

10. 両コンポーネントで対応するアイテムをマッピングします。行アイテムをスキーマターゲットの**Person** アイテムへ接続し、出力タブをクリックしてマッピングの結果を確認します。

> CSV コンポーネント内の行アイテムからのスキーマ内の `Person` アイテムへの接続は、どの要素が反復するか定義するため必要不可欠です。CSV ファイル内の各行に対して、新しい `Person` 要素がXML 出力ファイル内で作成されることを意味します。



11. 出力タブをクリックして結果を確認します。

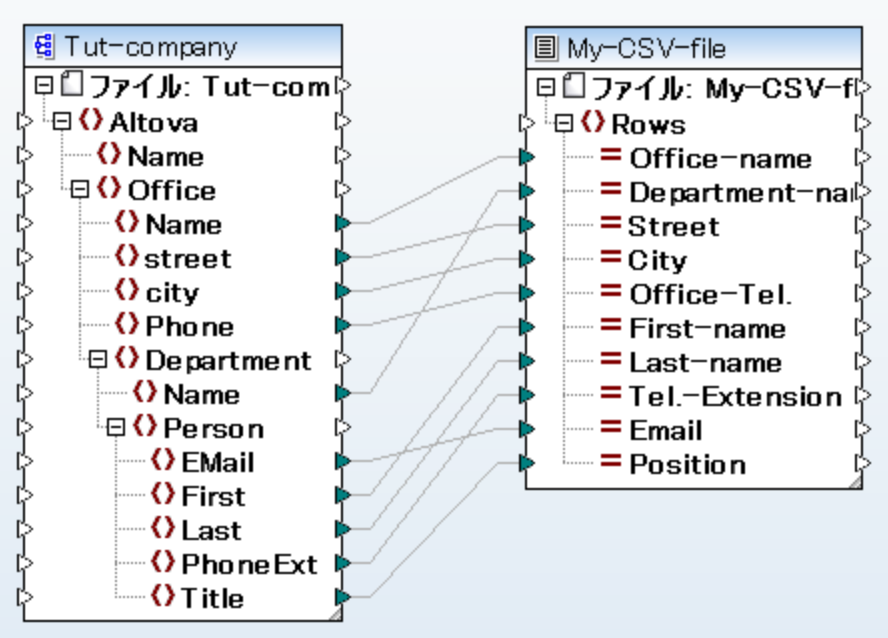
```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <Company xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:
3    <Person Manager="true">
4      <First>Vernon</First>
5      <Last>Callaby</Last>
6      <PhoneExt>582</PhoneExt>
7      <Email>v.callaby@nanonull.com</Email>
8    </Person>
9    <Person Manager="true">
10     <First>Frank</First>
11     <Last>Further</Last>
12     <PhoneExt>471</PhoneExt>
13     <Email>f.further@nanonull.com</Email>
14   </Person>
15   <Person Manager="true">
```

CSV ファイルからのデータがXML ファイルへマッピングされました。

6.3.2 例: アイテムの反復

ターゲット CSV ファイル内でアイテムの反復（複数の行）の方法を説明します。このサンプルで使用されるマッピングデザインファイルは、<マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥Tutorial¥フォルダー内にある **Tut-xml2csv.mfd** で確認できます。

(C) 2015-2021 Altova GmbH

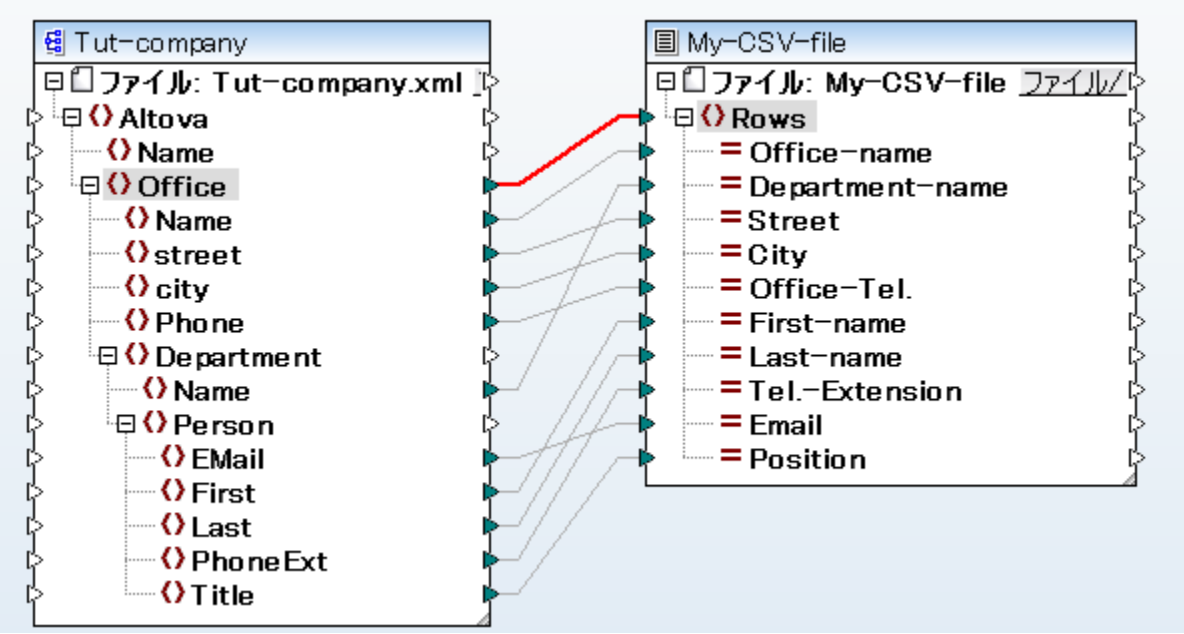


Tut-xml2csv.mfd

このマッピングは、意図的に不完全し作成されています。このサンプルを、メニューコマンド「**ファイル | マッピングの検証**」を使用して検証しようと試みると、検証エラーが生じる可能性があります。また、マッピング出力をプレビューすると、目的に合った、または合わない単一の行が作成されます。

XML ファイル内のアイテムのシーケンスから CSV ファイル内に複数の行を作成することが目的だと仮定します。これはターゲット CSV ファイルの `Rows` アイテムに接続を描くことで達成することができます。

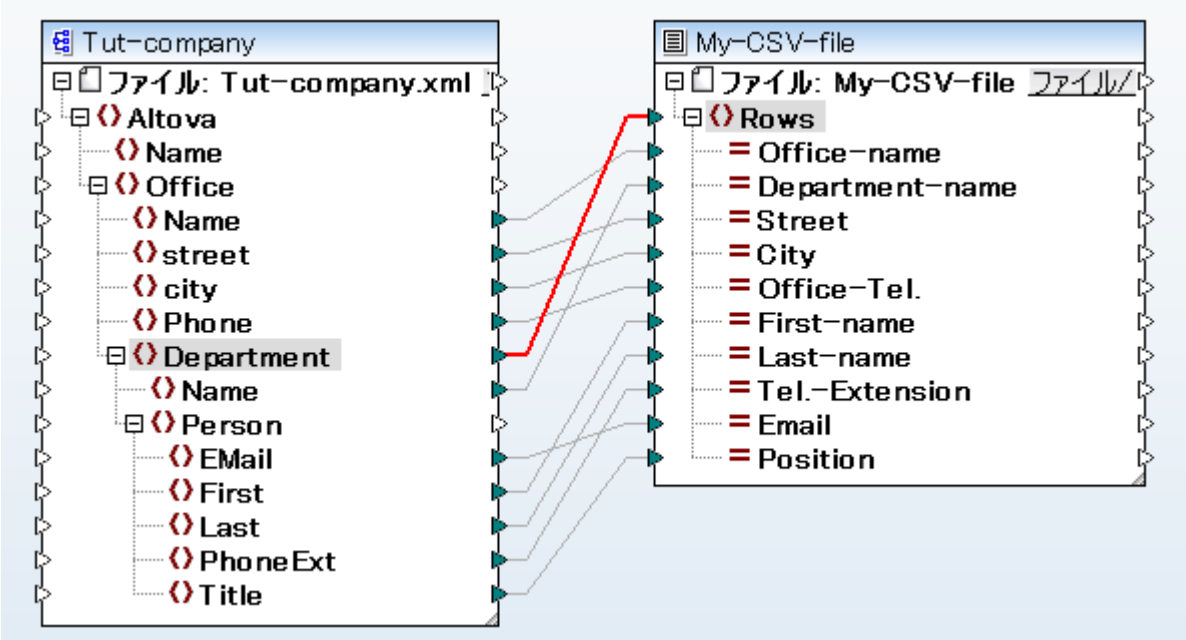
全てのオフィスに関するデータを CSV ファイルへ出力するには、Office を行アイテムへ接続する必要があります。MapForce で以下の命令を与えます：Office アイテムを行アイテムへ接続することで、各 Office（とマッピングされたアイテム）がターゲット CSV ファイルに表示されます。



CSV コンポーネント内の Rows アイテムは、接続されたアイテムのシーケンスのための反復子としての役割を果たします。ですから、Office アイテムを接続すると、出力はソース XML 内で見つかる各のために行を作成します。

```
1   "Microtech, Inc.",Level 1 support,Major Ave 1,Vancouver,558833
2   "Microtech Partners, Inc.",Level 2 support,Perro Bvd 1324,Otto
3
```

Department から **Rows** アイテムへのマッピングを行うことで、ソース内の各 department のために行が作成されます。



出力は、以下のようになります:

```
1   "Microtech, Inc.",Admin,Major Ave 1,Vancouver,5588339,Clive,Clo
2   "Microtech, Inc.",Sales and Marketing,Major Ave 1,Vancouver,558
3   "Microtech, Inc.",Manufacturing,Major Ave 1,Vancouver,5588339,K
4   "Microtech, Inc.",Level 1 support,Major Ave 1,Vancouver,5588339
5   "Microtech Partners, Inc.",Admin,Perro Bvd 1324,Ottowa,3549202,
6   "Microtech Partners, Inc.",Sales and Marketing,Perro Bvd 1324,O
7   "Microtech Partners, Inc.",Level 2 support,Perro Bvd 1324,Ottow
8
```

最後に、Person を Rows アイテムにマッピングすると、結果はすべての Person が出力になります。この場合、MapForce は、記録を以下のように反復します: 各 Department 内の各 Office 内の各 Person.

6.3.3 例: CSV と FLF ファイルから階層を作成する

このサンプルは、以下のパスで見つけることができます: **<マイドキュメント>\Altova\MapForce2021\MapForceExamples\Tutorial\Tut-headerDetail.mfd**。このサンプルは次の書式を持つ CSV ファイル(Orders.csv) を使用します:

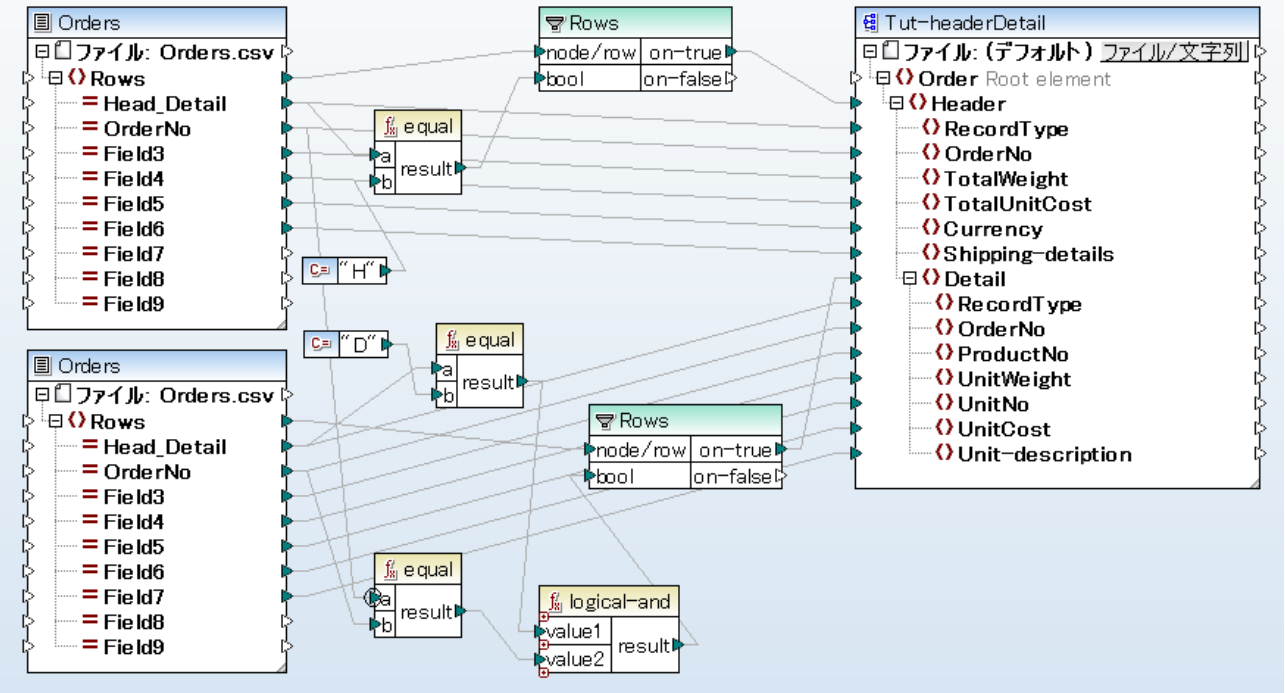
- フィールド 1 : H という値はヘッダーが記述されたレコードを表し、D は詳細情報が記述されたレコードを表します。
- フィールド 2 : ヘッダーと詳細レコードの共通キーです。
- 各ヘッダー/詳細レコードは別々の行にて表されます。

Orders.csv ファイルの内容を以下に示します。

```
H,111,332.1,22537.7,,Container ship,,,
D,111,A-1579-227,10,3,400,Microtome,,
D,111,B-152-427,7,6,1200,Miscellaneous,,
H,222,978.4,7563.1,,Air freight,,,
D,222,ZZ-AW56-1,10,5,10000,Gas Chromatograph,,
```

マッピングの目的を以下に示します:

- フラットファイルCSV を階層構造のXML ファイルへマッピングする
- H により表示されるヘッダーレコードをフィルタリングする
- 各ヘッダーレコードに対してD により表される詳細レコードを関連付ける



tut-headerDetail.mfd

上の目的を達成するには、ヘッダーと詳細レコードに共通のフィールドが与えられている必要があります。共通のフィールド(またはキー)とは、この場合 CSV ファイルのレコードである OrderNo のことになります。CSV ファイルの内容を見ると、最初のヘッダーレコードとそれ以降にある 2 つの詳細レコードに111 という共通した値が与えられていることが分かります。

Order.csv ファイルが2 度挿入され、マッピングがより直感的に表示されています。

Tut-headerDetail.xsd スキーマファイルは階層構造により構成されています。Order はルート要素で、子要素にHeader を持ち、更にHeader 以下にDetail 要素が配置されています。

最初のOrders.csv コンポーネントからは**Header** レコード(ならびにその他のフィールド)がスキーマターゲットファイルのHeader アイテムにマッピングされます。フィルターコンポーネントを使用することで、H レコードのフィルタリングを行います。行アイテムにより、フィルタリングされたレコードがスキーマファイル内にあるHeader アイテムに与えられます。

2 番目のOrders.csv コンポーネントからは、Header レコードにあるOrderNo の値にマッチした**Detail** レコードが与えられます。以下の処理が行われます:

- ヘッダーレコードの**OrderNo** フィールド を詳細レコードの同じフィールドと**equal** 関数を使用して比較します(優先コンテキスト は拡張されたパフォーマンスのためのパラメーター上で設定されています)。
- **Logical-and** 関数を使用することで、Header レコードのOrderNo と同じ値を含むDetail レコードだけが渡されるようになります。

行アイテムからはフィルターコンポーネントのon-true パラメーターを介して、フィルタリングされたレコードがスキーマコンポーネントのHeader ならびにDetail アイテムへ渡されます。

出力タブをクリックすることで、以下に示されるようなXML ファイルが表示されます。各ヘッダーレコードには対応するデータが、そしてDetail レコードには同じ OrderNo を持つ詳細情報が表示されます。2 番目の例では、別のCSV ファイルを使用します。

```
<?xml version="1.0" encoding="UTF-8"?>
<Order xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation
  <Header>
    <RecordType>H</RecordType>
    <OrderNo>111</OrderNo>
    <TotalWeight>332.1</TotalWeight>
    <TotalUnitCost>22537.7</TotalUnitCost>
    <Currency/>
    <Shipping-details>Container ship</Shipping-details>
    <Detail>
      <RecordType>D</RecordType>
      <OrderNo>111</OrderNo>
      <ProductNo>A-1579-227</ProductNo>
      <UnitWeight>10</UnitWeight>
      <UnitNo>3</UnitNo>
      <UnitCost>400</UnitCost>
      <Unit-description>Microtome</Unit-description>
    </Detail>
    <Detail>
      <RecordType>D</RecordType>
      <OrderNo>111</OrderNo>
      <ProductNo>B-152-427</ProductNo>
      <UnitWeight>7</UnitWeight>
      <UnitNo>6</UnitNo>
      <UnitCost>1200</UnitCost>
      <Unit-description>Miscellaneous</Unit-description>
    </Detail>
  </Header>
```

若干異なるCSV ファイルを使用し<**マイドキュメント**>¥Altova¥MapForce2021¥MapForceExamples¥Tutorial¥フォルダー内で**Head-detail-inline.mfd** として使用することのできる他のサンプルを見てみましょう。以下に特徴を示します:

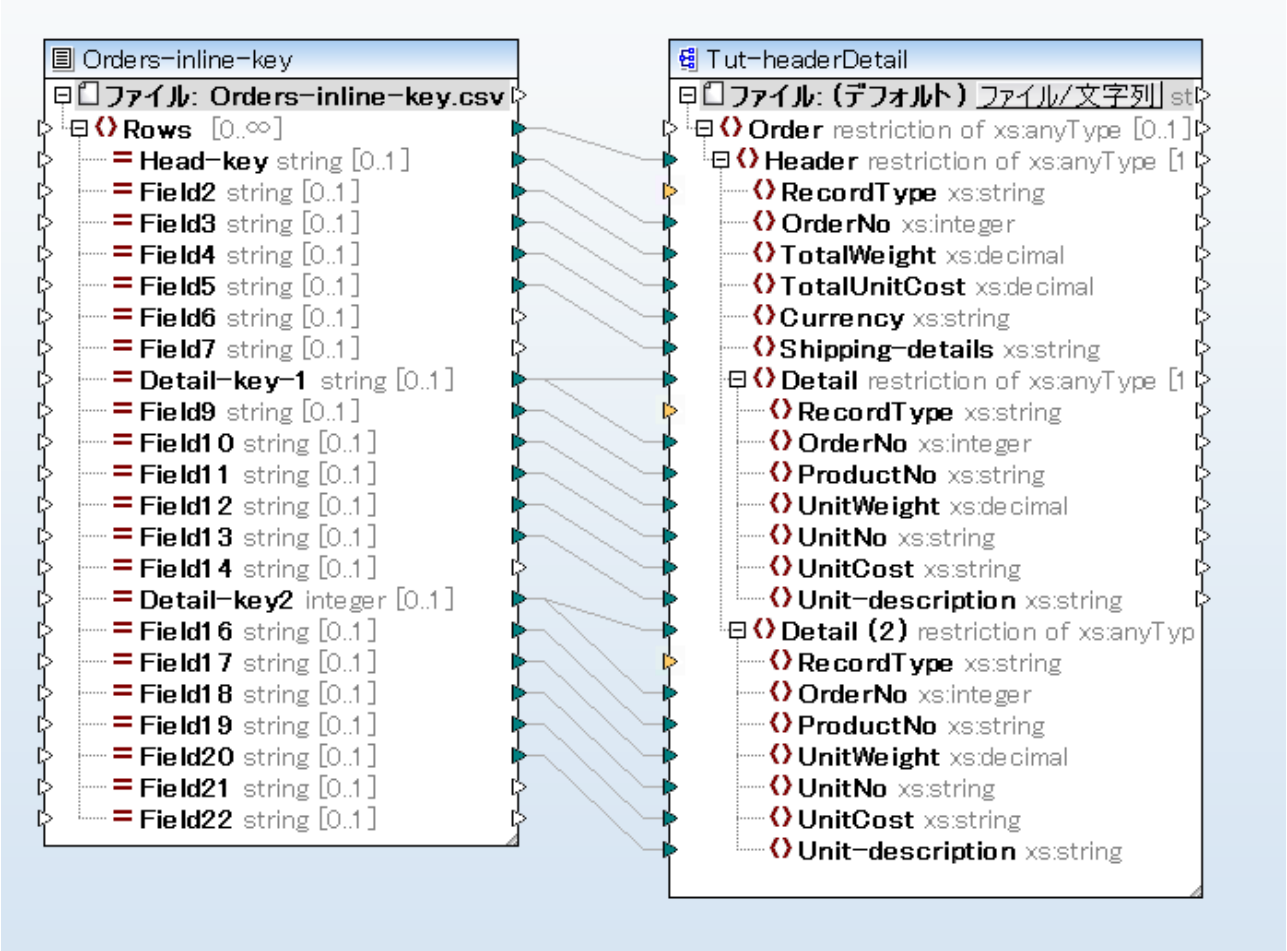
- レコードの識別子(H またはD)は利用できません。
- 共通/キーフィールドはCSV ファイル内の最初のフィールドに存在しており、ヘッダーならびに詳細レコード(Head-key、Detail-key)が収められています。フィールドはスキーマターゲットのOrderNo へマッピングされます。
- ヘッダーと関連する詳細フィールドは全て同じ行に収められています。

(C) 2015-2021 Altova GmbH

```
111,332.1,22537.7,,Container ship,,,111,A-1579-227,10,3,400,Microtome,,111,B-15
222,978.4,7563.1,,Air freight,,,222,ZZ-AW56-1,10,5,10000,Gas Chromatograph,,
```

マッピングは以下のようにデザインされます:

- キーフィールドがスキーマターゲット内の対応する OrderNo アイテムへマッピングされます。
- スキーマターゲットファイル内の詳細アイテムが複製され、以下のように表示されます: **Detail (2)**。これにより詳細レコードの2番目のセットを正確なアイテムにマップすることができます。
- このマッピングの結果として、上にあるサンプルと同様の XML ファイルが作成されます。



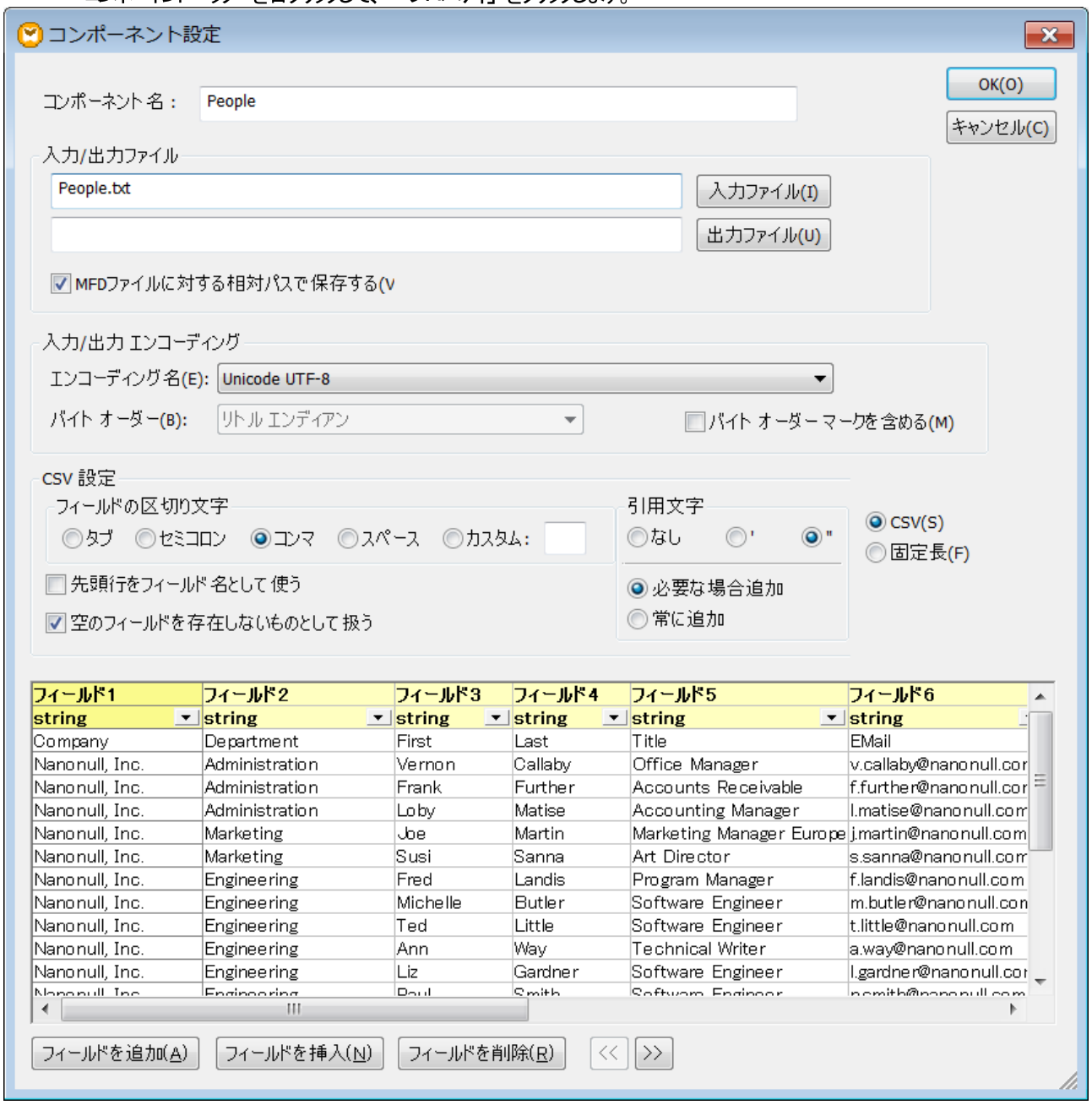
Head-detail-inline.mfd

6.3.4 CSV オプションの設定

テキストコンポーネントをマッピングエリアに追加した後、コンポーネント設定ダイアログボックス空適用できる設定を構成することができます。コンポーネント設定 ダイアログボックスを以下の方法で開くことができます:

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

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

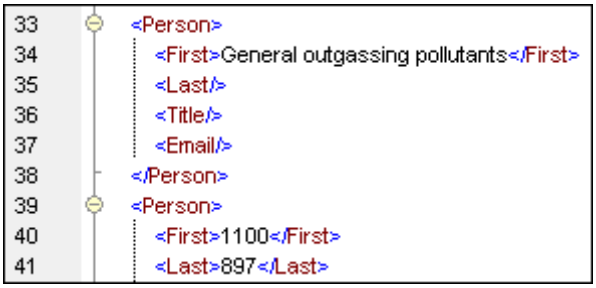
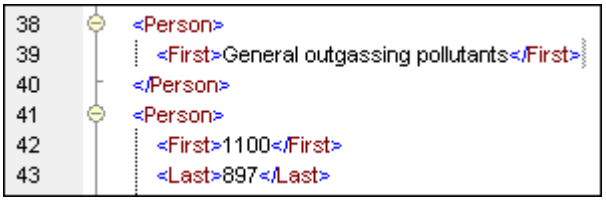
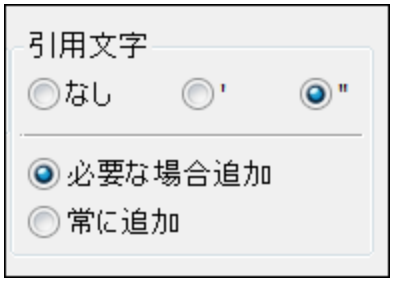


テキストコンポーネント設定ダイアログボックス（CSV モード）

使用することのできる設定は、以下のとおりです.

コンポーネント名	コンポーネント名はコンポーネントを作成する際に自動的に生成されます。ですが、後に名前を変更することが可能です。 コンポーネント名はスペースを含むことができ、またフルストップ文字も含むことができます。コンポーネント名は、スラッシュ、バックスラッシュ、コロン、二重引用符、行頭と末尾スペー

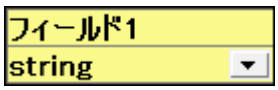
	スを含む場合があります。全般的に、コンポーネントの名前を変更する際、以下の影響を考慮してください: • マッピングを FlowForce Server にデプロイする場合、コンポーネント名は一意である必要があります。 • コマンドラインに入力できる文字のみを使用することが奨励されます。コマンドラインと W indows 内では、それぞれの文字で異なったエンコードを使用する場合があります。
入力ファイル	MapForce がデータを読み込むファイルを指定します。このフィールドはソースコンポーネントにとり重要で、最初のコンポーネントを作成すると入力され、テキストファイルに割り当てられます。マッピングのためにターゲットとしてテキストファイルが使用されている場合は、フィールドは空のままにしておくことができます。 ソースコンポーネント内で、MapForce は、このフィールドの値を使用し、列名を読み込み、インスタンステキストファイルのコンテンツをプレビューします。 新しいファイルを選択するには、「**入力ファイル**」をクリックします。
出力ファイル	MapForce がデータを書き込むファイルを指定します。このフィールドはターゲットコンポーネントにとり重要です。 新しいファイルを選択するには、「**出力ファイル**」をクリックします。
MFD ファイルに相対的なすべてのファイルパスを保存する	このオプションが有効化されると、MapForce は、コンポーネント設定ダイアログボックスに表示された、MapForce Design (.mfd) ファイルの場所に相対したファイルパスを保存します。この設定は、テキストコンポーネントにより使用された入力と出力ファイルを影響します。以下も参照: コンポーネント上で相対パスを使用する。
入力/出力 エンコード	出力インスタンスファイルの次の設定を指定することを許可します: • エンコード名 • バイトオーダー • バイトオーダーマーク (BOM) 文字が含まれるか否か。 デフォルトでは、「**新しいコンポーネントのためのデフォルトのエンコード**」オプション内で定義されたエンコードを持つ新しいコンポーネント。「**ツール	オプション**」からこのオプションにアクセスし、タブを生成します。
フィールド区切り記号	テキストファイルの区切りを選択します (CSV ファイルの場合、デフォルトの区切りである "," が使用されます)。このオプションにより、タブ、セミコロン、スペース文字を区切り記号として選択できます。独自の区切り文字をカスタムフィールドにて入力することもできます。
最初の行をフィールドの名前と見なす	テキストファイル内にある最初のレコードの値をカラムのヘッダーとして定義します (プレビューウィンドウに表示されます)。テキストコンポーネントがマッピングに表示された際には、カラムヘッダーがアイテム名として表示されます。
空のフィールドを存在しないものとして扱う	このオプションが有効化されると、ソースファイル内の空のフィールドは対応する空のアイテム (要素または属性) をターゲットファイル内に作成しません。 例: CSV レコード "`General outgassing pollutants,,,,`" は、4 つのフィールドから構成されており、少なくとも 3 つのフィールドは空です。

	出力がXML ファイルと仮定し、このオプションが無効化されている場合、空の値を持つ空のフィールドが出力に作成されます（このサンプルでは、要素 `Last`、`Title`、と `Email`）:  このオプションが有効化されると、空のフィールドは出力に作成されません 
囲み文字	入力ファイル内のフィールド値が囲み文字で囲まれている場合、ソースファイルと同じ囲み文字を選択してください。同じ設定が出力ファイルにも使用されます。  出力ファイルのために、追加設定を指定することができます: **必要な場合追加する** フィールド 区切り記号、または改行を含むテキストのフィールドに選択された囲み文字を追加します。 **常に追加する** 生成されたCSV ファイルのすべてのフィールドに選択された囲み文字を追加します。
CSV／固定値	コンポーネントの型をCSV またはFLF（固定値フィールド）に変更します。
プレビューエリア	ダイアログボックスの下の部分では、入力または出力として選択されたファイルの20 の行を表示します。 必要であれば、以下のように、ファイルの構造を作成（または既存の構造を変更）することができます。

(C) 2015-2021 Altova GmbH

フィールドの追加	最後のCSV レコードの後に、新規のフィールドを作成しま
フィールドの挿入	現在選択されているCSV レコードの前に、新規のフィー すぐに作成します。
フィールドの削除	現在選択されているフィールドを削除します。
<<	現在選択されているフィールド左に1つ移動します。
>>	現在選択されているフィールド右に1つ移動します。

フィールドの名前を変更するには、ヘッダーをクリックして(例: **フィールド1**)、新しい値を入力します。**最初の行をフィールドの名前とみなす** オプション が有効化されている場合、フィールド名は編集することができません。



ドロップダウンリストから必要とされる値を選択し、フィールドのデータ型を変更します。MapForce は、入力データとフィールドフォーマットが一致するかデータ型をチェックします。フィールド フォーマットが一致しない場合、はデータが赤では以来とされます。

フィールド型はデフォルトのXML スキーマデータ型をベースにしています。例: データ型は以下のフォームで表示されます: **YYYY-MM-DD**。

6.3.5 例: 固定長テキストファイルを(データベースへ)マッピング

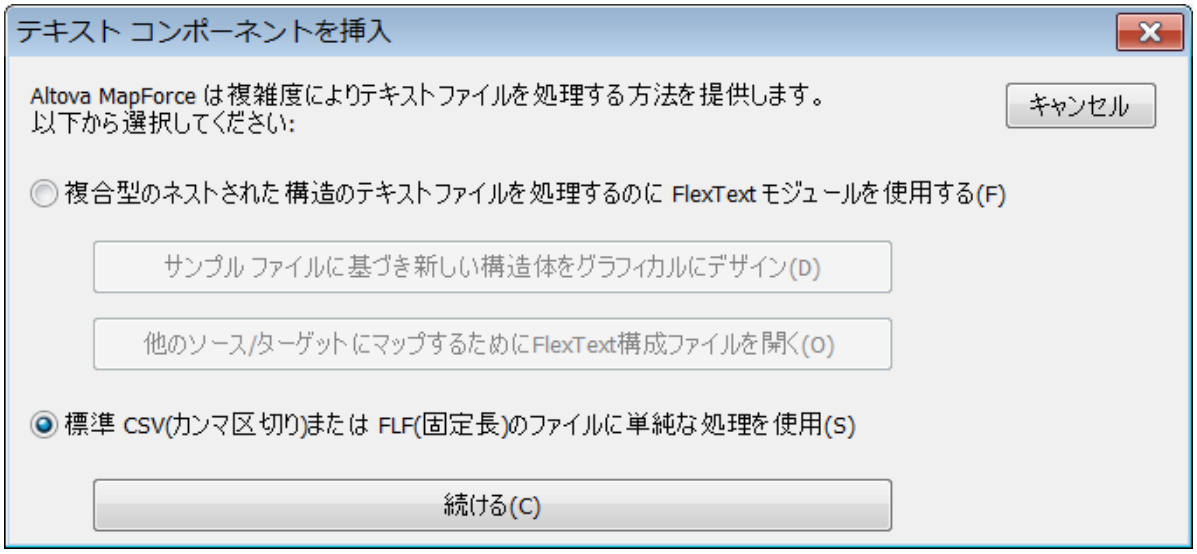
このサンプルでは固定長テキストファイル(FLF) とMS Access データベースの間のマッピングを行います。このサンプルで使用されるファイルは、<**マイドキュメント**>¥Altova¥MapForce2021¥MapForceExamples¥Tutorial¥ フォルダーに収められています。ソーステキストファイルとターゲット データベースは、従業員のリストを保管しています。ソースファイル内では、記録は明示的にサイズにより以下のようにカンマで区切られています:

フィールドの位置と名前	サイズ(文字数)
Field 1 (First name)	8
Field 2 (Last name)	10
Field 3 (Phone extension)	3
Field 4 (Email)	25
Field 5 (Position)	25

マッピングの目的は、データベース内の各社員の内線番号を、各内線番号のプレフィックスに"100" を追加し、ソースファイル内のものに更新することです。このゴールを達成するには、以下を行います:

ステップ 1: テキストコンポーネントの挿入と構成

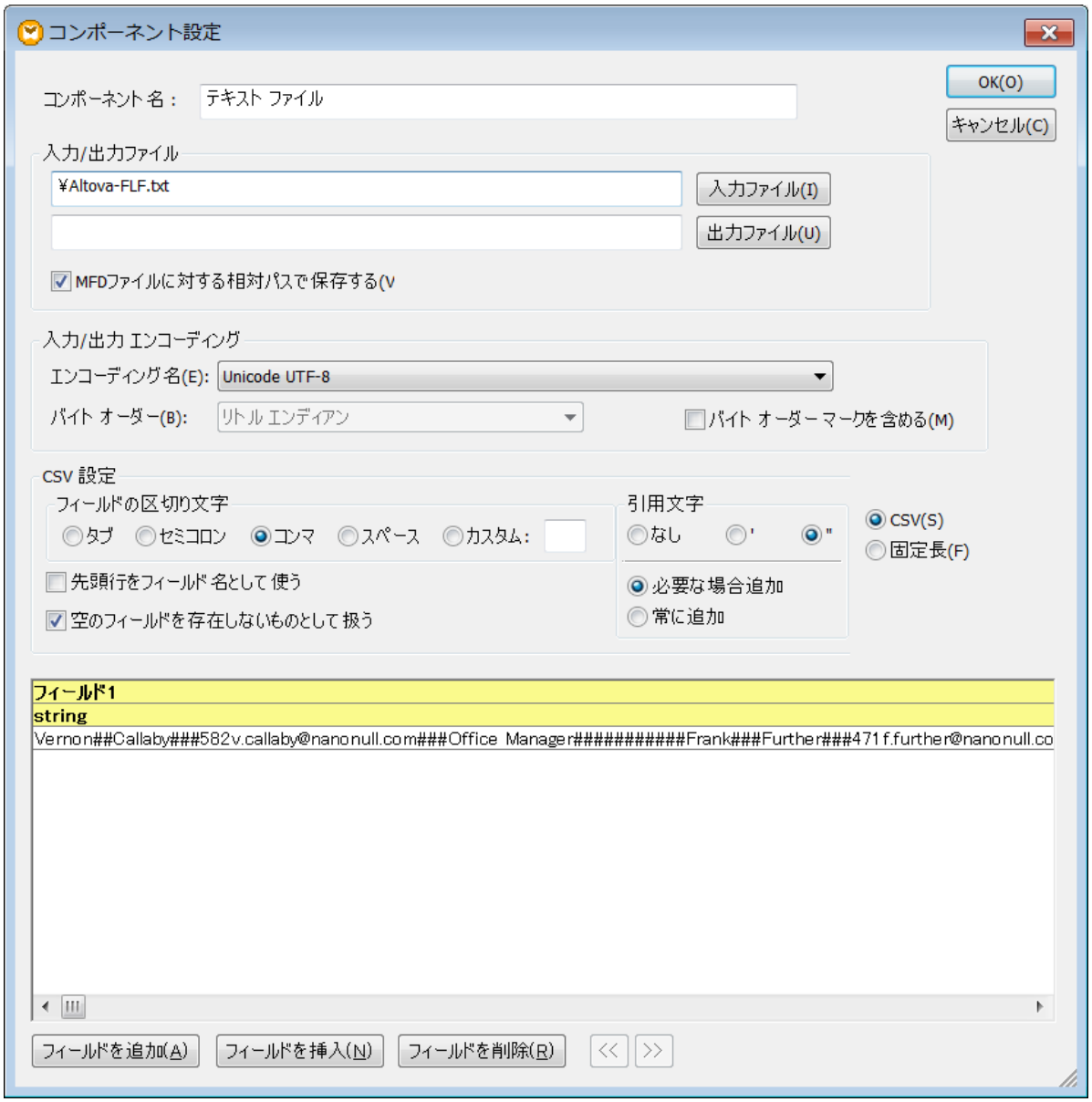
1. メニューオプションの「挿入 | テキストファイル」を選択するか、テキストファイルを挿入アイコン をクリックします。テキストコンポーネントを挿入ダイアログボックスが表示されます。



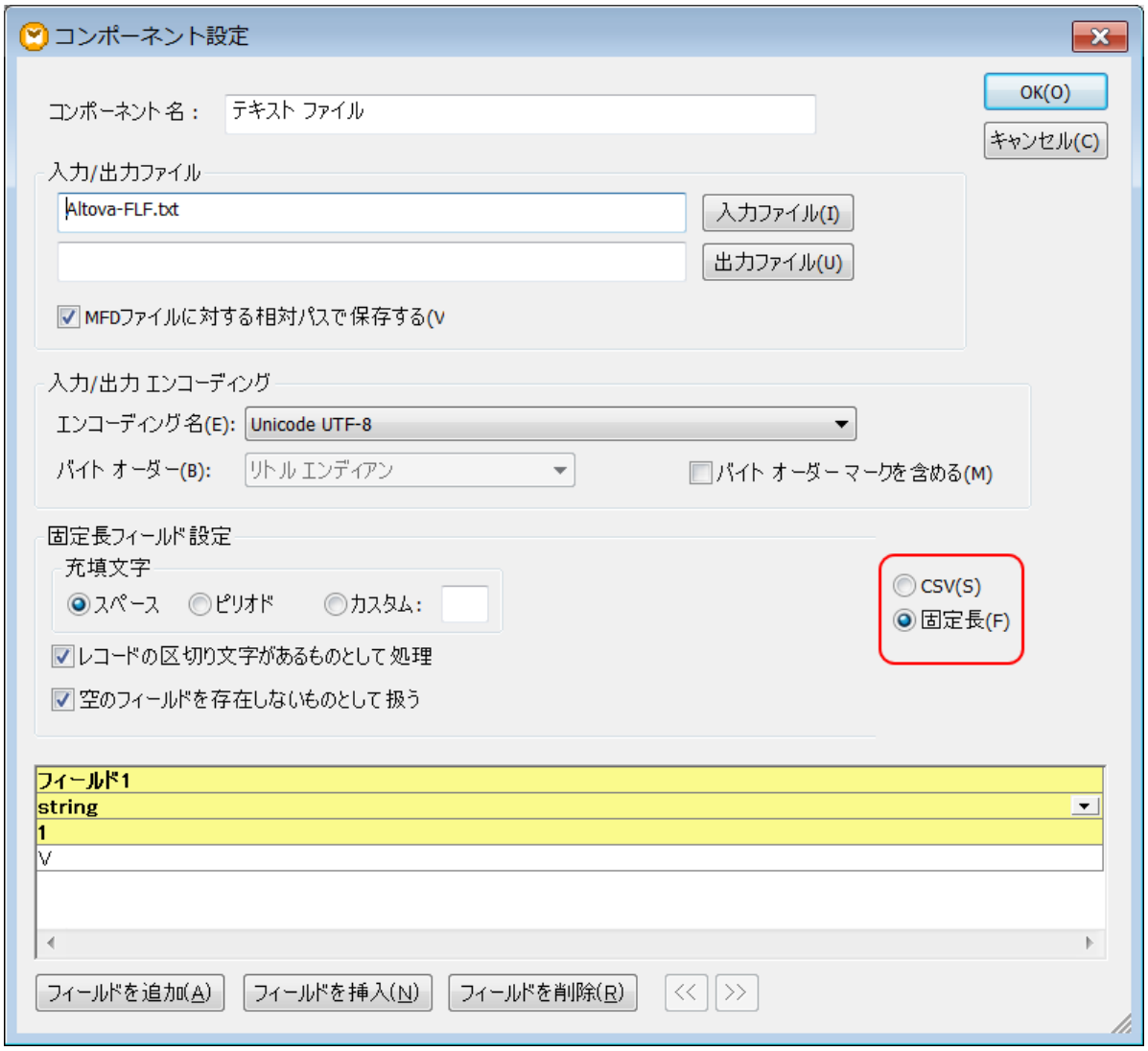
単純な処理を使用... を選択して、「続ける」ボタンをクリックします。

2. 入力ファイルボタンをクリックして **マイドキュメント >\Altova\MapForce2021\MapForceExamples\Tutorial\Altova-FLF.txt** ファイルを選択します。ファイルが1 つの(大きな) 文字列と、# の充填文字から構成されていることが確認できます。

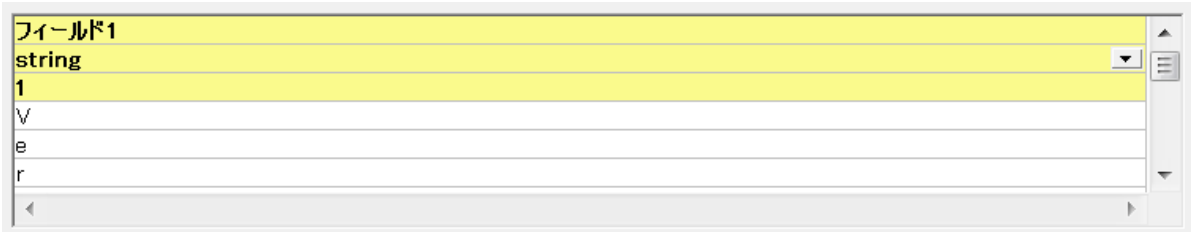
(C) 2015-2021 Altova GmbH


コンポーネント設定
コンポーネント名：
テキスト ファイル
OK(O)
キャンセル(C)
入力/出力ファイル
¥Altova-FLF.txt
入力ファイル(I)
出力ファイル(U)
MFDファイルに対する相対パスで保存する(V
入力/出力 エンコーディング
エンコーディング名(E):
Unicode UTF-8
バイト オーダー(B):
リトル エンディアン
バイト オーダー マークを含める(M)
CSV 設定
フィールドの区切り文字
タブ
セミコロン
コンマ
スペース
カスタム:
引用文字
なし
'
"
CSV(S)
固定長(F)
先頭行をフィールド名として使う
空のフィールドを存在しないものとして扱う
必要な場合追加
常に追加
フィールド1
string
Vernon##Callaby###582v.callaby@nanonull.com###Office Manager##########Frank###Further###471f.further@nanonull.co
フィールドを追加(A)
フィールドを挿入(N)
フィールドを削除(R)

3. **固定長**を選択します。

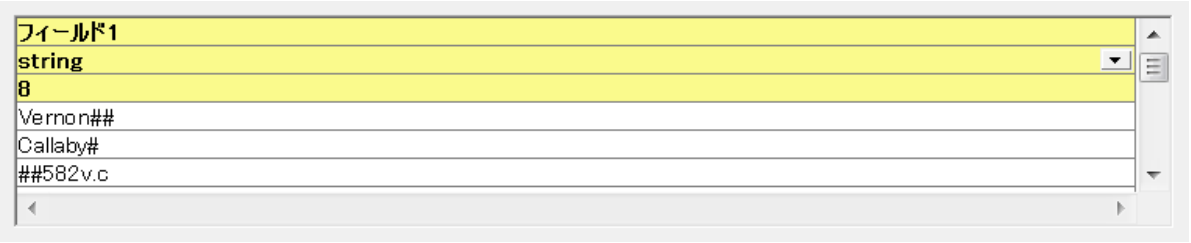


4. **レコードの区切り文字があるものとして処理**チェックボックスをクリアします。

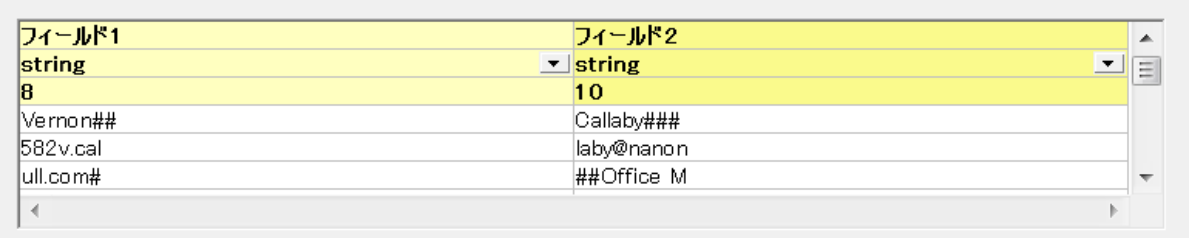


5. 黄色でハイライトされた3行は編集することができます。また、以下を指定することができます: i) フィールド名、ii) データ型、iii) フィールドのサイズ。値を 8 に変更した後 Enter キーを押下します。8文字ごとにフィールドが区切られるため、より多くのデータが最初のレコードに表示されます。

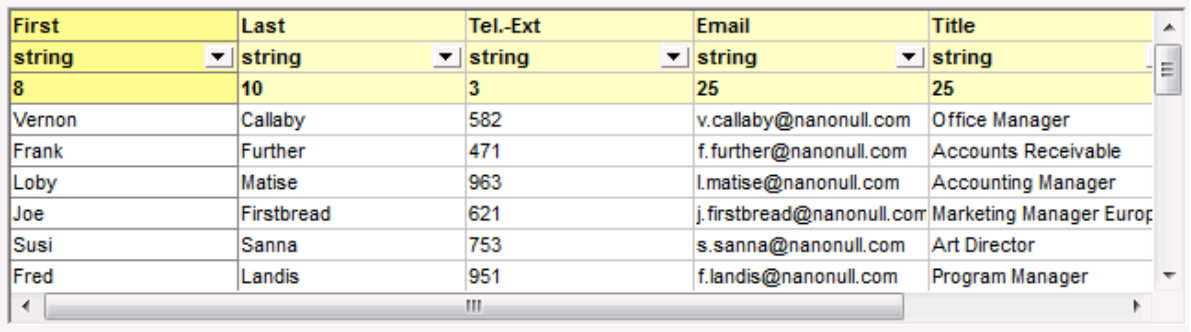
(C) 2015-2021 Altova GmbH



6. フィールドを追加ボタンをクリックして、新たなフィールドを追加します。2番目のフィールド長さを10文字に指定します。

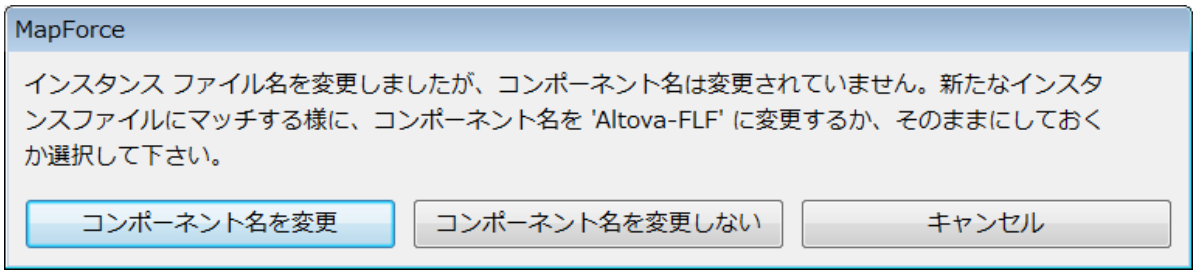


7. 同様の方法を使用して、更に3つのフィールドを作成します。長さはそれぞれ3、25、25 文字になります。更にフィールドの名前をそれぞれFirst、Last、Tel.-Ext、Email、Title とすることで、簡単にマッピングを行えるようにします。プレビューは以下のように表示されます:

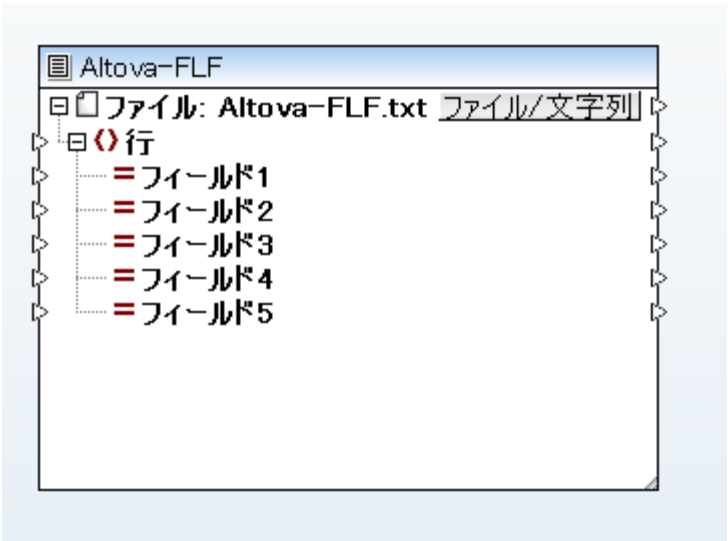


8. 固定長フィールド設定グループのカスタムテキストボックスをクリックして、ハッシュ文字 (#) を入力します。この操作により、充填文字として使用されている文字が入力文字列から取り除かれます。

9. 「OK」をクリックします。

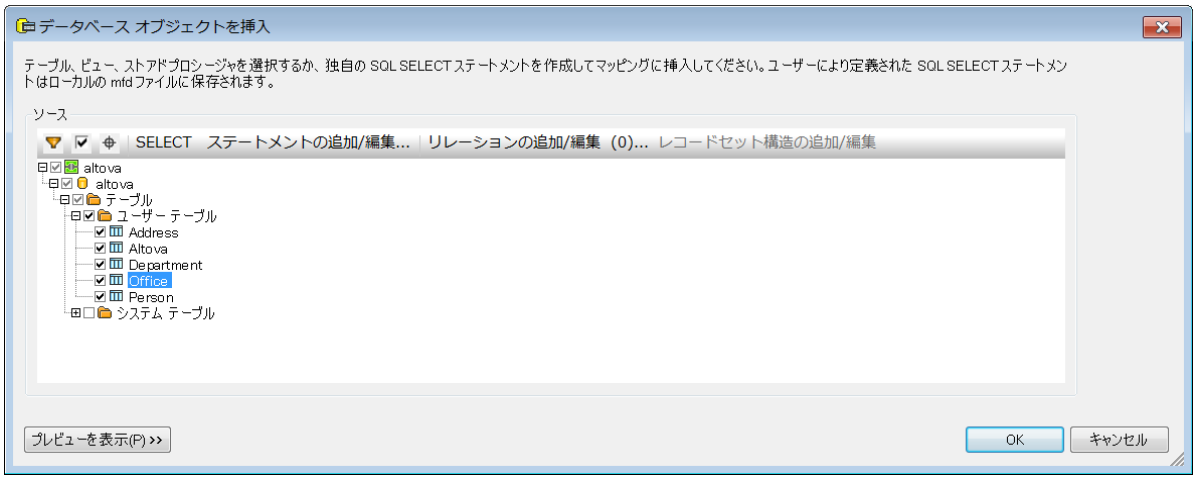


10. 「コンポーネント名を変更」ボタンをクリックして、名前を変更します。テキストファイルコンポーネントがマッピングウィンドウに表示されます。このコンポーネントを使ったデータのマッピングが行えるようになります。



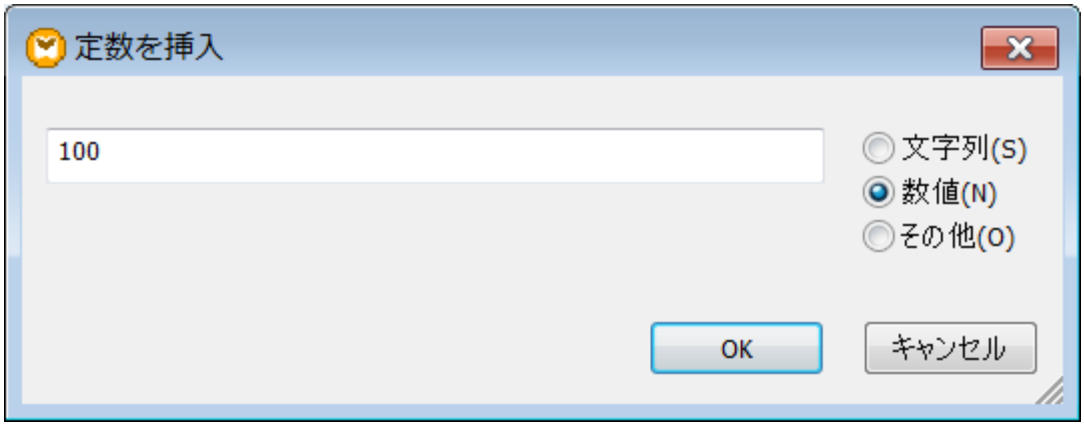
ステップ 2: データベースコンポーネントの挿入

1. メニューオプションの「挿入 | データベース」を選択し、**Microsoft Access** ラジオボタンを選択して「次へ」をクリックします。
2. <マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥Tutorial¥ フォルダー内にある **altova.mdb** データベースを選択し、「接続」をクリックします。
3. Person テーブルのチェックボックスを選択し、「OK」をクリックします。

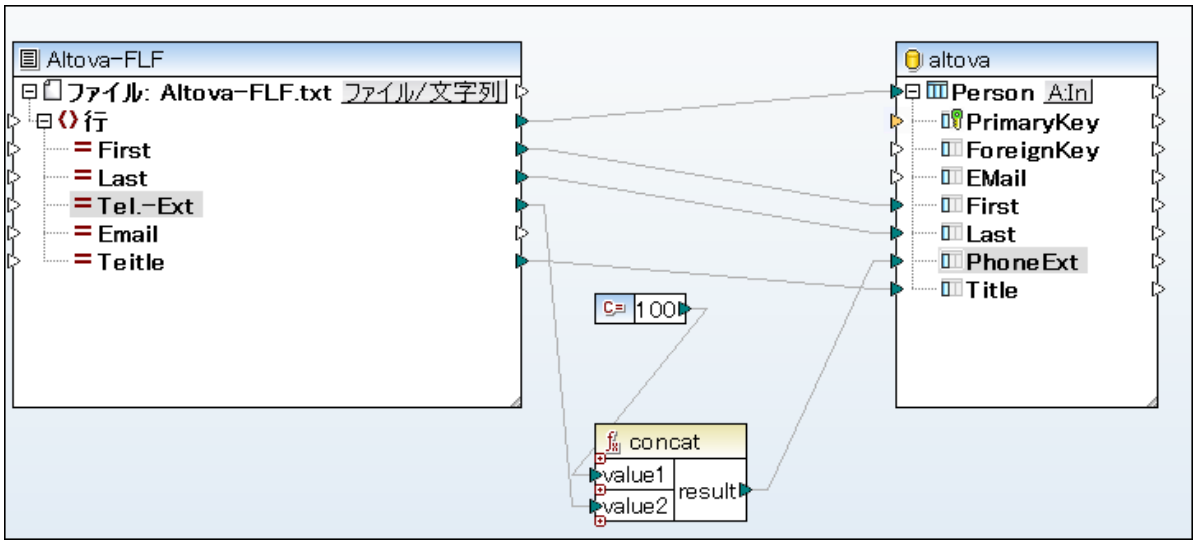


ステップ 3: マッピングのデザイン

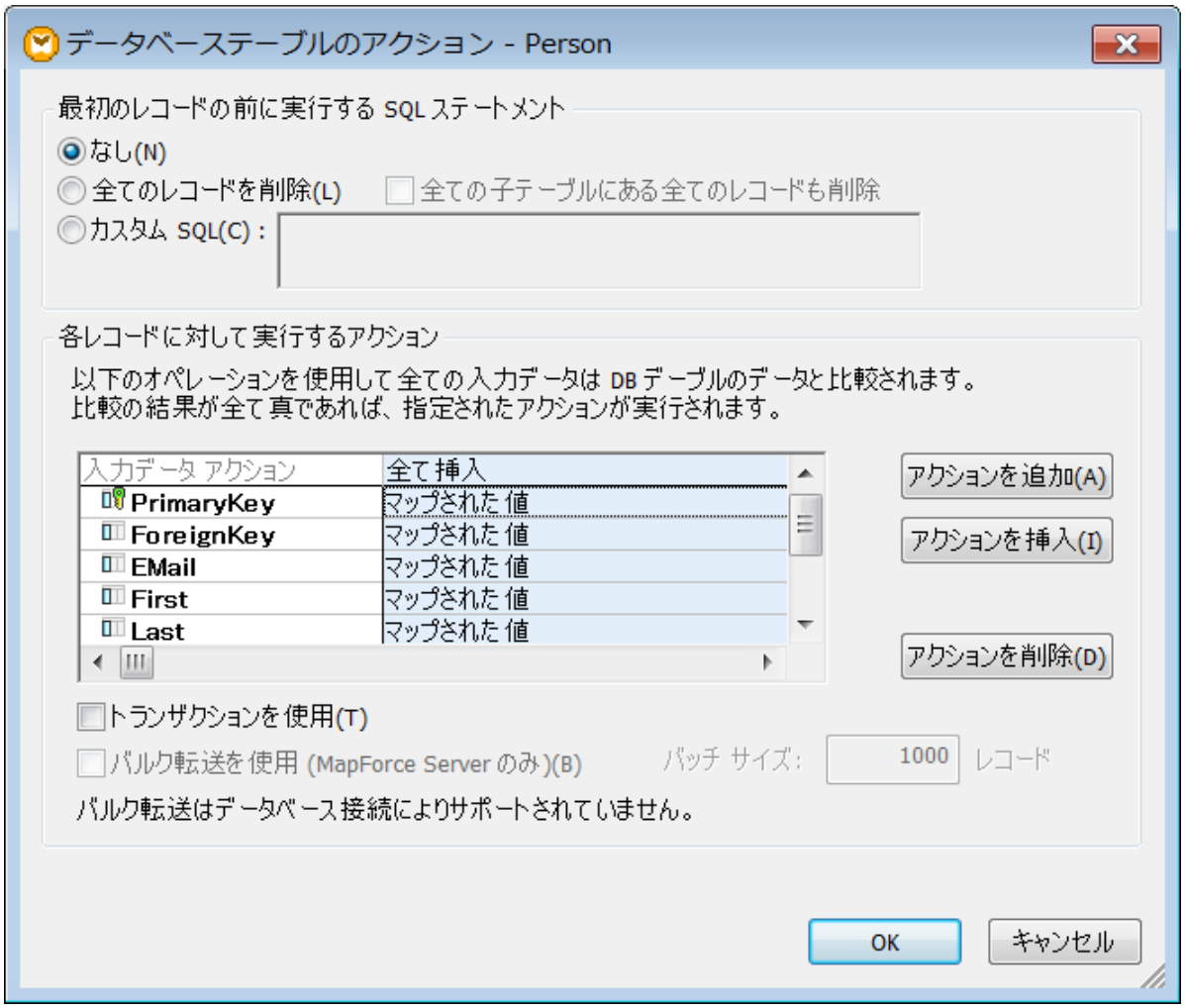
1. core | concat 関数をライブラリウィンドウからマッピングタブへドラッグします。
2. メニューオプションの「挿入 | 定数」を選択し、数値ラジオボタンをクリックした後に、新たな内線番号のプレフィックスとして使用する 100 を入力します。この定数が内線番号のプレフィックスを保管します。



3. 以下のスクリーンショットに示されるマッピングを作成します。



4. **Person** エントリーの隣にあるテーブルアクションボタン A:In をクリックします。
5. 「**入力データアクション**」のコンボボックスをクリックし、「**以下の条件で更新..**」 エントリーを選択し、**First** と **Last** のフィールドが一致するように設定します。この操作により、**First** の内容がソーステキストファイルとデータベースにて一致する場合のみ、Person テーブルが更新されるようになります。条件が真の場合に取られるアクションはマッピングにより定義されており、この場合は "100" というプレフィックスが追加された内線番号が **Person** テーブルの **PhoneExt** フィールドに与えられます。



ステップ 4: マッピングの実行

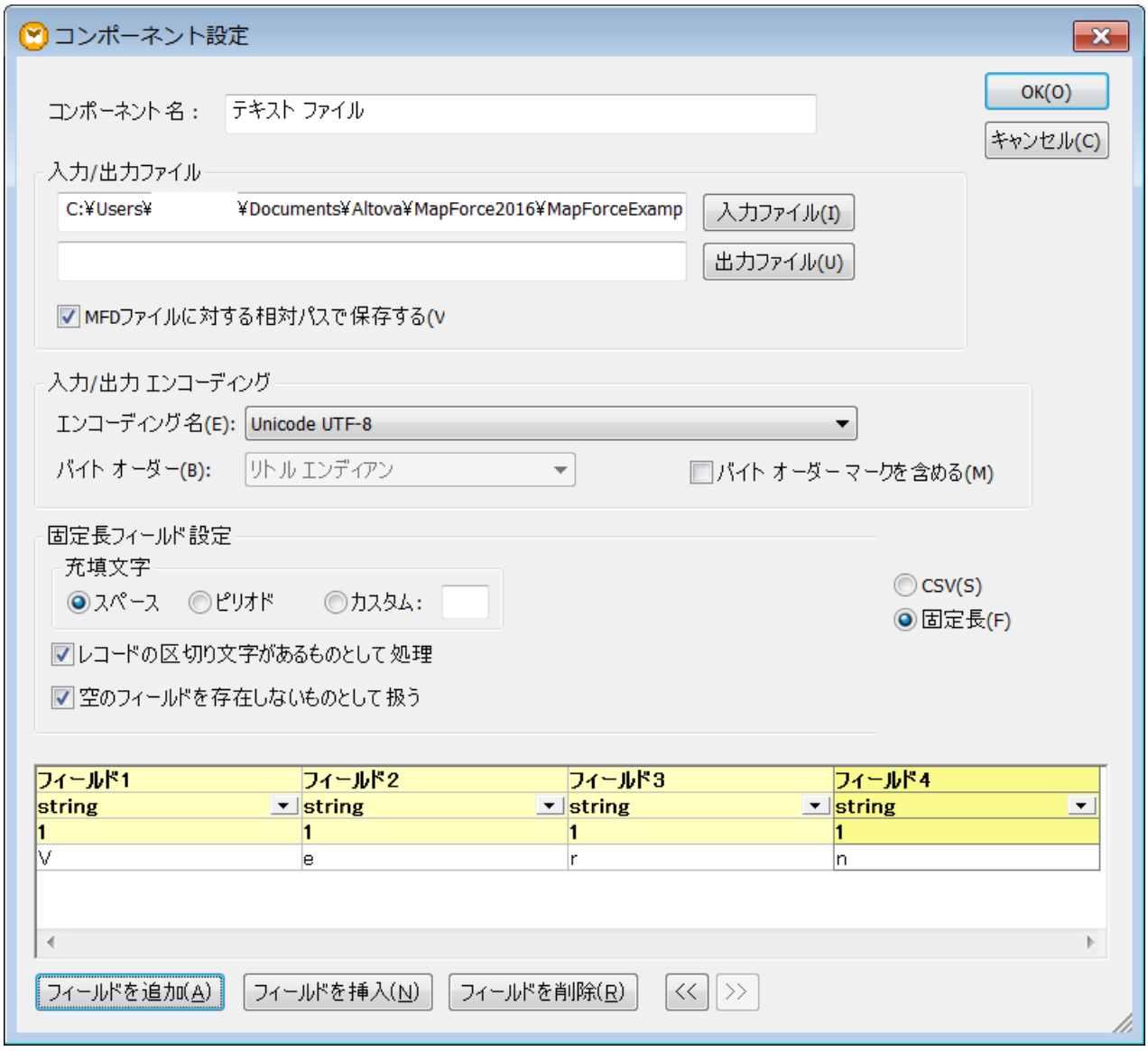
出力タブをクリックして、生成されたSQL ステートメントをプレビューします。SQL スクリプトの実行ボタン をクリックして、SQL ステートメントを実行します。

6.3.6 FLF オプションの設定

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

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

(C) 2015-2021 Altova GmbH

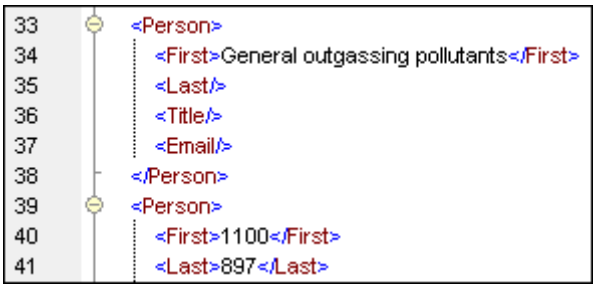


テキストコンポーネント設定ダイアログボックス（固定値フィールドモード）

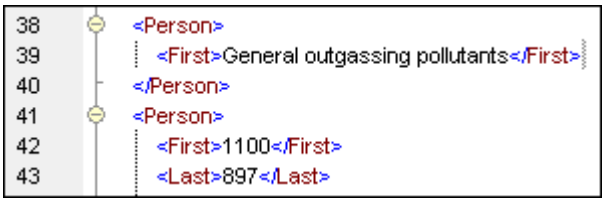
使用することのできる設定は、以下のとおりです.

コンポーネント名	コンポーネント名はコンポーネントを作成する際に自動的に生成されます。ですが、後に名前を変更することが可能です。 コンポーネント名はスペースを含むことができ、またフルストップ文字も含むことができます。コンポーネント名は、スラッシュ、バックスラッシュ、コロン、二重引用符、行頭と末尾スペースを含む場合があります。全般的に、コンポーネントの名前を変更する際、以下の影響を考慮してください： • マッピングをFlowForce Server にデプロイする場合、コンポーネント名は一意である必要があります。 • コマンドラインに入力できる文字のみを使用することが奨励されます。コマンドラインとW indows 内では、それぞれの文字で異なったエンコードを使用する場

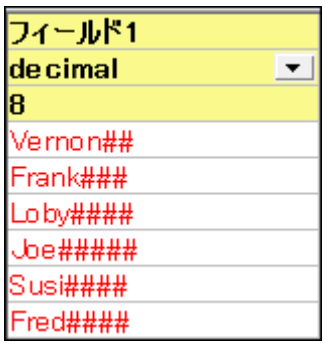
	合があります。
入力ファイル	MapForce がデータを読み込むファイルを指定します。このフィールドはソースコンポーネントにとり重要で、最初のコンポーネントを作成すると入力され、またテキストファイルに割り当てられます。テキストファイルコンポーネントをマッピングのためにターゲットとして、使用する場合、フィールドは空にしてください。 ソースコンポーネント内で、MapForce はこのフィールドの値を使用して、列名を読み込み、インスタンステキストファイルのコンテンツをプレビューします。 新しいファイルを選択するには、「**入力ファイル**」をクリックします。
出力ファイル	MapForce がデータを書き込むファイルを指定します。このフィールドはターゲットコンポーネントにとり重要です。 新しいファイルを選択するには、「**出力ファイル**」をクリックします。
MFD ファイルに相対的なすべてのファイルパスを保存する	このオプションが有効化されると、MapForce は、コンポーネント設定ダイアログボックスに表示された、MapForce Design (.mfd) ファイルの場所に相対したファイルパスを保存します。この設定は、テキストコンポーネントにより使用された入力と出力ファイルに影響を与えます。以下も参照: コンポーネント上で相対パスを使用する。
入力／出力 エンコード	出力インスタンスファイルの次の設定を指定することを許可します: • エンコード名 • バイトオーダー • バイトオーダーマーク（BOM）文字が含まれるか否か。 デフォルトでは、「**新しいコンポーネントのためのデフォルトのエンコード**」オプション内で定義されたエンコードを持つ新しいコンポーネント。「**ツール	オプション**」からこのオプションにアクセスし、タブを生成します。
充填文字	データの長さがフィールドで定義されたものよりも短い場合、このオプションで指定された文字により残りのスペースが充填されます。カスタムフィールドにより独自の充填文字を定義することができます。 カスタムフィールドに入力された充填文字と同じ文字が入力データに含まれている場合、入力データにある文字も充填文字として切り取られます！
レコードの区切り文字があるものとして処理	このオプションは、ソースフラットファイルから CR/LF などの区切り記号 を含まないデータを読み込むために役に立ちます。または、区切り文字を含まないターゲットフラット FLF ファイルを作成する際にも役に立ちます。 下の レコードの区切り文字があるものとして処理するオプション セクションを参照してください。
空のフィールドは存在しないものとして扱う	このオプションが有効化されると、ソースファイル内の空のフィールドは対応する空のアイテム（要素または属性）をターゲットファイル内に作成しません。 出力が XML ファイルと仮定し、このオプションが無効化されている場合、空の値を持つ空のフィールドが出力に作成されます（このサンプルでは、要素 `Last`、`Title`、と `Email`）:



このオプションが有効化されると、空のフィールドは出力に作成されません

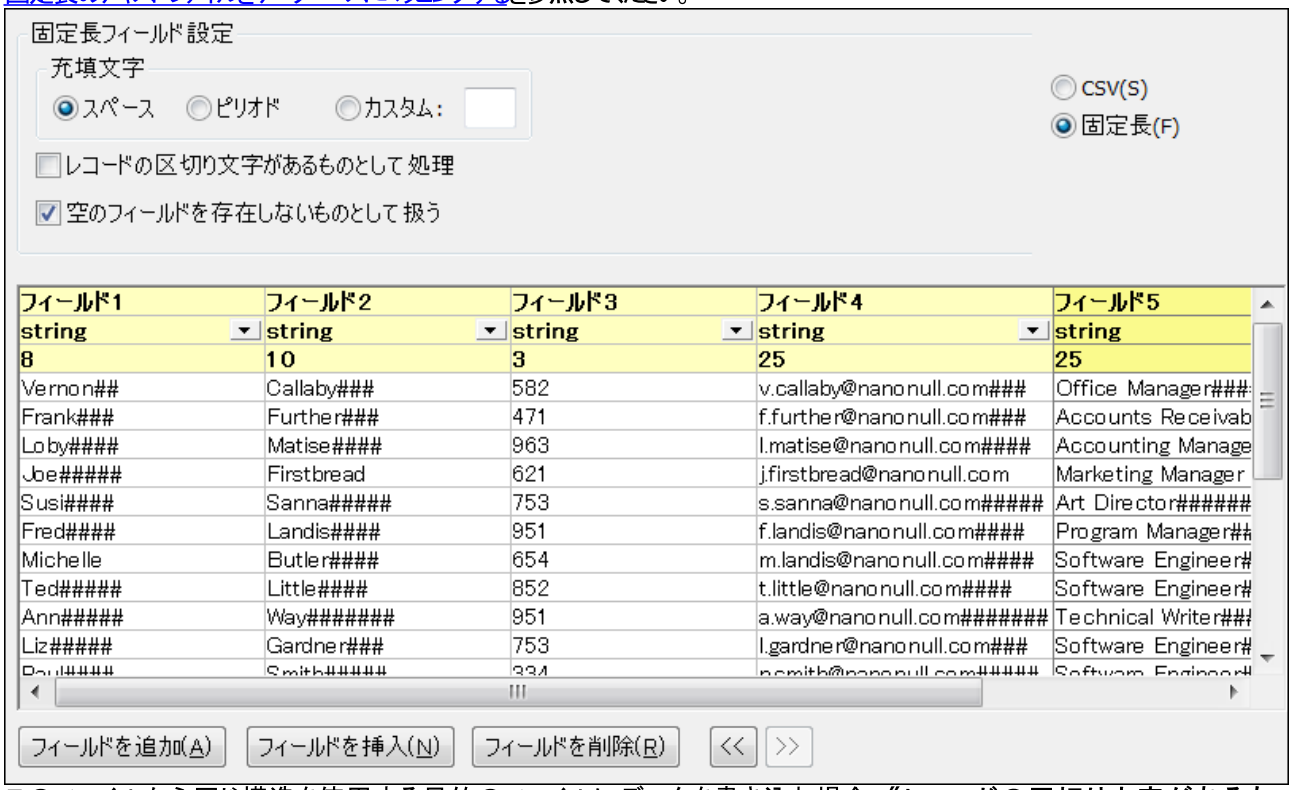


CSV／固定	コンポーネントの型をCSV またはFLF（固定値フィールド）に変更します。
プレビューエリア	ダイアログボックスの下の部分では、入力または出力として選択されたファイルの20 の行を表示します。 必要であれば、以下のように、ファイルの構造を作成（または既存の構造を変更）することができます。 **フィールドの追加** 最後のレコードの後に新規のフィールドを作成します。 **フィールドの挿入** 現在選択されている記録の前に、新規のフィールドをすぐに します。 **フィールドの削除** 現在選択されているフィールドを削除します。 **<<** 現在選択されているフィールド左につ 移動します。 **>>** 現在選択されているフィールド右につ 移動します。 フィールドの名前を変更するには、ヘッダーをクリックして（このサンプルでは、**フィールド1**）、新しい値を入力します。 **フィールド1** **string** **8** Vernon## Frank### Loby#### Joe##### Susi#### Fred####

	フィールドのデータ型を変更するには、ドロップダウンリストから必要な値を選択してください。MapForce は、入力データとフィールドフォーマットが一致しない場合赤でハイライトできるように、データ型をチェックします。  フィールド内の文字のサイズを設定できるように、上から3番目にフィールドのサイズを入力します。

"レコードの区切り文字があるものとして処理" オプションを理解する

このオプションを理解するためには、<マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥Tutorial¥フォルダー内の **Altova-FLF.txt** ファイルを開きます。ファイルは、CR/LF などの区切り文字を使用しない、71 文字の長さのレコードから構成されています。このファイルからデータを読み込む場合、最初にこのファイルをレコードに分割する必要があります。すなわち、下に表示されるようにフィールドが 1 文字である複数のフィールドを作成し、**レコードの区切り文字があるものとして処理** を無効化します。順序を追った例は、 例: 固定長のテキストファイルをデータベースにマッピングするを参照してください。



このファイルから同じ構造を使用する目的のファイルにデータを書き込む場合、"**レコードの区切り文字があるものとして処理**" を有効化すると、新規のレコードが72文字目から新たに表示されます。

```
1   Vernon##Callaby###582v.callaby@nanonull.com###Office Manager##########
2   Frank###Further###471f.further@nanonull.com###Accounts Receivable######
3   Loby####Matise####963l.matise@nanonull.com####Accounting Manager#######
4   Joe#####Firstbread621j.firstbread@nanonull.comMarketing Manager Europe#
5   Susi####Sanna#####753s.sanna@nanonull.com#####Art Director#############
6   Fred####Landis####951f.landis@nanonull.com####Program Manager##########
7   MichelleButler####654m.landis@nanonull.com####Software Engineer########
8   Ted#####Little####852t.little@nanonull.com####Software Engineer########
9   Ann#####Way#######951a.way@nanonull.com#######Technical Writer#########
10  Liz#####Gardner###753l.gardner@nanonull.com###Software Engineer########
11  Paul####Smith#####334p.smith@nanonull.com#####Software Engineer########
12  Alex####Martin####778a.martin@nanonull.com####IT Manager###############
13  George##Hammer####223g.hammer@nanonull.com####Web Developer############
14  Jessica#Bander####241j.band@nanonull.com######Support Engineer#########
15  Lui#####King######345l.king@nanonull.com######Support Engineer#########
16  Steve###Meier#####114s.meier@nanonull.com#####Office Manager###########
17  Theo####Bone######331t.bone@nanonull.com######Accounts Receivable######
18  Max#####Nafta#####122m.nafta@nanonull.com#####PR & Marketing Manager US
19  ValentinBass######716v.bass@nanonull.com######IT Manager###############
20  Carl####Franken###147c.franken@nanonull.com###Support Engineer#########
21  Mark####Redgreen##152m.redgreen@nanonull.com##Support Engineer#########
22
```

"レコードの区切り文字があるものとして処理 "が有効化された場合のマッピングの結果

"レコードの区切り文字があるものとして処理 " が無効化されている場合、マッピングされたデータが、充填文字とともに、1 つの長い文字列として表示されます。

```
1   Vernon##Callaby###582v.callaby@nanonull.com###Office
    Manager##########Frank###Further###471f.further@nanonull.com###Accounts
    Receivable######Loby####Matise####963l.matise@nanonull.com####Accounting
    Manager#######Joe#####Firstbread621j.firstbread@nanonull.comMarketing Manager
    Europe#Susi####Sanna#####753s.sanna@nanonull.com#####Art
    Director#############Fred####Landis####951f.landis@nanonull.com####Program
    Manager##########MichelleButler####654m.landis@nanonull.com####Software
    Engineer########Ted#####Little####852t.little@nanonull.com####Software
    Engineer########Ann#####Way#######951a.way@nanonull.com#######Technical
    Writer#########Liz#####Gardner###753l.gardner@nanonull.com###Software
    Engineer########Paul####Smith#####334p.smith@nanonull.com#####Software
    Engineer########Alex####Martin####778a.martin@nanonull.com####IT
    Manager###############George##Hammer####223g.hammer@nanonull.com####Web
    Developer############Jessica#Bander####241j.band@nanonull.com######Support
    Engineer#########Lui#####King######345l.king@nanonull.com######Support
    Engineer#########Steve###Meier#####114s.meier@nanonull.com#####Office
    Manager###########Theo####Bone######331t.bone@nanonull.com######Accounts
    Receivable######Max#####Nafta#####122m.nafta@nanonull.com#####PR & Marketing Manager
    USValentinBass######716v.bass@nanonull.com######IT
    Manager###############Carl####Franken###147c.franken@nanonull.com###Support
    Engineer#########Mark####Redgreen##152m.redgreen@nanonull.com##Support
    Engineer#########
```

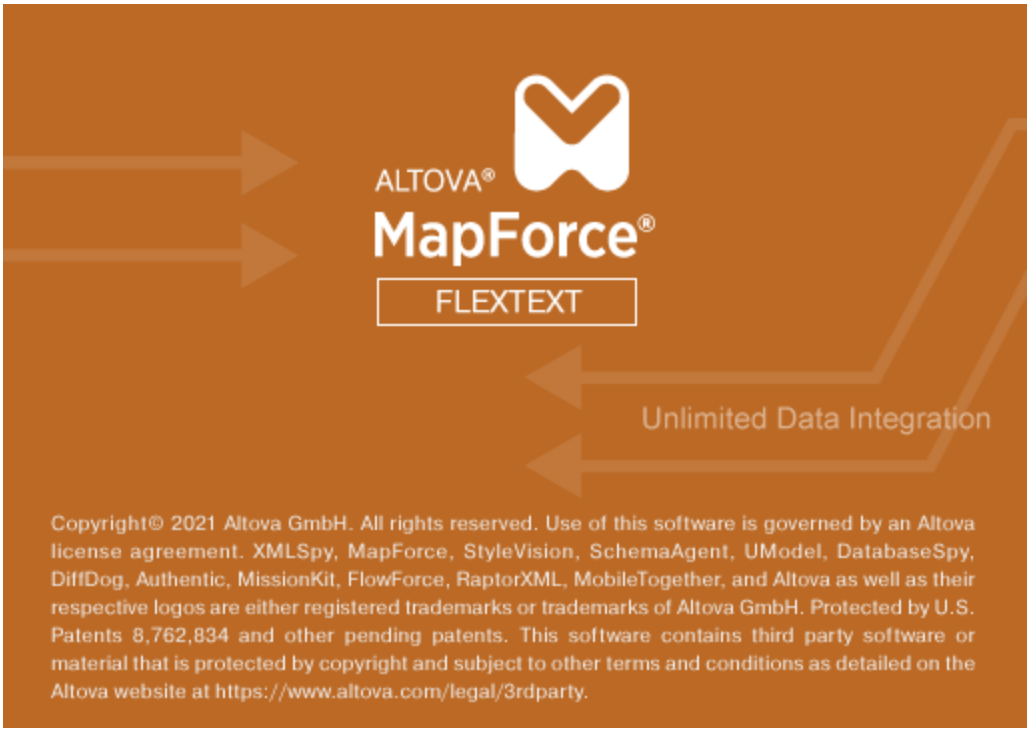
"レコードの区切り文字があるものとして処理 "が無効化された場合のマッピング結果

6.4 MapForce FlexText

Altova W eb サイト: テキストとフラットファイルマッピング

FlexText は、複雑な非標準、または、レガシーテキストファイルからデータを MapForce によりサポートされる他のフォーマットに変換する MapForce モジュールです。XML ファイルには、 MapForce がスキーマから抽出することのできる スキーマ、または、構造が存在しますが、テキストファイルは異なります。特に、CSV または 固定-長さ フィールド パターンに当てはまらない 複雑かつユニークな構造を持つ場合があります。更に、レガシーテキストファイルから必要な一部のデータを抽出する必要がある場合があります。FlexText はこれらの問題をルールまたはテンプレートを視覚的に定義し、テキスト データをマップすることのできるアイテムに分割して解決します。FlexText テンプレートは、基本的には、他のフォーマットにマッピングする目的のため、指定する条件に従い カスタムテキスト データの構造モデルを定義します。

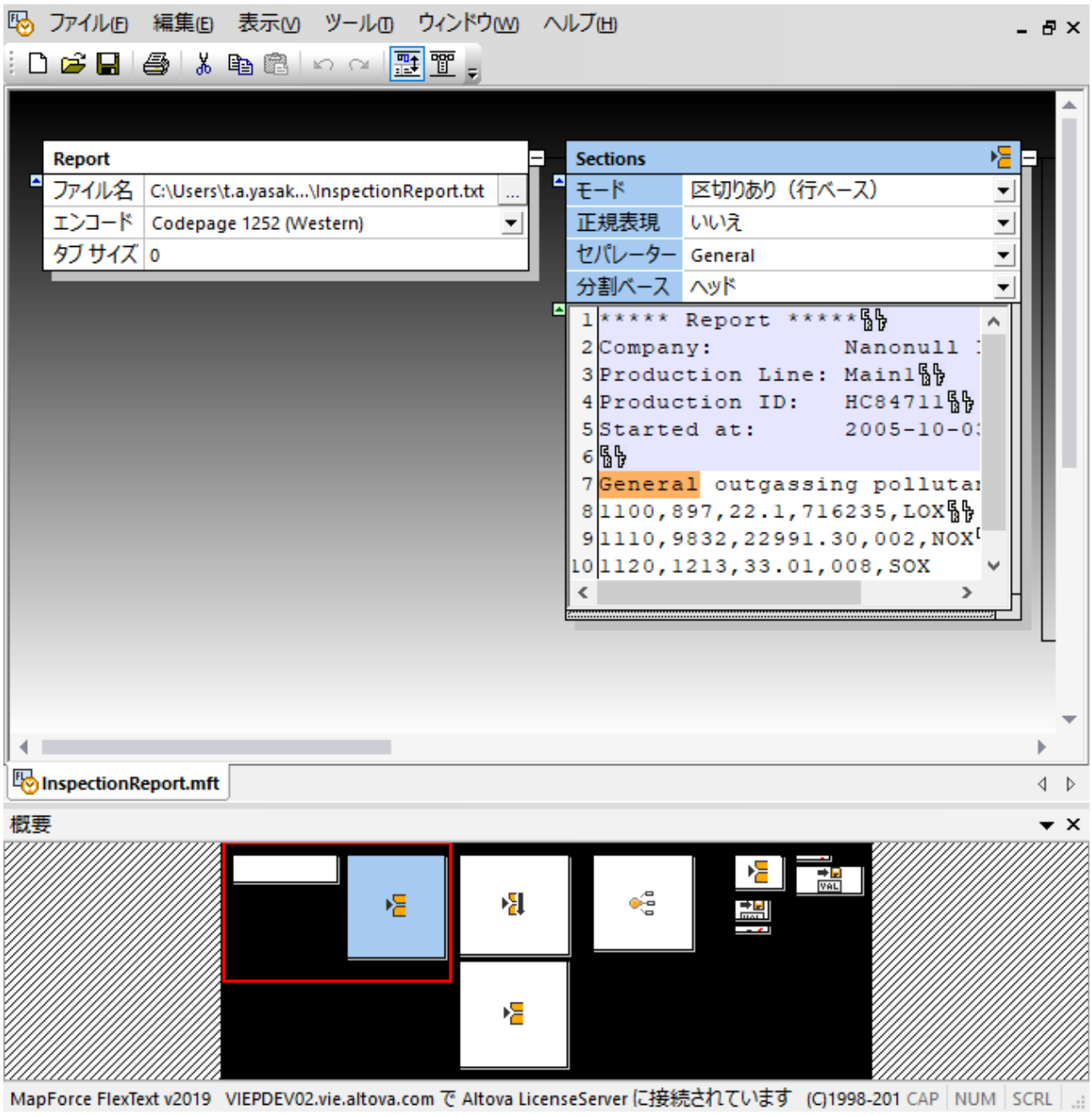
FlexText テンプレートを1 度定義すると、MapForce マッピングエリアにソースまたはターゲット コンポーネントとして追加し、MapForce によりサポートされる他のフォーマットに複雑で標準では内テキスト データをマップすることができます。FlexText テンプレート ファイルには、.mft (MapForce FlexText) 拡張子があります。複数のマッピン内で同じ FlexText テンプレートを再利用し、多数のテキストファイルを処理することができます。



6.4.1 概要

(「**挿入 | テキスト ファイル**」メニューコマンドを使用して) マッピング コンポーネントとしてテキストファイルを追加する際に、MapForce から、FlexText を直接開始することもできます。MapForce インストールディレクトリ内の *Altova MapForce FlexText* 実行可能ファイルとして実行することにより、FlexText をスタンドアロンプログラムとして開始することができます。

下のスクリーンショットは、 **<マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥** ディレクトリ内にある他の FlexText サンプルと共にサンプルFlexText テンプレート (**InspectionReport.mft**) を表しています。



FlexText グラフィカル インターフェイス

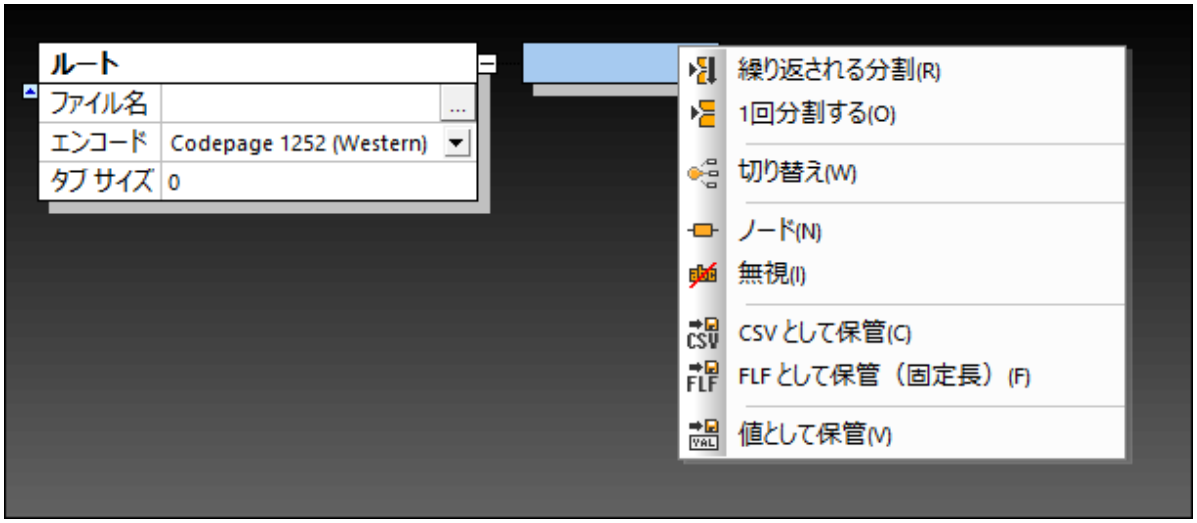
FlexText インターフェイスは次のペインにより構成されています: デザイン ペイン、概要ペイン、とサンプルテキスト ペイン。次に示されるように、これらのペインの位置と外観は、優先順位によりカスタム化することができます。

デザイン ペイン

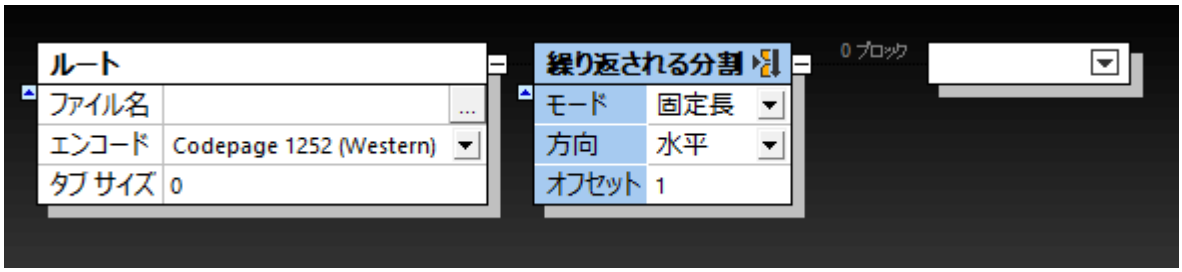
デザイン ペインは、コンテナを使用して、テキストの構造を定義する作業エリアです。 コンテナは、指定する条件に従い、テキスト データを扱うためのルールを定義します。FlexText テンプレートの最初のコンテナは、全体のテキスト ファイルを示す *ルート* コンテナです。すべての他のコンテナ は、（通常、テキストを意味のある単位に分割して、データをマップするため）次のコンテキストの処理ロジックを定義します。例えば、*「1回分割する」* コンテナは、テキストのフラグメントを 2つのフラグメントに分割します。デフォルトでは、コンテナの名前は、関数を説明しています（例えば、*繰り返される分割*）。しかしながら、必要に応じて名前を変更することができます。作業中のテキスト ファイルの構造により必要とされる数のコンテナを定義することができます。

このドキュメントのFlexText レファレンス セクションは、コンテナについての詳細をカバーしています。しかしながら、コンテナに関しては、以下に注意してください:

- コンテナの右上の角のアイコンをクリックすると、ポップアップメニューが開かれ、コンテナ型を選択することができます。



- 各コンテナには、構造内の特定の箇所でテキストデータに何が行われるかを指定することができます。このオプションにより、コンテナのコンテンツを決定し、MapForce コンポーネントに与える前に設定を直すことができます。



- コンテナコンテンツを表示、または、非表示するには、「**展開**」 または「**折りたたむ**」 アイコンをそれぞれクリックします。
- グループとしてコンテナを折りたたむには、**Shift** キーを押してください。2つの二重山かっこ が表示されます。ハンドルをクリックすると、コンテナツリーのセクションクリックされたアイテムの右側に折りたたみます。
- デザインペイン内で、特定のコンテナにより作成されたテキストをプレビューするには、「**デザインビュー内のノードテキスト**」 ツールバーボタンを有効化してください。
- 「**選択されていないノードテキストを自動的に折りたたむ**」 ツールバーボタンが有効化されていると、コンテンツ of a コンテナ のコンテンツは、選択されると、展開されます。すべての他のコンテナへ 折りたたまれます。

概要ペイン

概要ペインは、デザインペインの概観図を表示しています。デザイン ペイン内を移動するには、赤い三角をクリックアンドドラッグします。概要ペインをデタッチして、インターフェイス内で位置を変更するには、タイトル バーをクリックして、ペインを希望する場所にドラッグします。

サンプルテキストペイン

サンプルテキストペインは、現在選択されている コンテナのコンテンツを表示します。(このペイン「**デザインビュー内のノードテキスト**」 ツールバーボタン が有効されていると、表示されず、代わりに、現在選択されている コンテナ のコンテンツがデザインペインに表示されます)。

(C) 2015-2021 Altova GmbH

テキスト内のプレビューで、「タブ」と「Enter キー/改行」文字を素早く識別するために、FlexText は、このような文字をそれぞれ ⇨ と ¶ などの記号を使用して表示します。

6.4.2 FlexText チュートリアル

このチュートリアルは、FlexText の最も一般的な使用方法と、MapForce 内で多種の方法でテキストファイルを処理するための FlexText の最もパワフルな機能について説明します。

このサンプルは、<マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥Tutorial¥ フォルダー内の **Flex-tutorial.txt** ファイルを使用します。.txt ファイルは次の書式で表示されます:

```
111,332.1,22537.7,5,Container ship,Mega,
111,A1579227,10,3,400,Microtome,
111,B152427,7,6,1200,Miscellaneous,
222,978.4,7563.1,69,Air freight,Mini,
222,ZZAW561,10,5,10000,Gas Chromatograph,,

General outgassing pollutants
1100,897,22.1,716235,LOX
1110,9832,22991.30,002,NOX
1120,1213,33.01,008,SOX
```

チュートリアルの目的:

- 111 と222 のキーを含むレコードを個別にマップ可能なアイテムに分離する。
- テキスト形式のレコードを破棄する。
- 残りのレコードのCSV ファイルを作成する。

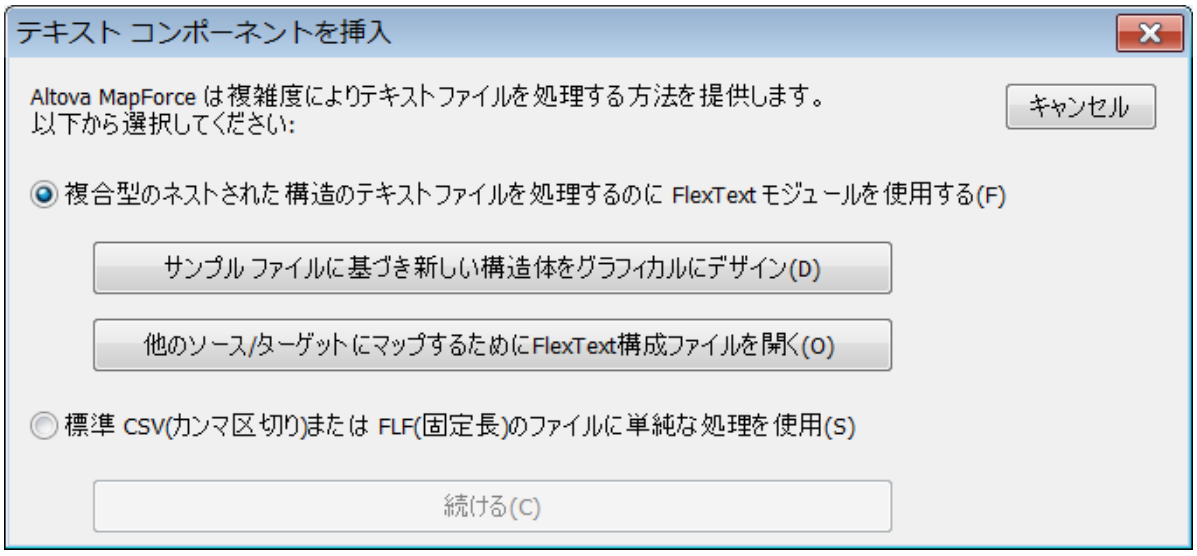
チュートリアルは、次のパートから構成されています:

- ステップ 1: FlexText テンプレートの作成
- ステップ 2: 分割の条件の定義
- ステップ 3: 各コンテナのための複数の条件の定義
- ステップ 4: ターゲット MapForce コンポーネントの作成
- ステップ 5: MapForce 内で FlexText テンプレートを使用する

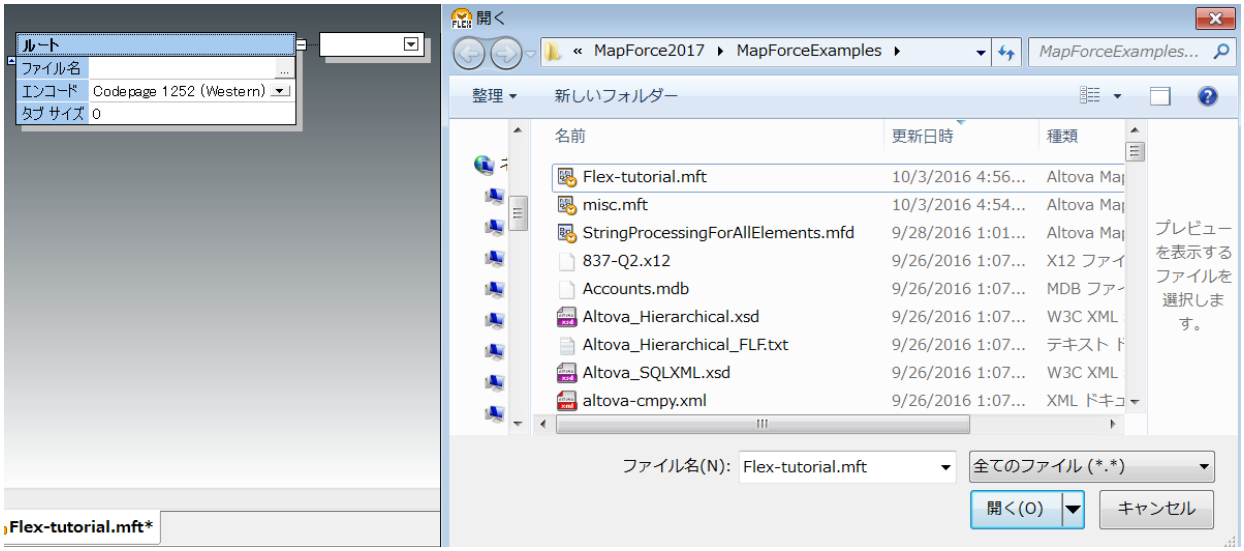
6.4.2.1 ステップ 1: FlexText テンプレートの作成

FlexText テンプレートを作成するには、以下を行います:

1. MapForce を開始して、新規のマッピングファイルを開きます。
2. 「**挿入 | テキストファイル**」を選択、または、テキスト ファイルの挿入アイコン をクリックします。
3. 「**新しい構造タイプをグラフィカルにデザイン ...**」ボタンをクリックします。

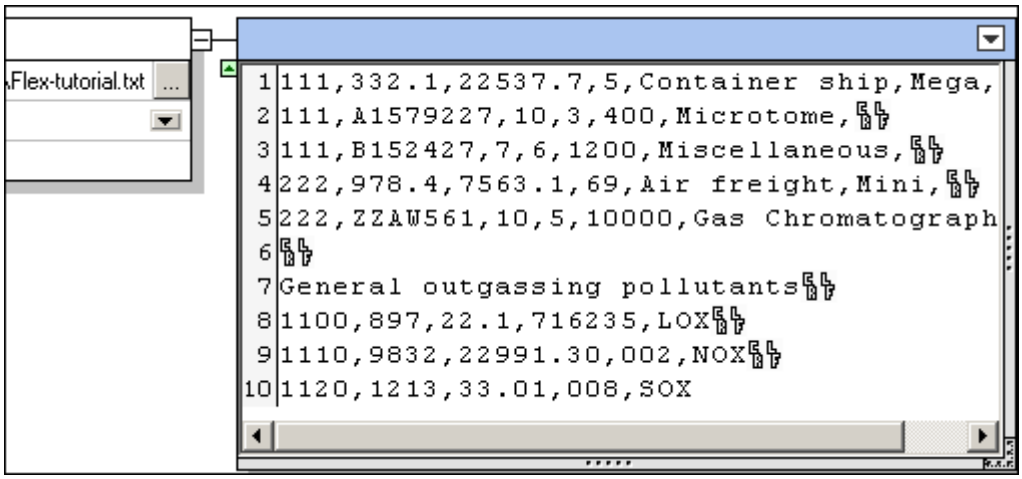


4. FlexText テンプレートのための名前を入力し、「保存」をクリックして継続します（例 Flex-tutorial.mft）。
空のデザインが、「開く」ダイアログボックスと共に表示されます。



5. **...\MapForceExamples\Tutorial** フォルダー内の **Flex-tutorial.txt** ファイルを選択し、「開く」をクリックして確認します。

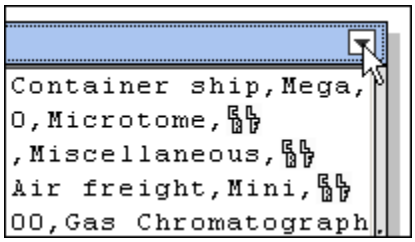
(C) 2015-2021 Altova GmbH



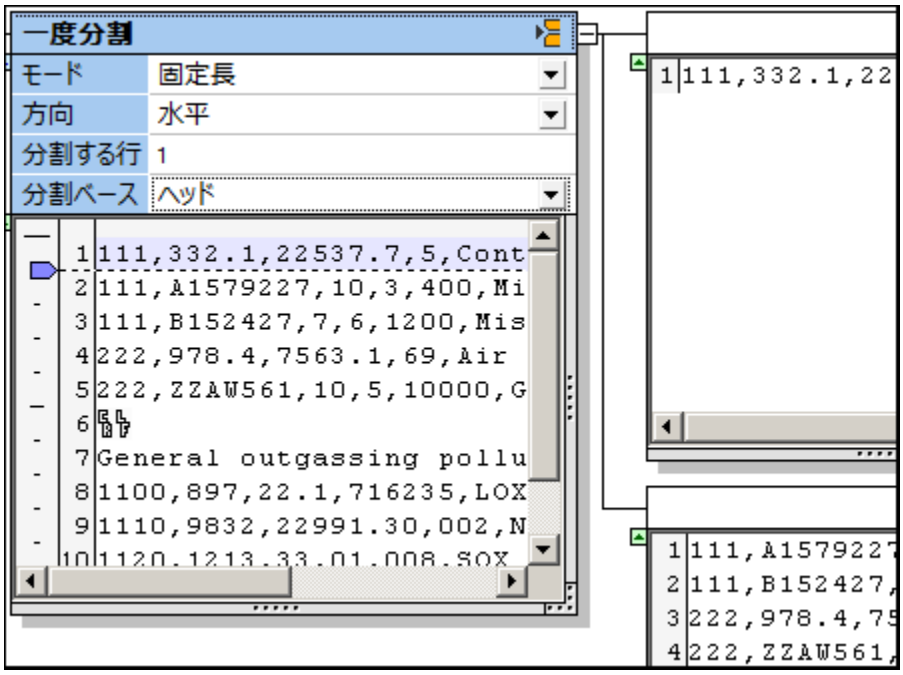
テキストファイルコンテンツが表示されます。

「デザインビューのソードテキスト」アイコン をクリックすると、アクティブなコンテナのコンテンツがサンプルテキストペイン内に表示されます。

「選択されていないソードテキストを自動的に折りたたむ」 を有効化すると、アクティブなコンテナのコンテンツが表示され、コンテンツを含む他のコンテナのコンテンツは折りたたまれます。



6. 右上のコンテナアイコンをクリックし、ポップアップメニューから「1回分割する」を選択します。
「1回分割する」コンテナの横に2つのコンテナが表示されます。「1回分割する」条件に関する詳細は、次を参照してください：「1回分割する」。



「1回分割する」コンテナのデフォルトの設定が表示されます: 固定長、水平、および、分割する行=1.

これらのデフォルトの設定の結果も表示されます:

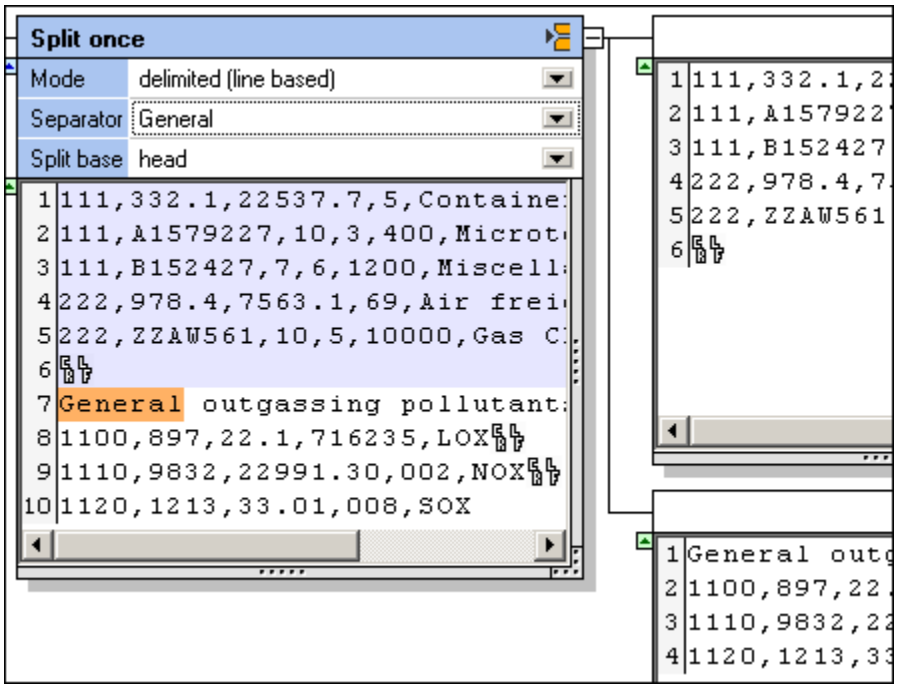
- 上のコンテナは、テキストファイルの最初のラインを含んでおり、「1回分割する」コンテナ内でハイライトされています。
- 下のコンテナは、テキストファイルの残りを含んでいます。

6.4.2.2 ステップ 2: 分割の条件の定義

分割の条件により、テキストのフラグメントを色々な方法で分割することができます。分割の条件を定義するには以下を行います:

1. **モード** コンボボックスをクリックして、「区切られた(行ベース)」を選択します。
2. セパレーターフィールドをダブルクリックして、「General」を入力します。

(C) 2015-2021 Altova GmbH



対応するコンテナ内のフラグメントが変更されます。

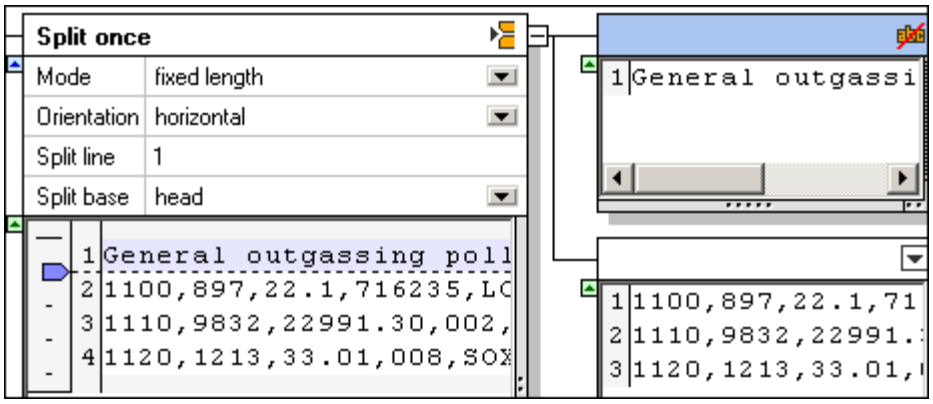
「General」を入力し、区切られた(行ベース)を使用して、文字列「General」を含むテキストのセクションを下のコンテナに分割することができます。セパレーターまでのテキストフラグメントは、上のコンテナ内に置かれます。

下のコンテナと作業して、1100 以上の数値を持つレコードを含むCSV ファイルを生成します。

3. 下のコンテナをクリックして、「1回分割する」に変更します。

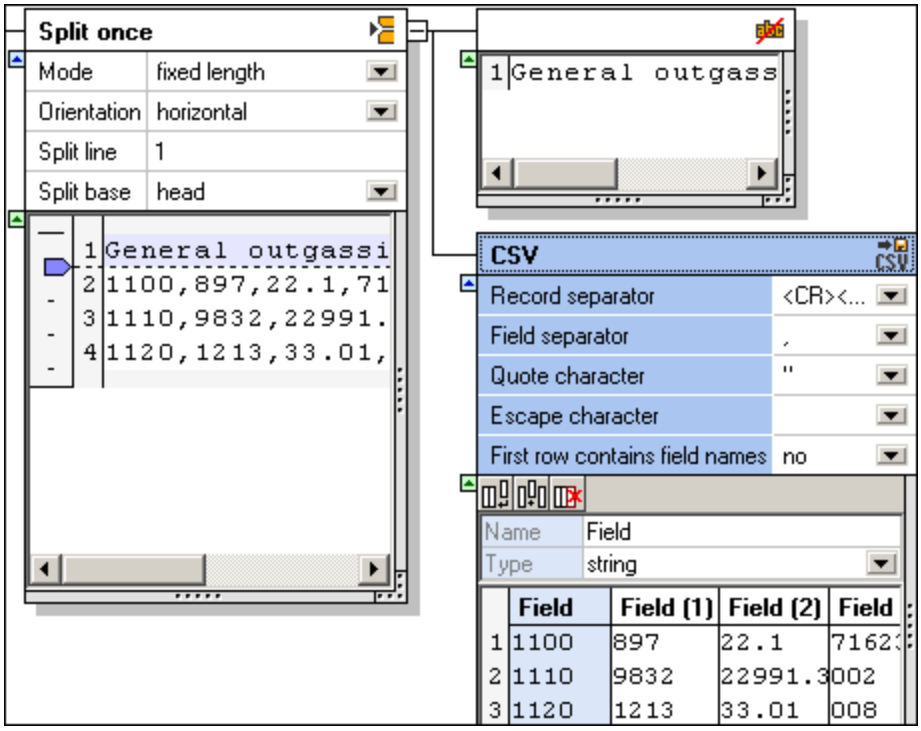
2つの新規のコンテナが作成されます。このテキストフラグメントの最初の行を分割するため、デフォルトの設定はそのままにし、無視することができます。残りのフラグメント in 下のコンテナ内の残りのフラグメントがCSV ファイルになります。

4. 上のコンテナをクリックして、「無視」に変更します。



このコンテナのテキストフラグメント、すなわち、マッピングアイテムが、MapForce 内のマッピングに対して使用できなくなりました。

5. 下のコンテナアイコンをクリックして、CSV として保管に変更します。



コンテナは、テーブル形式でテキストフラグメントを表示しています。デフォルトの設定は、保持することができます。

CSV ファイルの構成

フィールド名を変更する場合は、テーブル内のフィールドをクリックし、「名前」フィールド内のエントリを変更します。このコンテナから列を追加、挿入、削除することもできます。更に詳しい情報に関しては、CSV として保管を参照してください。

残りのテキストフラグメントの定義を続けます。

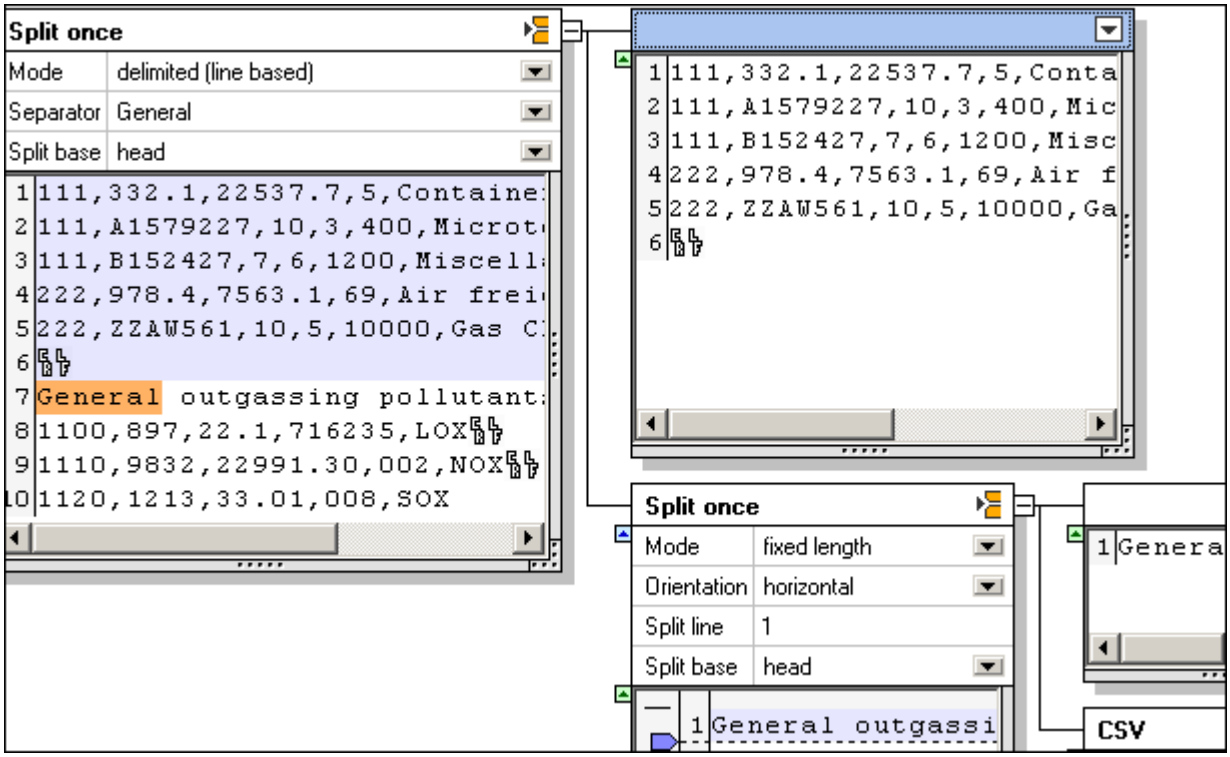
(C) 2015-2021 Altova GmbH

6.4.2.3 ステップ 3: 各コンテナーのための複数の条件の定義

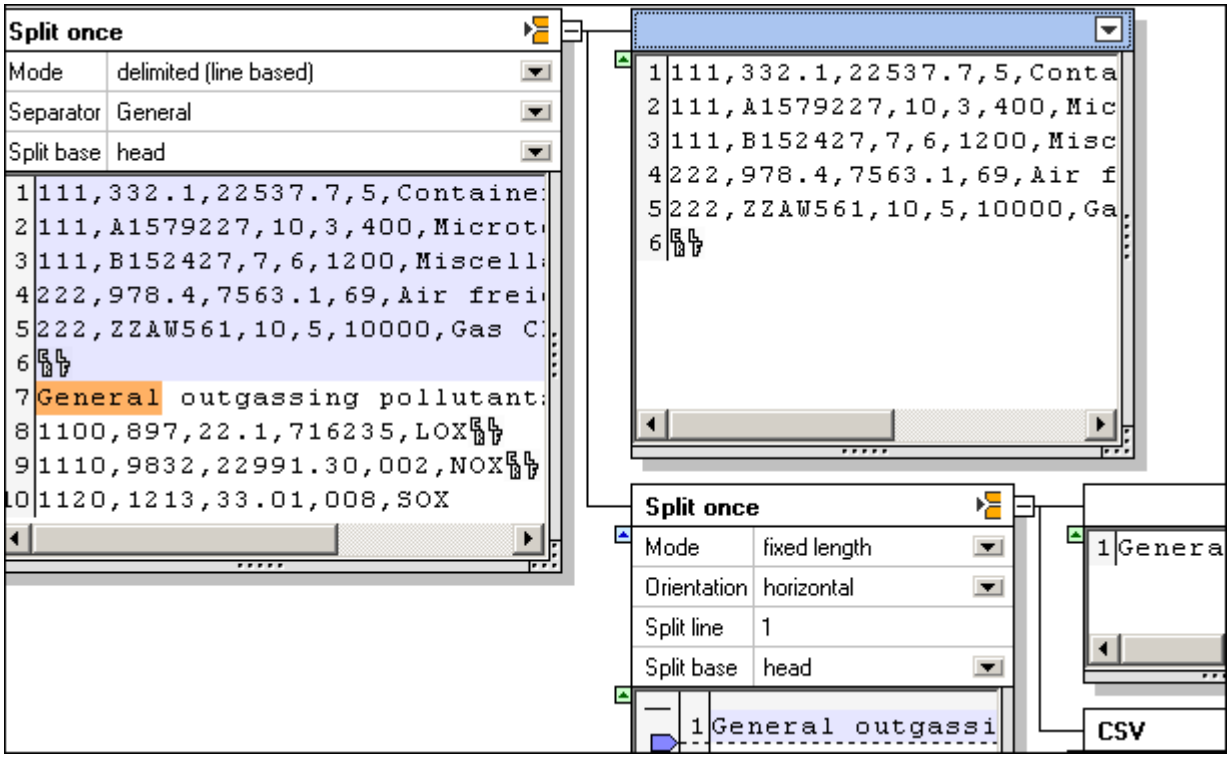
FlexText により切り替え コンテナを使用して、各テキスト フラグメントに対して複数の条件を定義することができます。関連したコンテナは、定義された各条件に対して与えられます。

この時点でのチュートリアルの現状は、最初の「1回分割する」コンテナの下のテキスト フラグメントが以下の条件で定義されています:

- 「1回分割する」コンテナは、最初の行を分割し、コンテナを無視します。
- 残りの セグメントは、CSV ファイルとして定義/保管されます。



1. 上のコンテナ アイコンをクリックして、繰り返される分割に変更します。

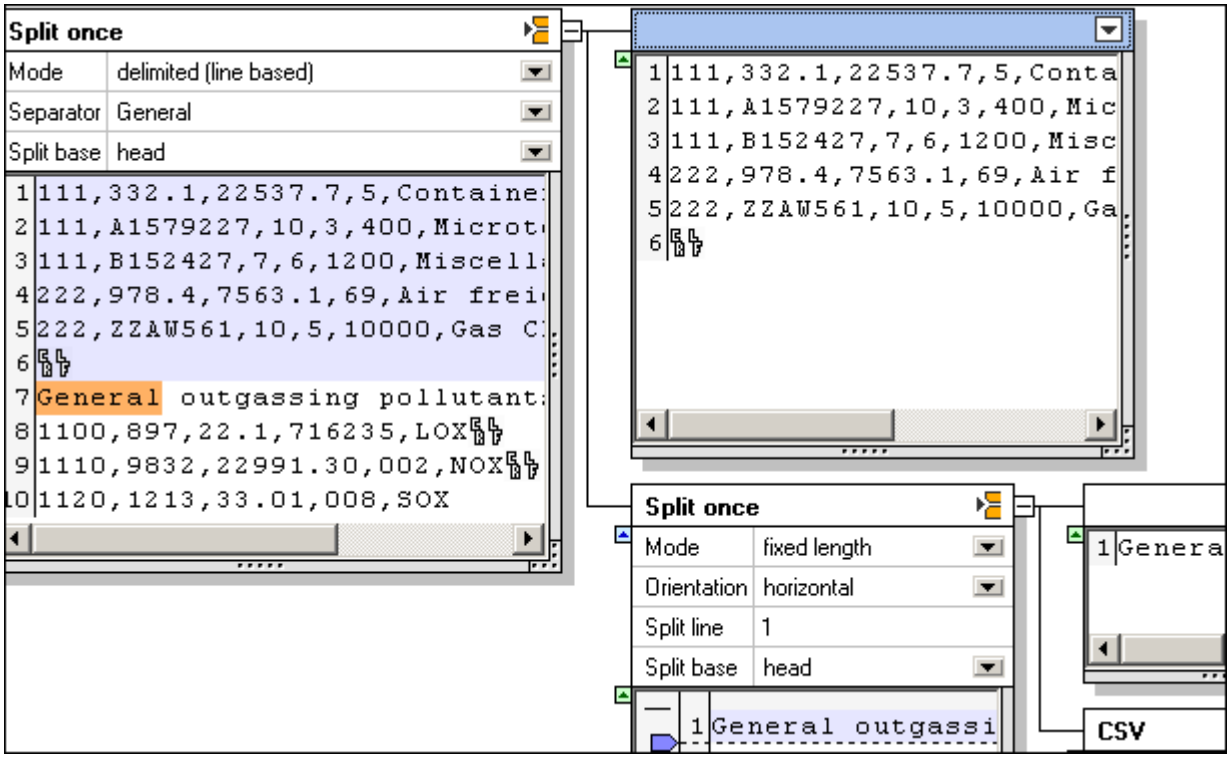


現時点では、デフォルトの設定が必要とされ使用されます。テキストフラグメントは、それぞれが1行からなる複数のテキストブロックに分割されます。関連したコンテナは、各テキストブロックのプレビューを表示します。

次のテキストブロックアイコン をクリックすると、6つある全てのテキストフラグメントを順番に確認することができます。

2. コンテナをクリックして、「切り替え」に変更します。

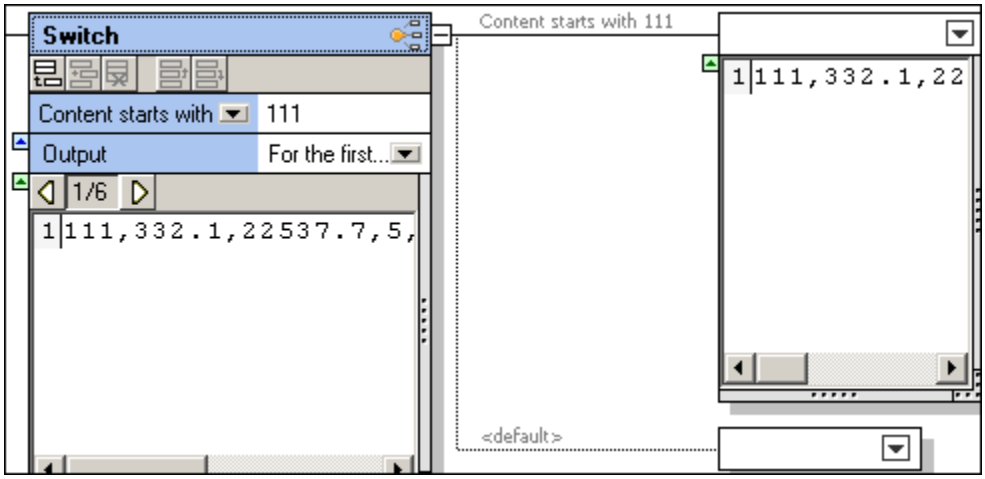
(C) 2015-2021 Altova GmbH



「切り替え」コンテナの最初の状態は、上に示されています。

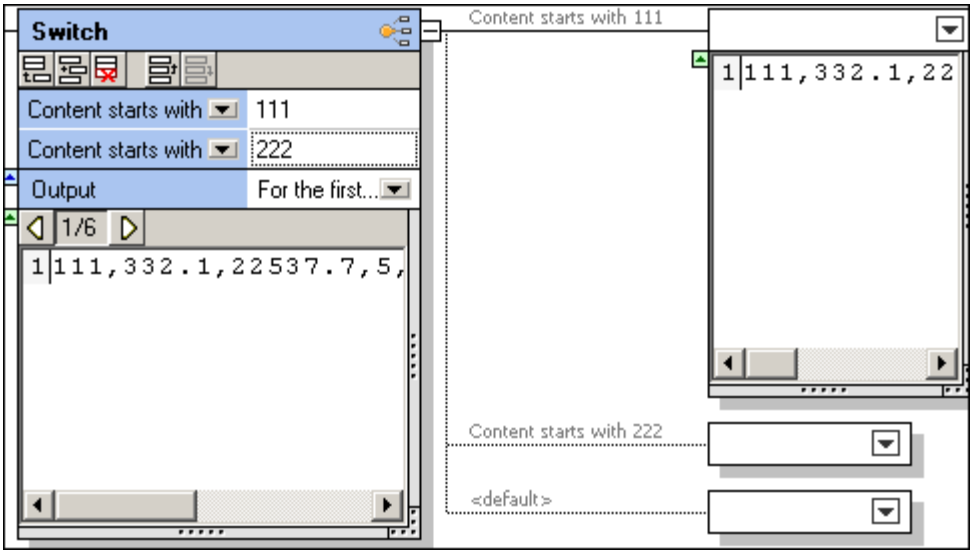
- 関連したコンテナ「デフォルト（default）」が追加されました。
- 最初のレコード 1/6 のコンテンツは、デフォルトのコンテナ内に表示されています。

3. 「切り替え」タイトルバー内の条件を追加アイコン をクリックして新規の条件を追加します。
4. 「先頭」内のフィールドをダブルクリックして、111 を入力します。



これにより最初の条件が定義されます。（コンテンツが111 から開始する）関連したコンテナ「デフォルト（default）」コンテナの上に追加されます。

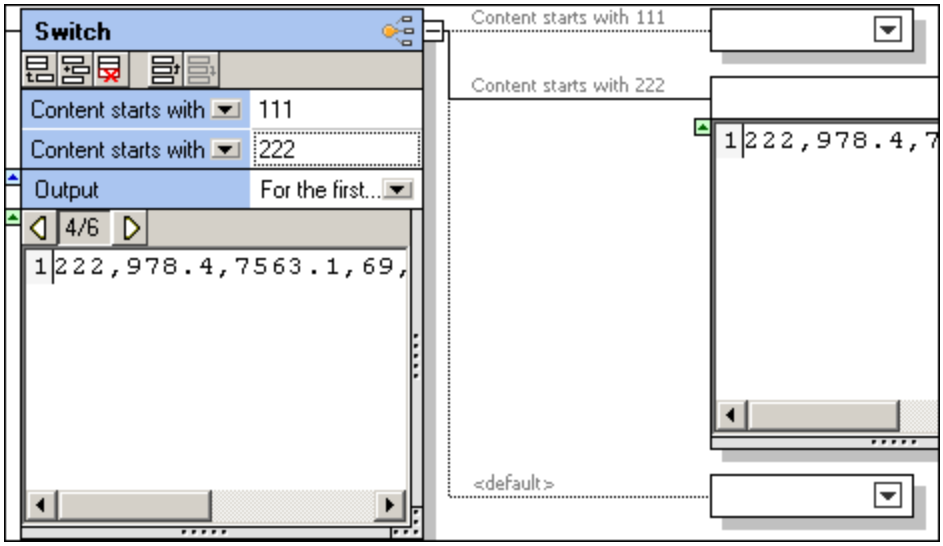
5. 追加アイコンをもう一度クリックし、「先頭」フィールドに「222」を入力します。



(数値 222 から始まる) 3番目のコンテナが追加されました。

メモ 「先頭」コンボボックスをクリックすると、「含む」オプションを選択することができます。これにより、テキストフラグメント内のどこでも発生することのできる「string」を指定することができます。

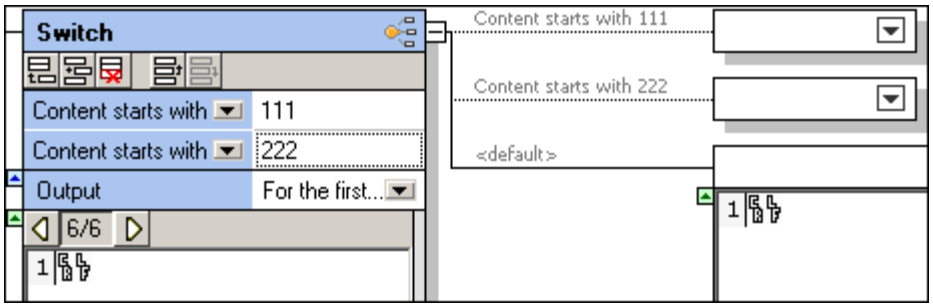
6. 次のテキストブロックアイコン を数回クリックして、結果を確認します。



6 件中 4 件目のレコードまで到達すると、コンテナ 222 が開かれ、コンテンツが表示されます。

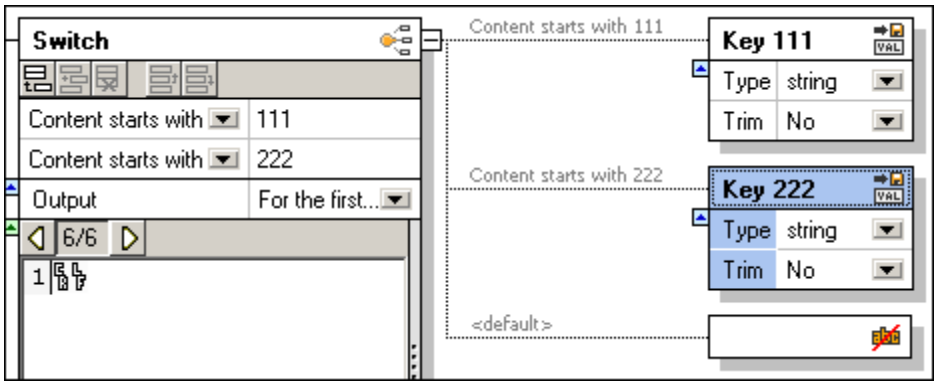
7. 6 件中の6 件目のまで到達するまでクリックし続けます。単一のCR / LF 文字 がデフォルトのコンテナに表示されます。

(C) 2015-2021 Altova GmbH



現在のブロック内のデータフラグメントが条件を満たす場合、そのブロックの「**完全なデータ**」が関連するコンテナにパスされます。
条件を満たす場合、データは分割されず、関連するコンテナ、または、デフォルトのコンテナにルートされます。

8. 最初の2つのコンテナをクリックして、「**値として保管する**」に変更します。 最後のコンテナをクリックして、「**無視**」に変更します。
9. 「保管」テキストをダブルクリックして、Key 111 とKey 222 などの説明テキストを追加します。

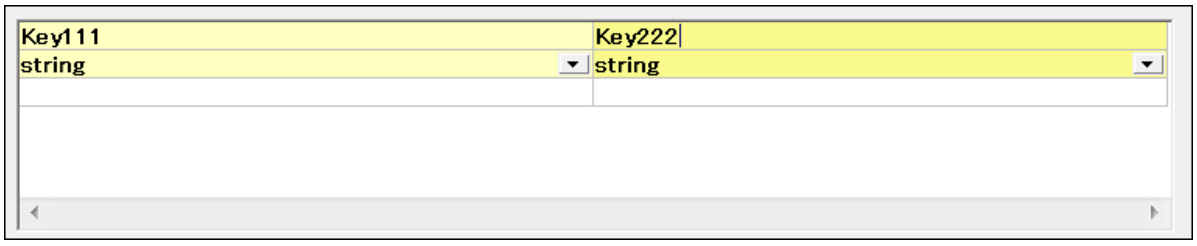


10. FlexText テンプレートを保存します。例 Flex-Tutorial.mft。

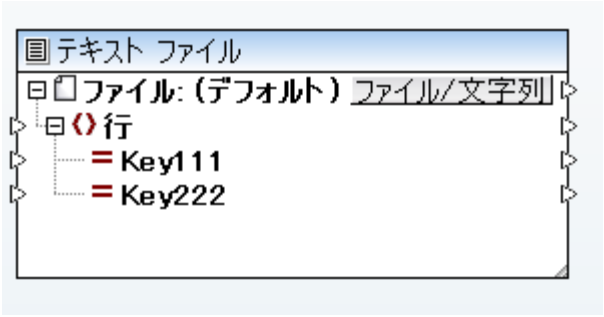
6.4.2.4 ステップ 4: ターゲット MapForce コンポーネントの作成

FlexText ソースで使用するためにターゲットコンポーネントを作成するには以下を行います:

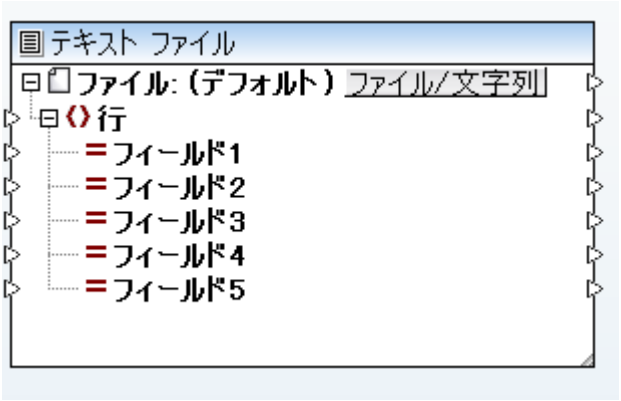
1. 「**挿入 | テキストファイル**」を選択、または、テキスト ファイルの挿入アイコン をクリックします。
2. 「**単純な処理を使用...**」ラジオボタンをクリックして、「**続ける**」をクリックします。
   これにより、コンポーネント設定ダイアログボックスが開かれます。
3. 「**フィールドを追加**」ボタンをクリックして新しいフィールドを追加します。
4. フィールド1 フィールド名をダブルクリックして、「Key 111」に変更します。



5. 他のフィールドも同様に処理し、**Key 222** と名前を付け、「OK」をクリックして確認します。
   2つのフィールドのあるテキストコンポーネントが作成されます。



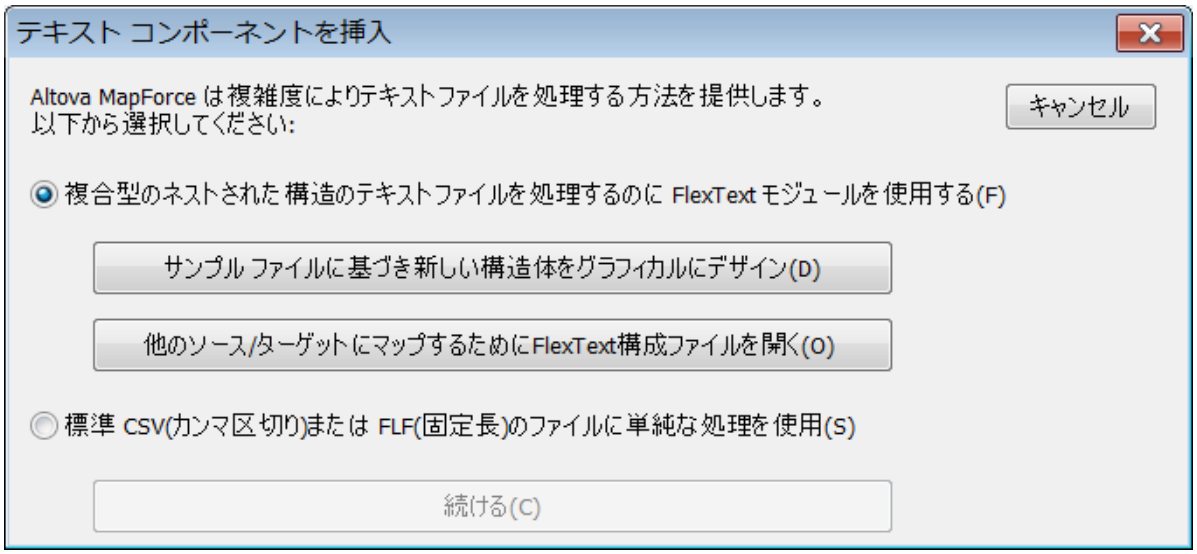
6. 同じ方法を使用して、5つのフィールドから構成される2番目のテキストコンポーネントを作成します。



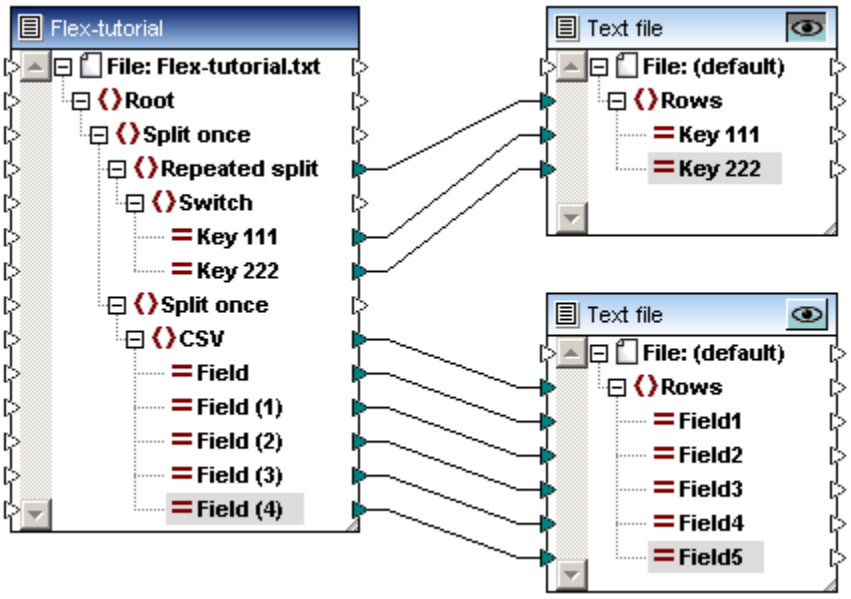
6.4.2.5 ステップ 5: MapForce 内で FlexText テンプレートを使用する

MapForce 内で FlexText テンプレートを使用する方法:

1. アプリケーションを開始、または、MapForce に切り替え、「**挿入 | テキスト ファイル**」を選択します。



2. 「既存の FlexText 構成ファイルを開く... 」ボタンをクリックして、以前に定義されている FlexText テンプレート（**Flex-tutorial.mft**）を選択します。
   MapForce コンポーネントの構造は、FlexText 内のデザインビュー内のコンテナを反映しています。



3. 既に定義済みのターゲット コンポーネントに多種のアイテムをマップし、出力タブをクリックして、結果をプレビューします。

```
1   "111,332.1,22537.7,5,Container ship,Mega,
2   ",
3   "111,A1579227,10,3,400,Microtome,
4   ",
5   "111,B152427,7,6,1200,Miscellaneous,
6   ",
7   ,"222,978.4,7563.1,69,Air freight,Mini,
8   "
9   ,"222,ZZAW561,10,5,10000,Gas Chromatograph,,
10  "
11
```

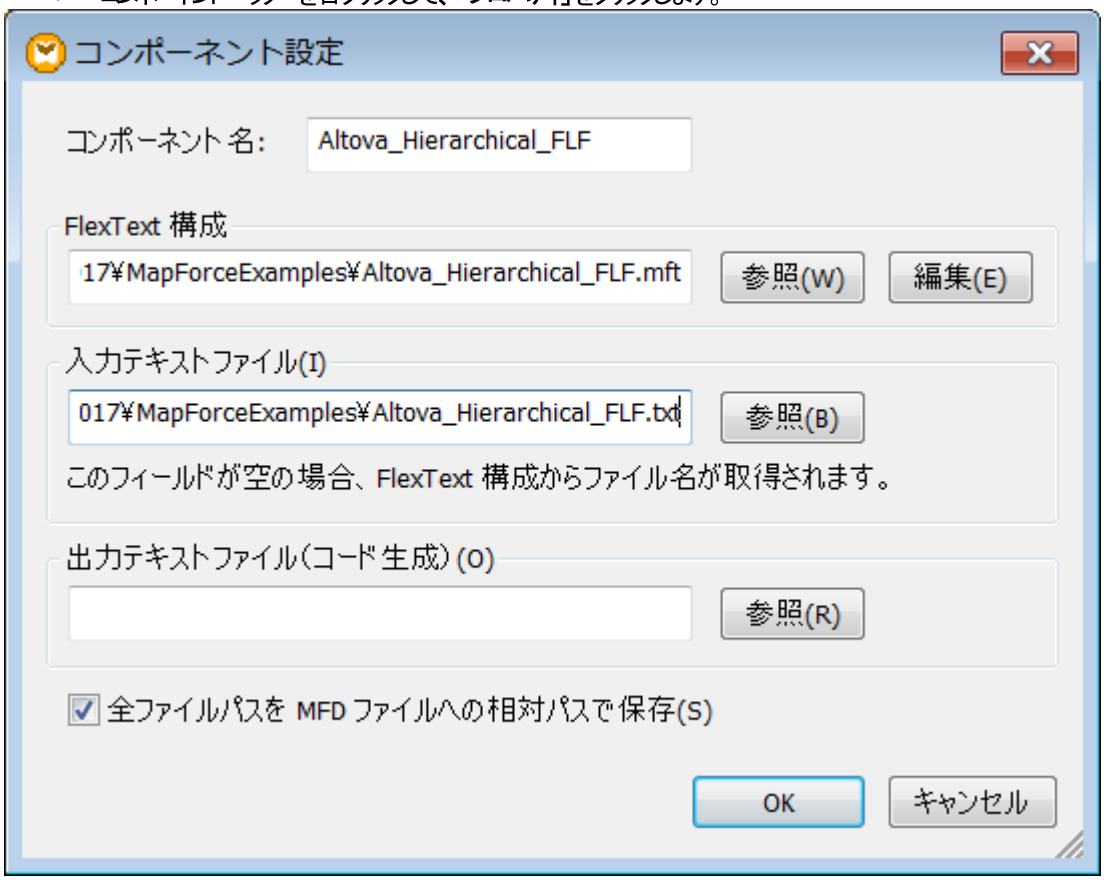
下のテキスト コンポーネントのマッピング プレビュー:

```
1   1100,897,22.1,716235,LOX
2   1110,9832,22991.30,002,NOX
3   1120,1213,33.01,008,SOX
4
```

6.4.3 FlexText コンポーネント設定

FlexText コンポーネントをマッピング エリアに追加すると、コンポーネント設定ダイアログ ボックスから、適用することのできる設定を構成することが可能になります。コンポーネント設定ダイアログ ボックスを以下の方法を用いて開くことができます:

- 「コンポーネント」メニューから、「プロパティ」をクリックします(このメニューアイテムは、コンポーネントを選択すると有効化されます)。
- コンポーネントヘッダーをダブルクリックします。
- コンポーネントヘッダーを右クリックして、「プロパティ」をクリックします。



FlexText コンポーネント設定ダイアログボックス

使用することのできる設定は以下の通りです。

コンポーネント名	コンポーネント名は、コンポーネントを作成すると自動的に生成されます。名前はいつでも変更することができます。 コンポーネント名が自動的に生成され、その後、インスタンスファイルを選択すると、MapForce は任意でコンポーネント名を更新するようにプロンプトします。 コンポーネント名には、スペースと終止符文字を含むことができます。コンポーネント名には、スラッシュ、バックスラッシュ、コロン、二重引用符、先頭および末尾のスペースを含むことができません。一般的には、コンポーネント名を変更する際、以下の点に注意してください: • FlowForce Server にマッピングをデプロイする場合、コンポーネント名を一意の名前にしてください。 • コマンドラインに入力することのできる文字のみを使用することが奨励されます。国別使用文字によってはWindows とコマンドライン内で異なるエンコードを使用する場合があります。

FlexText 構成	コンポーネントにより使用されているFlexText テンプレート(.mft) の名前、または、パスを指定します。 新規のテンプレートを選択するには、「参照」をクリックします。FlexText 内のテンプレートを編集するには、「編集」をクリックします。
入力テキストファイル	MapForce がデータを読み取るテキストファイルを指定します。このフィールドは、ソースコンポーネントにとり重要で、最初にコンポーネント作成した際に作成され、テキストファイルに割り当てられます。 ソース テキスト ファイルは、FlexText テンプレート内で定義されているものを優先します。FlexText テンプレート内で定義されているファイル名がソースとして使用されます。
出力テキストファイル(コード生成のため)	MapForce がデータを書き込むためのテキストファイルのパス名を指定します。このフィールドは、コードを生成する際に、ターゲットコンポーネントにとって重要です。フルパスを入力することにより、ターゲットディレクトリを特定的に定義することができます。例えば、**c:\myfiles\sequence.txt** などがあげられます。
MFD ファイルに相対的な全てのファイルパスを保存する	このオプションが有効化されていると、MapForce は、コンポーネント設定ダイアログボックスに表示されているMapForce Design (.mfd) ファイルの場所に対して相対的なファイルパスを保存します。

6.4.4 ターゲットコンポーネントとして FlexText を使用する

FlexText コンポーネントの主な使用方法は、テキストファイルのセクションを削減し、分割し、FlexText コンポーネントの対応するアイテムを他のターゲット コンポーネントにマップすることです。FlexText を、しかしながら、ターゲット コンポーネントとして使用し、個別のファイルを構成、または、再構成して、単一のファイルに構成することができます。期待する結果を達成するには、試行錯誤を必要とする可能性があります。
FlexText コンポーネントを ターゲット として使用すると、以前に定義された操作が元に戻されます。ファイルを多数のサブセクションに分割する代わりに、ファイルを構成、または、再構成します。

一般的には、ターゲットとして使用される場合、 FlexText テンプレート内で定義されている各操作を元(ボトムアップ)に戻します、:

- 「分割」は、「マージ」になります。例 繰り返される分割のマッピングは、「,」により区切られ、マージ アイテムは「,」により分割されます。
- 「保管」は、「ロード」になります。
- 「切り替え」は、「最初の一致項目を選択する」になります。

ターゲット コンポーネントとしてFlexText を使用する場合、以下に注意してください:

- データ ソース コンポーネントとFlexText コンポーネントの入力 アイテム1つの間の接続が構築されると、 FlexText コンポーネント データ ソースは、無視されます。データは、新規にマップされたソースコンポーネントにより与えられ、優先順位を有します。
- テキストは、(CSV とFLFとして保管)「保管...」 コンテナにマップされ、セパレーターは保持されます。しかしながら、「保管..」ノードの上のノード内で固定長 分割が起こる場合、テキストは切り捨てられる可能性があります。
- 固定された幅の分割では、分割ベース=ヘッドの場合、左/上のセクションが定義されている長さに対し切り捨てられる可能性があります。または、分割ベース=テールの場合、右/下 セクションが定義されている長さに対し切り捨てられる場合があります。テキストが短すぎる場合、スペース 文字が、セクションに挿入されます。
- FlexText は、通常、分割 操作のアイテムの間にセパレーター(固定された分割のための空白スペース) を挿入します。しかし、これは、「区切られた(行ベース)」分割には、適応されません。「区切られた(行ベース)」操作は、操作を完全に元に戻すことはできません。「区切られた」テキストは、最初の行内で発生し、テキストに含まれます、このための、自動的な処理により 正確に追加することはできない可能性があります。
  - 区切られた(行ベース) は、最初の行が不在の場合セパレーターは追加されません。
  - 区切られた(フロート) は、2つのセクション間にセパレーターを追加します。

- 切り替え 操作は、簡単なケース以外は元に戻すことはできません。このデータ 切り替えは、自身のブランチをデータが含まれる最初のブランチのために スキャンし、このデータを使用、または、挿入します。切り替え操作の最初の接続のみがマップされます。 残りの切り替えコンテナにデータを転送するには、フィルターを残りの接続のために定義し、繰り返される分割アイテムに入力されるように、切り替えアイテムが単一アイテムを返すように切り替え親アイテムの複製を作成する必要があります。
- 単一の 分割 コンテナの子に対してマッピングすると、最後のアイテム以外すべてのマッピングの結果は破棄されます。複数の結果が生成された場合でも、単一の 結果が 時保持されます。

次のXML スキーマ コンテンツモデルの例は、ターゲットとして使用されている FlexText の振る舞いについて説明しています:

- 繰り返される分割は、繰り返し要素です。
- 単一の 分割は、'シーケンス' コンテンツモデル グループの一部を作成します。
- 保管は'choice' コンテンツモデル グループを作成します。各ケースが可能な子要素です。
- 保管は、単純型の要素を作成します。

6.4.5 FlexText レファレンス

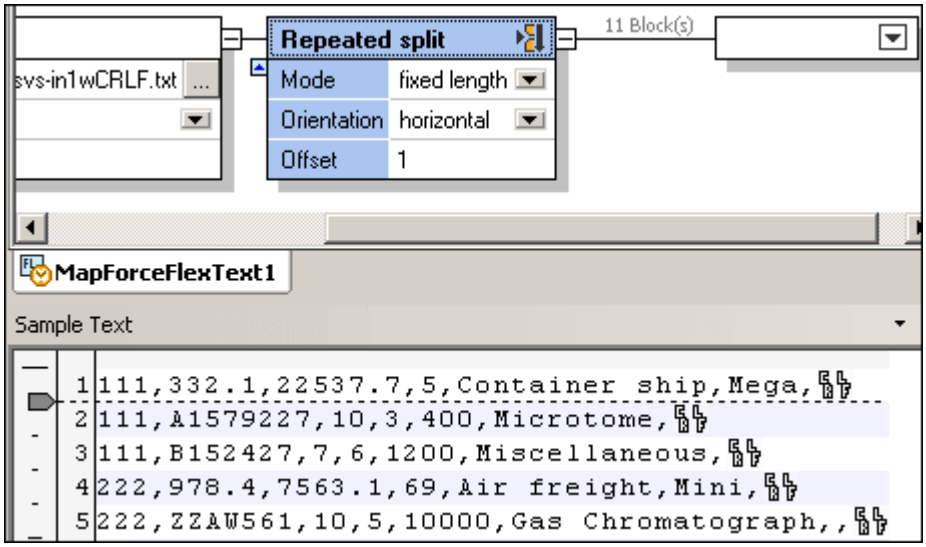
レファレンスのセクションは、FlexText の異なる機能について説明し、特定の結果を取得するための使用方法示しています。

6.4.5.1 繰り返される分割

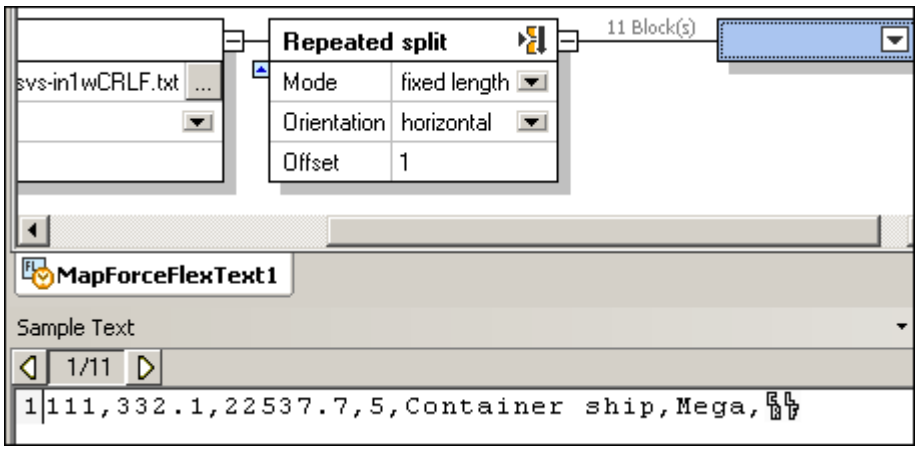
このオプションを使用すると、単一のコンテナが最初に作成されます。コンテナには、**繰り返される分割**の条件のセットにより定義されたテキストが含まれます。繰り返される分割のバージョンは以下の通りです: 固定長、区切られた フロート、区切られた 行ベース、と区切られた 行の開始...。
このオプションを最初に選択すると、デフォルトのパラメーターは、自動的に設定され、結果フラグメントは、関連したコンテナに表示されます。繰り返される分割 コンテナが現在アクティブであり、サンプルテキストペイン内にプレビューは、すべての現在の レコード/列を表示することに注意してください。

コンテナのデフォルトの設定:

モード	固定長
方向	水平
オフセット	1



デフォルトの結果:
テキストの各行は、新しいコンテナ内の行/レコードとして表示され、オフセットは1です。新しいコンテナをクリックして、コンテンツをプレビューします。 サンプルテキスト「スクロールの矢印」により、この設定により生成された11 件のブロック/フラグメントのそれぞれをスクロールすることができます。



6.4.5.1.1 モード – 固定長

繰り返される分割（固定長）モード を使用して、テキストを複数の水平、または、垂直なフラグメントに分割することができます。**繰り返される分割（固定長）モード**に適用することのできる設定は、以下で説明されています。

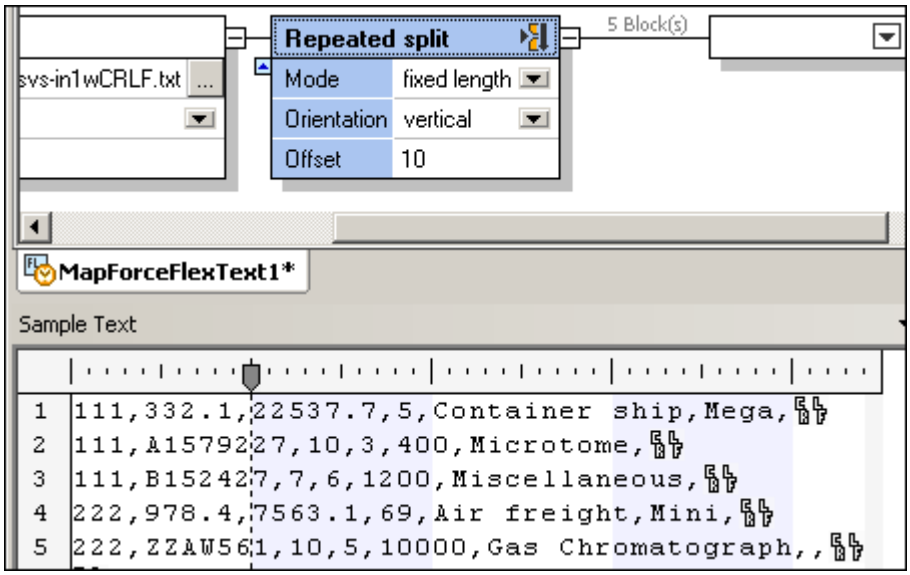
方向

テキストフラグメントが水平方向、または、垂直方向に分割されるかを指定します。フラグメントを複数の水平のセクションに分割する場合は、「水平」を選択します。フラグメントを複数の垂直のセクションに分割する場合は、「垂直」を選択します。

オフセット

方向が「水平」に設定されている場合、これにより設定は、各フラグメントが含む列数を指定します。 方向が「垂直」に設定されている場合、この設定は、各フラグメント内の幅を指定します。デフォルトのオフセットは、「1」です。この設定を変更するには、以下のうちの1つをおこなってください:

- オフセット フィールドに値を挿入します。
- タブを垂直、または、水平ルーラーにドラッグします。



6.4.5.1.2 モード - 区切られた(フロート)

「区切られた(フロート)を使用した繰り返される分割」モードを以下のシチュエーションで使用することができます:

- 指定されたセパレーター文字が結果フラグメントから取り除く。
- セパレーターがインラインの箇所のテキストを分割する(例えば、CR/LF 文字を含まないテキスト)。

メモ フラグメントは、セパレーターの後の最初の文字と同じセパレーターの次のインスタンスの最後の文字の前の間として定義されます。下の例で示されるように、このルールに対しての例外は、最初を最後のフラグメントです。

「区切られた(フロート)を使用した繰り返される分割」) モードに適用することのできる設定は以下にリストされています。

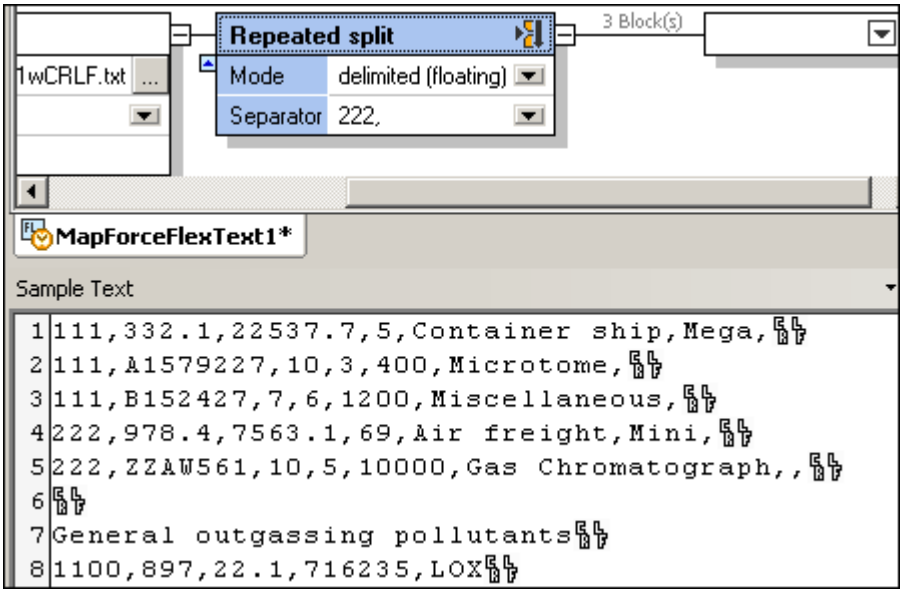
正規表現

これは、正規表現がマッチする都度、テキストをフラグメントに分割する任意の設定です。(次を参照: 正規表現を使用してテキストを分割する)。デフォルトの設定は「いいえ」です。

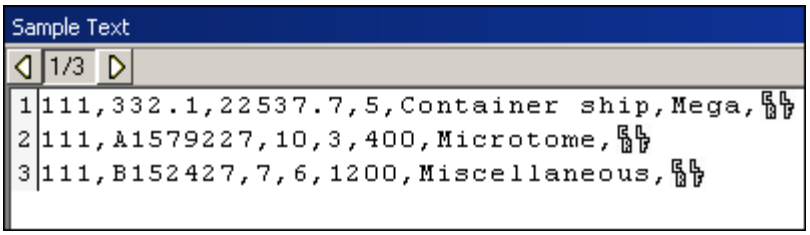
セパレーター

セパレーターとして使用される文字を指定します。デフォルトの値は(セパレーター無し)「無し」です。

例えば、セパレーター「222」を下に表示されるテキストに対して使用すると、3つの個別のフラグメントが作成されます。



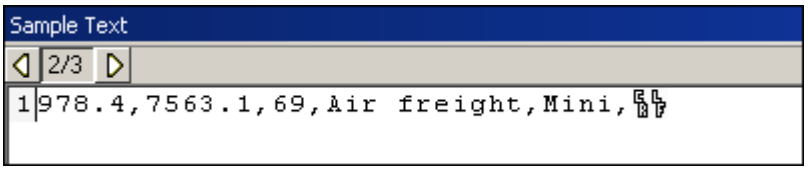
最初のフラグメントは、フラグメントの開始から最初のセパレーター(「222」)の開始、すなわち「111」から「Miscellaneous」まで、の全ての文字 を含みます。



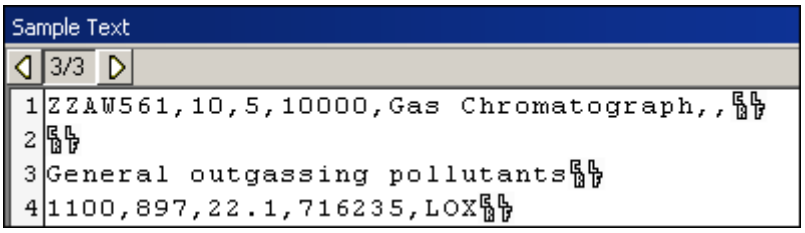
セパレーターが、フラグメント内の最初の行の文字の最初のセットではない場合、このサンプルで示されるように、最初のフラグメントには、セパレーターの最初のインスタンスまでの全てのテキストが含まれます(例えば、「222」)。

「111」がセパレーターの場合、最初のフラグメントは、ソースフラグメントの最初の行の最初に現れるセパレーターである、ゼロの長さの文字列です。

2番目のフラグメントは、セパレーター222 を含む最初の行を、セパレーター自身は無しで含みます。



3番目のフラグメントは、セパレーター222 を含む次の行を、セパレーター自身は無しで、テキストファイル/フラグメントの終わりまで含みます。



6.4.5.1.3 モード - 区切られた(行ベース)

「区切られた(行ベース)を使用した繰り返される分割」モード は、次のように振る舞い、テキストを複数のフラグメントに分割します:

- このモードは、セパレーターフィールドに入力したセパレーター文字により定義された複数のフラグメントを作成します。
- セパレーター文字は、フラグメント内に含まれています。
- フラグメントがセパレーターを含む最初の行までのセパレーターを含む全体の行として定義されます。
- 最初の行でセパレーターが発生しない場合、最初のフラグメントには、セパレーターを含む最初の行までの行が含まれます。

「区切られた(行ベース)を使用した繰り返される分割」モードに適用することのできる設定は以下にリストされています。

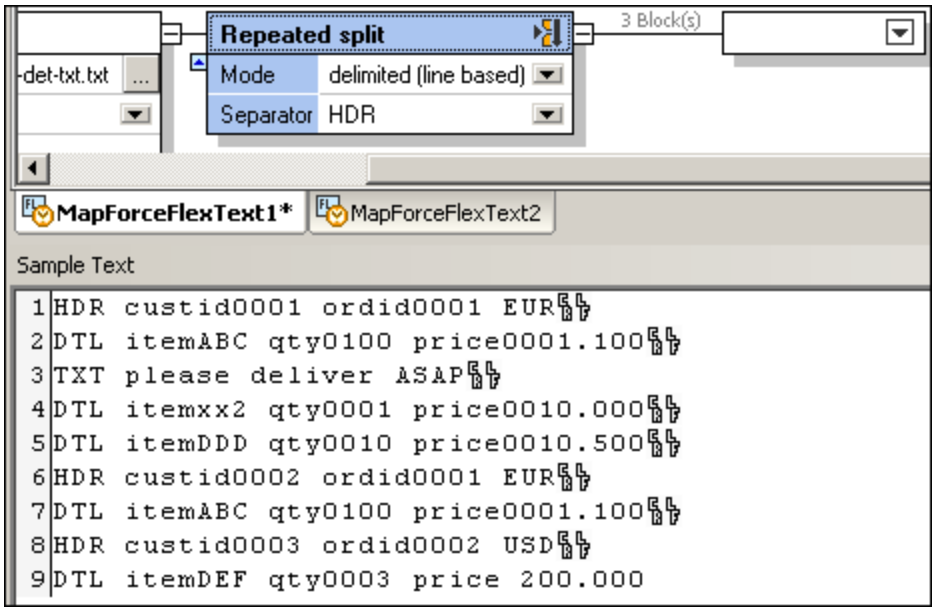
正規表現

これは、正規表現がマッチする都度、テキストをフラグメントに分割する任意の設定です。(次を参照: 正規表現を使用してテキストを分割する)。デフォルトの設定は「いいえ」です。

セパレーター

セパレーターとして使用される文字を指定します。デフォルトの値は「無し」(セパレーター無し)。

例えば、セパレーター「HDR」をテキストに対して下に表示されるように使用すると、3つの個別のフラグメントが作成されます。



最初のフラグメントは、全ての行から同じセパレーターを含む次の上までのファイルフラグメントから開始する全ての文字を含んでいます。

(C) 2015-2021 Altova GmbH

```
1 HDR custid0001 ordid0001 EUR
2 DTL itemABC qty0100 price0001.100
3 TXT please deliver ASAP
4 DTL itemxx2 qty0001 price0010.000
5 DTL itemDDD qty0010 price0010.500
```

このオプションにより、2つのセパレーターの間の無数の行にアクセスすることが許可されます。これは、(このサンプルでは、「DTL」または「TXT」などの)任意の、または、シーケンスではない、レコード型をもつファイル内で役に立ちます。

2番目のフラグメントには、2番目の「HDR」の発生から次の発生までのすべてのテキストが含まれています。

```
1 HDR custid0002 ordid0001 EUR
2 DTL itemABC qty0100 price0001.100
```

3番目のフラグメントには、3番目の「HDR」の発生からの全てのテキストが含まれています。

```
1 HDR custid0003 ordid0002 USD
2 DTL itemDEF qty0003 price 200.000
```

6.4.5.1.4 モード - 区切られた(行の開始)

「区切られた(行の開始)を使用した繰り返される分割」モードは、次のように振る舞い、テキストを複数のフラグメントに分割します:

- このモードは、セパレーターフィールド内に入力するセパレーター文字により定義される複数のフラグメントを作成します。
- セパレーター文字 は、フラグメント内に含まれています。
- フラグメント、セパレーターから開始して、行の最初に同じセパレーターが含まれる次の行までの全体の行として定義されます。
- セパレーターが、最初に行に存在しない場合、セパレーターを含む最初の行までの行を含んでいます。

「区切られた(行の開始)を使用した繰り返される分割」モードに適用することのできる設定は以下にリストされています。

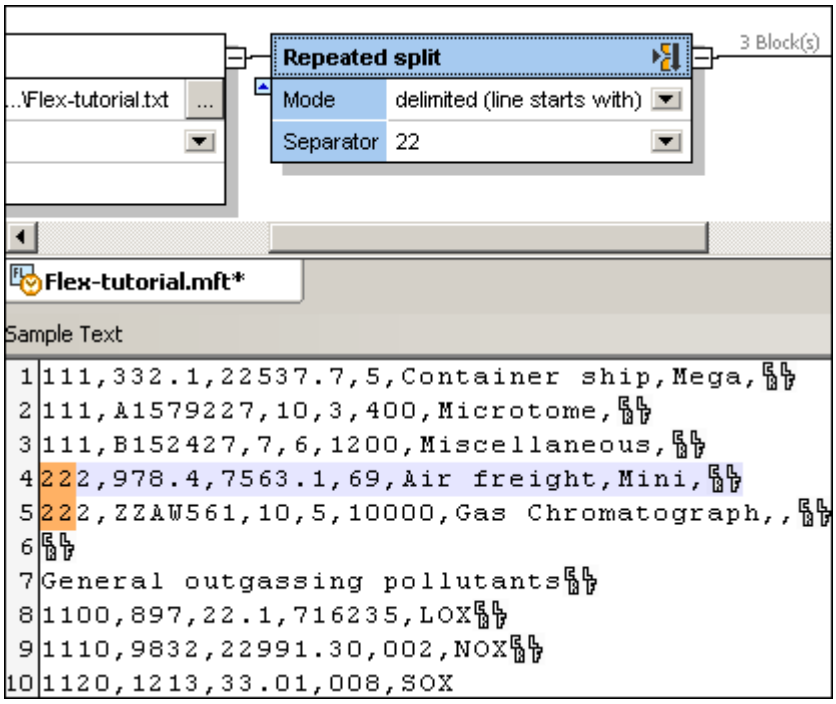
正規表現

これは、正規表現がマッチする都度、テキストをフラグメントに分割する任意の設定です。(次を参照: 正規表現を使用してテキストを分割する)。デフォルトの設定は「いいえ」です。

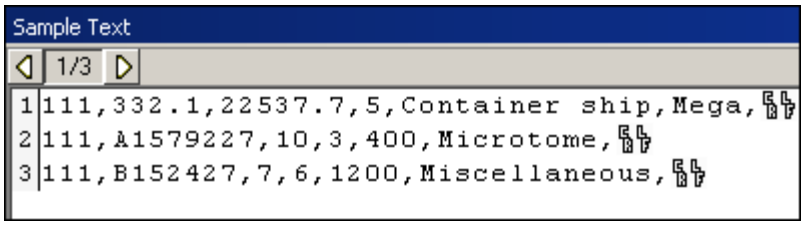
セパレーター

セパレーターとして使用される文字を指定します。デフォルトの値は「無し」(セパレーター無し).

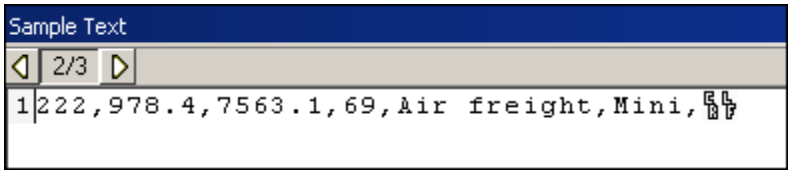
例えば、セパレーター「22」を下のテキストに対して使用すると、3つの個別のフラグメントが作成されます:



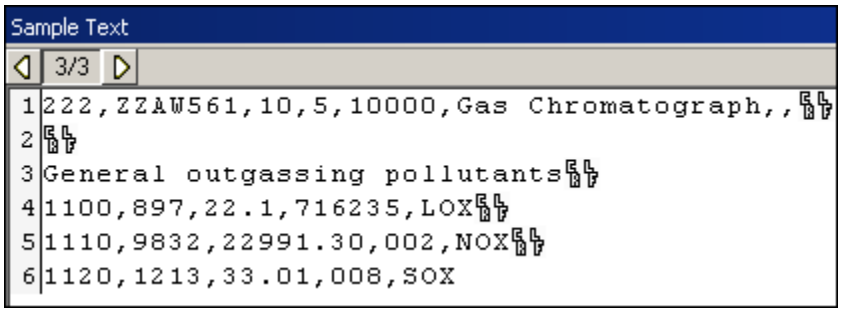
最初のフラグメントでは、セパレーター「22」を含む行までの全ての行を含む、ファイル/フラグメントから開始する全ての文字が含まれています。



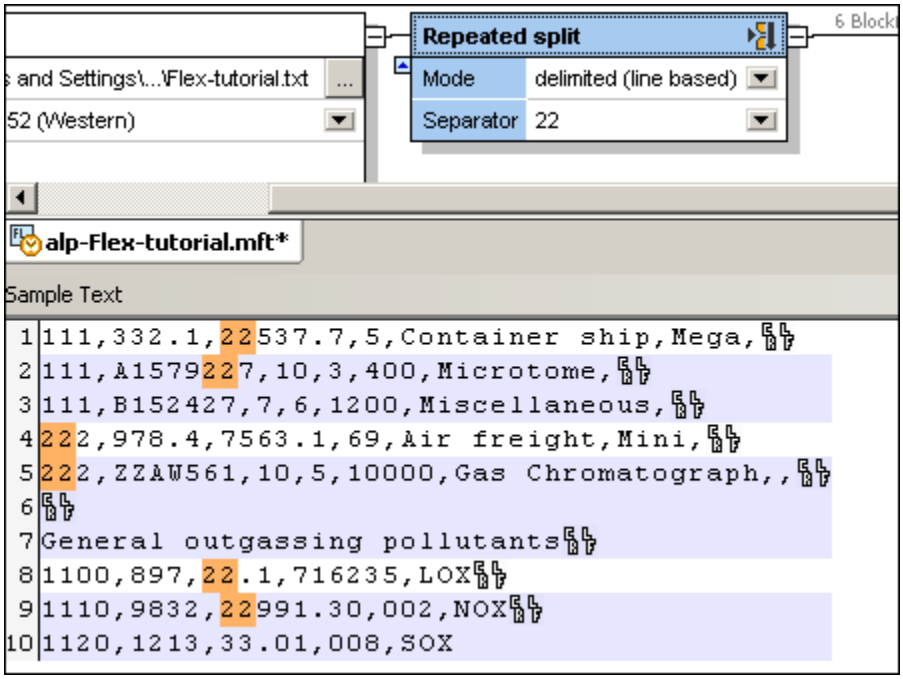
2番目のフラグメントでは、「22」の2番目の出現からのすべての文字/列から、の次の出現までが含まれています。この場合、は1行のみが条件を満たします。



3番目のフラグメントでは、「22」の3番目の出現から終わりまでのすべての文字/列が含まれます。



比較のために、区切られた（行ベース）モードとセパレーター「22」としてセパレーターが使用された場合、の結果は以下のとおりです:



行内に22 を含む行により構成される、6件のフラグメントが存在します。

6.4.5.2 1回分割する

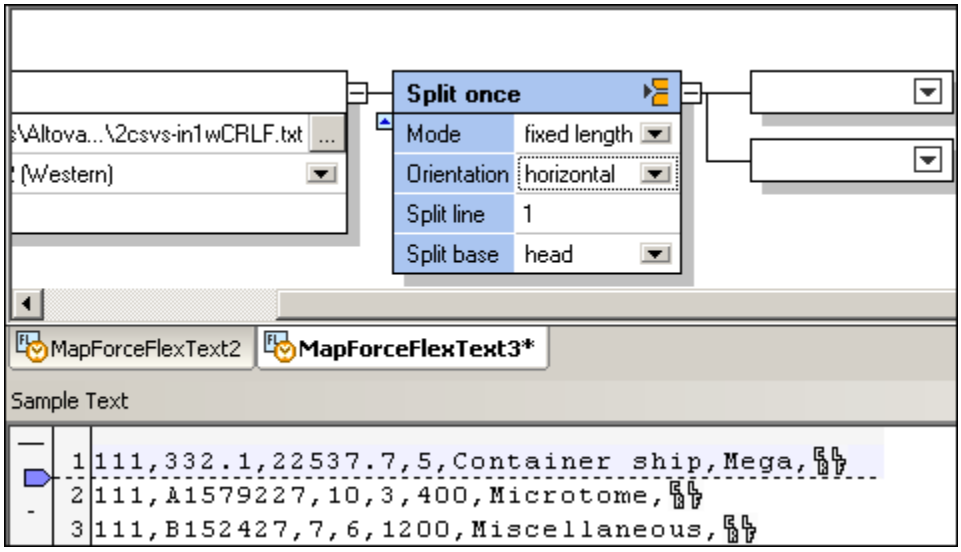
このオプションを使用すると、2つの上下に並ぶコンテナが作成されます。上のコンテナには、「1回分割する」コンテナ内で設定された条件のセットにより設定されているテキストが含まれています。下のコンテナには、テキストファイルフラグメントの残りが含まれています。「1回分割する」オプションには次のバージョンが存在します: 固定長、区切られたフロート、と区切られた行ベース。

このオプションを最初に選択すると、デフォルトのパラメーターが自動的に設定され、結果のフラグメントは両方のコンテナに表示されます。「1回分割する」コンテナが現在アクティブであり、すべての現在のレコード/列のプレビューをサンプルテキストペイン内に、表示することにご注意してください。

コンテナのデフォルトの設定:

モード	固定長
方向	水平
分割する行	1

分割ベース ヘッド



デフォルトの結果:
テキストの最初の行は、上のコンテナ内に表示されます。下のコンテナは、テキストファイルフラグメントの残りを含んでいます。

6.4.5.2.1 モード - 固定長

テキストの始め、または、終わりに相対的なある特定の行、または、列で「1回分割する」(固定長) モードを使用して、水平 または、垂直な フラグメントに分割することができます。 「1回分割する」(固定長) モードに適用することのできる設定は、以下で説明されています。

方向

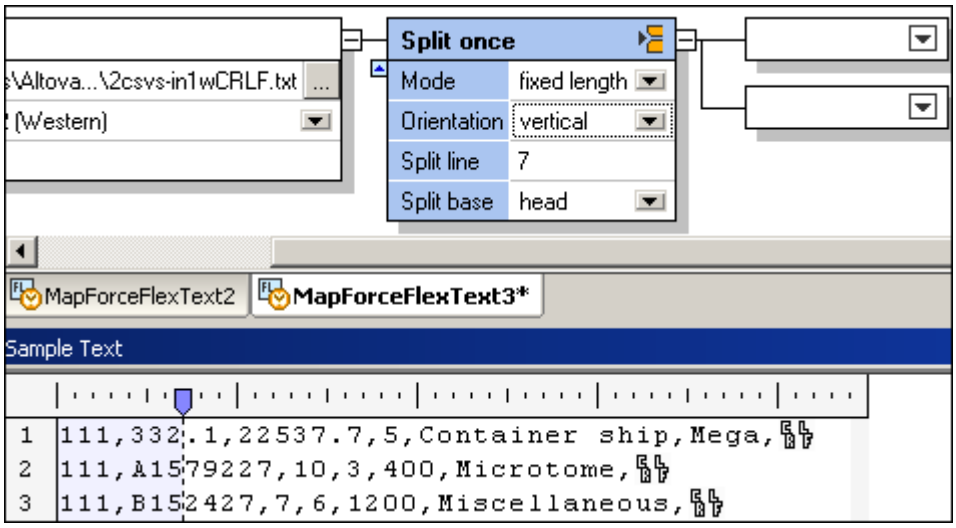
テキスト フラグメントが水平方向、または、垂直方向に分割されるかを指定します。フラグメントを複数の垂直のセクションに分割する場合は、「垂直」を選択します。デフォルトの方向は、「水平」です。

分割する行

フラグメントが2つに分割された後の行数(、または列数) を指定します。デフォルトのオフセットは、「1」です。この設定を変更するには、以下のうちの1つをおこなってください:

- **分割する行** フィールドに値を挿入します。
- タブを垂直、または、水平ルーラーにドラッグします。

(C) 2015-2021 Altova GmbH



分割ベース

ヘッド	テキストの始まりからセパレーターの最初の出現を検索します。
テール	テキストの終わりからセパレーターの最初の出現を検索します。

6.4.5.2.2 モード – 区切られた（フロート）

「区切られた（フロート）モードを使用した1回分割」モードを使用して、カスタムセパレーターを用い、テキストを2つのフラグメントに分割します。これは、通常、CR、または、LF 文字を含まないファイルに対して使用し、特定の行内の場所でフラグメントを2つに分割する場合役に立ちます。以下に注意してください：

- このモードは、セパレーターフィールドに入力する、セパレーター文字により定義される2つのフラグメントを作成します。
- セパレーター文字 は、フラグメント内に含まれていません。
- 最初のフラグメントは、ファイル/フラグメントの最初の文字とセパレーターの前の最後の文字の間のテキストとして定義されています。
- 2番目のフラグメントは、セパレーターの後の最初の文字から、ファイル/フラグメント内の最後の文字として定義されています。
- セパレーターが、ファイル/フラグメントの最初の/最後の 位置で表示される場合、2つの結果 コンテナの最初は空のままです。

「区切られた（フロート）モードを使用した1回分割」モードに適用することのできる設定は以下にリストされています。

正規表現

これは、正規表現がマッチする都度、テキストを2つのフラグメントに分割する任意の設定です。（次を参照: 正規表現を使用してテキストを分割する）。デフォルトの設定は「いいえ」です。

セパレーター

セパレーターとして使用される文字を指定します。デフォルトの値は「無し」（セパレーター無し）。

分割ベース

ヘッド	テキストの始まりからセパレーターの最初の出現を検索します。
テール	テキストの終わりからセパレーターの最初の出現を検索します。

6.4.5.2.3 モード - 区切られた(行ベース)

「区切られた(行ベース) モードを使用した1回分割」を使用して、セパレーターが行に存在するテキストを2つのフラグメントに分割します。セパレーターを含む列は、分割されず、全体が保留されます。これは、レコード区切り文字(CR/LF)を含むファイルに対して、役にたち、フラグメントを2つの個別のフラグメントに分割します。以下に注意してください:

- このモードは、セパレーターフィールドに入力するセパレーター文字により定義されている2つのフラグメントを作成します。
- セパレーター文字 は、フラグメント内に含まれています。
- 最初のフラグメントは、含まれているセパレーターまでのすべての行のテキストとして定義されています。
- 2番目のフラグメントは、ファイル/フラグメントの終わりのセパレーターを含むテキストと行として定義されています。
- 最初の/最後の行でセパレーターが出現すると、ファイル/フラグメント内に、表示されます。上のコンテナは、空のままです。

「区切られた(行ベース) モードを使用した1回分割」モードに適用することのできる設定は以下にリストされています。

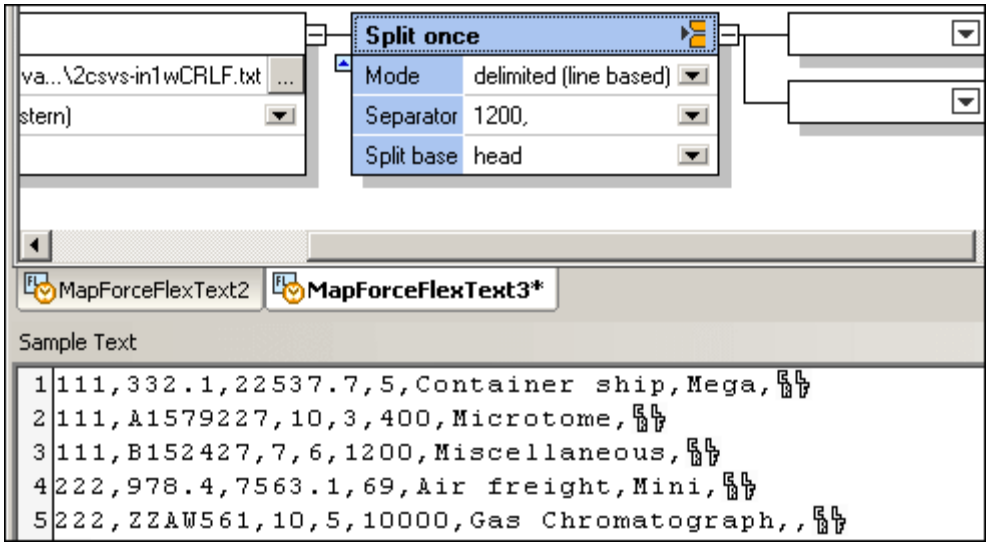
正規表現

これは、正規表現がマッチする都度、テキストを2つのフラグメントに分割する任意の設定です。(次を参照: 正規表現を使用してテキストを分割する)。デフォルトの設定は「いいえ」です。

セパレーター

セパレーターとして使用される文字を指定します。デフォルトの値は「無し」(セパレーター無し).

例えば、セパレーター「1200 」 を下のテキストに対して、使用すると、2つの フラグメントが作成されます。

- 

最初のフラグメントには、セパレーターを含む行までが含まれます。

(C) 2015-2021 Altova GmbH

```
1 111,332.1,22537.7,5,Container ship,Mega,
2 111,A1579227,10,3,400,Microtome,
```

2番目のフラグメントには、セパレーターを含む全体の行、および、終わりまでの残りのすべての行をが含まれます。

```
1 111,B152427,7,6,1200,Miscellaneous,
2 222,978.4,7563.1,69,Air freight,Mini,
3 222,ZZAW561,10,5,10000,Gas Chromatograph,,
```

分割ベース

ヘッド	テキストの始まりからセパレーターの最初の出現を検索します。
テール	テキストの終わりからセパレーターの最初の出現を検索します。

6.4.5.2.4 モード - 区切られた(行の開始)

分割が、指定されたセパレーターで開始する最初の行で発生する場合、「**区切られた(行の開始)モードを使用した1回分割**」モードを使用します。 このモードを選択すると、テキストの2つのフラグメントは以下のように作成されます:

- 最初のフラグメントには、セパレーターが存在するまでの行の全てのテキストが含まれています。
- 2番目のフラグメントには、セパレートの存在する行を含む残りのテキストが含まれます。
- 最初の行にセパレーターが存在する場合、分割はおこなわれません(最初の結果的なフラグメントは、空のままです)。

「**区切られた(行の開始)モードを使用した1回分割**」モードに適用することのできる設定は以下にリストされています。

正規表現

これは、正規表現がマッチする都度、テキストを2つのフラグメントに分割する任意の設定です。(次を参照: 正規表現を使用してテキストを分割する)。デフォルトの設定は「いいえ」です。

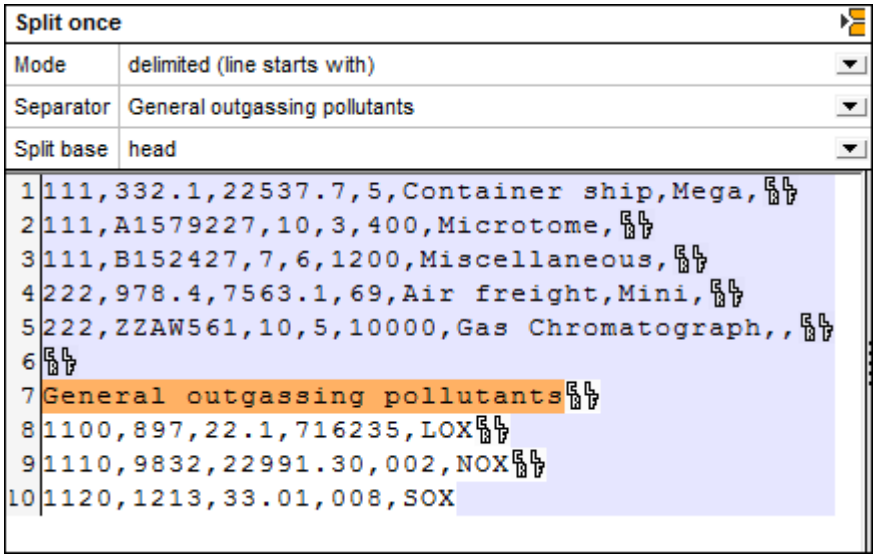
セパレーター

セパレーターとして使用される文字を指定します。デフォルトの値は「無し」(セパレーター無し)。

分割ベース

ヘッド	テキストの始まりからセパレーターの最初の出現を検索します。
テール	テキストの終わりからセパレーターの最初の出現を検索します。

例えば、下のイメージでは、指定されたセパレーター(「General outgassing pollutants」)問い足する最初の行で分割は発生します。分割ベースオプションが「テール」に設定されている場合、分割は、(このサンプルでは同じ行に存在する)終わりから開始する指定されたセパレーターから開始する最初の行で発生します。



最初の結果的なフラグメント は、以下のとおりです:

```
111,332.1,22537.7,5,Container ship,Mega,
111,A1579227,10,3,400,Microtome,
111,B152427,7,6,1200,Miscellaneous,
222,978.4,7563.1,69,Air freight,Mini,
222,ZZAW561,10,5,10000,Gas Chromatograph,,
```

2番目の結果的なフラグメント は、以下のとおりです:

```
General outgassing pollutants
1100,897,22.1,716235,LOX
1110,9832,22991.30,002,NOX
1120,1213,33.01,008,SOX
```

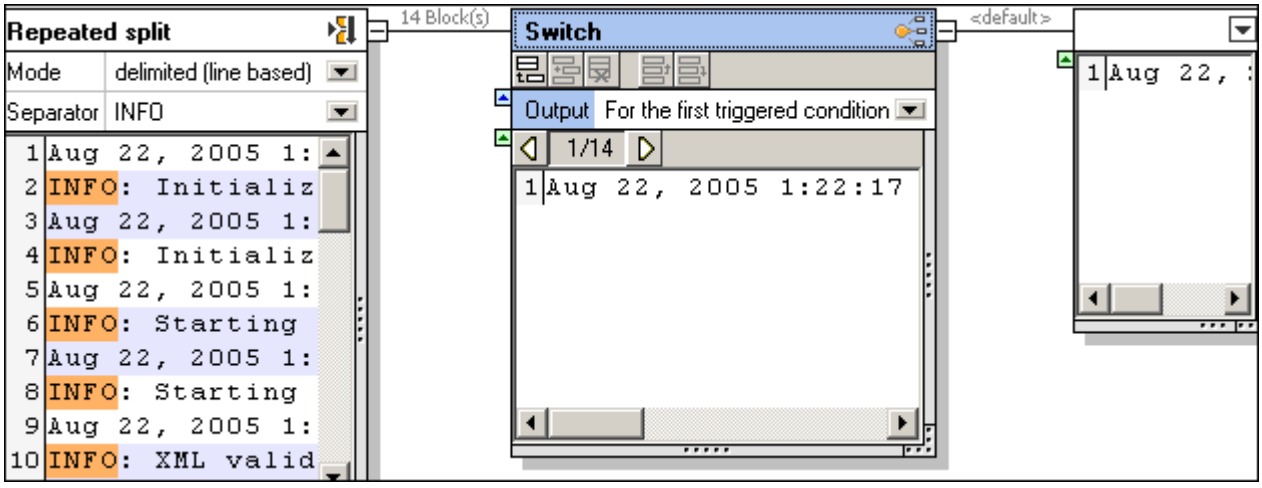
6.4.5.3 切り替え

「切り替え」オプションを使用すると、単一のテキストフラグメントのために複数のキーワード、条件を定義することができます。定義する各キーワードには、特定の条件が満たされた場合データを受け取る、自身の関連したコンテナが存在します。 条件が満たされない場合、特定のフラグメントは、「デフォルト」コンテナにマップされます。

コンテナのデフォルトの設定:
出力 最初のトリガーされた条件

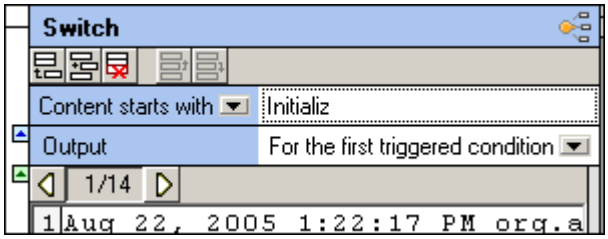
下のサンプルは、個別のプロセスが分けられ、マップすることのできる Tomcat ログファイルを処理します。「切り替え」コンテナを最初に定義する場合、**デフォルトの**コンテナのみが「切り替え」コンテナの右に表示されます。全てのデータが自動的にパスされます。

(C) 2015-2021 Altova GmbH



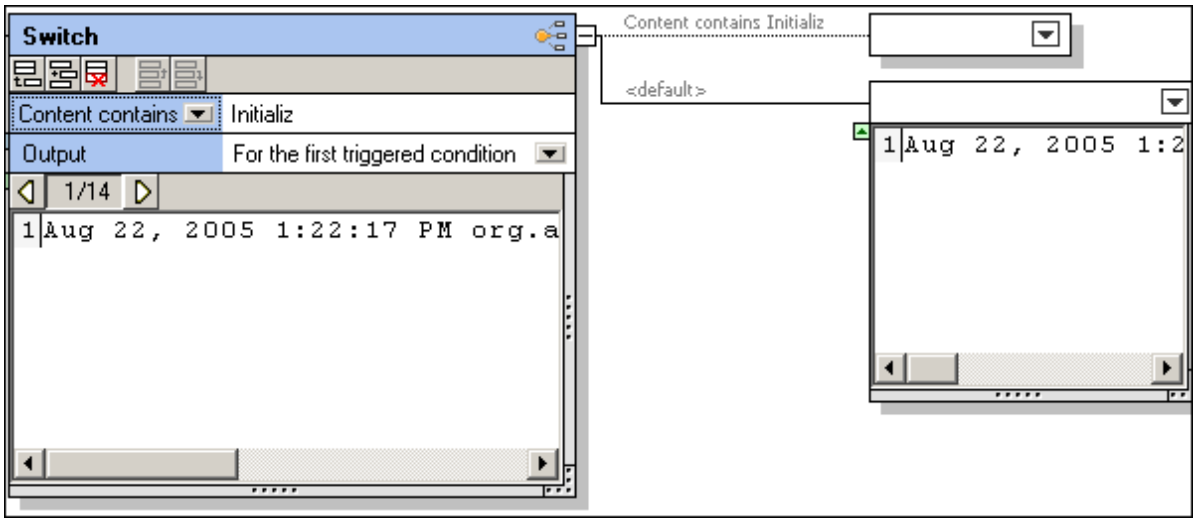
繰り返される分割 コンテナでは、区切られた(行ベース)を使用して、ログファイルからすべてのINFO のセクションを分離し、「切り替え」コンテナに入れます。

1. 追加アイコン をクリックして、新しい条件を「切り替え」コンテナに追加します。
2. 「Content starts with」フィールドをダブルクリックして、「Initializ」を入力し、Enter キーを押します。

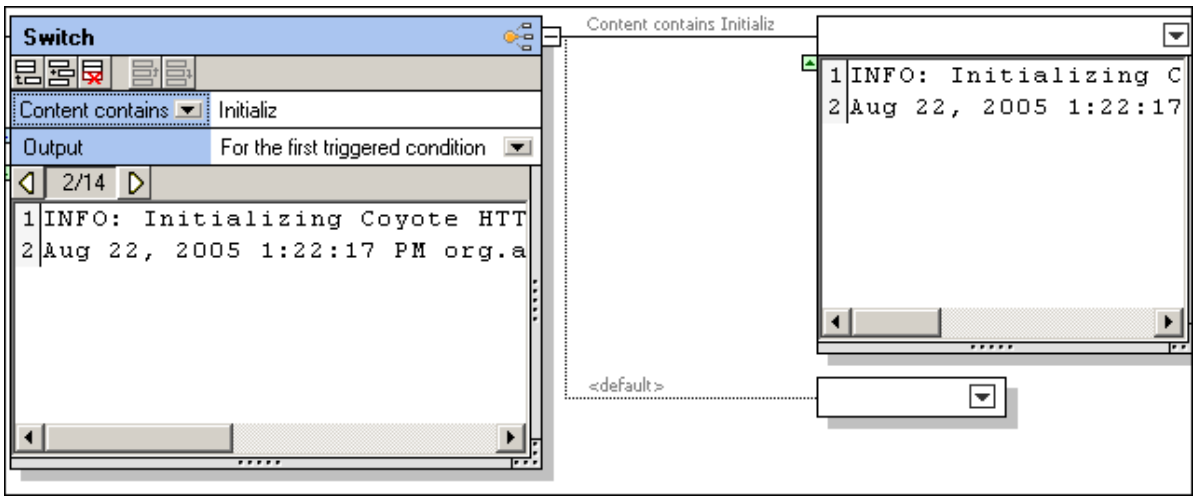


新しいコンテナが追加されます。条件が満たされる場合、データがこのコンテナに送られます。条件を満たさない場合、データはデフォルトのコンテナに送られます。

3. 「先頭 コンボボックス をクリックして、「含む」に変更します。最初の条件は、定義されており、下で結果を確認することができます。
最初のフラグメントは、「Initializ」を含みませんが、そのコンテンツは、デフォルト の コンテナに送られます。

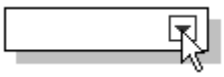


4. 次を表示するアイコン をクリックして、次のテキストフラグメントを確認します。



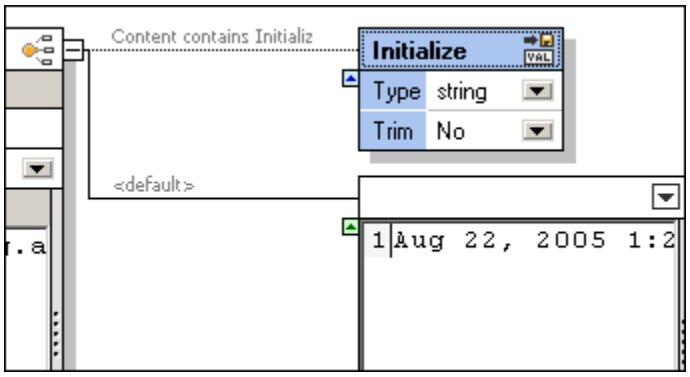
「Initializing…」フラグメントが関連したコンテナ内に表示され、デフォルトのコンテナは空です。フラグメントをステップスルーすることにより、個別のコンテナの内容をプレビューすることができます。

5. コンテナアイコンボタンをクリックして、「値として保管」を選択します。

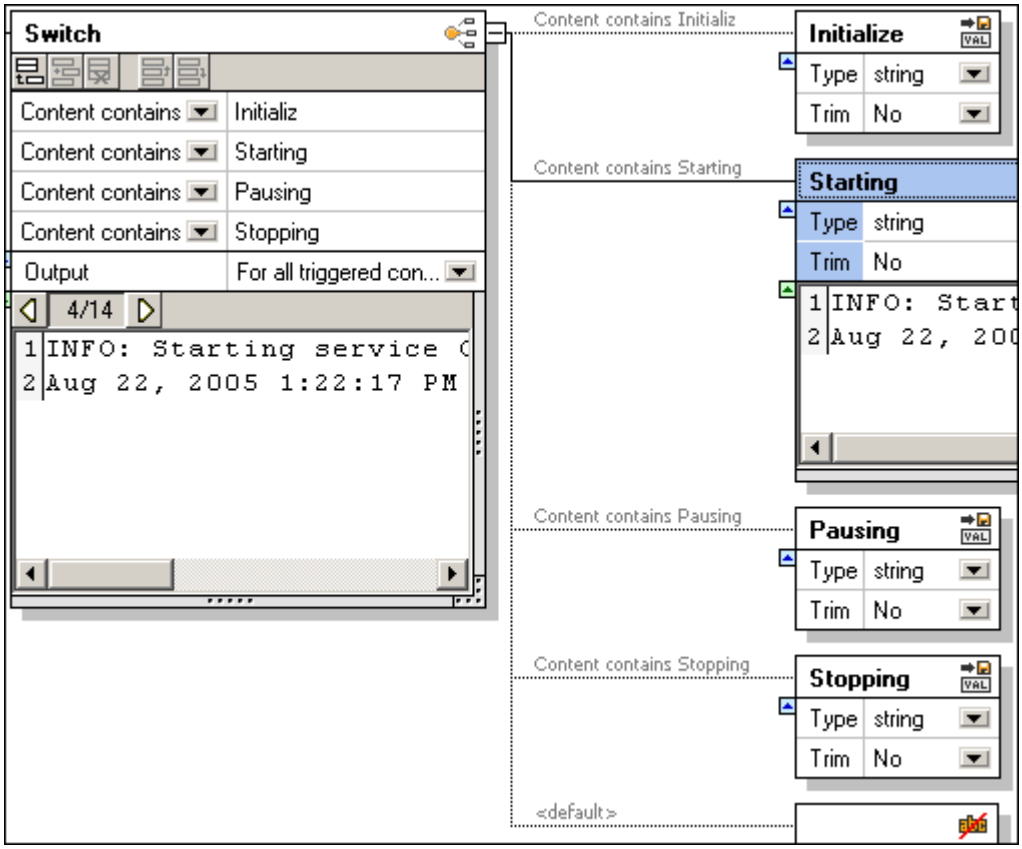


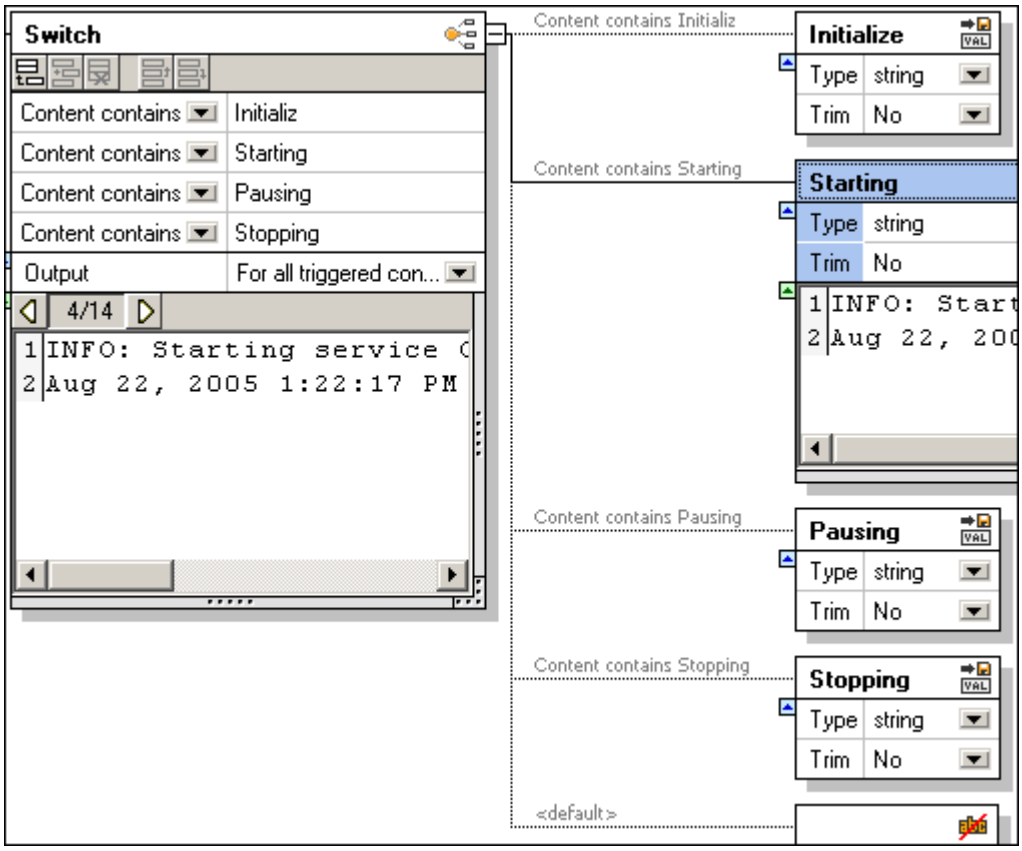
6. 「保管」タイトルバー内をダブルクリックして、テキストを変更します。例 Initialize。

(C) 2015-2021 Altova GmbH



7. 追加アイコン をクリックして、「切り替え」コンテナに新しい条件を追加します。
8. 「先頭」フィールドをダブルクリックして、「Starting」を入力し、Enter キーを押します。
   追加することのできる条件に制限はありません。例「Pausing」および「Stopping」。
   MapForce で識別しやすいように、各関連したコンテナに名前を与えます。

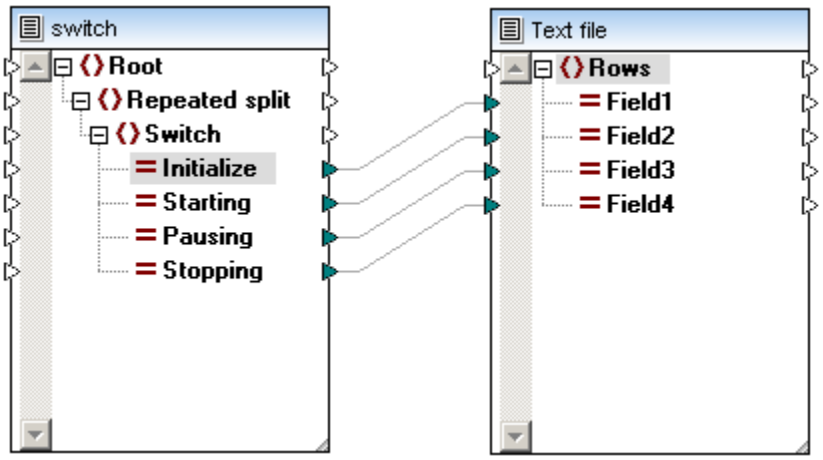




上のスクリーンショットは、4つすべての条件とブロック/フラグメント第4番目の「Starting」コンテナのコンテンツを表示しています。関連したコンテナは、MapForce コンポーネント内で簡単に識別できるように、すべて名前が付け替えられています。

上へ/下へ移動ボタン、または、 を使用して、条件リスト内で条件を上下に移動することができます。

9. テンプレートを保存して、MapForce に挿入します。



メモ 現在のフラグメント内のテキストフラグメントが条件を満たす場合、フラグメントのデータ全体が、関連したコンテナにパスされます。データは分割されず、関連したコンテナ、または、定義された条件を1つも満たさない場合、デフォルトのコンテナにルートされます。

コンテンツの先頭:

(C) 2015-2021 Altova GmbH

テキストフラグメントの最初に条件文字列が表示される場合、関連したコンテナのみにデータがパスされます。

コンテンツに含まれる内容:
テキストフラグメントに条件文字列が表示される場合、関連したコンテナのみにデータがパスされます。

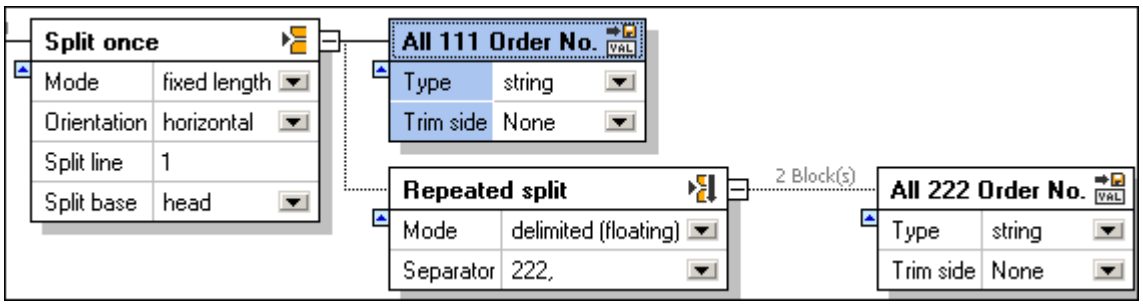
最初のトリガーされた条件のため
条件リスト内の条件 が1つでも **true** の場合、true である他の条件は、無視され、データは関連したコンテナにパスされません。

すべてのトリガーされた条件のため
条件リスト内で true のすべての条件のためにデータを出力します。複数の関連したコンテナ内の同じ データ/フラグメントが同時に発生することを許可します。これは、テキストフラグメントが、2つの条件を同時に満たすテキストを含む場合、発生する可能性があります。例 上のサンプル内の「シーケンスの開始」。

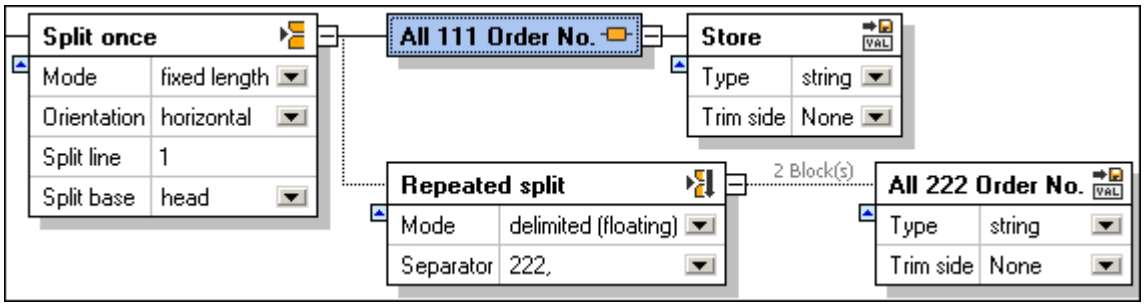
6.4.5.4 ノード

FlexText とMapForce ツリー構造に新しい階層構造レベルを追加することを許可します。次のノード/コンテナが含むデータは、以下のようにパスされます。

下のスクリーンショットでは、「All 111 Order No.」コンテナがトップのブランチの最後のコンテナです。



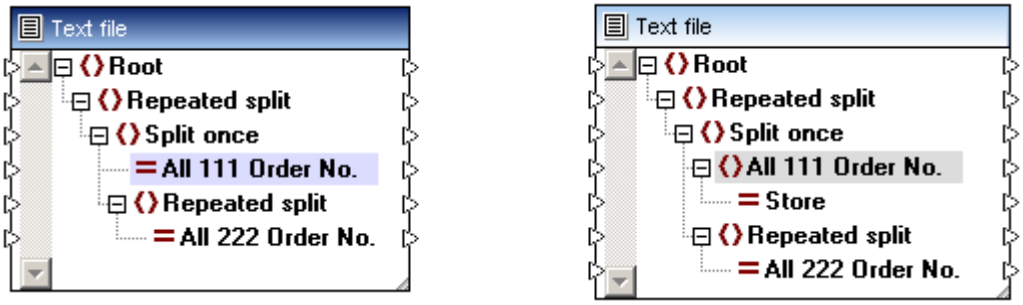
コンテナの右上をクリックして、コンテキストメニューから「Node」を選択します。



新しいコンテナが現在のコンテナの右に追加されました。

メモ 自動的に追加されたコンテナを、手動で「値として保管する」と定義することができます。

下のスクリーンショットは、MapForce 内に挿入された時のテンプレート構造を表示しています。

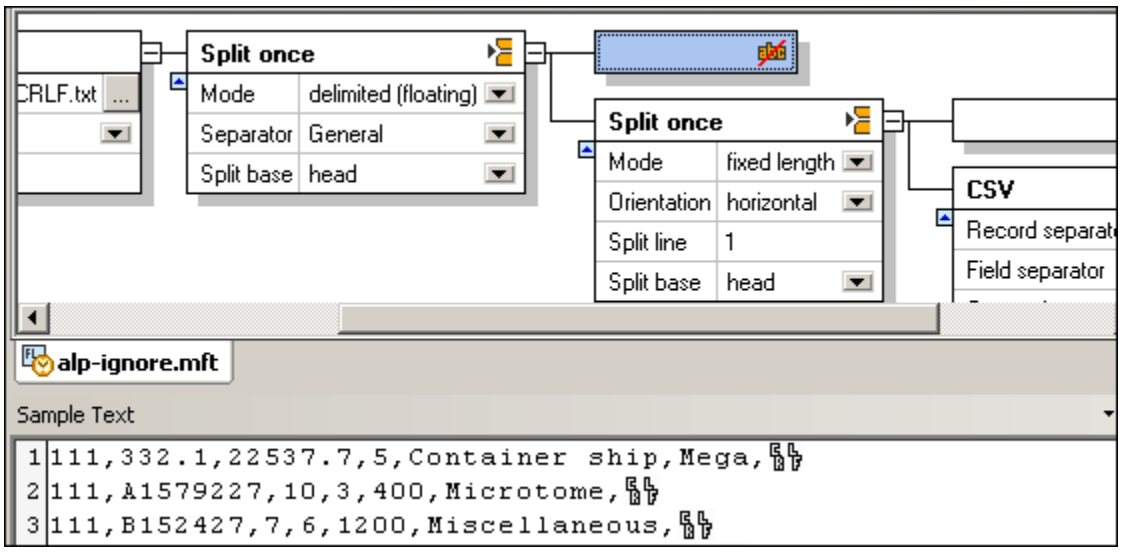


左のコンポーネントでは、新しいノードが追加される前の最初の構造が表示されています。

右側のコンポーネントは、コンポーネント 構造が変更されていることを表示しています。「All 111…」が親 アイテムになり、,新しい子アイテム「保管」が追加されました。

6.4.5.5 無視

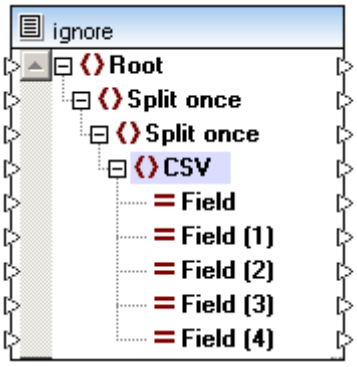
特定のテキスト フラグメントの出力を抑制することができます。データが含まれているコンテナがMapForce 内のFlexText コンポーネントで、マップすることのできるアイテムとして使用できないようにすることを意味します。



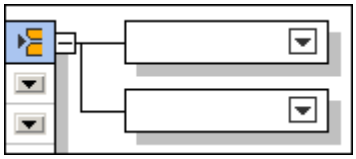
上に示される例では、アクティブなコンテナは、「無視する」に設定されています。このため、含まれるサンプルテキストは、MapForce 内のマップすることのできるアイテムとして表示されません

MapForce に挿入されたテキストテンプレートは、構造は下に表示されています。2つの「1回分割する」アイテムの間には、マップすることのできるアイテムは存在しません。

(C) 2015-2021 Altova GmbH



メモ　デフォルトの「無視する」コンテナも存在します。これらは、「1回分割する」と「繰り返される分割」などを選択すると、自動的に追加される新規のコンテナです。



これらのコンテナのコンテンツは、テンプレートが挿入された時点では、MapForce で使用することはできません。マップするためには、FlexText 内で次のコンテナオプションの1つを選択する必要があります：値として保管、CSV として保管 など。

6.4.5.6 CSV として保管(区切られた)

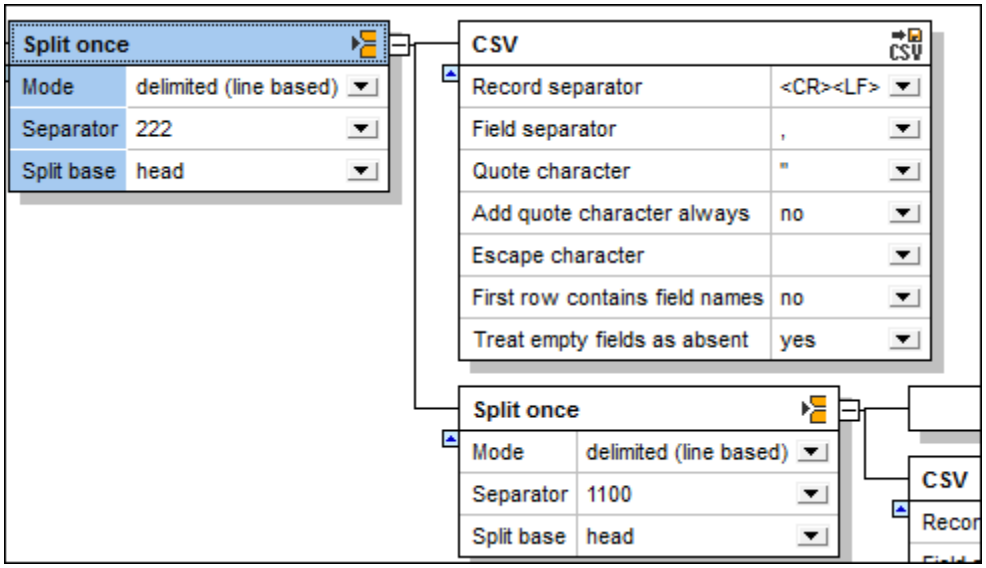
CSV として保存することにより、テキスト フラグメントを CSV テキストとして保管し、 MapForce で個別の列にマップすることができます。FlexText 内で作成することのできる CSV コンテナ/コンポーネント の数量には制限がありません。CSV コンテナには、異なるセパレーターが存在する場合があります。

サンプルテキストペインは、現在の CSV フラグメントの概要を提供し、個別のフィールド名とフィールド型を指定することを許可します。MapForce 内の FlexText コンポーネント内のマップすることのできるアイテムとして、各列は表示されます。

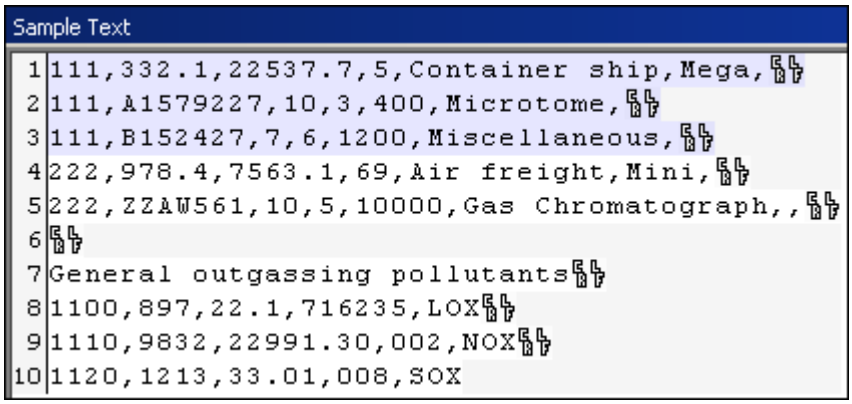
コンテナのデフォルトの設定:

レコードセパレーター	CR LF
フィールドセパレーター	,
引用 文字	"
引用文字を常に追加する	いいえ
エスケープ文字	(無し)
最初の行を含むフィールド名	いいえ
空のフィールドを不在として扱う	はい

次のサンプルは、小さなテキストファイル内のデータを2つの CSV ファイルに分割し、MapForce 内の個別の XML ファイルにマップする方法を示しています。

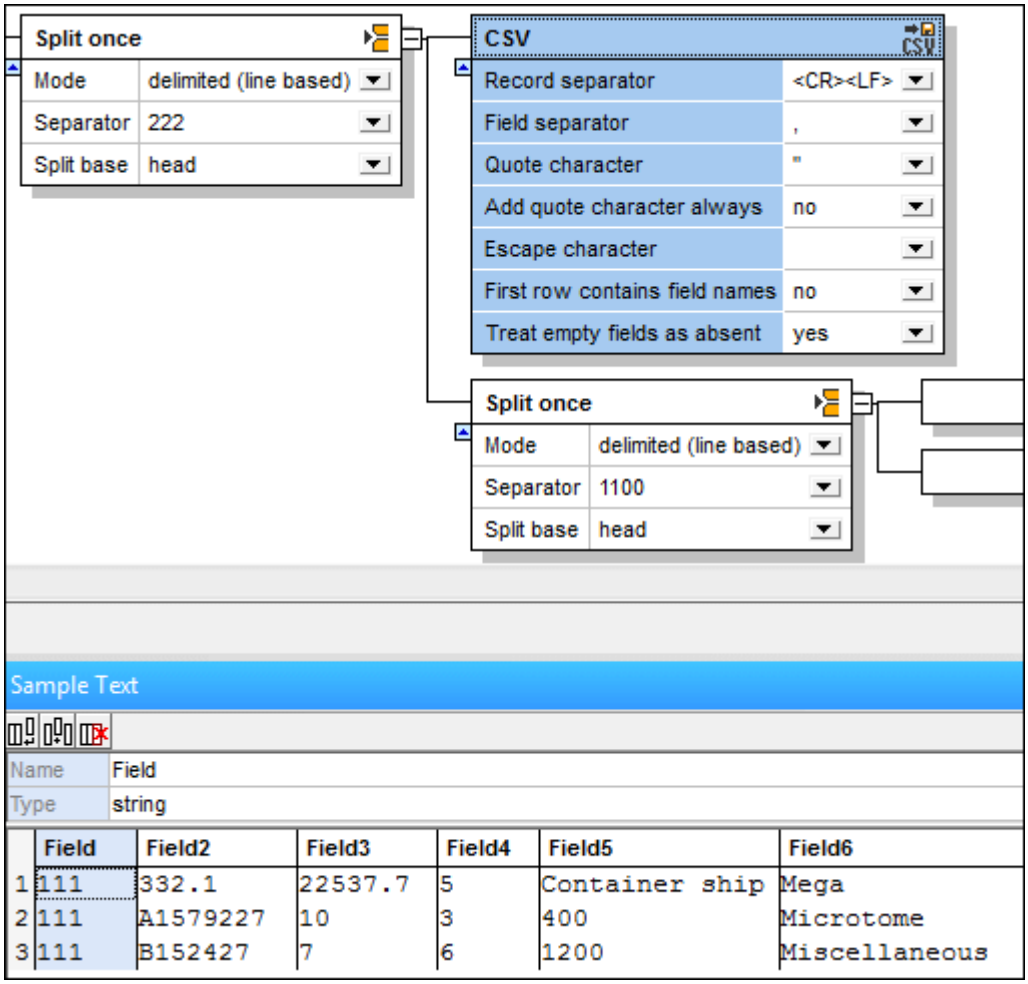


上に表示される、「1回分割する」コンテナは、2つのコンテナを作成するために使用されています。セパレーター222 を持つ、区切られた (行ベース) 関数は、これを達成するために使用されています。222 の最初の発生までのすべてのレコードは、CSV コンテナにパスされます。 111 を含むすべてのレコードから構成されている最初のコンテナは、CSV コンテナとして定義されています。サンプルテキストペインは、現在 アクティブなコンテナ「1回分割する」のコンテンツを表示します。

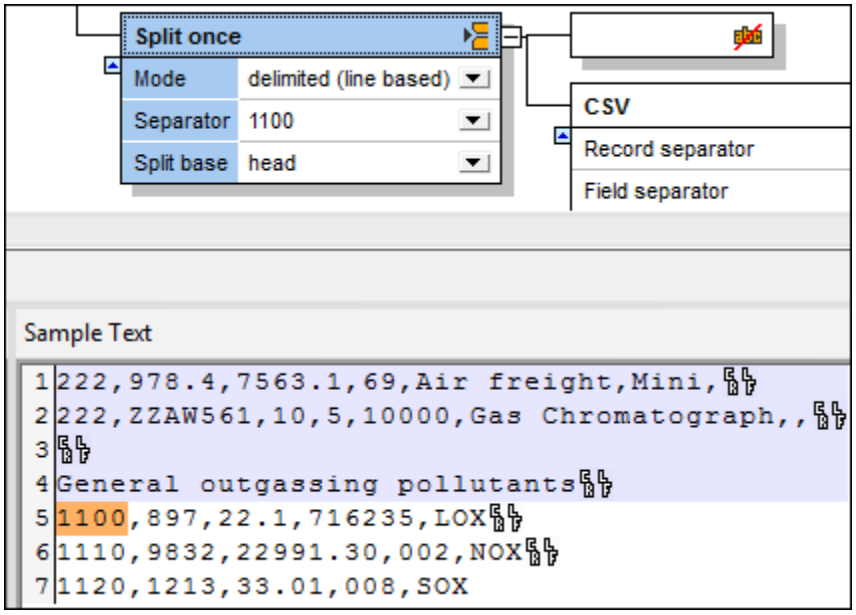


デフォルトのCSV 設定 は変更されていません。CSV コンテナクリックすると、コンテンツがテーブルのフォームで表示されます。

(C) 2015-2021 Altova GmbH



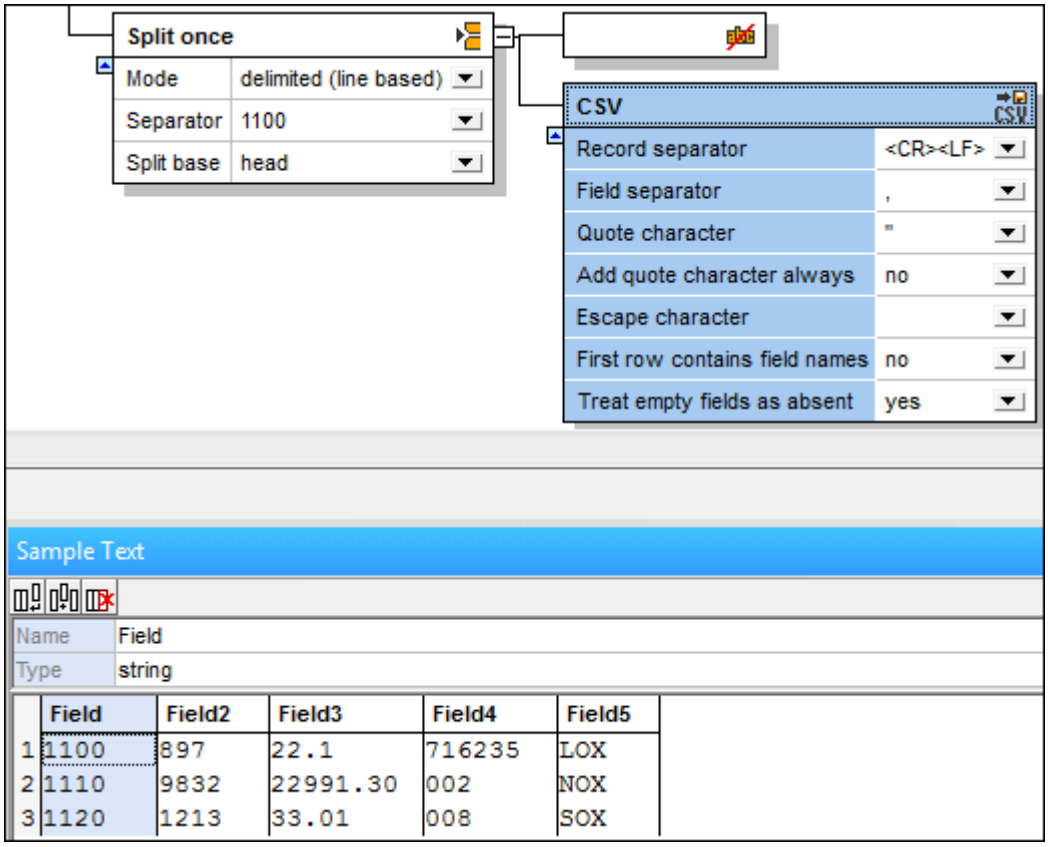
2番目のコンテナには残りのデータが保管されており、他の「1回分割する」コンテナが作成されます。これにより、2つのコンテナが作成され、そのうちの1つが2番目のCSV になります。「1回分割する」コンテナをクリックすると、現在のコンテンツが表示されます。



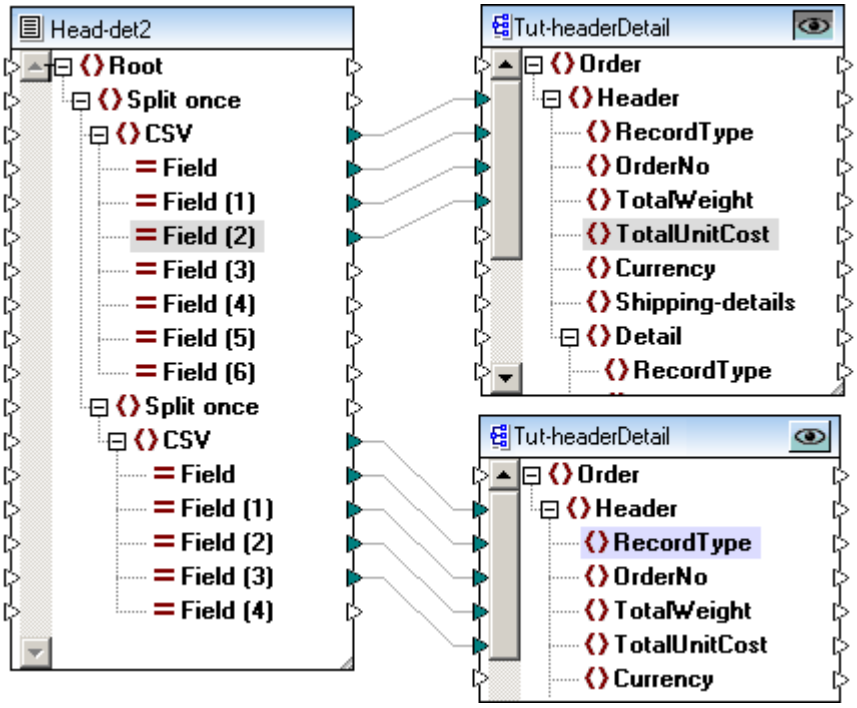
1100 をセパレーターとして使用して、区切られた（行ベース）関数は、残りのデータを2つのセクションに分割するために使用されます。

- 1100 の最初の発生までのすべてのレコードは、「無視する」 と定義することにより、マップすることのできない最初のコンテナにパスされます。
- 2番目のコンテナはCSV として定義されています。デフォルトの設定 は変更されていません。CSV コンテナをクリックすると、コンテンツは、テーブル書式で表示されます。

(C) 2015-2021 Altova GmbH



MapForce にFlexText テンプレートを挿入すると、サポートされるターゲット ファイルにマップすることができます。このサンプルでは、CSV アイテムのそれぞれが、2つの個別のXML ファイルにマップされています。



CSV セクション内のすべてのアイテムがターゲットファイルにマップされないことにご注意してください。最初のXML ファイルには、すべての111 レコード型が含まれています。

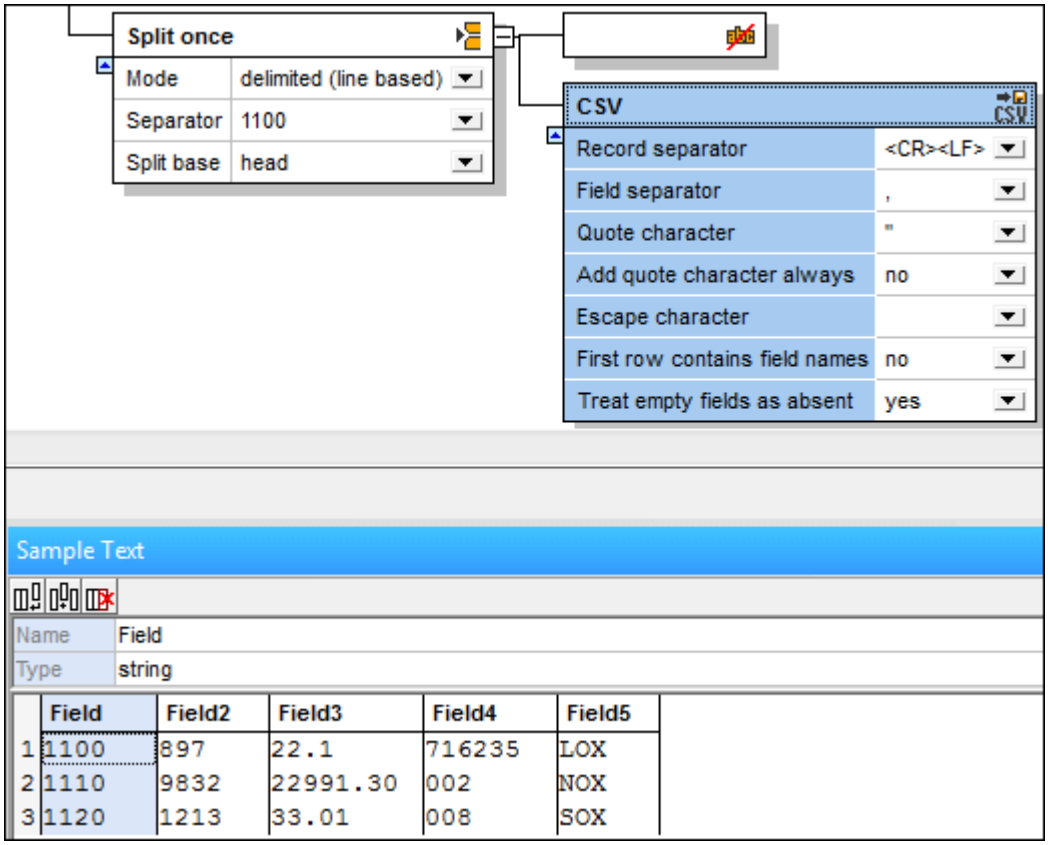
```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <Order xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3    <Header>
4      <RecordType>111</RecordType>
5      <OrderNo>332</OrderNo>
6      <TotalWeight>22537.7</TotalWeight>
7    </Header>
8    <Header>
9      <RecordType>111</RecordType>
10     <OrderNo>0</OrderNo>
11     <TotalWeight>10</TotalWeight>
12   </Header>
13   <Header>
14     <RecordType>111</RecordType>
15     <OrderNo>0</OrderNo>
16     <TotalWeight>7</TotalWeight>
17   </Header>
18 </Order>
19
```

2番目のXML ファイルには、 1100 から開始するすべてのレコードが含まれています。

(C) 2015-2021 Altova GmbH

```
<?xml version="1.0" encoding="UTF-8"?>
<Order xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  <Header>
    <RecordType>1100</RecordType>
    <OrderNo>897</OrderNo>
    <TotalWeight>22.1</TotalWeight>
    <TotalUnitCost>716235</TotalUnitCost>
  </Header>
  <Header>
    <RecordType>1110</RecordType>
    <OrderNo>9832</OrderNo>
    <TotalWeight>22991.3</TotalWeight>
    <TotalUnitCost>2</TotalUnitCost>
  </Header>
  <Header>
    <RecordType>1120</RecordType>
    <OrderNo>1213</OrderNo>
    <TotalWeight>33.01</TotalWeight>
    <TotalUnitCost>8</TotalUnitCost>
  </Header>
</Order>

```

CSV コンテナー/データの構成



サンプルテキストペイン内のフィールドをクリックすると、ハイライトされ、更に構成することができます。

- 名前 フィールドをクリックして、存在するデフォルトのテキストを編集します。
- 型 フィールドをクリックして、次のフィールド データ 型を定義します: 文字列、ブール、小数など。
- 追加アイコン をクリックして、新規フィールドを追加します。
- 挿入アイコン をクリックして、現在 アクティブなフィールドの前にフィールドを挿入します。
- 削除アイコン をクリックして、現在アクティブなフィールドを削除します。

メモ フィールド 境界をマウスでドラッグして、データを表示することができます。

引用文字を常に追加する

指定された 引用 文字が生成された CSV ファイルのすべてのフィールドに追加するかを定義することを許可します。

空のフィールドを不在として扱う

ソースファイル内の空のフィールドを定義し、ターゲットファイル内に対応する空のアイテム（要素、または、属性）を作成することを許可します。

ソースファイル内の空のフィールドのための区切り文字は、存在する必要があります。例「General outgassing pollutants,,,,」。

(C) 2015-2021 Altova GmbH

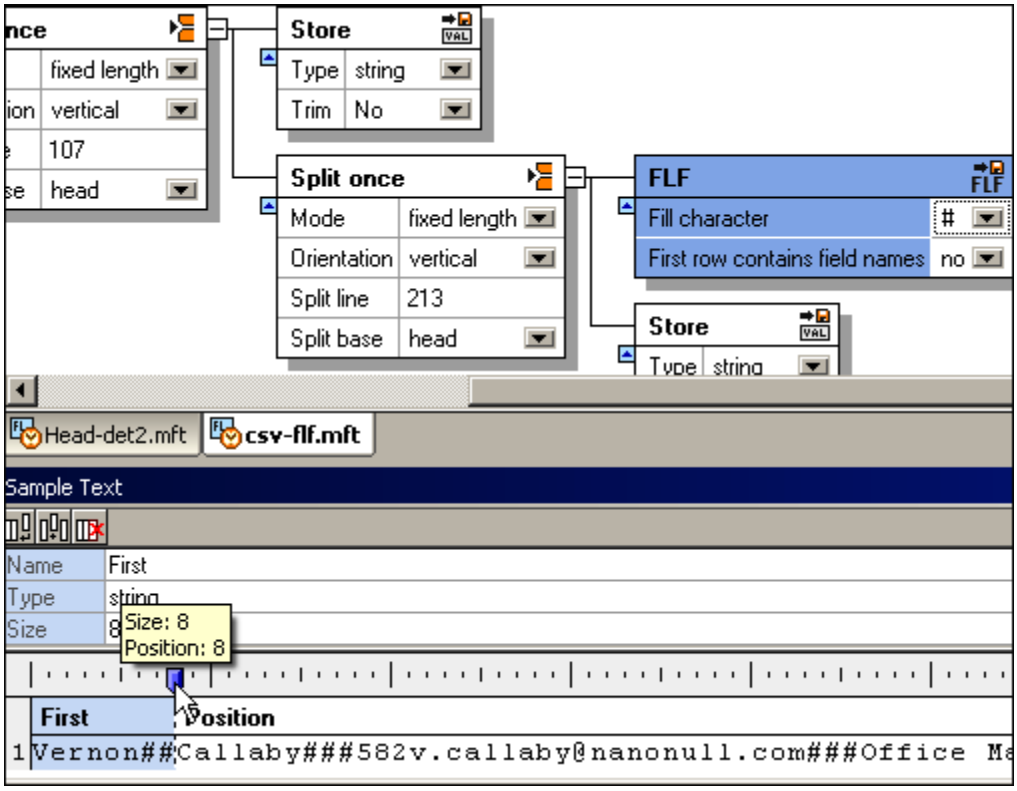
6.4.5.7 FLF として保管 (固定長)

FLF は、テキスト フラグメントは、固定長 テキストとしてテキスト フラグメントを保管し、MapForce の個別の列としてマップすることを許可します。FlexText 内で作成することのできる FLF コンテナ/コンポーネントの数量に制限はありません。各 FLF コンテナには異なるフィル文字が存在する場合があります。

サンプルテキスト ペインは、現在の FLF フラグメントの概要を提供し、個別のフィールド名、長さ、幅を指定することを許可します。MapForce 内の FlexText コンポーネント内のマップすることのできるアイテムとして、各列は表示されます。

コンテナー デフォルトの設定:

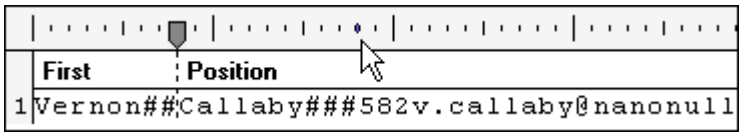
フィル文字	(無し)
最初の行を含むフィールド名	いいえ
空のフィールドを不在として扱う	はい



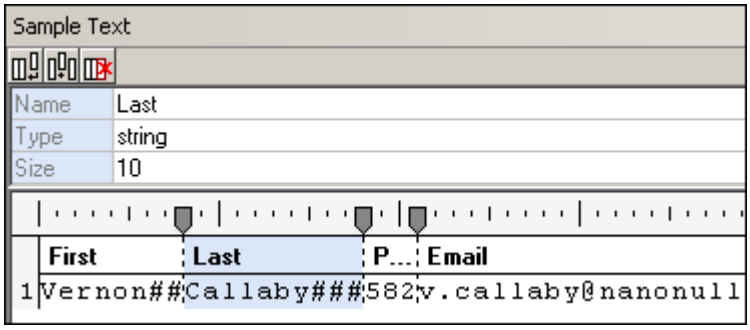
FLF コンテナ/データの構成

ルーラー上のタブ アイコンをクリックして、位置を移動します。現在の位置を示すヒントが表示されます。

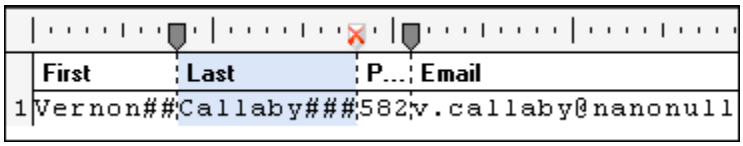
- ルーラー上のタブ アイコンをクリックして、位置を移動します。現在の位置を示すヒントが表示されます。
- カーソルの位置をルーラー上で選択すると、「ドット」で位置が示されます。新しいタブの位置をクリックして位置を設定します。



- 新しい位置を定義し、フィールドをクリックして選択し、名前 フィールド内の名前を編集します。



- フィールドを削除するには、タブ アイコンをクリックして、ルーラーからドラッグします。タブ アイコンは、このアクションの操作に成功すると変更されます。



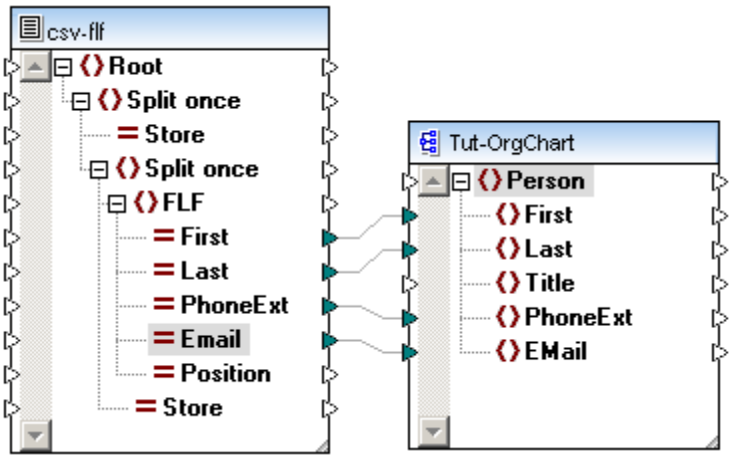
メモ タブをルーラーにドラッグすると、右側のすべてのタブが自動的に調整されます。他のタブの位置を保持するには、**Shift** キーを移動キーを押す前に押します。

サンプルテキスト ペイン内のフィールドをクリックすると、ハイライトされ、更に構成を設定することができます。

- 追加アイコン をクリックして、新しい長さの10 フィールドを追加します。
- 挿入アイコン をクリックして、現在 アクティブな、長さ 10 のフィールドの前にフィールドを挿入します。
- 削除アイコン をクリックして、現在 アクティブなフィールドを削除します。
- 名前 フィールドをクリックして、表示されるデフォルトのテキストを編集します。
- 型 フィールドをクリックして次のフィールド データ 型を定義します: 文字列、ブール、小数、など。

MapForce にFlexText テンプレートを挿入すると、サポートされるターゲット ファイルにデータをマップすることができます。このサンプルでは、FLF アイテムは、XML アイテムにマップされています。

(C) 2015-2021 Altova GmbH



オプション 空のフィールドを不在として扱う が「**はい**」に設定されている場合、ソースファイル内の空のフィールドは、ターゲットファイル内に対応する空のアイテム(要素、または、属性)を作成しません。2つのフィル文字の間にデータが存在しない場合、フィールドは不在として扱われます。

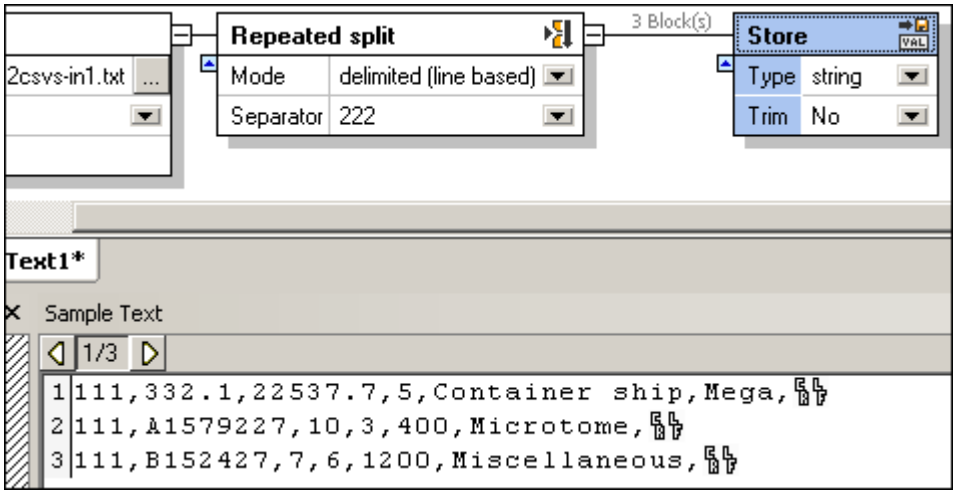
6.4.5.8 値の保管

MapForce 内でマップ可能なアイテムとしてデータを処理するコンテナを定義することができます。FlexText 内でコンテナの名前を変更しない場合、マップ可能なアイテムは「保管」という名前で表示されます。

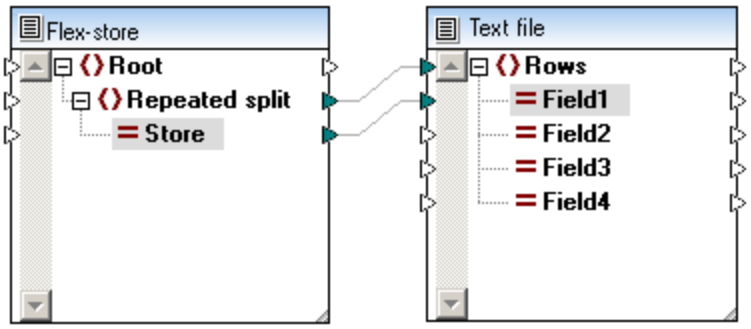
コンテナー デフォルト の設定:

型 文字列
トリム いいえ

下のスクリーンショットは、サンプルテキスト ペイン内で表示されているコンテンツがある「保管」コンテナを示しています。



このテンプレートを保存して、MapForce 内で開くと、ターゲットコンポーネント内で他のアイテムに保存されたアイテムをマップすることができます。



メモ　ターゲット テキスト ファイル内のフィールド1 アイテムは、結果をプレビューするために出力タブをクリックすると、保管アイテムにより与えられる3 つ全てのフラグメントを含みます。

型

テキストフラグメントのデータ型を定義することができます。

サイドのトリム

文字がトリムされるサイドを定義します。左、右 または両方。「はい」を選択すると、「文字 セットのトリム」オプションを有効化します。

文字セットのトリム

このテキストフラグメントからトリムする文字を定義します。入力することのできる文字数に制限は無く、フィールドをダブルクリックして行います。入力された文字がフラグメントのトリムするサイドから削除されます。

(C) 2015-2021 Altova GmbH

6.4.6 FlexText と正規表現

MapForce FlexText 内で正規表現を以下のように使用することができます:

1. (一致するテキストがセパレーターとして役割を果たすことにより)一致が検索されるとテキストコンテナを分割する。一致が検索されると、テキスト コンテナを分割します。この場合、正規表現は、明示的にアンカーされます。このため、キャレット (^) とドル記号 ($) 文字は、使用されません。例えば、3つの連続した数字に一致する場合、`^[0-9]{3}$` の代わりに `[0-9]{3}` を使用します。
2. テキストに正規表現の一致が含まれる場合、「切り替え」コンテナからテキストをリダイレクトします。

次のMapForce ターゲット言語のFlexText コンポーネント内で正規表現を使用することができます :

- Built-in (マッピングをプレビューする際に使用されます)
- Built-in (MapForce Server 実行可能ファイルを実行する際に使用されます)
- コード生成言語 (C++、C#、Java)。これらの言語では、正規表現の高度な機能は、特定の言語の正規表現の実装により異なることにご注意してください。

FlexText 内の正規表現 構文とセマンティクスは http://www.w3.org/TR/xmlschema-2/#regexs をベースにしており、MapForce コア関数 **tokenize-regexp** に類似しています。以下にご注意してください:

- 分割 条件が連続する2つのシーケンスに一致する場合、FlexText はからの結果を間に作成します (この振る舞いは、正規表現を使用しない場合でも同じようにふるまいます)。
- 正規表現 フラグ (http://www.w3.org/TR/xquery-operators/#flags) はサポートされません。

このセクションには以下のトピックが含まれています:

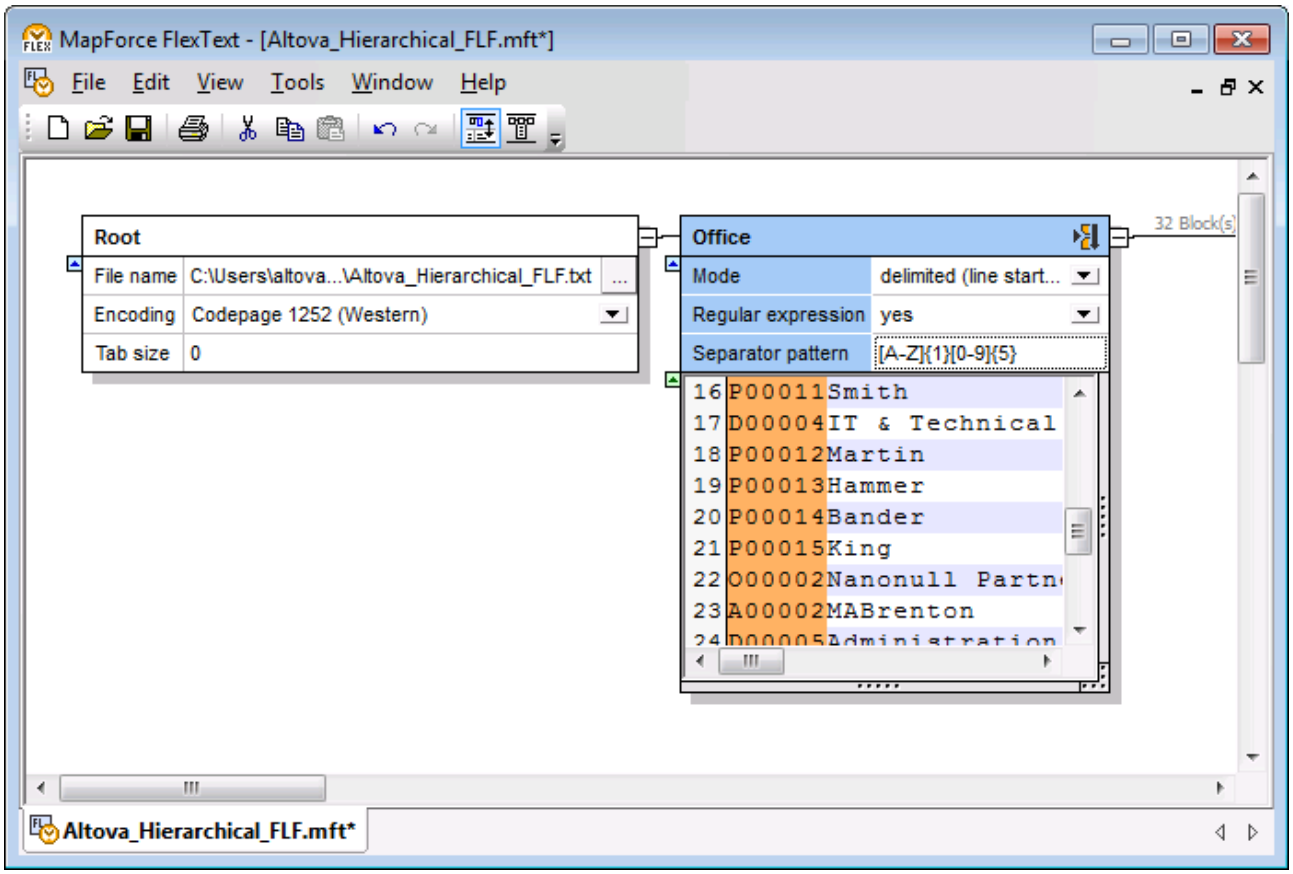
- 正規表現を使用してテキストを分割する
- 条件を変更するために正規表現を使用する

6.4.6.1 正規表現を使用してテキストを分割する

テキストを2つの、または、複数のフラグメントに分割する場合、正規表現を任意でセパレーターとして使用することができます。これは、単一または複数の連続した文字によるテキストの分割などの更に高度なオプションの代替です。

次の条件がtrue の場合、正規表現を使用してテキストを分割するオプションは、FlexText 内で使用することができるようになります:

- コンテナは、型「1回分割する」または繰り返される分割です。
- モード オプションが区切られた (フロート)、区切られた (行ベース)、または、区切られた (行の開始) に設定されている場合。



個別のテキストに対して正規表現を使用するサンプルのFlexTextテンプレート

正規表現に適用することのできるオプションは以下の通りです。

正規表現	テキスト セパレーターとして正規表現を使用するには、このオプションを「はい」に切り替えます。セパレーターとしてシンプルなテキストを使用する場合、このオプションを「いいえ」に切り替えてください（これはデフォルトの値ではありません）。
セパレーターパターン	*正規表現* オプションが「はい」に切り替えられていると、セパレーターとして振舞う正規表現を入力する箇所で、*セパレーターパターン* テキストボックスを使用できるようになります。上で説明されているFlexTextテンプレートでは、正規表現 `[A-Z]{1}[0-9]{5}` が1つのアルファベット文字とその後に続く5つの数字文字（例えば、「P00011」）が一致しています。この条件を満たすすべての発生は、プレビューでハイライトされ、テキストの分割セパレーターとしての役割を果たします。
書き込みのためのセパレーター	次の条件がtrue の場合、*書き込みのためのセパレーター* オプションには意味があります： • FlexText コンポーネントがターゲット コンポーネントの場合 • *モード* が区切られた（フロート）に設定されている場合 正規表現のマッチが検出された箇所にあるターゲットコンポーネントに書き込まれる文字列をこのフィールドに入力します。

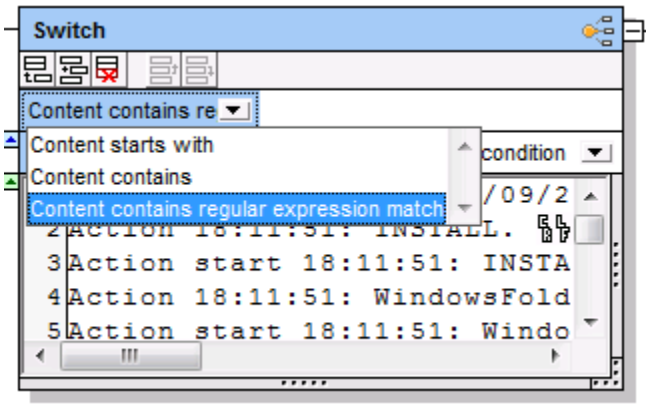
(C) 2015-2021 Altova GmbH

6.4.6.2 条件を変更するために正規表現を使用する

切り替えコンテナと作業する場合、（次を参照: 切り替え）、正規表現 の一致が含まれる場合、テキスト フラグメントをリダイレクト するために、「切り替え」 コンテナ内で条件を任意で作成することができます。 このような場合は、 C++ がターゲット 変換言語として設定されている場合を除き、テキストの始めと終わりで一致が検索されるように、 キャレット（ ^ ）とドル記号（ $ ）文字を使用することができます。 （C++ 内で キャレットとドル記号 文字は、テキスト全体の始め、または、 終わりではなく、行の始め、または、 終わりとして解釈されます）。

条件を変更するために正規表現を使用する:

1. （コンテナの右上の角をクリックし、「切り替え」をクリックして、）コンテナ 型を切り替えとして定義します
2. 「条件を追加する」（ ）ボタンをクリックして、新しい条件を追加します。
3. 条件 型を「コンテンツは正規表現 の一致を含む」 に設定します、隣接した テキスト ボックス内に正規表現を入力します。



サンプル

次に示されるデータベース ログファイルからデータをマップすると仮定します（このファイルを次の パスで見つけることができます: <**マイ ドキュメント>\Altova\MapForce2021\MapForceExamples\Tutorial\SampleDatabaseLog.txt**）。

```
Action 18:11:51: INSTALL.
Action start 18:11:51: INSTALL.
Action 18:11:51: WindowsFolder.68B7C6D9_1DF2_54C1_FF1F_C8B3B9A1E18E.
Action start 18:11:51: WindowsFolder.68B7C6D9_1DF2_54C1_FF1F_C8B3B9A1E18E.
Action ended 18:11:51: WindowsFolder.68B7C6D9_1DF2_54C1_FF1F_C8B3B9A1E18E. Return value
1.
Action 18:11:51: SystemFolder.68B7C6D9_1DF2_54C1_FF1F_C8B3B9A1E18E.
Action start 18:11:51: SystemFolder.68B7C6D9_1DF2_54C1_FF1F_C8B3B9A1E18E.
Action ended 18:11:51: SystemFolder.68B7C6D9_1DF2_54C1_FF1F_C8B3B9A1E18E. Return value
0.
Action 18:11:51: WindowsFolder.9BAE13A2_E7AF_D6C3_FF1F_C8B3B9A1E18E.
Action start 18:11:51: WindowsFolder.9BAE13A2_E7AF_D6C3_FF1F_C8B3B9A1E18E.
Action ended 18:11:51: WindowsFolder.9BAE13A2_E7AF_D6C3_FF1F_C8B3B9A1E18E. Return value
0.
Action 18:11:51: SystemFolder.9BAE13A2_E7AF_D6C3_FF1F_C8B3B9A1E18E.
Action start 18:11:51: SystemFolder.9BAE13A2_E7AF_D6C3_FF1F_C8B3B9A1E18E.
Action ended 18:11:51: SystemFolder.9BAE13A2_E7AF_D6C3_FF1F_C8B3B9A1E18E. Return value
1.
Action 18:11:51: WindowsFolder.63E949F6_03BC_5C40_FF1F_C8B3B9A1E18E.
Action start 18:11:51: WindowsFolder.63E949F6_03BC_5C40_FF1F_C8B3B9A1E18E.
```

```
Action ended 18:11:51: WindowsFolder.63E949F6_03BC_5C40_FF1F_C8B3B9A1E18E. Return value
1.
```

目標は以下のとおりです:

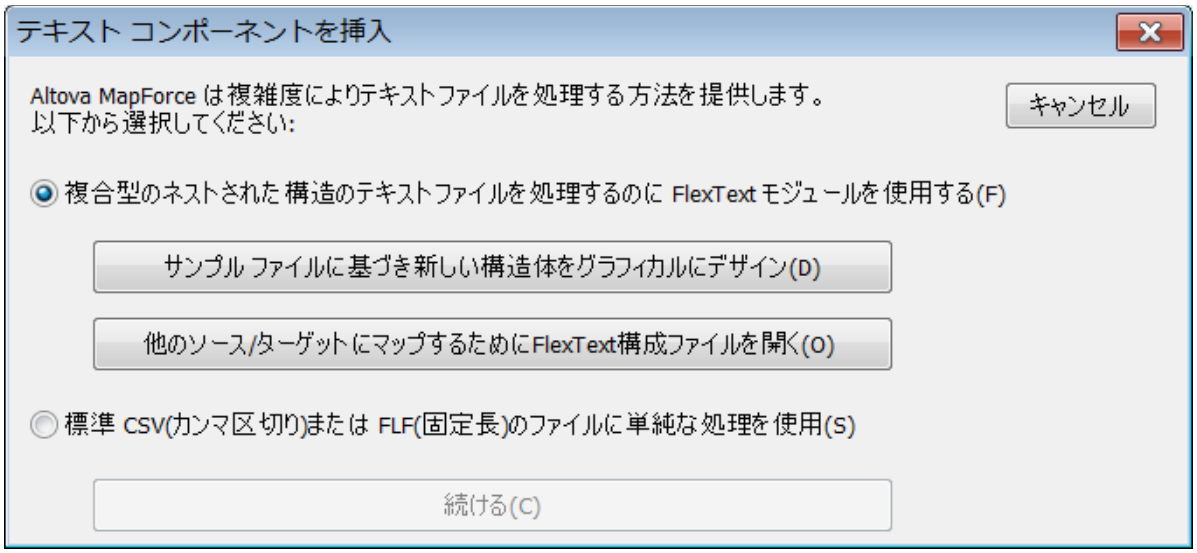
1. 戻り値が存在する箇所のすべての行を集めてリストにします。値「戻り値 1」または値「戻り値 0」、または、数値として表示される 他の 戻り値 を含む行をリストは含みます。
2. テキストが値「Action start」を含む箇所のすべての行を他のリストに集めます。

この目的を達成するために、*繰り返される分割* コンテナを使用して、テキストを個別の行に分割します。この後、*切り替え* コンテナを使用して、各行を必要とされるようにリダイレクトします。具体的には、*切り替え* コンテナ は、次に表示されるように3つの条件から構成されています:

1. 出力 A が値「Action started」を含む場合、現在の行がリダイレクトされます。型「Content starts with」の条件を使用して、そのような行を検索し、「Action started」 を値として入力します。
2. 出力 B が戻り値を含む場合、現在の行がリダイレクトされます。正規表現 戻り値 [0-9]¥. を使用してこのような行を検索することができます。この正規表現は、行が単一の数字と終止符が後に続く テキスト「戻り値」を含む場合一致を返します。 終止符がメタ文字ではなく通常の文字として扱われるように、終止符の前の バックスラッシュ（¥）がエスケープ文字としての役割を果たします。
3. 行が上の条件を満たさない場合、現在の行を 出力 C（<default>）にリダイレクトします。

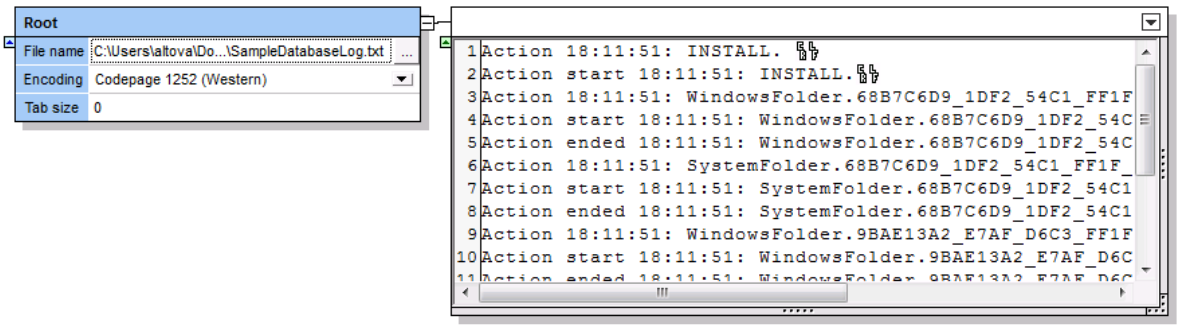
上のタスクを実行する FlexText テンプレートの作成:

1. 「挿入」メニューから、「テキスト ファイル」をクリックします。

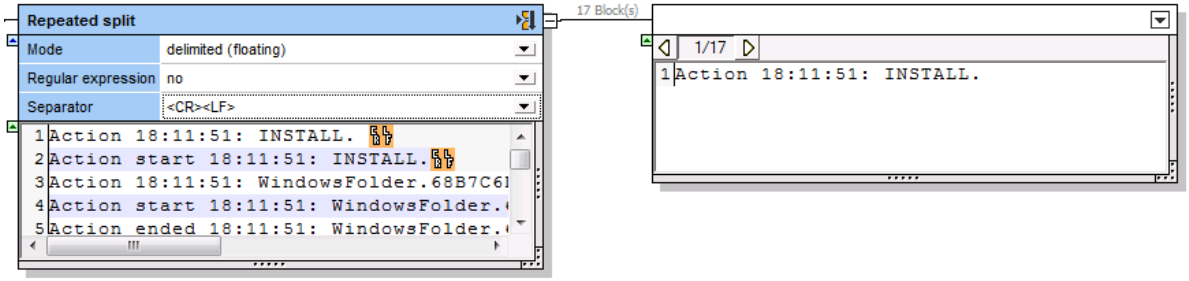


2. 「サンプルファイルに基づき新しい構造タイをグラフィカルにデザイン ...」ボタンをクリックし、 FlexText .mft テンプレートを選択するディレクトリに保存します。
3. テキスト ファイルを開くようにプロンプトされると、**<マイ ドキュメント >\Altova\MapForce2021\MapForceExamples\Tutorial\SampleDatabaseLog.txt** ファイルを参照してください。

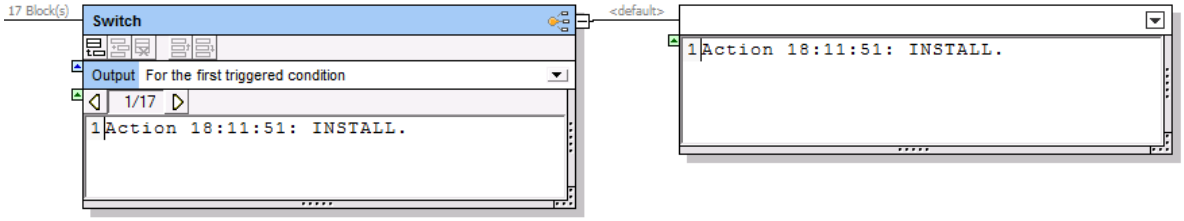
(C) 2015-2021 Altova GmbH



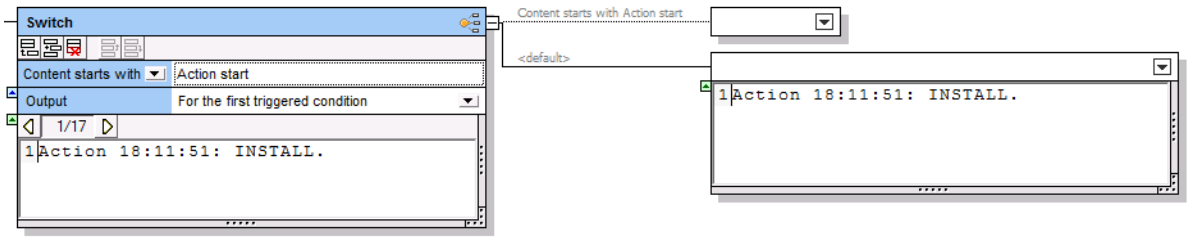
4. 出力コンテナの右上の角をクリックして、 **繰り返される分割**を選択します。分割文字としてリターンキーを使用しているため、**区切られた（フロート）モード**を選択し、 **<CR><LF>** をセパレーターとして選択します。（各行に1つの）17件のブロックにより構成されるこれにより新しい出力 コンテナが作成されます。



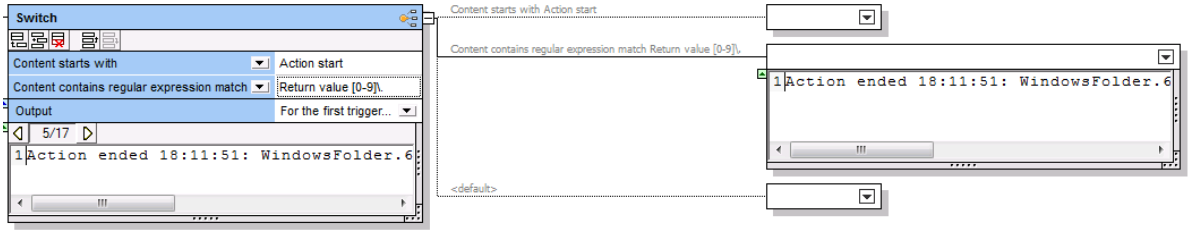
5. 新しい出力コンテナの右上の角をクリックして、**切り替え**を選択します。FlexText は、コンテナのコンテンツを置換えとして扱います。下に表示されるように、1つの<default> 変更条件が自動的に作成されます。この条件は、他の条件を満たさないのテキストを新しいコンテナにリダイレクトします。 この段階では、他の条件は定義されていません。ですから、すべてのテキストは、現在 **<default>** 出力にリダイレクトされています。



6. 「**条件の追加**」（ ）ボタンをクリックして、下に示されるように値「Action start」を持つ型「**Content starts with**」条件 を追加します。 この条件 は、「Action start」から開始するテキストを新しいコンテナにリダイレクトします。

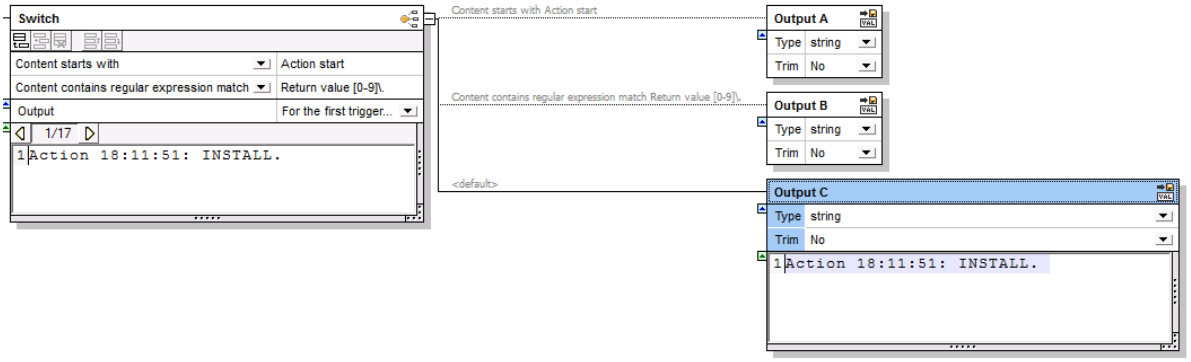


7. 条件の追加（ ）ボタンをクリックして、 戻り値 [0-9]¥. を持つ型「コンテンツは正規表現 の一致を含む」の条件を追加します。17 件中5番目のブロックへ移動する場合、ブロックには正規表現のための一致が含まれるため、この条件は、ブロックを新しいコンテナにリダイレクトされます。



条件をベースにした異なる出力へテキストをリダイレクトするために、切り替えコンテナが構成されました。3つの切り替え条件と3つの可能な出力（各条件に対してそれぞれ1つの出力）が存在します。このチュートリアルの残りのステップは、個別のテキストファイルにそれぞれの出力を書き込む方法について説明しています。

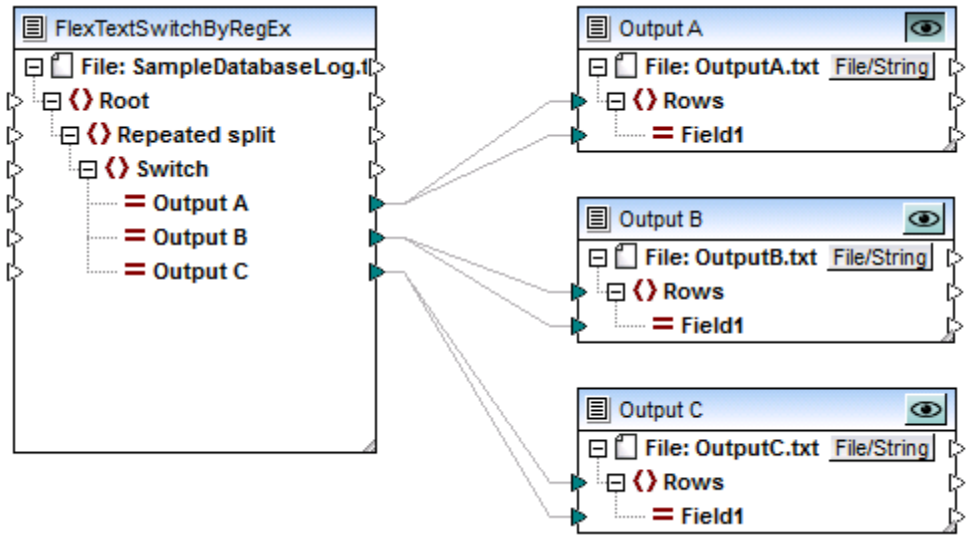
8. 出力コンテナの右上の角をクリックして、「値として保管」を選択します。更に、各出力のタイトルバーをダブルクリックし、名前を入力します: 出力 A、出力 B、と出力 C。



9. FlexText テンプレートを保存して閉じます。

必要とされる FlexText テンプレートが作成されました。次のチュートリアルファイルを開き、作業することにより、どのようにこのテンプレートが作動するかを確認することができます：<**マイドキュメント >\Altova\MapForce2021\MapForceExamples\Tutorial\FlexTextSwitchByRegEx.mfd**。

(C) 2015-2021 Altova GmbH



FlexTextSwitchByRegEx.mfd

スクリーンショット内で表示されている**FlexTextSwitchByRegEx.mfd** マッピングは、テキストデータを**<マイドキュメント>\Altova\MapForce2021\MapForceExamples\Tutorial\SampleDatabaseLog.txt** ファイルから読み取り、3つの個別のテキストファイルに変換します: 出力 A、出力 B、と出力 C。各ターゲット テキストファイルには、FlexText テンプレートで以前に定義された3つの条件の1つを満たす行のみが含まれます。特別のターゲット コンポーネントの出力を確認するには、, コンポーネントの右上の角のコンポーネントのプレビュー ( ) ボタンをクリックして、そして 出力 タブをクリックします。

6.5 EDI

Altova Web サイト: EDI マッピング

EDI (Electronic Data Interchange) は、機関とビジネスの間の電子データを有効化する一連の標準です。

MapForce は、少なくとも1つの以下の変換言語内でデータへの、またはEDI フォーマットデからの変換をサポートします: BUILT-IN、C#、C++、Java。EDI データは、マップすることができ、データベース、フラットファイル、XML ドキュメントなど MapForce がサポートするフォーマットにマッピング、またこれらのフォーマットからマッピングすることができます。各言語に適用可能なサポートされる機能に関してはサポートメモを参照してください。

以下のテーブルはMapForce 内でサポートされるEDI に関連した標準をまとめています。

標準	説明
ASC X12	ASC X12 はドキュメントのインターチェンジのための業界の標準です。MapForce は以下のバージョンをサポートします: 3040、3050、3060、3070、4010、4020、4030、4040、4041、4042、4050、4051、4052、4060、5010、5011、5012、5020、5030、5040、5050、6010、6020、6030、および6040。 **MapForce のデフォルトの ASC X12 バージョンは 6040 です。** ASC X12 コンポーネントには、EDI 検証 設定ダイアログボックス内で選択された設定によりEDI パーサーエラー情報やデータが書き込まれる「仮想」ノードがあります (EDI コンポーネントの検証を参照)。 X12 997 Functional Acknowledgement はすべてのX12 ドキュメントから生成することができます。 ASC X12 に関しての詳しい情報は、 http://www.x12.org/ を参照してください。
HIPAA X12	HIPAA はX12 EDI 5010 標準をベースにしていますが、MapForce 2011 リリース3 または以降によりネイティブにサポートされている自身の特別なバージョンがあります。 MapForce 内のデフォルトのHIPAA X12 バージョンは**HIPAA 実装スペック (TR3) のリリース A2** です。 Altova Web サイトのMapForce コンポーネントページで前のリリースをダウンロードすることができます (https://www.altova.com/mapforce/download/libraries)。
HL7	HL7 は医療アプリケーションのデータ交換のための業界の標準で、"Health Level Seven" の省略形です。MapForce はバージョン 2.2 から 2.6 をサポートします。 **MapForce 内のデフォルトの HL7 バージョンは 2.6 です。** 追加 HL7 V2.2 – V2.5.1 XML スキーマと構成ファイルのための個別のインストーラーが Altova Web サイト MapForce コンポーネントのページ (https://www.altova.com/ja/mapforce/download/libraries) で使用することができます。 XML-ベースのHL7 バージョン 3.x は、XML スキーマコンポーネントを使用して、MapForce 2021 内でサポートされています。 **MapForceExamples** プロジェクトは、HL7 V2.6 からHL7 V3 XML ファイルへマップするサンプル (HL7V260_to_HL7V3.mfd) が含まれています。 HL7 に関しての詳しい情報は、http://www.hl7.org/ を参照してください。

(C) 2015-2021 Altova GmbH

標準	説明
IATA PADIS	PADIS (Passenger and Airport Data Interchange Standard) は、EDI ドキュメントを使用した乗客と飛行場のデータのやり取りのための業界の標準です。MapForce は、以下のバージョンをサポートします: 00.1 から08.1。 **MapForce 内のデフォルトの PADIS バージョンは 08.1 です。** IATA PADIS に関しての詳しい情報は、http://www.iata.org/Pages/default.aspx を参照してください。
NCPDP SCRIPT	SCRIPT は NCPDP (National Council for Prescription Drug Programs) により開発された米国の医療分野での電子情報の交換のための標準です。詳細に関しては、次を参照してください: http://www.ncpdp.org。MapForce は、SCRIPT フォーマットと、XML、CSV、Microsoft Excel、データベースなどの MapForce によりサポートされる他のフォーマットのファイル間のマッピングデータをサポートします。サポートされる NCPDP SCRIPT バージョンは、以下のとおりです: 8.1、10.6、10.10。 **MapForce 内のデフォルトの NCPDP SCRIPT バージョンは 10.10. です。**
SAP IDocs	SAP IDocs (中間ドキュメント) ドキュメントは SAP R/3 と非-SAP アプリケーション間でビジネスデータを交換する際に使用されます。ドキュメントは異なるシステム間で交換することのできる中間データストレージです。 SAP IDocs に関しての詳しい情報は、http://help.sap.com/saphelp_nw70/helpdata/en/0b/2a6095507d11d18ee90000e8366fc2/frameset.htmを参照してください。
SWIFT	SWIFT (Society for Worldwide Interbank Financial Telecommunication 国際銀行間通信協会https://www.swift.com) は金融メッセージサービスのグローバルなプロバイダです。SWIFT MT (メッセージテキスト) は SWIFT ネットワークで使用されるメッセージ型です。MapForce は SWIFT MT データからによりサポートされる (XML、CSV、JSON、データベースなど) 他のフォーマットへのマッピング、および他のフォーマットからのマッピングをサポートします。MapForce によりサポートされる SWIFT MT リリースは以下のとおりです: • 2020 年11月(**デフォルト**) • 2019 年 11 月 フォーマットが XML ベースのため SWIFT MX /ISO 20022 フォーマットかへの、またはフォーマットからのデータのデータのマッピングは暗示的です。
TRADACOMS	TRADACOMS (Trading Data Communications) は、小売業で使用される英国特有の電子データインターチェンジの標準です。 MapForce は、ベース TRADACOMS 仕様を Article Numbering Association (ANA) UK により出版された GS1 UK (https://www.gs1uk.org)として知られる 1993 年 1 月に出版された"電子データインターチェンジのための標準の TRADACOMS マニュアル" 内で指定されたとおり実装します。またはその他の TRADACOMS バージョンは、MapForce は、新規のメッセージの型、データ要素、コードの値、を構成ファイルによりカスタム化することができます。
UN/EDIFACT	UN/EDIFACT (United Nations rules for Electronic Data Interchange for Administration, Commerce and Transport) は電子ドキュメントのインターチェンジの標準です。詳細に関しては、次を参照してください: http://www.unece.org/cefact/edifact/welcome.html。MapForce は、D93.A - D.20A ディレクトリ内に含まれるメッセージへをサポートします。

標準	説明
	MapForce 内のデフォルトの UN/EDIFACT バージョンは D.20A です。 EDI マッピングに必要な構成ファイルを含む D93.A からの全てのバージョンのための個別のインストーラーは Altova MapForce ライブラリページ (https://www.altova.com/ja/components_mapforce.html) でご利用いただけます。

(上記のテーブルで太字でフォーマットされている) MapForce を使用してデフォルトでインストールされた EDI 構成バージョンは新しいリリースごとに変更されます。例えば、UN/EDIFACT 標準のバージョンが表示されると MapForce 内で次のリリースのためのデフォルトのバージョンになります。デフォルト EDI バージョンは生産ではなくトライアルとデモの目的に主に適切であることに注意してください。

To design an EDI マッピングを生産のためにデザインするには、Altova W eb サイト (https://www.altova.com/ja/mapforce/download) から EDI 構成ファイルの特定のバージョンをダウンロードしてインストールしてください。インストール後、インストールされた EDI バージョンがデフォルトなバージョンに加え MapForce 内で使用可能になると、EDI コンポーネントをマッピングに追加する を参照してください。

マッピングエリアに追加した、単一の EDI コンポーネントのために、MapForce は、単一の標準 EDI リリースの複数のメッセージの型を処理することができます。例えば、これにより 1 つ以上のメッセージ型を含むデータを交換可能なファイルから読み込むことができます。また、異なったメッセージの型を含む複数のインスタンスファイルからデータを処理することも可能です。上記のオペレーションは基になる仕様がサポートされる場合のみ意味を成します。

所属機関が EDI 仕様を使用する場合、カスタム化はすでに MapForce によりサポートされています。また、構成ファイルを使用して、カスタム EDI 仕様に対して MapForce が対応することもできます。例えば、カスタムメッセージ、セグメント、データ要素、コードリストなどを追加 (既存のものを変更) することができます。ですから、MapForce により既にサポートされている EDI に関連したディレクトリに類似したカスタム EDI コレクションを作成することができます。詳細に関しては、EDI 構造のカスタム化 と EDI 検証のカスタム化を参照してください。

6.5.1 EDI 用語解説

次のリストでは、EDI に関連した標準と共に使用される一般に使用される用語について説明されています。

メッセージ

メッセージとは、ビジネスドキュメントを表す集合のことです。ASC X12 内では、メッセージは、"トランザクション" と呼ばれます。メッセージは、"セグメント" と呼ばれる複数のビルディングブロックから構成されます。MapForce では複数種類のメッセージが単一の EDI コンポーネントにてサポートされます。(次を参照: EDI コンポーネントをマッピングに追加する)。

セグメント

セグメントとは、メッセージに含まれる単一の「レコード」のことです。セグメントは、2 または 3 文字からなる ID により識別されます。関連する要素のグループによりセグメントタグ (ANSI X12 におけるセグメント ID) が構成されます。トランザクションのセグメントは、必須または条件 (オプション) セグメントとして定義することができます。

データ要素

セグメント内にある個々のデータフィールドのことです。要素はフィールドとして考えることができ、例えば 1 種類のデータ、名前、または住所などを含めることができます。要素は更に、コンポーネントの要素やサブ要素からなる複合要素へ分けることができます。

(C) 2015-2021 Altova GmbH

セパレーター

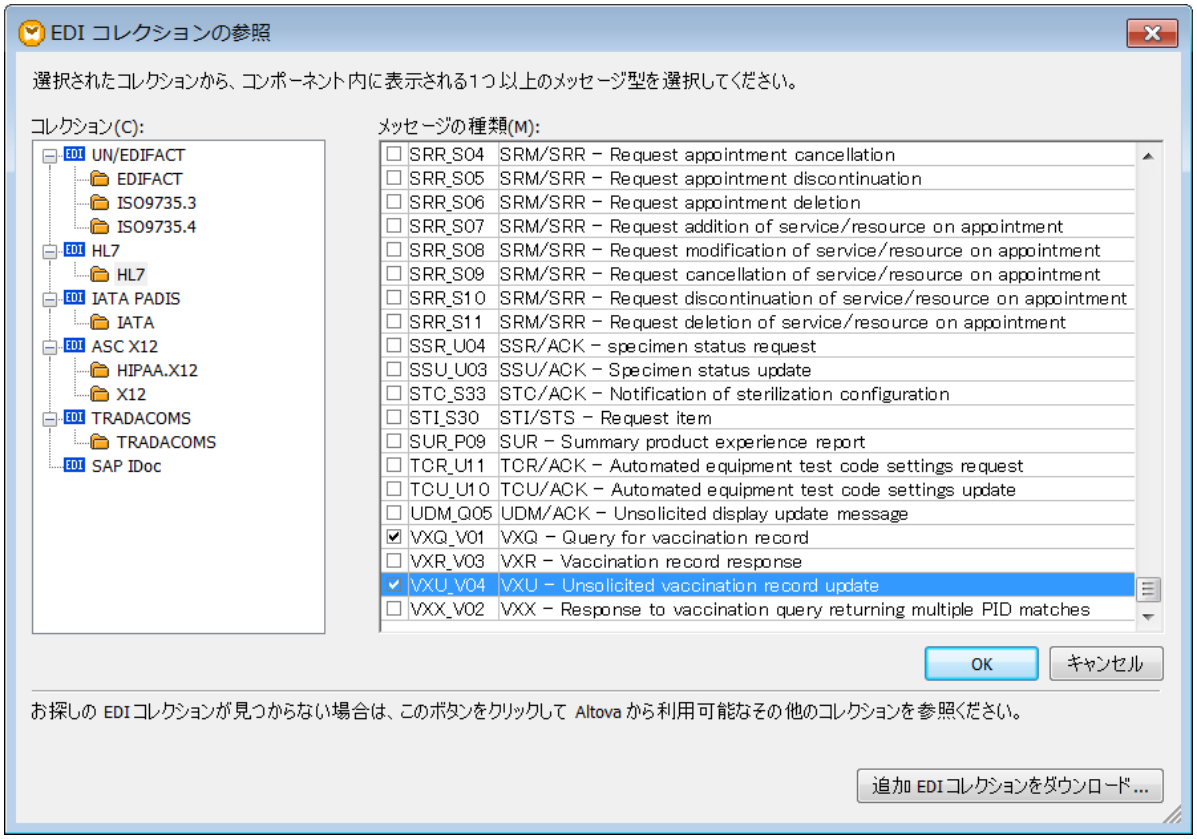
要素は「セパレーター文字」で区切られます。セパレーターは、通常テキストフィールドで発生するASCII 文字です（例えば、コロン、アスタリスク、パイプ、バー、アンパサンド など）。EDI コンポーネント型によりセパレーターは異なります。デフォルトでは、各 EDI フレーバーのために適用することのできる標準のセパレーターを MapForce は認識します。しかしながら、セパレーターは必要に応じてカスタマイズすることができます（次を参照: EDI コンポーネント設定）。

6.5.2 EDI コンポーネントをマッピングに追加する

EDI ファイルからデータを、またはEDI ファイルへデータをマップするには、EDI コンポーネントを最初にマッピングに追加する必要があります。EDI コンポーネントは、マッピング上のEDI メッセージの構造を表しています。EDI コンポーネントがマッピングに追加されると、MapForce によりサポートされるコンポーネント型と同様にマッピングの接続を追加することができます（次を参照: コンポーネントと作業する）。必要であれば、複数のメッセージを含むためにEDI コンポーネントを構成することができます。これにより、複数の入力または出力ファイルを動的に処理で説明されているように、複数のEDI ファイルからデータ（各ファイルから1つのメッセージ）を読み取る、または同じファイルから異なるメッセージを処理することができます。

EDI コンポーネントをマッピングに追加する:

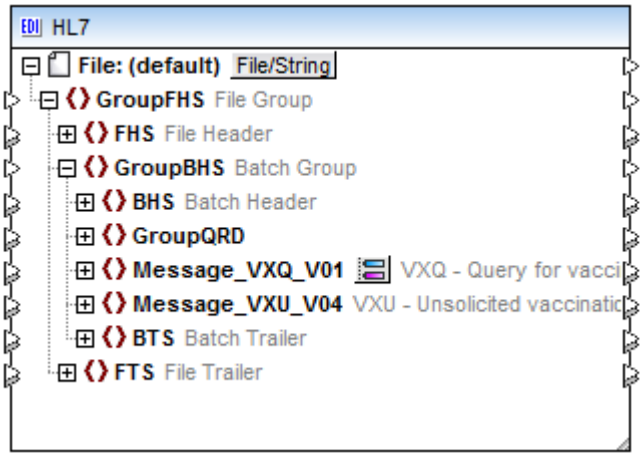
1. 「挿入」メニューから、「EDI」をクリックします。
1. EDI コレクションをクリックし、含むメッセージの型の横のチェックボックスをチェックします。複数のメッセージを、対応するチェックボックスを選択して行います。例えば、下のイメージ内では、HL7 コレクションの2件のメッセージがマッピングに追加されています。



2. 「OK」をクリックします。サンプルEDI ファイルを提供するように問われます。

3. EDI コンポーネントからデータを読み込む場合、サンプルEDI ファイルを参照します。それ以外の場合、「スキップ」をクリックします。

選択済みのメッセージの構造を含むコンポーネントがマッピングに表示されます。追加メッセージを追加する場合、またはメッセージを削除する場合、最初のメッセージの横に表示される ボタンをクリックします。



2 つのメッセージを含むサンプル HL7 マッピングコンポーネント

このEDI コンポーネントからマッピング接続を描くことができます。複数のEDI フレーバーからのマッピングを表す部のMapForce マッピングデザイン(.mfd) ファイルは、<マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥フォルダー内で確認することができます。

フォーマット	マッピングサンプル
ASC X12	• **X12_To_Xml_Order.mfd** • **Xml_To_X12_Order.mfd**
HIPAA X12	• **HIPAA_837D.mfd**
HL7	• **HL7_MultiMessageTypes.mfd** • **HL7V260_To_HL7V3.mfd**
IATA	• **IATA_FlightInformationReport.mfd**
SAP IDoc	• **IDOC_Order.mfd**
UN/EDIFACT	• **EDI_Order.mfd** • **EDI_Order_DB.mfd** • **EDI_OrderReverse.mfd**

6.5.3 EDI マッピングの実行

BUILT-IN 変換言語にマッピングが設定されている場合にEDI コンポーネントを含むマッピングは通常サポートされています。これによりMapForce 内で直接マッピングをプレビューできるだけでなくMapForce Server により実行されるためのサーバー環境にデプロイ可能になります。XML への クイック(非標準) EDI 変換が有効化されており、エンジンを必要とする箇所のマッピングに関しては EDI から XML を生成を参照してください。

(C) 2015-2021 Altova GmbH

EDI-関連のフォーマットの大半では、更にマッピングコードをC++、C#、Java で生成することが可能です。詳細に関しては サポートメモを参照してください。

MapForce 内でEDI マッピングがデザインされると以下の方法で実行することができます:

- 出力 タブをクリックしてMapForce 内で直接データ変換の結果をプレビューすることができます。マッピングにより生成される出力は、MapForce とターゲットコンポーネントの構成に従い、希望する場所に希望する名前で保存することができます。
- MapForce Server を介してAPI 呼び出し、またはコマンドラインインターフェイスを介して、スタンドアロンでインストールされたMapForce Server を使用してサーバーマシン(Windows、Linux、macOS) 上でマッピングを実行することができます。詳細に関してはMapForce Server 実行ファイルにマッピングをコンパイルするを参照してください。
- MapForce Server がFlowForce Server の管理下で作動するHTTP (またはHTTPS) を介してマッピングをサーバーマシン(Windows、Linux、macOS) にデプロイすることが可能です。このようなセットアップで、マッピングを計画されたまたはオンデマンドのジョブ、またはWeb サービスとして実行することができます。FlowForce Server にマッピングをデプロイするを参照してください。
- C++、C#、またはJava 言語をサポートするマッピングに関しては、マッピングからプログラムコードを生成することができます。生成されたプログラムは、マッピングがMapForce 内でプレビューされた状態と同じ結果を生成します。スタンドアロン、または既存のコードに統合して実行することができます。コードジェネレーターを参照してください。

EDI コンポーネントに適用することのできる設定(ソース、またはターゲット)を構成するには、コンポーネントヘッダーを右クリックし、コンテキストメニューから[プロパティ]を選択します。EDI コンポーネント設定を参照してください。他のコンポーネント型に適用することのできる設定に関しては、コンポーネント設定の変更を参照してください。

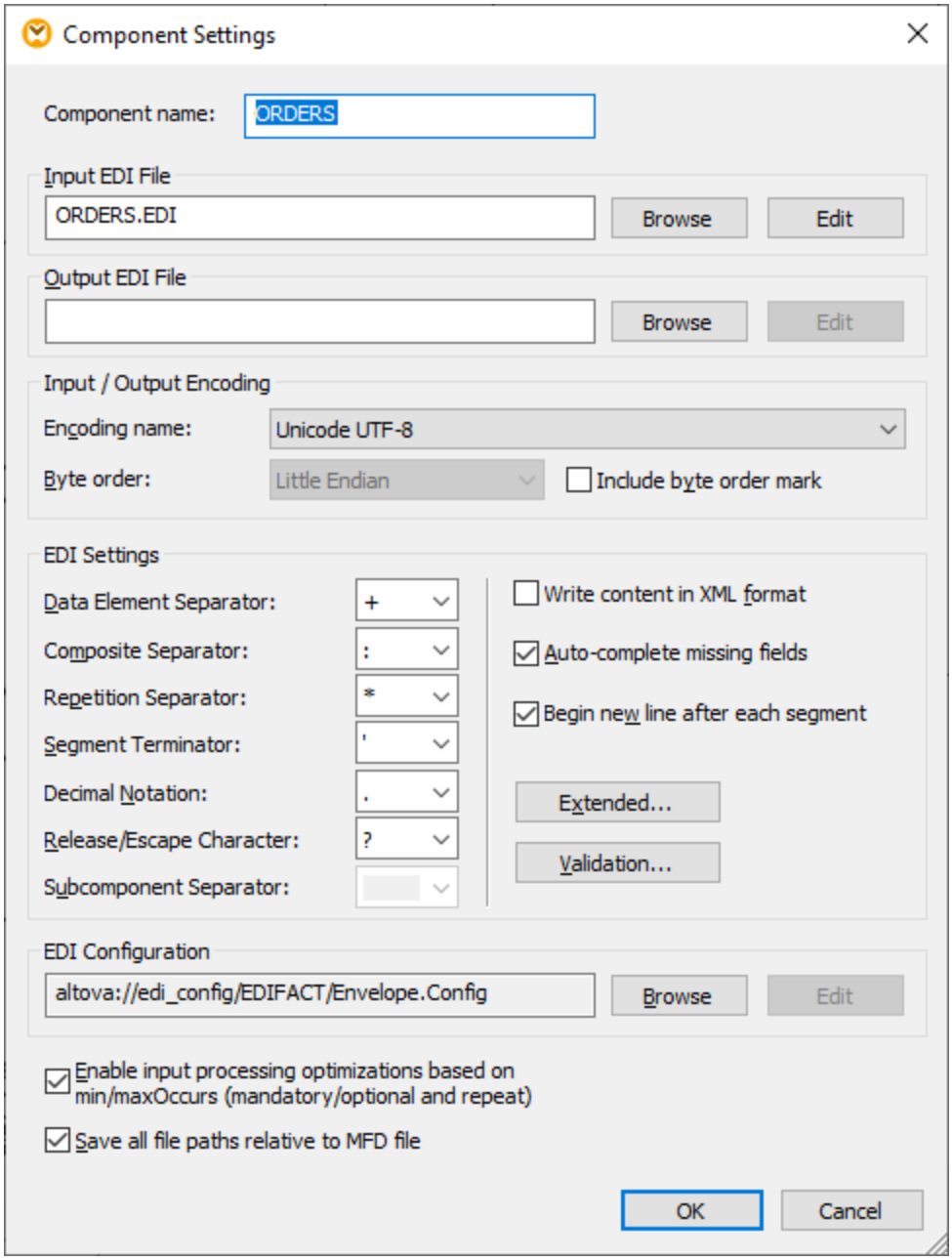
マッピング全体のためにマッピングの設定を構成するには、マッピングを右クリックして、コンテキストメニューから[マッピングの設定] を選択し、マッピング設定の変更を参照してください。

コード生成設定にアクセスするには[ツール | オプション] に移動し、[生成] タブをクリックします。

6.5.4 EDI コンポーネント設定

マッピングエリアにEDI コンポーネントを追加すると、コンポーネント設定ダイアログボックスから適用することのできる設定を構成することができます。コンポーネント設定ダイアログボックスを以下の方法を用いて開くことができます:

- マッピングのEDI コンポーネントをクリックし、 コンポーネント メニューから「プロパティ」をクリックします。
- EDI コンポーネントのタイトルバーを右クリックします。
- EDIコンポーネントのタイトルバーを右クリックして、「プロパティ」をクリックします。



EDIコンポーネント設定ダイアログボックス

使用することのできる設定以下にリストされています。現在選択されているコンポーネントがサポートしない場合、一部の設定を使用不可能なことご注意してください。

コンポーネント名	コンポーネント名はコンポーネントを作成する際に自動的に生成されます。ですが、後に名前を変更することが可能です。 コンポーネント名はスペースを含むことができ、またフルストップ文字も含むことができます。コンポーネント名は、スラッシュ、バックスラッシュ、コロン、二重引用符、行頭と末尾スペー

(C) 2015-2021 Altova GmbH

	スを含む場合があります。全般的に、コンポーネントの名前を変更する際、以下の影響を考慮してください: • マッピングを FlowForce Server にデプロイする場合、コンポーネント名は一意である必要があります。 • コマンドラインに入力できる文字のみを使用することが奨励されます。コマンドラインと Windows 内では、それぞれの文字で異なったエンコードを使用する場合があります。
入力 EDI ファイル	MapForce がデータを読み込む EDI ソースファイルを指定します。このフィールドはソースコンポーネントにとり重要で、最初のコンポーネントを作成すると入力され、EDI インスタンスファイルに割り当てます。MapForce がデータを読み込む Microsoft Excel ファイルを指定します。
出力 EDI ファイル	MapForce がデータを書き込む EDI ターゲットファイルを指定します。このフィールドはターゲットコンポーネントにとって重要です。
入力/出力エンコード	出力インスタンスファイルの次の設定を指定することを許可します: • エンコード名 • バイトオーダー • バイトオーダーマーク (BOM) 文字が含まれるか否か。 デフォルトでは、「**新しいコンポーネントのためのデフォルトのエンコード**」オプション内で定義されたエンコードを持つ新しいコンポーネント。「**ツール	オプション**」からこのオプションにアクセスし、タブを生成します。
EDI 設定	この設定グループにより、カスタム区切り文字、セパレーター、終端文字を定義することができます (EDI フォーマットによりサポートされている場合のみ、この設定を使用することができます) このダイアログボックスに入力される EDI セパレーターは、EDI ファイルに書き込みが加えられると常に効果を表します。EDI ファイルを読み込む場合、セパレーターは入力ファイルが自身のセパレーターを定義、または含まない場合のみ効果を表します (例えば、UNA "サービス文字列アドバイズ" セグメントを持たない EDIFACT ファイル) EDI 入力コンポーネント/ファイルがセパレーター定義を含む場合、(例えば、ISA セグメントを持つ X12 ファイル)、既存のセパレーターはコンポーネント設定 ダイアログボックス内でそのファイルのために定義されたセパレーターをオーバーライドします。 コンボボックスの 1 つに文字コードを入力することにより印刷することのできない文字を、"x" の後に 16 進数 ASCII 文字 を入力してセパレータとして使用することができます。例えば、RS コントロール文字のために "x1e" を入力します (ASCII レコードパラメーター、小数コード 30)。 「**サブコンポーネントセパレーター**」は HL7 基準に適用されます。このセパレーターのデフォルトの値はアンパサンド (**&**) です。
XML 書式でコンテンツを書く	このチェックボックスの選択はターゲット EDI コンポーネントのためにのみ意味があります。コンポーネントを変更し EDI 出力の代わりに XML 出力を生成します 例: XML と EDI 出力間の切り替えを参照してください。

不足するフィールドの自動補完	このオプションはターゲット EDI コンポーネントに適用されます。。このチェックボックスが選択されると、MapForce は、データフィールドの一部を自動的に補完します。 これは、操作が現在選択されているEDI フォーマットの仕様に相反しない場合のみ適用することができます。このような振る舞いを無効化するには、このチェックボックスをクリアしてください。 次も参照してください: • ASC X12 自動データ補完のルール • UN/EDIFACT 自動データ補完のルール
各セグメントの後の改行	このオプションはターゲット EDI コンポーネントに適用されます。。 チェックボックスが選択されると、MapForce は、CR/LF (リターン キー / 改行) 文字を、各 EDI セグメントの後に追加します。 メッセージ内に存在する場合 EDI 標準はこれらのラインを無視します。
追加設定	現在の現在のEDI コンポーネントの追加設定を定義することのできるダイアログボックスを開きます 使用することのできる設定はEDI 種類により異なります。使用することのできる設定はEDI コンポーネント型により異なります。
検証	現在のEDI コンポーネントの検証設定を定義することのできるダイアログボックスを開きます (EDI コンポーネントの検証を参照)。
EDI 構成	現在のコンポーネント内で使用されるEDI 構成に関する情報を提供します。MapForce がEDI 構成を解決する方法を参照してください。
min/maxOccurs をベースにした入力処理の最適化の有効化	このオプションは、`minOccurs` と`maxOccurs="1"` などの必要な属性または子要素が 1 つのアイテムのみを含むシーケンスのための特別な処理を許可します。この場合、シーケンスの最初のアイテムが抽出されると、アイテムは、シーケンスとしてではなく直接動的な値として処理されます。 入力データがスキーマに対して有効でない場合、マッピングをエラーメッセージとともに停止する空のシーケンスがマッピング内で生じる場合があります。このような無効な入力を処理するには、このチェックボックスをクリアしてください。
MFD ファイルに相対的なすべてのファイルパスを保存する	このオプションが有効化されると、MapForce は、コンポーネント設定ダイアログボックスに表示された、MapForce Design (.mfd) ファイルの場所に相対したファイルパスを保存します。この設定 は以下のファイルに影響を与えます: • 入力 EDI ファイル(存在する場合) • 出力 EDI ファイル(存在する場合) Uコンポーネント上で相対パスを使用するを参照してください。

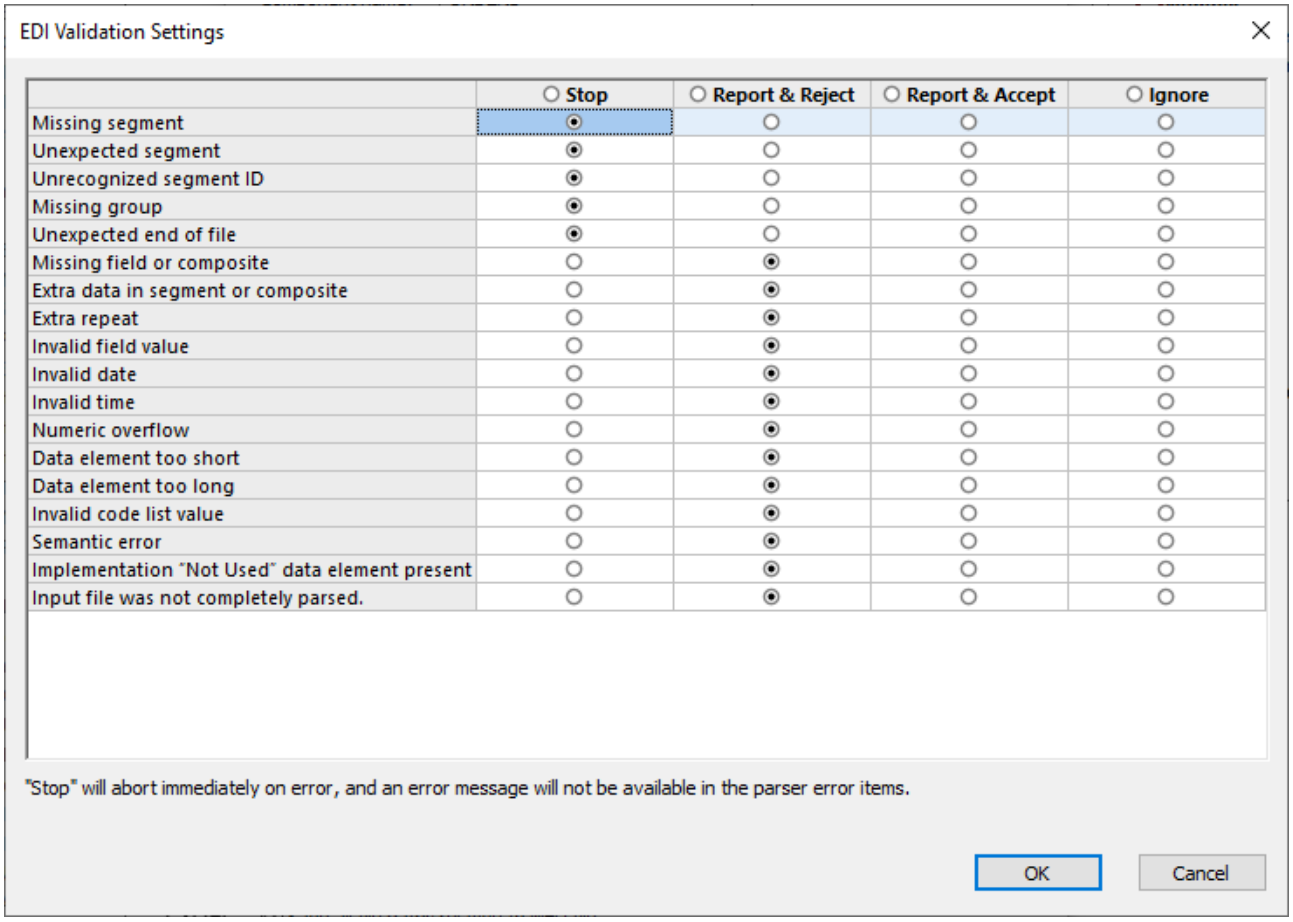
6.5.5 EDI コンポーネント検証

MapForce は、マッピングが実行される際にサポートされる全てのEDI ソースとターゲットドキュメントを検証します。これはMapForce 内でマッピングをプレビューする際に、出力タブをクリックして、またはMapForce Server、またはC++、C#、やJava プログラムなどの外部の実行エンジンにより実行される場合に発生します。マッピングにより生成されたEDI 出力のみを検証することもできます。これを行うには、「出力」タブをクリックして、「出力の検証」 ツール バーボタン をクリックします(マッピング出力の検証も参照してください)。

MapForce は、EDI ドキュメントをこのドキュメント内で指定されているとおり、個々のサポートされるフォーマットのために検証します。検証処理は、しかしながら、完全な EDI 構文またはセマンティックの検証ではありません。例えば、使用することのできるフィールドの値が、処理の可能性を予想する無限かつ完全な値のセットで簡単に示すことができない場合があります。このような制限を解決するには、構成ファイルを使用して、ケースバイケースで検証ルールをカスタム化することが可能です（ EDI 検証のカスタム化を参照してください）。

基本的なレベルでは、検証パラメーターは、MapForce から特定のニーズに対応するために直接カスタム化することができます。例えば、エラーを無視、特定のエラーでの検証処理の中止、エラーの受け入れ、拒否、または報告などを選択することができます。

検証設定をカスタム化するには、EDI コンポーネントをダブルクリックして、コンポーネント設定 ダイアログボックス内の「検証」ボタンをクリックします（次EDI コンポーネント設定も参照してください）。デフォルトの検証設定 は下に示されています。



EDI 検証設定 ダイアログボックス

上にリストされる検証イベントでは、以下のアクションをとることができます: **停止**、**報告して拒否**、**報告して受理**、**無視**。

停止は致命的なエラーをキャッチして、レポートメッセージを生成すること無くマッピングの実行を停止します

報告して拒否 はエラーの情報を提供し、現在のメッセージを拒否します。**報告して受理** はエラーの情報を提供し、現在のメッセージを受理します。実際のエラーは、MapForce のメッセージウィンドウ内で確認することができます。ソースX12 とHIPAA コンポーネントに関しては、検証エラーは、X12 997 またはX12 999 レポートを作成するマッピングコンポーネント内でログされます（検証結果とX12 Acknowledgementを参照してください）。

無視 は、検証イベントを無視します。このアクションを選択すると、エラーに関しての情報は表示されませんので、注意して使用してください。

EDI マッピングの処理中に予期されないセグメントが発生した場合、MapForce とMapForce Server のみが復元および解析を継続する機能を備えています。生成されたC++、C#, またはJava コードは予期されないセグメントから復元することはできず、処理は停止されます。予期されないセグメントには次が含まれます: 間違っているセグメントの順序、不足している必須のセグメント、間違っているセグメントの名前、構成ファイル内に存在しない追加セグメントなど。

EDI 検証イベントのために構成することのできるカスタムアクション(**停止**、**無視** など) は、以下のとおりです

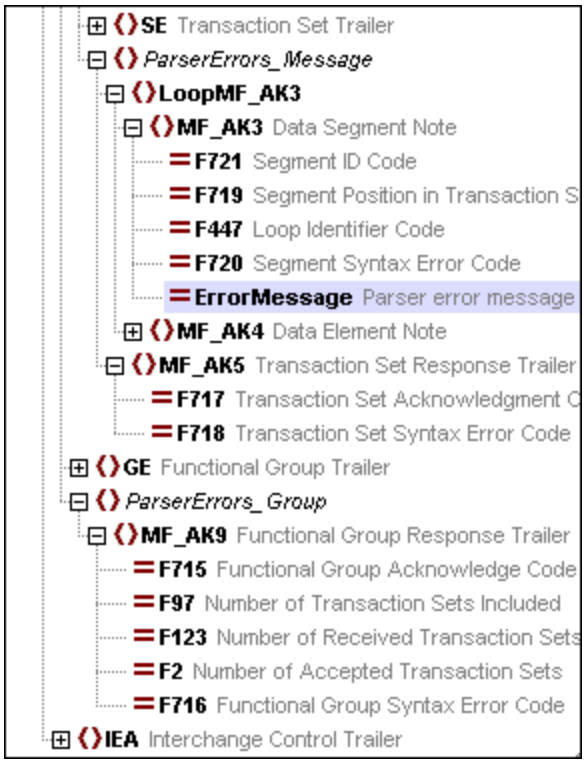
イベント	説明	エラーの発生箇所
見つからないセグメント	必須セグメントが見つからないか、最低出現数未満の数です。	ソースコンポーネント
予期せぬセグメント	セグメントは定義されていますが、このメッセージには出現しません。	ソースコンポーネント
認識できないセグメント ID	この仕様では定義されていないセグメントを発見しました。	ソースコンポーネント
見つからないグループ	必須グループが見つからないか、最低出現数未満の数です。	ソースコンポーネント
予期せぬファイルの終端	見つからないデータのため、インスタンスの解析を行うことができません。	ソースコンポーネント
見つからないフィールドまたはコンポジット	必須フィールドまたは複合要素が見つからないか、最低出現数未満の数です。	ソースまたはターゲットコンポーネント
セグメントまたはコンポジットの冗長なデータ	入力インスタンスに、構文では指定されていないデータが含まれています。	ソースコンポーネント
不要な繰り返し	セグメント/複合内に実際にあるフィールドの数が、指定された最大数を超えました。	ソースまたはターゲットコンポーネント
無効なフィールド値	数値フィールドに無効な文字が含まれています。	ソースコンポーネント
無効な日付	日付フィールドに無効な文字が含まれているか、月または日の値が無効です。	ソースコンポーネント
無効な時間	時間フィールドに無効な文字が含まれているか、時または分の値が無効です。	ソースコンポーネント
数値のオーバーフロー	指定されたドメインを数値が超えてしましました。このエラーは生成されたコードでのみサポートされます。	ソースコンポーネント
短すぎるデータ要素	データ要素の長さが指定された最低値を下回っています。	ソースまたはターゲットコンポーネント
長すぎるデータ要素	データ要素の長さが指定された最大値を超えてします。	ソースまたはターゲットコンポーネント
無効なコードリスト値	コードリストの値が指定された値のセットにマッチしません。データ要素の可能なコード値を変更するを参照してください。	ソースまたはターゲットコンポーネント
論理エラー	論理エラーが発生しました。	ソースまたはターゲットコンポーネント
実装にて「使用されない」となっているデータ要素が存在	入力ファイル内に存在する要素はHIPAA 構成ファイル (`maxOccurs="0"`) では許可されていません。	ソースコンポーネント

(C) 2015-2021 Altova GmbH

イベント	説明	エラーの発生箇所
入力ファイルが未解析	入力が完全に解析されませんでした。	ソースコンポーネント

検証結果とX12 Acknowledgment

ソースEDI コンポーネントに対して、検証結果(レポート)はEDI コンポーネントをベースとする「仮想」アイテムへ配置されます(ParserError_ Message ならびにParserErrors_ Group 以下)。これらのアイテムはX12 997 または999 Acknowledgment にて使用することができます。インターチェンジの状態を報告するX12 Acknowledgment ファイルを作成する方法については[X12 997 Functional Acknowledgement の生成]と[X12 999 Implementation Acknowledgment の生成]を参照してください。



X12 またはHIPAA.X12 コンポーネントに関しては、EDI 検証設定 ダイアログボックス上で使用することのできる検証アクションがあり、(上記参照) 以下を意味します:

停止: レポートメッセージを生成すること無くマッピングの実行を停止します。

報告して拒否ならびに**報告して受理** : EDI コンポーネントのParser_ Errors_ Message ならびにParser_ Errors_ Group アイテムに情報を与え、更に処理を継続することができます。

拒否するか、**受理** するかにより、Functional Group Acknowledge Code F715 ならびにTransaction Set Acknowledgment Code F717 に含まれる値が以下のように変化します:

*"Rejected" (拒否)*を表す値 'R' または
値 'E' は *受け入れられましたが、エラーが発生しました*を表しています。エラーはメッセージウィンドウにも表示されます。

無視: 特定のエラーを無視します。Parser_ Errors_ Message ならびにParser_ Errors_ Group アイテムに情報が表示されることはありません。

6.5.6 EDI から XML を生成する

前記のとおり、EDI データを MapForce によりサポートされている他のデータフォーマットにマップまたは他のデータフォーマットからマップすることができます。EDI マッピングは非-EDI マッピングとは大幅に異ならず、生成されたデータに対して同じレベルの管理を必要とします。

デザインする度量を必要とする通常のマッピング以上に、MapForce は XML スキーマを必要とせずマッピング接続線を手動で書く必要もなく ボタンクリック数回で EDI ファイルを *素早く* XML フォーマットに変換する機能も提供しています。これは EDI ファイルを他の代替フォーマットにエクスポートすることを可能にする読みやすい便利な機能です。The XML 出力は整形式で、MapForce により生成された多くの出力同様にファイルに保存することが可能です。

この機能の使用は XML スキーマを必要とせず、スキーマを提供しません。生成された XML は要素の階層を考慮して *ソースコンポーネント* からすべてのデータを複製し、データ要素名とその値を出力します。データを含まないフィールドは省略されます。生成された XML のそのままの構造はソース EDI データにより異なり、変更することはできません。生成された XML データのフルコントロールが必要な場合、個別のマッピングアイテムをタックルし標準の EDI から XML へのマッピングを代わりに作成します。標準 EDI から XML へのマッピングのサンプルに関しては例: UN/EDIFACT を XML にマッピングする、例: HIPAA X12 から XML へのマッピングと例: HL7 から XML へのマッピングを参照してください。

必要条件

クイック EDI ファイルを XML に MapForce を使用して変換するには、マッピングの変換言語が BUILT-IN である必要があります。他の変換言語はサポートされていません。この必要条件は特に *クイック* EDI から XML への変換に適用されます。標準 EDI から XML へのマッピングは追加言語をサポートします、EDI を参照してください。

EDI から XML へのクイック変換

EDI コンポーネントをマッピングに追加した後、XML を生成する方法が 2 つあります:

1. EDI コンポーネントが EDI *インスタンス* ファイルからデータを読み取る場合 (すなわち、*ソース* コンポーネントからの場合)、タイトルバーを右クリックし **XML にマッピングを作成する** をコンテキストメニューから選択します。*ステップバイステップ* のサンプルに関しては 例: EDI から XML へのクイック変換を参照してください。

メモ 既存の EDI 出力に加え新規の出力を生成し、マッピングを「チェーンされた」マッピングにするために、ターゲットコンポーネントのためにこれを行うこともできます。詳細に関しては、チェーンされたマッピングを参照してください。

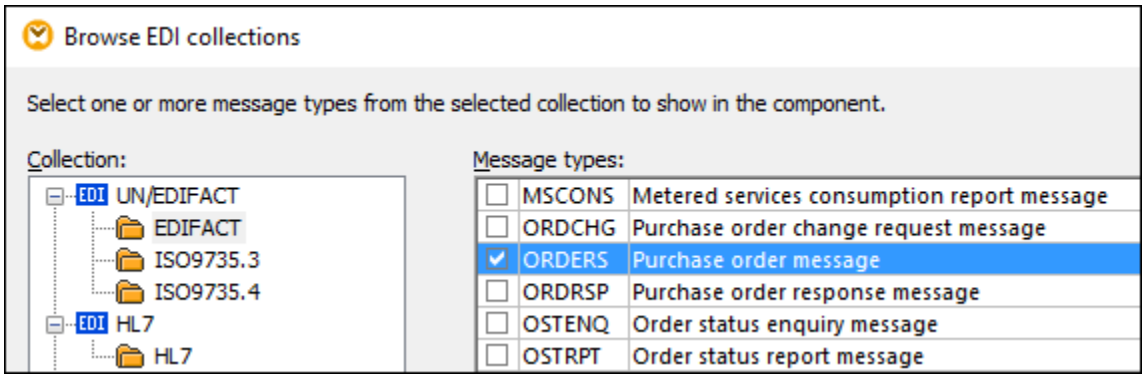
2. If the EDI コンポーネントが *ターゲット* である場合、タイトルバーを参照にダブルクリックしてコンポーネント設定を開き、**XML フォーマットでコンテンツを書く** チェックボックスを選択します。*ステップバイステップ* のサンプルに関しては 例: XML と EDI 出力間の切り替えを参照してください。

6.5.6.1 例: EDI から XML へのクイック変換

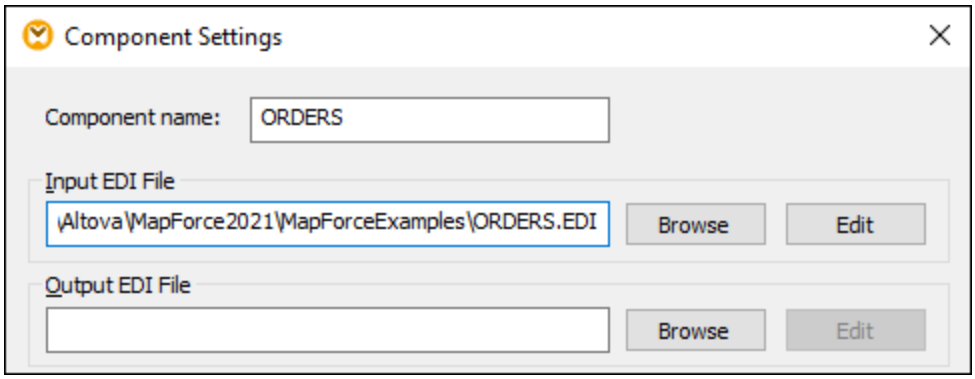
このサンプルは少ない労力で EDI ファイルから XML 書式にデータを変換する方法について説明しています。サンプルは具体的に *クイック* 変換を扱っており、手動のマッピング接続を描くことは含まれていません。標準 EDI から XML への変換サンプルに関しては、例: UN/EDIFACT を XML にマッピングするを参照してください。

1. 新規のマッピングを作成し BUILT-IN を変換言語として選択します。変換言語の選択を参照してください。
2. [**挿入**] メニューから **[EDI]** をクリックします。
3. **[EDIFACT]** コレクションと **[ORDERS]** メッセージ型を選択し **[OK]** をクリックします。

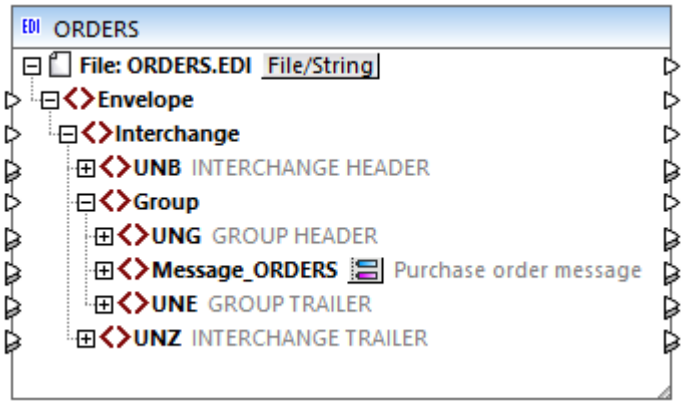
(C) 2015-2021 Altova GmbH



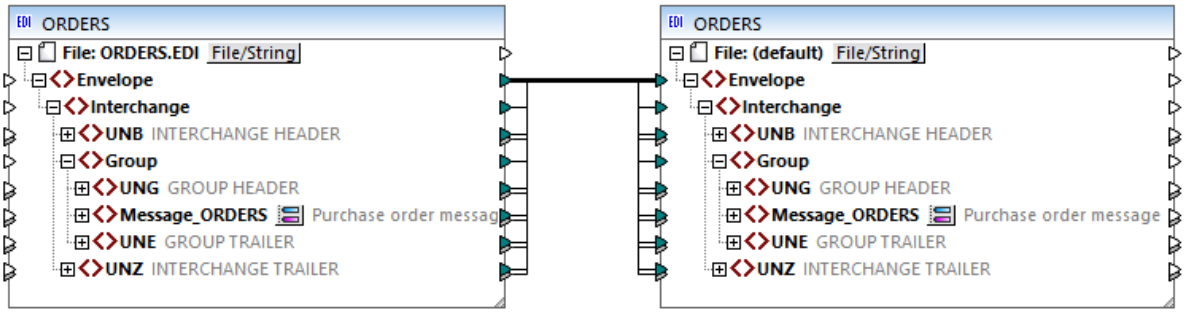
4. サンプルEDI ファイルを提供するようにプロンプトされると[参照] をクリックして **[ORDERS.EDI]** ファイルを<**マイドキュメント>\Altova\MapForce2021\MapForceExamples** ディレクトリから開きます。ファイルを開くとコンポーネント設定ダイアログボックスが開かれます。これによりマッピングに追加する前にEDI コンポーネントの設定をレビューできるようになります。これらの設定を後に必要に応じて変更することができます。**ORDERS.EDI** ファイルは**入力** EDI ファイルとして表示されていることに注意してください。



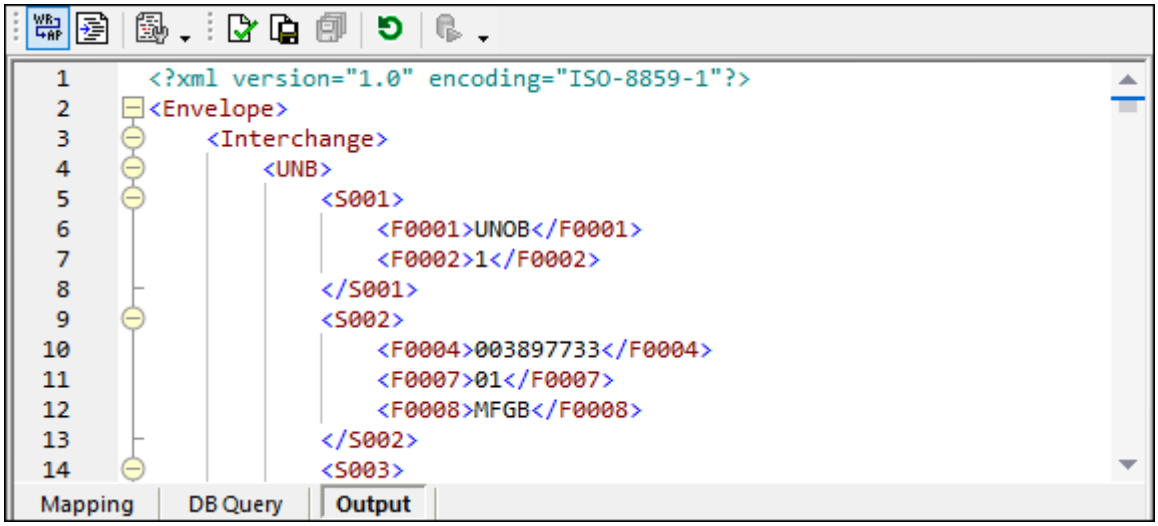
5. **[OK]**をクリックします。EDI コンポーネントがマッピングウィンドウ内で表示されています。



6. ORDERS コンポーネントのタイトルバーを右クリックし、コンテキストメニューから「XML へマッピングを作成 」を選択します。これはターゲット XML コンポーネントを作成し全ての接続が全てコピー接続を介して自動的に行われるようにします。



7. **出力** タブをクリックして生成されたXML を確認します。



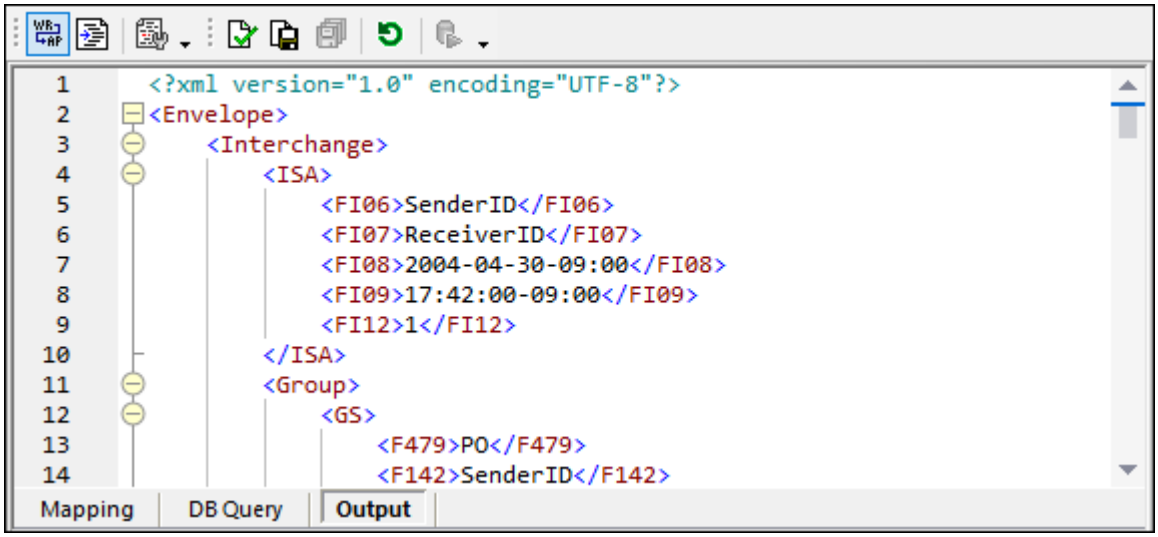
6.5.6.2 例: XML と EDI 出力間の切り替え

このサンプルはEDI 出力の代わりにXML 出力を生成するためのターゲット EDI コンポーネントの構成方法について説明しています。構成は最小限で、以下に示される通り単一のプロパティの変更を含んでいます。

1. 次のマッピングを開きます: **<マイドキュメント>\Altova\MapForce2021\MapForceExamples\Xml_To_X12_Order.mfd**。
2. 変換言語をBUILT-IN に変更します（**[出力]** メニューから**[ビルトイン実行エンジン]**を選択します）。
3. ターゲット **850** コンポーネントのタイトルバーをダブルクリックします。[EDI コンポーネント設定] ダイアログボックスが開かれます。
4. **[XML フォーマットでコンテンツを書く]** チェックボックスを選択します。

出力 タブをクリックして、マッピングをプレビューすることができます。下で説明されているとおりEDI コンポーネントは出力 XML をEDI の代わりに生成します。

(C) 2015-2021 Altova GmbH



6.5.7 ASC X12

ASC X12 はドキュメントのインターチェンジのための業界の標準です。MapForce は以下のバージョンをサポートします: 3040、3050、3060、3070、4010、4020、4030、4040、4041、4042、4050、4051、4052、4060、5010、5011、5012、5020、5030、5040、5050、6010、6020、6030、および6040。

MapForce のデフォルトの ASC X12 バージョンは 6040 です。

ASC X12 コンポーネントには、EDI 検証 設定ダイアログボックス内で選択された設定により EDI パーサーエラー情報やデータが書き込まれる「仮想」ノードがあります（EDI コンポーネントの検証を参照）。 X12 997 Functional Acknowledgement はすべての X12 ドキュメントから生成することができます。

ASC X12 に関しての詳しい情報は、http://www.x12.org/ を参照してください。

6.5.7.1 ASC X12 検証ルール

ANSI X12 ドキュメントの検証時には以下の項目がチェックされます:

- ISA ならびに IEA セグメントが存在するか。
- ISA/I01 内に、法的承認情報修飾子が含まれるか。
- ISA/I03 内に、法的セキュリティー情報修飾子が含まれるか。
- 2 つの ISA/I05 セグメントに法的インターチェンジ ID 修飾子が含まれるか。
- ISA/I08 内に整形式の日付値が含まれるか。
- ISA/I09 内に整形式の時刻値が含まれるか
- ISA/I13 内に法的 boolean 値が含まれるか。
- ISA/I14 内に法的インターチェンジ使用インジケーターが含まれるか。
- ISA/I12 と IEA/I12 内に含まれる値が同一のものか。
- IEA/I16 内に含まれる値が、インターチェンジ内に含まれる関数グループの正しい数か

各関数グループに対して、以下の項目がチェックされます:

- マッチするGS とGE のペアが存在するか。
- GS/373 内に整形式の日付値が含まれるか。
- GS/337 内に整形式の時刻値が含まれるか。
- GS/28 ならびにGE/28 内に含まれる値が同一のものか。
- GE/97 内に含まれる値が、関数グループ内に含まれるメッセージの正しい数か。

各メッセージに対して、以下の項目がチェックされます:

- マッチするST とSE のペアが存在するか。
- ST/143 内に正しいメッセージ識別子が含まれるか。
- ST/329 とSE/329 内に含まれる値が同一のものか。
- SE/96 内に含まれる値が、メッセージ内に含まれるセグメントの正しい数か。

6.5.7.2 ASC X12 自動データ補完ルール

ASC X12 ファイルを生成する際、MapForce は、("自動データ保管" または "自動補完" ルールとして説明されている) ルールに従い、自動的に特定のフィールドまたは値を生成する可能性があります。この振る舞いを無効化するには、 EDI コンポーネント設定内のチェックボックスの「不足しているフィールドの自動補完」のチェックを解除してください。

EDI/X12 の自動データ補完により、以下の項目が保証されます::

- インターチェンジレベルで、ISA とIEA のペアが存在する。
- GS またはGE の一方が存在する場合、もう一方のID も存在する。
- メッセージレベルにて、少なくとも 1 つのST/SE ペアが存在する。
- ISA/101 ならびにISA/103 が存在する。データが含まれていない場合、00 が挿入されます。
- ISA/102 ならびにISA/104 が存在する。データが含まれていない場合、10個の空白が挿入されます。
- ISA/105 セグメントが存在する。データが含まれていない場合、"ZZ" が挿入されます。
- ISA/108 が存在する。データが含まれていない場合、EDI フォーマット内にある現在日時が挿入されます。
- ISA/109 が存在する。データが含まれていない場合、EDI フォーマット内にある現在時刻が挿入されます。
- ISA/165 が存在する。データが含まれていない場合、繰り返しセパレーターが挿入されます。
- ISA/111 が存在する。データが含まれていない場合、ユーザー定義設定から得られたインターチェンジ制御バージョン番号が挿入されます。EDI コンポーネント設定 の「拡張 | インターチェンジ制御バージョン番号」を参照してください。
- ISA/112 の存在。
- ISA/113 が存在する。データが含まれていない場合、確認を要求設定が使用されます。EDI コンポーネント設定 の「拡張 | 確認を要求」を参照ください。
- ISA/114 の存在。データが含まれない場合、P が挿入されます。
- ISA/115 が存在する。データが含まれていない場合、ユーザー定義設定から得られた複合セパレーターが挿入されます。EDI コンポーネント設定 の「設定 | 複合セパレーター」を参照ください。
- IEA/116 が存在する。データが含まれていない場合、インターチェンジ内にある関数グループの数が計算され、挿入されます。
- IEA/112 が存在する。データが含まれていない場合、ISA/112 がコピーされます。

メモ　このセクションにて言及されなかったフィールドが (自動データ補完により) 挿入または作成されることはありません。正しい値を自動的に確定することはできません。例えば、(EDI コンポーネント内に) ターゲット親要素のA と、それらの子アイテムx、y、z が有り、そのうちy が必須アイテムとなっている場合、子要素の "y" がターゲットコンポーネントへマップされた時だけ、親要素のA が作成されます。

潜在的に存在する関数グループがチェックされます:

- GS/373 が存在する。データが含まれていない場合、EDI フォーマットにある現在の日付が挿入されます。
- GS/337 が存在する。データが含まれていない場合、EDI フォーマットにある現在の時刻が挿入されます。

- GE/97 が存在する。データが含まれていない場合、関数グループ内にあるメッセージの数が計算され、挿入されます。
- GE/28 が存在する。データが含まれていない場合、GS/28 から得られた値がコピーされます。

メッセージのチェックにより、以下の項目が保証されます:

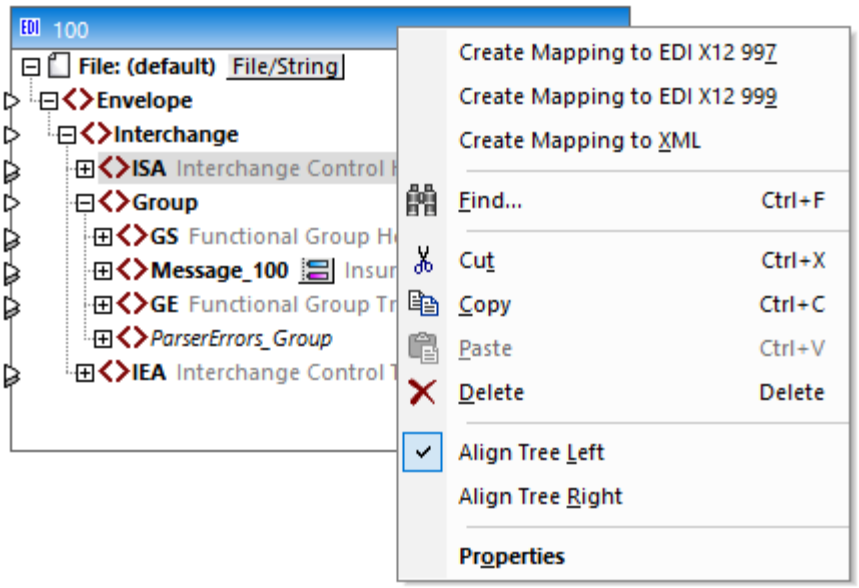
- ST/143 が存在する。データが含まれていない場合、メッセージの名前が挿入されます。
- SE/96 が存在する。データが含まれていない場合、メッセージ内にあるセグメントの数が計算され、挿入されます。
- ST/329 ならびに SE/329 が存在する。SE/329 内にデータが含まれていない場合、ST/329 から得られた値がコピーされます。

6.5.7.3 X12 997 Functional Acknowledgemen の生成

X12 997 (Functional Acknowledgement) は、EDI インターチェンジの状態を報告します。ドキュメントの処理中に発生するすべてのエラーは、この中に報告されます。MapForce は、自動的にメインのマッピングエリア内に X12 997 コンポーネントを生成し、自動的に必要なコネクタを作成します。

EDI 997 Functional Acknowledgement の生成:

1. 「挿入」メニューから、「EDI」をクリックします。X12 コンポーネントをマッピングに追加します。インスタンスファイルとして選択するように問われると、X12 997 確認を作成するためのファイルを参照します。
2. EDI コンポーネントのタイトルバーを右クリックし、「**EDI X12 997 へマッピングを作成する**」を選択します(このコマンドは、X12 コンポーネントのためのみに有効化されています)。



これは、EDI 997 マッピングコンポーネントを作成し、X12 997 確認を生成するために必要なアイテムを自動的に接続します。<**マイドキュメント**>¥Altova¥MapForce2021¥MapForceExamples¥フォルダー内の**X12_To_XML_Order.mfd** マッピングデザインファイルを参照してください。このマッピングから生成された確認ファイルを確認するには、EDI 997 コンポーネントの「**プレビュー**」( ) ボタンをクリックし、そして「**出力**」タブをクリックします。

X12 997 Acknowledgement ファイルを保存する:

- 「出力」タブをクリックして、そして「生成された出力を保存」( ) ツールバーボタンをクリックします。マッピングからプログラムコードを生成する場合、コンポーネント設定 ダイアログボックスの"Output EDI File" フィールド内にファイル名を入力します(次を参照: EDI コンポーネント設定)。

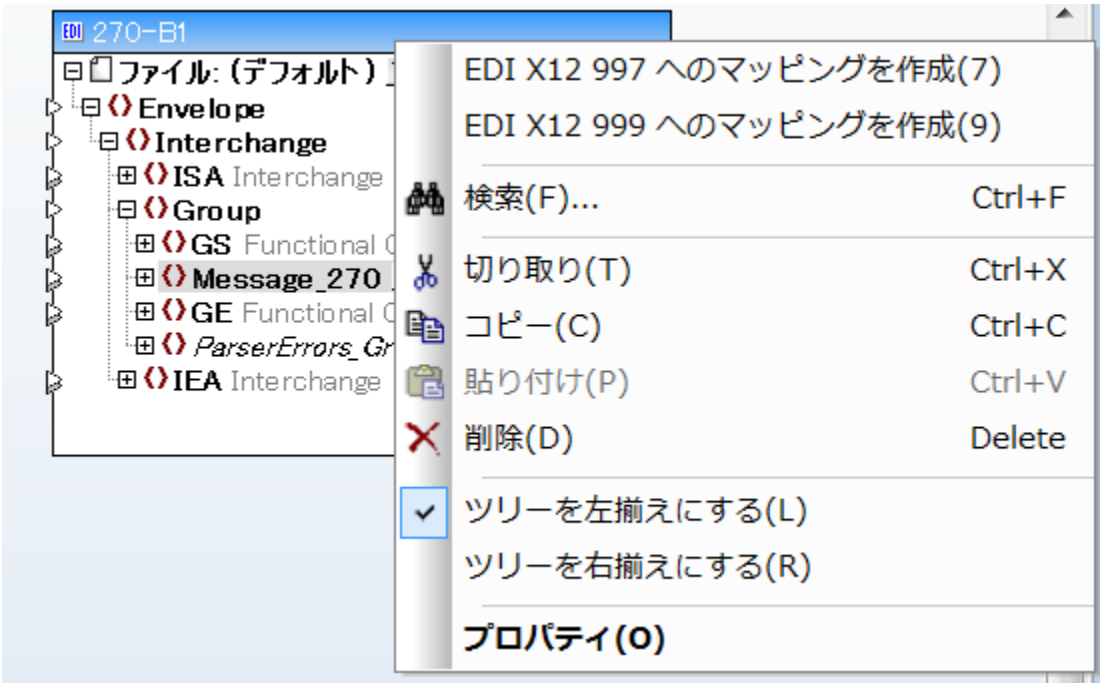
6.5.7.4 X12 999 Implementation Acknowledgment の生成

X12 999 (Implementation Acknowledgment) は、X12 非準拠、および、アプリケーションエラーを報告します。997 Functional Acknowledgement のスーパーセットであるため、X12 またはHIPAA 実装ガイド構文の必要条件をベースにしたトランザクションを受理、または拒否するために、999 を997 の代わりに使用することができます。

ドキュメントの処理中に発生するエラーは、このファイルに報告されます。(例えば、"必要とされるセグメントが不足しています"、"必要とされるデータ要素 が不足しています"、"この実装内でコード値は使用されていません"、など)。MapForce は、自動的にマッピング上にX12 999 コンポーネントを生成、および、必要なマッピングの接続を自動的に作成することができます。

X12 999 Implementation Acknowledgment の生成:

1. 「挿入」メニューから、「EDI」をクリックします。X12 またはHIPAA コンポーネントをマッピングに追加します。インスタンスファイルとして選択するように問われると、X12 999 確認を作成するためのファイルを参照します。
2. ソースEDI コンポーネントのタイトルバーを右クリックし、「**EDI X12 999 へマッピングを作成する**」を選択します(このコマンドは、X12 とHIPAA コンポーネントのためのみに有効化されています)。



これは、EDI 999 マッピングコンポーネントを作成し、X12 999 実装確認を生成するために生成するために必要なアイテムを自動的に接続します。生成された確認ファイルをプレビューするには「出力」タブをクリックします。マッピングに複数のターゲットコンポーネントが存在する場合、最初に「プレビュー」( ) ボタンをクリックし、そして タブをクリックします。

999 Acknowledgement ファイルを保存する

- 「出力」タブをクリックし、そして「生成された出力を保存」( ) ツールバーボタンをクリックします。マッピングからプログラムコードを生成する場合、コンポーネント設定 ダイアログボックスの「出力 EDI ファイル」フィールドをクリックします（次を参照: EDI コンポーネント設定）。

6.5.8 HIPAA X12

HPAA X12 は、米国保険社会福祉省により制定された最新バージョンの電子ヘルスケアレコード標準で、EDI X12 バージョン 5010 に従ったかたちで、保険会社、医療機関、雇用主間における電子的な医療データのやり取りを行うために使用されます。

MapForce では最新のリリースである HPAA 実装仕様（TR3）の A2 がサポートされています。それ以前のリリースは ZIP ファイルとして Altova W eb サイトからダウンロードいただけます。

HIPAA コンポーネントは通常の ANSI X12 コンポーネントと類似しており、MapForce では以下のトランザクションがサポートされます:

X12 名	メッセージ名
X279A1	"Health Care Eligibility Benefit Inquiry (270)"
X279A1	"Health Care Eligibility Benefit Response (271)"
X212	"Health Care Claim Status Request (276)"
X212	"Health Care Information Status Notification (277)"
X214	"Health Care Claim Acknowledgment (277)"
X217	"Health Care Services Review - Request for Review (278)"
X217	"Health Care Services Review - Response (278)"
X218	"Payroll Deducted and Other Group Premium Payment for Insurance Products (820)"
X220A1	"Benefit Enrollment and Maintenance (834)"
X221A1	"Health Care Claim Payment/Advice (835)"
X222A1	"Health Care Claim: Professional (837)"
X224A2	"Health Care Claim: Dental (837)"
X223A2	"Health Care Claim: Institutional (837)"
X231A1	"Implementation Acknowledgment For Health Care Insurance (999)"

MapForce における標準的な X12 メッセージ処理方法との違い:

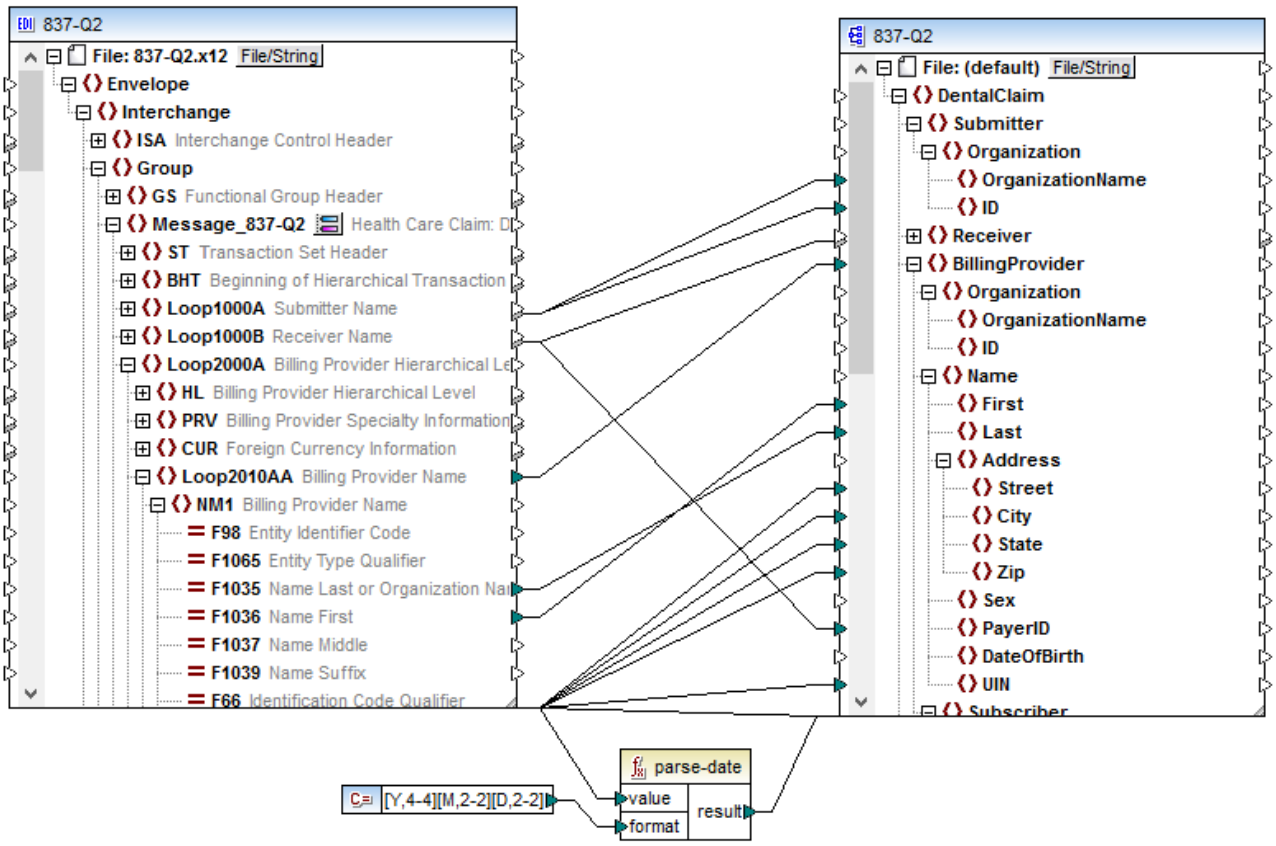
- HL セグメントの階層を自動的に管理
- （837 メッセージにおける）浮動構造のサポート
- さらなるフィールドの自動補完と検証

MapForce では、標準的な X12 における既存の999 や997 Functional Acknowledgement と似た、HIPAA X12 999 Implementation Acknowledgment の自動生成をサポートしています（次を参照: X12 999 Implementation Acknowledgment の生成）。

HPAA コンポーネントでは、インターチェンジ(EDI コンポーネント)における複数のメッセージがサポートされています。

6.5.8.1 例: HIPAA X12 から XML へのマッピング

このサンプルは、"837-Q2 Health Care Claim: Dental" トランザクションを含むファイルの XML ターゲットファイルへマップする方法について説明しています。このマッピングサンプルは以下のパスで見つけることができます: **<マイドキュメント>\Altova\MapForce2021\MapForceExamples\HIPAA_837D.mfd**。



HIPAA_837D.mfd

マッピングは下に示されているステップどおり作成することができます。

ステップ 1: ソース HIPAA X12 ファイルをマッピングに追加する

1. 「挿入」メニューから、「EDI」をクリックします。
2. "HIPAA.X12" コレクションを選択し、メッセージ型 "837-Q2 Health Care Claim: Dental" の横のチェックボックスを選択します。
3. インスタンスファイルを提供するように問われると、次のパスから使用することのできるソースファイルを開きます: **<マイドキュメント>\Altova\MapForce2021\MapForceExamples\837-Q2.x12**.

(C) 2015-2021 Altova GmbH

4. プロンプトされると、必要に応じて、新規に追加されたHIPAA X12 コンポーネントの設定を調整します（詳細に関しては、次を参照してください：EDI コンポーネント設定）。

ステップ 2: マッピングにターゲットスキーマを追加する

1. 「挿入」メニューから、「XML スキーマ/ファイル」をクリックし、<マイドキュメント>\Altova\MapForce2021\MapForceExamples\ ディレクトリから **837-Q2.xsd** ファイルを開きます。
2. サンプルXML ファイルを提供するように問われると、「スキップ」をクリックします。ターゲットドキュメントのルートとして **DentalClaim** を選択します。

ステップ 3: 日付をフォーマットするために関数を追加し、マッピング接続を描く

ソースHIPAA ファイル内では、DMG セグメントの日付は、`19490501` のフォーマットで表示されています。このフォーマットをターゲット XML ファイル`1949-05-01` に適応するために変更するには、日付の変換関数を以下のように追加します:

1. `parse-date` 関数をライブラリウィンドウからマッピングにドラッグします。関数の詳細に関しては、次を参照してください：関数と作業する
2. 「挿入」メニューから、「関数」をクリックします。定数として `[Y,4-4][M,2-2][D,2-2]` を入力します。
3. 以下の接続を描きます:

接続元	接続先	目的
関数	`parse-date` 関数の `format` 入力アイテム	日付が解析されるフォーマットに従い関数を提供します。
`F1251`	`parse-date` 関数の `value` 入力アイテム	フォーマットする実際の値を関数に提供します。
`parse-date` 関数の `result` 出力アイテム	XML ターゲットの `DateOfBirth` 入力アイテム	`DateOfBirth` XML ノードに対する関数に返された値を提供します。

4. サンプルのマッピングに表示されるように残りのマッピング接続を描きます（リストするには、接続の数が多すぎます。基本的には、各接続はHIPAA ソースファイルからターゲット XML ファイルに値をコピーします。マッピングの接続に関する基本の上は、次を参照してください：接続と作業する）。

6.5.8.2 HIPAA トランザクションのマッピング時に考慮する点

HL セグメント階層構造

HL セグメントが自動的に生成され、ユーザーが手動でマッピングを行う必要はありません。必要な場合、値のマッピングを行うこともできます。

自動補完

HIPAA 変換に使用される構成ファイルには数多くのコードが含まれており、フィールドの自動補完に使用されます。自動補完は状況に応じて変化するもので、フィールドの位置により異なる値が含まれることもあります。コンポーネント設定ダイアログボックスにある「空のフィールドを自動補完」チェックボックスのチェックを外すことで、自動補完を無効にすることができます（次を参照: EDI コンポーネント設定）。

検証

状況を意識するという意味で、検証と自動補完は共通しています。検証エラーは、X12 検証と同様に扱われます（次を参照: ASC X12 検証ルール）。

構成ファイルの編集

HIPAA 構成ファイルは、X12 構成ファイルより先新しいスキーマバージョンを使用します（次を参照: EDI 構成スキーマ）。このため、HIPAA 構成ファイルは、X12 構成ファイルに類似しています:

- フィールドレベルのコード値
- 条件
- 補完プラグ
- 「使用されない」データ要素

コード値

フィールドインスタンスには、検証や自動補完に使用される内蔵のコードリストを含めることができます。値は手動で追加、削除することができます。（次を参照: データ要素のコードの値の変更の可能性）。

```
<Data ref="F365" info="Communication Number Qualifier" nodeName="F365_1">
<Values>
        <Value Code="EM" />
        <Value Code="FX" />
        <Value Code="TE" />
</Values>
</Data>
```

条件

条件は値とともに使用されます。繰り返されている複数のセグメントやグループ（ループ）が存在しうるため、特定の条件に従ってそれらを識別する必要があります。条件には値を含める必要があり、値が含まれない場合、特定のグループを識別することは不可能になります。

例えば、Loop1000 が複数回繰り返されており、各インスタンスがセマンティック上で異なる意味を持つような状況が考えられます。

どれが目的の要素なのか識別するため、以下のコードにより NM1 セグメントのフィールド F98 にある "41" という値コードを持った Loop100A が MapForce で識別されます。このような条件に合致する要素が存在しない場合、Loop1000A が発見されることはありません。

```
<Group name="Loop1000A" info="Submitter Name">
  <Segment name="NM1" info="Submitter Name">
    <Condition path="F98" />
    <Data ref="F98" info="Entity Identifier Code">
        <Values>
          <Value Code="41" />
        </Values>
    </Data>
(…)
```

条件コードに値が含まれる場合、条件コードがターゲットコンポーネントにて自動生成されます。

メソッドにより生成された構成ファイルにより、繰り替えされたセグメント/ループのシーケンスにて条件の値が一意であることが保証されます。

条件やその中にある値を編集する場合、値によりターゲットが一意に定まることを考える必要があります。

補完プラグ

ファイルレベルとともに、自動補完の調整を行うことができます。新たに用意された "Completion" 要素に記述される 3 つの属性により、以下にあるような定義が行われます:

(C) 2015-2021 Altova GmbH

- singleConditions – ターゲットコンポーネント内にある全てのフィールドに対して、単一の条件を自動補完します。
- singleValues – ターゲットコンポーネント内にある全てのフィールドに対して、単一の値を自動補完します。
- HL – ターゲットコンポーネント内にあるHL セグメントにて、適切なフィールドを生成します(全 HL セグメントが対象)。

  (Where "1" means true)

```
<Message>
  <MessageType>837-Q2</MessageType>
  <Completion singleConditions="1" singleValues="1" HL="1" />
  <Description>Health Care Claim: Dental</Description>
(…)
```

"Not Used" データ要素

標準的な X12 トランザクションでは表示されるオプション要素のいくつかは、HIPAA にて省略されます。これらのオプション要素は MapForce コンポーネントにて隠されます。

検証エンジンのチェックにより、これら使用されないフィールドがソースファイルに存在しないことが保証されます。これらのフィールドは隠されており、ターゲットコンポーネントにマッピングすることができないため、ターゲット出力ファイルに表示されることもありません。

構成ファイルではこれらのフィールドに対して値が0 となっている maxOccurs 属性が定義されます。必要な場合は、maxOccurs 属性の値を手動で変更することにより、特定のフィールドを隠す(または表示する)ことができます。

```
<Data ref="F1037" minOccurs="0" info="Submitter Middle Name or Initial" />
<Data ref="F1038" minOccurs="0" maxOccurs="0" />
<Data ref="F1039" minOccurs="0" maxOccurs="0" />
```

検証のスコープ

MapForce ではセマンティックな検証はサポートされていません。状況に応じて意味が異なるフィールドはオプションとして扱われます。

6.5.9 HL7 Version 2

HL7 は医療アプリケーションのデータ交換のための業界の標準で、"Health Level Seven" の省略形です。MapForce はバージョン 2.2 から2.6 をサポートします。

MapForce 内のデフォルトの HL7 バージョンは 2.6 です。

追加 HL7 V2.2 – V2.5.1 XML スキーマと構成ファイルのための個別のインストーラーが Altova W eb サイト MapForce コンポーネントのページ(https://www.altova.com/ja/mapforce/download/libraries) で使用することができます。

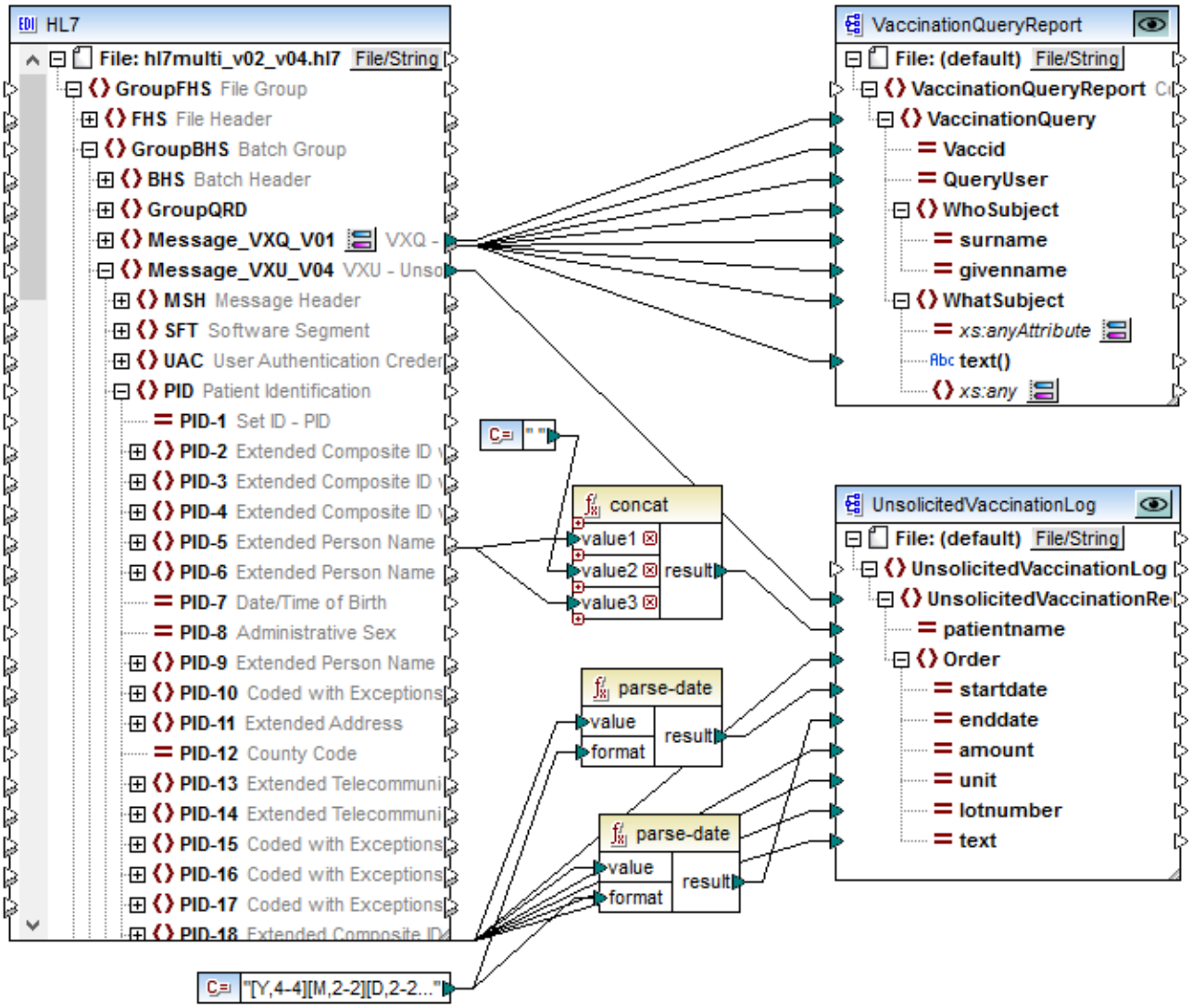
XML-ベースの HL7 バージョン 3.x は、XML スキーマコンポーネントを使用して、MapForce 2021 内でサポートされています。

MapForceExamples プロジェクトは、HL7 V2.6 から HL7 V3 XML ファイルへマップするサンプル(HL7V260_ to_ HL7V3.mfd)が含まれています。

HL7 に関しての詳しい情報は、http://www.hl7.org/ を参照してください。

6.5.9.1 例:HL7 から XML へのマッピング

このサンプルは、2つのHL7 メッセージ("Query for vaccination record" と"Unsolicited vaccination record update") を含むファイルの2つの個別のXML ターゲットファイルへマップする方法について説明しています。このマッピングサンプルは以下のパスで見つけることができます: <マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥**HL7_MultiMessageTypes.mfd**。



HL7_MultiMessageTypes.mfd

マッピングは下に示されているステップどおり作成することができます。

ステップ 1: ソース HL7 ファイルをマッピングに追加する

1. 「**挿入**」メニューから、「**EDI**」をクリックします。
2. "HL7" コレクションを選択し、メッセージ型 "VXQ - Query for vaccination record" と"VXU - Unsolicited vaccination record update" の横のチェックボックスを選択します。
3. インスタンスファイルを提供するように問われると、次のパスから使用することのできるソースファイルを開きます: <**マイドキュメント>\Altova\MapForce2021\MapForceExamples\hl7multi_v02_v04.hl7**.

(C) 2015-2021 Altova GmbH

4. プロンプトされると、必要に応じて、新規に追加されたHL7 コンポーネントの設定を調整します（詳細に関しては、次を参照してください：EDI コンポーネント設定）。

ステップ 2: マッピングにターゲットスキーマを追加する

1. 「挿入」メニューから、「XML スキーマ/ファイル」をクリックし、<マイドキュメント>\Altova\MapForce2021\MapForceExamples\ ディレクトリから **VaccinationQueryReport.xsd** ファイルを開きます。
2. サンプルXML ファイルを提供するように問われると、「スキップ」をクリックします。

マッピングコンポーネントとして **UnsolicitedVaccinationLog.xsd** を追加するには、同じ手順を踏みます。

ステップ 3: 関数を追加し、マッピング接続を描く

ソースHL7 ファイル内では、日付は、`YYYYMMDD` のフォーマットで表示されています。このフォーマットをターゲット XML ファイル `YYYY-MM-DD` に適応するために変更するには、日付の変換関数を以下のように追加します:

1. `parse-date` 関数をライブラリウィンドウからマッピングにドラッグします。関数の詳細に関しては、次を参照してください 関数と作業する
2. 「挿入」メニューから、「定数」をクリックします。定数として `[Y,4-4][M,2-2][D,2-2]` を入力します。
3. 以下の接続を描きます:

接続元	接続先	目的
定数	**parse-date** 関数の format 入力アイテム	日付が解析されるフォーマットに従い関数を提供します。
RXA-3	**parse-date** 関数の value 入力アイテム	フォーマットする実際の値を関数に提供します。
parse-date 関数の result 出力アイテム	XML ターゲットの startdate 入力アイテム	startdate XML ノードに対する関数に返された値を提供します。

新規 **parse-date** 関数と同じステップを踏み、RXA-4 の値を変換するために、ターゲット XMLの enddate アイテムにコピーします。

2番目のターゲット XML 内の patientname アイテムは、ソースHL7 ファイルの2つのフィールドを結合することにより作成されます: FN-1（Surname）とXPN-2（Given Name）。これを達成するには、ライブラリウィンドウから、**concat** 関数をマッピングにドラッグします。また、2つの名前を分けるスペース文字を提供する定数を追加します。

サンプルのマッピングに表示されるように残りのマッピング接続を描きます（リストするには、接続の数が多すぎます。基本的には、各接続はソースHL7 ファイルからターゲット XML ファイルに値をコピーします。マッピングの接続に関する基本の上は、次を参照してください 接続と作業する）。

ターゲット XML コンポーネントにより生成された出力をプレビューするには、「プレビュー」（ ）ボタンをクリックし、「出力」タブをクリックします。

6.5.10 IATA PADIS

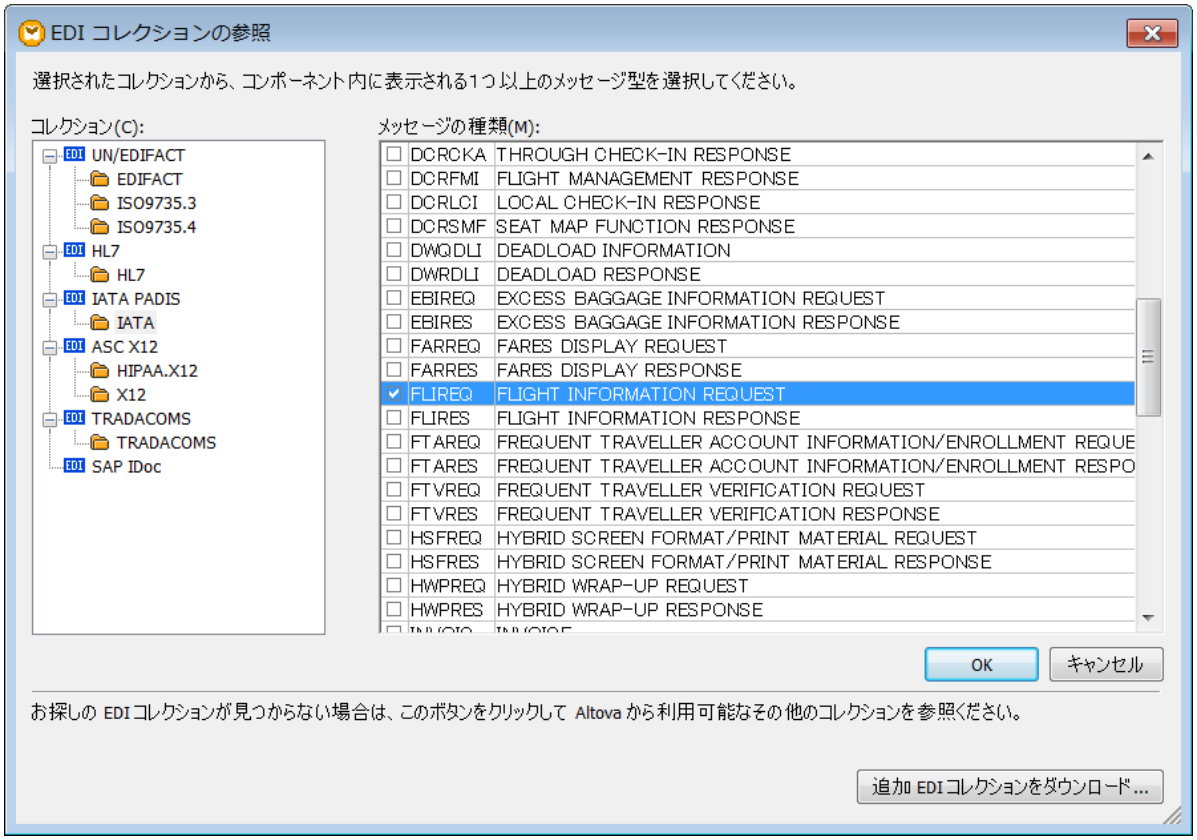
PADIS（Passenger and Airport Data Interchange Standards）は、EDIFACT（ISO 9735）構文を使用したメッセージのセットです。

- MapForce は、現在メッセージヘッダーとトレーラーセグメントを使用したメッセージのみをサポートしています。

- MapForce は、IATA.00.1 からIATA.08.1. までのコレクションをサポートしています。

IATA PADIS ファイルをマッピングコンポーネントとして追加する

1. 「挿入」メニューから、「EDI」をクリックします。
2. "IATA PADIS" から "IATA" コレクションをクリックし、必要なメッセージ型の横のチェックボックスを選択します)。

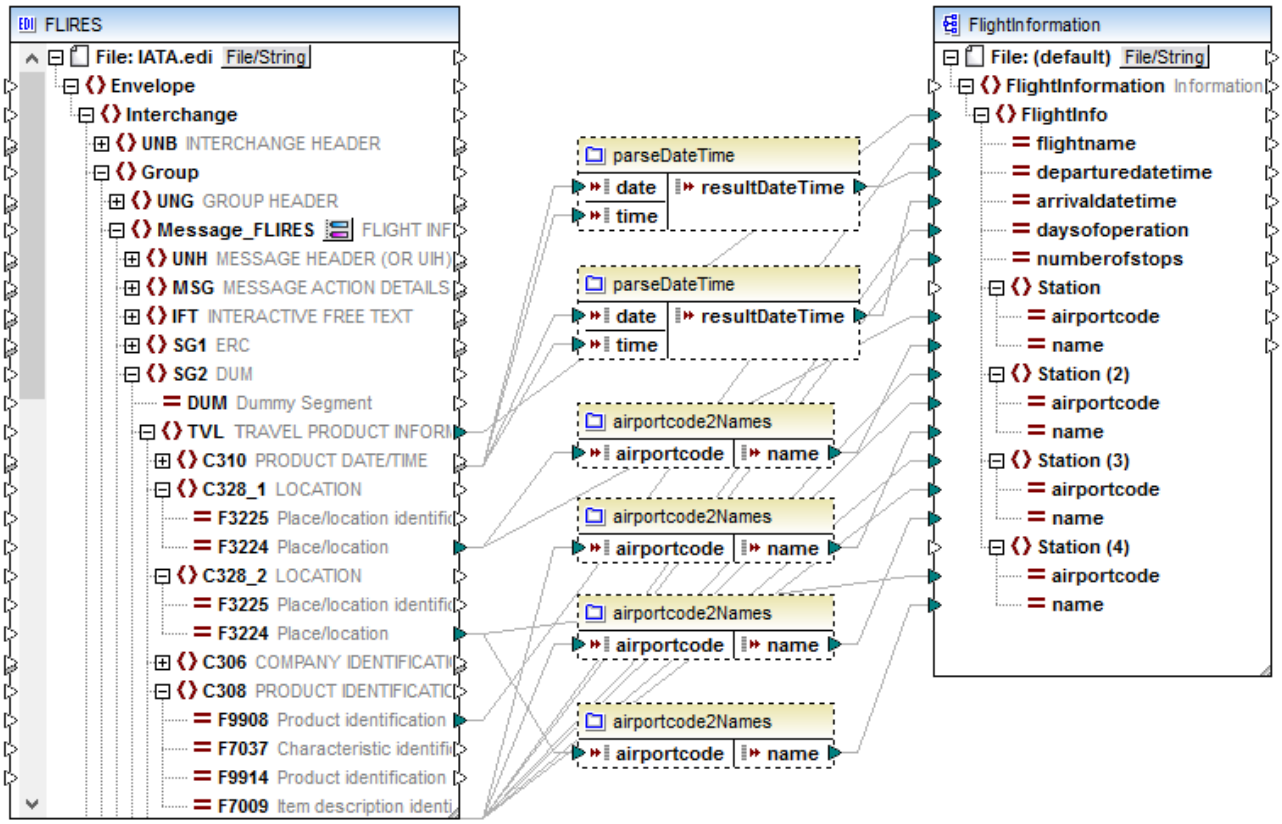


3. インスタンスファイルを提供するようにプロンプトされると、「参照」をクリックして、ソースIATA ファイルをマッピングソースとして使用するために開きます。IATA PADIS コンポーネントにマッピングする場合、「スキップ」をクリックします。

IATA コンポーネントがマッピングに追加され、マッピングの接続を描くことができます。IATA コンポーネントの設定を変更するには、右クリックして、コンテキストメニューから「プロパティ」を選択します(次も参照: EDI コンポーネント設定)。

6.5.10.1 例: IATA を XML にマッピングする

<マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥フォルダー内の **IATA_FlightInformationReport.mfd** ファイルは、 XML ターゲットファイルにIATA PADIS ファイルをマッピングする例を示しています。



IATA_FlightInformationReport.mfd

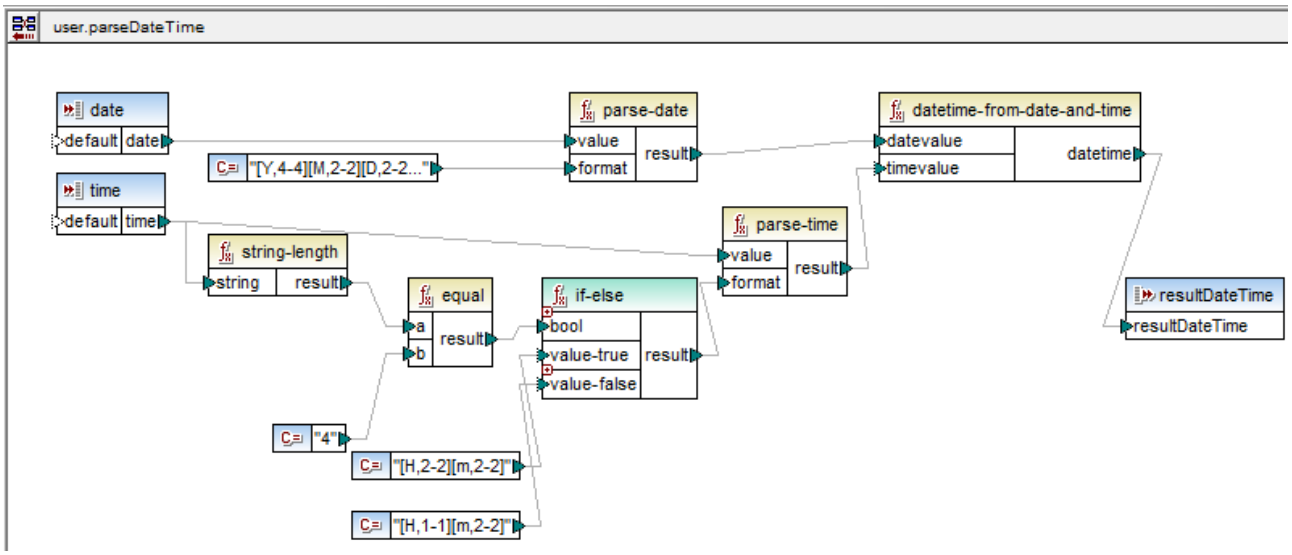
このマッピングは、FLIRES メッセージ、ターゲット XML コンポーネント、および 2 つのユーザー定義関数 `parseDateTime` と(複数回使用されている)`airportcode2Names` を含むソース IATA コンポーネントにより構成されています。

ソース、またはターゲットコンポーネント、および他の設定からインスタンスファイルへの読み込み、および書き込みを確認するには、コンポーネントを右クリックして、「プロパティ」を選択してください(次も参照: EDI コンポーネント設定)。

各 ユーザー定義関数を確認するには、ダブルクリックします。メインのマッピングに戻るには、マッピングエリアの左上にある「メインのマッピングに戻る」( ) ボタンをクリックしてください。詳細に関しては、次を参照してください: ユーザー定義関数。

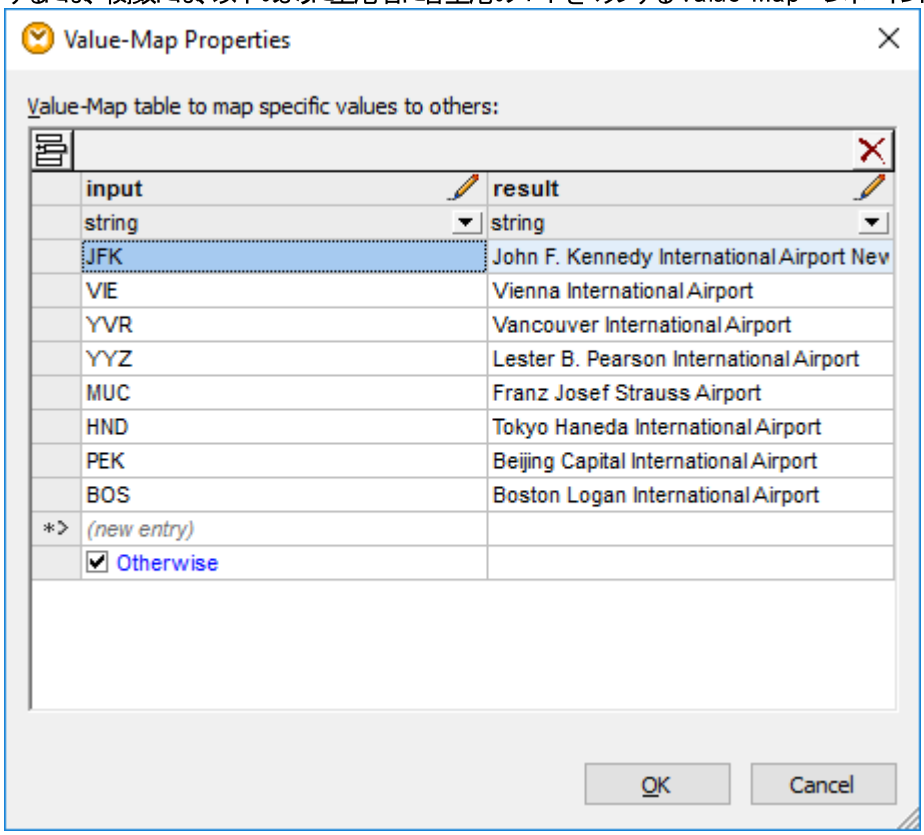
このマッピング内では、ユーザー定義関数には以下の目的があります:

1) `parseDateTime` 関数は、ターゲット XML ファイル内で必要とされる `xs:dateTime` 値を処理するために、引数として、IATA 日付と時刻を日付および時刻要素から受け取ります。このゴールを達成するために、MapForce コアライブラリ内の多種の関数を活用することができます。



parseDateTime 関数の構造

2) **airportcode2Names** 関数は、F3223 または F3224 フィールド値の引数として受け取られ、空港の名前を返します。これを達成するには、関数には、以下のように空港名に各空港のコードをマップする Value-Map コンポーネントが含まれている必要があります:



Value-Map コンポーネントの構成

Value-Map コンポーネントに関する詳細は、次を参照してください Value-Map を使用する。

(C) 2015-2021 Altova GmbH

ターゲット側では、XML コンポーネントは、"ノードの複製"呼ばれる技術を使用しています。具体的には、ノード `Station` がマッピングコンポーネント上で複数回複製されることを意味します。これにより、同じターゲットアイテムに複数のソースアイテムからマップ値をマップすることができます（次を参照: 入力の複製）。

マッピングにより生成された出力をプレビューするには、「出力」タブをクリックします。

6.5.11 NCPDP SCRIPT

SCRIPT はNCPDP (National Council for Prescription Drug Programs) により開発された米国の医療分野での電子情報の交換のための標準です。詳細に関しては、次を参照してください: http://www.ncpdp.org。MapForce は、SCRIPT フォーマットと、XML、CSV、Microsoft Excel、データベースなどの MapForce によりサポートされる他のフォーマットのファイル間のマッピングデータをサポートします。サポートされるNCPDP SCRIPT バージョンは、以下のとおりです: 8.1、10.6、10.10。

MapForce 内のデフォルトのNCPDP SCRIPT バージョンは 10.10. です。

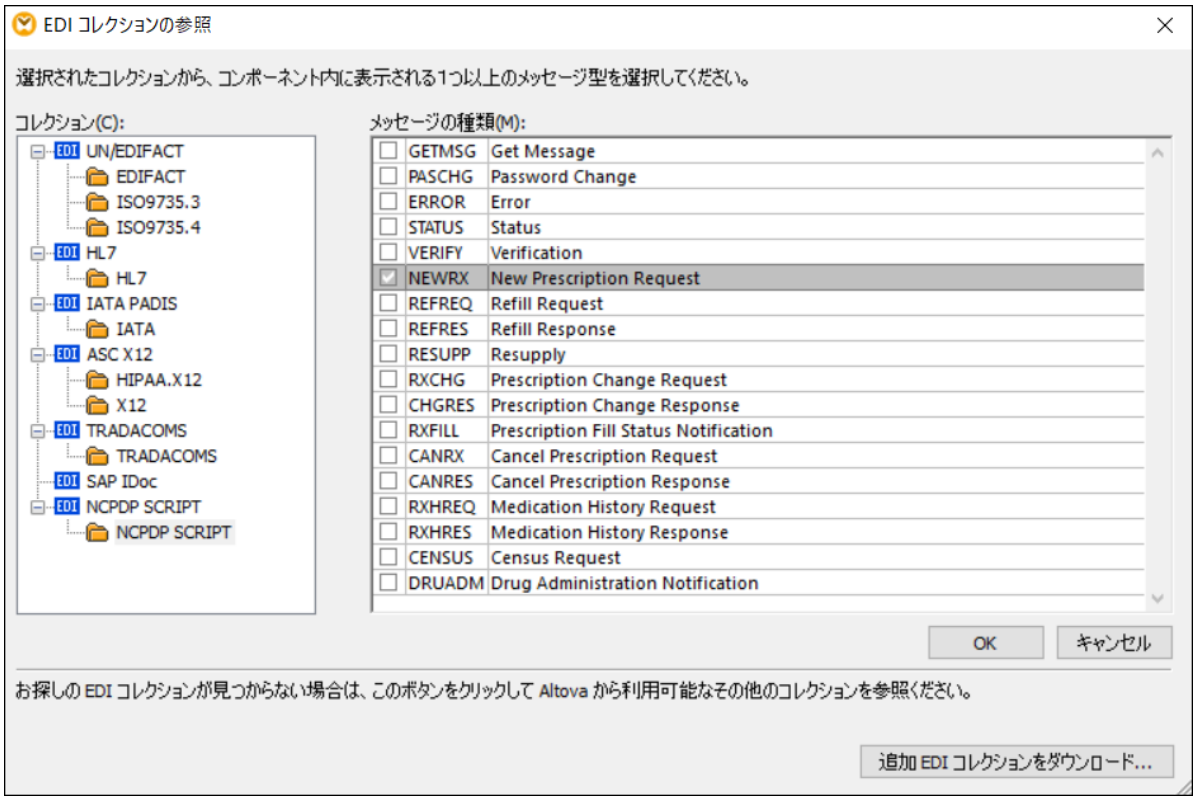
メモ デフォルトのバージョン以外のNCPDP SCRIPT バージョンをサポートするには、「NCPDP SCRIPT 構成ファイル」パッケージを Altova ダウンロードセンターからダウンロードしてください (https://www.altova.com/ja/mapforce/download/libraries)。

NCPDP SCRIPT はXML フォーマット 、またはUN/EDIFACT フォーマットを使用することができます。このドキュメントはNCPDP SCRIPT のUN/EDIFACT バージョンに適用することができます。NCPDP XML スキーマが存在する場合、MapForce は、標準の XML マッピング コマンドと検証 を使用したSCRIPT への、またはSCRIPT からのマッピングデータをサポートします（ XML ファイルをマッピングに追加する場合、メニューコマンド「挿入 | XML スキーマ/ファイル」を使用します）。

MapForce はSCRIPT メッセージをSCRIPT パーティ間のデータの通信の一部としてを送信、または受け入れないことご注意してください。MapForce の役割は、SCRIPT フォーマット内のデータを読み取り、他のフォーマットにマップし、または他のフォーマットからのデータをSCRIPT フォーマットに変換することです。

UN/EDIFACT フォーマットのNCPDP SCRIPT ファイルは、次のようにご追加することができます:

1. 「挿入」メニューから「EDI」をクリックします。
2. 「NCPDP SCRIPT」コレクションを選択し、必要なメッセージ方の横のチェックボックスを選択します。



3. 「OK」をクリックします。インスタンスファイルを提供するようにプロンプトされます。
4. 以下を行います
   - NCPDP SCRIPT メッセージから他のフォーマットにデータをマップする場合、「参照」をクリックして、ソースSCRIPT ファイルを選択します。URL からファイルを開くこともできます。次を参照してください: URL からコンポーネントを追加する
   - NCPDP SCRIPT メッセージにデータをマップ、または後にソースファイルを与える場合、「スキップ」をクリックします。

この段階では、選択されたNCPDP SCRIPT メッセージの構造はマッピング内に表示されています(MapForce の用語では、「EDI コンポーネント」と呼ばれます)。(データをマップするために)このコンポーネントにマッピング接続を描く、またはこのコンポーネントからマッピング接続を描くことができます。次も参照してください:

- EDI コンポーネントをマッピングに追加する
- EDI マッピングの実行
- EDI コンポーネント設定
- EDI マッピングの実行

6.5.11.1 NCPDP SCRIPT 検証ルール

NCPDP SCRIPT メッセージに関しては、MapForce は次の検証チェックをおこないます:

- `UIB/S001/F0001` は、"UNOA" です。
- `UIB/S001/F0002` は、"0" です。
- `UIH/S306/F0329` は、メッセージ型 "SCRIPT" を含みます。
- `UIH/S306/F0316` は、選択された構成ファイル内で定義されているメッセージバージョン番号を含みます。
- `UIH/S306/F0318` は、選択された構成ファイル内で定義されているメッセージリリース番号を含みます。
- `UIH/S306/F0326` は、メッセージ関数(または、MapForce 用語内のメッセージ型)を含みます。

- UIT/F0062 は対応する UIH/F0062 (メッセージ ヘッダー/トレーラー)に一致する、または存在しません(任意のフィールド)。
- UIT/F0074 はセグメントの数を含む、または、存在しません(任意のフィールド)。
- UIZ/F0036 はメッセージの数を含む、または、存在しません(任意のフィールド)。

次も参照してください: EDI コンポーネントの検証.

6.5.11.2 NCPDP SCRIPT 自動補完ルール

NCPDP SCRIPT ファイルを生成する際、MapForce は自動的に特定のフィールド、または値を(「自動データ保管」、または「自動補完」ルールと称される)ルールに従い生成します。振る舞いを無効化するには、EDI コンポーネント設定内の「不足しているフィールドの自動補完」チェックボックスをクリアします。

次のフィールドが自動的に補完されます:

インターチェンジ ヘッダー

- UIB/S001/F0001 = 制御機関 + 構文バージョン番号 = "UNOA"
- UIB/S001/F0002 = 構文バージョン番号 = "0"
- UIB/S002/F0004 = "Sender"
- UIB/S003/F0010 = "Recipient"
- UIB/S300/F0017 = 現在の日付
- UIB/S300/F0114 = 現在の時刻

メッセージ ヘッダー

- UIH/S306/F0329 = "SCRIPT", メッセージ型
- UIH/S306/F0316 =10.6 のための"010"、構成ファイルからのメッセージ バージョン番号
- UIH/S306/F0318 =10.6 のための"006" 構成ファイルからのメッセージ リリース番号
- UIH/S306/F0326 = メッセージ関数(または、MapForce パースペクティブからのメッセージ型)
- UIH/F0062 = "0"、メッセージ レファレンス番号

メッセージ トレーラー

- UIT/F0062 = UIH/F0062
- UIT/F0074 = メッセージ内のセグメントの数

インターチェンジ トレーラー

- UIZ/F0036 = メッセージ数

6.5.12 SAP IDoc

SAP IDocs (中間伝票)ドキュメントは、SAP と非 SAP アプリケーション間におけるデータのやり取りに使用されます。ドキュメントは、異なるシステム間における中間データとして機能します

IDoc は以下のような構造を持ちます:

- コントロール レコード: 送信者や受信者、メッセージや IDoc の種類といった IDoc に関する制御情報が含まれます。制御レコードは、あらゆる種類の IDoc にて同じような書式になります。

- **データセグメント**: セグメントに含まれる実際のデータや、その他メタデータのヘッダー、セグメントの番号や種類、データが含まれるフィールドといったデータが含まれます。
- **ステータスレコード**: ドキュメントの現在ステータス(現在処理されているステージ)と、これから処理を行うステージに関する情報が含まれます。ステータスの書式は、全種類のIDoc にて共通しています。

Port 定義にある Version 番号にて、通信を行なっているシステムが定義されます。バージョン間における主な違いは、各要素にて使用される名前の長さと、拡張の使用に関するものです。SAP R3 バージョン 4.X では、これまでのバージョンには無かった長い名前(そして拡張)がサポートされます。

ポート バージョン 1: リリース 2.1 および 2.2.
ポート バージョン リリース 3.0、3.1、R/2 システム
ポート バージョン 3: リリース 4.x (デフォルト値)

MapForce において、IDoc コンポーネントはメッセージの種類に対して30文字、IDoc の種類に対して30文字、セグメントフィールドに対して27文字が割り当てられた固定長のファイルとして扱われます。

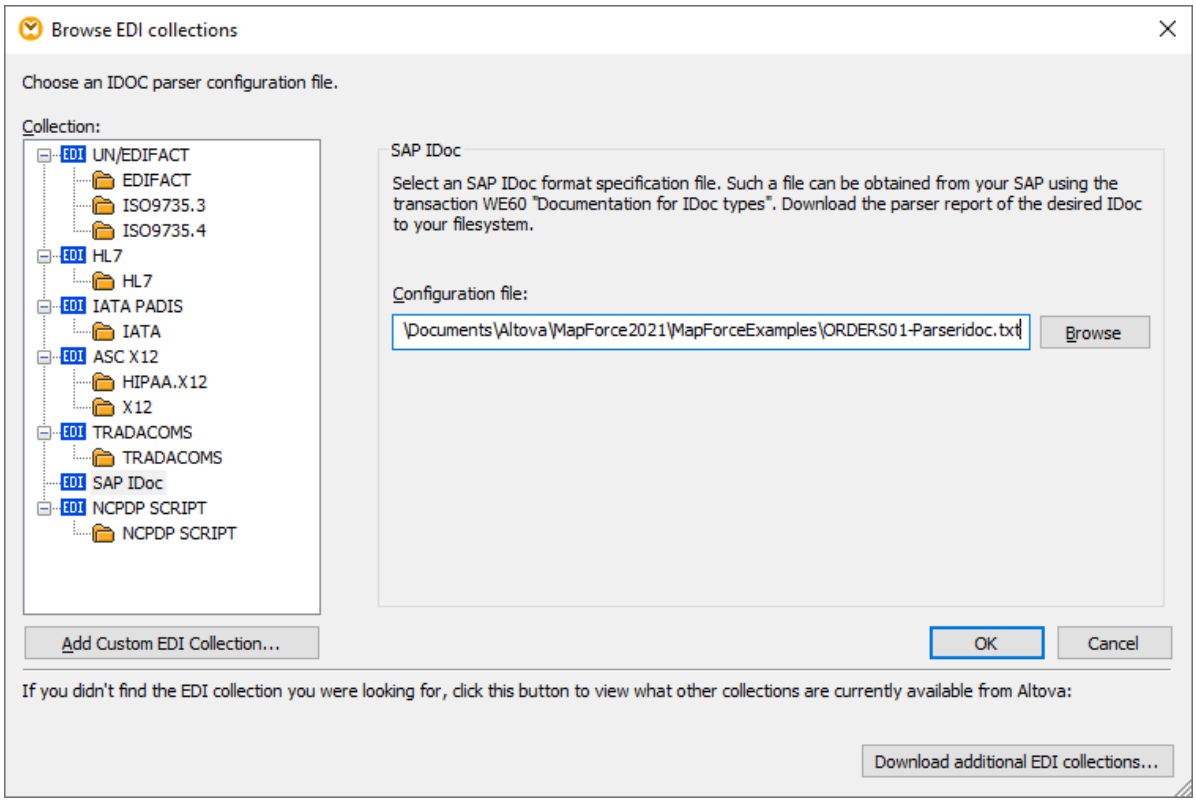
..¥MapForceExamples フォルダー内にある<マイドキュメント>**\Altova\MapForce2021\MapForceExamples\IDoc_Order.mfd** ファイルでは、SAP IDoc から XML スキーマターゲットファイルへのマッピングサンプルを確認することができます。

SAP IDoc ドキュメントをマッピングコンポーネントとして挿入する

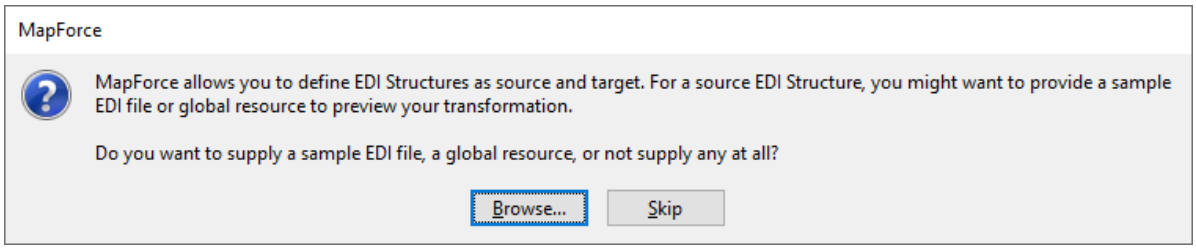
1. メニューオプションから「**入力 | EDI**」を選択します。EDI コレクションの参照 ダイアログボックスが表示されます。
2. リストボックスから **SAP IDoc** エントリーをクリックします。
3. 「**参照**」ボタンをクリックして、("IDoc 定義ファイル"としても知られる) IDoc パーサーレポートを選択します.

> IDoc パーサーレポートは、使用中のSAP システムからWE60 "Documentation for IDoc types" を使用することで得ることができます。SAP から圧縮されていない書式でファイルがエクスポートされる必要があります。WE60 を呼び出す方法に関する情報とレポートの解析に関しては、SAP IDoc ドキュメントを参照してください。IDoc_ Order.mfd サンプルに対して構成ファイルの**ORDERS01-Parseridoc.txt** <マイドキュメント>**\Altova\MapForce2021\MapForceExamples** フォルダーに収められています。

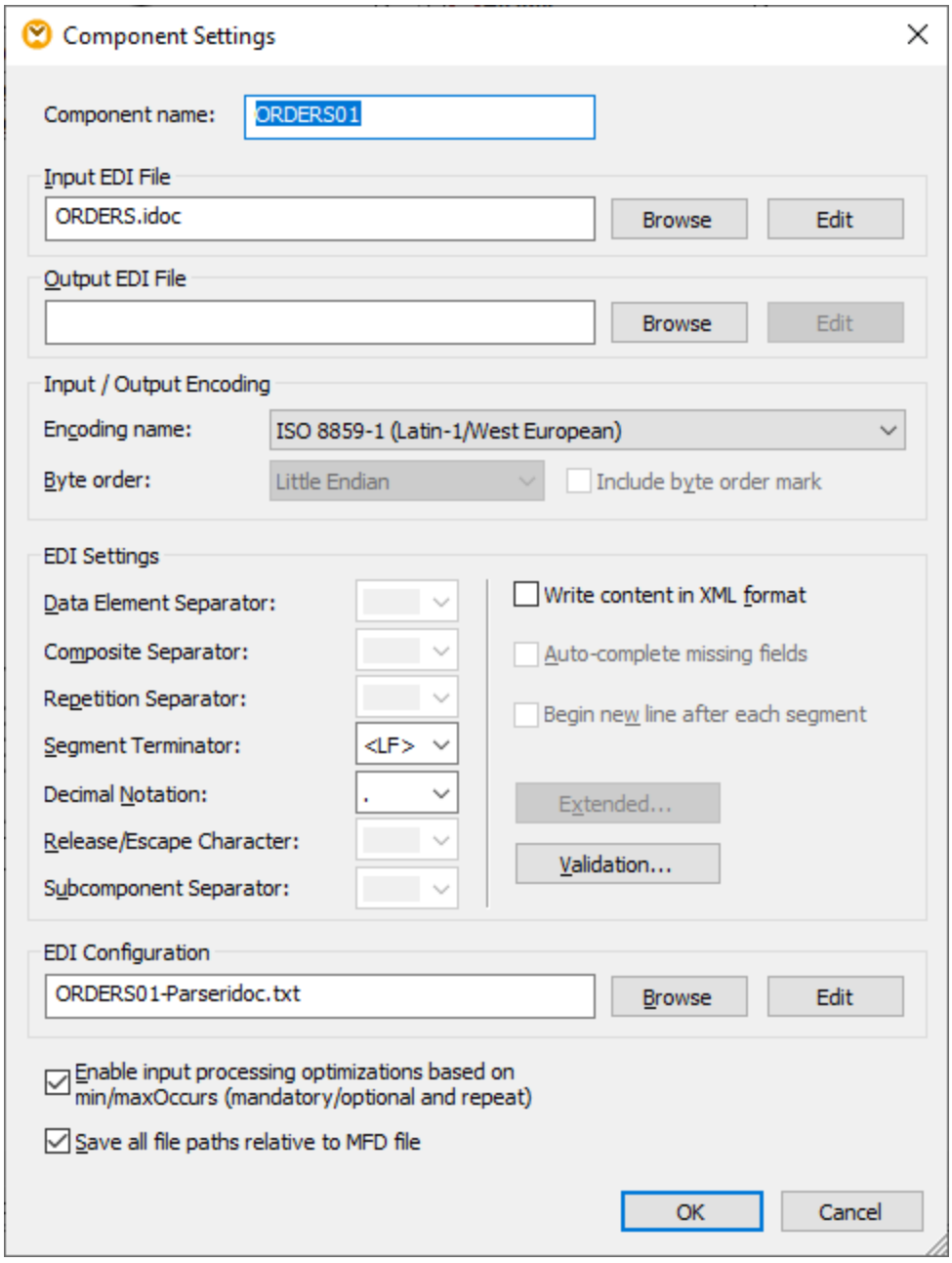
(C) 2015-2021 Altova GmbH



4. 「OK」をクリックして、EDI ファイルを選択します



5. IDoc ファイルからデータをマップする場合、「参照」をクリックして、データを提供するIDoc (*.idoc) ファイルを選択します。サンプルIDoc ファイルは、MapForce Examples フォルダーにあります (<**マイドキュメント>\Altova\MapForce2021\MapForceExamples\ORDERS.idoc** を参照する)。その他の場合は、「スキップ」をクリックします。
6. コンポーネント設定ダイアログボックスが表示されます。これにより、マッピングに追加する前にIDoc コンポーネントの設定をレビューすることができます。必要な場合この設定を後に変更することができます。EDI コンポーネント設定 を参照してください。



7. 「OK」をクリックして、コンポーネント設定ダイアログボックスを閉じます。IDoc コンポーネントが挿入されます。

6.5.13 SWIFT MT

SW IFT (Society for W orldwide Interbank Financial Telecommunication 国際銀行間通信協会 https://www.swift.com) は金融メッセージサービスのグローバルなプロバイダです。SW IFT MT (メッセージテキスト) は SW IFT ネットワークで使用されるメッセージ型です。MapForce は SW IFT MT データからによりサポートされる (XML、CSV、JSON、データベースなど) 他のフォーマットへのマッピング、および他のフォーマットからのマッピングをサポートします。MapForce によりサポートされる SW IFT MT リリースは以下のとおりです:

- 2020 年11月(**デフォルト**)
- 2019 年 11 月

メモ SWIFT MT からデータをマッピングする、または、バイナリファイルにデータをマッピングするには、変換言語として BUILT-IN が必要になります。MapForce 内でマッピングをプレビュー(し存在する場合は出力ファイルを保存)、または(個別にライセンス供与されている) MapForce Server を使用して異なるコンピューターまたはプラットフォームで実行することを選択できます。実行可能な C#, C++, または Java プログラムを SWIFT MT データを読み取るまたはバイナルファイルに書き込むマッピングから生成することはサポートされていません。

SWIFT MT を含むEDI-関連フォーマットからのデータのマッピング、または関連フォーマットへのデータのマッピングをサポートするために MapForce は各リリース固有の(EDI コレクションとして既知の) 構成ファイルを使用します。EDI コレクションはXML 書式のマシンで読解可能な構成ファイルを含むディレクトリです。デフォルトではEDI コレクションは**C:\Program Files\Altova\MapForce2021\MapForceEDI** ディレクトリにインストールされます。SWIFT に適用可能なコレクション(ディレクトリ)の名前は"SWIFT" で開始します。

サポートされる最新のSWIFT コレクションのみがデフォルトでインストールされることに注意してください。Altova W eb サイトから追加バージョンをダウンロードしてインストールすることが可能です。https://www.altova.com/ja/mapforce/download を参照してください。

制約

次の制約はSWIFT MT コンポーネントから、またはコンポーネントへデータをマッピングする際に適用されます。これらの制約は、MapForce またはMapForce Server などのマッピングを実行するエンジンに関わらず適用されます。

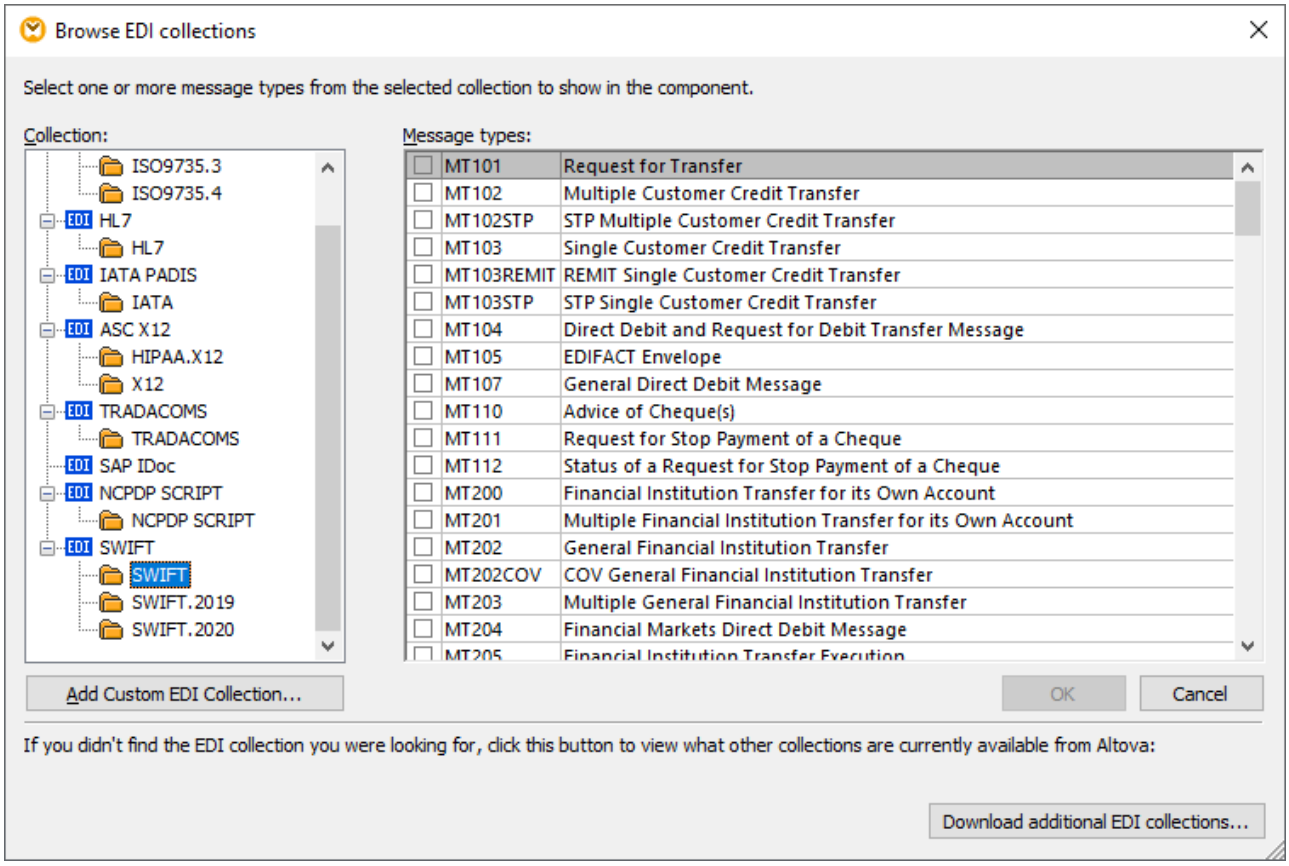
- 次のメッセージ型はサポートされていません
  - 非-FIN メッセージ
  - カテゴリ0 メッセージ(SWIFT システムメッセージ)
  - カテゴリn メッセージ(MTx9x)
- SWIFT MT メッセージを生成するには、コンテンツデータから自動的に生成されないためアプリケーションヘッダー内に正確なメッセージ型を提供する必要があります。また、正確なメッセージが他のためにデータがメッセージテキストボディ(ブロック4)にマップされていることを確認してください。
- MapForce はSWIFT MT コンテンツのバリエーションは提供しません。メッセージを解析するために必要な基本のチェックのみが実行されます。
- メッセージの最高の長さに関するチェックは行われません。
- 無効な入力の場合、パサーはエラーメッセージと共に停止され、回復オプションは存在しません。

SWIFT MT コンポーネントをマッピングに追加する方法

SWIFT MT メッセージには他のEDI メッセージとは大幅に異なる構造が存在します。しかしながらMapForce 内でSWIFT MT 構造を全てのEDI ベースのフォーマットに共通の方法でマップすることができます。具体的にはSWIFT MT コンポーネント(ソースまたはターゲット) をマッピングに以下のアプローチを使用して追加することができます:

- [**挿入**] メニューから **[EDI]** コマンドを選択します。
- 「**EDI の挿入**」 ツールバーボタンをクリックします。

[EDI コレクションの参照] ダイアログボックスで必要とされるSWIFT リリースを選択して、マッピングに追加する必要のあるメッセージ型に対応するチェックボックスを選択します。



メモ　デフォルトで、[SWIFT と呼ばれるジェネリックなコレクションのみが上記の[EDI コレクションの参照] ダイアログボックスで使用可能です。他のEDI に関連した標準同様にMapForce の新規のバージョンでバージョンが変更される可能性があるためこのデフォルトのSWIFT コレクションは生産にはお適していません。生産のためには、リリース固有のSWIFT 構成を Altova Web サイトからダウンロードしてインストールすることが奨励されます。https://www.altova.com/ja/mapforce/download を参照してください。インストール後、これらのリリース固有の構成はデフォルトの[SWIFT] 構成に加え[EDI コレクションの参照] ダイアログボックス内に表示されます。

EDI コンポーネントをマッピングに追加するも参照してください。マッピング例に関しては例: SWIFT MT SWIFT MT をSWIFT MX に変換するを参照してください。

SWIFT MT 設定

入力ファイル（または生成されたファイルのパス）SWIFT MT コンポーネントの他の設定を[コンポーネント設定]ダイアログボックスから変更することができます。[コンポーネント設定]ダイアログボックスを開くには、MapForce コンポーネントのタイトルバーをダブルクリックします。（またはタイトルバーを右クリックし、コンテキストメニューから[プロパティ]を選択します。）

このダイアログボックスで使用可能なすべてのオプションのレファレンスに関してはEDI コンポーネント設定を確認してください。上記の通り、コンポーネントがサポートされていないためSWIFT MT コンポーネントのための検証設定が無効化（灰色表示）されていることにご注意してください。

SWIFT MT データ型の扱い方

MapForce はSWIFT MT-固有の単純データ型を以下のように処理します:

(C) 2015-2021 Altova GmbH

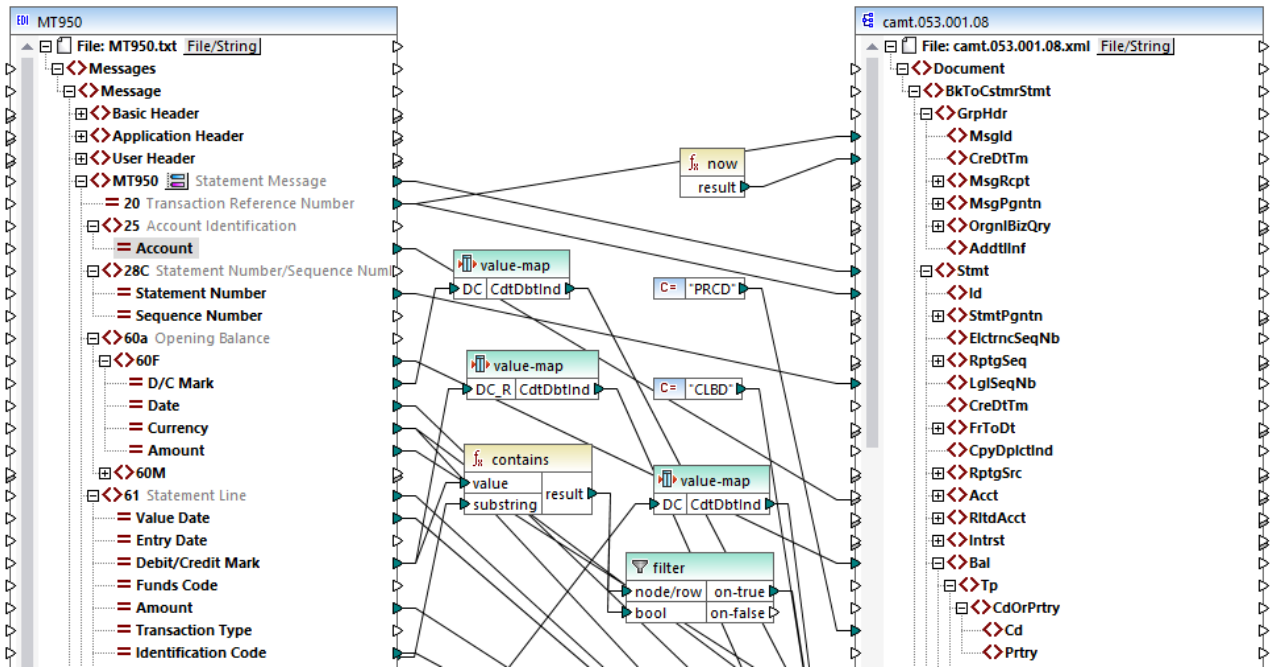
- SWIFT MT 日付型を型 `xs:date` のアイテムに、または `xs:date` を入力として取る関数に変換することができ、これは特別な処理方法を必要とせず正確に処理されます。例えば型 `swift:date6` のアイテムを文字列にマップすることができます。結果する文字列は、例えば、"20200922"になります。しかしながら同じ型を `xs:date` にマップすると、結果は型 `xs:date` の `2020-09-22` になります。
- ターゲットアイテムが文字列で、日付の表示を"2020-09-22" にする場合、of 型 `xs:date` の variable をマッピングに追加し、結果をターゲット文字列アイテムにマップします。
- `swift:decimal` 値 **42,66** を型 `xs:integer` のアイテムにマップすると、結果は **42.66** になります。しかしながら同じ型を `xs:string` にマップすると、結果は"42,66" になります。
- データをターゲットコンポーネントに書き込む場合も同じ振る舞いが当てはまります。例えば、`xs:decimal` 値 **15** を型 `swift:decimal` のアイテムにマップすると、結果は型 `swift.decimal` の `15` になります。

XML へのクイック変換

他のEDI-関連書式同様、マッピング接続を描くことなく SWIFT MT fファイルを XML に簡単に変換することができます。詳細に関しては EDI から XML を生成する を参照してください。

6.5.13.1 例: SWIFT MT を SWIFT MX に変換する

このサンプルは SWIFT MT メッセージを含むファイルを SWIFT MX フォーマットの XML ターゲットファイルにマップする方法を説明しています。この例には以下のパスで見つけることのできるマッピングが存在します: **<マイドキュメント>\Altova\MapForce2021\MapForceExamples\SWIFT_MT950_camt053.mfd**。このマッピングは大きなサイズであるため、マッピングの一部のみが以下で表示されています。



SWIFT_MT950_camt053.mfd

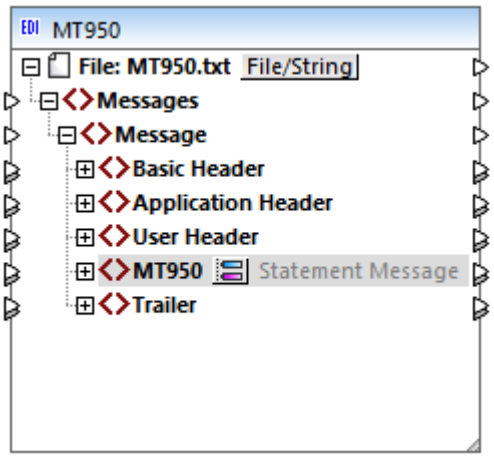
上で説明されているようなマッピングは以下のステップにより作成することができます。

ステップ 1: ソース SWIFT MT ファイルをマッピングに追加します。

1. [**挿入**] メニューから **[EDI]** をクリックします。

2. [SWIFT] を選択して、メッセージ型 [MT950 – Statement Message] を選択します。
3. インスタンスファイルを提供するようにプロンプトされると、以下のパスで使用可能なソースファイルを開きます: **<マイドキュメント>\Altova\MapForce2021\MapForceExamples\MT950.txt**。コンポーネントの設定 ダイアログボックスが表示されます。このサンプルのスコープのために、設定をそのままにし、**[OK]**を押すこともできます。これらの設定を後に必要に応じて変更することができます。EDI コンポーネント設定 を参照してください。

SWIFT MT コンポーネントがマッピングに追加されました。

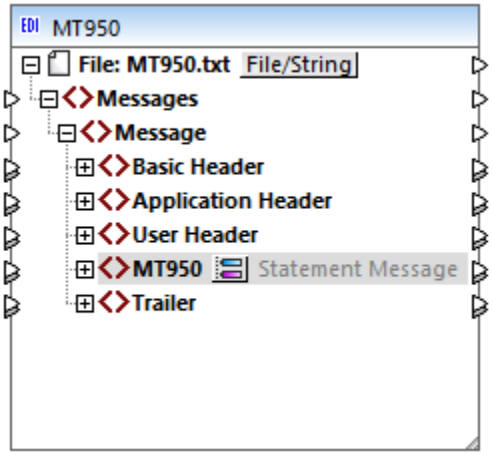


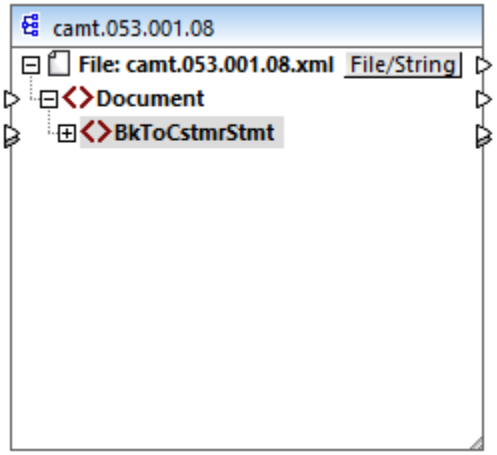
この時点で何も行っていない場合、マッピングをディスクに保存する必要があるかもしれません。マッピングデザインファイルをソースとターゲットファイルと同じディレクトリに保存することはグッドプラクティスです。このようにすると、ソースまたはターゲットファイルに対する全ての相対的なパスを管理することができます。ソースファイルとターゲットスキーマがロケートされる個所でこれはディレクトリであるためこのサンプルでは<マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥にマッピングを保存することができます。

ステップ 2: ターゲットスキーマをマッピングに追加します。

1. **挿入** メニューから XML スキーマ/ファイル をクリックして**camt.053.001.08.xsd** ファイルを **<マイドキュメント>\Altova\MapForce2021\MapForceExamples** ディレクトリから開きます。
2. サンプル XML ファイルを選択するように問われると、「スキップ」をクリックします。

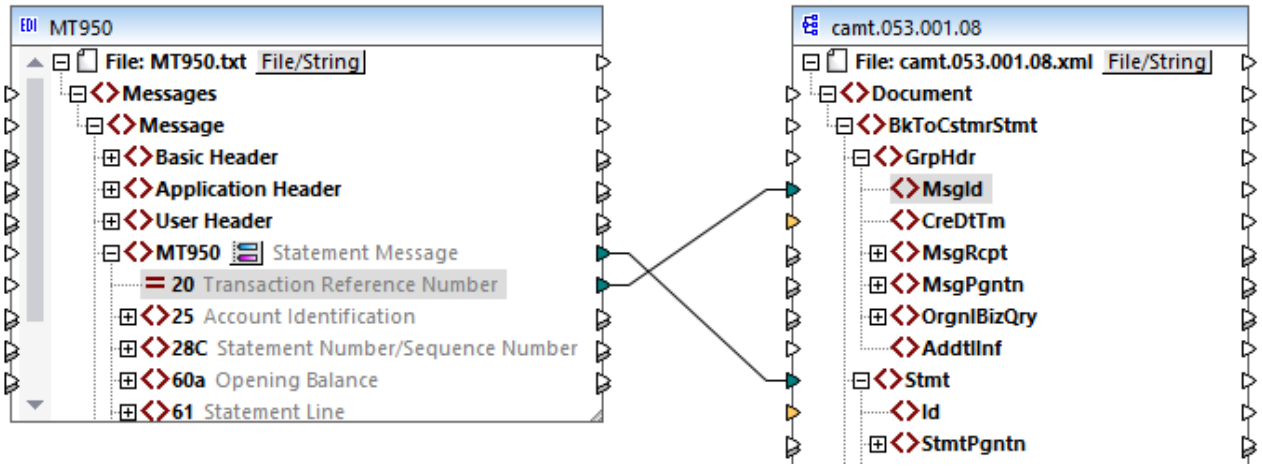
ターゲット XML コンポーネントがマッピングに追加されました。





ステップ 3: 中間コンポーネントを追加します

残りのマッピングはソースとターゲットコンポーネント間に接続線を描くことにより構成されています。例えば、ソース内のアイテム **MT950** はターゲット内にアイテム **Stmt** をマップしますが、下で説明されるとおり **20 (Transaction Reference Number)** は **MsgId** をマップします。MapForce の初心者の場合、接続線を描く詳細に関しては 接続との作業 とチュートリアルセクションを参照してください。



ソースからターゲットに少なくとも1つの接続が存在する場合 [出力] タブをクリックしてマッピング出力を既に確認することができます。結果する XML 出力は必須の接続がすべて入力を与えられるまでスキーマに対して有効ではありません。必須の接続はオレンジでハイライトされています（例えば、上のイメージ内の **CreDtTm** ）。

ソースからターゲットへの判り易い接続の他に、完全なマッピングのサンプル（**SWIFT_MT950_camt053.mfd**）にはいくつかの中間コンポーネントが含まれています。具体的には

1. 値マップコンポーネント（ルックアップテーブル）。これらは2つのセット間の値を変換する際に役に立ちます（例えば、[C] から [CRDT] へ、または、[D] から [DBIT] へなど）。詳細に関しては、Value-Map の使用を参照してください。
2. 関数 `now` と `contains`。`now` 関数は現在の日時をかえしますが、`contains` 関数は最初の引数に2番目の文字列の引数が含まれている場合にブール値 true を返します。詳細に関しては、マッピングに関数を追加するを参照してください。
3. （[PRCD], [CLBD] などの）多種の文字列定数に関しては 定数をマッピングに追加するを参照してください。

例を挙げると、以下のテーブルはターゲットコンポーネントからの接続が入力を受け取るかを説明しています。

ターゲットコネクタ	説明
`GrpHdr`	このターゲットアイテムは入力を明示的に受け取りませんが、子要素 **MsgId** は受け取ります（以下を参照）。このため、親 **GrpHdr** もマッピングを実行する際に作成されます。
`MsgId`	（**MT950**の下のアイテム**20**）ソース内の**Transaction Reference Number** から判り易い接続を受け取ります。
`CreDtTm`	現在の日時を返す MapForce ビルトイン関数である関数 now の直接の出力を受け取ります。
`Stmt`	**MT950** ソースアイテムから直接の接続を受け取ります。
`Stmt/Bal/Tp/CdOrPrtry/Cd`	テキスト定数 [PRCD] から入力を受け取ります。

マッピングの実行

MapForce 内で生成されたファイルをプレビューするには、「出力」タブをクリックします。下のイメージは結果する XML ファイルの一部のみを表示しています。

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <Document xsi:schemaLocation="urn:iso:std:iso:20022:tech:xsd:camt.053.001.08
3      <BkToCstmrStmt>
4          <GrpHdr>
5              <MsgId>MF12345</MsgId>
6              <CreDtTm>2020-09-28T14:09:53.277+02:00</CreDtTm>
7          </GrpHdr>
8          <Stmt>
9              <Id>MF12345</Id>
10             <LglSeqNb>42</LglSeqNb>
11             <Acct>
12                 <Id>
13                     <IBAN>NL58NANO3141592658</IBAN>
14                 </Id>
15             </Acct>
16             <Bal>
17                 <Tp>
18                     <CdOrPrtry>
19                         <Cd>PRCD</Cd>
20                     </CdOrPrtry>
21                 </Tp>
22                 <Amt Ccy="EUR">3723495</Amt>
23                 <CdtDbtInd>CRDT</CdtDbtInd>
```

マッピングの出力

マッピング出力ファイルに保存するには、以下を行います:

- 「出力」メニューで「出力ファイルを保存」をクリックします。
- [生成された出力を保存する] ツールバーボタンをクリックします。

MapForce Server を使用した自動化

ライセンス供与済みの MapForce Server が存在する場合、Linux、macOS、または Windows マシン上のコマンドラインでマッピングを実行することができます: サーバー上でマッピングを実行するには通常 サーバー実行のためにマッピングを準備する内で述べられているとおりいくつかの追加ステップを必要とします。この特定のマッピングをサーバー上で実行するためのステップは以下の通りです:

1. マッピングの空白の部分を右クリックし、コンテキストメニューから「マッピングの設定」を選択します。
2. [生成されたコード内でパスを絶対パスにする] チェックボックスをクリアして、同じディレクトリにマッピングを保存します。
3. [ファイル] メニューで、[MapForce Server 実行可能ファイルにコンパイルする] をクリックして、<マイドキュメント>\Altova\MapForce2021\MapForceExamples\ ディレクトリに .mfx ファイルを保存します。
4. .mfx ファイルをサーバーマシン上のディレクトリにコピーします ([作業ディレクトリ]と呼ぶことにします)。
5. ファイル<マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥**MT950.txt** をサーバーマシンに作業ディレクトリにコピーします。
6. ターミナルを開き、現在のディレクトリを作業ディレクトリに変更します。
7. MapForce Server を下のコマンドを使用して実行します。**mapforceserver_exec** は、使用中のオペレーティングシステムに適用することのできる MapForce Server 実行可能へのパスです。

(C) 2015-2021 Altova GmbH

```
mapforceserver_exec run SWIFT_MT950_camt053.mfx
```

サーバー実行の際は、API 呼出、またはFlowForce Server ジョブとしてマッピングを実行することができます。詳細に関しては MapForce Server 実行ファイルにマッピングをコンパイルするを参照してください。

6.5.14 SWIFT MX / ISO 20022

SWIFT MX /ISO 20022 標準はXML をベースにしています。このためこのようなコンポーネントを他のXML スキーマに追加する方法と同じ方法でマッピングに追加することができます。このようなデータをマップするために他の特別な必要条件はありません。

https://www2.swift.com/knowledgecentre/products/Standards%20MX で標準 SWIFT MX スキーマをダウンロードすることができます。

https://www.iso20022.org/iso-20022-message-definitions で標準 ISO 20022 スキーマをダウンロードすることができます。

これらは上で説明されているスキーマの非標準スキーマバリエーションです。有効な XML スキーマである限りすべてのXML スキーマを MapForce 内のマッピングコンポーネントとして使用することができます。

マッピング例に関しては例: SWIFT MT SWIFT MT をSWIFT MX に変換するを参照してください。

6.5.15 TRADACOMS

TRADACOMS (Trading Data Communications) は英国内で使用される小売業の電子データ交換の標準です。1982 に最初の EDI 標準として英国の経済と産業のために導入されました。 UN/EDIFACT 標準と多くの類似点を有することから、TRADACOMS は、EDIFACT の先駆の1つでありますが、メッセージ内で異なる構造を使用します。

MapForce は、ベースTRADACOMS 仕様を1993 年1 月に、現在ではGS1 UK (https://www.gs1uk.org) として知られる、英国 Article Numbering Association (ANA) により出版された「TRADACOMS Manual of Standards for Electronic Data Interchange」内で説明されているように実装します。その他のTRADACOMS バージョンのために、MapForce を、新規メッセージの型、データ要素、およびコード値を構成ファイルを使用して処理するようにカスタム化することができます。

MapForce 内で、TRADACOMS フォーマットと他のサポートされるEDI フォーマットと同様に以下のように作業できます:

- TRADACOMS ファイルをMapForce によりサポートされる他の全てのデータフォーマットで(データソースまたはデータターゲットとして)両方の方向にマップすることが出来ます。
- (マッピングのプレビューに使用される、またはMapForce Server とFlowForce Server 実行に使用される) BUILT-IN 言語とコード生成 言語 (C++、C#、Java) 内でTRADACOMS フォーマットにデータをマッピング、またはTRADACOMS フォーマットからデータをマッピングすることができます。異なるマシン上で動作するFlowForce Server にマッピングをデプロイする場合、デプロイされたパッケージは、自動的に選択されたTRADACOMS メッセージ型の構成とデータ要素により使用されたすべてのコードリストを含みます。同様に、生成されたコード内では、MapForce は、構成グループ、セグメント、データ要素のためにクラスを生成します。
- 拒否される(または受け入れられる)または無視されるレコードデータの変換を停止する検証イベントを柔軟に定義することができます。
- データの自動補完を有効化または無効化することができます。"自動的なデータの保管" (または"自動補完") が有効化されると、MapForce は、TRADACOMS ファイルの生成の際に、値を自動的にフィルします。
- メッセージの型、データ要素、およびコードリストを構成ファイルを使用し、グローバルまたはローカルでをカスタム化することができます (構成ファイルを参照)。
- 解析されたまたは生成されたファイルのエンコードを設定することができます。

以下のTRADACOMS ファイル型とメッセージの型がデフォルトでMapForce 内で使用することができます:

ファイル型	メッセージ型	バージョン
THE PRODUCT INFORMATION FILE	PROHDR (Product File Header)	8
	PROINF (Product Details)	8
	PROTLR (Product File Trailer)	8
THE PRICE INFORMATION FILE	PRIHDR (Price File Header)	8
	PRIINF (Price Details)	8
	PRITLR (Price File Trailer)	8
CUSTOMER INFORMATION FILE	CUSHDR (Customer Information Header)	8
	CUSINF (Customer Information Details)	8
	CUSTLR (Customer Information Trailer)	8
THE ORDER FILE	ORDHDR (Order File Header)	9
	ORDERS (Order Details)	9
	ORDTLR (Order File Trailer)	9
THE PICKING INSTRUCTIONS FILE	PICHDR (Picking Instructions File Header)	4
	PICKER (Picking Instructions File Details)	4
	PICTLR (Picking Instructions File Trailer)	4
THE DELIVERY NOTIFICATION FILE	DELHDR (Delivery File Header)	9
	DELIVR (Delivery Details)	9
	DELTLR (Delivery File Trailer)	9
DELIVERY CONFIRMATION FILE	DLCHDR (Delivery Confirmation Header)	5
	DLCDET (Delivery Confirmation Details)	5
	DLCTLR (Delivery Confirmation Trailer)	5
THE INVOICE FILE	INVFIL (Invoice File Header)	9
	INVOIC (Invoice Details)	9
	VATTLR (VAT Trailer)	9
	INVTLR (Invoice File Trailer)	9
THE CREDIT NOTE FILE	CREHDR (Credit Note File Header)	9
	CREDIT (Credit Note Details)	9
	VATTLR (File VAT Trailer)	9

(C) 2015-2021 Altova GmbH

ファイル型	メッセージ型	バージョン
	CRETLR (Credit Note File Trailer)	9
STATEMENT/REMITTANCE FILE	SRMHDR (Statement/Remittance Details File Header)	9
	SRMINF (Statement/Remittance Line Details)	9
	SRMTLR (Statement/Remittance Details File Trailer)	9
UPLIFT INSTRUCTION FILE	UPLHDR (Uplift File Header)	4
	UPLIFT (Uplift File Details)	4
	UPLTLR (Uplift File Trailer)	4
UPLIFT CONFIRMATION FILE	UCNHDR (Uplift Confirmation Header)	3
	UCNDET (Uplift Confirmation Details)	3
	UCNTLR (Uplift Confirmation Trailer)	3
THE STOCK SNAPSHOT FILE	SNPHDR (Stock Snapshot Header)	3
	SNPSTS (Stock Snapshot Details)	3
	SNPTLR (Stock Snapshot Trailer)	3
THE STOCK ADJUSTMENT FILE	SADHDR (Stock Adjustment Header)	3
	SADDET (Stock Adjustment Details)	3
	SADTLR (Stock Adjustment Trailer)	3
AVAILABILITY REPORT FILE	AVLHDR (Availability File Header)	4
	AVLDET (Availability Report Details)	4
	AVLTLR (Availability File Trailer)	4
GENERAL COMMUNICATIONS FILE	GENHDR (General Communications File Header)	3
	GENRAL (General Communications Text)	3
	GENTLR (General Communications Trailer)	3
COMPLEX ORDER FILE	CORHDR (Complex Order File Header)	6
	CORDER (Complex Order Details)	6
	CORTLR (Complex Order Trailer)	6
THE ACKNOWLEDGEMENT OF ORDER FILE	ACKHDR (Acknowledgement File Header)	4

ファイル型	メッセージ型	バージョン
	ACKMNT (Acknowledgement Details)	4
	ACKTLR (Acknowledgement Trailer)	4
PRODUCT PLANNING REPORT FILE	PPRHDR (Product Planning Report Header)	2
	PPRDET (Product Planning Report Details)	2
	PPRTLR (Product Planning Report Trailer)	2
THE PAYMENT ORDER FILE	PAYHDR (Payment Order File Header)	3
	PAYINF (Payment Order Line Details)	3
	PAYTLR (Payment Order File Trailer)	3
THE DEBIT ADVICE FILE	DRAHDR (Debit Advice File Header)	3
	DRAINF (Debit Advice Line Details)	3
	DRATLR (Debit Advice File Trailer)	3
THE CREDIT ADVICE FILE	CRAHDR (Credit Advice File Header)	3
	CRAINF (Credit Advice Line Details)	3
	CRATLR (Credit Advice File Trailer)	3
THE EXCEPTION CONDITION FILE	EXCHDR (Exception Condition File Header)	3
	EXCINF (Exception Condition Line Details)	3
	EXCTLR (Exception Condition File Trailer)	3
LOCATION PLANNING REPORT FILE	LPRHDR (Location Planning Report Header)	2
	LPRDET (Location Planning Report Details)	2
	LPRTLR (Location Planning Report Trailer)	2
THE UTILITY BILL FILE	UTLHDR (Utility Bill File Header)	3
	UTLBIL (Utility Bill File Details)	3
	UVATLR (Utility Bill VAT Trailer)	3
	UTLTLR (Utility Bill File Trailer)	3

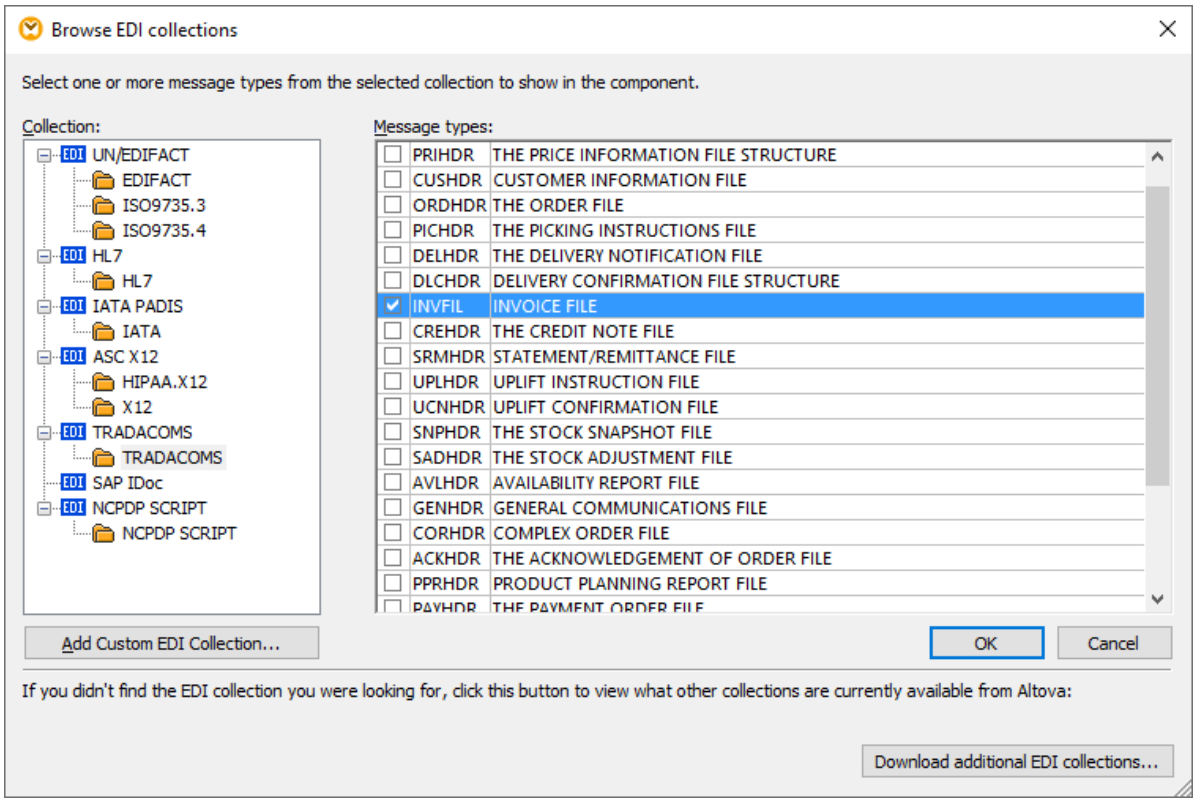
メモ RSGRSG（調整メッセージ）バージョン 2 もサポートされています。このメッセージの型は全てのファイルの型で発生する可能性があります。

(C) 2015-2021 Altova GmbH

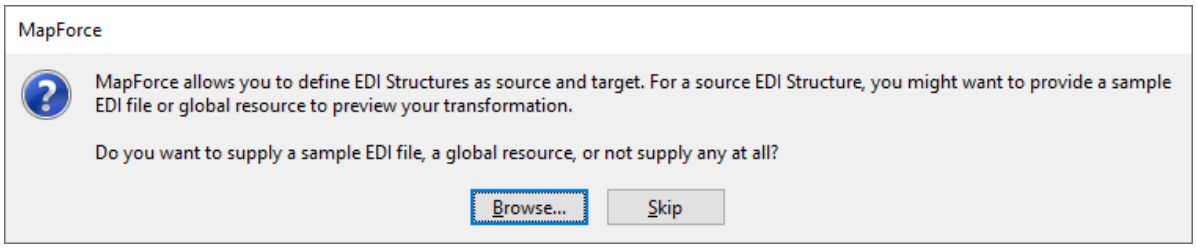
6.5.15.1 TRADACOMS ファイルをマッピングコンポーネントとして追加する

データマッピングソースとして、TRADACOMS ファイルを追加する:

1. 以下を行います:
   - 「挿入」メニューから「EDI」をクリックします。
   - 「EDI ファイルの挿入」( ) ツールバーボタンをクリックします。
2. 「EDI コレクションの参照」ダイアログボックスから、「TRADACOMS」をクリックして、メッセージに含まれるメッセージの型を選択します。



3. 「OK」をクリックします。インスタンスファイルを提供するようにプロンプトされます。



4. ソース TRADACOMS ファイルを後で指定する場合、「スキップ」をクリックします。それ以外の場合、「参照」をクリックして TRADACOMS ファイルの開き方を選択します:
   - ローカルドライブまたはネットワークからファイルを開く場合、ファイルを参照して、「開く」をクリックします。

- URL からファイルを開く場合、「URL に切り替える」をクリックします（詳細に関しては、URL からコンポーネントを追加するを参照してください）。
- ファイルを以前グローバルリソースとして定義した場合、グローバルリソースとしてファイル開きます「グローバルリソースに切り替える」をクリックします。（Altova グローバルリソースを参照）。

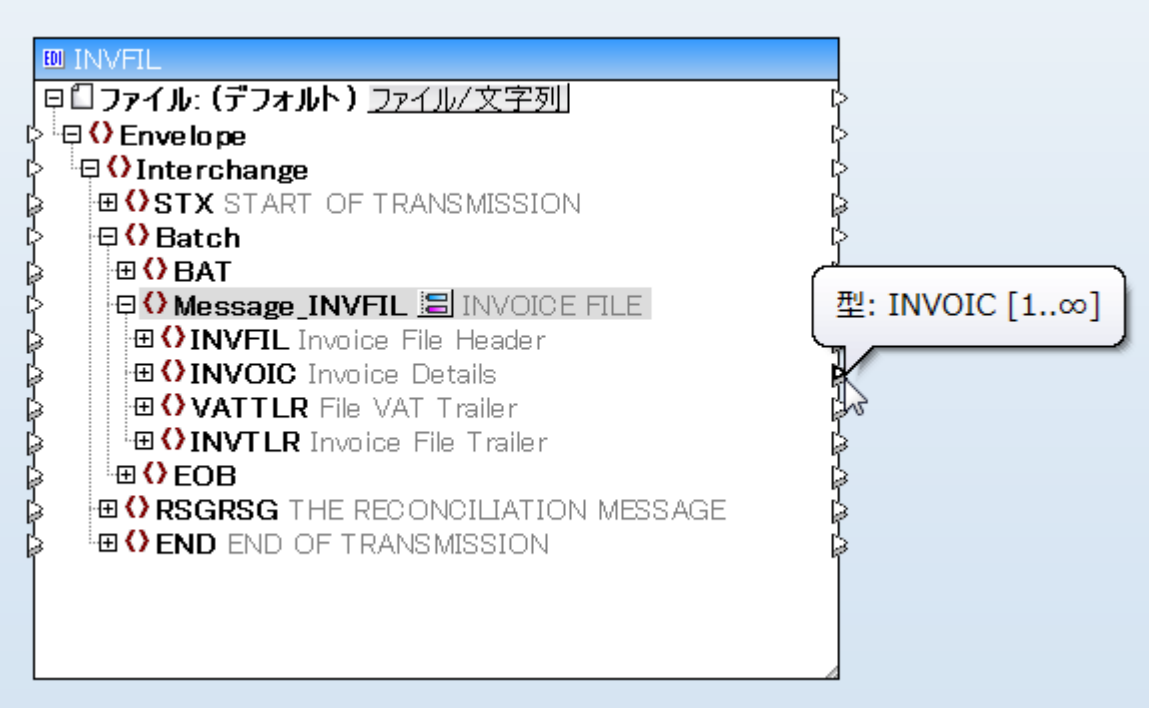
5. コンポーネント設定ダイアログボックスを設定または必要な場合設定を変更します（EDI コンポーネント設定 を参照）。

データマッピングターゲットとして TRADACOMS ファイルを追加する:

1. 上記のステップ 1、2、を繰り返します。
2. インスタンスファイルを提供するようにプロンプトされると、「スキップ」をクリックします。

6.5.15.2 MapForce 内の TRADACOMS コンポーネント

MapForce では、MapForce EDI に関連したコンポーネントに共通する全ての機能を保持しつつ、RADACOMS コンポーネントが、（任意のバッチ機能を使用して）TRADACOMS 伝送の構造をレプリケートします。



MapForce 内のサンプル TRADACOMS コンポーネント

コンポーネントの上のアイテム「ファイル」は、処理されている、またはコンポーネントにより生成された、TRADACOMS インターチェンジ ファイル名を表示します。"デフォルト" は、またファイルが割り当てられていないことを意味します。「ファイル/文字列」（File/String）ボタンは、すべてのファイルベースの MapForce コンポーネントに共通する高度な機能を、複数のファイルを動的に読み込み、処理するなどの他のオプションと共に表示しています（詳細に関しては、複数の入力または出力ファイルを動的に処理を参照してください）。

"エンベロープ" と "インターチェンジ" 構造 はすべての MapForce EDI に関連したコンポーネントに対してジェネリックです。適応できる場合、 MapForce コンポーネント内で複数のインターチェンジを処理できる機能を与えます。

STX と END 構造は、TRADACOMS フォーマットに対して特有であり、"伝送の開始" および "伝送の完了" セグメントを個別に表します。

BAT とEOB 構造は、TRADACOMS フォーマットに特有です。これらは"バッチの開始" および"バッチの完了" セグメントをそれぞれ表します。このようなセグメントは、TRADACOMS インターチェンジ ファイルからデータをバッチで読み込むの際に、またはファイルをバッチで生成する際に適用することができます。同じ伝送内に複数のバッチが存在する可能性があり、「バッチ」シーケンスは、BAT シーケンスの親として使用することができます。

Message_*Code* 構造（上のサンプル内の**Message_INVFIL**）は、TRADACOMS "ファイル"に対応します。「**EDIメッセージの型の選択**」ボタン（ ）は、ファイルの型を変更することのできるダイアログボックスを開きます（例: "請求書ファイル"、"注文ファイル" 等）。 既存のMapForce EDI 機能により、同じコンポーネント内に複数のファイル型を含むことができます。しかしながら、TRADACOMS 仕様は、各インターチェンジ伝送内で1つのファイルと1つのファイルの型を使用することを奨励しています。

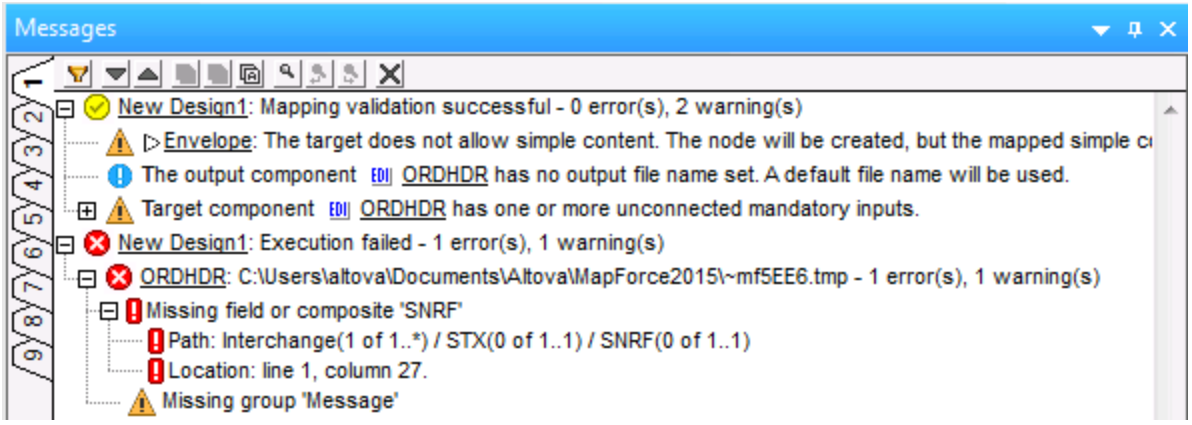
TRADACOMS ファイル（この例では、"請求書ファイル"）には、ヘッダーメッセージが含まれます。1つまたは複数の詳細メッセージ、トレーラーメッセージが含まれています。上のコンポーネントでは INVFIL（請求書ファイルヘッダー）、INVTLR（請求書ファイルトレーラー）、VATTLR（ファイルVAT トレーラー）および、複数の INVOIC（請求書の詳細）メッセージが表示されています。

MapForce コンポーネントのそれぞれの両脇に、入力と出力コネクタ（小さな三角形）が表示されており、それぞれのデータ要素、またはセグメントを MapForce によりサポートされる他のデータ型、またはフォーマットに接続することができます。「**表示 | ヒントの表示**」メニューでヒントの表示が有効化されていると想定して、コンポーネント上のアイテムにマウスをポイントすると、（発生の最高数および最小数など）それぞれに関する追加情報を確認することができます。

最後に、他の設定をコンポーネントヘッドをダブルクリック、またコンポーネント設定ダイアログボックスを開くことにより変更することができます（EDI コンポーネント設定 を参照）。このダイアログボックスは、すべてのEDI-に関連したコンポーネント型のためのジェネリックなダイアログボックスであり、このため、オプションの一部はTRADACOMS に適用できない可能性があります。

6.5.15.3 検証と自動データ補完

TRADACOMS 構造からデータを読み込む、またはデータを書き込むマッピングを実行すると、MapForce は、構造データの検証を行い、TRADACOMS 仕様に従いチェックし、メッセージウィンドウ内に全ての検証エラーを表示します。



検証エラーのあるメッセージウィンドウ

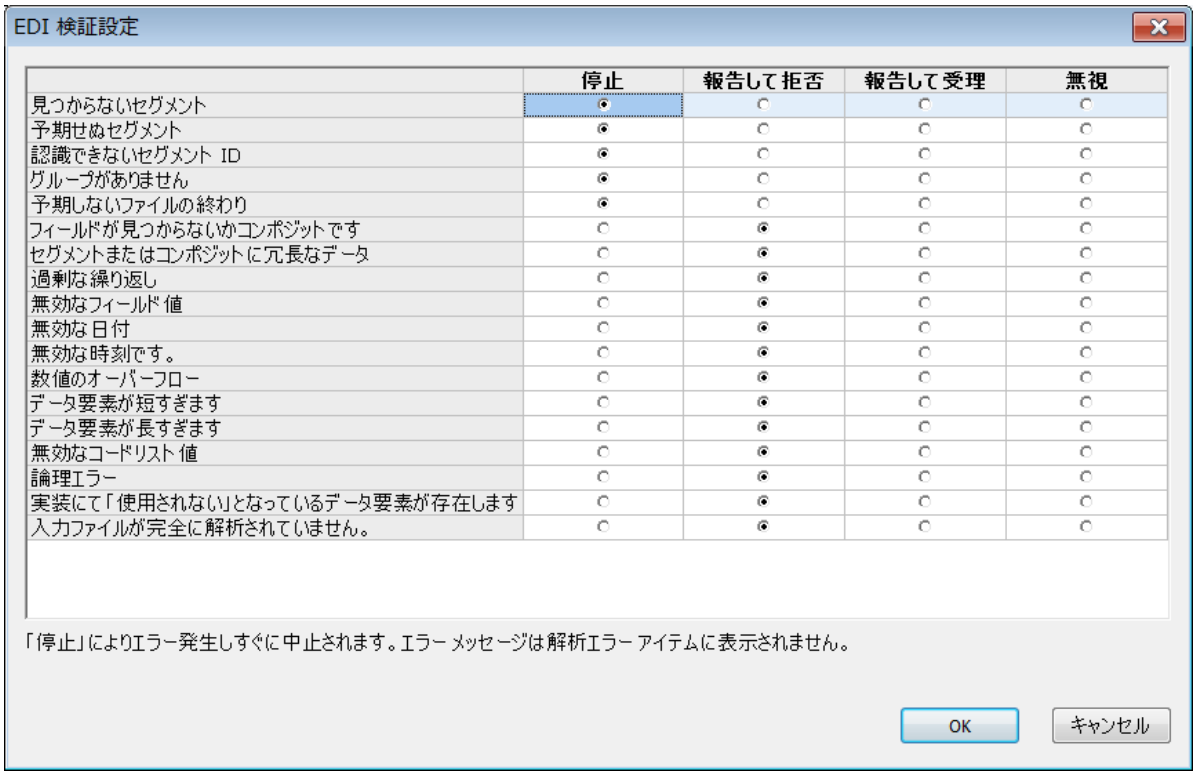
上のサンプルで示されるとおり、TRADACOMS に特定の検証メッセージ（情報、警告、とエラー）が、MapForce マッピングに共通するジェネリックな検証メッセージに加え表示されます。MapForce 検証全般に関しての詳しい情報は、マッピングの検証を参照してください。

以下の要素が、MapForce が解析された、または生成されたTRADACOMS ファイルをどのように検証するかを決定します:

- 構成ファイル内出定義された検証 制約は、MapForce インストールフォルダー内で使用することができます（サブフォルダー **MapForceEDI\TRADACOMS**）。これらの構成ファイルは、一方、TRADACOMS 仕様のデフォルトの検証ルールを与えます。その一方で、カスタム化された必要性にTRADACOMS フォーマットを適用する手段を与えます。特に、構成ファイル内

で定義されているデータ要素、セグメント、またはコード値を変更することは可能ですが、検証の結果とマッピングの実行に影響を与えます。構成ファイルに関しての詳しい情報は、構成ファイルを参照してください。

- MapForce に内蔵の検証ロジックは、構成ファイルを使用せずに実行される MapForce 内部データの整合性のチェックも含まれます。
- MapForce グラフィカルユーザーインターフェイスから定義したカスタム検証の設定（EDI コンポーネントの検証 を参照）。TRADACOMS を含む EDI コンポーネントの現在の EDI 検証設定を確認するには、コンポーネント設定ダイアログボックスのコンポーネントのヘッダーをダブルクリックし、「検証」をクリックします。



TRADACOMS 構造に書き込む際、MapForce は、値が計算することのできる、または事前に定義することのできる、データ要素のコンテンツを自動的にフィルします。この機能は"自動的な補完"（または"自動補完"）と呼ばれます。この振る舞いを無効化するには、コンポーネント設定ダイアログボックスから、「不足しているフィールドを自動補完する」チェックボックスのチェックを解除します（EDI コンポーネント設定 を参照）。

以下の TRADACOMS 検証ルールに従い、MapForce は、（ファイル解析 または生成中に）検証エラーを表示、またはファイル生成中に不足しているフィールドの自動補完を行います：

1. セグメント STX（伝送の開始）および END（伝送の完了）は、存在しなくてはなりません。
2. STDS-1 が値 'ANAA' を持つ場合、調整メッセージ（RSGRSG）が、伝送の終わり（END）の前に存在しなくてはなりません。それ以外の場合、調整メッセージ（RSGRSG）は、存在してはなりません。
3. STDS-1 が値 'ANAA' を持つ場合:
   a. 調整メッセージ内の RSGA の値は、STX segment 内の SNRF の値と同じでなくてはなりません。
   b. 調整メッセージ内の RSGB の値は、STX segment 内の UNTO-1 の値と同じでなくてはなりません。
4. TRDT-1 は、日付（YYMMDD）を含む必要があり、TRDT-2 は、伝送（現在の日時）の時刻（HHMMSS）を含む必要があります。
5. バッチヘッダー（BAT）が存在する場合、バッチトレーラー（EOB）も存在する必要があり、NOLI（バッチごとのメッセージ数）データ要素内でバッチ内のメッセージの数を確認することができます。
6. MSRF（メッセージ参照）データ要素 メッセージ内で ヘッダー（MHD）は、1 から開始される伝送内のメッセージの連番を含む必要があります。

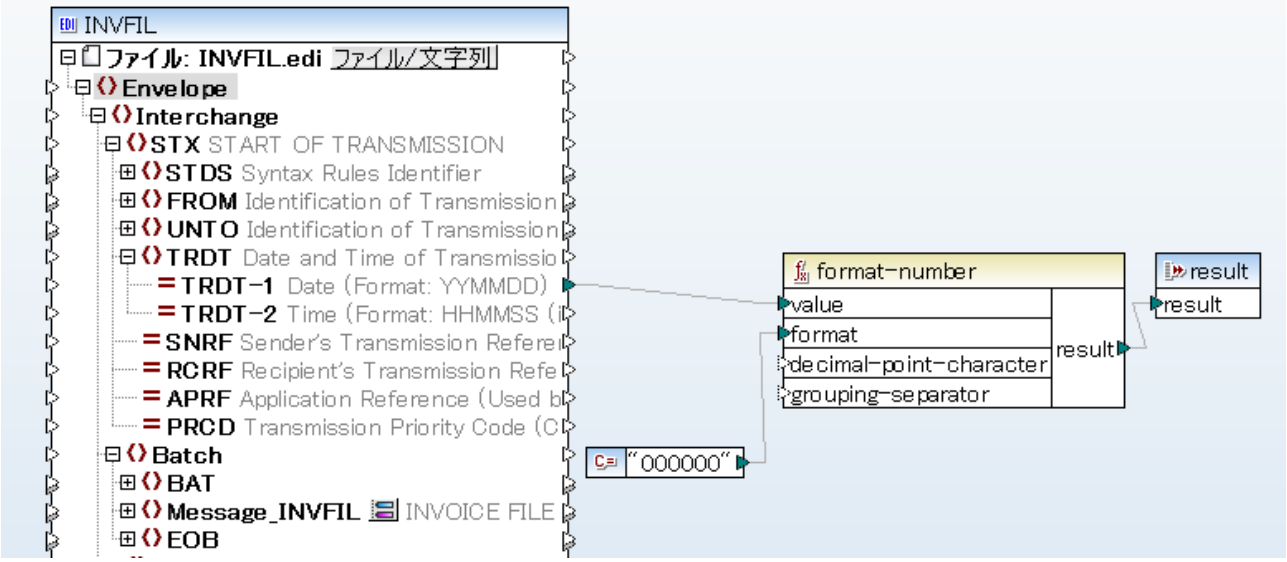
7. NOSG (メッセージ内のセグメントの数) データ要素 メッセージ内で Trailer (MTR) は、 MHD と MTR を含むセグメントの数を含む必要があります。
8. 存在する場合、調整メッセージ (RSGRSG) は、メッセージ ヘッダーとメッセージ トレーラーを含まない 1 つのセグメント (RSG) により構成されている必要があります。
9. END セグメント内の NMST (伝送内のメッセージの数) データ要素は、インターチェンジ (MHD セグメントの数) 内のメッセージの数を含む必要があります。
10. 全般的に、TRADACOMS 構造を読み取る際に MapForce は、インターチェンジ環境が型 "コンピューターからコンピューター" (または、in TRADACOMS 用語では、"インテリジェントなターミナルからインテリジェントなターミナル") であることを期待します。ですから、 `MHD = 12 + ORDHDR :3` などのセグメントは余分な行頭と末尾のスペースが含まれるため、検証エラーがトリガーされます。
11. 文字列データは大文字である必要があります。TRADACOMS 出力を生成する際、MapForce は、文字列データを大文字に変換します。

更に、前述のように、構成ファイルで定義された検証ルールは、TRADACOMS の解析と生成に影響を与えます。

6.5.15.4 変換中に先頭の 0 (ゼロ) を保持する

TRADACOMS 仕様では、TRDT-1 と TRDT-2 (伝送の日時) データ要素のデータ型は、"小数" として定義されています。これは、デフォルトでは、 TRDT-1 または TRDT-2 フィールドからデータをマップすると、これらのフィールドの先頭のゼロが変換中トリムされ、期待されない結果を生成するということを意味します。これは、"小数" と定義されており、値を文字列として扱うように定義されている他のアイテムでも同様の結果が出る可能性があります。

`core | format-number` MapForce 関数を適用することにより、MapForce に数値 フィールドを文字列として扱うように命令することができます (、故に、先頭のゼロを保持することが可能になります) 。下のサンプルでは、ソースファイル内の TRDT-1 アイテムの値は、"020312" です。(小数点の変換により) このアイテムの通常の出力は、期待されない結果である "20312" です。ですから、先頭のゼロを保持するために、 `format-number` 関数 がマッピングに追加されます。(関数との作業方法に関する詳しい情報は、関数と作業を参照してください) 。



format-number 関数を使用して先頭の 0 を保持する

上に示されるとおり、関数には次の 2 つの入力引数があります:

1. フォーマットされる値。(この場合、"020312")。
2. フォーマット マスク "000000"。

関数の出力をテストするには、この例は単純型出力コンポーネントを使用します（ 単純型出力を参照）。「出力」ボタンをクリックすると、マッピングの出力は期待される（関数の結果と同じ）"020312" となります。

6.5.15.5 構成ファイル

デフォルトでは、MapForce 構成ファイルと基となるMapForce 処理ロジックがTRADACOMS 標準（931）のベースバージョンを反映します。ですから、このTRADACOMS バージョンを使用する場合、TRADACOMS 構成ファイルのカスタマイズは必要ありません。しかしながら、既存の構成への変更をチェックすることは可能です。標準TRADACOMS 構成ファイルに加えられた変更は、MapForce がTRADACOMS 構造を解析する、または読み込む振る舞いに、直ちに影響が出ます。

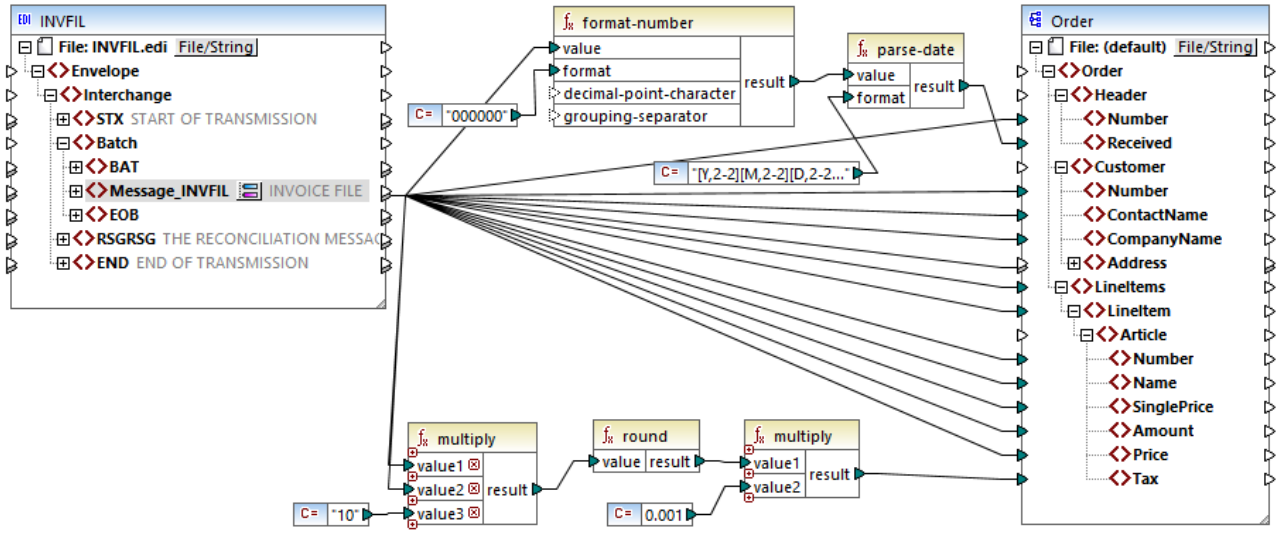
TRADACOMS に適用することのできるMapForce 構成ファイルは、以下のディレクトリに保管されます: **C:\Program Files\Altova\MapForce2021\MapForceEDI\TRADACOMS**。TRADACOMS 構成ファイルに変更を加える場合、TRADACOMS ディレクトリの兄弟ディレクトリにコピーを作成します。例えば、"TRADACOMS1" ディレクトリのパーミッションを "読み取り専用"にせず、カスタムの必要条件に応じてファイルを編集します。すべての構成ファイルは、XML 構文に準じますので、XML エディターを使用してファイルを変更することが奨励されます。

ファイル	目的
EDI.Collection	**EDI.Collection** ファイルは、現在のTRADACOMS 構成ディレクトリ内のすべてのメッセージのリストを含みます（例: **C:\Program Files\Altova\MapForce2021\MapForceEDI\TRADACOMS**）。 コレクションファイルにリストされるメッセージは、マッピングにTRADACOMS コンポーネントを（「入力 | EDI」を使用して）追加すると表示される"EDI コレクションの参照" ダイアログボックス内で選択のために使用できます。ですから、このダイアログボックス内でメッセージの型を抑制するには、コメントアウトを使用するか、EDI.Collection ファイルから対応するXML 要素を削除してください。 例: 以下のリンクのコメントアウトは`LPRHDR` メッセージ型 の使用を不可能にします: `<!--<Message Type="LPRHDR" File="LPRHDR.Config" Description="LOCATION PLANNING REPORT FILE"/>-->`
Envelope.Config	MapForce 内でTRADACOMS コンポーネントの構造を定義します。（すなわち、要素が **Envelope** シーケンスにより囲まれています）。このファイルは、MapForce により内部的に処理されたスキーマに対して検証され、編集されてはなりません。
Tradacoms.Codelist	このファイルはTRADACOMS コードと含まれている可能性のある値を定義します。入力/出力 TRADACOMS ファイルの検証にも使用されます。必要な場合、このファイルにカスタムコードリストを追加することができます。
Tradacoms.Segment	このファイルは、TRADACOMS ファイルのセグメント、複合、および、フィールド名を定義し、TRADACOMS ファイルを解析する際に使用されます。このファイルに加えられる変更は、グローバルにカスタマイズすることができ、全てのセグメントとメッセージに適用できます。
[Message].Config	単一のTRADACOMS メッセージの構造を定義します。このファイルへの変更はローカル（インライン）カスタマイズとして扱われます。

(C) 2015-2021 Altova GmbH

6.5.15.6 例:TRADACOMS 請求書をXML に変換する

このサンプルは、TRADACOMS 請求書ファイルのデータを XML フォーマットに変換するマッピングデザインの作成方法について説明します。このサンプルで作成されたマッピングデザインファイルは、以下のパスで参照することができます: **<マイドキュメント>\Altova\MapForce2021\MapForceExamples\TRADACOMS_Invoice.mfd** 。



TRADACOMS_Invoice.mfd

マッピングを以下を目標とします:

1. ソース TRADACOMS ファイル(**INVFIL.edi**) を既存のスキーマ(**Order.xsd**) に対して有効な XML ファイルに変換します。MapForce Examples フォルダー(<**マイドキュメント**>¥Altova¥MapForce2021¥MapForceExamples¥) 内にソース TRADACOMS ファイルとスキーマファイル(**Order.xsd**) があります。
2. TRADACOMS ファイル内の日付には、`decimal` フォーマットがあり、ターゲット XML ファイル内の `xs:dateTime` フォーマットに変換される必要があります。
3. 注文のオーダー内の各ラインアイテムに対して、税金は、3桁の小数点を持つ小数の値まで計算される必要があります。このサンプルのスコープのために、税金は以下の式に従い計算されると想定します:

   ```
   Tax = round(LEXC * VATP * 10) * 0.001
   ```

   `round` は最も近い整数で値を四捨五入します。

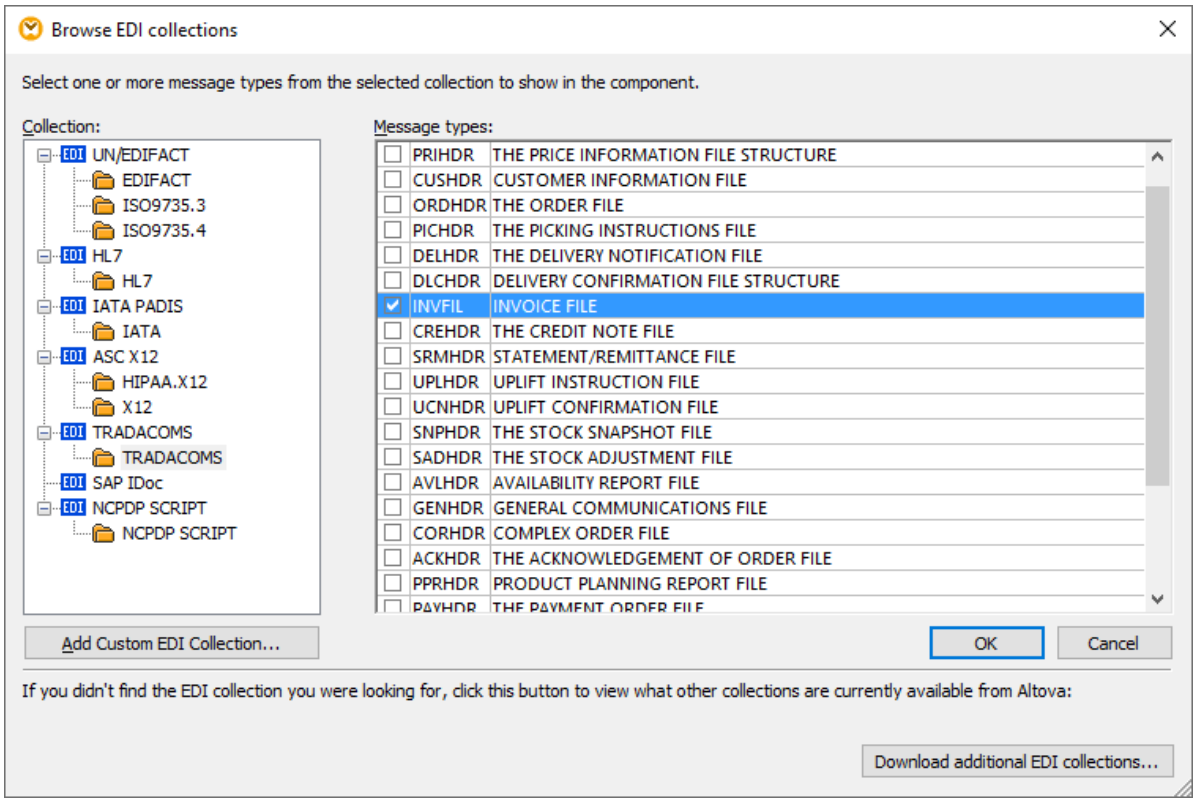
この目的を達成するには、以下の手順を踏みます:

1. ソースとターゲットコンポーネントをマッピングエリアに追加し、その間にマッピング接続を描きます。
2. 日付関数を使用して、日付を必要とされるフォーマットに変換します。
3. 数学関数を使用して、消費税を計算します。

ステップ 1: マッピングエリアにコンポーネントを追加する

ソース TRADACOMS コンポーネントを追加します:

1. 「**挿入**」メニューから「**EDI**」をクリックします。
2. TRADACOMS コレクションフォルダーをクリックして、メッセージの型の下から、「**INVOICE FILE**」を選択します。



3. サンプルファイルを提供するようにプロンプトされると、MapForce Examples フォルダー（<**マイドキュメント>\Altova\MapForce2021\MapForceExamples**）内の**INVFIL.edi** を参照します。
4. コンポーネント設定を入力するようにプロンプトされると、デフォルトの値をそのままにし、「OK」をクリックします。

ターゲット XML コンポーネントを追加する

1. 「挿入」メニューから「XML スキーマ/ファイル」をクリックします。
2. サンプルファイルを提供するようにプロンプトされると、「スキップ」をクリックします。（これはターゲットコンポーネントであるため、ファイルが生成されるため、既存のファイルを参照する必要はありません）。
3. ルート要素を選択するようにプロンプトされると、"Order" を選択します。

このサンプルでは、ソースがソース TRADACOMS ファイルからターゲット XML ファイルに以下の通りマップされます:

ソース（TRADACOMS）	ターゲット（XML）
INVN	/Order/Header/Number
IVDT*	/Order/Header/Received
CIDN-2	/Order/Customer/Number
CADD-1	/Order/Customer/ContactName
CNAM	/Order/Customer/CompanyName
CADD-2	/Order/Customer/Address/Street

(C) 2015-2021 Altova GmbH

ソース（TRADACOMS）	ターゲット（XML）
CADD-3	/Order/Customer/Address/City
CADD-5	/Order/Customer/Address/ZIP
CADD-4	/Order/Customer/Address/State
Group_IRF**	/Order/LineItems
Group_ILD**	/Order/LineItems/LineItem
SPRO-1	/Order/LineItems/LineItem/Article/Number
TDES-1	/Order/LineItems/LineItem/Article/Name
AUCT-1	/Order/LineItems/LineItem/Article/SinglePrice
QTYI-1	/Order/LineItems/LineItem/Article/Amount
LEXC	/Order/LineItems/LineItem/Article/Price
LEXC*, VATP*	/Order/LineItems/LineItem/Article/Tax

* ソースデータは、ターゲットに書き込まれる前に関数により処理されます。
** これらのフィールドにより同じ型のフィールドのグループを介して反復が有効になります。

上のテーブルを参照として使用して、ソースコンポーネント内の上のアイテム全てから、ターゲットコンポーネント内の接続先へ、マッピング接続を描きます。この時点では、計算される値が必要なフィールドを省略します（すなわち、テーブル内で * のマークを持つフィールド）。これらは次のステップで処理されます。MapForce 内で接続を描くための一般的な情報に関しては、接続との作業を参照してください。

ヒント: MapForce コンポーネント内で名前別にフィールドを検索するには、「CTRL + F」を押します。

ステップ 2: 変換 関数を追加する

IVDT アイテムの値を、小数点の値から xs:dateTime 型に変換するには、マッピングは、**core | conversion functions** ライブラリから **format-number** 関数を使用します。成功した例としては、変換中に先頭の0（ゼロ）を保持するを参照してください。

第2番目の関数 **parse-date** の役割は、**format-number** により返された YYMMDD 文字列の値を xs:dateTime 型に変換することです。2 つの入力引数が存在します: (a) フォーマットされる値 (b) フォーマット マスク。フォーマット マスクは、基本的には、MapForce に、年、月、日付は、それぞれ2つの文字から構成されているように処理するように命令します。

上の関数をマッピングに追加するには、関数をライブラリウィンドウからマッピングエリアにドラッグします。関数に引数を提供するには、（メニューコマンド「**挿入 | 定数**」を使用して）定数を入力します。MapForce 内の関数との作業に関しての情報は、関数を参照してください。

ステップ 3: 税率計算関数を追加する

式に従い税率を計算する関数は、**core | math functions** ライブラリ内で使用することのできる **multiply** と **round** です。また、関数は拡張可能です。（すなわち、引数の変数を取ります）。このサンプルでは、 LEXC、VATP、整数 10 を引数として取ります。最初の2 つの値は、ソース TRADACOMS コンポーネント、から接続されており、定数により整数が与えられています。**multiply** 関数の結果は、**round** 関数を使用して最も近い整数に四捨五入されます。最後に、**round** 関数の結果は、2番目の **multiply** 関数を使用して、0.001 により掛けられます。この結果はアイテム XML コンポーネント (/Order/LineItems/LineItem/Article/Tax) の目的のアイテムに、に接続されます。

マッピングデザインの出力をプレビューするには、マッピング ウィンドウの下の部分にある「**出力**」タブ をクリックします。

ステップ 4 (オプション): StyleVision でのスタイルマッピング出力

オプションで、ターゲット XML コンポーネントを StyleVision Power StyleSheet (.sps) ファイルにリンクすることができます。 (StyleVision, http://www.altova.com/stylevision を参照してください)。これにより、HTML、RTF、PDF、または 2007+ ファイルとして、マッピングの出力を生成することができます。に関しての詳しい情報は、(StyleVision がインストールされている場合、各フォーマットの必要よう条件がそろっている場合、 StyleVision を使用してマッピングの出力のスタイルを指定するを参照してください)。任意で、ステップこのステップを行う場合、 XML コンポーネントのヘッダーをダブルクリックし、"StyleVision Power スタイルシートファイル"の横の、MapForce Examples フォルダー内の**Order.sps** ファイルを参照します。マッピングエリアでは、HTML、PDF、RTF、および W ord 2007+ タブをクリックし対応するフォーマットでマッピング出力を確認します。

6.5.16 UN/EDIFACT

UN/EDIFACT (United Nations rules for Electronic Data Interchange for Administration, Commerce and Transport) は電子ドキュメントのインターチェンジの標準です。詳細に関しては、次を参照してください: http://www.unece.org/cefact/edifact/welcome.html。MapForce は、D93.A – D.20A ディレクトリ内に含まれるメッセージへをサポートします。

MapForce 内のデフォルトの UN/EDIFACT バージョンは D.20A です。

EDI マッピングに必要な構成ファイルを含む D93.A からの全てのバージョンのための個別のインストーラーは Altova MapForce ライブラリページ (https://www.altova.com/ja/components_mapforce.html) でご利用いただきます。

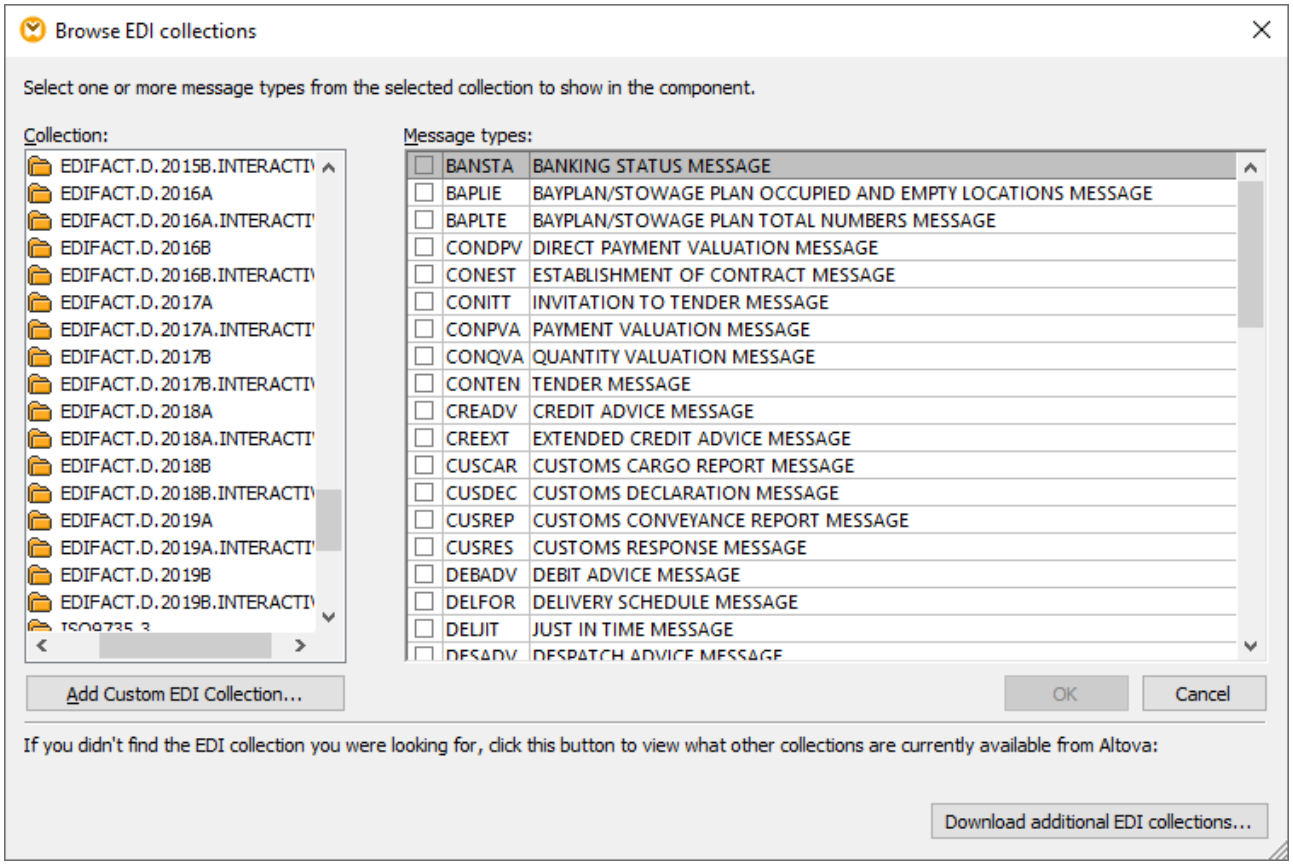
6.5.16.1 バッチとインタラクティブな EDIFACT メッセージ

MapForce は、バッチとインタラクティブな UN/EDIFACT メッセージへのデータのマッピング、からのデータのマッピングをサポートします。すなわち、MapForce 内の UN/EDIFACT コンポーネントは、(バッチ、または インタラクティブ インターチェンジのための) 文法的に有効な UN/EDIFACT フォーマットファイルを読み取る、または書き込むために必要な構造を提供します。基本の検証と自動補完ルールは UN/EDIFACT フォーマットファイルを読み取る、または書き込む際に強制されます。この点については後に詳細に説明されています。MapForce は実際に UN/EDIFACT インターチェンジプロセスに参加するパーティ間のファイルの転送は行わないことにご注意してください。MapForce の役割はこのようなファイルの生成、またはファイルを他のフォーマットに変換することです。

MapForce がデータをマップし、バッチ EDIFACT メッセージからデータをマップするために特別な条件は存在しませんが、インタラクティブ EDIFACT メッセージへデータをマップ、またはインタラクティブ EDIFACT メッセージからデータをマップする場合、次の必要条件が適用されます:

1. MapForce 2018 または以降が必要とされます。
2. インタラクティブ EDI メッセージをサポートするために必要とされる MapForce 構成ファイルが個別にインストールされている必要があります。インストールファイルは Altova MapForce ライブラリページ (https://www.altova.com/ja/mapforce/download/libraries) から無料でダウンロードしインストールすることができます。これをおこなうと、必要とされるコレクションとメッセージ方が、下で示される「EDI コレクションの参照」ダイアログボックスで使用できるようになります。

バッチ、またはインタラクティブ UN/EDIFACT コンポーネントにマッピングを追加するには、「EDI コレクションの参照」ダイアログボックスから対応するコレクションを選択してください。このダイアログボックスを開くには、「**挿入 | EDI**」メニューコマンドを実行してください (、または**EDI ファイルの挿入** ツールバーボタンをクリックしてください)。次も参照してください: EDI コンポーネントをマッピングに追加する。



EDI コレクションダイアログボックスの参照

メモ 上のダイアログボックスでは、サフィックス "INTERACTIVE" はインタラクティブ EDIFACT 構造に定期用することのできるコレクションを示しています。例えば "EDIFACT.D.2016A.INTERACTIVE"。バッチ構造に適用することのできるコレクションは、タイトル内にUN/EDIFACT ディレクトリバージョンのみを含んでいます。例えば "EDIFACT.D.2016A"。同じ変換を MapForce のEDI 構成フォルダー内で使用することのできる構成ディレクトリ(コレクション) に適用することができます(**C:\Program Files\Altova\MapForce2021\MapForceEDI**)。

UN/EDIFACT バッチファイルの構造は、インタラクティブなインターチェンジを想定したファイルの構造とは異なります。このため、MapForce 内の EDIFACT コンポーネントは、バッチ、またはインタラクティブであることができ(このためバッチ、またはインタラクティブなメッセージを含み)ますが、同時に両方を含むことはできません。同様に検証と自動補完のルールも両方のシナリオに対しては異なります。詳細は次のトピックを参照してください:

- UN/EDIFACT 検証ルール(バッチ)
- UN/EDIFACT 自動補完ルール(バッチ)
- UN/EDIFACT 検証ルール(インタラクティブ)
- UN/EDIFACT 自動補完ルール(インタラクティブ)

6.5.16.2 UN/EDIFACT 検証ルール (バッチ)

UN/EDIFACT ドキュメントの検証時には以下の項目がチェックされます:

- UNB ならびにUNZ セグメントが存在するか。

- UNB/S004 内に、妥当な日付/時刻が指定されているか。
- UNB/0020 ならびにUNZ/0020 に同一の値が含まれるか。
- UNZ/0036 内に、(それが存在する場合) 関数グループの数、またはメッセージ正しい数が含まれているか。関数グループが存在する場合その関数グループの数、またはインターチェンジ内に含まれているメッセージの数が記載されることになります。

各関数グループに対して、以下の項目がチェックされます:

- マッチするUNG とUNE のペアが含まれるか。
- UNG/S004 内に妥当な日付/時刻の指定が含まれるか。
- UNE/0060 内に、関数グループ内に含まれているメッセージの正しい番号が含まれているか。

各メッセージに対して、以下の項目がチェックされます:

- マッチするUNH とUNT のペアが含まれるか。
- UNH/S009/0052 内に、含まれている関数グループのUNG/S008/0052 と同じ値が含まれているか。
- UNH/0062 ならびにUNT/0062 に同一の値が含まれているか。
- UNH/S009/0065 内に、正しいメッセージ型指定子が含まれているか。
- UNT/0074 内に、メッセージの中にあるセグメントの正しい数が含まれているか。

6.5.16.3 UN/EDIFACT 自動データ補完ルール(バッチ)

UN/EDIFACT ファイルを生成する際、MapForce は、("自動データ保管" または "自動補完" ルールとして説明されている) ルールに従い、自動的に特定のフィールドまたは値を生成する可能性があります。この振る舞いを無効化するには、 EDI コンポーネント設定内のチェックボックスの **不足しているフィールドの自動補完** のチェックを解除してください。

UN/EDIFACT の自動データ補完により、以下の項目が保証されます:

- UNB ならびにUNZ セグメントが存在する。
- UNG またはUNE の一報が存在する場合、もう一方のID も存在する。
- UNH ならびにUNT セグメントが存在する。
- UNB/S001 が存在する。その中にデータが含まれていない場合、ユーザーにより定義された設定から構文レベルと構文バージョンが使用されます。EDI コンポーネント設定の「拡張 | 構文バージョン番号」 を参照ください。
- UNB/S002 ならびにUNB/S003 が存在する。
- UNB/S004 が存在する。データが含まれていない場合、現在使用されているEDI フォーマットの日付/時刻が挿入されます。
- UNZ/0036 が存在する。データが含まれていない場合、関数グループまたはメッセージの数が計算され、挿入されます。
- UNZ/0020 の存在。データが含まれない場合、 UNB/0020 からの値がコピーされます。

メモ このセクションにて言及されなかったフィールドが (自動データ補完により) 挿入または作成されることはありません。正しい値を自動的に確定することはできません。例えば、(EDI コンポーネント内に) ターゲット親要素のA と、それらの子アイテムx、y、z があり、そのうちy が必須アイテムとなっている場合、子要素の "y" がターゲットコンポーネントへマップされた時だけ、親要素のA が作成されます

関数グループのチェックにより、以下の項目が保証されます:

- UNG/0038 の存在。データが含まれない場合、メッセージ名が挿入されます。
- UNG/S006 ならびにUNG/S007 が存在する。
- UNG/S004 が存在する。データが含まれていない場合、現在使用されているEDI フォーマットの日付/時刻が挿入されます。
- UNG/0051 が存在する。データが含まれていない場合、ユーザー定義設定から得られたコントロール代理 (制御機関) にある最初の2 文字が挿入されます。EDI コンポーネント設定 の「拡張 | コントロール代理」を参照ください。
- UNE/0060 が存在する。データが含まれていない場合、メッセージの数が計算され、挿入されます。
- UNE/0048 が存在する。値が含まれていない場合、UNG/0048 から得られた値がコピーされます。

(C) 2015-2021 Altova GmbH

メッセージのチェックにより、以下の項目が保証されます:

- UNH/S009/0065 の存在。データが含まれない場合、メッセージ名が挿入されます。
- UNH/S009/0052 とUNH/S009/0054 が存在する。
- UNH/S009/0051 の存在。が存在する。データが含まれていない場合、ユーザー定義設定から得られたコントロール代理(制御機関)にある最初の2 文字が挿入されます。EDI コンポーネント設定 の「拡張 | コントロール代理」を参照ください。
- UNT/0074 の存在。データが含まれない場合、メッセージ内のにあるセグメントの数が計算され、挿入されます
- UNT/0062 の存在。データが含まれない場合、 UNH/0062 からの値がコピーされます。
- UNH/0062 が存在する。データが含まれていない場合、UNT/0062 から得られた値がコピーされます。(trailer セグメント番号だけがマッピングされている場合、ヘッダーセグメントの対応するフィールドが同じ値とともに与えられます)

6.5.16.4 UN/EDIFACT 検証ルール (インタラクティブ)

次の検証ルールは、インタラクティブな EDIFACT メッセージを含むMapForce コンポーネントに適用することができます:

- UIB が存在する場合、次のUIH/S302 はUIB/S302 に一致します。
- UIH/S306/F0065 は、(パーサーにより検証される)メッセージ型 を含みます。
- UIH/S306/F0052 は、 選択された構成ファイル内で定義されているメッセージ バージョン番号を含みます。
- UIH/S306/F0054 は、選択された構成ファイル内で定義されているメッセージ リリース番号を含みます。
- UIT/F0340 は対応するUIH/F0340 (メッセージ ヘッダー/トレーラー) に一致する、または存在しません(任意のフィールド)。
- UIT/F0074 は、セグメントの数を含む、または存在しません(任意のフィールド)。
- UIZ/S302 は、UIB/S302 に一致する、または存在しません(任意のコンポジット)。
- UIZ/F0036 は、メッセージの数を含む、または存在しません(任意のフィールド)。

6.5.16.5 UN/EDIFACT 自動補完ルール (インタラクティブ)

次の自動補完ルールは、インタラクティブな EDIFACT メッセージを含むMapForce コンポーネントに適用することができます:

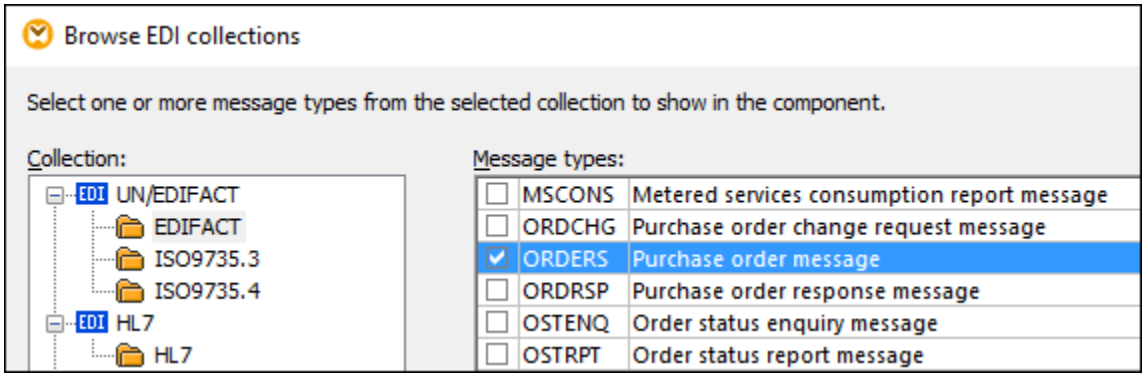
- UIB/S001/F0001 は制御機関 + 構文バージョン番号を使用してフィルされます。
- UIB/S001/F0002 は構文バージョン番号を使用してフィルされます。
- UIB/S002/F0004 は"Sender" を使用してフィルされます。
- UIB/S003/F0010 は"Recipient" を使用してフィルされます。
- UIZ/S302/* はUIB/S302/* を使用してフィルされます。
- UIZ/F0036 はメッセージ数 を使用してフィルされます。
- UIH/S302/* は(UIB とUIB/S302 が存在する場合) UIB/S302/* を使用してフィルされます。
- UIH/S306/F0065 はをメッセージ型使用してフィルされます。
- UIH/S306/F0052 は構成ファイルからのメッセージ バージョン番号を使用してフィルされます。
- UIH/S306/F0054 は構成ファイルからのメッセージ リリース番号を使用してフィルされます。
- UIT/F0340 はをUIH/F0340 使用してフィルされます(これは任意のフィールドですが、補完されます)。
- UIT/F0074 はメッセージ内のセグメントの数を使用してフィルされます(これは任意のフィールドですが、補完されます)。

6.5.16.6 例: UN/EDIFACT を XML にマッピングする

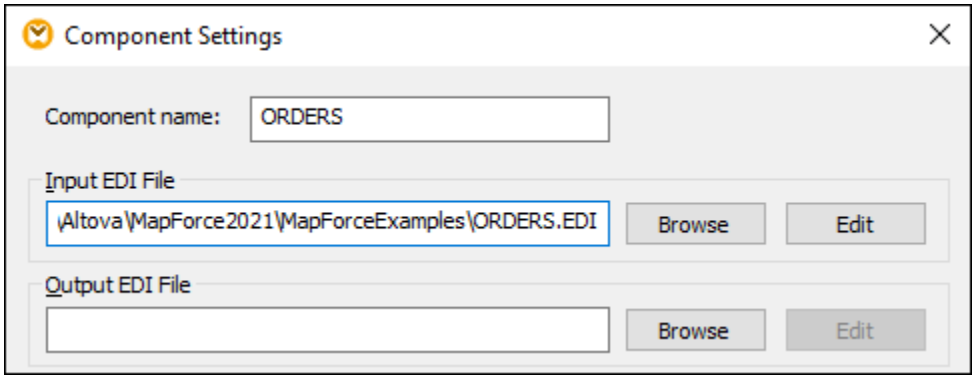
この演習では、UN/EDIFACT メッセージからのデータを XML スキーマ/ドキュメントへマッピングして、更なる処理を行うための XML ファイルを作成する方法について記述します。このサンプルで紹介されているマッピングは以下のパスで確認することができます: **<マイ ドキュメント>\Altova\MapForce2021\MapForceExamples\EDI_Order.mfd**.

ステップ 1: UN/EDIFACT コンポーネントをマッピングに追加する

1. 新規マッピングを作成し、次の変換言語から1つ選択します: Java、C#、C++、またはBUILT-IN。このサンプルでは、が変換言語として使用されています。
2. 「挿入」メニューから、「EDI」をクリックします。
3. EDI のコレクションダイアログボックスから、EDIFACT を選択し、**ORDERS** を選択し、「OK」 をクリックします。

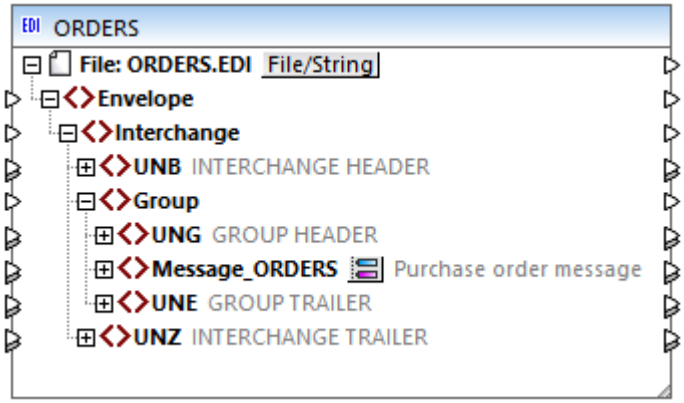


4. サンプルEDI ファイルを提供するようにプロンプトされると、参照 をクリックして、<**マイドキュメント >\Altova\MapForce2021\MapForceExamples** ディレクトリから **ORDERS.EDI** を選択し、開くをクリックします。コンポーネント設定ダイアログボックスが開かれ、入力ならびに出力 EDI インスタンスを選択することができます。これによりマッピングに追加する前に、EDI コンポーネントの設定を確認することができます。必要な場合これらの設定をいつでも変更することができます(次を参照: EDI コンポーネント設定)。ORDERS.EDI ファイルが入力 EDI ファイルとして表示されることを確認してください。



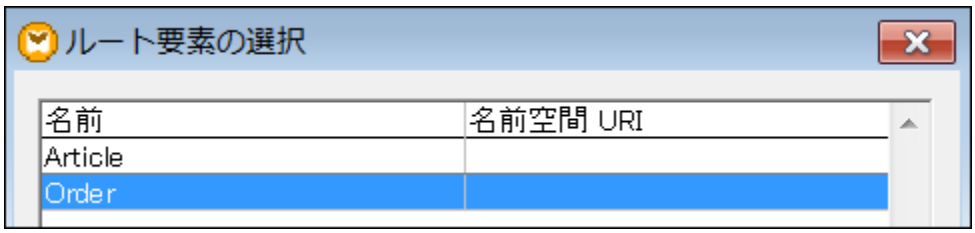
5. 「OK」をクリックします。EDI コンポーネントがマッピング領域に表示されます。Message_ ORDERS ノードをダブルクリックして、子アイテムを確認します。コンポーネントウィンドウの右下をドラッグすることで、コンポーネントウィンドウの大きさを変更することができます。

(C) 2015-2021 Altova GmbH

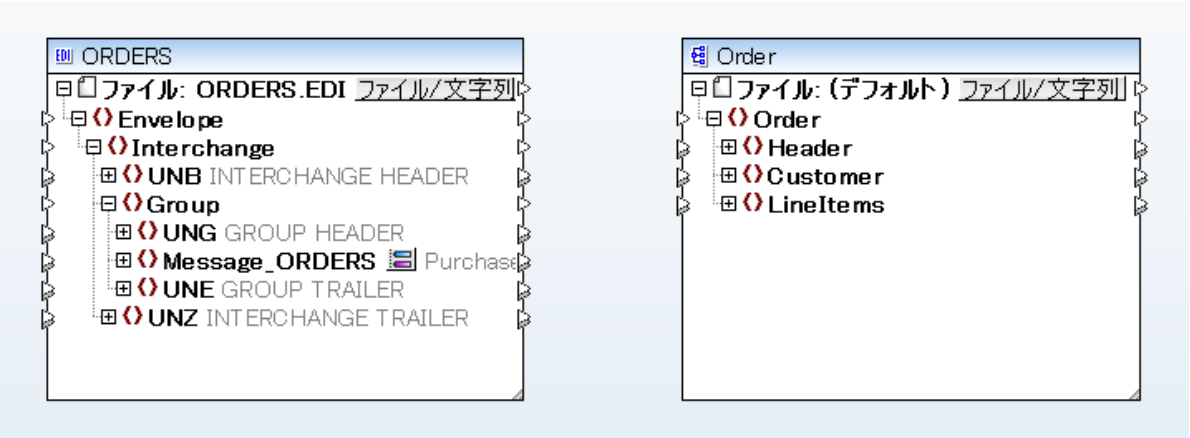


ステップ 2: ターゲットスキーマコンポーネントをマッピングに追加する

1. 「**挿入**」メニューから、XML スキーマ/ファイルをクリックして、 <**マイドキュメント>\Altova\MapForce2021\MapForceExamples** ディレクトリから **Order.xsd** を選択します。
2. サンプルXML ファイルを指定するよう促されるので、「**スキップ**」をクリックして、**Order** をターゲットドキュメントのルートとして選択します。



この時点で、ソースEDI コンポーネントとターゲット XML スキーマはマッピングエリアに表示されており、マッピング接続を開始することができます。



ステップ 3: EDI アイテムをマップする

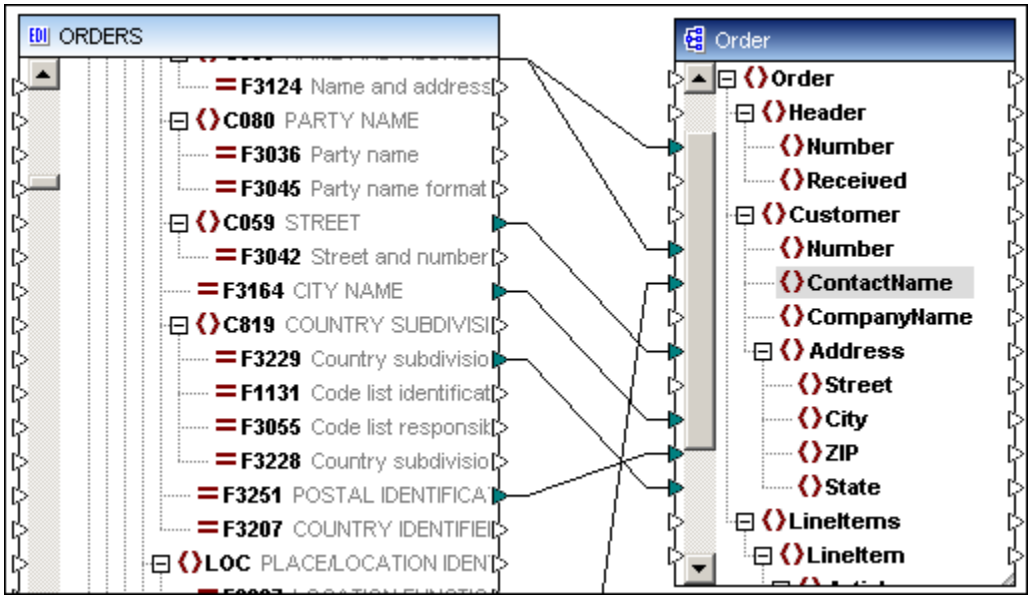
EDI コンポーネントには、選択したコレクション（ORDERS）から得られたメッセージの構造が表示されます。通常、全てのノードにデータが含まれることはなく、EDI を伴うプロジェクトを扱う際には、使用する EDI ドキュメントに十分精通している必要があります。

MapForce では、ソースアイテムとターゲットアイテム間に接続を描くことでマッピングを行うことができます。手順を追っての MapForce 内でのマッピング接続に関しては、接続と作業するを参照してください。

このサンプルでは、（ **Group/Message_ORDERS** ノードから始まる）次のノードが最初にマップ（接続）されます:

ソース	ターゲット
BGM/C106/F1004	Order/Header/Number
SG2/NAD/C082/F3039	Order/Customer/Number
SG2/NAD/C080/F3036	Order/Customer/CompanyName
SG2/NAD/C059/F3042	Order/Customer/Address/Street
SG2/NAD/F3164	Order/Customer/Address/City
SG2/NAD/C819/F3229	Order/Customer/Address/State
SG2/NAD/F3251	Order/Customer/Address/ZIP
SG2/SG5/CTA/C056/F3412	Order/Customer/ContactName

この時点では、マッピングは以下にあるようなものになるはずです:



マッピングプロセスを継続します:

(C) 2015-2021 Altova GmbH

ソース	ターゲット
SG29	Order/LineItems
SG29/LIN/C212/F7140	Order/LineItems/LineItem/Article/Number
SG29/IMD/C273/F7008	Order/LineItems/LineItem/Article/Name
SG29/QTY/C186/F6060	Order/LineItems/LineItem/Article/Amount
SG29/SG33/PRI/C509/F5118	Order/LineItems/LineItem/Article/SinglePrice

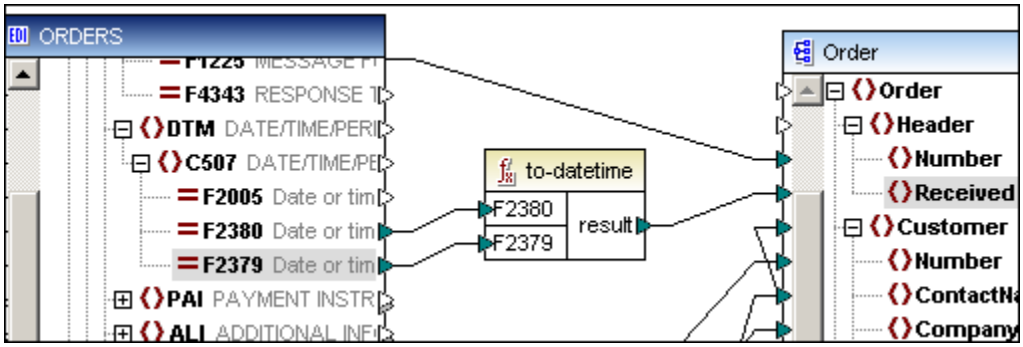
ステップ 4: 日付のフォーマット

edifact ライブラリから **to-datetime** 関数をマッピング領域にドラッグします。関数をマッピングに追加するを参照してください

DTM/C507 要素の**F2380** と**F2379** コンポーネントを適用することで、適正な方法でフォーマットされた受信側のdatetime を作成することができます。

ですから以下のフィールドをマップします:

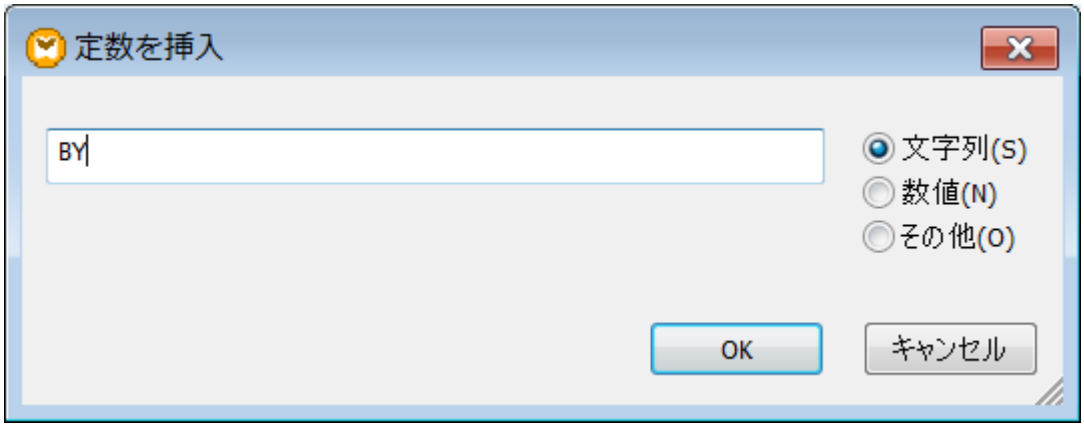
ソース	ターゲット
DTM/C507/F2380	to-datetime 関数の **F2380** 入力
DTM/C507/F2379	to-datetime 関数の **F2379** 入力
to-datetime 関数の**result** 出力	Order/Header/Received



ステップ 5: からの注文をフィルターする

ここで、"Buyer" からの購入注文をフィルタリングすることにします。フィルタリングはNAD (Name とAddress) セグメントのパーティーを識別することで行われます。この場合、'BY' という値により "Buyer" (商品やサービスが販売される相手) が示されます。

1. **core** ライブラリからマッピング領域へ、`equal` 関数をドラッグします。
2. マッピングにフィルターを追加します（「**挿入**」メニューから、「**フィルター: ノード/行**」をクリックします）。
3. マッピングに定数を追加します（「**挿入**」メニューから、「**定数**」をクリックします）。"BY" という値にテキストフィールドに "BY" を入力して定数を割り当てます:

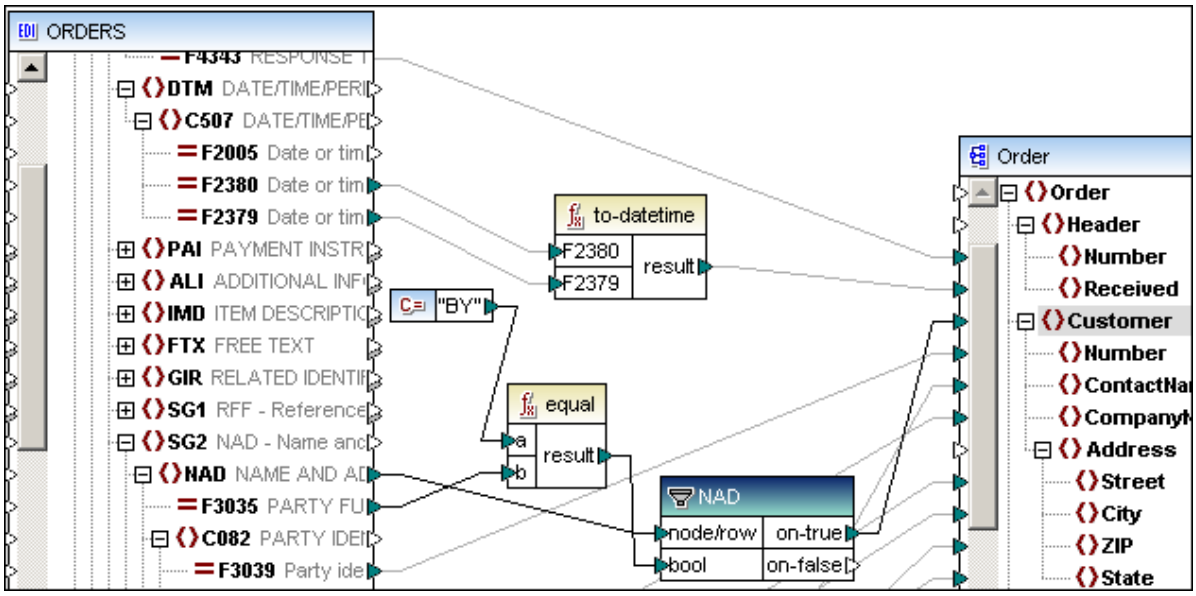


以下のアイテムをマップします:

ソース	ターゲット
SG2/NAD/F3035	`equal` 関数の**b** 入力
"BY" 定数	`equal` 関数の**a** 入力
`equal` 関数の結果	**filter** コンポーネントの**bool** 入力
SG2/NAD	フィルターコンポーネントの **node/row**
on-true の場合の**filter** コンポーネントの結果	スキーマの Order/Customer

この操作により、NAD ノードが、パーティーコードの "BY" により識別される "Buyer" を示している時だけ、データのマッピングが行われるようになります。

(C) 2015-2021 Altova GmbH



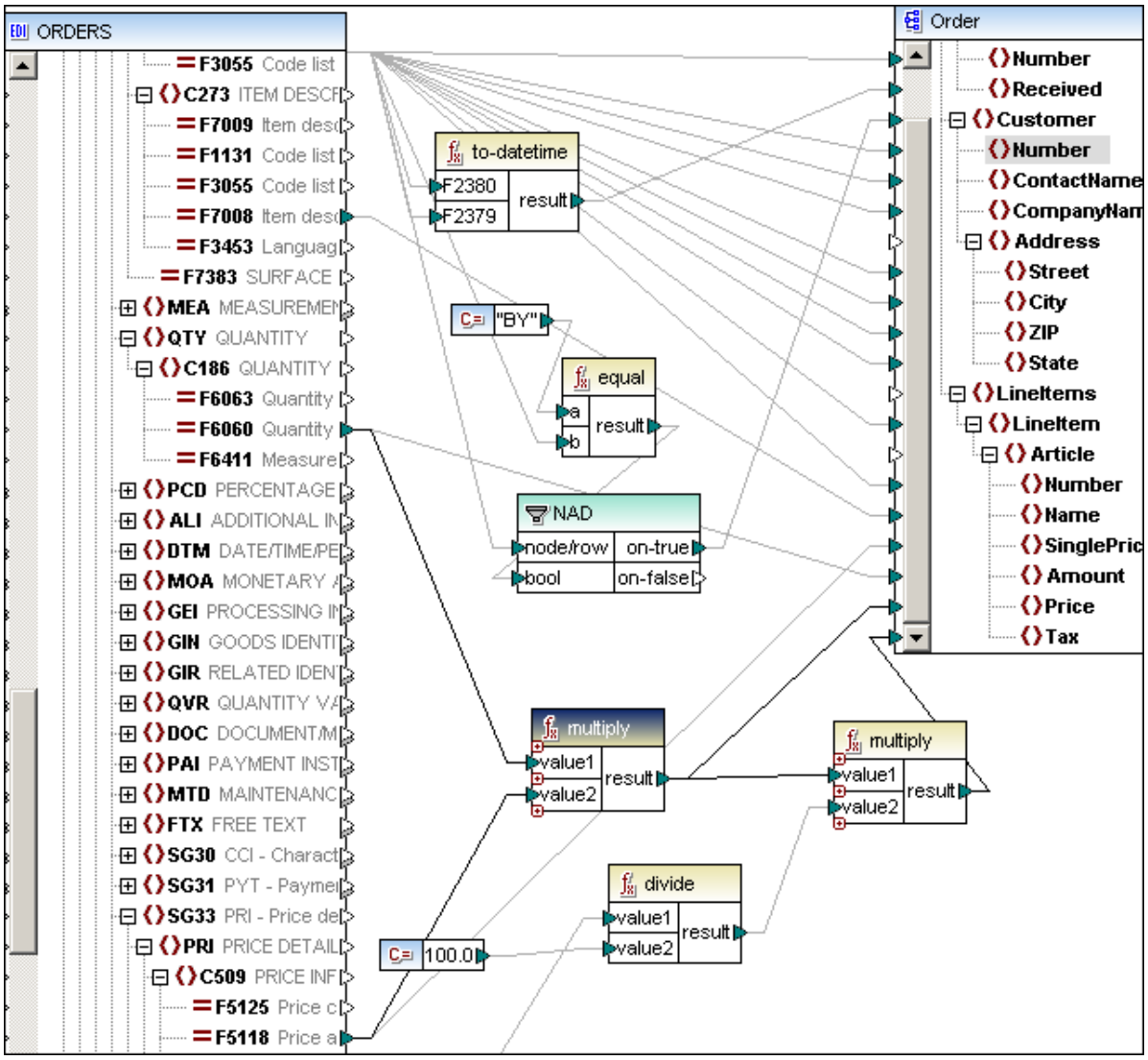
ステップ 6: 価格と税金の計算

このタスクの最終的な仕上げとして、価格と税金の計算を行います。

1. ライブラリウィンドウから (core ライブラリの) **multiply** 関数を 2 つ、そして **divide** 関数を 1 つ、マッピング領域へドラッグします。
2. 定数コンポーネントを挿入します (「**挿入 | 定数**」)。数値が選択された状態で、"100.0" という値をテキストフィールドへ入力します。
3. 以下のアイテムをマップします:

ソース	ターゲット
SG29/QTY/C186/F6060	最初の**multiply** 関数の**value1**
SG29/SG33/PRI/C509/F5118	最初の**multiply** 関数の**value2**
最初の**multiply** 関数の結果	Order/LineItems/LineItem/Article/Price
SG29/SG40/TAX/C243/F5278	**divide** 関数の**value1**
定数 "100.0"	**divide** 関数の**value2**
最初の**multiply** 関数の結果	2番目の**multiply** 関数の**value1**
divide 関数の結果	2番目の**multiply** 関数の**value2**
2番目の**multiply** 関数の結果	Order/LineItems/LineItem/Article/Tax

マッピングは以下に示されているようなものになります:



出力タブをクリックすることで、変換が実行され、XML ドキュメントの変換結果が表示されます:

(C) 2015-2021 Altova GmbH

```
<?xml version="1.0" encoding="UTF-8"?>
<Order xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  <Header>
    <Number>ABC123456XYZ</Number>
    <Received>2004-04-30T17:42:00-09:00</Received>
  </Header>
  <Customer>
    <Number>123</Number>
    <ContactName>Michelle Butler</ContactName>
    <CompanyName>Nanonull, Inc.</CompanyName>
    <Address>
      <Street>119 Oakstreet Suite 4876</Street>
      <City>Vereno</City>
      <ZIP>29213</ZIP>
      <State>CA</State>
    </Address>
  </Customer>
  <LineItems>
    <LineItem>
      <Article>
        <Number>42</Number>
        <Name>Pizza Pepperoni</Name>
        <SinglePrice>7.2</SinglePrice>
        <Amount>1</Amount>
        <Price>7.2</Price>
        <Tax>0.648</Tax>
      </Article>
    </LineItem>
```

6.5.17 EDI 構造のカスタム化

デフォルトでは、公式の仕様に従い、MapForce は、マッピング上のEDI ファイルの構造を示します。多くの場合、デフォルトのEDI 構成を手動でカスタム化する必要はありません。しかしながら、多種の非標準 EDI フォーマットとバリエーションを処理する場合、カスタム化を必要とします。例えば、特定のデータ要素の構造または検証ルールを若干適用する場合があります。これは、カスタムEDI コレクションをビジネスのニーズに合わせ、MapForce に追加することにより達成することができる場合があります。

EDI バリエーション（カスタムEDI コレクション）をMapForce により既にサポートされる次の標準のために作成することができます: EDIFACT（ISO 9735）、ASC X12、HIPAA X12、HL7、NCPDP SCRIPT、SWIFT MT、またはTRADACOMS. 新規のEDI 標準の始めからの作成はサポートされていません。

6.5.17.1 カスタム EDI コレクションの作成

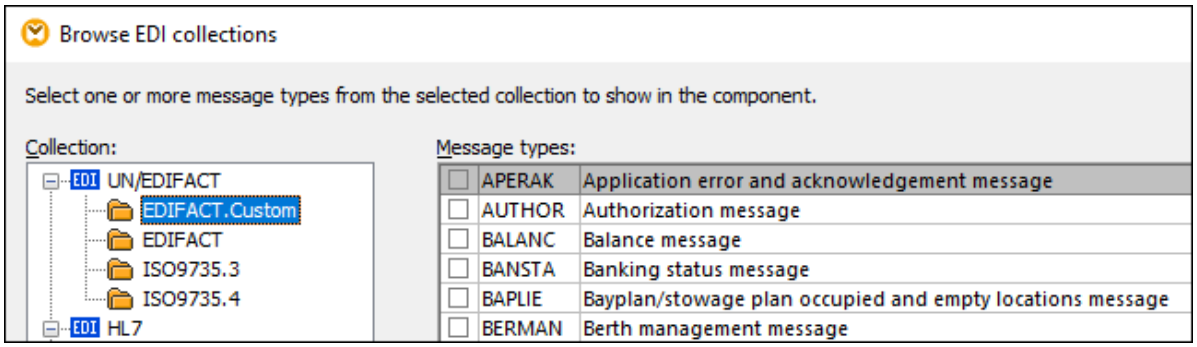
「EDI コレクション」は、特定のEDI 標準に適用することのできるすべての構成ファイルを含むディレクトリです。各 EDI コレクションは [EDI.Collection] という名前のファイルと直接的または間接的に参照される多種の他のファイルを含んでいます。パス上の最後のディレクトリはEDI の名前でEDI コンポーネントをマッピングに追加するとMapForce の視覚的なユーザーインターフェイスに表示されます（例えば、UN/EDIFACT、HL7、ASC X12、などが表示されます）。

MapForce のインストール後、複数のビルトイン EDI コレクションが以下のパスで見つけることができます: **C:\Program Files\Altova\MapForce2021\MapForceEDI**。32-ビット MapForce を 64-ビット オペレーティングシステム上で作動している場合、ディレクトリパスを **C:\Program Files (x86)\Altova\MapForce2021\MapForceEDI** に調整してください。

デフォルトで、上のディレクトリにはビルトイン EDI コレクションのみが含まれています。各 EDI コレクションのすべてのサポートされるバージョンを Altova ダウンロードセンターからダウンロードすることができます(https://www.altova.com/ja/mapforce/download)。

MapForce 内でカスタム EDI コレクションを3つの方法で作成することができます:

1. カスタム EDI コレクションを以下のディレクトリに作成します: **C:\users\<name>\Documents\Altova\MapForce2021\MapForceEDI**. 例えば **C:\Program Files\Altova\MapForce2021\MapForceEDI\EDIFACT** を **C:\users\<name>\Documents\Altova\MapForce2021\MapForceEDI\EDIFACT.Custom** にコピーすると、EDI コンポーネントをマッピングに追加すると [EDIFACT.Custom] コレクションが EDI コレクションのリストに表示されます。



通常 EDI コレクションはビルトイン EDI コレクションと区別できるようになっています。この目的のため、ビルトインコレクションと同じ名前を与えてはなりません。例えば、**C:\Program Files\Altova\MapForce2021\MapForceEDI\EDIFACT** からのビルトインコレクションにより名前が既に使用されているためカスタム EDI コレクションのディレクトリに [EDIFACT] という名前を与えてはなりません。

カスタム EDI コレクションが **C:\Program Files\Altova\MapForce2021\MapForceEDI** からのデフォルトの EDI コレクションよりも高い優先順序が必要な場合、カスタム EDI コレクションにビルトイン EDI コレクションと同じ名前を与えます。

2. EDI コレクションが保管されるカスタムディレクトリをポイントします。例えば **C:\Program Files\Altova\MapForce2021\MapForceEDI\EDIFACT** を **C:\EDIFACT.Custom** にコピーすることができます。上のダイアログボックスの上の **カスタム EDI コレクションの追加** ボタンをクリックし **C:\EDIFACT.Custom** を参照します。EDI コレクションのパスがマッピング (.mfd) ファイルに対して相対的であることを確認し、 **MFD ファイルに対して全てのファイルパスを保存する** チェックボックスをクリックします。

このように追加されたカスタム EDI コレクションはマッピング上の現在の EDI コンポーネントに影響を与えます。

3. (奨励されません) **C:\Program Files\Altova\MapForce2021\MapForceEDI** ディレクトリ内にカスタム EDI コレクションを作成します。このディレクトリ内に変更を加えるには管理者の権利が必要になります。このアプローチは下位互換性のためにのみサポートされ、奨励されません。

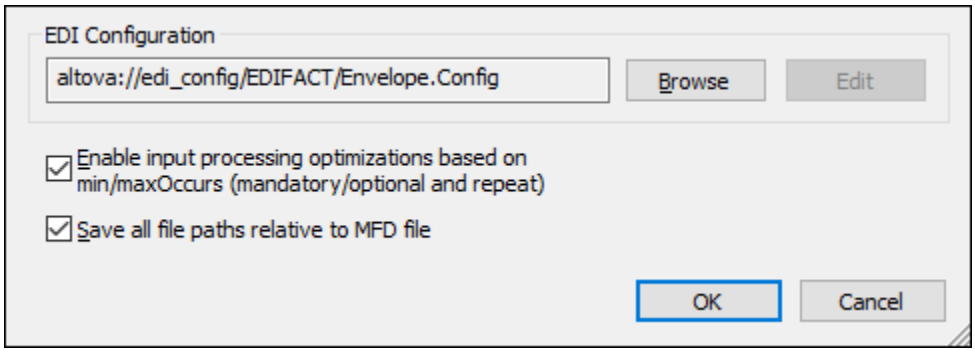
C:\Program Files\Altova\MapForce2021\MapForceEDI ディレクトリ内のオリジナルのファイルとフィルダーを変更することは奨励されず自身が責任をもって MapForce 内のデフォルトの EDI 検証と処理をオーバーライドを実行してください。代わりに、上記の最初または2番目のアプローチを使用してください。これにより、カスタム構成を新しいバージョンの MapForce に移行することができます。

(C) 2015-2021 Altova GmbH

MapForce がEDI 構成を解決する方法

EDI コンポーネントにより使用される構成（EDI コレクション）をビューする

- マッピング上のコンポーネントのタイトルバーをダブルクリックします。（またはコンポーネントタイトルバーを右クリックして、コンテキストメニューからを [プロパティ] 選択します。EDI コンポーネント設定 ダイアログボックスが表示されます。



EDI 構成 プロパティはコンポーネントにより使用される構成（EDI コレクション）へのパスを示します。`altova://edi_config/`, からパスが開始される場合 MapForce は対応する構成を以下のディレクトリ内で検索します（順序は重要です）:

1. Windows 上のユーザーのプロファイルディレクトリ内の（**C:\users\<name>\Documents\Altova\MapForce2021\MapForceEDI**）。このパスで構成が見つかると **C:\Program Files\...** からのパスより高い優先順位が与えられます。（以下の点を参照してください）。
2. **C:\Program Files\Altova\MapForce2021\MapForceEDI** ディレクトリ内。

EDI 構成 プロパティが標準のWindows パスを指す場合、対応するEDI コレクションがコンポーネントにより使用されます。

例

EDI カスタム化のステップバイステップのサンプルに関しては以下を参照してください:

- 例: TA1 セグメントの追加
- 例: EDIFACT メッセージのカスタム化
- 例: ANSI X12 トランザクションのカスタム化

6.5.17.2 EDI 構成ファイル

EDI コレクション（ディレクトリ）には次の構成ファイルが含まれます:

ファイル	目的
EDI.Collection	**EDI.Collection** ファイルは実装中のカスタムEDI コレクション内のすべてのメッセージ型を定義します。MapForce 内の選択のために使用することのできるメッセージ型を追加、または削除するために、このファイルを編集することができます（「EDI コレクションの参照」ダイアログボックス）。次を参照してください: メッセージ型の追加または削除.
Envelope.Config	**Envelope.config** ファイルは、MapForce 内のEDI マッピングコンポーネントの構造を定義します。（すなわち、**Envelope** シーケンスに囲まれた要素）。通常、このファイルを手動で編集してはなりません。

ファイル	目的
<Message>.Config	EDI コレクション ディレクトリは、1つまたは複数の**<Message>.Config** ファイルを必要とします。このようなファイルは、EDI メッセージ型、およびに対応し、メッセージ内で使用されるグループとセグメント定義が含まれます。**<Message>.Config** ファイルは、**EDI.Collection** ファイルから参照されます。 特定のメッセージのフィールドに適応することのできる構造または検証ルールを変更する場合は、<Message>.Config ファイルを変更してください。このファイルに加えられる変更は、ローカルのカスタマイズと考えられます。(すなわち、これらは、特定のメッセージに適用されます)。例に関しては、次を参照してください EDIFACT メッセージのカスタム化 と ANSI X12 トランザクションのカスタム化.
<Filename>.Segment	このファイルは、EDI ファイルのデータ要素、セグメント、複合、およびサブ複合を定義し、EDI ファイルを解析する際に使用します。 すべてのメッセージ内のフィールド (データ要素) に適用することのできる構造または検証ルールを変更する場合、このファイルに加えられる変更はグローバルにカスタム化されます。
<Filename>.Codelist	このファイルは、MapForce 内の入力と出力 EDI ファイルを検証するために、MapForce により使用される EDI コード値を定義します。通常、ここで定義されているコード値は、それぞれの EDI 標準による定義に対応します。このルールの例外は、標準が外部コード値に頼り、可能な値の値を完全に強制できない場合です (次も参照: EDI コンポーネント検証)。 デフォルトの検証ルールを変更する場合、.codelist ファイルを変更することができます。(例えば、検証を緩和、または EDI コードリスト内で定義されていない新しいコードを追加する場合)。詳細に関しては、次を参照してください: フィールドの検証の弛緩 と 外部ソースからコードの値を適用する。
ParserErrors.Config	このファイルは、X12 と HIPAA X12 標準に対してのみ使用することができます。MapForce 内の EDI コンポーネント上の "ParserErrors_Message" と "ParserErrors_Group" アイテムの構造を定義します。これらのアイテムは、ファイルの処理中の解析エラーの情報、および X12 997 (Functional Acknowledgement) と X12 999 (Implementation Acknowledgement) メッセージを作成するために使用することのできる情報を含んでいます。

ファイルのすべては、XML ファイルで、XMLSpy などの XML エディターで編集することができます。**EDI.Collection** ファイルの例外を除いて、一部の XML スキーマが通常参照されます (次を参照: EDI 構成スキーマ。) ファイルには、定義された明確なスキーマがありませんが、MapForce により内部で検証されます。

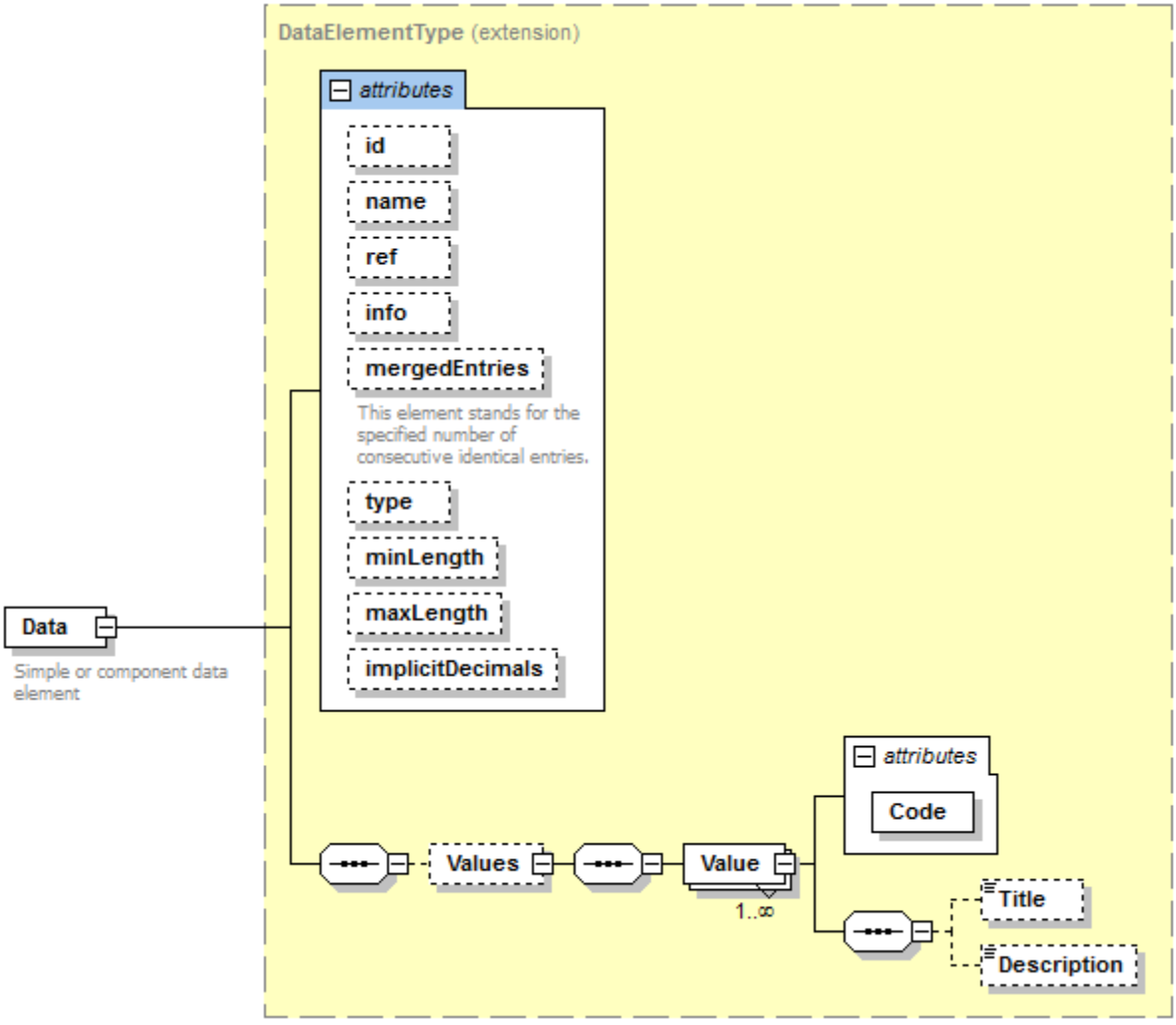
6.5.17.3 EDI 構成スキーマ

構成ファイルに変更する場合、使用することのできる XML スキーマに対して検証することは重要です。ディレクトリ **C:\Program Files\Altova\MapForce2021\MapForceEDI** で使用することのできるスキーマを確認することができます、以下のリストはリストの一部です:

ファイル	目的
EDIConfig.xsd (バージョン "3")	構成スキーマは、MapForce 2011 リリース 2 から有効です。バージョン 3 前のスキーマを使用する構成ファイルが存在する場合、このスキーマ (または、以降) を使用するためにアップグレードされる必要があります。詳細に関しては、次を参照してください: 古い構成ファイルのアップグレード.

ファイル	目的
EDIConfig4.xsd	構成スキーマは、MapForce 2011、リリース3 から有効です。これには、X12 またはHL7 フォーマットにデータをマップするために必要な可能性がある、カスタムの再帰的な階層構造を定義することのできる機能を含む、元の**EDIConfig.xsd** スキーマに加えられた複数の拡張が含まれています。
EDIConfig5.xsd	構成スキーマは、MapForce 2017 から有効です。不完全な検証ルールを定義する機能が搭載されています。このスキーマ内では、文字レベルで高度なフィールドの検証ルールを定義することができます（次を参照: 文字レベルの検証の構成）。
EDIConfig6.xsd	この構成スキーマは、MapForce 2018 から有効です。 NCPDP SCRIPT フォーマットからの、または NCPDP SCRIPT フォーマットへのデータのマップを搭載しています。NCPDP SCRIPT を参照してください。
EDIConfig7.xsd	この構成スキーマはMapForce 2021 から使用可能です。SWIFT MT へ、または、からデータをマップする機能が搭載されています。

EDI 構成スキーマは、構成ファイル内に存在するすべての要素の構造に関する情報を提供します。例えば、 下に表示される **EDIConfig4.xsd** スキーマ内に表示されるはEDI データ要素構造 以下のとおりです:



データ要素の構造(XMLSpy のスキーマビュー)

6.5.17.4 古い構成ファイルのアップグレード

EDI 構成ファイルには、**MapForceEDI** ディレクトリ内に含まれる XML スキーマの 1 つへの参照が含まれています(次を参照: EDI 構成スキーマ)。構成ファイルのスキーマとバージョンをビューするには、テキストエディターでファイルを開き、最初の行を確認してください:

```
<?xml version="1.0" encoding="UTF-8"?>
<Config xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="../EDIConfig5.xsd" Version="5">
```

MapForce 2021 は、MapForce 2011、リリース 3 から搭載されている **EDIConfig.xsd** スキーマバージョン 3 または、以降をベースにしたカスタム EDI 構成 を識別します。以前のバージョンと比較すると、このスキーマは同じ EDI コンポーネント内の複数のメッセージ型から、データをマップすることができます。バージョン 3 以前のスキーマバージョンをポイントするカスタム EDI 構成ファイルを使用する場合、構成ファイルをアップグレードする必要があります。それ以外の場合、カスタム EDI コレクションをマッピングに追加する場合、次のメッセージが「EDI コレクションの参照」ダイアログボックスに表示されます:

> "選択された EDI コレクションは、各コンポーネントに対し単一のメッセージ型をサポートする古い構成ファイルバージョンを使用しています。"

EDI 構成ファイルのスキーマを **EDIConfig.xsd** バージョン 2 から **EDIConfig.xsd** バージョン 3 にアップグレードする方法が次に説明されています。

構成ファイルのアップグレード:

1. カスタム化された EDI コレクション(例えば、"EDIFACT.Custom") を含むフォルダーに **Envelope.Config** を元の構成フォルダー(例えば、"EDIFACT") からコピーします。
2. **EDI.Collection**、を編集して、ルート要素の `Version` 属性を "2" から "3" に変更します。
3. `</Meta>` タグの後に **`<Root File="Envelope.Config"/>`** を追加します:

```
<?xml version="1.0" encoding="UTF-8"?>
<Messages xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" Version="3">
  <Meta>
    <Version>D</Version>
    <Release>04B</Release>
    <Agency>UN</Agency>
    </Meta>
    <Root File="Envelope.Config"/>
    <Message Type="ORDERS" File="ORDERS.Config" Description="Purchase order
message"/>
</Messages>
```

4. **ORDERS.Config**、を編集して、ルート要素の `Version` 属性を "2" から "3" に変更します。
5. **`<Format standard="EDIFACT"/>`** (、または場合によっては、X12、または HL7) を `</Meta>` タグの後に追加します。
6. **`<Group name="Message"...>`** を **`<Group name="Message_ORDERS"...>`** (または、カスタムメッセージ型)に名前を変更し、外部のグループレベルを削除します("Envelope"、"Interchange" およびそのセグメント) は以下のとおりです:

```
<?xml version="1.0" encoding="UTF-8"?>
<Config xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" Version="3">
   <Meta>
      <Version>D</Version>
      <Release>04B</Release>
      <Agency>UN</Agency>
   </Meta>
   <Format standard="EDIFACT"/>
   <Include href="Admin.Segment"/>
   <Include href="EDSD.Segment"/>
   <Include href="UNCL.Codelist"/>
   <Message>
      <MessageType>ORDERS</MessageType>
      <Description>Purchase order message</Description>
      <Revision>14</Revision>
      <Date>2004-11-23</Date>

      <!-- DELETED TEXT -->

               <Group name="Message_ORDERS" maxOccurs="unbounded" info="UNH -
Message header">
                  <Segment ref="UNH"/>
                  <Segment ref="BGM"/>
                        <Segment ref="UNT"/>
               </Group>

     <!-- DELETED TEXT -->

   </Message>
</Config>
```

構成ファイルを編集中にマッピング開かれている場合、再ロードされる必要があります。"Message" to "Message_ORDERS" アイテムから自動的に接続が再マップされます。

6.5.17.5 メッセージ型の追加または削除

カスタムEDI メッセージ型（または、X12 用語内のトランザクション型）を次の方法で追加することができます:

1. カスタムEDI コレクションが保管されているディレクトリ内に、コレクションに追加されるメッセージ（トランザクション）の各型のために **<Message>.Config** ファイルを作成します。
2. **EDI.Collection** ファイルを、使用することのできるメッセージ（トランザクション）型への参照を作成するために変更します。

既存のメッセージ型を削除する

- **EDI.Collection** ファイル内の必要とされる `Message` 要素 のコメントアウト。

<Message>.Config ファイルの追加および構成

作成する各 **<Message>.config** ファイルは、構成スキーマ（次を参照: EDI 構成ファイル）に対し妥当である必要があります。問題を回避するには、各メッセージ構成ファイルの本文内の次の要素に注目してください:

- Meta
- Format
- Include

- Message

Meta 要素は、サンプルのために、仕様バージョン、リリース、機関を表示します:

```
<Meta>
   <Version>D</Version>
   <Release>14B</Release>
   <Agency>UN</Agency>
</Meta>
```

Format 要素は、ベース標準の名前を示します、例:

```
<Format standard="EDIFACT"/>
```

以下の値は妥当です:

- Fixed
- EDIFACT
- X12
- HL7
- TRADACOMS (この値は、**EDIConfig4.xsd** スキーマまたは、以降、を必要とします、次を参照してください EDI 構成ファイルとスキーマ)

<Message>.Config ファイル内の Include 要素は、追加構成ファイルへの参照を指定します。例えば、下のコードリストは2つの .segment ファイルと1つの .codelist ファイルへの参照を含んでいます。ファイルは、すべてのセグメント、複合、およびデータ要素の定義をグローバルなレベルで保管します(これは、各メッセージ内でこれらが複数回発生することを回避するためです)。同様に、.codelist ファイルは、グローバルなレベルで、各フィールドに適用することのできる検証ルールを定義します。

```
<Include href="Admin.Segment"/>
<Include href="EDSD.Segment"/>
<Include href="UNCL.Codelist"/>
```

Include 要素による参照は、**<Message>.Config** ファイルで同じディレクトリで存在する必要があります。

各 **<Message>.Config** ファイル内の Message 要素は、メッセージの構造(すなわち、グループ、セグメント、およびデータ要素の階層構造)、および検証のために使用されるコードの値を(適用することができる場合)定義します。メッセージの構造に関する情報は、次を参照してください メッセージの構成の変更。既存の メッセージの変更のサンプルに関しては、次のトピックを参照してください:

- EDIFACT メッセージのカスタム化
- ANSI X12 トランザクションのカスタム化
- メッセージレベル(インライン) でコードの値を定義する

EDI.Collection ファイルの変更

カスタム **<Message>.Config** ファイルを追加すると、Message 要素が **EDI.Collection** ファイルに追加される必要があります。各 Message 要素 には次の属性が存在します:

型	メッセージ型の公式なコードを指定します(例: "BALANC", "999")。

(C) 2015-2021 Altova GmbH

ファイル	この特定のメッセージの型の仕様をふくむファイル名を指定します。実際のファイルは同じディレクトリに存在する必要があります。
説明	MapForce の視覚的なユーザーインターフェイスに表示されるこのメッセージの型の説明を提供します。

妥当な **EDI.Collection** ファイルのサンプルに関しては、**C:\Program Files\Altova\MapForce2021\MapForceEDI** ディレクトリ内のMapForce 内蔵のEDI コレクションを参照してください

6.5.17.6 メッセージの構成の変更

各 **<Message>.config** ファイル内では、`Message` 要素の構造がカスタム化されるEDI 標準のXML スキーマにより定義されています。このスキーマは、**EDIConfig.xsd**（または、類似した名前で呼ばれます）次を参照してください EDI 構成スキーマ。メッセージの構造を構築、または変更するには、カスタムEDI 標準とXML スキーマ。のドキュメントを確認する必要があります。

セグメント、データ要素、複合の追加方法

データ要素、セグメント、複合、およびサブ複合を追加するには、`<Data>`、`<Segment>`、`<Composite>` と`<SubComposite>` 要素 を以下のように作成します:

- .segment ファイルには、再利用できる、グローバルに定義されたセグメント、とデータ要素のリストが含まれている必要があります。
- .config ファイルは、各 メッセージ（または、トランザクション）型に対応しており、そのメッセージ（セグメントの階層構造、複合、および、データ要素）の実際の構造を含んでいる必要があります。

データ要素 を（.segment ファイル内で）1度定義することができるようになり、（.config ファイル内の）複数のメッセージ内で再利用することができます。

データ要素を参照するには、`ref` 属性を使用します。例えば、下のコードリストは、データ要素 F100 がHIPAA.X12 コレクションの **X12.Segment** ファイル内で定義されていることを示しています。

```
<Data name="F100" type="string" maxLength="3" minLength="3" info="Currency Code" />
```

X12.segment ファイル

すべてのセグメントでは、このデータ要素を必要とするすべてのトランザクション型内で、このデータ要素を参照できます。下のコードリストは "Health Care: Professional"（837P）トランザクションの "CUR" セグメントが前に定義されたF100 データ要素を参照していることを示しています:

```
<Include href="X12.Segment" />

...

<Segment name="CUR" minOccurs="0" info="Foreign Currency Information">
    <Data ref="F100" info="Currency Code" />
</Segment>
```

837P.Config ファイル（X12）

参照は、同じ ファイル、または外部 ファイルに存在することができます。上のコードリストでは、.config ファイルには外部 .segment ファイルが含まれています。このため、`<Include href="X12.Segment" />` ステートメントが、外部 **X12.Segment** ファイル。に参照を加えるために、**837P.Config** ファイルの最初に存在します。

カスタムセグメントの追加方法を説明する他のサンプルに関しては、次を参照してください サンプル: TA1 セグメントの追加.

グループとループの作成方法

EDI 構造を作成し、変更するには、親ノード（グループ）の下のセグメントまたはデータ要素をグループ化することを必要とします。このメッセージを構成する複数のセグメントをグループ化するため、EDI コンポーネント上のメッセージもグループです。グループは、ネストされている可能性があり、（子グループを持つこともできます）。また、複数回繰り返される可能性もあります（この場合は、ループになります）。グループとループは、<Group> 要素を追加することで作成することができます。maxOccurs 属性を使用して、ループの発生回数を設定します。例えば、次のコードリストは、ループが25回まで任意で発生でき、2つのセグメントにより構成されていることを示しています。

```
<Group name="LoopSAC" maxOccurs="25" minOccurs="0">
   <Segment ref="SAC"/>
   <Segment ref="CUR" minOccurs="0"/>
</Group>
```

850.Config ファイル（X12）

データ要素の型とサイズの指定する方法

データ要素またはセグメントの type 属性を使用して、（文字列、小数など）型を指定します。可能なデータ型（ type 属性）は、通常デフォルトで搭載されている構成ファイル内で使用されている XML スキーマ型であることができます。他の XML スキーマ単純型は、一時的に使用することができますが、保証はされません。次の単純型はサポートされていません: "anyType"、"ENTITIES" と "QName"。

データ要素の minLength と maxLength 属性を使用して、最小値 と 最大値を指定するために使用します。

必須と任意のデータ要素の指定方法

データ要素が必須であることを指定するには、 minOccurs 属性の値を設定しないでください（この属性には、デフォルトとして "1" を持ちます。これはフィールドがデフォルトで必須であることを意味します）。データ要素が任意であることを指定するには、 minOccurs 属性を "0" に設定します。例えば、 下のコードリストは 2つの必須のデータ要素（F1331、F1332）と1つの任意のデータ要素（F1325） を定義しています。

```
<Composite name="C023" info="Health Care Service Location Information">
   <Data ref="F1331"/>
   <Data ref="F1332"/>
   <Data ref="F1325" minOccurs="0"/>
</Composite>
```

6.5.17.7 複数の連続したデータ要素の扱い

EDI セグメントは、同じ名前の複数の連続した要素の使用を許可します。例えば、 X12 標準の "850 - PURCHASE ORDER" トランザクションセットの N2 セグメントに適用することができます。仕様に従うと、N2 セグメントは、（1つの必須、1つの任意）2つの連続した英数字 "Name" フィールド を許可します:

REF	ELE ID	NAME	RPT	ATTRIBUTES
01	93	Name	M	AN 1/60
02	93	Name	O	AN 1/60

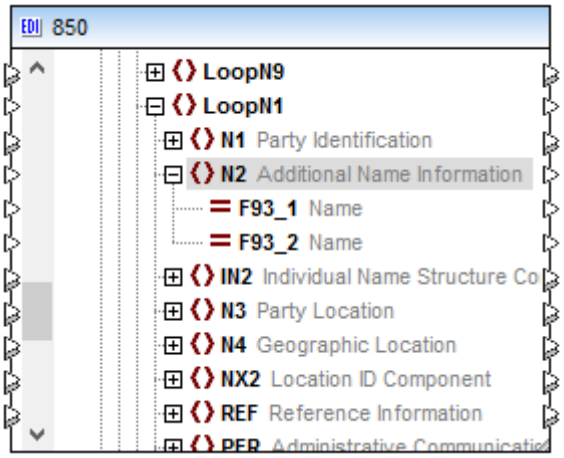
EDI 構成を作成、または変更するために、データ要素を扱うには以下の2つの可能性があります。

アプローチ 1: 各連続したデータ要素のための個別の マッピング ノードを作成する

このアプローチでは、セグメント ファイル内のデータ要素の複数の発生を作成します。この場合、新規のノードは、データ要素の各発生のためにマッピング上に表示されます。この構成を考慮してください:

```
<Segment name="N2" info="Additional Name Information">
   <Data ref="F93" nodeName="F93_1"/>
   <Data ref="F93" minOccurs="0" nodeName="F93_2"/>
</Segment>
```

マッピング上では、この構成は、2つの個別のF93 ノード, を作成します。それぞれは、F93 の発生に対応しています:



個別のF93 ノードを持つX12 EDI コンポーネント

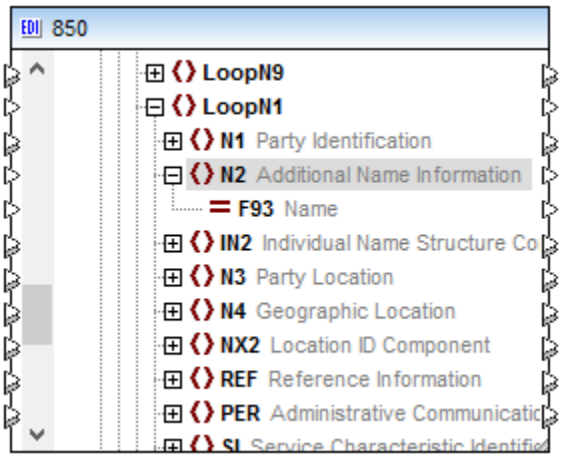
このアプローチの使用は、データ要素の発生が意味を成す場合役にたちます。このため、マッピング上に個別のノードを表示するためにも役に立ちます。

アプローチ 2: データ要素のすべての発生のために単一ノードを作成する

このアプローチでは、データ要素のすべての発生を単一のノードとして扱います。これを行うには、属性を対応するデータ要素に追加します。下のコードリスト内は、データ要素 F93 の `mergedEntries` 属性 は"2" に設定されています。EDI インスタンス ファイル内のこのフィールドの2つの連続した 発生を許可します。

```
<Segment name="N2" info="Additional Name Information">
   <Data ref="F93" mergedEntries="2"/>
</Segment>
```

マッピング上では、F93 データ要素に対応するノードは、単一ノードとして表示されます:



個別のF93 ノードを持つX12 EDI コンポーネント

ソースEDI ファイルにこの要素の(空ではない)複数の発生が存在する場合、エントリをマージしたデータ要素からのデータのマッピングは、ターゲット側に複数の複数要素を作成します。例えば、N2 セグメントが2つの連続した発生を含む箇所のソースEDI ファイルを想定します。最初の発生は、"Michelle Butler" で、2番目の発生は、"Mrs" です。

```
N2+Michelle Butler+Mrs+'
```

このようなEDI ファイルからのデータのマッピングは、ターゲット側に次の出力を作成します(<ContactName> の発生に注意してください):

```
<Customer>
      <Number>123</Number>
      <ContactName>Michelle Butler</ContactName>
      <ContactName>Mrs</ContactName>
      <CompanyName>Nanonull, Inc.</CompanyName>
</Customer>
```

データ要素の各発生が意味をなさない場合、また、各発生のために、マッピングに表示するためにノードを分割することを希望しない場合、このアプローチの使用は適切です。例えば、複数の発生を持つデータ要素は、住所を構成する複数のラインエントリを保管することができます。このような場合、"マージされたエントリ" のアプローチは、各住所のラインのために個別のノードを使用するより先役に立つ可能性があります。

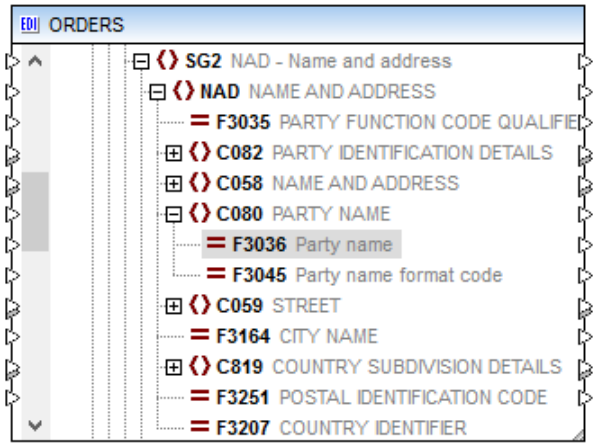
MapForce によりデフォルトで与えられる構成ファイルの一部は、マージされたエントリを使用する場合があり、このため、データ要素の複数の連続した発生のためのマッピング上で1つのノードのみが表示されます。データ要素の各発生のために、個別のノードが必要な場合は、EDI 構成をカスタム化して行ってください。例えば、次を参照してください [マージされたエントリを異なるノードに分割する](#)。

6.5.17.8 マージされたエントリを異なるノードに分割する

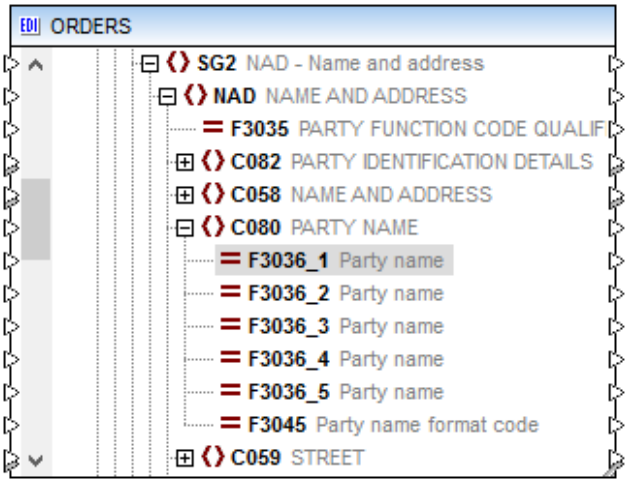
MapForce によりデフォルトで与えられるEDI 構成ファイル内のデータ要素の一部は、ソースEDI ファイル内で対応するデータ要素が複数回発生する可能性があるにも関わらず、マッピング上で1つのノードのみを表示するように構成されている場合があります(次も参照: [複数の連続したデータ要素の扱い](#))。このようなデータ要素の一例は、UN/EDIFACT 標準のF3036 です。仕様に従うと、F3036 は、("Party name") 複合内の C080 内で連続して5回まで発生することができます(次を参照: http://www.unece.org/fileadmin/DAM/trade/untdid/d15b/trcd/trcdc080.htm)。

MapForce 内のデフォルトのUN/EDIFACT コレクションでは、データ要素 F3036 は、マッピング上で単一のノードで表示されています。マッピング接続をこのノードからターゲットノードに描くと、ソース内のF3036 の発生が存在するため、接続はターゲット内に制限なくアイテムを作成することができます。(これには、ソース内の各アイテムのために、ターゲット内に 1つのアイテムが作成される MapForce 接続の基本ルールに従っています。)。しかしながら、これが希望される結果ではない場合、データ要素の各発生のために個別のノードが必要とされます。

この例の目的は、ですから、F3036 の各発生のための個別のマッピングノードを表示するカスタムUN/EDIFACT 構成を作成することです。下のイメージは、カスタム化の前後のEDIFACT マッピングコンポーネントを表示しています。



カスタム化の前



カスタム化の後

F3036 データ要素を複数のマッピングノードに以下のように分割します:

1. 既存のUN/EDIFACT コレクションを開始点として使用し、カスタムEDI コレクションを作成します(次を参照: カスタムEDI コレクションの作成)。これにより、変更は、EDI 構成には、影響をおよぼさないことを保証します。
2. XML エディター内でカスタムコレクションの**EDSD.Segment** ファイルを開きます。(EDI フレーバーをカスタム化する場合、特定のEDI フレーバーの対応する.segment ファイルを変更します。例えば、X12 のためには、**X12.Segment** ファイルになります。)
3. 複合 C080 を検索し、F3036 データ要素の`mergedEntries="5"` 属性を削除します。

```
<Composite name="C080" info="PARTY NAME">
   <Data ref="F3036" mergedEntries="5"/>
   <Data ref="F3045" minOccurs="0"/>
</Composite>
```

4. `minOccurs="0"` 属性を、最初のフィールド以外のすべてのフィールドに追加します。理由は、UN/EDIFACT 標準によると、フィールドの最初の発生のみが必須であるからです。その他は任意です。
5. 各フィールドのために一意のノード名を入力します(例えば、F3036_1、F3036_2、など)。この時点では、構成は以下のようになります:

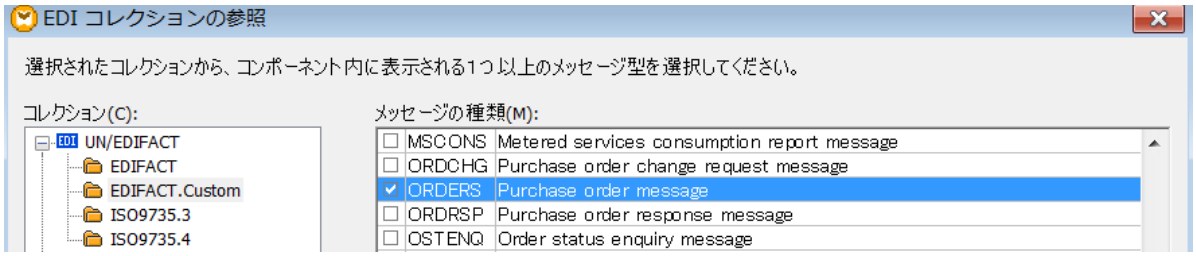
```
<Composite name="C080" info="PARTY NAME">
   <Data ref="F3036" nodeName="F3036_1"/>
   <Data ref="F3036" nodeName="F3036_2" minOccurs="0"/>
   <Data ref="F3036" nodeName="F3036_3" minOccurs="0"/>
   <Data ref="F3036" nodeName="F3036_4" minOccurs="0"/>
   <Data ref="F3036" nodeName="F3036_5" minOccurs="0"/>
```

```
    <Data ref="F3045" minOccurs="0"/>
</Composite>
```

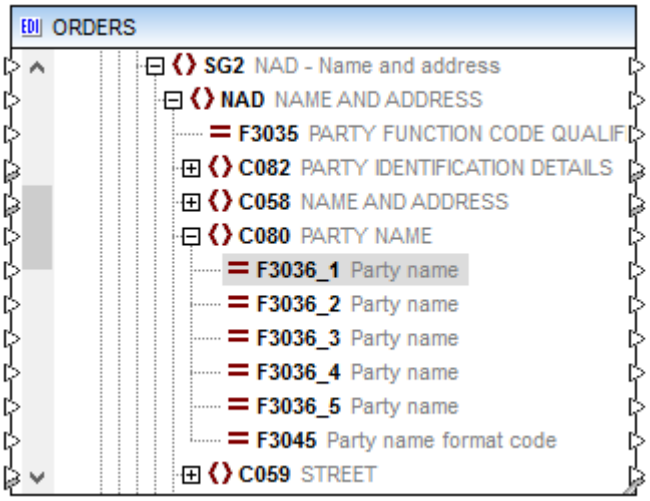
6. **EDSD.Segment** ファイルを保存して編集します。

MapForce 内で変更されたEDI コレクションを以下の方法で確認、および作業することができます:

1. 「挿入」メニューから、「EDI」をクリックします。
2. プロンプトされると、作成済みのカスタムコレクションを選択します(このサンプルでは、"EDIFACT.Custom")。
3. C080 セグメントを使用するメッセージ型を選択してください(このサンプルでは、"ORDERS - Purchase Order Message")。



変更されたEDI コンポーネントがMapForce 内に表示されます(F3036 データ要素の各発生には、マッピング上で個別のノードが存在します)。



EDI フレーバーに関係なく、このサンプル内で説明されているこのテクニックは、他のデータ要素のために マージされたエントリを分割するために使用することができます。

6.5.17.9 データ要素のコードの値の変更の可能性

データ要素の可能の値は構成ファイルを使用して変更することができます。他の構成情報同様、メッセージレベル(ローカルでのカスタム化)、またはグローバル(特定のデータ要素を使用する可能性のあるすべてのメッセージで)で可能なコード値を構成することができます。

他のカスタムの構成のように、コード値をカスタマイズするには、デフォルトのMapForce 構成を変更する代わりに、カスタムEDI コレクションを作成し、すべてのカスタム化を適用することが奨励されます（次を参照: カスタムEDI コレクションの作成）。

使用することのできるカスタム化オプションは、カスタムEDI コレクションにより使用されるスキーマバージョンにより異なります（次を参照: EDI 構成スキーマ）。つまり、新しいスキーマに対して更に高度な構成オプションを使用することができるのです。特に、各スキーマはカスタムコードを以下のようにサポートします:

- **EDIConfig.xsd**、を使用する場合、グローバルなレベルでカスタムコードを追加するために既存の.codelist ファイルを変更することができます。
- **EDIConfig4.xsd**、を使用する場合、カスタムコード値をグローバルなレベルだけではなく、メッセージレベルで追加することができます。
- **EDIConfig5.xsd**、を使用する場合、高度な構成ルールを文字レベルで定義する機能を含む、上記のすべてのオプションを使用することができます。（次を参照: 文字レベルの検証の構成）。また、データ要素には不完全な検証が存在すると宣言することができます（次を参照: フィールドの検証の弛緩）。

グローバルなカスタム化

データ要素のためにグローバルにカスタムコード値を追加するには、カスタムEDI コレクションの.codelist ファイルを編集します。最初に、値を変更するデータ要素（フィールド）検索し、新しい<Value> 要素として必要とされるカスタム値をすべて以下に示されるように追加します:

```
<?xml version="1.0" encoding="UTF-8"?>
<Config xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" Version="5"
xsi:noNamespaceSchemaLocation="../EDIConfig5.xsd">
   <Include href="ISO6346.Codelist"/>
   <Codelist>
      <Field Name="F1001" Length="3" Type="Alphanumeric">
         <Title>Document name code</Title>
         <Description>Code specifying the document name.</Description>
         <Values>
            <Value Code="1">
               <Title>Certificate of analysis</Title>
               <Description>Certificate providing the values of an
analysis.</Description>
            </Value>

>>> add your custom values here...

         </Values>
      </Field>
   </Codelist>
</Config>
```

UNCL.Codelist ファイル（UN/EDIFACT コレクション）

上のコードリストに示されているように、各 <Value> 要素には、このデータ要素のために期待される実際の値コードを指定するCode 属性が存在します。<Title> と<Description> 要素は、この値コードに関する詳細の情報を提供します。

上のコードリストには、MapForce に**UNCL.Codelist** ファイルを同じスキーマを使用する、。外部ファイル（**ISO6346.Codelist**）内で定義されている追加コード値を検索するように命令するInclude ステートメントが存在します。

例えば、次を参照してください 外部ソースからコードの値を適用する。

ローカルなカスタム化

メッセージ レベルでカスタムコード値を追加するには、メッセージ型に対応した.config ファイルを編集する必要があります。詳細に関しては、次を参照してください: メッセージレベル(インライン) でコードの値を定義する。

6.5.17.10 例: TA1 セグメントの追加

TA1 セグメントは、任意のセグメントで、インターチェンジのレセプション、および、エンベロープ セグメントの文法的な正確性を確認するために使用されます。構成ファイルを変更することにより、セグメントを MapForce 内のEDI X12 とEDI HIPAA コンポーネントに追加することができます。

EDI コンポーネント内に TA1 セグメントを含む:

1. HIPAA.X12 コレクションをモデルにして、カスタムEDI コレクションを作成します(次を参照: カスタムEDI コレクションの作成)。これは、変更がデフォルトの構成に影響を与えないことを保証します。
2. カスタムEDI コレクションの**Envelope.config** ファイルを XML エディター内で開きます。
3. ISA とGroup セグメントの間のテキスト `<Segment ref="TA1" minOccurs="0" maxOccurs="unbounded"/>` を解析します。そしてファイルを保存します。これは以下のようになります:

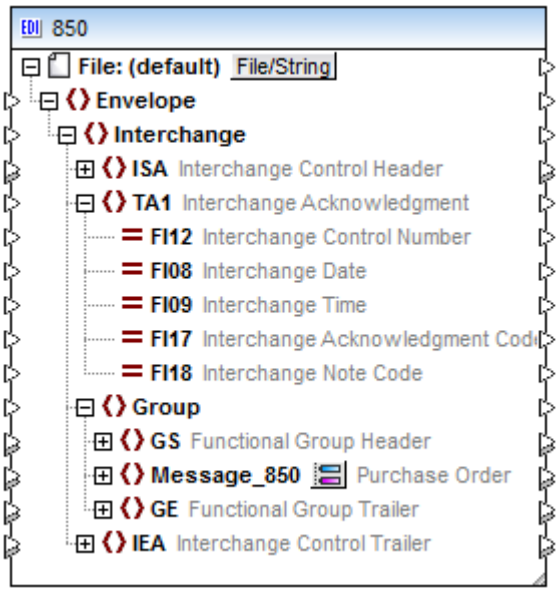
```
<?xml version="1.0" encoding="UTF-8"?>
<Config xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="../EDIConfig.xsd" Version="3">
   <Meta>
      <Release>6020</Release>
      <Agency>X12</Agency>
   </Meta>
   <Format standard="X12"/>
   <Include href="X12.Segment"/>
   <Include href="X12.Codelist"/>
   <Include collection="EDI.Collection"/>
   <Group name="Envelope">
      <Group name="Interchange" maxOccurs="unbounded">
         <Segment ref="ISA" minOccurs="0"/>
         <Segment ref="TA1" minOccurs="0" maxOccurs="unbounded"/>
         <Group name="Group" maxOccurs="unbounded">
            ...
         </Group>
         <Segment ref="IEA" minOccurs="0"/>
      </Group>
   </Group>
</Config>
```

上でハイライトされたラインは、**X12.Segment** ファイル内で既に定義されている TA1 セグメントに参照を追加します。MapForce 内で TA1 セグメントの構造を確認することができます:

1. 「挿入」メニューから、「EDI」をクリックします。
2. 「EDI コレクションの参照」ダイアログボックスから、作成されたカスタムコレクションとメッセージ型を選択します(例えば、"850 - Purchase Order")。

カスタム化されたコンポーネントには、TA1 セグメントが追加されています。必要に応じてこのカスタム化されたコンポーネントにマッピングの接続を描くことができます。

(C) 2015-2021 Altova GmbH



任意のTA1 セグメントが追加されたカスタム化された X12 EDI コンポーネント

6.5.17.11 例: EDIFACT メッセージのカスタム化

この例は、EDIFACT メッセージをカスタム化し、MapForce が非標準、または変更されたEDIFACT フォーマットを処理する方法について説明しています。カスタマイズのサンプルは具体的にEDIFACT のバージョン **19B** をベースにしています。命令は他のEDIFACT バージョンとは若干異なります。

ORDERS メッセージの若干拡張されたCTA (Contact Information) セグメントを使用して、カスタムEDIFACT ファイルを処理するサンプルのマッピングは次のパスで見つけることができます: **<マイドキュメント>\Altova\MapForce2021\MapForceExamples\Tutorial\ExtractCustomEDIFACT.mfd**。このマッピングを次のカスタム化を行う前に開いた場合、MapForce は、正確に検証できないことに注意してください。この理由は、マッピングが、使用中のマシン上にまだ存在しないカスタムEDIFACT 構成を参照しているからです(この例の中でこの構成は作成されます)。

Orders-Custom.EDI ファイルからデータを読み取るサンプルのマッピングは**<マイドキュメント>\Altova\MapForce2021\MapForceExamples\Tutorial** ディレクトリ内にあります。このファイルをテキストエディターで開くと、カスタム化されたCTA セグメントが使用されていることに気がつきます。具体的には、9 列目では、「**Mr** 」エントリを含みライン 11 列目では、「 **Mrs** 」エントリを含みます:

Orders-Custom.edi

```
1  UNB+UNOB:1+003897733:01:MFGB+PARTNER ID:ZZ:ROUTING ADDR+970101:1230+00000000000001++ORDERS++++1'
2  UNH+0001+ORDERS:D:19B:UN'
3  BGM+221+ABC123456XYZ+9'
4  DTM+4:202004301742PDT:303'
5  FTX+PUR+3++Pizza purchase order'
6  RFF+CT:123-456'
7  RFF+CR:1122'
8  NAD+SE+999::92++24h Pizza+Long Way+San-Francisco+CA+34424+US'
9  CTA+SR+:Ted Little:Mr'
10 NAD+BY+123::92++Nanonull, Inc.+119 Oakstreet Suite 4876+Vereno+CA+29213+US'
11 CTA+PD+:Michelle Butler:Mrs'
12 NAD+ST+123::92++Nanonull, Inc.+769 Oakstreet Suite 4876+Vereno+CA+29213+US'
13 TAX+9++++++1-12345-6789-0'
14 CUX+2:USD:9'
15 PCD+12:2'
```

上記のようなカスタムEDIファイルからデータをマップする前に、カスタムEDIFACT構成が作成される必要があります。この例では、個人のタイトルに対応するようにCTAセグメントに新規フィールドを持つように拡張するために、デフォルトのEDIFACT構成をカスタマイズします。参照のために、カスタマイズの最終結果は、ZIPアーカイブ(**EDIFACT.Nanonull.zip**)として**<マイドキュメント>\Altova\MapForce2021\MapForceExamples\Tutorial**ディレクトリ内にあります。

必要条件

このサンプルを完了するために、最初に[EDIFACT構成ファイル]実行可能ファイルをAltovaダウンロードセンターからダウンロードしてください(https://www.altova.com/ja/mapforce/download)。MapForceのバージョンとプラットフォーム(32-ビット、64-ビット)に適用可能なパッケージが選択されていることを確認してください。

このインストールパッケージはこのサンプル内で使用されているバージョン[19B]を含むMapForceによりサポートされているすべてのEDIFACTバージョンを含んでいます。インストールを実行するには、ダウンロードされた実行可能ファイルをダブルクリックし、画面上の指示に従います。インストール後、インストールされたEDIFACT構成バリエーションに対応した新規のディレクトリが**C:\Program Files\Altova\MapForce2021\MapForceEDI**内に作成されます。

32-ビットMapForceを64-ビットオペレーティングシステム上で作動している場合、上のディレクトリパスを**C:\Program Files (x86)**に調整してください。

カスタムEDIコレクションの作成

1. カスタムEDIコレクションを保管する新規のディレクトリを作成します: **C:\users\<name>\Documents\Altova\MapForce2021\MapForceEDI\EDIFACT.Nanonull**. [EDIFACT.Nanonull]という名前はEDIコレクションとMapForceによりインストールされたデフォルトのコレクションを区別する手助けをしています。
2. **C:\Program Files\Altova\MapForce2021\MapForceEDI\EDIFACT.2019B**から**C:\users\<name>\Documents\Altova\MapForce2021\MapForceEDI\EDIFACT.Nanonull**に以下のファイルをコピーします:

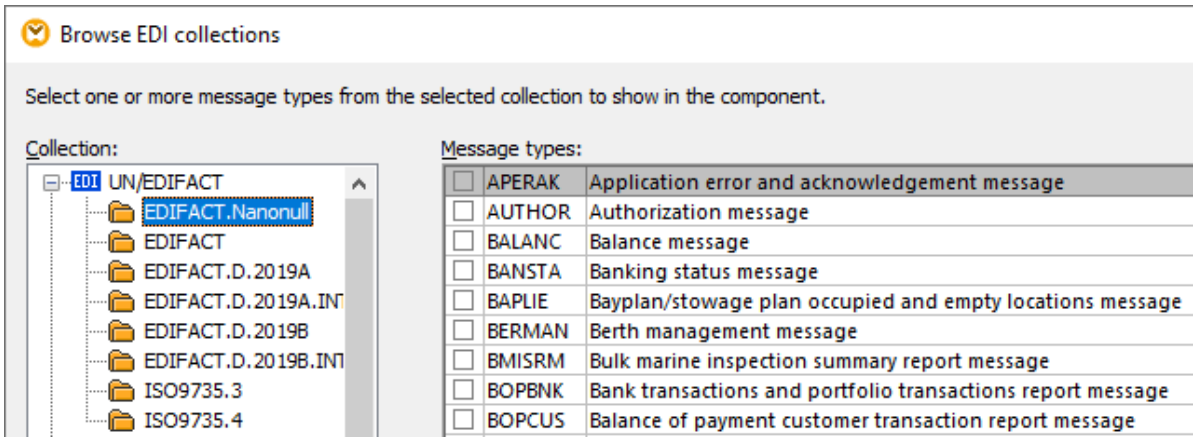
   - Admin.Segment
   - EDI.Collection
   - EDSD.Segment
   - Envelope.Config
   - ISO6346.Codelist
   - ORDERS.Config
   - UNCL.Codelist

(C) 2015-2021 Altova GmbH

ORDERS.Config 以外はこれらすべてのファイルは**Envelope.Config** ファイルにより参照されておりEDI コレクション内に存在する必要があります。各ファイルの目的に関する詳細に関しては[EDI 構成ファイル]を参照してください。

新規の**EDI.Collection** ファイルは、MapForce により使用できるようになっています。以下のようにこれをテストすることができます:

1. MapForce を開始します(または作動している場合再起動します)。
2. 「挿入」メニューから「EDI」をクリックします。または、「EDI の挿入」 ツールバーボタンをクリックします。[EDIFACT.Nanonull] という名前を持つ新規のコレクションを表示したダイアログボックスが開かれます。



メモ [EDI コレクションの作成]で説明されているとおり自身のカスタムEDI コレクションを別のディレクトリに作成することができます。

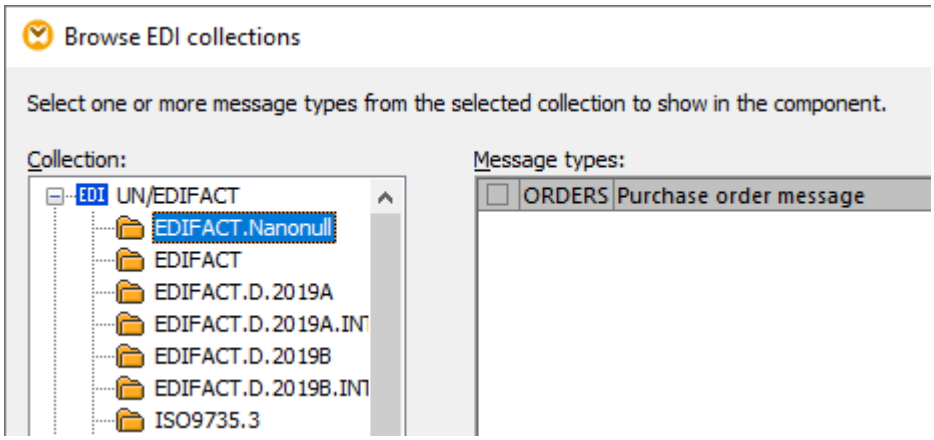
EDI.Collection ファイルを構成する

1. (例えば、XMLSpy などの) XML エディターで**EDI.Collection** ファイルを開きます。
2. "ORDERS" メッセージ以外のすべての"Message" 要素を削除します。ファイルは以下のようになります:

```
<?xml version="1.0" encoding="UTF-8"?>
<Messages Version="3">
   <Meta>
      <Version>D</Version>
      <Release>19B</Release>
      <Agency>UN</Agency>
   </Meta>
   <Root File="Envelope.Config"/>
   <Message Type="ORDERS" File="ORDERS.Config" Description="Purchase order
message"/>
</Messages>
```

3. ファイルを保存します。

[挿入 | EDI] メニューコマンドを実行すると、コレクションはメッセージ型のみを表示します: "ORDERS"。



グローバルなカスタマイズとインラインなカスタマイズ

この例の目的は、CTA (Contact Information) セグメントを再定義することです。CTA セグメントは、1つのフィールド (F3139) と1つのコンポジット (C056) により構成されています。(例えば、"Mr" または、"Mrs" などの) タイトルデータを保管するには、C056 コンポジットに X1000 と呼ばれる新規のフィールドを追加します。カスタマイズを行うには複数の方法があります:

- **EDSD.segment** ファイルをカスタマイズすることにより、グローバルにカスタマイズを行うことができます。全てのメッセージ内の全てのセグメントは、コンポジット C056 を使用し、新しい要素を含むまたは参照します。
- , **ORDERS.Config** ファイルをカスタマイズすることにより、インラインにカスタマイズを行うことができます。現在のメッセージ内のカスタマイズされたセグメント (CTA) のみが新規の要素を含みます。

グローバルなカスタム化

新規の X1000 フィールドにグローバルにアクセスを作成する場合、**EDSD.Segment** ファイルのみに変更が加えられます。コンポジット C056 を使用する全てのメッセージ内の全てのセグメントは、新しい要素を含むまたは参照します。

コンポジット C056 をグローバルに再定義します:

1. **EDSD.Segment** ファイルをテキストまたは XML エディター内で開き、「**構成 | 要素 | コンポジット | C056**」にナビゲートします。

```
<Composite name="C056" info="CONTACT DETAILS">
  <Data ref="F3413" minOccurs="0"/>
  <Data ref="F3412" minOccurs="0"/>
</Composite>
```

2. 次のラインを C056 セグメントの "F3412" の下に挿入します:

```
<Data name="X1000" type="string" maxLength="35" minOccurs="0" info="New
Element"/>
```

`Composite` 定義は以下のようになります::

```
<Composite name="C056" info="CONTACT DETAILS">
  <Data ref="F3413" minOccurs="0"/>
  <Data ref="F3412" minOccurs="0"/>
```

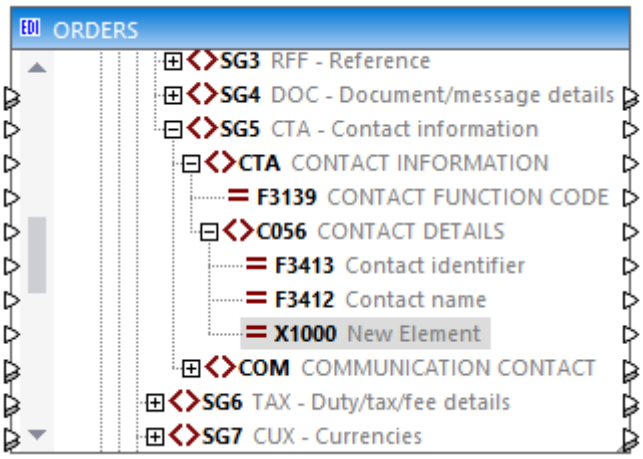

(C) 2015-2021 Altova GmbH

```
  <Data name="X1000" type="string" maxLength="35" minOccurs="0" info="New
Element"/>
</Composite>
```

上のコードリストでは、"ref" 属性。を使用するセグメントの他のフィールドに対して、新規のX1000 フィールドが"name" 属性を使用して定義されています。他の2つのフィールドは、ファイルの最初に定義されており、参照されているだけです。

C056 を使用する全てのメッセージで新規のX1000 フィールドを使用することができます。MapForce 内の新しいフィールドを以下のようにプレビューすることができます:

1. 「挿入」メニューから「EDI」をクリックします。または、「EDI の挿入」 ツールバーボタンをクリックします。
2. "EDIFACT.Nanonull" コレクションをクリックして、ORDERS メッセージを選択します。
3. ソースEDI ファイルを選択するように問われると、「スキップ」をクリックします。ORDERS が、マッピングウィンドウ内で表示されています。
4. コンポーネントヘッダーをクリックし、「Ctrl + F」を押して、"X1000" を検索テキストとして入力します。「次を検索」をクリックし、(**Envelope > Interchange > Group > Message_ORDERS > SG2 > SG5 > CTA > C056** の下に存在する)新規のX1000 要素の次の発生に移動します。



インラインのカスタマイズ

新規のX1000 フィールドへ*ローカルに*(または、*インラインで*) アクセスするには、**ORDERS.config** ファイルのみに変更が加えられなければなりません。この場合、現在のメッセージ内の再定義された CTA セグメントは、新しい要素を含むまたは参照します。すなわち、CTA セグメントは、カスタム新規のフィールド X1000 を持つ再定義されたコンポジット C056 を含むようにローカルで再定義されます。

コンポジット C056 をローカルで再定義する

1. **ORDERS.Config** ファイルをテキストまたはXML エディター内で開き、`Group name="SG5"` にナビゲートします(または **SG5**を検索します).

```
<Group name="SG5" minOccurs="0" maxOccurs="5" info="CTA - Contact information">
   <Segment ref="CTA"/>
   <Segment ref="COM" minOccurs="0" maxOccurs="5"/>
</Group>
```

2. ライン `<Segment ref="CTA"/>` を以下のラインと置き換えます:

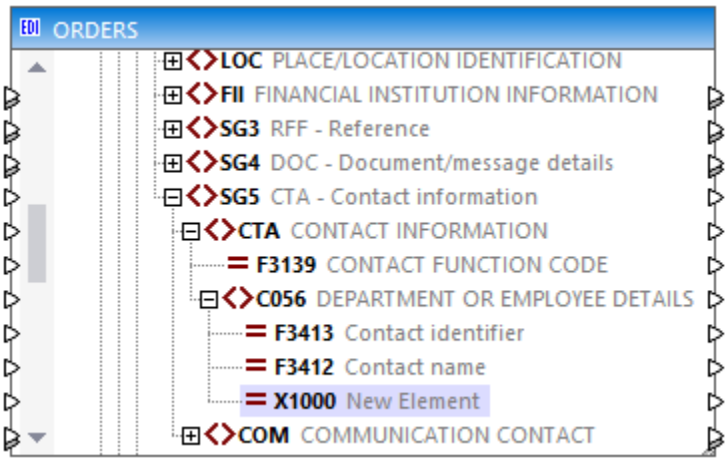
```
<Segment name="CTA" id="CTA_ORDERS_SG5" info="CONTACT INFORMATION">
    <Data ref="F3139" minOccurs="0"/>
    <Composite name="C056" minOccurs="0" info="DEPARTMENT OR EMPLOYEE DETAILS">
           <Data ref="F3413" minOccurs="0"/>
           <Data ref="F3412" minOccurs="0"/>
           <Data name="X1000" type="string" maxLength="35" minOccurs="0"
info="New Element"/>
    </Composite>
</Segment>
```

Group 定義は以下のようになります:

```
<Group name="SG5" minOccurs="0" maxOccurs="5" info="CTA - Contact information">
   <Segment name="CTA" id="CTA_ORDERS_SG5" info="CONTACT INFORMATION">
    <Data ref="F3139" minOccurs="0"/>
    <Composite name="C056" minOccurs="0" info="DEPARTMENT OR EMPLOYEE DETAILS">
           <Data ref="F3413" minOccurs="0"/>
           <Data ref="F3412" minOccurs="0"/>
           <Data name="X1000" type="string" maxLength="35" minOccurs="0"
info="New Element"/>
    </Composite>
   </Segment>
   <Segment ref="COM" minOccurs="0" maxOccurs="5"/>
</Group>
```

MapForce 内の新しいフィールドを以下のようにプレビューすることができます:

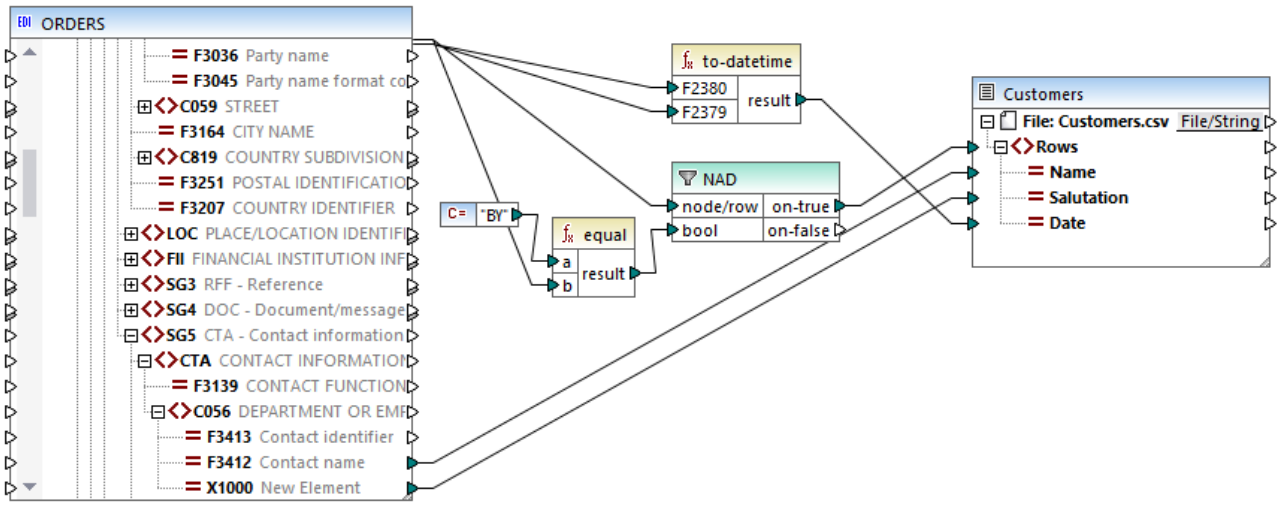
1. 「挿入」メニューから「EDI」をクリックします。または、「EDI の挿入」 ツールバーボタンをクリックします。
2. "EDIFACT.Nanonull" コレクションをクリックして、ORDERS メッセージを選択します。
3. ソース EDI ファイルを選択するように問われると、「スキップ」をクリックします。ORDERS が、マッピングウィンドウ内で表示されています。
4. **Envelope > Interchange > Group > Message_ORDERS > SG2 > SG5 > CTA > C056** にナビゲートすると、新規の X1000 要素を確認することができます。



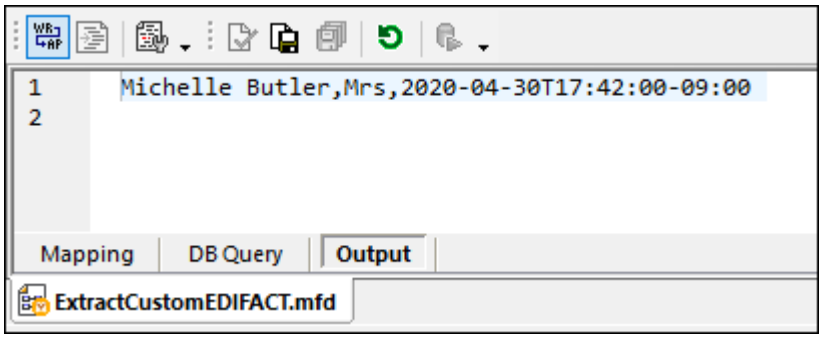
(C) 2015-2021 Altova GmbH

マッピング内でカスタマイズされたメッセージを使用する

上記の手順に従った場合、カスタムEDIFACT 構成 をマッピング内で使用することができます。上記のカスタムEDIFACT 構成を使用するマッピングの例は、<マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥Tutorial¥ フォルダー内の **ExtractCustomEDIFACT.mfd** です。この中では**ORDERS-Custom.EDI** ファイルがOrder-EDI スキーマにマップされます。カスタムフィールドがEDI 構造X1000,に追加され、ターゲットスキーマ内のSalutation アイテムにマップします。



マッピングの結果をプレビューするには、「出力」タブをクリックします。予想された通りEDIFACT ファイルのカスタムX1000 データ要素から抽出された値 "Mrs" はCSV ファイル("Salutation") の2番目のフィールドに表示されます。



MapForce 内の他のマッピングデザインファイルと類似して、EDI マッピングは、MapForce 外で実行されることもできます。例えば、他のサーバーにデプロイし、MapForce Server または、FlowForce Server で実行する、またはプログラムコードを生成し、マッピングを実行することなど。この例のスコープのために、C++ コードを生成してマッピングを実行することを選択した場合、"CC056Type" クラスからアクセスすることのできる "CX1000Type" という名前のクラスが生成されます。

6.5.17.12 例: ANSI X12 トランザクションのカスタム化

この例は非標準またはX12 フォーマットに変更されたフォーマットを処理できるようにMapForce のカスタム化の方法を説明しています。カスタマイズのサンプルは具体的にANSI X12 のバージョン **6020** をベースにしています。命令は他のX12 バージョンとは若干異なります。

サンプルには以下のパスで見つけることのできるサンプルマッピングが存在します: **<マイドキュメント>\Altova\MapForce2021\MapForceExamples\Tutorial\ExtractCustomX12.mfd**。このマッピングを次のカスタム化を行う前に開いた場合、MapForce は、正確に検証できないことに注意してください。この理由は、マッピングが、使用中のマシン上にまだ存在しないカスタム X12 コレクション構成を参照しているからです(この例の中でこの構成は作成されます)。

Orders-Custom.X12 ファイルからデータを読み取るサンプルのマッピングは**<マイドキュメント>\Altova\MapForce2021\MapForceExamples\Tutorial** ディレクトリ内にあります。このファイルをテキストエディターで開くと、カスタム化された N2 セグメントが使用されていることに気がつきます。具体的には行 6 にお追加の++**Mrs** エントリが含まれています:

Orders-Custom.X12

```
1  ISA+00+          +00+          +ZZ+SenderID       +ZZ+ReceiverID
2  GS+PO+SenderID+ReceiverID+20060308+182347+1+T+005011'
3  ST+850+12345'
4  BEG+00+SA+ABC123456XYZ++20200430'
5  N1+001+Nanonull, Inc.++123'
6  N2+Michelle Butler++Mrs'
7  N3+119 Oakstreet Suite 4876'
8  N4+Vereno+CA+29213'
9  PO1++1++7.2'
10 PO3+AQ+++7.2++1+US'
11 PID+F++++Pizza Pepperoni'
```

上記のような カスタム X12 ファイルからデータをマップする前に、カスタム X12 構成が作成される必要があります。この例では、個人のタイトルに対応するように N2 セグメントに新規フィールドを持つように拡張するために、デフォルトの X12 構成をカスタマイズします。参照のために、カスタマイズの最終結果は、ZIP アーカイブ(**X12.Nanonull.zip**)として**<マイドキュメント>\Altova\MapForce2021\MapForceExamples\Tutorial** ディレクトリ内にあります。

必要条件

このサンプルを完了するために、最初に[X12 構成ファイル] 実行可能ファイルを Altova ダウンロードセンターからダウンロードしてください (https://www.altova.com/ja/mapforce/download)。MapForce のバージョンとプラットフォーム(32-ビット、64-ビット)に適用可能なパッケージが選択されていることを確認してください。

このインストールパッケージはこのサンプル内で使用されているバージョン[6020] を含む MapForce によりサポートされている すべての X12 バージョンを含んでいます。インストールを実行するには、ダウンロードされた実行可能ファイルをダブルクリックし、画面上の指示に従います。インストール後、インストールされた X12 構成バリエーションに対応した新規のディレクトリが **C:\Program Files\Altova\MapForce2021\MapForceEDI** 内に作成されます。

32-ビット MapForce を 64-ビット オペレーティングシステム上で作動している場合、上のディレクトリパスを **C:\Program Files (x86)** に調整してください。

カスタムファイルのセットアップ方法

1. カスタム EDI コレクションを保管する新規のディレクトリを作成します: **C:\users\<name>\Documents\Altova\MapForce2021\MapForceEDI\X12.Nanonull**.[X12.Nanonull] という名前は EDI コレクションと MapForce によりインストールされたデフォルトのコレクションを区別する手助けをしています。
2. **C:\Program Files\Altova\MapForce2021\MapForceEDI\X12.6020** から以下のファイルを **C:\users\<name>\Documents\Altova\MapForce2021\MapForceEDI\X12.Nanonull** にコピーします:

   - 850.Config
   - EDI.Collection

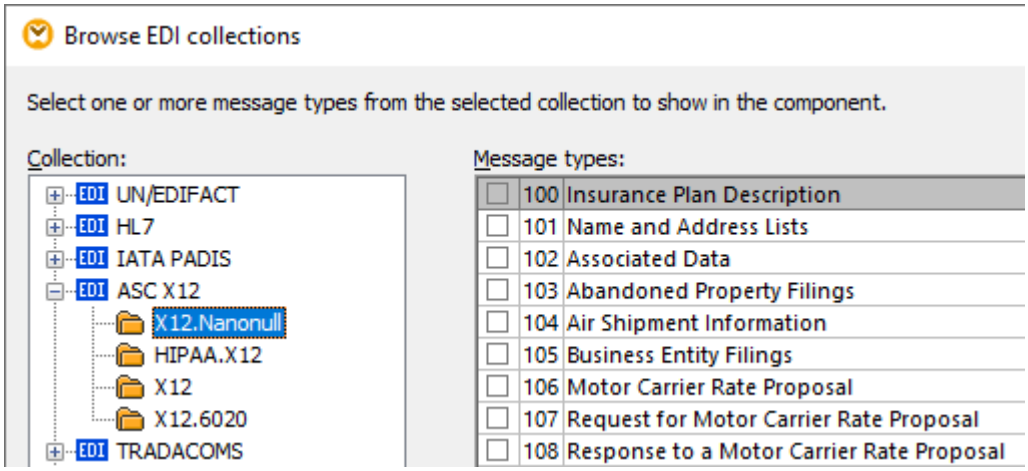
(C) 2015-2021 Altova GmbH

- Envelope.Config
- ParserErrors.Config
- X12.Codelist
- X12.Segment

各ファイルの目的に関する詳細に関しては EDI 構成ファイル を参照してください。

新規の **EDI.Collection** ファイルは、MapForce により使用できるようになっています。以下のようにこれをテストすることができます:

1. MapForce を開始します。
2. 「挿入」メニューから「EDI」をクリックします。または、「EDI の挿入」 ツールバーボタンをクリックします。"X12.Nanonull" という名前の新規のコレクションを表示する以下のようなダイアログボックスが開かれます。



メモ EDI コレクションの作成 で説明されているとおり自身のカスタム EDI コレクションを別のディレクトリに作成することができます。

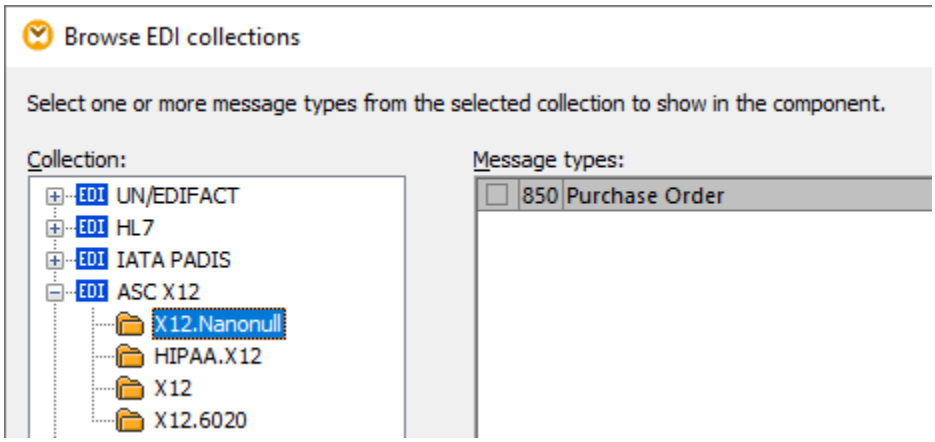
EDI.Collection ファイルを構成する

1. **EDI.Collection** ファイルを（例えば、XMLSpy などの）XML エディターで開きます
2. "Purchase Order" メッセージ以外のすべての "Message" 要素を削除します。<Messages> タグが保たれていることを確認してください。

```
<?xml version="1.0" encoding="UTF-8"?>
<Messages Version="3">
   <Meta>
      <Release>6020</Release>
      <Agency>X12</Agency>
   </Meta>
   <Root File="Envelope.Config"/>
   <Message Type="850" File="850.Config" Description="Purchase Order"/>
</Messages>
```

3. ファイルを保存します。

[挿入 | EDI] メニューコマンドを実行すると、コレクションはメッセージ型のみを表示します: "850 – Purchase Order"。



グローバルなカスタマイズとインラインなカスタマイズ

このサンプルの目的は個人のタイトル（例えば"Mrs" など）を保管するカスタムX1000 データ要素を含むようにN2 "Additional Name Information" セグメントを再定義することです。カスタム化の前はN2 は1つのデータ要素 "F93 Name" により構成されています。カスタム化を達成するには複数の方法があります:

- [グローバル]，**X12.Segment**ファイルをカスタム化して行います。[グローバル]はN2 を使用するすべてのトランザクション内でのすべてのセグメントが新規の要素を含む参照することを意味します。
- [インライン]，**850.Config** ファイルをカスタム化して行います。ローカルは現在のトランザクション内で カスタム化されたセグメント（N2）のみが新規に要素を含むことを意味します。

グローバルなカスタム化

新規のX1000 フィールド にグローバルにアクセスを作成する場合、**X12.Segment** ファイルのみに変更が加えられます。N2 を使用するすべてのトランザクション内のすべてのセグメントは新しい要素を含むまたは参照します。

N2 セグメントをグローバルに再定義する方法:

1. **X12.Segment** ファイルをXML エディター内で開き、「**構成 | 要素 | セグメント名="N2"**」にナビゲートします。

```
<Segment name="N2" info="Additional Name Information">
   <Data ref="F93" mergedEntries="2"/>
</Segment>
```

2. 次のラインをF93 の下に挿入し、ファイルを保存します。

```
<Data name="X1000" type="string" maxLength="35" minOccurs="0" info="New
Element"/>
```

セグメントの定義は以下のようになります::

```
<Segment name="N2" info="Additional Name Information">
   <Data ref="F93" mergedEntries="2"/>
   <Data name="X1000" type="string" maxLength="35" minOccurs="0" info="New
Element"/>
</Segment>
```

メモ

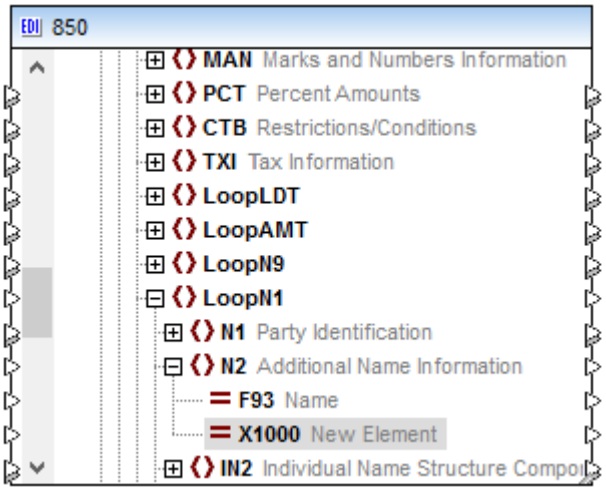
- ref 属性を使用するセグメントの他のフィールドに対して、新規のX1000 フィールドがname 属性を使用して定義されています。The F93 フィールドは**X12.Segment** ファイルの最初で定義されており、ここでのみ参照されています。
- F93 のmergedEntries 属性は同じ名前を持つこの要素の2 つの連続した発生を許可することを指定しています。これはASC X12 仕様で定義されている通りにこのフィールドの定義を反映しています。この理由のためここでカスタム化されているEDI ファイル内ではN2 セグメントは以下のようになります:

```
N2+Michelle Butler++Mrs'
```

上のラインで"Michelle Butler" と"Mrs" の間に2 つの連続した"+" 区切り文字が存在することに気が付きます。これはN2 セグメントがカスタム"X1000" データ要素("Mrs") が後に続く、(空である) 2番目の"F93" 要素が後に続く、"+" 区切り文字が後に続く("Michelle Butler") "F93" データ要素により構成されているためです。mergedEntries 属性に関する詳細は複数の連続したデータ要素の処理方法を参照してください。

セグメント N2 を使用する全てのトランザクションで新規のX1000 フィールドを使用することができます。MapForce 内の新しいフィールドを以下のようにプレビューすることができます:

1. 「挿入」メニューから「EDI」をクリックします。または、「EDI の挿入」 ツールバーボタンをクリックします。
2. "EDIFACT.Nanonull" コレクションをクリックして"Purchase Order" メッセージを選択します。
3. ソースEDI ファイルを選択するように問われると、「スキップ」をクリックします。850 コンポーネントがマッピングウィンドウ内で表示されています。
4. コンポーネントヘッダーをクリックし、「**Ctrl + F**」を押して、"X1000" を検索テキストとして入力します。「次を検索」をクリックし、(**Envelope > Interchange > Group > Message_850 > Loop_N1 > N2** の下に存在する) 新規のX1000 要素の次の発生に移動します。



ローカルでのカスタム化

新規のX1000 フィールドをローカルでアクセスするには**850.Config** ファイルのみが編集される必要があります。この場合、"850 - Purchase Order」トランザクション内のセグメント N2のみが新規のX1000 フィールドを含むまたはフィールドを参照します。

N2 セグメントをローカルで再定義する

1. **850.Config** ファイルをテキストまたはXML エディター内で開き、Group name="LoopN1" にナビゲートします(または**LoopN1**を検索します)。

```
<Group name="LoopN1" maxOccurs="10" minOccurs="0">
   <Segment ref="N1"/>
   <Segment ref="N2" minOccurs="0" maxOccurs="2"/>

   ......

</Group>
```

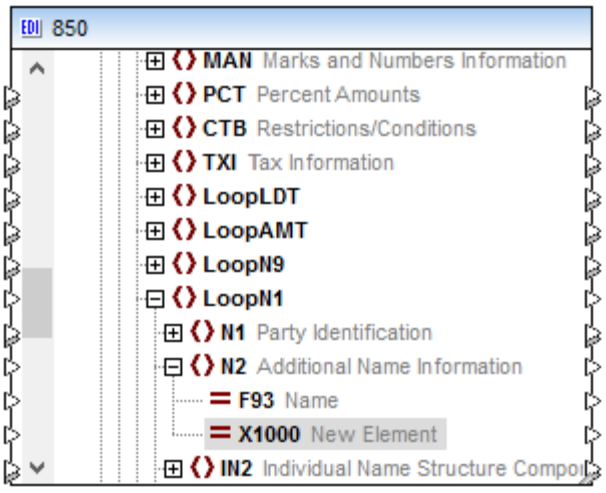
2. ライン `<Segment ref="N2" minOccurs="0" maxOccurs="2"/>` を以下のラインと置き換えます:

```
<Segment name="N2" info="Additional Name Information">
   <Data ref="F93" mergedEntries="2"/>
   <Data name="X1000" type="string" maxLength="35" minOccurs="0" info="New
Element"/>
</Segment>
```

`Group` 定義は以下のようになります:

```
<Group name="LoopN1" maxOccurs="10" minOccurs="0">
   <Segment ref="N1"/>
   <Segment name="N2" info="Additional Name Information">
      <Data ref="F93" mergedEntries="2"/>
      <Data name="X1000" type="string" maxLength="35" minOccurs="0" info="New
Element"/>
   </Segment>

   ......

</Group>
```

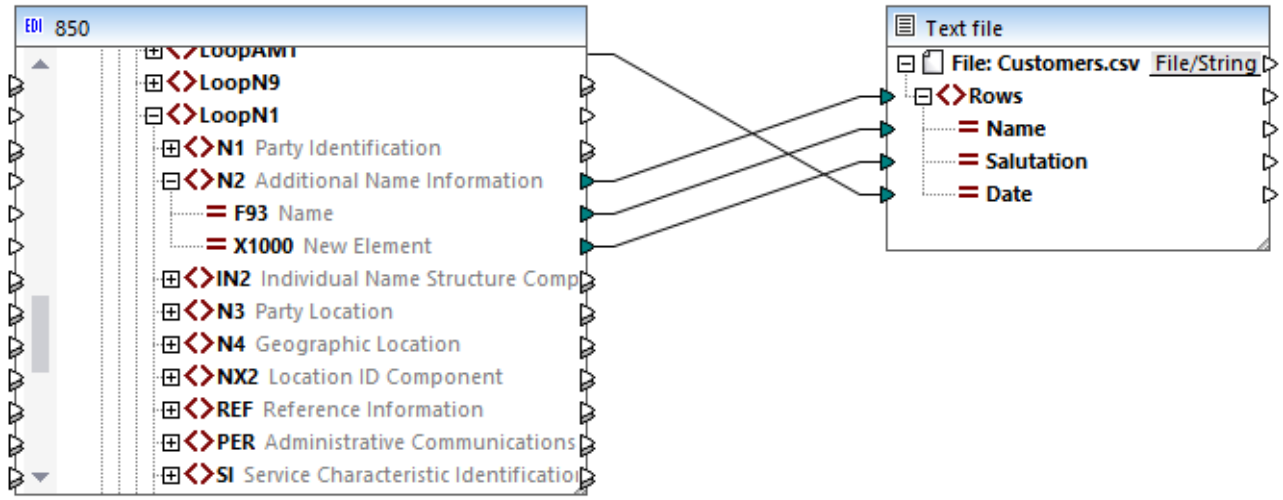
グローバルなカスタム化のために説明されている同じ命令を使用して MapForce 内の新規のフィールドをプレビューすることができるようになります。



(C) 2015-2021 Altova GmbH

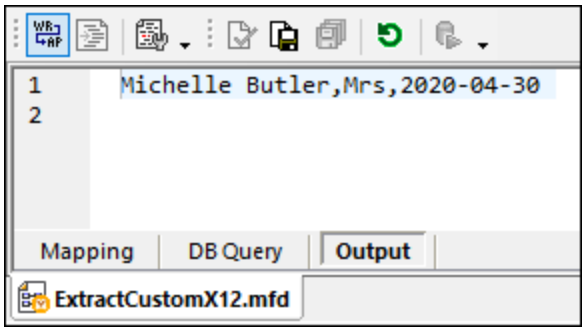
マッピング内でカスタマイズされたメッセージを使用する

上記の手順に従った場合、カスタム X12 構成 をマッピング内で使用することができます。上記のカスタム EDIFACT 構成を使用するマッピングの例は、<マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥Tutorial¥ フォルダー内の **ExtractCustomX12.mfd** です。サンプルは **ORDERS-Custom.X12** ファイルを CSV ファイルにマップします。上記のステップに追加されたカスタム `X1000` フィールドがターゲットファイル内で `Salutation` アイテムにマップされます。



ExtractCustomX12.mfd

マッピングの結果をプレビューするには、「**出力**」タブをクリックします。期待されるとおり `Salutation` ターゲットアイテムが X12 ファイルのカスタム X1000 データ要素から抽出された値 [Mrs] と共に作成されます。



6.5.18 EDI 検証のカスタマイズ

このセクションは、EDI メッセージの検証を MapForce 内でカスタム化して、以下の目的を達成する方法について説明します:

- 検証ルールが標準により完全に実施することのできないフィールドの検証を弛緩し、外部の標準を利用して拡張する場合もあります。

  例えば、定義済みのコードの値に追加して、ISO 外部標準、または国連欧州経済委員会(UN/ECE) の勧告により指定されたコードの値を受け入れる EDI データ要素があるとします。この場合、このような EDI フィールドを処理する場合、

MapForce を構成して、外部コードが発生すると（検証エラーではなく）警告を返し、マッピング処理を再開することができます。詳細に関しては、次を参照してください：フィールドの検証の弛緩。

- カスタム構成ファイルを MapForce に適用し、上記にようなフィールドを検証します。

  例えば、UN/EDIFACT D.16A 標準のデータ要素 4405 を検証するには、国連欧州経済委員会(UN/ECE) 勧告第 24 号(「貿易および運送のステータスコード」により定義された検証ルールを含む、カスタム構成ファイルを追加する必要があるかもしれません。国連欧州経済委員会(UN/ECE) 勧告第 24 号内で定義されたコード値は、EDIFACT 標準に対して外部となります(このため、MapForce により与えられる EDIFACT 構成ファイルのデフォルトのセットの一部ではありません)。詳細に関しては、次を参照してください：外部ソースからコードの値を適用する。

- フィールドが使用されている箇所、または特定のメッセージのためにローカルで使用されている全てのメッセージのためにグローバルに検証を行います。

  グローバル、またはローカルに検証を構成することは可能です。グローバルな検証は、全てのメッセージ内のフィールドでの全ての発生に影響を与えます。ローカル(または「インライン」) 検証は、特定のメッセージのみに適用することができます。詳細に関しては、次を参照してください：メッセージレベル(インライン)でコードの値を定義する。

- フィールドレベルだけではなく、文字レベルでも検証が必要なフィールドのために高度な検証ルールを定義します。

  UN/EDIFACT D.16A の VERMAS (大容量のメッセージの検証) などの特定のメッセージは、(データ要素 8155 などの) 高度な検証ルールを必要とするフィールドを含んでいます。このフィールドは、ISO 6346 内で指定されているコード値を受け入れる可能性があります。ISO 6346 などのフィールドを検証するために、構成ファイルを使用して、フィールド内で特定の文字が特定の場所で発生するかをチェックするように、MapForce をカスタム化することができます。詳細に関しては、次を参照してください：文字の構成レベルの検証。(ISO 6346 は、ここで例として記述されており、MapForce 構成ファイルは既に ISO 6346 のための検証ルールを既に含んでいます)。

6.5.18.1 フィールドの検証の弛緩

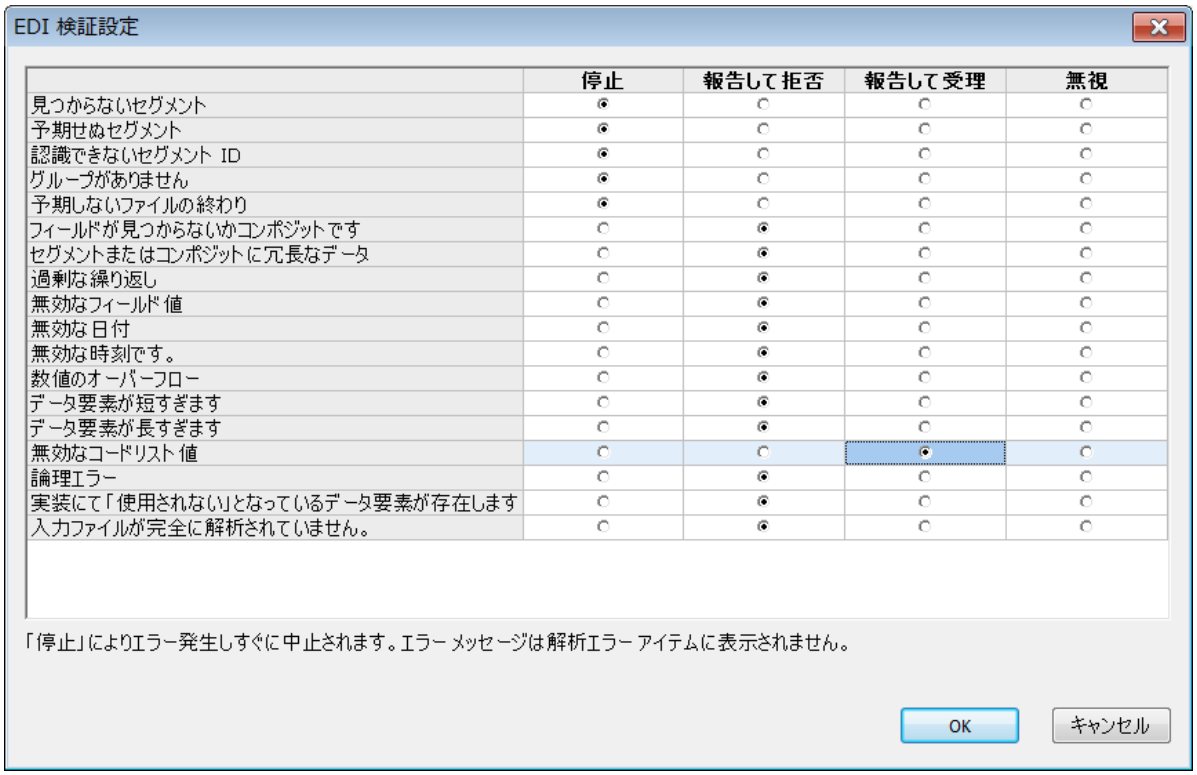
標準がフィールド内のコードの値を完全に適用しない場合、外部の標準または勧告により定義されたコードを許可する場合があります。EDIFACT D16A を例としてみましょう。フィールドの一部は、外部 ISO 標準内、またはUN/ECE 勧告で定義されたコードの値を許可します。例えば、データ要素 4405 は、146 コードの値を定義しますが、346 コードの値を指定する外部 UN/ECE 勧告第 24 号の使用も許可しています。

MapForce 内では、フィールドのためのコードの値は(以降、「.codelist」ファイルとして称される).codelist 拡張子を持つ構成ファイルに対して検証されます。MapForce が EDI ファイルを処理する際、フィールドがその種類の EDI ための.codelist ファイル内で指定されているコードの値を持たない場合、通常の場合検証エラーが表示されます。(これは、無効なフィールドの値を無視するために、デフォルトの検証オプションが変更されていないと想定されています。EDI コンポーネント検証を参照してください)。**C:\Program Files\Altova\MapForce2021\MapForceEDI** ディレクトリ内で各 EDI 種類のための.codelist ファイルを使用することができます。例えば、**C:\Program Files\Altova\MapForce2021\MapForceEDI\EDIFACT** 内で使用することのできる、**UNCL.codelist** ファイルは、UN/EDIFACT フィールド内のフィールドのための使用できるコードの値です。

デフォルトの.codelist ファイルは、対応する EDI 標準に従い、フィールドのために使用できるすべてのコードの値を含みます。コードの値が標準により明確に定義されておらず、外部の標準により強制されている場合、検証は不完全とみなされます。具体的には、EDI 標準が(UN/EDIFACT D16A のデータ要素 4405 などの) 特定のフィールドのためにコードの値のリストを強制しない場合、マッピングの実行中のフィールドの処理は、通常検証エラーを表示します。具体的には、MapForce (または、MapForce Server、や C#、C++、または Java プログラムなどの EDI マッピングを作動する実行エンジン) は、次のフォーマットでエラーを報告します: {Value} はフィールド {Field} の法的な値ではありません。このエラーは、.codelist ファイルで定義されている既存のコードリストの値のリスト内でフィールドの値が見つからなかったことを意味します。

マッピングの処理が上記のエラーなどにより中断されることを避けるためには、次を行います:

1. EDI 検証設定から検証の振る舞いを変更します（例えば、EDI 検証設定 ダイアログボックス上の「無効な コード リスト値」の横の「報告して受け入れる」チェックボックスを選択します。（EDI 検証設定を構成するには、EDI コンポーネント設定 ダイアログボックスの「検証」 ボタンをクリックしてください。）



2. 構成ファイから固定の検証を弛緩する方法は以下に表示されています。

MapForce は、フィールドに対して「未完成」検証ルールが存在するとマークするメカニズムを適用します。これにより、MapForce（または、MapForce Server、 やC#、C++、またはJava プログラムなどのEDI マッピングを作動する実行エンジン）は、フィールドが処理される際、エラーメッセージを作成しません。その代わりに、警告が表示され、マッピングの処理が再開されます。すなわち、「未完成な」 検証は、を意味します。外部で適宜されているコードの値ではなく、MapForce デフォルトの.codelist ファイル内で定義されているコードの値のみが、検証のために考慮されます。

複雑性を追加することなく、柔軟性を追加するには、 MapForce により与えられているデフォルトの.codelist 構成ファイル特定のフィールドで不完全な検証が存在すると扱うことです。

デフォルトでは、次のフィールド値が外部仕様内で定義されているコードの値を含むと、 MapForce は、（エラーではなく）警告を作成します:

- UN/EDIFACT D.16A のデータ要素 4053
- UN/EDIFACT D.16A のデータ要素 4277
- UN/EDIFACT D.16A のデータ要素 4405
- UN/EDIFACT D.16A のデータ要素 8179

警告は、このフィールドの検証が不完全であることを通知するために作成されます。不完全な検証は、正確では内データがマッピングにより生成されるために発生する場合があります。これを回避するには、カスタムの.codelist ファイルを MapForce に対して使用し、フィールド全体の検証をおこなうことが奨励されます。カスタムの.codelist ファイルは、を外部のコードの値を MapForce が理解できるフォーマットで定義し、フィールド全体の検証を行います。詳細に関しては、外部ソースからコードの値を適用するを参照してください。

メモ データ要素 8155 は、検証のために、外部 標準 (ISO 6346 標準)に依存しています、このフィールドのために、MapForce は、個別の.codelist ファイルを使用して、全体の検証行うこともできます。外部ソースからコードの値を適用するを参照してください。

必要であれば、特定のフィールドに対して、不完全な検証ルールが適応されるようにMapForce をカスタム化することができます。MapForce に、フィールドに不完全な検証ルーツが存在すると命令するには、以下を行います:

1. 検証ルールへの変更を保管するカスタムEDI コレクションを作成します。MapForceによりデフォルトでインストールされるEDI コレクションへの変更は奨励されません。このステップに関する詳細はカスタムEDI コレクションの作成を参照してください。
2. .codelist ファイルを カスタムEDI コレクション内で検索し、XMLSpy.などのXML エディターで開きます。
3. 必要とされるデータ要素に対応する <Field> 要素を検索し、(下でハイライトされている) 属性 Incomplete="1" を追加します。例:

```
<Field Name="F8179" Length="8" Type="Alphanumeric" Incomplete="1">
   <Title>Transport means description code</Title>
   <Description>Code specifying the means of transport.</Description>
   <Note>1 Use UN/ECE recommendation 28.
</Note>
   <Values>
      <Value Code="1">
         <Title>Box tricycle</Title>
         <Description>Tricycle with a basket for delivering goods.</Description>
      </Value>
   </Values>
</Field>
```

「未完成」とマークされるフィールドに対して処理の必要条件が警告ではなくエラーを表示することを必要とする場合、属性 Incomplete を「1」ではなく、値「0」に変更してください。この方法で、フィールド検証が厳密に行われ、フィールドの値が、.codelist ファイル内で定義されている値のセットの一部ではない場合、(警告ではなく) エラーが表示されます。

6.5.18.2 外部ソースからコードの値を適用する

MapForce により、デフォルトで提供されたEDI 構成ファイル内では、データ要素の一部は、フィールドの検証の弛緩で説明されているように不完全な検証が存在するように構成されています。これはこれらのデータ要素の可能なコード値は、EDI 標準自身の外部である仕様に依存するからです。

不完全な (または、弛緩されている) フィールドの検証が受け入れられない、または希望されない場合、MapForce を構成してフィールド全体を検証することができます。このアプローチは、MapForce 内で使用できるデフォルトのものに加え、カスタムの.codelist カスタムに対しても使用することができます。.codelist ファイルの役割は、デフォルトでEDI 標準内で定義されていないためにMapForce 構成ファイルに存在しないコードに値を与えます。これは、外部の標準、または勧告内で定義されているコードの値に通常適用されます。

カスタムの.codelist ファイルは、検証される使用できるコードの値のために エントリを必要とします (場合によっては数百個のエントリになる場合があります)。UN/EDIFACT D16A のデータ要素 4405 はつの例です。UN/EDIFACT 標準で定義されているコードの値に追加して、このフィールドは、UN/ECE 勧告 24 (「貿易および運送のステータスコード」 内で指定されているコードの値も許可します。346 個のコードを持つ次を参照してください:
https://www.unece.org/fileadmin/DAM/cefact/recommendations/rec24/rec24_ecetrd258e.pdf) 。勧告内の346 個のそれぞれのコードのためのエントリを含む外部のコードの値すべてに対応するため、カスタムの.codelist ファイルを作成する必要があります。ですから、特定のビジネスのケースで不完全な検証の使用が許可されている場合、不完全な検証は簡単な解決方法です (フィールド検証の弛緩を参照してください)。不完全な検証は、マッピングの出力内で無効なデータが生成されることを許可する可能性があるため、代わりにフィールド全体の検証が奨励されます。

(C) 2015-2021 Altova GmbH

フィールド全体を検証するには、外部の標準内で定義されているコード値で定義されているコードの値を使用します。次の手順を追ってください:

1. デフォルトの検証ルールへの変更を保管するカスタムEDI コレクションを作成します。このステップに関する詳細は[カスタムEDI コレクションの作成]を参照してください。

   > MapForce により提供される既存の.codelist ファイル(この例の場合は**UNCL.codelist**) を編集することにより外部コードリストを定義することは可能ですが、これは奨励されません。カスタムファイル内の外部コードリストを常定義し、デフォルトのMapForce 構成ファイルへの変更を回避することはグッドプラクティスです。

2. カスタムEDI コレクションのディレクトリ内で、テンプレートとして使用される.codelist ファイルを検索します。例えば、カスタム.codelist ファイルがUN/EDIFACT メッセージを検証するために追加される場合テンプレートとして**UNCL.Codelist** ファイルと使用します。
3. テンプレート .codelist ファイルの名前を変更して短く説明するファイル名を持つ新規の.codelist ファイルを取得します。このファイルは他のMapForce 構成ファイルから参照されます。このサンプルのスコープのためにカスタム.codelist ファイルは**UNECE24.codelist** と呼ばれます。
4. XMLSpy などのXML エディターを使用して <Codelist> と</Codelist> タグ間に含まれる新しいファイルを削除します。ファイルは以下のようになります:

```
<?xml version="1.0" encoding="UTF-8"?>
<Config xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" Version="5"
xsi:noNamespaceSchemaLocation="../EDIConfig5.xsd">
   <Codelist>

...

   </Codelist>
</Config>
```

5. <Codelist> の下の新規の<Field> 構成は下に示されています。このサンプルでは、UN/EDIFACT. のデータ要素4405 のために外部のコードの値を追加します。これ以外の場合、フィールドの属性を特定のケースで必要とされるとおり置き換えます。MapForce により提供されるデフォルトの.codelist ファイル内の各フィールドの名前、長さ、型を確認することができることに注意してください。(このサンプルでは、**UNCL.Codelist** です)。

```
<?xml version="1.0" encoding="UTF-8"?>
<Config xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" Version="5"
xsi:noNamespaceSchemaLocation="../EDIConfig5.xsd">
   <Codelist>
      <Field Name="F4405" Length="3" Type="Alphanumeric">
         <Title>Trade and transport status codes</Title>
         <Description>Code specifying a status.</Description>
         <Values>

...

         <Values>
      </Field>
   </Codelist>
</Config>
```

6. <Values>の下に、各外部のコードのために新規の<Value> を以下のように追加します。このサンプルでは、UN/ECE 勧告24「貿易および運送のステータスコード」内で定義されている最初の2 つのコードを追加します(次を参照してください: https://www.unece.org/fileadmin/DAM/cefact/recommendations/rec24/rec24_ecetrd258e.pdf)。

```
<?xml version="1.0" encoding="UTF-8"?>
<Config xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" Version="5"
xsi:noNamespaceSchemaLocation="../EDIConfig5.xsd">
   <Codelist>
      <Field Name="F4405" Length="3" Type="Alphanumeric">
         <Title>Trade and transport status codes</Title>
         <Description>Code specifying a status.</Description>
         <Values>
            <Value Code="1">
               <Title>Arrival, completed</Title>
               <Description>The goods/consignment/equipment/means of transport
has arrived.</Description>
            </Value>
            <Value Code="2">
               <Title>Loading, authorized</Title>
               <Description>Authorization to load has been given.</Description>
            </Value>
         <Values>
      </Field>
   </Codelist>
</Config>
```

この段階では、カスタムの.codelist ファイルの準備はできていますが、MapForce にバウンドされてはいません。このサンプルのスコープのために、2 つの外部コード（「1」と「2」）のみが追加されていますが、実社会のシナリオでは、カスタムの.codelist ファイルは、外部の標準により指定される全てのコードの値を含むよう期待されます。L.codelist ファイルを下記の通り変更します。

最後に、カスタムの.codelist ファイルがMapForce により認識される前に、以下を行う必要があります:

1. データ要素 4405 には不完全な検証がもう存在しないため、MapForce はフルの検証を適用するように命令されなければなりません。具体的には、`Incomplete="1"` 属性は、**UNCL.codelist** ファイル内で、このフィールドのために変更される必要があります（属性が削除されるか、または `Incomplete="0"` に設定されなければなりません）。このため、**UNCL.codelist** ファイルを下記の通り変更します。

```
...

<Field Name="F4405" Length="3" Type="Alphanumeric" Incomplete="0">
         <Title>Status description code</Title>
         <Description>Code specifying a status.</Description>
         <Note>1 For transport status, use UN/ECE Recommendation 24.</Note>

...
```

2. **UNCL.codelist** ファイルを変更して、新規に作成された**UNECE24.codelist**ファイルにリファレンスを追加する方法は、は以下のとおりです:

```
<?xml version="1.0" encoding="UTF-8"?>
<Config xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" Version="5"
xsi:noNamespaceSchemaLocation="../EDIConfig5.xsd">
   <Include href="UNECE24.codelist"/>
   <Codelist>

....
```


(C) 2015-2021 Altova GmbH

カスタムファイル内で定義されたコードの値からのフィールド 4405 を検証するための、MapForce 構成ファイルの適用が完了しました。このアプローチは、外部の標準をベースにしたフィールド全体の検証を達成するために、他のEDI 種類のために使用することもできます。

上記の命令は、フィールド 4405 を使用する可能性のあるセグメントのために、フィールド検証をグローバルにします。一部のケースでは、特定のメッセージのみにカスタムのコードリストを適用する場合があります（すなわち、構成は、全てのメッセージのためにグローバルに適用されてはならず、特定のメッセージのためのみにインライン、またはローカルに適用されます）。「インライン」アプローチの使用に関しては、メッセージレベル（インライン）でコードの値を定義するを参照してください。

6.5.18.3 メッセージレベル（インライン）でコードの値を定義する

前のトピックの 外部ソースからコードの値を適用する では、（特定のフィールドを使用しているメッセージに適用する）カスタムのコードリストをグローバルに使用してフィールド検証を構成する方法について説明がされました。特定のメッセージのみに検証を適用する必要がある場合は、（特定のメッセージのために ローカルに適用し）カスタムのコードリストをインラインに追加することが可能です。以下の手順に従ってください：

1. これを既に行っていない場合 EDI コレクションを作成します。カスタム EDI コレクションの作成を参照して下さい。
2. カスタム EDI コレクションのディレクトリ内で興味のあるメッセージに対応した **{Message}.config** ファイルを検索し、XML エディター内で開きます。
3. 必要とされる の下で <Data> 要素を検索し <Values>, の子の値としてカスタムコードを追加します。例:

```
<Group name="Message_270-B1" info="Health Care Eligibility Benefit Inquiry"
maxOccurs="unbounded">
      <Segment name="ST" info="Transaction Set Header">
        <Condition path="F1705" value="005010X279A1" />
        <Data ref="F143" info="Transaction Set Identifier Code">
          <Values>
            <Value Code="270" />

[>>>> custom values can be added here...]

          </Values>
        </Data>
        <Data ref="F329" info="Transaction Set Control Number" />
        <Data ref="F1705" info="Implementation Convention Reference" />
      </Segment>
```

以下の点に注意してください：

- フィールドが（上のコードリスティング内で説明されているような）ローカルで定義された、および、（外部ソースからコードの値を適用するで表示されている .codelist ファイルのような）グローバルで定義されたコードの値 の両方を有する場合、ローカルのコードの値がグローバルなコードの値としても定義されていることを確認してください。これ以外の場合、検証エラー*** が挙げられます: {Value} はフィールドの {Field} の法的な値ではありません。
- インラインの（local）検証は、フィールドレベル（すなわち、そのフィールドでの可能な値がリストとして表示されることができるレベル）で検証が必要な場合使用することができます。インラインの検証は、文字レベルで複雑な検証を必要とするフィールドのためには使用することができません。（UN/EDIFACT 標準のデータ要素 8155 に適用することのできる ISO 6346 内で定義されているような）フィールドが複雑な検証ルールを文字レベルで必要とする場合、外部ソースからコードの値を適用するで示されるとおり、コードの値をグローバルに追加し、検証を文字レベルの検証の構成で示されるとおり構成します。文字レベルの検証の構成で示される通り検証を構成することができます

*** 実際の検証の結果はEDI 検証設定 ダイアログボックス内のルール「無効なコード リストの値」のために構成された設定を考慮することに注意してください（このダイアログボックスを開くには、 EDI コンポーネント設定 ダイアログボックスの「検証」ボタンをクリックしてください）。

6.5.18.4 文字の構成-レベルの検証

多くの場合、フィールドの検証は、マッピングランタイムでのフィールドの値を MapForce 構成ファイルにより定義された値のリストと比較することを意味します。例えば、フィールドの値が構成ファイル内で定義されている使用できる値の1 つに一致する場合、有効とみなされます。

しかしながら、特定のフィールドは、更に高度な検証ルールを必要とする場合があります。これは、検証のロジックが使用できる値の簡単なリストとして表示することができないことを意味します。複雑な検証を必要とするフィールドのサンプルは、UN/EDIFACT D16A の VERMAS (Verification of Mass) メッセージ内で使用されている、データ要素です。このフィールドは、UN/EDIFACT 標準内で定義されているコードの値だけではなく、ISO 6346 標準により指定されている外部コードも受け入れます。ISO 6346 標準は、文字レベルで検証ルールを定義しています。具体的には以下が挙げられます:

- 最初の文字は次の1 つであることができます: 1、2、3、4、A、B、C、D、E、F、G、H、K、L、M、N、P
- 2 番目の文字は次の1 つであることができます: 0、2、4、5、6、8、9、C、D、E、F、L、M、N、P
- 3 番目、4 番目の文字は次の組み合わせの1 つであることができます: G0、G1、G2、G3、など。(全ての値を表示することはできません)。

MapForce より高度な検証 ロジックを適用することもできます。この場合、MapForce は、データ要素 8155 を、UN/EDIFACT コードの値だけではなく、ISO 6346 標準に適用することができるルールにも従い、検証します。

MapForce 構成パースペクティブからは、上記の3 つのルールは、3 つの個別の値のリストとして示されています。最初のリストおよび2 番目のリストは、1 文字で、3 番目のリストは、2 文字です。3 つ全てのリストは、有限で値は既知です。この情報は、MapForce 構成を使用してデータ要素の値を検証することを可能にします。

文字レベル検証の構成を確認するには、**ISO6346.Codelist** ファイルを **C:\Program Files\Altova\MapForce2021\MapForceEDI\EDIFACT** ディレクトリから開きます。このファイルはデフォルトで与えられており、MapForce は、上記のISO 6346 ルールに従い、データ要素の検証を行うように構成されています。

```
<?xml version="1.0" encoding="UTF-8"?>
<Config xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" Version="5"
xsi:noNamespaceSchemaLocation="../EDIConfig5.xsd">
   <Codelist>
      <Field Name="F8155" Type="Alphanumeric" ListName="ISO6346" Length="1" Offset="1">
         <Title>ISO 6346</Title>
         <Description>Code specifying seagoing container type.</Description>
         <Values>
            <Value Code="1"/>

            ..............

            <Value Code="N"/>
         </Values>
      </Field>
      <Field Name="F8155" Type="Alphanumeric" ListName="ISO6346" Length="1" Offset="2">
         <Title>ISO 6346</Title>
         <Description>Code specifying seagoing container type.</Description>
         <Values>
            <Value Code="0"/>

            ..............

            <Value Code="F"/>
         </Values>
      </Field>
      <Field Name="F8155" Type="Alphanumeric" ListName="ISO6346" Length="2" Offset="3">
         <Title>ISO 6346</Title>
         <Description>Code specifying seagoing container type.</Description>
         <Values>
            <Value Code="G0"/>

            ..............

            <Value Code="A0"/>
         </Values>
      </Field>
   </Codelist>
</Config>
```

ISO6346.Codelist 構成 file

メモ 上記のサンプルのコードリスティングの値の一部は、簡素化のために削除されました。

上のコード リスティング内で、 `<Field>` 要素は、3 回定義されています。これにより、フィールドのコンテンツは文字レベルで検証することが可能になります。具体的には、`<Field>` が発生する都度、上記のルールのうち 1 つを検証します。これを達成するには、各 `<Field>` には、上のコードリスティングでハイライトされているように複数の属性が存在します:

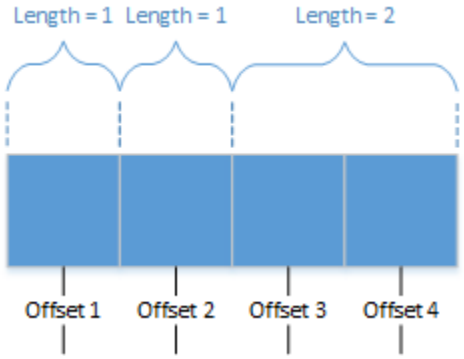
- `Name` - この属性は、検証される値のリストは同じフィールドに属することを示します。すなわち、これは3 つのリストを(この場合、「F8155」)同じフィールドにバインドします。
- `ListName` - この属性は、値のリストに名前を与えます。同じ名前を持つ値のリストは、1 つのコードリストとしてグループ化されます。上のサンプルでは、3 つ全てのリストは、同じ `ListName="ISO6346"` を持ちます。これは、文字レベルの検証に必要です。グループ内の各リストの `Length` と `Offset` 属性により指定されている実際の文字の検証に関しては、下で説明されています。
- `Length` – この属性 は、検証される文字の長さを指定します（文字の開始のポジションは、`Offset` 属性により定義されているフィールド内に存在します)。文字の長さは、それぞれの `<Field>` により異なります: 最初と2 番目の値のリストは、1 文字で、3 番目のリストは2 文字です。

- `Offset` – この属性は、MapForce（または、外部の実行エンジン）が検証されるデータフィールドから文字を読み込む開始の位置を指定します。最初の文字は、オフセット 1 を持ち、2 番目の文字はオフセット 2 を持つなどです。実際の文字数は、`Length` 属性により定義されます。

> `Length` 属性が定義されている場合、 `Offset` 属性も定義される必要があります。また、`Length` と`Offset` の値は0 より大きくなければなりません。

上のコード リスティングでは、以下の長さ-オフセット 組み合わせが定義されています:

- `Length="1", Offset="1"` – この組み合わせは、オフセット1から開始するフィールドの1 文字を検証します。
- `Length="1", Offset="2"` – この組み合わせは、オフセット2から開始するフィールドの1 文字を検証します。
- `Length="2", Offset="3"` – この組み合わせは、オフセット3から開始するフィールドの2 文字を検証します。



`<Values>`の下で定義されているコードの値と組み合わされている各長さ-オフセット 組み合わせは、検証の条件を文字レベルで示しています。フィールド検証が成功するためには、マッピングの作動時、すべての文字レベル条件が満たされている必要があります（これは全てが同じ `ListName` 属性値持つためです） すなわち、データ要素 8155 を含むメッセージを検証する場合、MapForce は次のロジックを適用します:

- フィールド文字は、長さ-オフセット 組み合わせの一致のために、構成ファイル（この場合、**ISO6346.Codelist**）内で定義されているコードの値のリストに対して検証されます。例えば、フィールドの最初の文字は、以下のリスト内の1 つであることができます: 1, 2, 3, 4, B, C, G, H, L, M, N.
- 検証を成功させるには、フィールドの値は、文字列での全ての条件を満たす必要があります（この場合、3 つの条件が満たされていなければなりません）。
- 条件の一つが満たされない場合、フィールドの値は有効ではなく、MapForce （または、MapForce Server、やC#、C++、またはJava プログラムなどの）EDI マッピングを実行する実行エンジン内で検証エラーが生成されます。
- フィールドは、使用できるコードの値の1 つ以上のリストを持つことができます。有効とみなされるためには、フィールドの値は、リストの1 つに存在する必要があります。

(C) 2015-2021 Altova GmbH

6.6 JSON

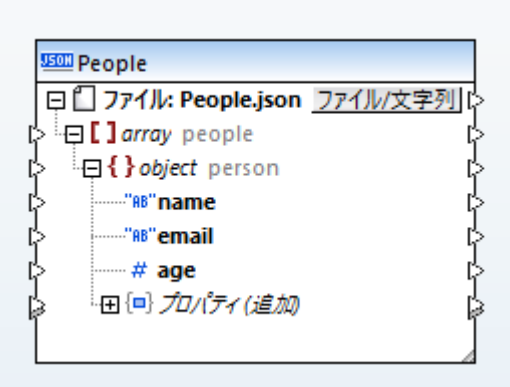
Altova Web サイト: JSON マッピング

JSON (JavaScript Object Notation) はデータを保管し転送するために人気のあるデータ書式です。MapForce Enterprise Edition では、MapForce によりサポートされる書式から JSON にデータをマップ、または逆方向にデータをマップすることができます。(例えば、データベースに書き込む場合など) of a コンポーネントの解析/シリアル化の助けを借りて、JSON データを文字列としてシリアル化、JSON 文字列を JSON 構造に解析することもできます。文字列の解析とシリアル化 を参照してください。

> JSON ファイルをマッピング内で使用するには、マッピング言語が以下のいずれかに設定されている必要があります: Built-in、C#、Java。詳細に関しては、 変換言語の選択 を参照してください。

MapForce によりサポートされる他のコンポーネント型同様、JSON データを読み取る、または書き込む前にスキーマファイルが必要になります。スキーマファイルが存在しない場合、, MapForce は自動的に生成することができます。JSON ファイルをマッピングコンポーネントとして追加する を参照してください。MapForce により生成されていない JSON スキーマファイルを提供する場合、MapForce は Draft 04、Draft 06、Draft 07 バージョン、または バージョン無しのものを期待します。

以下では JSON 構造が MapForce 内で表示されるサンプルを示しています:



MapForce 内の JSON コンポーネント

上記のとおり、MapForce 内の JSON コンポーネントは、通常 JSON インスタンスファイルの構造を持ちます。すなわち、JSON コンポーネント内のノードの構造は、JSON インスタンスファイルの構造を反映します。基本的な JSON 型は、MapForce 内で以下のテーブルのように表示されます。

型	MapForce 表示
配列	[]
ブール値	01
Null	∅
数値または整数	#
オブジェクト	{}
文字列	"AB"

特別なケースは、以下のとおりです。

ケース	説明
additionalProperties、*patternProperties*	ノード は、`additionalProperties` プロパティが`true` に設定されている、またはスキーマ内に存在しない、オブジェクトの下のJSON コンポーネントに表示されます。これにより、明示的にスキーマにリストされていないプロパティにマップ、またはプロパティからマップすることができます（例: JSON からCSV へマッピングも参照）。 このノードは、`patternProperties` プロパティのために表示されることもできます。
サブタイプ	JSON スキーマは、オブジェクトと配列（`anyOf`, `allOf`, `oneOf`）のためにサブタイプを定義することを許可します。MapForce は、JSON インスタンスファイル内で直接の表記を持たない特別な構造のノード（ ）を使用してこのようなサブタイプを表示します。 マッピングがJSON ファイルから読み込んでいる場合、このようなサブタイプノード は、現在の入力の値が、サブ型スキーマに従い有効な 場合のみ値を与えます。 マッピングがJSON ファイルに書き込む場合、正しいサブタイプが選択されていることを確認してください。複数のサブタイプへの入力は、オブジェクトプロパティの重複に繋がる可能性があり、このため、出力 JSON ファイルが無効になる可能性があります。
同じ場所での複数の型	JSON スキーマは、同じ場所で複数の型が派生します。このような場合、MapForce コンポーネントは、この場所で派生するすべての基本型のための個別の構造ノードを表示します。
型の名前	MapForce は、存在する場合、JSON スキーマ内の`title` と`description` プロパティの型を"型" と"注釈" フィールドに表示します。`title` が存在しない場合、MapForce は、`$ref` プロパティのURI の一部を型の名前として使用する場合があります。
複合型 アイテムの型を含む配列	JSON スキーマ内でアイテムの異なる種類を持つ配列の場合（例: 文字列と番号）、MapForce は、それぞれのアイテムの型のための"アイテム" ノードを表示します。JSON ファイルに書き込む場合、これにより、異なる種類のアイテムを含む配列を作成することができます。
タプルとして定義される配列	JSON スキーマ内のポジションにより割り当てられる型を持つアイテムが配列に存在する場合、MapForce は、ゼロをベースにしたアイテムのインデックス個別の構造ノード（例えば、**item[0]**、**item[1]**、など）としてを表示します。JSON ファイルに書き込む場合、これにより、配列内のそれぞれの個別のアイテムの型をゼロをベースにしたインデックスで指定することができます。

6.6.1 JSON ファイルをマッピングコンポーネントとして追加する

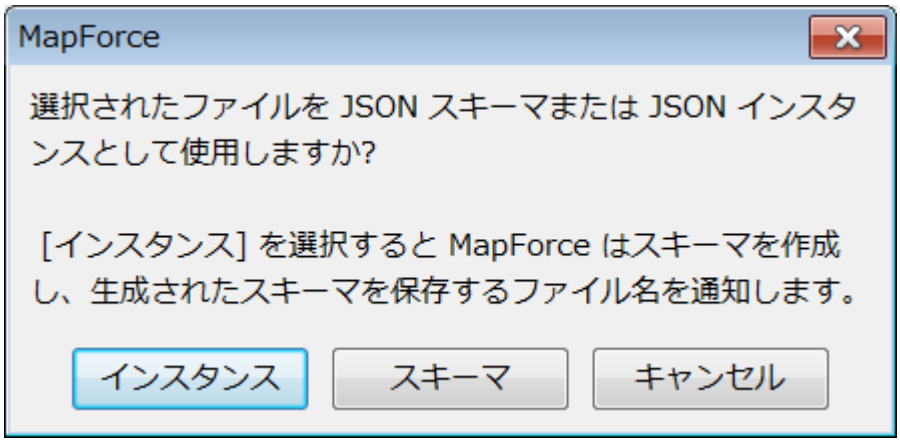
MapForce を使用して JSON インスタンスファイルを読み取る、または書き込む場合、JSON スキーマファイルを必要とします。JSON スキーマファイルを MapForce を使用して生成、またはカスタムファイルとして提供することができます。

このドキュメントが作成されている現時点ではJSON スキーマファイルに名前を当てる正式な規則はありません。このため、JSON スキーマファイルとインスタンスを区別するために、**schema.json** 拡張子を JSON スキーマファイルを使用し、（例えば、**Example.schema.json**）MapForce は拡張子を識別します。

JSON ファイルをマッピングコンポーネントとして追加する

1. 「**挿入**」メニューから「**JSON スキーマ/ファイル**」をクリックします。ダイアログボックスがJSON ファイルの型（スキーマまたはインスタンス）を選択するようにプロンプトする可能性があります 。このダイアログボックスは、MapForce がJSON スキーマとしてファイルを認識した場合のみ表示されます。

(C) 2015-2021 Altova GmbH

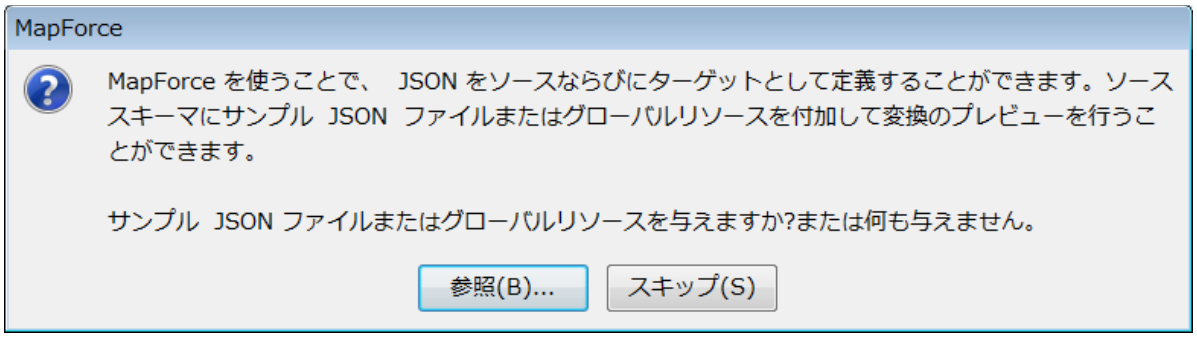


2. 必要に応じて、「**スキーマ**」または「**インスタンス**」を選択します。

JSON *インスタンス*ファイルのサイズはシステムで使用できるメモリを超えてはなりません。

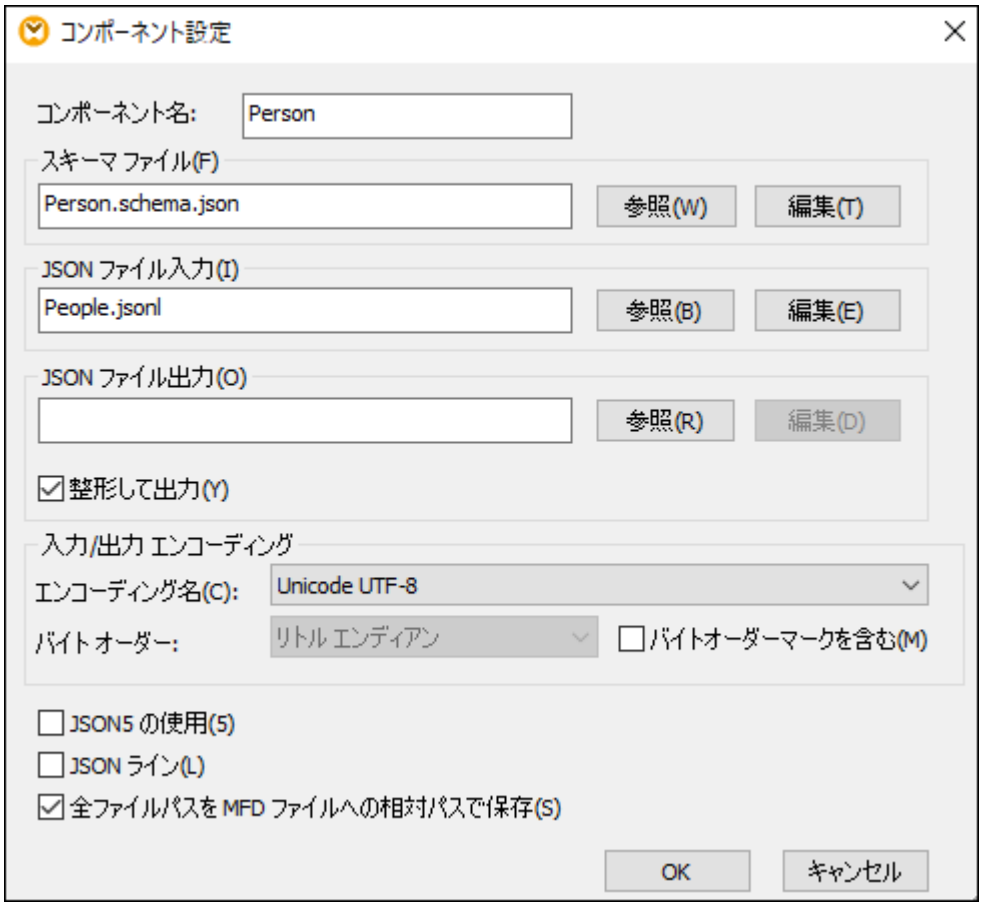
「**インスタンス**」を選択した場合、MapForce *インスタンス*ファイルをベースにしたスキーマを自動的に生成し、スキーマを保存する場所をプロンプトします。

「**スキーマ**」を選択した場合、MapForce はサンプル JSON ファイルまたは*グローバルリソース*を指定するようにプロンプトします。(*グローバルリソース*を参照)。変換をプレビューすることは必須です。これは変換をプレビューする場合必要とされ、JSON ファイルからデータを読み取る場合のみ意味を成します。マッピングがデータを JSON に書き込む場合、「**スキップ**」をクリックします。



6.6.2 JSON コンポーネント設定

JSON コンポーネントの設定を変更するには、JSON コンポーネントヘッダーを右クリックして、「**プロパティ**」をクリックします。(または、JSON コンポーネントヘッダーをダブルクリックします)。



JSON コンポーネント設定 ダイアログボックス

使用することのできる設定は、以下のとおりです.

コンポーネント名	コンポーネント名はコンポーネントを作成する際に自動的に生成されます。ですが、後に名前を変更することが可能です。 コンポーネント名はスペースを含むことができ、またフルストップ文字も含むことができます。コンポーネント名は、スラッシュ、バックスラッシュ、コロン、二重引用符、行頭と末尾スペースを含む場合があります。全般的に、コンポーネントの名前を変更する際、以下の影響を考慮してください: • マッピングを FlowForce Server にデプロイする場合、コンポーネント名は一意である必要があります。 • コマンドラインに入力できる文字のみを使用することが奨励されます。コマンドラインと Windows 内では、それぞれの文字で異なったエンコードを使用する場合があります。
スキーマファイル	スキーマファイルのファイル名とパスを指定します。

	ファイルのロケーションを変更するには、「参照」をクリックして、新しいファイルを選択します。 JSON エディター（例: XMLSpy）内でファイルを編集するには、「編集」をクリックします。
入力 JSON ファイル	マッピングが JSON ファイルからデータを読み取る場合のみこのテキストボックスは意味があります。入力 JSON インスタンスのファイル名とパスを指定します。 ファイルのロケーションを変更するには、「参照」をクリックして、新しいファイルを選択します。「MFD に相対的なすべてのファイルパスを保存する」チェックボックスが選択されている場合にも相対パスを入力することができます（以下を参照してください）。 JSON エディター（例: XMLSpy）内でファイルを編集するには、「編集」をクリックします。
出力 JSON ファイル	マッピングがデータを JSON ファイルに書き込む場合のみこのテキストボックスは意味があります。MapFrce Server によりマッピングが実行される場合、これはターゲット JSON インスタンスが作成される個所でファイル名とパスを指定します。 「出力」タブから出力を保存する際のデフォルトのロケーションです。 ファイルのロケーションを変更するには、「参照」をクリックして、新しいファイルを選択します。「MFD に相対的なすべてのファイルパスを保存する」チェックボックスが選択されている場合にも相対パスを入力することができます（以下を参照してください）。 JSON エディター（例: XMLSpy）内でファイルを編集するには、「編集」をクリックします。
整形出力	構造化された表示のため、マッピングが実行される際、JSON ドキュメントを再フォーマットします。各子ノードは、単一タブ文字ごとにそれぞれの親のオフセットです。
エンコード	出力インスタンスファイルの次の設定を指定することを許可します: • エンコード名 • バイトオーダー • バイトオーダーマーク（BOM）文字が含まれるか否か。 JSON ファイルは、UTF エンコードを持つことが期待されます。（ http://tools.ietf.org/html/rfc7159#section-8.1 を参照）。他のエンコードは、非標準と考えられます。
JSON5 の使用	コンポーネントが入力 JSON ファイルを読み取るデータが JSON5 構文を使用する場合このチェックボックスを選択します。 ターゲットコンポーネント内での、チェックボックスの選択は、引用符で囲まれていない、オブジェクトキーが有効な ECMAScript5 識別子である場合、.json5 ファイルを作成するように命令します。次を参照します: JSON5 へのサポート。
JSON ライン	JSON ライン書式内でターゲットファイルにデータを書き込むには、JSON ライン書式内のソースファイルからデータを読み取る場合、このチェックボックスを選択しま

	す。詳細に関しては、JSON ラインを参照してください。
MFD ファイルに相対的なすべてのファイルパスを保存する	このオプションが有効化されると、MapForce は、コンポーネント設定ダイアログボックスに表示された、MapForce Design (.mfd) ファイルの場所に相対したファイルパスを保存します。この設定 は、以下のファイルに影響を与えます: • JSON スキーマファイル • JSON 入力ファイル • JSON 出力ファイル コンポーネント上で相対パスを使用するも参照してください。

6.6.3 JSON5 へのサポート

JSON5 は、JSON ファイルを手動で作成するユーザーが使用しやすいように提案された JSON への拡張子です。重要な点は、JSON5 は JSON の公式な後継ではありません(詳細に関しては、次を参照してください: http://json5.org/)。

JSON5 ファイルは、.json5 拡張子を使用します。ですが、このドキュメントの作成時では、.json5 ファイルのための公式の MIME 型は存在しません。.json5 拡張子は、しかしながら、マッピングへそのようなファイルが追加されると、MapForce により認識されます。マッピングを FlowForce Server にデプロイすると、.json5 ファイルの MIME 型は `application/json` としてレポートされます(これは、.json5 ファイルのために MIME 型が存在するからです)(次を参照してください: FlowForce Server にマッピングをデプロイする)。

必要条件

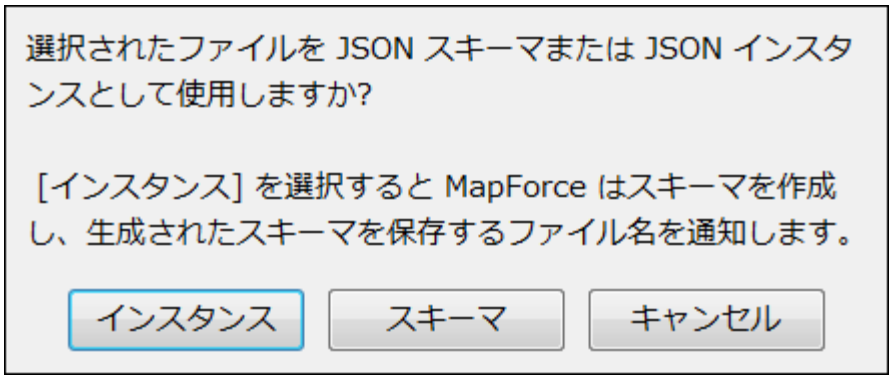
- JSON5 ファイルからのデータをマップするには、(JSON5 構文を使用しない) 有効な JSON スキーマを MapForce に提供する必要があります。下に示されるように、ファイルのために JSON スキーマが存在しない場合、MapForce はスキーマを自動的に生成します。MapForce により自動的に生成されるスキーマは(JSON5 ではなく) JSON スキーマです。XMLSpy を使用して、.json5 のために JSON スキーマを生成することは可能です。

制限

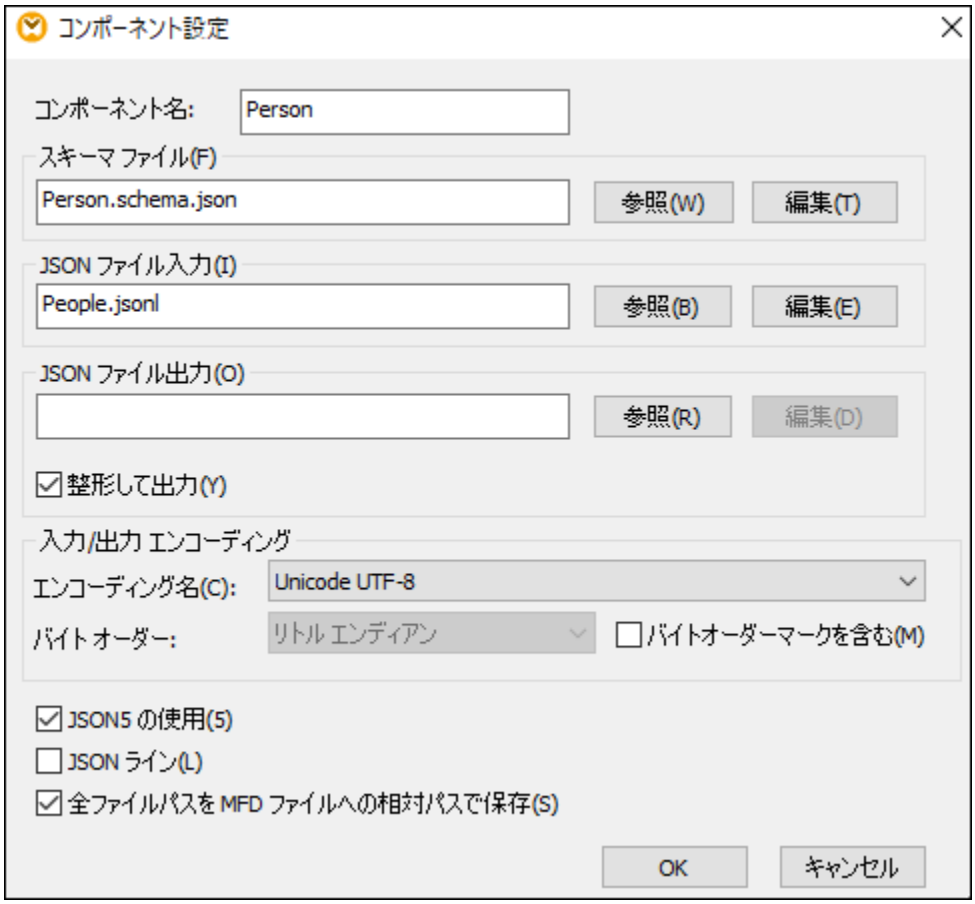
- JSON5 構文を使用する .json5 ファイルの生成は、サポートされていません。生成された .json5 ファイルは、しかしながら、JSON に準拠します。
- .json5 ファイルからのマッピングコメント、または json5 ファイルへのマッピングコメントはサポートされていません。
- MapForce は、厳格なデータ型と作業するため、`Infinity`、`-Infinity`、`NaN`、または `-NaN` などの値を修飾、またはマップすることはできません。

.json5 ファイルからデータをマップする方法

1. 「挿入」メニューから、**JSON** をクリックし、.json5 インスタンス ファイルを参照します(または、 ツールバーボタンをクリックします)。JSON スキーマファイルが存在し、.json5 インスタンス ファイルを後で提供する場合は、有効な(JSON5 ではなく) JSON スキーマを参照してください。これはプレーンな JSON ファイルが作動すると同様に作動します(次を参照してください: JSON ファイルをマッピングコンポーネントとして追加する)。



2. JSON コンポーネントがマッピングに追加された後、ヘッダーを右クリックし、**プロパティ** を選択し、「**JSON5 の使用**」チェックボックスをコンポーネント プロパティから選択します（次も参照してください: JSON コンポーネント 設定）。これにより、明示的に MapForce にインスタンスデータをそのように扱うように命令することができます。



MapForce を使用して、JSON5 ファイルを生成するには、**JSON5 の使用** チェックボックスを上に示されるように選択します。JSON コンポーネントを必要とされる入力データにマッピング上で接続します。**JSON5 の使用** チェックボックスを使用する際には、MapForce は、生成された JSON5 インスタンスが無効になる場合を除き、オブジェクト キーを引用符で囲みません。望ましいファイル拡張子（例えば「json5」）は、上に示されるダイアログボックスで**出力 JSON ファイル** フィールド内に適切な値を入力することにより、管理することができます。

6.6.4 JSON Lines

JSON ライン（http://jsonlines.org/）はJSON を拡張するテキスト書式です。ログファイルなど、新規のラインの区切りが必要な場合にJSON データを保管、または処理するために役に立つ可能性があります。Each line in a JSON ラインファイル内の各ラインがJSON 値を保存します。例えば

```
{"name":"Alethia Alonso","email":"alethia@example.org"}
{"name":"Klaus Mauer","email":"klaus@example.org"}
{"name":"Natsuo Shinohara","email":"natsuo@example.org"}
```

MapForce Enterprise Edition を使用する場合、JSON ライン書式内のデータを読み取ることができ、MapForce によりサポートされる他の書式に変換することができます。同様に、MapForce によりサポートされるデータを JSON ライン書式に変換することができます。

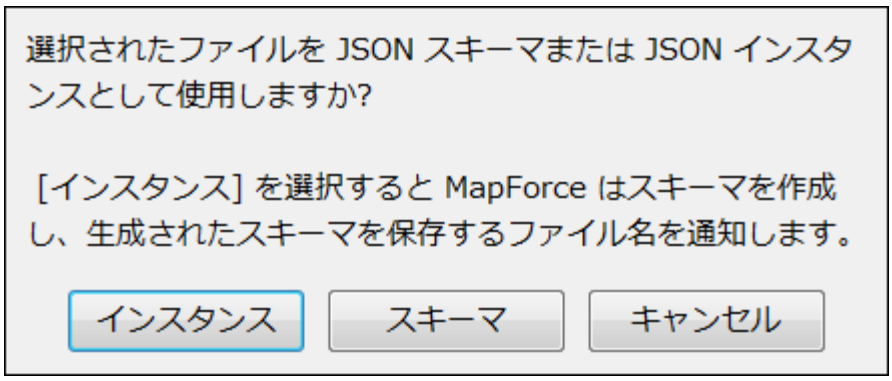
メモ MapForce を使用してW eb サービスの呼び出しが作成された場合、標準のJSON をリクエストまたはレスポンスの構造として使用することができます。JSON ライン、またはJSON5 構造はサポートされていません。

JSON ライン書式内のデータを読み取る、またはJSON ライン書式内にデータを書き込む場合、標準のJSON ファイル同様にMapForce はJSON スキーマを必要とします。違いは、JSON ラインの場合、検証のためにそれぞれのラインが個別のJSON ファイルとして扱われる点です。MapForce にスキーマを提供する場合、個別のラインはJSON ドキュメントとして扱われます。

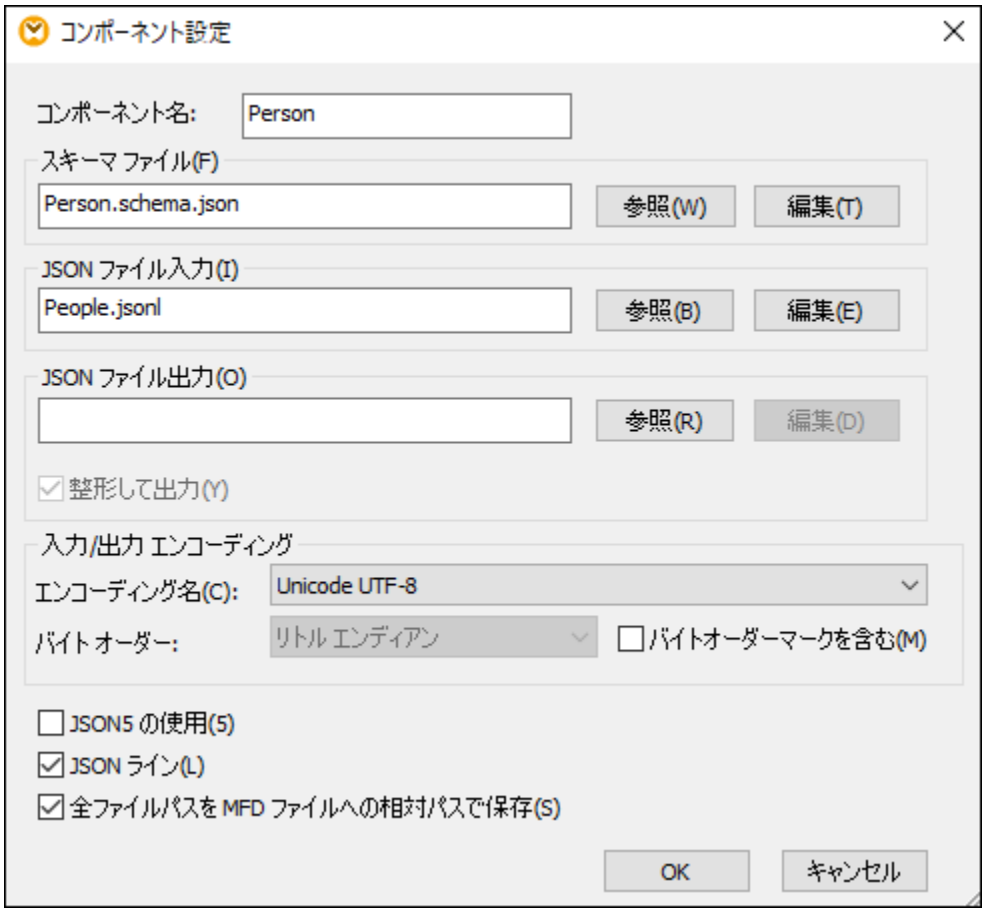
JSON ライン書式の作者により奨励されるファイル拡張子は **.jsonl** です。

JSON ライン書式へデータをマップ、またはJSON ライン書式からデータをマップする方法。

1. 「**挿入**」メニューから「**JSON**」をクリックし、**.jsonl** インスタンスファイルを参照します。または、 ツールバーボタンをクリックします。JSON スキーマファイル既存の場合、インスタンスファイルを後で提供する場合、有効な JSON スキーマを参照します。これはプレーンな JSON ファイルと作業する方法と同様に作動します（JSON ファイルをマッピングコンポーネントとして追加する を参照してください）。JSON ライン書式の場合、JSON スキーマはファイル全体としてではなく、インスタンスファイル内の各行のコンテンツを説明する必要があります。



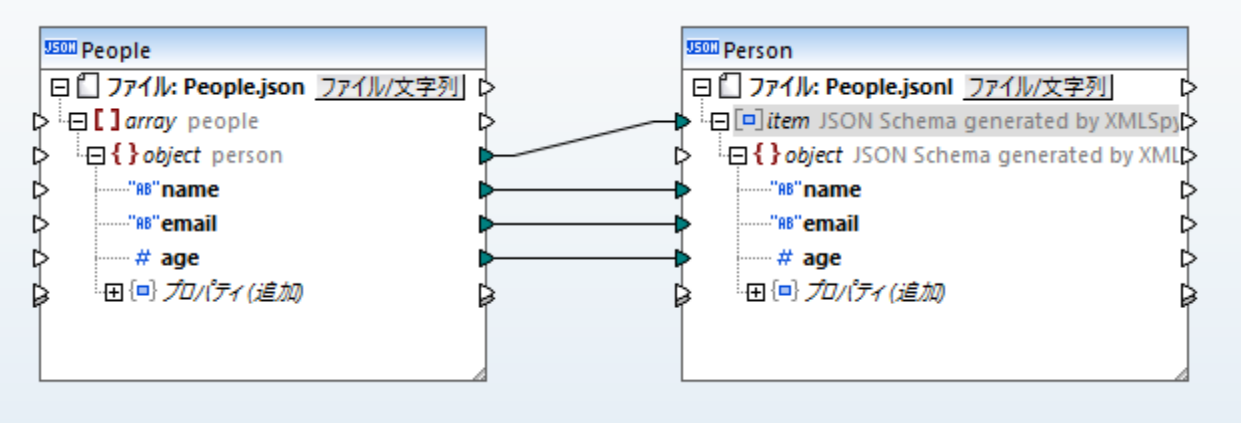
2. JSON コンポーネントがマッピングに追加された後、タイトルバーを右クリックし、「**プロパティ**」を右クリックし、「**JSON ライン**」チェックボックスを選択します。



マッピングがデータを読み取る JSON インスタンスファイルを設定、または変更するには、「**入力 JSON ファイル**」テキストボックス内にパスを入力します。

生成されたファイル名を設定または変更するには、(例えば、"somefile.jsonl" などの) 希望するファイル名を上のダイアログボックスの「**出力 JSON ファイル**」テキストボックス内に入力します。マッピングが JSON ファイルからデータを読み取る必要がない場合、このテキストボックスに入力する必要はありません。他の設定の詳細に関しては、JSON コンポーネント設定を参照してください。

JSON ラインチェックボックスが選択されていると、マッピング上の JSON コンポーネントに新規のアイテムノードが表示されます。(例えば、CSV ソースコンポーネントの **Rows** アイテムなど)、値のシーケンスを接続すると、(シーケンス内の各アイテムに一行ずつ作成することにより) JSON ラインファイル内の複数のラインを作成することができます下のマッピングでは、ソースデータは標準の JSON ファイルで、ターゲットは JSON ラインファイルです。マッピングが実行されると一番上の接続のために、ソース **People.json** ファイル内の各オブジェクトのために新規の JSON ラインがターゲット **People.jsonl** ファイル内に作成されます:



同様に、マッピングはJSON ラインファイルからデータを読み取る場合、(例えば、XML アイテムなどの)希望するターゲットアイテムにアイテムノードを接続することができます。これはソース内のJSON ラインに存在するようにターゲット内にXML 要素が作成されます。

6.6.5 例: JSON を CSV に変換する

このサンプルはJSON ファイルからデータを読み取る方法、および、MapForce を使用してコンマで区切られたテキストファイルに変換する方法について説明しています。また、このサンプルではJSON インスタンスファイル内で存在する可能性のある追加プロパティからデータをマップする方法について学びます。スキーマにより定義されていませんが、追加プロパティはJSON インスタンスファイル内に存在するデータを参照しています。ルールとして、追加プロパティに頼らず、JSON スキーマ内ですべてのプロパティを定義します。JSON インスタンスが追加プロパティを含む場合でも、このサンプル内で説明されているテクニックを使用することができます。

このサンプルには以下のパスで見つけることのできるデモマッピングファイルが存在します: **<マイドキュメント>\Altova\MapForce2021\MapForceExamples\Tutorial\\ReadJSON.mfd**。下のステップに従い、最初からこのようなマッピングファイルの作成方法を学ぶことができます。

ソースJSON ファイルには、CSV に変換される必要のあるメンバーの記録が含まれています。ハイライトされているテキストに注意してください: 最初と2番目のスキーマ内で定義されていない追加プロパティ(具体的には`birthday`)が存在します。

```
[
	{
		"name": "Alethia Alonso",
		"email": "altethia@example.com",
		"age": 35,
		"birthday": "4 July"
	}, {
		"name": "Klaus Mauer",
		"email": "klaus@example.com",
		"age": 57,
		"birthday": "31 August"
	}, {
		"name": "Natsuo Shinohara",
		"email": "natsuo@example.com",
		"age": 29
	}
]
```

People.json

(C) 2015-2021 Altova GmbH

このサンプル内で使用されているJSON スキーマは以下に表示されています。$schema キーワードが示す通り、スキーマはDraft 04 JSON Schema に対してJSON インスタンスを検証しています。複数のperson オブジェクトにより構成されているpeople 配列に関して説明しています。people オブジェクトは少なくとも1つの有効なperson オブジェクトを含んでいる必要があります。各 person オブジェクトにはname、age、およびemail がプロパティとして存在します。name とemail はstring の型を持ち、age はinteger の型を持っています。また、name とemail プロパティは必須ですが、age は任意です。

```
{
  "$schema": "http://json-schema.org/draft-04/schema#",
  "title": "people",
  "type": "array",
  "items": {
       "title": "person",
       "type": "object",
    "required": [
      "name",
      "email"
    ],
    "properties": {
      "name": {
        "type": "string"
      },
      "email": {
        "type": "string",
        "format": "email"
      },
      "age": {
        "type": "integer"
      }
    }
  },
  "minItems": 1
}
```

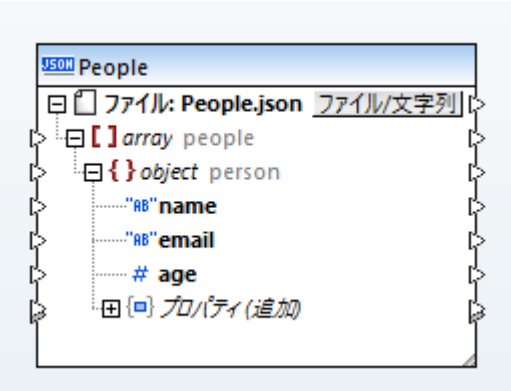
People.schema.json

重要な点は、上記のJSON スキーマはperson オブジェクトのためにadditionalProperties プロパティを持っていない点です。JSON スキーマに存在しない場合、additionalProperties プロパティはデフォルトの値として、**true** の値を持つことです。これは、JSON インスタンス内のオブジェクトは必要とされる数の追加プロパティを持ち有効であることができる点です。このため、**People.json** のようなJSON インスタンスファイルはこのスキーマに対しての検証に成功することができます。

このサンプルのマッピングの目的を達成するには、以下のステップに従ってください。

ステップ 1: JSON ファイルをマッピングに追加する方法

1. 「挿入」メニューから「JSON スキーマ/ファイル」をクリックし、次のファイルを参照します: <**マイドキュメント >\Altova\MapForce2021\MapForceExamples\Tutorial\People.schema.json**。インスタンスを指定するようにプロンプトされると、同じフォルダー内から**People.json** ファイルを選択します。この時点では、MapForce コンポーネントは以下のようになります:

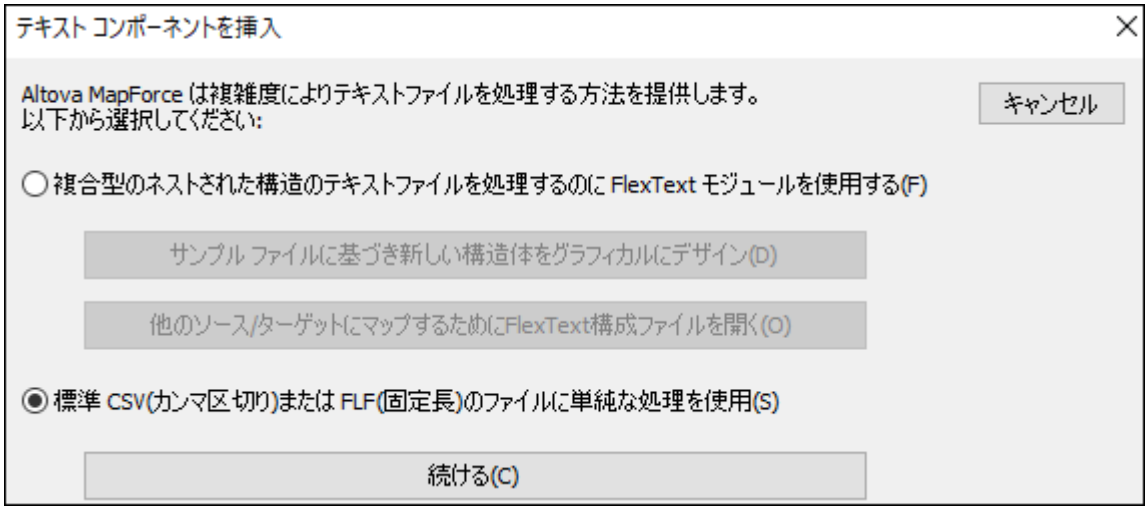


MapForce コンポーネントの構造は**プロパティ(追加)**ノードの例外を持つJSON ファイル自身に類似しています。このノードは、`person` オブジェクトの`additionalProperties` プロパティがスキーマ内で不足しているか、**true** に設定されていることを示しています。これは、スキーマは追加プロパティを含むことができ、オブジェクトの追加プロパティをマップする場合にMapForce がノードを表示できることを意味します(次のステップ上記の実行ステップが示されています)。

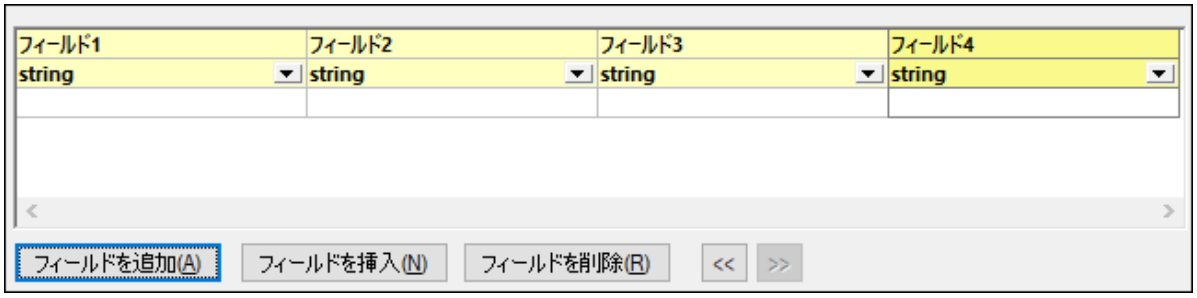
ソースJSON ファイル、またはJSON スキーマファイルを変更する場合、JSON コンポーネントのタイトルバーをダブルクリックし、設定を引き出します。 JSON コンポーネント設定 を参照してください。

ステップ 2. ターゲット CSV をマッピングに追加する方法

1. 「**挿入**」メニューから「**テキストファイル**」をクリックします。



2. 「**標準 CSV のための単純な処理**」を選択して、「**継続**」をクリックします。



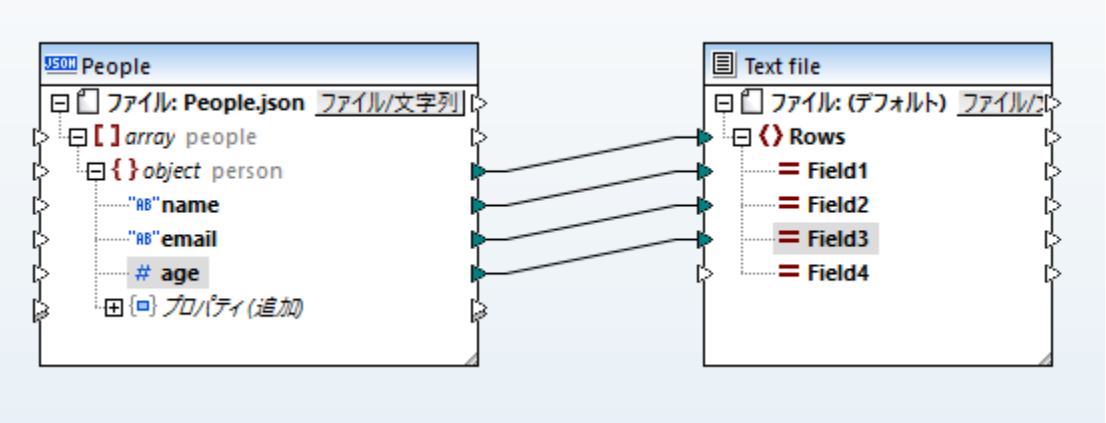
(C) 2015-2021 Altova GmbH

3. 「フィールドを追加」を数回クリックして、4つのCSV フィールドを上記の通り追加します。「OK」をクリックします。

テキストコンポーネントに関する詳細は、 CSV とテキストファイル を参照してください。

ステップ 3: マッピング接続の描き方

1. マッピング上で、JSON コンポーネントとCSV コンポーネント間に接続線を下に示されている通り描きます。



詳細に関しては、 接続との作業 を参照してください。

ステップ 4: マッピングをプレビューする方法

「出力」タブをクリックして、変換出力を素早く確認することができます。結果は以下の通りです:

```
Alethia Alonso,altethia@example.com,35,
Klaus Mauer,klaus@example.com,57,
Natsuo Shinohara,natsuo@example.com,29,
```

出力が表示するとおり、JSON ファイル内の個人のオブジェクトのために新規の行が作成され、予想されるとおり、オブジェクトプロパティがコンマで区切られて表示されています。

しかしながら、JSON インスタンスファイル内に存在し、スキーマ内に存在しない `birthday` フィールドはマップされていません。

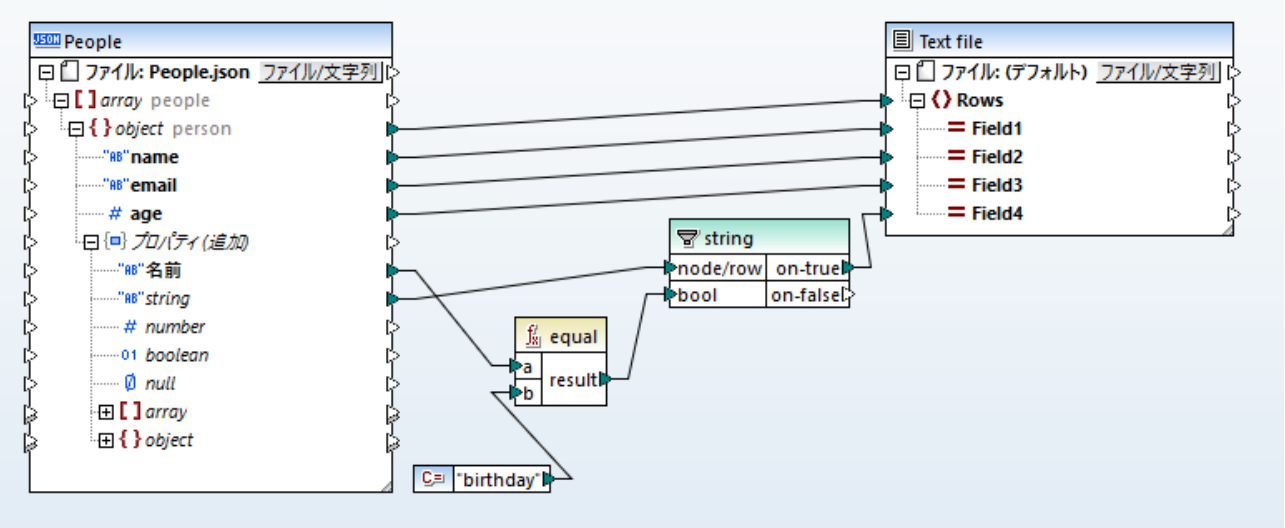
ステップ 5: 追加プロパティをマップする方法

フィールド `birthday` はスキーマ内で定義されていない追加プロパティであることを思い出してください。このようなフィールドからのデータを読み取るために、MapForce はJSON コンポーネント上に追加アイテムを提供します。

このサンプルでは、`birthday` フィールドからデータを読み取るためには、以下をマッピングに追加します:

1. 値「birthday」を持つ定数。 マッピングに定数を追加する を参照してください。
2. フィルターコンポーネント。 フィルターと条件 を参照してください。
3. ロジカルな関数 **equal**。 マッピングに関数を追加する を参照してください。

次に、コンポーネントを以下に示されるように接続します。コンポーネントをこの方法で接続すると、名前 **birthday** を持つ追加プロパティを検索するようにMapForce に命令します。文字列の値が見つかると、テキストコンポーネントの **Field4** にコピーされます。



上記のマッピングでは、**プロパティ(追加)** の下の**name** アイテムにより、名前別に追加プロパティにアクセスできるようになります。JSON データ型に対応する次のアイテム(**string**、**number**、**boolean** など) はJSON データ型に対応します。その型の実際 JSON 値にアクセスするために使用されます。サンプル内では、追加プロパティの名前は、「birthday」であり、型は「string」です。このために、これら2 つのアイテムはフィルターに接続されています。

変換出力をプレビューすると、結果は以下の通りです:

```
Alethia Alonso,altethia@example.com,35,4 July
Klaus Mauer,klaus@example.com,57,31 August
Natsuo Shinohara,natsuo@example.com,29,
```

出力が示す通り、4番目のCSV フィールドには、型「string」の追加プロパティの値のみが含まれています。この場合は`birthday` です。また、3番目の個人には誕生日が存在しなく、対応する箇所のCSV では値を使用することができません。

6.6.6 例: Excel を JSON に変換する

このサンプルはMapForce を使用してExcel ファイルからJSON ドキュメントを作成する方法について示しています。ソースExcel ワークブックは2 つのワークシートから構成されています: **Office** と**Staff**。**Office** ワークシートは企業のアドレスに関する情報を提供し、**Staff** ワークシートはすべての従業員をリストしています。

	A	B	C	D	E	F	G
1	First	Last	Title	Phone Ext.	E-mail	Department	Shares
2	Jessica	Bander	Support Engineer	241	j.band@nanonull.com	IT & Technical Support	0
3	Michelle	Butler	Software Engineer	654	m.butler@nanonull.com	Engineering	1500
4	Vernon	Callaby	Office Manager	582	v.callaby@nanonull.com	Administration	1500
5	Frank	Further	Accounts Receivable	471	f.further@nanonull.com	Administration	0
6	Liz	Gardner	Software Engineer	753	l.gardner@nanonull.com	Engineering	
7	George	Hammer	Web Developer	223	g.hammer@nanonull.com	IT & Technical Support	0

スタッフ ワークシート

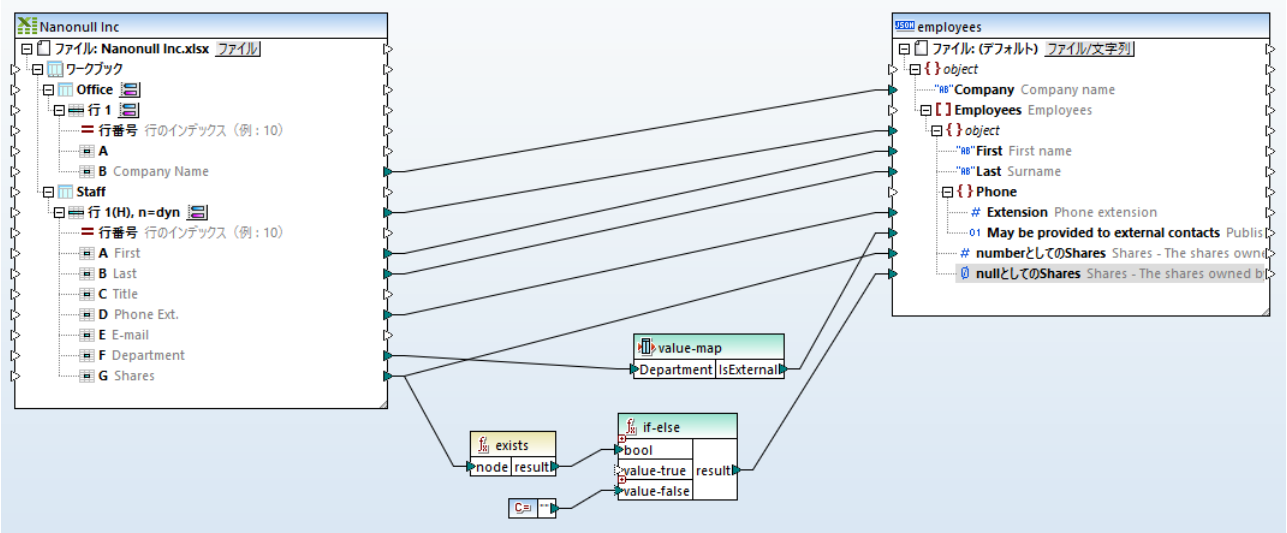
ビジネスの必要条件は以下の通りです:

1. Excel ファイルからデータを抽出し、JSON ドキュメントに変換します。

(C) 2015-2021 Altova GmbH

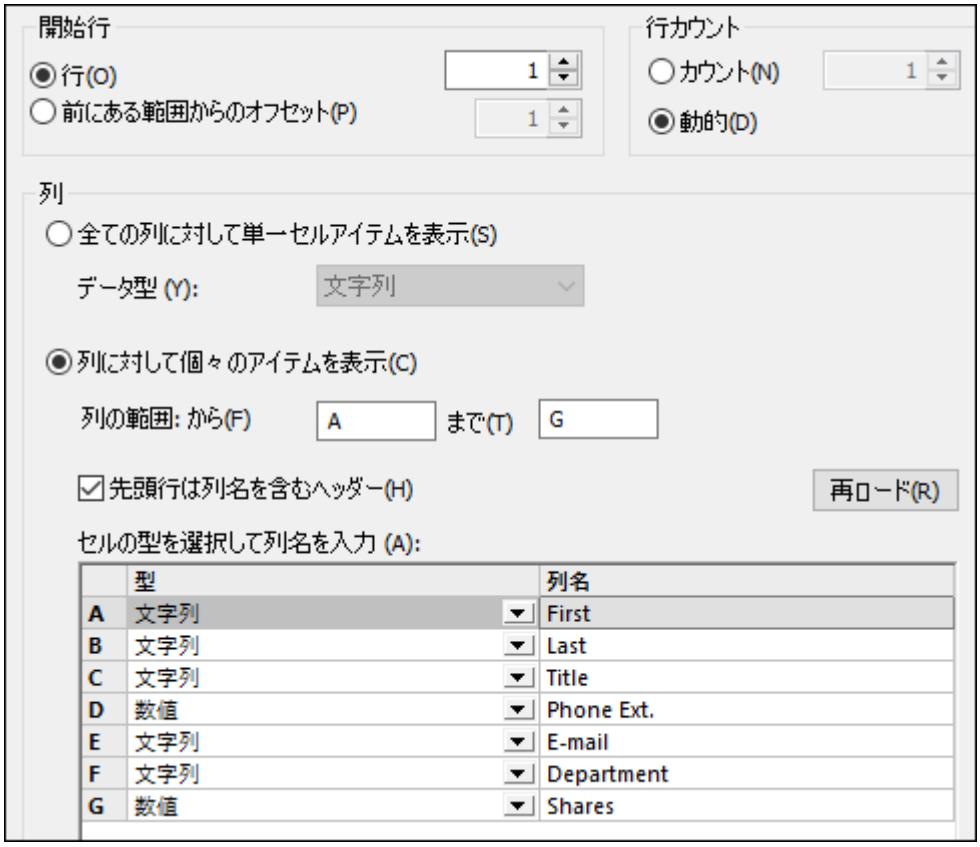
2. JSON ファイルは、各従業員のために、電話の内線番号を外部の連絡先と共有することができます。マーケティングと管理部署からの従業員の内線番号が外部と共有することができます。
3. JSON ファイル内での従業員のシェアはゼロを含む数値である必要があります。既知でない場合、シェアはJSON ファイル内の `null`を示す必要があります。

上記のゴールを達成するマッピングは下に説明されています。このマッピングは次のパスで見つけることができます: **<マイドキュメント>\Altova\MapForce2021\MapForceExamples\Excel_Company_To_JSON.mfd**。



Excel_Company_To_JSON.mfd

上に示される通り、マッピングはソースExcel コンポーネント とターゲット JSON コンポーネントにより構成されています。Excel ファイルの2つのワークシートはソースExcel コンポーネントとして表示されていることにご注意してください。 **Office** ワークシートから企業の名前はマップされますが、フィールドのその他は**Staff** ワークシートからマップされます。各ワークシートから行データを読み取るように、マッピングがどのように構成されているかを確認するには、対応する ボタンをクリックしてください。**Staff** ワークシートに関しては、設定は以下に示されているとおりです。

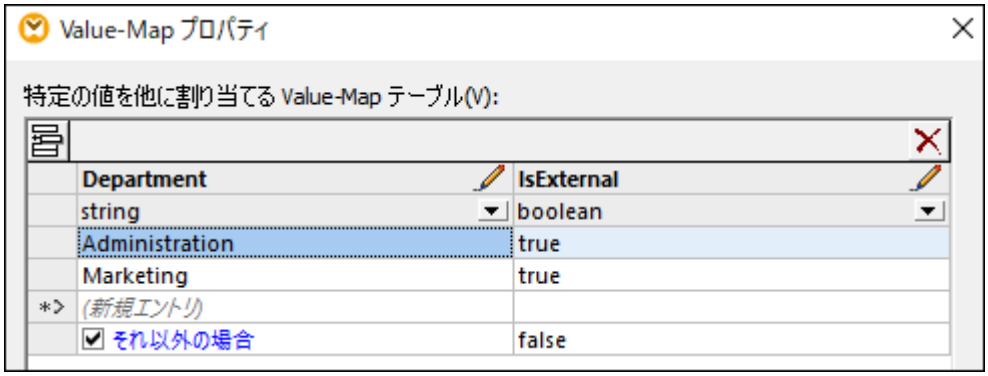


上記の設定は1 から終わり(動的)までの行を読み取るようにマッピングに命令しています。最初の行は列名を持つヘッダーで、「**最初の行はヘッダー**」チェックボックスが選択されています。ダイアログは列 A から G への型を指定します。 Excel コンポーネントに関する詳細は、Microsoft OOXML Excel 2007+ を参照してください。

ターゲット コンポーネントはJSON 構造です。JSON コンポーネントを JSON ファイルをマッピングコンポーネントとして追加する方法で説明される通りマッピングに追加します。構造はマッピングと同じフォルダー内で見つけることのできる JSON スキーマ **employees.schema.json** を反映しています。JSON コンポーネントの構成に関する詳細は JSON コンポーネント設定 を参照してください。

マッピング上には複数の中間コンポーネントが存在します。最初はvalue-map コンポーネントです。value-map は部署に従い、電話番号が外部と共有できるかを決定します。これは、上記のビジネス条件 #2 に対応しています。タイトル バーをダブルクリックすることにより、value-map コンポーネントの設定をビューすることができます:

(C) 2015-2021 Altova GmbH



上記の構成は以下の通り要約することができます: **Department** が「Administration」または「Marketing」の場合 `true` が返されます。それ以外の場合、`false` が返されます。この方法で計算された`true` または`false` 値はターゲット JSON プロパティに書き込まれます。値マップに関する詳細は、Value-Map の使用を参照してください。

最後に、マッピングはソースExcel ファイル内に**Shares** 値が存在する場合、各行のコンテキスト内で目的を決定するためのexists 関数を活用します。この関数は、値が存在すると、Boolean `true` を返し、それ以外の場合は、`false` を返します。結果の`true` または`false` 値は、**If-Else** 条件に入力として与えられます。`false` の場合、定数により提供されている空の文字列が「null として共有」ターゲットアイテムにパスされます。「null として共有」アイテムのデータ型であるため、 ∅ アイコンにより示されている通りこの空の値は結果のJSON ファイル内で`null`になります。

従業員の共有する値が数値の場合、**If-Else**条件を満たしません。代わりに、個別の接続を介して、(数値の型である #)「**番号として共有**」ターゲットアイテムに書き込まれます。

マッピングに関数を追加する詳細に関しては、マッピングに関数を追加する を参照してください。**If-Else** 条件はフィルターと条件で詳しく説明されています。

マッピングの実行方法

MapForce 内で生成されたファイルをプレビューするには、「出力」タブをクリックします。下のイメージでは結果のJSON ファイルのフラグメントのみが表示されています。

```
{
    "Company": "Nanonull, Inc.",
    "Employees": [
        {
            "First": "Jessica",
            "Last": "Bander",
            "Phone": {
                "Extension": 241,
                "May be provided to external contacts": false
            },
            "Shares": 0
        }, {
            "First": "Michelle",
            "Last": "Butler",
            "Phone": {
                "Extension": 654,
                "May be provided to external contacts": false
            },
            "Shares": 1500
```

Mapping output

マッピングの出力をファイルに保存するには、次のいずれかを行います。:

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

MapForce Server を使用して自動化する方法

MapForce Server にライセンスを供与済みの場合、Linux、macOS、または Windows マシン上でマッピングをコマンドラインで実行することができます。サーバー上でのマッピングの実行は、サーバーでの実行のためにマッピングを準備する内で説明されている通り、いくつかの追加ステップが必要になります。サーバー上でこの特定のマッピングを実行するために必要なステップは以下の通りです:

1. マッピングを右クリックし「マッピングの設定」をコンテキストメニューから選択します。「生成されたコード内でパスを絶対的にする」チェックボックスがクリアされている状態でマッピングを保存します。
2. 「ファイル」メニューから「MapForce Server 実行可能ファイルにコンパイルする」をクリックします。これにより実行可能 .mfx ファイルが<マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥ ディレクトリ内で生成されます。
3. 生成された .mfx ファイルファイルをサーバーマシン上のディレクトリにコピーします(「作業ディレクトリ」と呼ぶことにします)。
4. Excel ファイルを<マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥**Nanonull Inc.xlsx** からサーバーマシン上の作業ディレクトリにコピーします。
5. ターミナルを開き、現在のディレクトリを作業ディレクトリに変更します。
6. MapForce Server を以下のコマンドを利用して実行します。使用中のオペレーティングシステムに適用できるように実行可能ファイルへのパスに対して **mapforceserver_exec** を調整します。

```
mapforceserver_exec run Excel_Company_To_JSON.mfx
```

サーバー実行内で、オンデマンド、または再帰的なベースで、API 呼出としてマッピングを実行、またはFlowForce Server ジョブとして実行することができます。詳細に関しては、 MapForce Server 実行可能ファイルにマッピングをコンパイルする を参照してください。

6.7 MS OOXML Excel 2007 以降のファイル

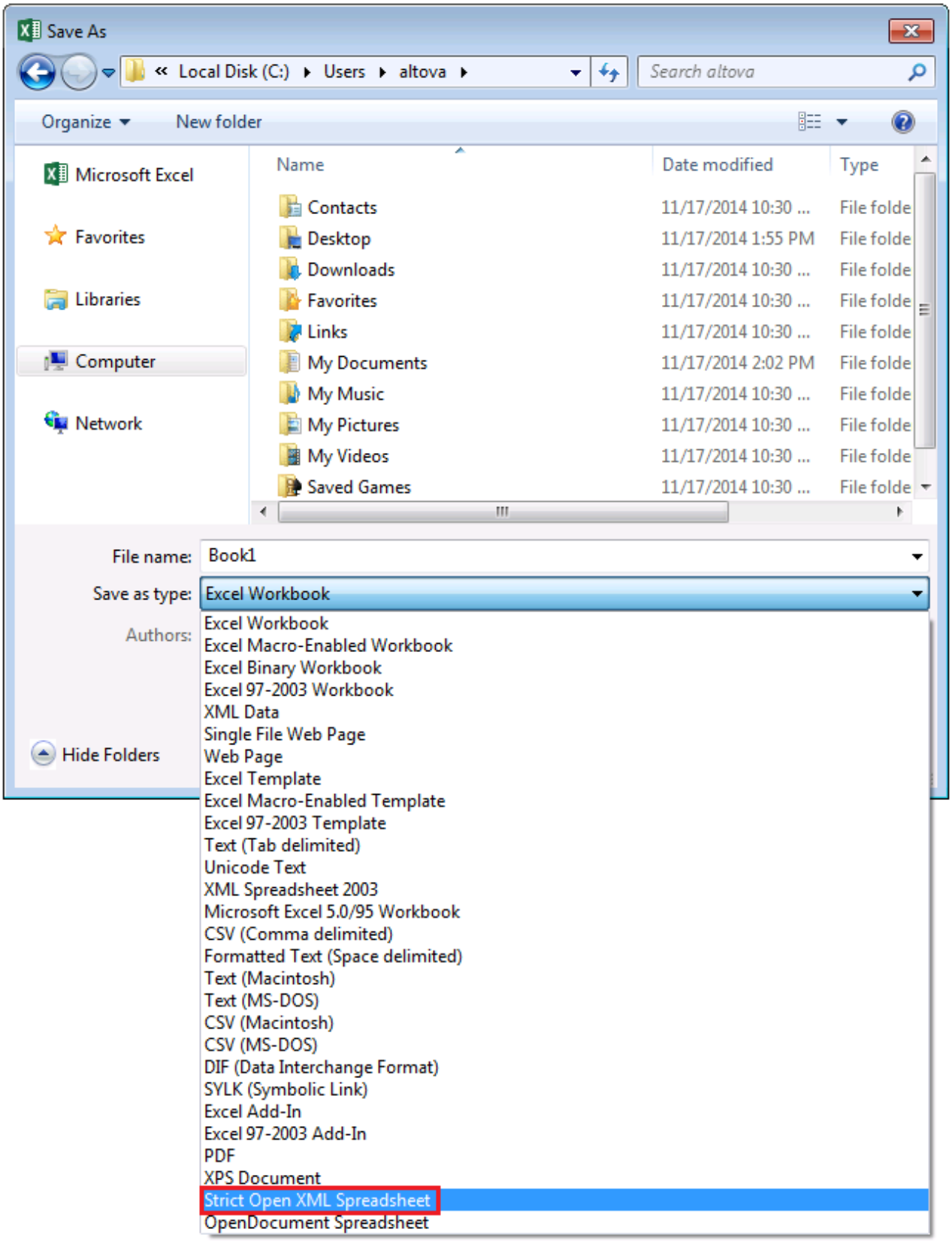
MapForce では、Microsoft Excel 2007 以降の(Open Office XML) ファイル(*.xlsx) をソースならびにターゲットコンポーネントとしてマッピングすることができます。このフォーマットは、デフォルトの .xlsx 拡張子に関連した Excel ワークブックのために最初に Microsoft Office 2007で紹介されました。MapForce インターフェイス内とこのドキュメントでは、Microsoft Excel 2007 と以降のファイルは、通常 "Excel 2007+" ファイルと呼ばれます

Microsoft Office 2013 は *Strict Open XML スプレッドシート* フォーマット (ISO/IEC 29500 Strict) へのサポートの搭載を開始しました。*Strict Open XML スプレッドシート* フォーマットも .xlsx 拡張子を有しますが、技術的には、更に厳格な検証ルールに準拠する特別のフォーマットです。

以下のテーブルは、MapForce 変換言語 によりサポートされるExcel 2007+ ワークブック から、またはへのデータの読み込みおよび書き込みを示したものです。 *Office Open XML* と *Strict Open XML スプレッドシート* フォーマットの違いに注意してください。

Microsoft Excel フォーマット	MapForce 言語	読み込み サポート	書き込み サポート
Office Open XML	BUILT-IN	はい	はい
	C#	はい	はい
	Java	はい	はい
	XSLT 2.0、XSLT 3.0	はい	いいえ
Strict Open XML スプレッドシート	BUILT-IN	はい	いいえ
	C#	はい	いいえ
	Java	はい	いいえ
	XSLT 2.0、XSLT 3.0	はい	いいえ

MapForce により生成された *Office Open XML* ファイルを *Strict Open XML スプレッドシート* フォーマットに変換する場合は、ワークブックを Excel 2013 で開き、*Strict Open XML スプレッドシート*として保存します。



(C) 2015-2021 Altova GmbH

Strict Open XML スプレッドシート フォーマットに保存する

MapForce 内のマッピングコンポーネントとして、Excel 2007+ ファイルの一般的な振る舞いは以下のとおりです:

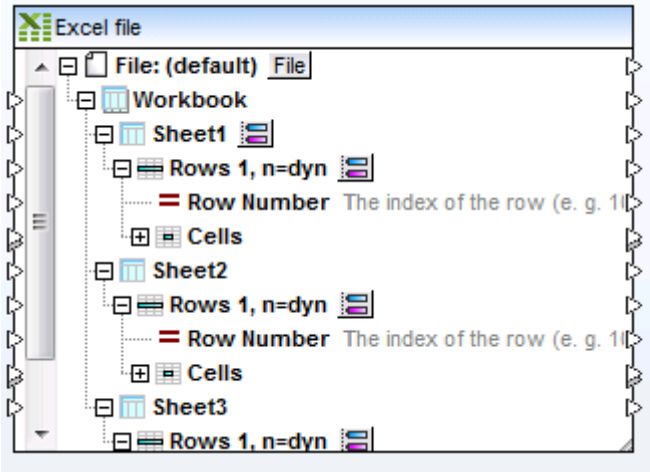
- XBRL タクソノミファイルを含む、Excel 2007+ からのデータを MapForce にサポートされるコンポーネントにマップ、またはその逆、を行うことができます（Excel から XBRL へマップの例を参照）。
- Microsoft Excel 2007 または以降 が使用中のマシンにインストールされている場合、変換の出力をすぐにマッピングウィンドウの「出力」タブで確認することができ、ファイルに保存することができます。Excel 2007 または以降が使用できない場合でも、Excel 2007+ ファイルからマップを行う、またはExcel 2007+ ファイルにマップを行うことはできますが、「出力」タブで、結果をプレビューすることはできません。ファイルは「**出力**」メニューの「**出力ファイルの保存**」をクリックすることにより、保存することができます。

  > 複数の高度-DPI モニターを使用する場合、異なるDPI-対応レベルが存在する場合、Excel 2007+ 出力を MapForce 内で直接プレビューすることはできません。以下に類似したメッセージが表示されます:「Excel の埋め込みは現在のシステムの構成では不可能です。」

- 新規のExcel ファイルをマッピングから作成、または、既存のExcel ファイルを更新することをコンポーネントの設定 から選択することもできます。
- 新規のExcel ファイルを作成する場合、Excel 2007+ 出力は書式設定は含まず、データのみを含んでいます。
- 既存のExcel ファイルを更新する場合、既存のExcel ファイルの書式情報はそのまま保管されます。フォーミュラを含むセルの更新はサポートされていません。代わりに、このようなセルに遭遇するとコンポーネント設定 から構成して（メッセージウィンドウに警告を表示したまま）エラーのあるマッピングの実行を停止、または無視することができます。
- 手動で出力を生成し保存する代替として、マッピングデザインを MapForce Server 実行ファイルにコンパイルすることができ、またはFlowForce Server にデプロイして、FlowForce Server ジョブにより必要とされる場合は、実行することもできます。更に詳しい情報に関しては、MapForce マッピングのコンパイルとMapForce マッピングのデプロイをそれぞれ参照してください。
- マッピングコンポーネントによりサポートされる場合、マッピングからプログラムコードを生成し、カスタムアプリケーションから外部的に実行することができます（上記のサポートされる言語を参照してください）。詳細に関してはコードジェネレーターを参照してください。「**既存のExcel ファイルの更新**」機能は変換言語がBUILT-IN の場合のみ使用できることに注意してください。

6.7.1 Excel 2007+ コンポーネントについて

サンプルファイルを指定せずに、Excel 2007+ ファイルをマッピングエリアに追加すると、（Microsoft Excel ファイルをマッピングコンポーネントとして追加するを参照）、MapForce は、（下に表示される）3つのワークシートを含むデフォルトのコンポーネントを作成します。サンプルファイルを与えると、MapForce は、サンプルファイルを読み込み、必要とされるワークシートのみを作成します。



デフォルトのExcel コンポーネント

MapForce 内のExcel 2007+ コンポーネントの構造は、MapForce では、それぞれのセルをマップできるツリー構造で表示されている点が異なりますが、Excel ワークブック内のデータ構造を反映します。

Excel 2007+ コンポーネントを他のコンポーネント型に接続する前に、 MapForce に正確にどの行と列がマッピングで使用されるかを命令する必要があります。XML または JSON などの他のMapForce コンポーネントとは異なり、Excel 2007+ ファイルには、MapForce がデータの構造を示すための明確なスキーマが存在しません。代わりに、定義することのできる設定がMapForce により与えられます:

- (ワークシート、範囲とテーブル、列、行など)どのデータが (ワークブックから読み込んでいる場合)ワークブックから明確に選択されるかを定義。
- MapForce が書き込む先のワークシート、列と行 (ワークブックに書き込む場合)を定義。

ワークブック内の複数の箇所から読み込無ため、同じマッピングオペレーション内の複数の場所に書き込むために、にコンポーネントを構成することもできます。

必要とするコンポーネント構成設定を直接コンポーネントに対して使用することができます。以下のテーブルを使用して、使用を開始してください。

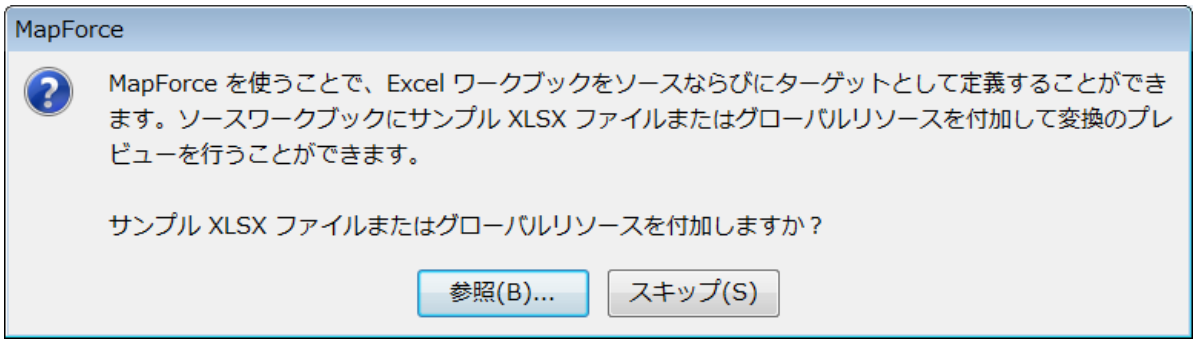
コンポーネントアイテム	アイコン	説明
ワークブック		Excel ワークブックを表します。
ワークシート		Excel ワークシートを表します。最初のワークシートの横に表示されるボタンによりワークシートに関連した設定を指定することができます。
行		Excel 行の範囲を表します。ワークシート内に複数の行の範囲を追加することができます。これにより、同じマッピングオペレーション内のセルの複数の範囲から読み込む、または範囲に書き込むことができます。 定義されたそれぞれの範囲のために、個別のデータ選択オプションを指定することができます。例: ソースExcel ファイル内のデータの量に従い、1 つの範囲は行 1 から開始し、その行のすべての列を含む一方、他の範囲は、行 3 から開始し、動的な数の行から構成されることができます。 以下を参照してください一目で全ての範囲の設定を確認することができます。コンポーネントは、以下に説明されるように視覚的なヒントを提供します。 **行 *n*** — ***n*** 行から開始する範囲を示します。 **行 *n*** — ***n*** 行から開始する1 行表示の範囲を示します。 **行 前+*n*** — 前の範囲の後の、 ***n*** 行から開始する範囲を示します。 **行 *n*(h)** — ***n*** 行から開始する範囲を示します。最初の行はヘッダー行として開始されます。 **n=*n*** — ***n*** 行から構成される範囲を示します。 **n=dyn** — 動的な範囲を示します。動的な範囲は無制限の数量の行を持つことができます。

コンポーネントアイテム	アイコン	説明
		各行の範囲の横に表示される ボタンにより、その範囲のための高度なデータの選択を指定することができます。
セル		特定の行の全てのセル(列) を示します。このアイテムは、コンポーネントが全てのれるのために単一のセルに構成されている場合表示されます(MapForce のデフォルトの振る舞いです)。 または、以下に表示されるように、コンポーネントが各列を個別に表示するようにも構成できます。
選択の変更		各ワークシート、または行の横に表示される ボタンにより、そのコンテキストで意味をなす設定を指定することができます。このボタンを使用して、Excel コンポーネントの構造 を必要に応じて修正することができます。 例:Excel ファイル空データを読み取る場合、MapForce がデータを読み取る ワークシート、行、列を指定することができます。Excel ファイルに書き込む場合、MapForce がデータを書き込む ワークシート、行、列を指定することができます。

6.7.2 Excel 2007+ ファイルをマッピングコンポーネントとして追加する

Microsoft Excel 2007+ (*.xlsx) コンポーネントをマッピングエリアに追加する

1. 以下を行います:
   - 「挿入」メニューから「Excel 2007+ ファイル」をクリックします。
   - 「Excel 2007+ ファイルの入力」( ) ツールバーボタンをクリックします。

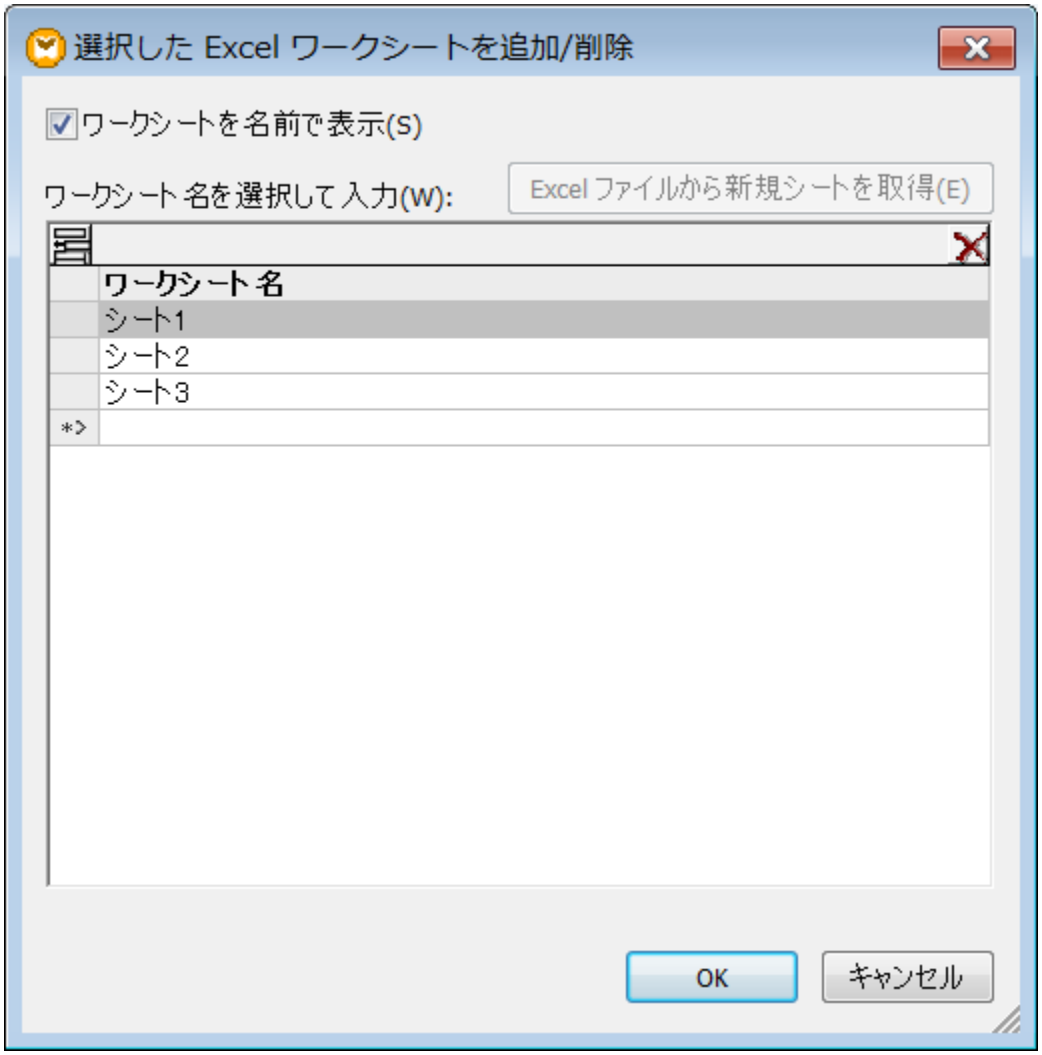


2. 以下を行います:

   - Microsoft Excel ワークブックからデータをマップする場合、「参照」をクリックしデータをマッピングする .xlsx ファイルから選択します。MapForce は、サンプル Excel ファイルを使用して、ワークシート名とセルの範囲を読み込みます。Excel ファイルをグローバルリソースとして定義した場合、グローバルリソースダイアログボックスからこれらを選択することもできます（グローバルリソースを参照）。サンプルファイルを後で追加する場合、「スキップ」をクリックします。
   - Microsoft Excel ワークブックへデータをマップする場合、「スキップ」をクリックします。デフォルトでは、マッピング変換が実行されると、MapForce は、**xlsx-mapforce.xlsx** という名前の出力 Excel ファイルマッピングフォルダー内に生成します。必要な場合は、出力ファイルの名前を Excel 2007+ コンポーネント設定から変更することができます（Excel 2007+ コンポーネントについてを参照）。

6.7.3 ワークシートの追加と削除

Excel 2007+ コンポーネントのワークシートを手動または入力 .xlsx ファイルをロードすることにより、追加または削除することができます。ワークシートノードの横の ボタンをクリックします。



Excel ワークシートの追加/削除選択ダイアログボックス

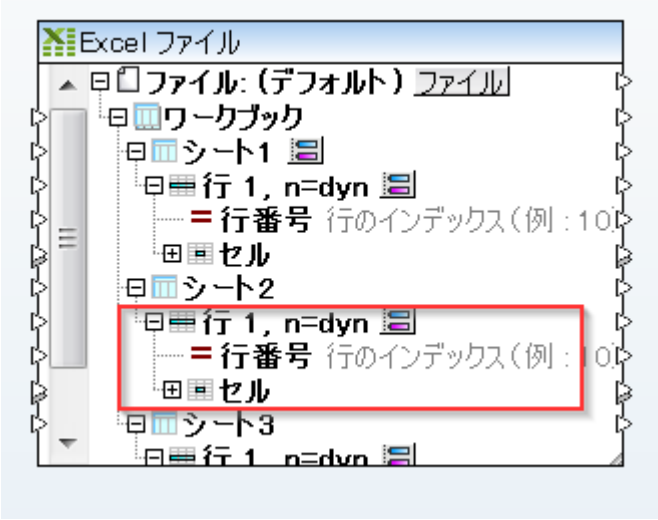
ワークシートに適用することのできるオプションは、以下のとおりです。

ワークシートを名前別に表示する	ワークブック内のそれぞれのワークシートが異なるレイアウトを有する場合、また、このためにMapForce 内で個別に表示される場合、このオプションが選択される必要があります。 ワークブック内のワークシートに同様の構造が存在する場合、このオプションを非アクティブに設定することができます。こうすることにより、MapForce は、ワークシートアイテムをすべてのワークシートの順序付けられたコレクションを表すように単一のアイテムに折りたたみます。 ワークブックに複数のワークシートが存在する場合 **ワークシートを名前別に表示する** オプションが非アクティブではなく MapForce はワークブックを単一のワークシートとして扱います。これにより任意の数のワークシートを一度に処理することができるようになりますが、そのためには処理するワークシート全てが同一の構造をしている必要があります。例: Excel 行を XML ファイルに変換するを参照してください。

挿入 ( )	現在選択されているワークシートの前に、新たなワークシートを挿入します。
追加 ( )	新たなワークシートを追加します。アイコンの右側にあるテキストフィールドに名前を入力してください。
削除 ( )	現在選択されているワークシートを削除します。
Excel ファイルから新規シートを取得	新しいワークシートが検索した新しいワークシート内のコンポーネントを取得します。この設定は、関連性が高く、入力Excel ファイルファイルを持つソースコンポーネント内で有効化されます。

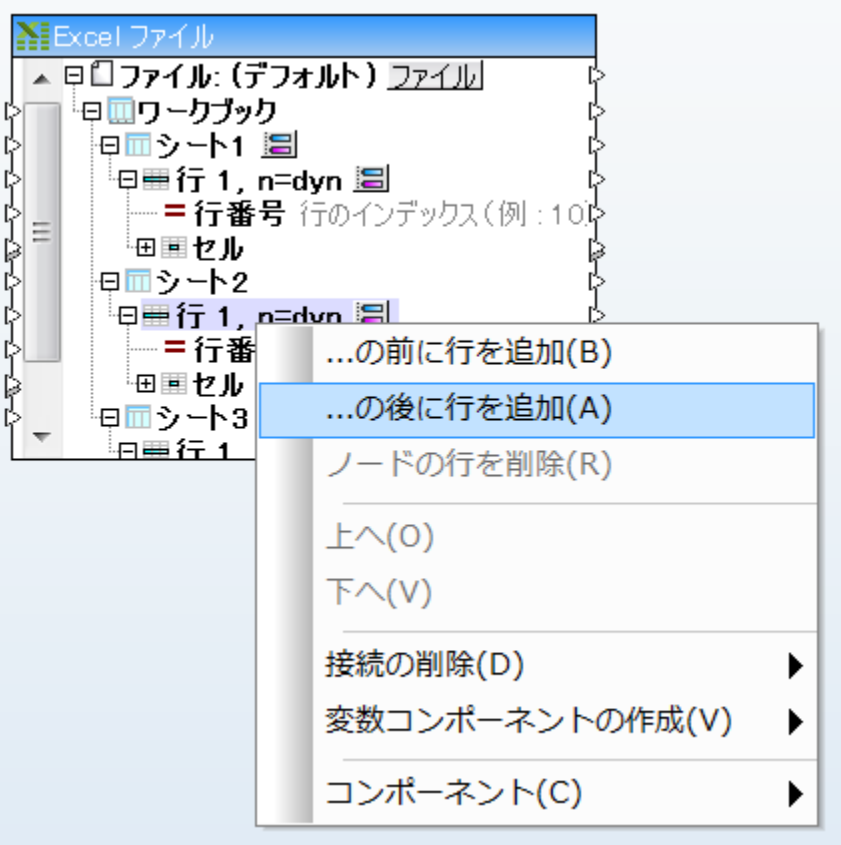
6.7.4 行の範囲の追加と移動

MapForce にワークシート内の特定の場所にあるセルの範囲に読み込み（または書き込み）を行うように命令することができます。以下のサンプルコンポーネントは、ワークブックのシート2で使用することのできる範囲について表示指しています。



デフォルトでは、行 1 から開始する範囲を設定し、￥n 行（**n=dyn**）反復することができます。必要な場合、しかしながらこれらと他の設定を変更することもできます（ セルの範囲の選択を参照してください）。

同じワークシート内で必要な数量だけ行の範囲を作成することができ、必要でない行を削除することもできます。新規の行の範囲を追加するには、行（ ）ノードを右クリックし、「**前に行を追加**」または「**後に行を追加**」をそれぞれ選択します。



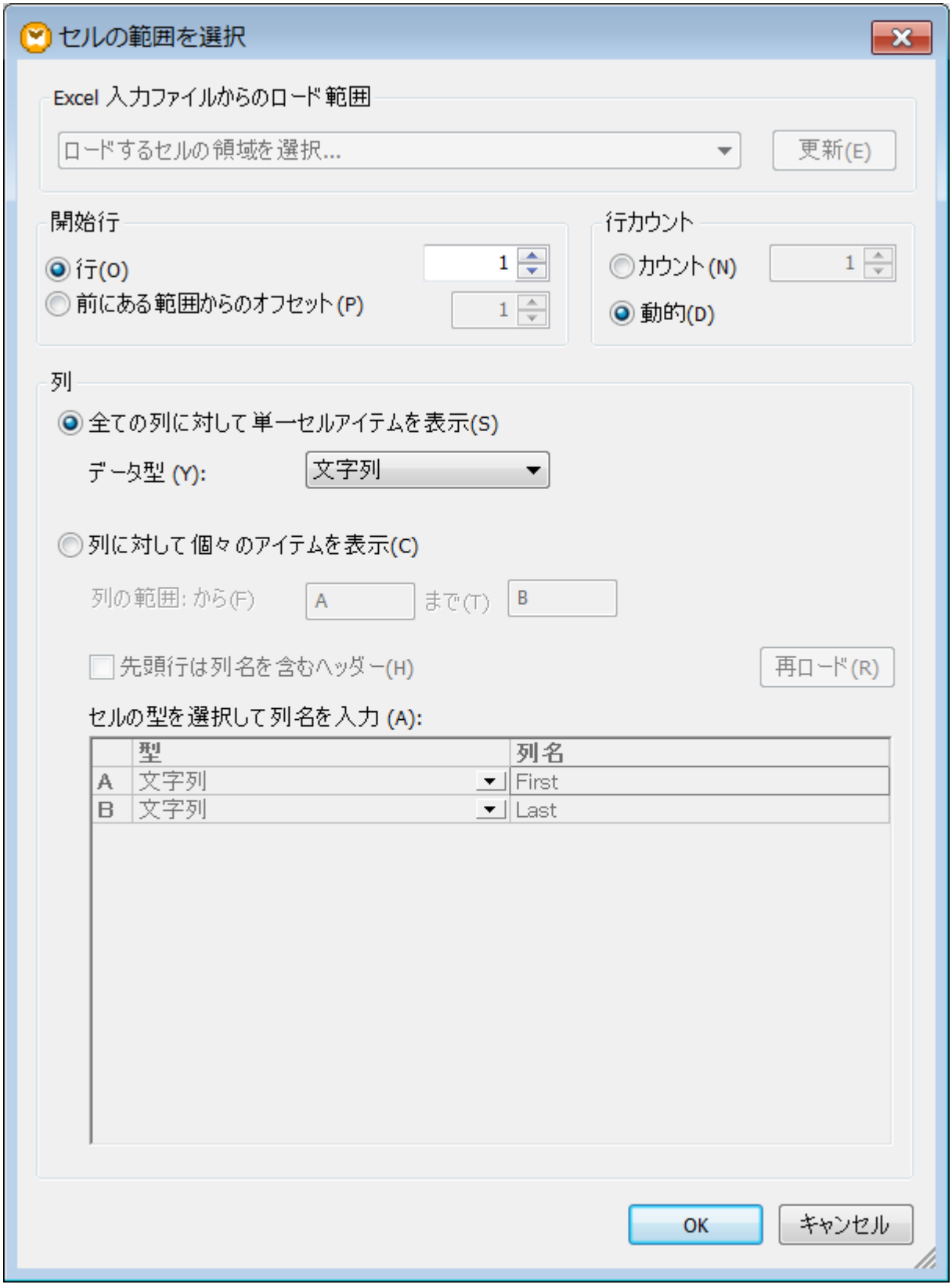
セルの範囲の追加

範囲を削除するには、右クリックし、「**行の削除**」を選択します。

コンポーネント内の範囲を上にまたは下に移動するには、右クリックし、「**上に移動**」(または「**下に移動**」) を選択します。

6.7.5 セルの範囲の選択

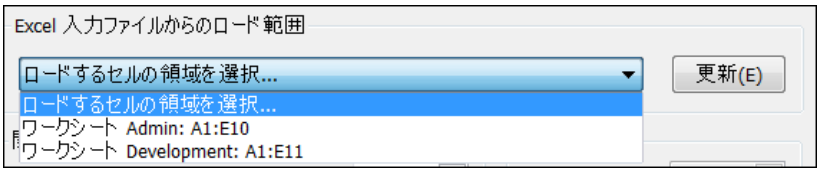
MapForce により(ワークブックから) 読み込まれる、またはワークブックに書き込まれるセルの範囲を"セルの範囲の選択" ダイアログボックス 定義することができます。このダイアログボックスを開くには、コンポーネントのセルの範囲の横の ボタン クリックします。

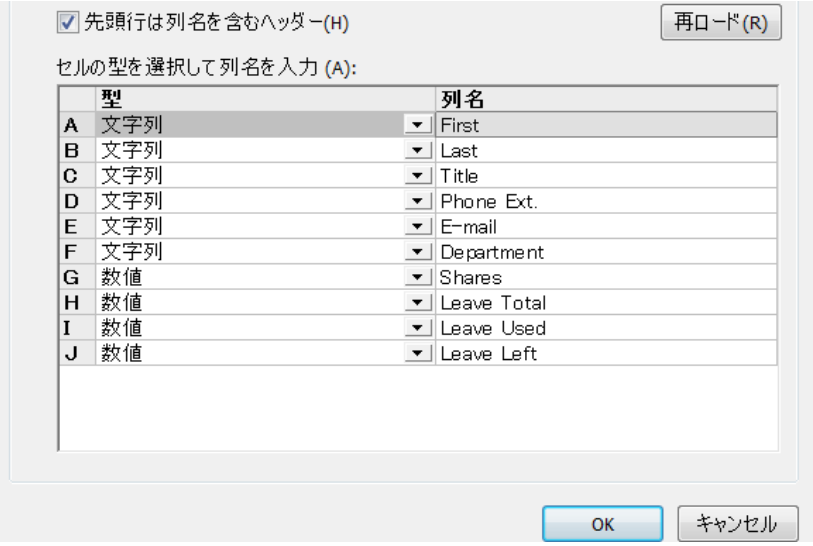


(C) 2015-2021 Altova GmbH

セルの範囲を選択 ダイアログボックス

使用することのできる設定は以下の通りです。

Excel 入力ファイルから範囲をロードする	ワークブックで作業する場合、このオプションを特定のワークシートの範囲、名前の付いた範囲、またはテーブルに対して使用します。  **ワークシートを名前順に表示する** オプションが無効化されている場合（ワークシートの追加と削除を参照して）、すべてのワークシートのデータがリストに表示されます。 「**更新**」ボタンは、入力 Excel ファイルからのセルの範囲を更新します。 MapForce では現在短形の範囲しかサポートされていないことにご注意してください。 ワークブックで作業する場合、このオプションを使用することはできません。
先頭行	**行** オプションにより特定の範囲のデータの最初の行を定義できます。"5" を開始される行に入力すると、MapForce はワークブックの 5 番目の行を読み込み（または書き込み）ます。 **前にある範囲からのオフセット** オプションは同じワークシート内に他の範囲が存在する場合役に立ちます。これは MapForce に現在の範囲の *N* 行を前に定義された範囲から移動するように命令します。最小のオフセットの値は 1 です。
行カウント	*ワークブックから読み込む場合* **カウント** は **行の開始** 内で定義された位置から開始してデータを読み取る行の正確な行数を定義します（前のオプションを参照してください）。この値は、Excel の名前の付けられた範囲またはテーブルを選択すると自動的に作成されます。 **動的** オプションは、MapForce に先頭行から開始するソースデータ内のすべての行を読み込みます。このオプションは、範囲がワークシートの最後に定義された範囲である場合のみ使用します。それ以外の場合、次の範囲はソース Excel ファイルからデータを選択しません。 *ワークブックに書き込む場合* **カウント** は **行の開始**（ 内で定義された位置から開始してデータを読み取る行の正確な行数を定義します（前のオプションを参照してください）。入力インスタンスが **カウント** により許可された値よりも多い行数を含む場合、MapForce は **カウント** により定義された行数のみを書き込み、残りのデータを警告なしに無視します。 オプション **動的** は MapForce に命令して、**開始行** から開始する入力インスタンス内で検知されたすべての行に書き込みます。 **先頭行が列名を含むヘッダー** オプションを使用するヘッダー行を定義すると、**カウント** はヘッダー行を数に数えません（ **Excel_Company_to_XML.mfd** サンプルを参照）。



すべての行のための単一セルアイテムを表示する

下で表示されるように、すべてのアイテムを単一のマップすることのできるセル *Cells* アイテムに折りたたみます。

ワークブックから読み込む場合
特定の行のすべてのセルから読み込む場合は、このオプションを使用します。例えば、**ExcelColumnsToRecords.mfd** と **ExcelWith2Dimensions.mfd** は **MapForceExamples** プロジェクト内で使用することのできるサンプルです。**OOXML Excel 2007+** フォルダーを参照してください。

ワークブックに書き込む場合
1つまたは複数のセルにデータを書き込む場合このオプションを使用します。例えば、**Altova_Hierarchical_Excel.mfd** サンプルを参照してください。

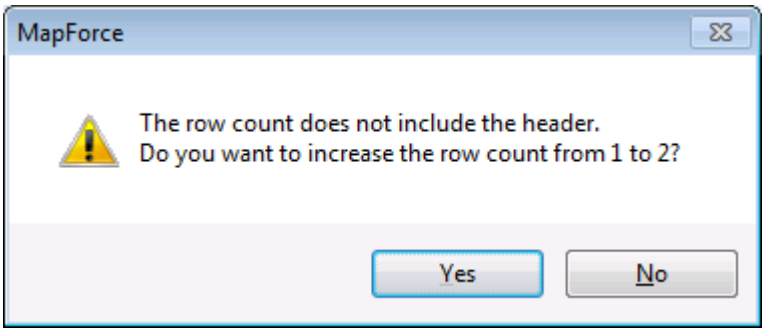
列のために個別のアイテムを表示する

このオプションにより与えられた行範囲の個別の行にアクセスすることができます。

範囲またはテーブルという名前のワークシートの範囲を選択した場合、列名は自動的に作成されます。それ以外の場合、特定の列名を対応するアルファベット文字を **from** から **to** へのテキストボックス入力することにより選択することができます。

入力 Excel ファイル内の範囲が変更されると、変更と共にコンポーネントを更新するために「**再ロード**」をクリックします。

(C) 2015-2021 Altova GmbH

	MapForce にその範囲の列ヘッダーとして最初の行を数えるように命令するには、**先頭行は列名を持つヘッダー** オプションを選択します。このオプションを有効化または無効化し **行カウント** が設定されると、MapForce は **行カウント** 値を任意で調整するようにプロンプトします。これは **行カウント** が 1 つ多く、または少なく数えられることを防ぐためのものです。  **行カウント** 設定に入力された行の数には、ヘッダー行が含まれないことにご注意してください。

6.7.6　既存の行の間に行を挿入する

Excel スプレッドシートにデータをマップする場合、既存のマッピング接続をやり直すことなく、既存の行の間に新規の列を簡単に追加することができます。これを行うには、Excel コンポーネント内のセルを右クリックし、コンテキストメニューから「**セルを前に追加する**」を選択します.

メモ　「**セルを前に追加する**」コマンドはセルが行の範囲に属する場合使用することができます。(すなわち、Excel コンポーネントの「行」アイテムの子である必要があります)。また、オプション「**行のために個別のアイテムを表示する**」が「セルの範囲の選択」ダイアログボックス内で有効化されている必要があります（次を参照してください: セルの範囲の選択)。

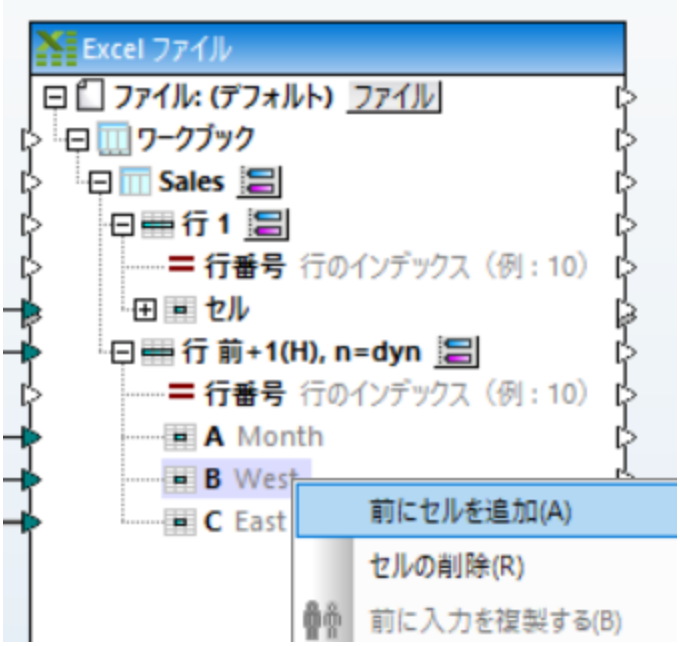
例えば、XML ファイルから Excel スプレッドシートにデータを書き込むマッピングを作成すると想定します（フォルダー <**マイドキュメント**>**\Altova\MapForce2021\MapForceExamples** の **Sales_to_Excel.mfd** など)。3 つの列（Month、W est、とEast) から構成される Excel スプレッドシートが下に示されるようにマッピングから出力されます。

	A	B	C	D
2	Month	West	East	
3	Jan	110.4	75.3	
4	Feb	114.3	65.2	
5	Mar	134.2	86.1	
6	Apr	107.3	112.1	
7	May	114.4	93.8	
8	Jun	113.9	72.4	
9	Jul	89.4	67.4	
10	Aug	95.3	84.9	
11	Sep	107.2	99.5	
12	Oct	129.7	82.5	
13	Nov	137.1	101.9	
14	Dec	152.6	120.6	
15				

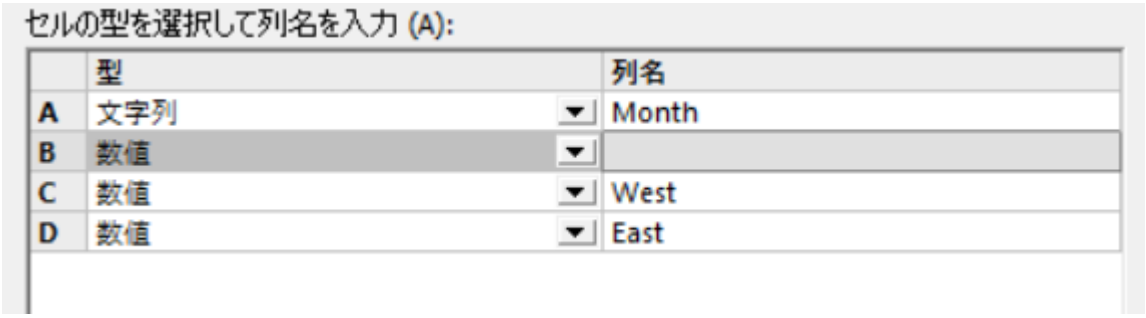
Sales_ to_ Excel.mfd のマッピング出力

「W est」（例えば、「North」）の前に新規の列を追加する必要があるとし、これは次のように既存の接続をやり直すことなく行うことができます:

1. マッピング上のセルB を右クリックし、コンテキストメニューから「セルを前に追加する」を選択します。



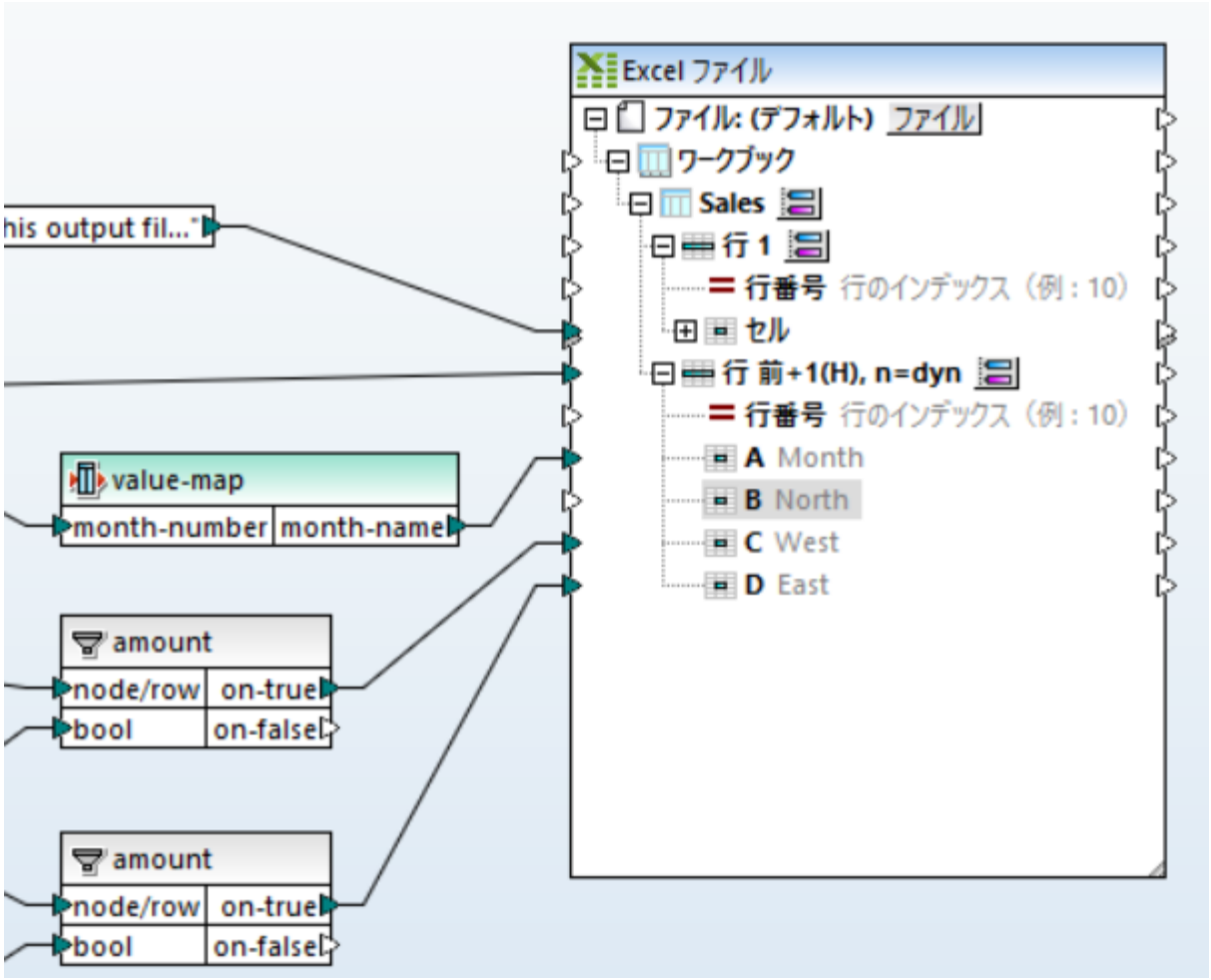
2. 任意で列に名前を入力します（例えば、「North」）。



3. 「OK」をクリックします。

新規の行（このサンプルでは、「North」）はマッピングに追加され、この列にマッピング接続を描くことが可能になります。すべての既存の接続は変更により影響を受けることはありません。

(C) 2015-2021 Altova GmbH



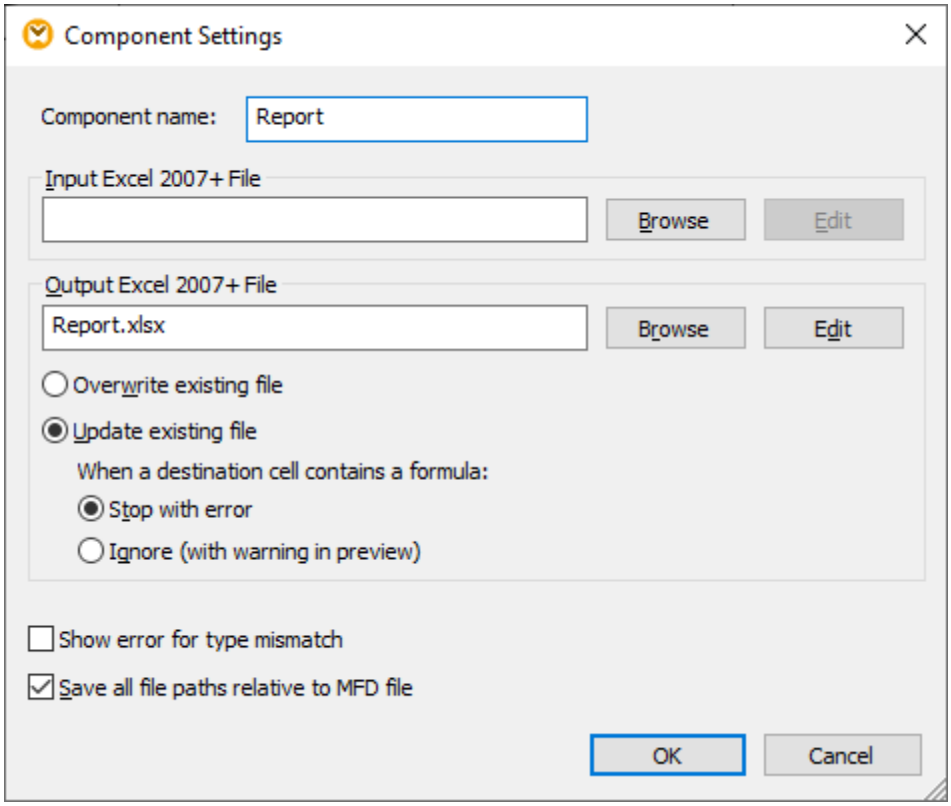
Sales_ to_ Excel.mfd マッピング（変更済み）

既存の行を同様に削除することができます: マッピング上のセルを右クリックし、「**セルを削除する**」をコンテキストメニューから選択します。

6.7.7 Excel 2007+ コンポーネント設定

マッピングエリアに"Excel 2007+"コンポーネントを追加すると、コンポーネント設定ダイアログボックスから適用することのできる設定を構成することができます。コンポーネント設定ダイアログボックスを以下の方法を用いて開くことができます:

- 「**コンポーネント**」メニューから、「**プロパティ**」をクリックします（このメニューアイテムは、コンポーネントを選択すると有効化されます）。
- コンポーネントヘッダーをダブルクリックします。
- コンポーネントヘッダーを右クリックして、「**プロパティ**」をクリックします。



コンポーネント設定ダイアログボックス

使用することのできる設定は以下の通りです。

コンポーネント名	コンポーネント名はコンポーネントを作成する際に自動的に生成されます。ですが、後に名前を変更することが可能です。 コンポーネント名はスペースを含むことができ、またフルストップ文字も含むことができます。コンポーネント名は、スラッシュ、バックスラッシュ、コロン、二重引用符、行頭と末尾スペースを含む場合があります。全般的に、コンポーネントの名前を変更する際、以下の影響を考慮してください： • マッピングを FlowForce Server にデプロイする場合、コンポーネント名は一意である必要があります。 • コマンドラインに入力できる文字のみを使用することが奨励されます。コマンドラインと Windows 内では、それぞれの文字で異なったエンコードを使用する場合があります。
Excel 2007+ ファイルの入力	MapForce がデータを読み込む Microsoft Excel ファイルを指定します。 MapForce がデータを読み込む Microsoft Excel ファイルを指定します。ソースコンポーネント内で、MapForce が、Excel ワークシートの名前とカラムを読み込むために使用します。

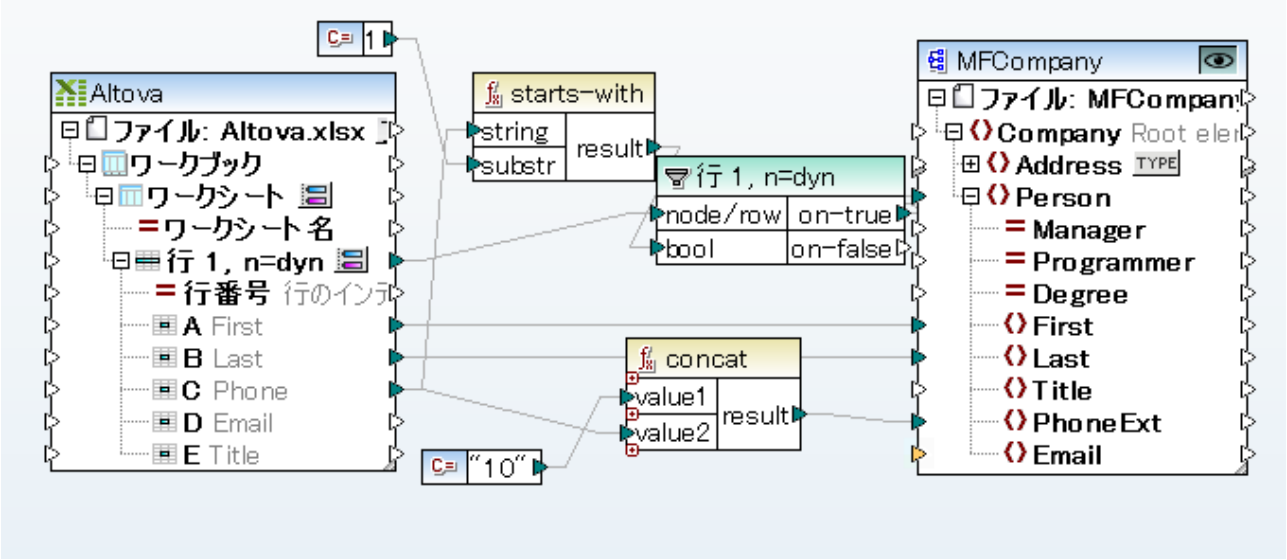
	XMLSpy 内のファイルを編集するには、「**編集**」をクリックします。Microsoft Excel 内のファイルを編集するには、「**編集**」をクリックします。
Excel 2007+ ファイルの出力	MapForce がデータを書き込む Microsoft Excel ファイルの名前/パスを指定します。このフィールドはターゲットコンポーネント内はとても重要です。 **XMLSpy 内のファイルを編集するには、**「参照」をクリックします。Microsoft Excel 内のファイルを編集するには、「**編集**」をクリックします。 出力ファイルを指定しない場合、MapForce は **xlsx-mapforce.xlsx** という名のファイルを生成します。デフォルトでは、このファイルは、異なるパスが構成されていない限り .mfd ファイルと同じフォルダー内に生成されます。
既存のファイルの上書き	このオプションが選択されると **Excel 2007+ 出力** テキストボックス内で指定されているファイルはマッピングが実行されると生成されます（上記参照）。生成された Excel ファイルには書式は含まれていません。 同じ名前を持つファイルが出力ディレクトリに既存の場合、マッピングの変換が実行されると上書きされます。 変換言語が C#、Java または BUILT-IN の場合このオプションはサポートされます。
既存のファイルの更新	このオプションが選択されると **Excel 2007+ 出力** テキストボックス内で指定されているファイルはマッピングが実行されると部分的に（上書きされず）更新されます（上記参照）。具体的には、既存ファイルからの目的とされるワークシート、行 とセルはマッピングからのデータと置き換えられます。既存のファイル内の書式はそのまま保管されます。 既存のファイルの更新はフォーミュラを含まないセルのためにのみ使用することができます。フォーミュラを含むセルに遭遇する都度「**エラーオプションで停止**」を選択してマッピングの実行を停止します。メッセージウィンドウに警告を表示しつつ（**プレビュー内に警告が表示される**）「**無視**」 オプションを選択してこのようなセルを無視します。 変換言語が BUILT-IN に設定されている場合のみこのオプションはサポートされます。
型の不一致のエラーを表示する	コンポーネント内で宣言されたデータ型と入力 .xlsx ファイル内のデータとの間に不一致が生じると、メッセージウィンドウ内でエラーメッセージが用事されることを有効化します。 例: この設定が有効化されていると仮定し、*セルの範囲を選択する*ダイアログボックス内で数値として列を宣言します。Excel ファイルがこの列にテキストデータを含む場合、この列を出力に接続し、出力を確認する場合にエラーメッセージが表示されます。
MFD ファイルに相対的なすべてのファイルパスを保存する	このオプションが有効化されていると、MapForce は、コンポーネント設定ダイアログボックスに表示されている MapForce Design (.mfd) ファイルの場所に対して相対的なファイルパスを保存します。コンポーネント上で相対パスを使用するを参照してください。

6.7.8 例:Excel 2007+ を XML にマッピングする

以下に示す例は<マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥Tutorial¥Excel-Mapping.mfd にて確認することができます。最初の2 つのマッピングがここでは取り上げられます。最初の2 つのマッピングがここでは取り上げられます。

このマッピングの目的:

- Excel のブック内から、(カラムC にある) PhoneExt が"1" から始まっているPerson だけをフィルタリングします。
- "10" をオリジナルの番号に追加/連結して、該当する従業員 (Person) の名前、苗字とともにMFCompany XML ファイルへ渡します



Excel-mapping.mfd (サンプル1)

マッピングは、以下のように構成されています:

- **Altova.xlsx** はソースExcel 2007+ ワークブックです。カラムA ならびにB にて名前と苗字がそれぞれ与えられており、カラムC には電話の内線番号が入力されています。
- ブックの各ワークシートに対して「**各シートを名前で表示**」オプションは無効になっています。これは「ワークブック」アイテム以下にあるワークシートのタイトルが、「ワークシート」となっていることからも確認できます。
- `starts-with` 関数により、(カラムC の) 内線番号が"1" から始まるかチェックされ、その結果が真 (true) である場合、そのレコードがフィルターコンポーネントから渡されます。
- **concat** 関数により、"10" という文字列が各内線番号の先頭に挿入され、PhoneExt アイテムへ渡されます。
- データの出力が行われる際には、ターゲットコンポーネントのMFCompany.xsd インスタンスファイルにフィルタリングされた従業員の情報が収められます。

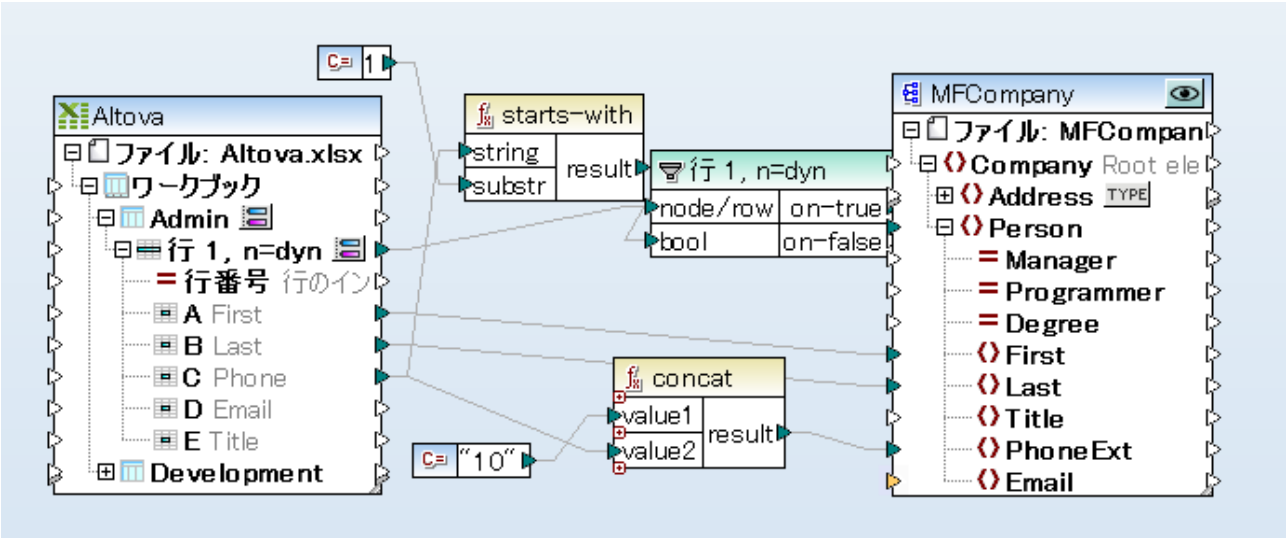
4人の社員がそれぞれの詳細と共にXML ファイルへマッピングされます。

(C) 2015-2021 Altova GmbH

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <Company xsi:schemaLocation="http://my-company.com/namespace
3    <Person>
4      <First>Steve</First>
5      <Last>Meier</Last>
6      <PhoneExt>10114</PhoneExt>
7    </Person>
8    <Person>
9      <First>Max</First>
10     <Last>Nafta</Last>
11     <PhoneExt>10122</PhoneExt>
12   </Person>
13   <Person>
14     <First>Carl</First>
15     <Last>Franken</Last>
16     <PhoneExt>10147</PhoneExt>
17   </Person>
18   <Person>
19     <First>Mark</First>
20     <Last>Redgreen</Last>
21     <PhoneExt>10152</PhoneExt>
22   </Person>
23 </Company>
```

2 番目のマッピングでは、「各ワークシートを名前で表示」オプションにより、ソースXLSX コンポーネントにある個々のワークシートが有効になった結果が表示されます。このマッピングは以下のように構成されています:

- **Admin** および **Development** ワークシートの両方がブックアイテム以下に表示されます。
- ターゲットコンポーネントへの接続は、**Admin** ワークシートからのみ行われています。



Excel-mapping.mfd (サンプル2)

マッピングの結果では2 人の従業員がXML ファイルへマッピングされます。

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <Company xsi:schemaLocation="http://my-company.com/namespace
3    <Person>
4      <First>Steve</First>
5      <Last>Meier</Last>
6      <PhoneExt>10114</PhoneExt>
7    </Person>
8    <Person>
9      <First>Max</First>
10     <Last>Nafta</Last>
11     <PhoneExt>10122</PhoneExt>
12   </Person>
13 </Company>
```

6.7.9 例: Excel 行を XML ファイルに変換する

Excel ファイルは異なる構造を持つ複数のワークシートから構成されている場合があります。これは列の数量と順序が異なっていることを意味します。一方 Excel ファイル内で全てのワークシートが同一の構造を持ち、ワークシートの名前とデータのみが異なる場合もあります。

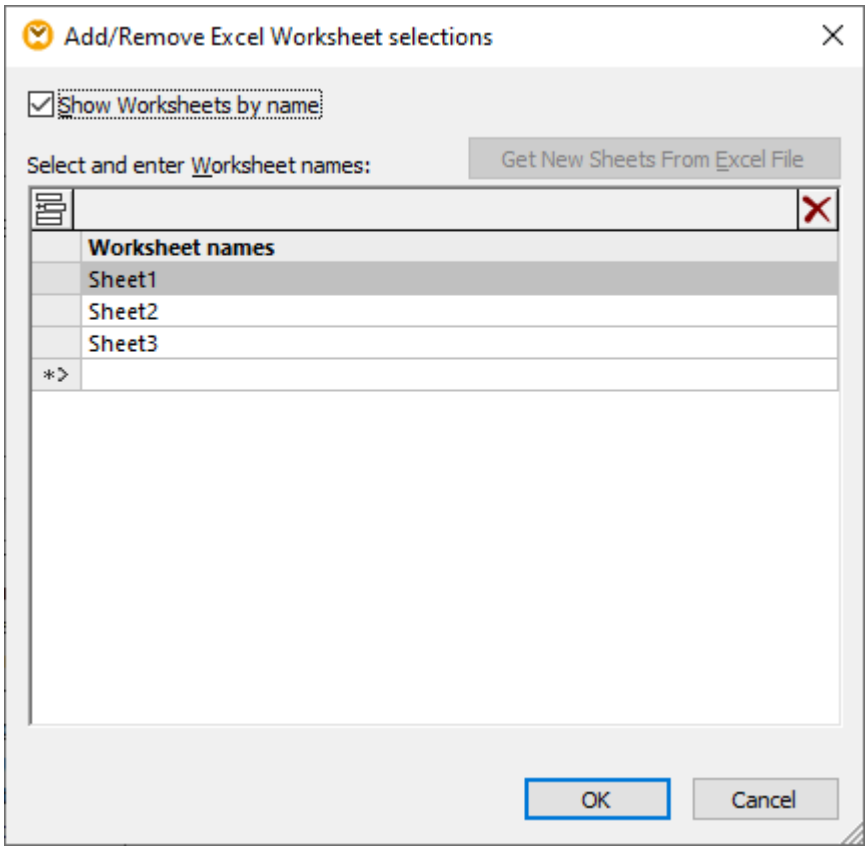
With MapForce を使用して上記のシナリオでマッピングを柔軟的に構成し Excel ワークシートからデータを抽出することができます。具体例:

1. 個別の Excel ワークシートをマッピング上で特殊の構成として表示することができます。このアプローチは各 Excel ワークシートが異なる構造を持つ場合役に立ちます。Excel ファイル内のワークシートのセルに基づくデータを柔軟的にマップすることができます。
2. ファイル内の Excel ワークシートを単一のワークシートとして扱うことができます。このアプローチはすべてのワークシートが同じ構造を持つ場合役に立ちます。マッピング上で多くの接続線を描く必要がない利点があります。各ワークシートの構造は同じと想定されているため、ワークシートの表示される個所に関わらず各列から一度のみ接続線を描きます。

Excel コンポーネントを構成するには上記の2つのアプローチがあります:

1. [挿入 | Excel 2007+ ファイル] メニューコマンドを使用して Excel ファイルをマッピングに追加します。Excel 2007+ ファイルをマッピングコンポーネントとして追加するを参照してください。
2. 最初のワークシートの横の ボタンをクリックします。次のダイアログボックスが表示されます:

(C) 2015-2021 Altova GmbH



3. 以下の一つを行います:

   a. 各ワークシートを個別に表示するには、チェックボックス **ワークシートを名前別に表示する** が選択されていることを確認してください。これがデフォルトのオプションです。
   b. 全てのワークシートを単一のマップ可能なコンポーネントとして扱うにはチェックボックス **ワークシートを名前別に表示する**をクリアします。

上記の2つのアプローチを表すデモマッピングデザインファイルは以下で見つけることができます: **<マイドキュメント>\Altova\MapForce2021\MapForceExamples\Tutorial\ConvertExcelRows.mfd**。このマッピングはExcel ファイルからデータを読み取る2つの異なるストラテジーを表しています。このマッピングは複数の関連しない出力を生成するため **プレビュー** ボタンが各ターゲットコンポーネントのタイトルバー内に表示されます。対応する出力ファイルをプレビューするために **出力** タブをクリックする前にこのボタンをクリックすることを忘れないでください。

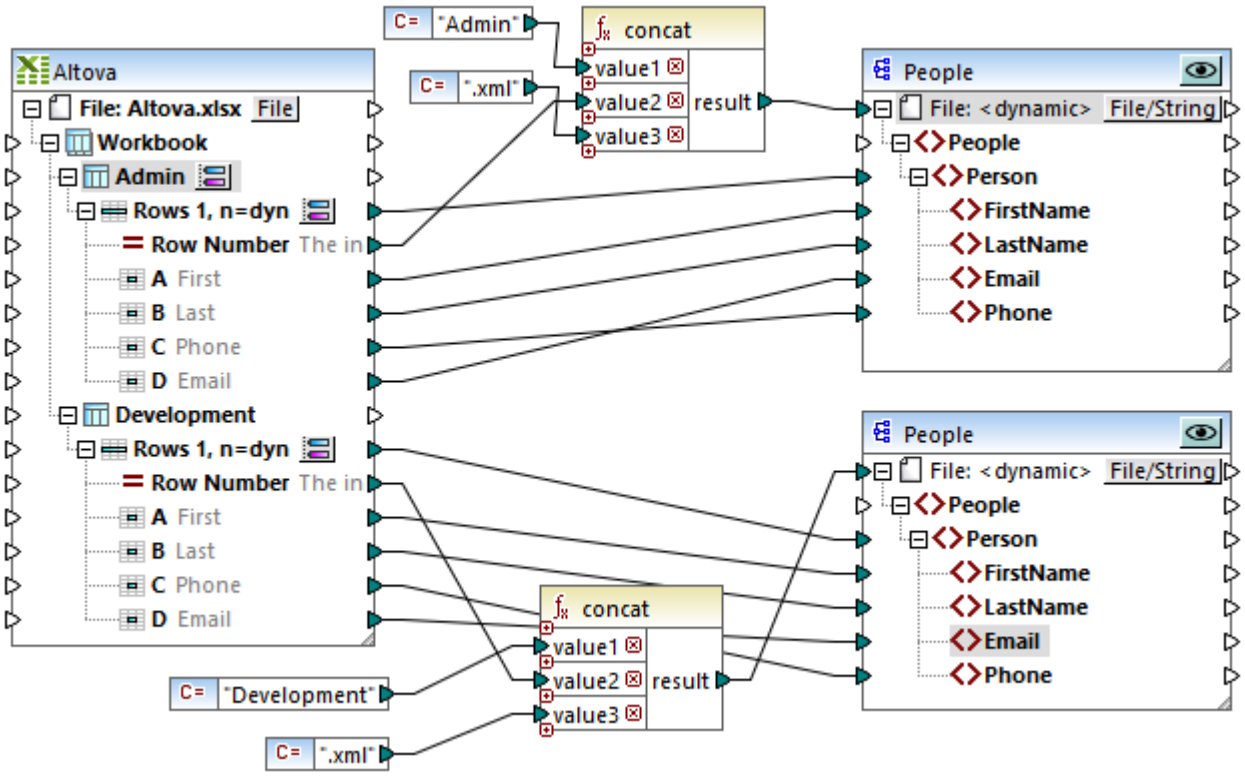
ソースExcel ファイルは **Altova.xlsx** と呼ばれ、2個のワークシートから構成されています: "Admin" と "Development"。各ワークシートの構造は同じで、各ワークシート内のデータのみが異なります(1個以上のデータアクセスストラテジーが可能な理由です)。

	A	B	C	D	E	F
1	Valentin	Bass	716	v.bass@nanonull.com	IT Manager	
2	Theo	Bone	331	t.bone@nanonull.com	Accounts Receivable	
3	Vernon	Callaby	582	v.callaby@nanonull.com	Office Manager	
4	Joe	Firstbread	621	j.firstbread@nanonull.com	Marketing Manager Europe	
5	Frank	Further	471	f.further@nanonull.com	Accounts Receivable	
6	Alex	Martin	778	a.martin@nanonull.com	IT Manager	
7	Loby	Matise	963	l.matise@nanonull.com	Accounting Manager	
8	Steve	Meier	114	s.meier@nanonull.com	Office Manager	
9	Max	Nafta	122	m.nafta@nanonull.com	PR & Marketing Manager US	
10	Susi	Sanna	753	s.sanna@nanonull.com	Art Director	
11						
12						

Admin Development

Ready 100%

マッピングの上の部分はワークシートを個別の構造として扱いExcel ファイルからのデータの読み取る方法を説明しています。**Admin** アイテムの横の ボタンをクリックし、チェックボックス **名前別にワークシートを表示する** が選択されていることを確認してください。両方のワークシート "Admin" と"Development" に影響します。**Workbook** の子としてソースコンポーネント上で明示的に表示されます。



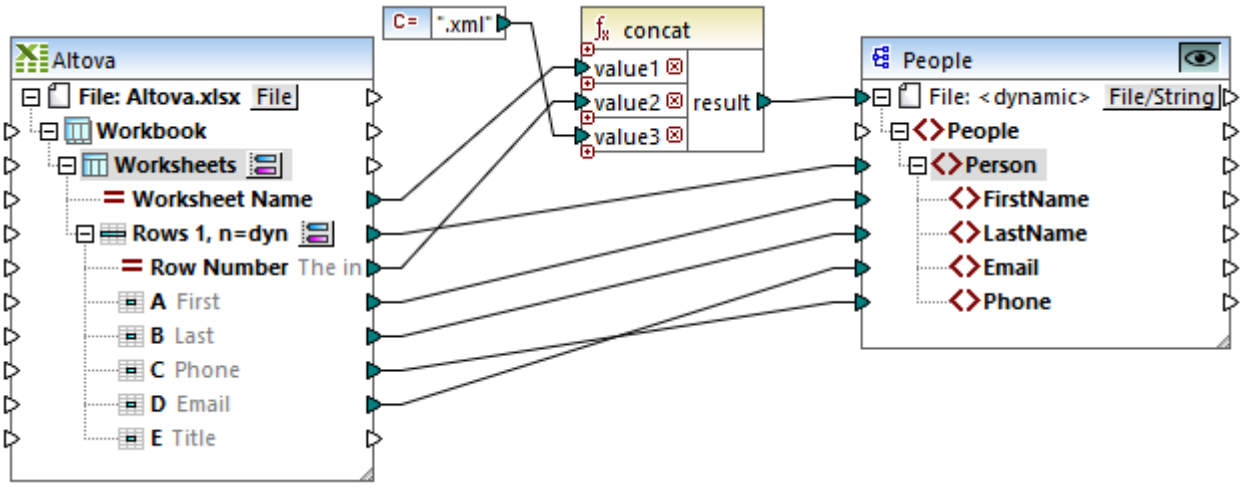
ConvertExcelRows.mfd (パート 1)

上で説明されているマッピングは両方のExcel ワークシートからXML ファイルにデータを変換します。個別のXML ファイルは各ソースワークシート内の各行から各個人のレコードのために生成される必要があります。ファイル名は部署名を識別し、数値 ID を持っている必要があります。例えば、ファイル**Admin1.xml** はワ"Admin" ークシート内の最初の個人（行）の詳細を保存します。このマッピングのゴールは以下のように達成することができます:

1. `concat` 関数各ターゲット XML のためにファイル名を生成します。（この順序で） 以下をジョインします: 各部署の名前、Excel 行番号、文字列 ".xml"。
2. **File:** への`concat` 関数の結果からの接続線。 **<dynamic>** ターゲットアイテムにはアイテムのシーケンス内の各アイテムのために新規のファイルがマッピングランタイムで生成される効果があります。このサンプルでは、入力されるシーケンスにはソースExcel ワークシートのすべての行が含まれています。新規 XML ファイルはExcel 各行のために生成されます。

サイドノートとしては、ファイル名を動的に生成するためにターゲットコンポーネントを構成するために**ファイル/文字列** ボタンをクリックし、オプション **マッピングにより提供される動的なファイル名を使用する**を選択します。詳細に関しては複数の入力または出力ファイルを動的に処理を参照してください。

ConvertExcelRows.mfd マッピングの2番目のパーツは以下で説明されています。マッピングのこの部分はパート1と同じ結果を達成しますが、更に簡素な方法で行われます。マッピングの接続線の数量は大幅に削減されます。



ConvertExcelRows.mfd (パート 2)

マッピングのロジックは以前に説明されているものと同じです。具体的には

1. 各ソースワークシート内の各行のために、ターゲット内で1 個の**Person** アイテムが作成されます。
2. `concat` 関数は各ターゲットファイルの名前を生成します。
3. ターゲットコンポーネントはファイル名を動的に生成するために切り替えられています。

唯一の違いはソースExcel ファイル内のすべてのワークシートが暗示的に反復されており、各ワークシートの構造を個別に表示しそれぞれのためにマッピング接続を描く必要がない点です。ソースExcel コンポーネントがすべてのワークシートを単一のものとして扱われるように構成されており、 **Worksheets** アイテムの横のボタンをクリックしてand 名前別にワークシートを表示する ボックスが選択されていないことを確認してください。

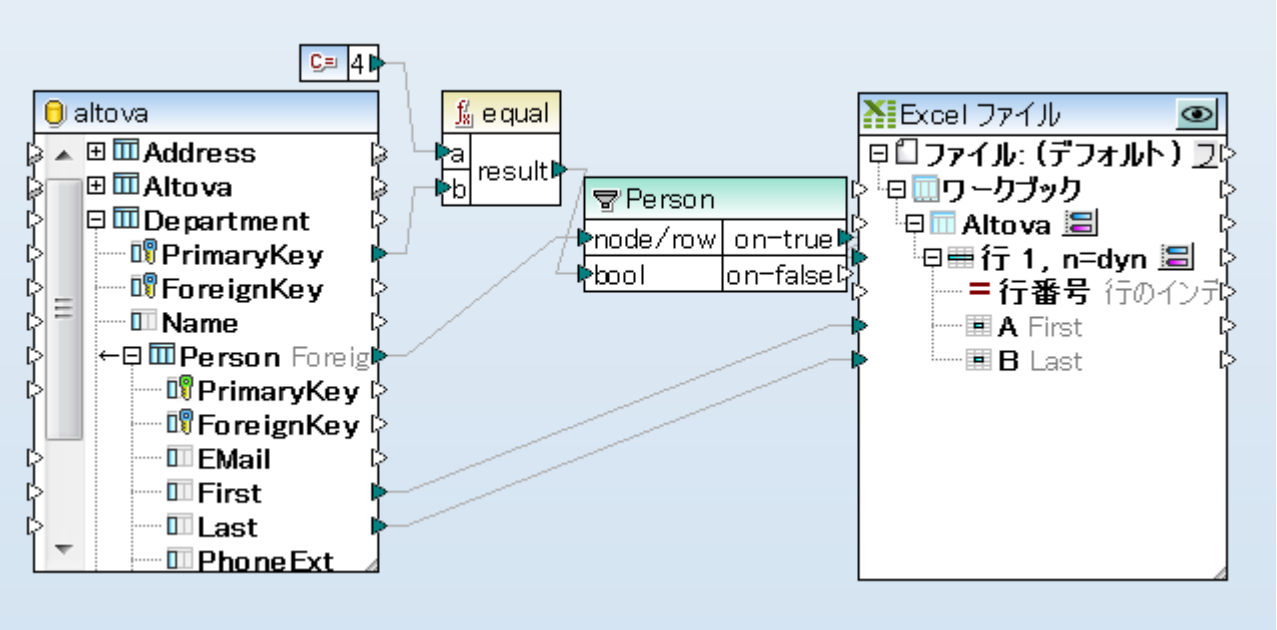
結論として、個別のExcel ワークシートからのデータのマッピングとすべてのワークシートを単一として扱うケースの違いがこのトピックで説明されています。2番目のアプローチはすべてのワークシートの構造が同じであることが必要とされます。このサンプルでは、各ワークシートの最初の列は個人の名が保存され、2番目の列は姓が保存されています。

6.7.10 例: データベースデータを Excel 2007+ にマッピング

以下に示す例は <マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥Tutorial¥ フォルダーに収められている **Excel-mapping.mfd** にて確認することができます。3 つあるうちの一番下のマッピングがここでは取り上げられます

マッピングの目的は、以下のとおりです:

- altova データベース内にある Person データのうち、Department の主キーが 4 (IT 部門) になっている従業員 (Person) をフィルタリングします
- IT 部門の従業員データを、「空の」Excel ファイルテンプレートへ挿入します。

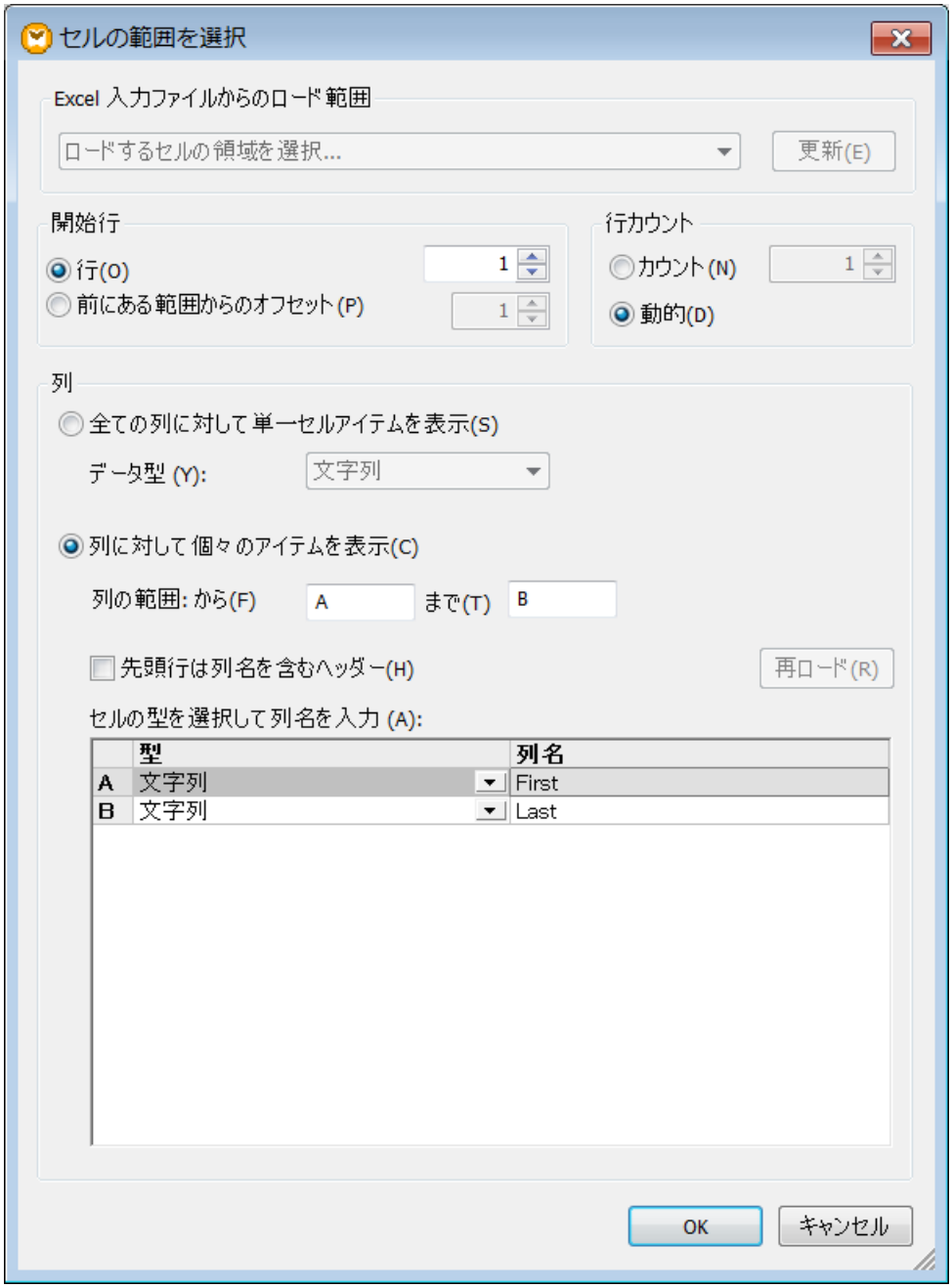


Excel-mapping.mfd (サンプル3)

マッピングは、以下のように構成されています:

1. 「**挿入 | データベース**」メニューコマンドを使用して、データベース「Altova」がマッピングエリアに<**マイドキュメント>\Altova\MapForce2021\MapForceExamples\Tutorial** フォルダーから追加されます。Microsoft Access データベースのためのウィザードに従います。
2. 「**挿入 | Excel 2007+ ファイル**」メニューコマンドを使用して、 Excel 2007+ が追加されます。サンプルファイルを与えるオプションはスキップされます。
3. ワークシート名の「Altova」は、シート1アイコン をクリックして、ワークシート名に「Altova」と入力することで定義することができます。
4. 行 1、n=dyn 範囲は ボタン をクリックすることにより、構成されます。セルの範囲のオプションは以下のように定義されています:

(C) 2015-2021 Altova GmbH



5. その他のオプションは以下のように定義されています:

   - **PrimaryKey** の値が、定数コンポーネントから与えられた"4"という値と`equal` 関数により比較されます。
   - フィルターコンポーネントは、ブールの値がtrue である場合(すなわち、 department 主キーが"4"の場合)に、値の受け渡しを行います。node/row パラメーターに接続されているソースコンポーネントの子アイテム/ノードのコンテンツがターゲットコンポーネントへ渡されます。
   - **on-true** アイテムはExcel ファイル。内の行 1, `n=dyn` アイテムに接続されています。

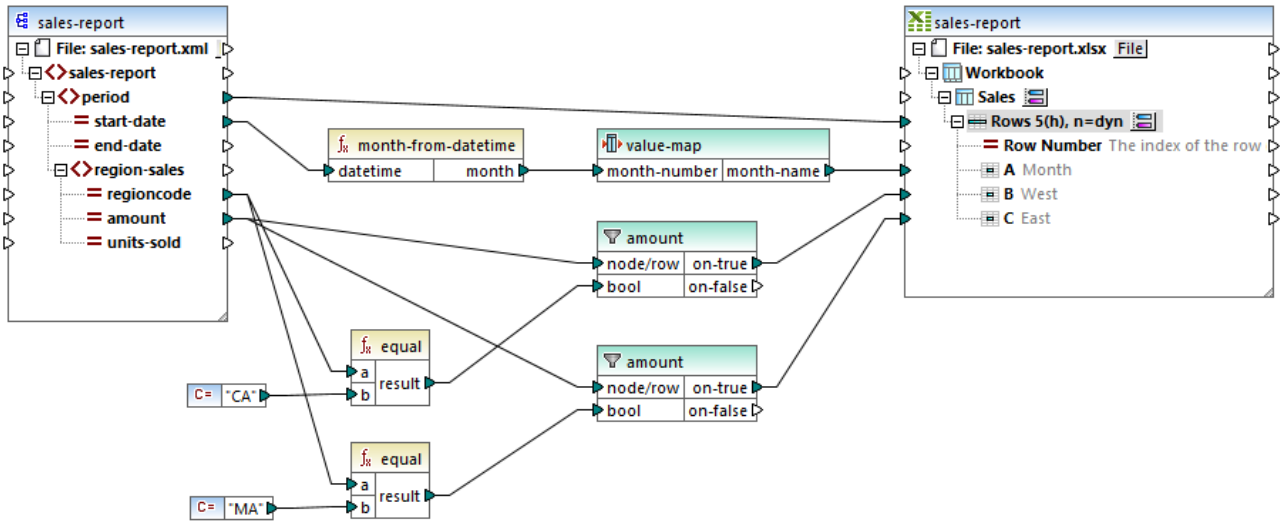
マッピングの結果は、IT 部門にいる4人の従業員名がExcel ブックに表示されます。

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

Altova

6.7.11 例: 既存の Excel ワークシートの更新

Excel シートを MapForce を使用して *生成する* 場合、生成されたシートは書式は含まずデータのみを含みます。(*生成* する代わりに既存の Excel ファイルを *更新* しファイル書式を保管することも可能です。これがこのサンプルの目的です。次のパスでこのサンプルで説明されているマッピングを見つけることができます: **<マイドキュメント>\Altova\MapForce2021\MapForceExamples\Sales_to_Excel.mfd**. 上記の通りにこのマッピングは XML 書式のファイルからデータを既存の Excel ファイルに読み取ります (**sales-report.xlsx**)。具体的には、列 A、B、および C 内の 5 番目の行から始まる "Sales" と呼ばれるシートを更新します。

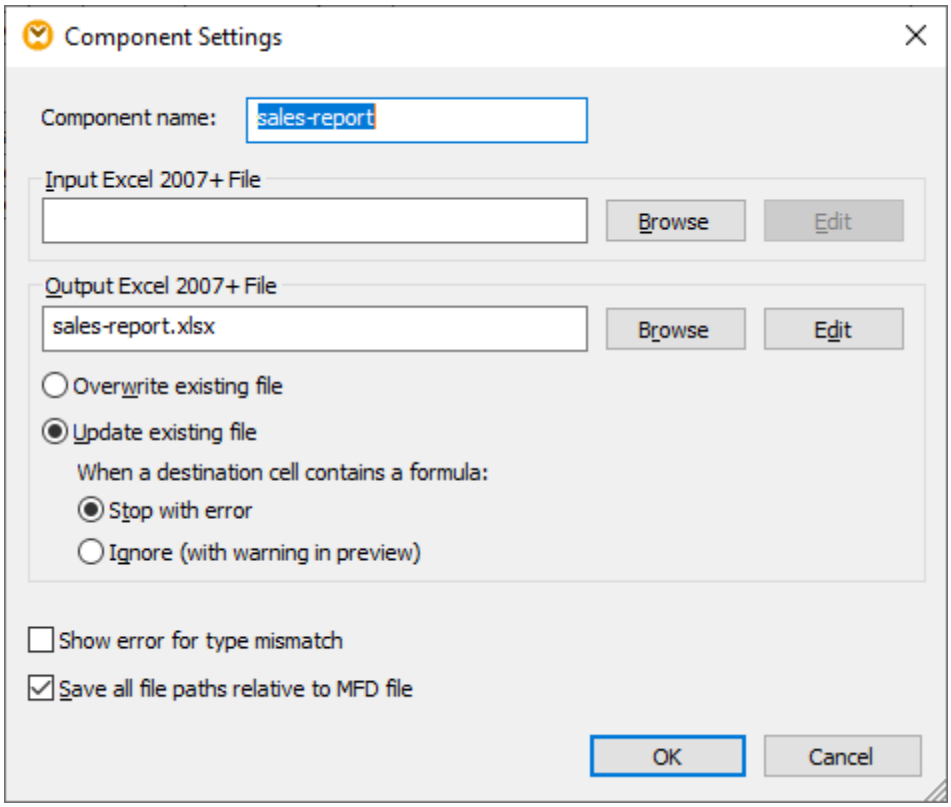


Sales_to_Excel.mfd

ターゲットコンポーネントは既存のファイルを以下のように更新するように構成されています:

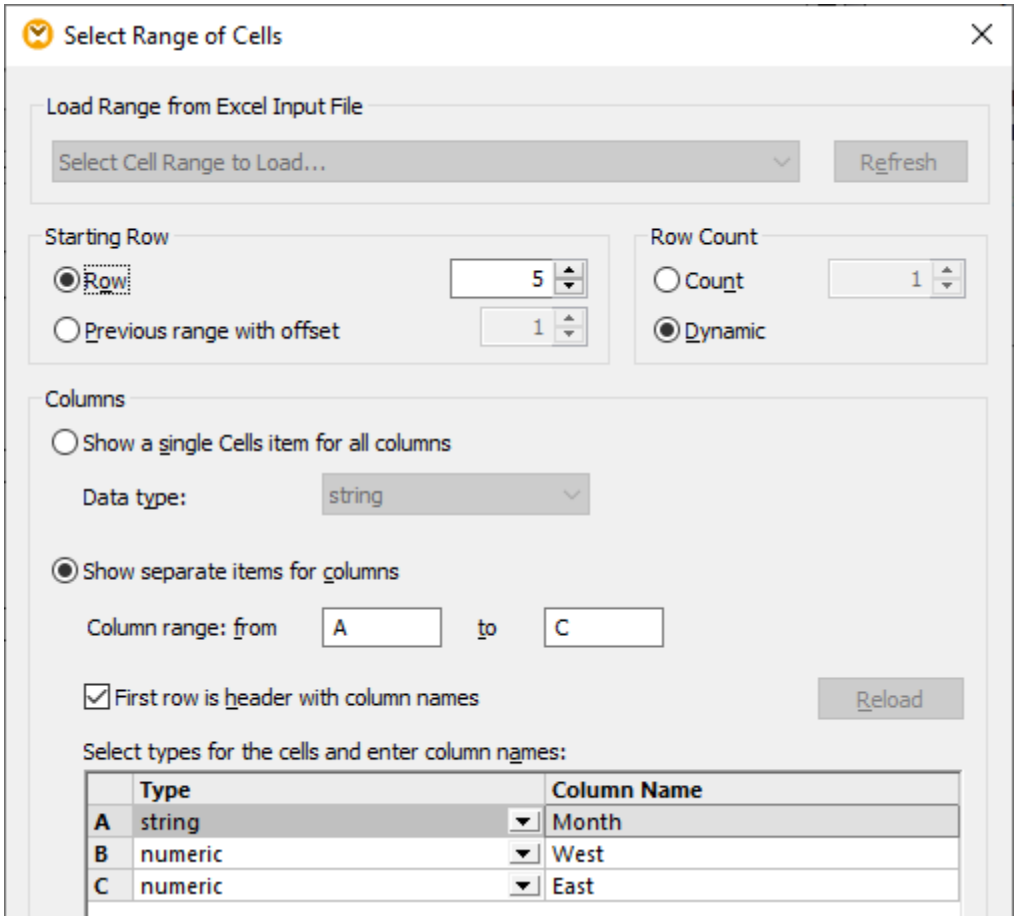
1. 変更言語を BUILT-IN に変更します。
2. ターゲットコンポーネントのタイトルバーを右クリックし、「**プロパティ**」を選択します。「**コンポーネントの設定**」ダイアログボックスが表示されます。
3. **Excel ファイル出力** テキストボックス内に既存の Excel ファイルへのパスを入力します。このサンプル内では、ファイルは同じフォルダー内に存在するため、パスは相対的であり「**全てのファイルパスを MFD ファイルに対して相対的に保存する**」が選択されます。
4. 「**既存のファイルの更新**」オプションを選択します。

(C) 2015-2021 Altova GmbH



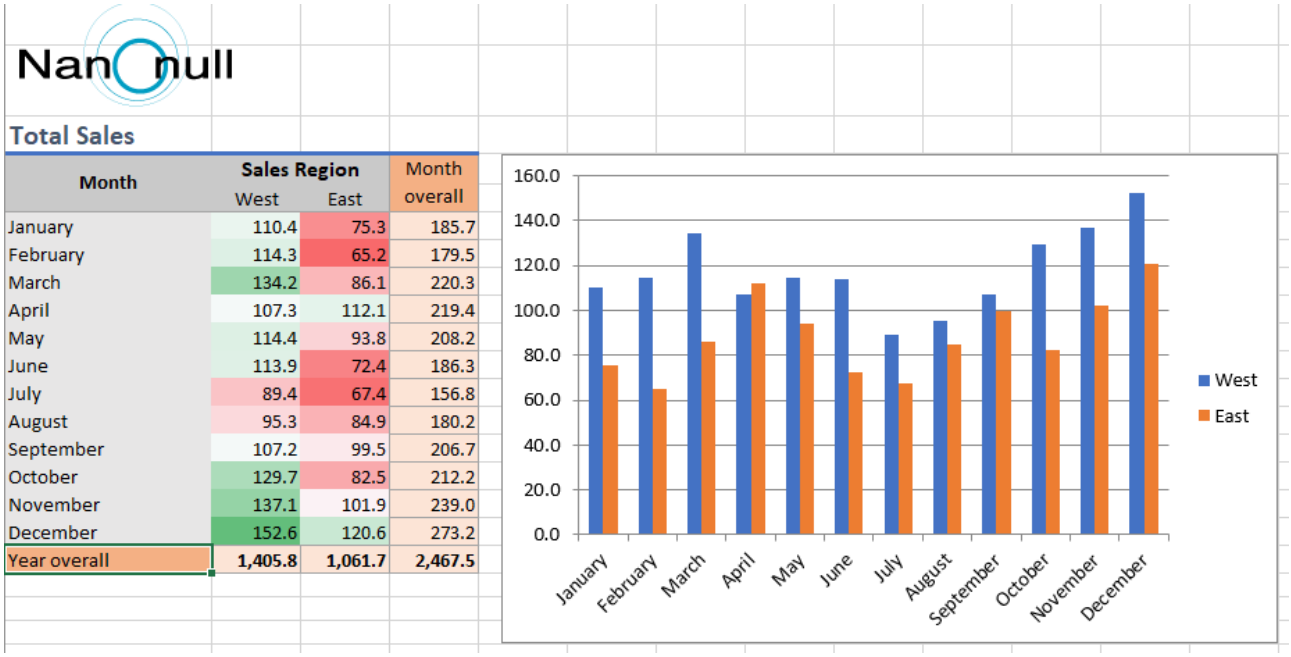
ターゲットワークシートは「Sales」ノードの横の ボタンをクリックして選択されます。(「Sales」がワークシートの名前です)。

「行」ノードの横の ボタンをクリックして書き込むセル/行の範囲を選択することができます。下の構成に従うとマッピングはワークシートを5番目の行からマップされたソースデータ内のアイテムの数だけ更新します。列 A、B、および C は簡単にマップできるように一意の名前が与えられます。



「出力」タブをクリックしてマッピングの結果をプレビューすると、既存のワークシートは更新され、既存の書式が保管されます。(チャートにより使用されたソースセルが値を変更するため)結果既存のチャートも更新されます

(C) 2015-2021 Altova GmbH



Total Sales

Month	Sales Region		Month overall
	West	East	
January	110.4	75.3	185.7
February	114.3	65.2	179.5
March	134.2	86.1	220.3
April	107.3	112.1	219.4
May	114.4	93.8	208.2
June	113.9	72.4	186.3
July	89.4	67.4	156.8
August	95.3	84.9	180.2
September	107.2	99.5	206.7
October	129.7	82.5	212.2
November	137.1	101.9	239.0
December	152.6	120.6	273.2
Year overall	1,405.8	1,061.7	2,467.5

マッピングの結果

マッピングのプレビューは変更をExcel ファイルに保存しません。「**出力**」ペイン内に表示されたファイルをディスクに保存するにはメニューコマンド「**出力 | 出力ファイルの保存**」または「**生成された出力を保存**」 ツールバーボタンをクリックします。代替として、オプション「**ツール | オプション | 一般** から「**最終出力ファイルに直接書き込む**」を有効化することもできます。更なる確認をリクエストすることなく既存のファイルが上書きされるため後者のオプションの選択は注意してください。

このようなマッピングの作動をボタンをクリックすることなくを自動化するには マッピングとMapForce の自動化 で説明されるとおりマッピングを準備し MapForce Server にデプロイすることができます。

6.8 XBRL

XBRL (Extensible Business Reporting Language) は、ビジネスの情報の交換を可能にするオープンな国際標準です。XBRL は、政府、金融機関、ビジネス、で金融および規制の報告のために使用されます。XBRL は、グローバルな有利を目的としない XBRL インターナショナル(https://www.xbrl.org)という協会により管理されています。

企業または機関は(例えば、XBRL 検証、XBRL データの作成、新規XBRL タクソノミの作成、XBRL データを他のフォーマットにエクスポートするなど) XBRL に関して異なるニーズを持っている可能性があります。多種のAltova 製品により XBRL に関連するタスクを達成することが可能になりました

- *XMLSpy* は、新規タクソノミを編集または作成し、XSLT 変換ファイルをベースにした XBRL レポートを生成します(XSLT 変換ファイルは StyleVision により作成することができます)
- *StyleVision* は、タクソノミ スタイルシート/テンプレートを作成し、XBRL レポートを生成します。
- *MapForce* は、XBRL インスタンスファイルからのデータをマップします。XBRL データを MapForce (Microsoft Excel 2007 と以降、データベース、またはCSV ファイルなど) がサポートするフォーマットからマップ、またはフォーマットにマップすることができます。例えば、既存のXBRL インスタンスドキュメントから特定のデータをフィルターすることができます。または、XBRL インスタンスをデータベースから抽出されたデータから作成することができます。
- *RaptorXML+XBRL Server* は、XBRL インスタンスとタクソノミを検証するクロスプラットフォーム XBRL プロセッサーです。
- *Altova の Excel のためのアドイン* により、Microsoft Excel を使用して XBRL レポートデータを準備し検証することができるようになりました。

XBRL の完全な紹介はこのドキュメントのスコープ外です。以下の用語は XBRL と共に頻繁に使用されます:

- *XBRL タクソノミ*。タクソノミは XBRL レポート内で発生するすべてのコンセプトの辞書に類似したものです。またコンセプトを階層的に整理するルールを定義します。
- *XBRL インスタンス*。XBRL インスタンスはレポートされるいわゆる[ファクト]を含むドキュメントです。.コンセプトの実際の定義と階層的なリレーションシップはタクソノミにより提供されます。

MapForce 内の XBRL インスタンスファイルからデータをマップ、またはインスタンスファイルにデータをマップするために、基となる XBRL タクソノミが必要になります。タクソノミは、機関から提供される、またはXBRL データ が提出されるエンティティである(通常 .zip 形式の) ファイルアーカイブから構成されています。次に挙げられるタクソノミの例は、MapForce を使用したデータマッピングのために使用されています:

- 財務会計基準審議会 (http://www.fasb.org により公開されている US-GAAP (一般に認められた会計原則)
- IFRS (国際財務報告基準) (http://www.ifrs.org) により公開されている IFRS (国際財務報告基準)
- 欧州銀行監督局 http://www.eba.europa.eu により公開されている COREP /FINREP (共通報告と財務報告)

XBRL タクソノミを XBRL-有効化済みの Altova 製品に XBRL タクソノミ マネジャー からインストールまたはアンインストールすることができます。

XBRL ドキュメント内で URI を解決するために MapForce はカローカルシステムに URI をマップするカタログメカニズムを使用します。これは大半タクソノミのサイズおよび数千のファイル含むファクトのためです。インターネット上でタクソノミファイルにアクセスすることは発行元の期間がそのようにサーブする場合でもとても時間がかかります。からログメカニズムは使用しているコンピューターにインストールされている XBRL タクソノミ パッケージに依存していることご注意してください。デフォルトで XBRL タクソノミ パッケージがインストールされています。結果、XBRL コンポーネントを含むマッピングを開く場合、MapForce は必要な XBRL タクソノミ パッケージをインストールすることを問うダイアログボックスを表示する場合があります。更に、インストールされているすべての XBRL タクソノミを XBRL タクソノミ マネジャー からビューし管理することができます。

MapForce は、次の XBRL に関連した標準をサポートします:

- XBRL 2.1
- XBRL Dimensions 1.0
- XBRL Table Linkbase 1.0 (XBRL テーブルと作業するを参照してください)。

- XBRL Taxonomy Packages 1.0 (XBRL タクソノミ パッケージを参照してください)
- Transformation Registry 3 を含むインライン XBRL 1.0 と1.1 (データの読み込みのみに関する情報は、インライン XBRL からデータを読み込むを参照してください)。

変換

このドキュメントは以下の仕様により定義されている XBRL 用語について言及されています:

仕様	URL
XBRL 仕様 2.1	https://www.xbrl.org/Specification/XBRL-2.1/REC-2003-12-31/XBRL-2.1-REC-2003-12-31+corrected-errata-2013-02-20.html
XBRL ディメンション 仕様 1.0	https://www.xbrl.org/specification/dimensions/rec-2012-01-25/dimensions-rec-2006-09-18+corrected-errata-2012-01-25-clean.html
Table Linkbase 仕様 1.0	https://www.xbrl.org/Specification/table-linkbase/REC-2014-03-18/table-linkbase-REC-2014-03-18.html
XBRL タクソノミ パッケージ 1.0	https://www.xbrl.org/Specification/taxonomy-package/REC-2016-04-19/taxonomy-package-REC-2016-04-19.html
インライン XBRL	https://specifications.xbrl.org/spec-group-index-inline-xbrl.html

このドキュメントでは、仕様列により示された仕様について参照しています。更に § 文字は仕様内の特定のセクションの番号を表すために使用されています。例: XBRL Table Linkbase 1.0 仕様のセクション 5.4 への参照は、以下の通りです:

テーブル (Table Linkbase 仕様 1.0、 §5.1) は、XBRL コンポーネント内の *アイコン により示されています。*

6.8.1 XBRL ファイルをマッピングコンポーネントとして追加する

XBRL ドキュメントにデータをマップ、または XBRL ドキュメントにデータをマップする前に、XBRL ドキュメントは、マッピングエリアにマッピングコンポーネントとして追加される必要があります。XBRL ドキュメントをマッピング コンポーネントとして追加するには以下の2 つの方法があります:

- フラット XML ドキュメントとして
- 階層的構造を持つ XBRL ドキュメントとして

最初のオプションは、を、ビジネスロジックコンセプトまたは特定の XBRL 階層構造無しに、フラット XML ファイルとして挿入します。このようなコンポーネントは、次元の処理や、自動的な XBRL コンテキストの生成が存在しないシンプルなマッピングにとても役にたちます。

2番目のオプションは、XBRL タクソノミを挿入し、そのコンテンツを XBRL に適する階層的な方法で表示します。この場合、MapForce は、検知可能なタクソノミ セット (DTS) のレファレンスを解決し、自動的に派生した XBRL 構造をビルドします。これは、XBRL マッピング コンポーネントが、XBRL 特有のアイテム (XBRL ファイル内の実際のノード または DTS アイテムから派生した "抽象的" ノード) を含むことを意味します 例: 次の階層的構造は、`xlink:from` と `xlink:to` プレゼンテーションアークから派生し、プレゼンテーション リンクベースとして振る舞います。("リンクベース" という用語は、XLink 仕様内で定義されている用語と同じ意味があります (http://www.w3.org/TR/xlink11/を参照してください)。具体的には、リンクまたはリンクデータベースのコレクションを示します。XLink リンクベース はタクソノミ内で定義された追加情報を与えるために、XBRL に拡張されます。)

このサンプルは、プレーン XML では無くマッピングオペレーションに適切な、概念的な方法でデータを表示するために、MapForce のタクソノミの基本的な解決方法を説明しています。かどのようにタクソノミを解決するかを説明する基本的な表示です。プレゼンテーション リンクベースに加え、MapForce は、Table Linkbase を含む他のリンクベースから作成された構造も表示することができます（ XBRL テーブルと作業するを参照してください）。

コンベンショナルなグラフィックの表示を含む XBRL コンポーネントに表示されることのできるアイテムに関する詳細は、 XBRL コンポーネント アイテムについてを参照してください。

XBRL ドキュメントをフラット XML として開く:

- 「挿入」メニューから「 XML/スキーマファイル」をクリックします。
- XBRL インスタンスファイルを選択し、「開く」を選択します。インスタンスはスキーマを参照する必要があることに注意してください。それ以外の場合、MapForce は生成するようにプロンプトします。

自動的なコンテキストの処理を使用して XBRL ドキュメントを開く

1. 「挿入」メニューから「 XBRL ドキュメントの挿入」をクリックします。
2. タクソノミファイル（*.xsd）または XBRL インスタンスファイル（*.xbrl または *.xml）を開きます。タクソノミを選択すると、有効なインスタンスファイルと選択するためのダイアログが表示されます。
3. この XBRL コンポーネントをソースインスタンスとして使用する場合、「参照」をクリックして、XBRL インスタンスファイルを選択します。

> XBRL タクソノミを開くと、ダイアログボックスが開き、XBRL コンポーネント内で表示される構造ビューを選択するようにプロンプトされます。どの構造ビューを選択するのか不定の場合は、デフォルトのオプションを選択してください。デフォルトでは、MapForce は自動的にロードされた Discoverable Taxonomy Set（DTS）からデータを分析してオプションを選択します。構造ビューを後に必要に応じて変更することができます（ 構造ビューの選択も参照）。

6.8.2 インライン XBRL からデータを読み込む

XBRL と作業する機関の一部は、 XBRL の代替（または、追加）として、インライン XBRL フォーマットの情報を提供、または送信の受付、をする場合があります。インライン XBRL、または iXBRL は、マシンにより読み取ることのできる HTML ドキュメントに埋め込まれた XBRL コンテンツを参照します。これにより、を W eb ブラウザー内で人間が判読できる XBRL データを表示することができます。iXBRL に関連する仕様のリストに関しては、次を参照してください： https://specifications.xbrl.org/spec-group-index-inline-xbrl.html。

MapForce では、（.htm、.html、または .xhtml ファイルなどの）インライン XBRL を含むファイルをマッピングにデータソースとして追加することができます。すなわち、MapForce は、サポートされるターゲットのフォーマットにマッピングするために、インライン XBRL インスタンスファイルからデータを読み取ることができます。インライン XBRL インスタンスへのデータの書き込みは、サポートされていないことに注意してください。

必要条件

- マッピング 変換言語は BUILT-IN に設定されている必要があります（次を参照してください： 変換言語の選択）。他の言語はサポートされていません。BUILT-IN 変換言語を使用して、マッピングを MapForce で直接確認するか、サーバー実行可能ファイルにコンパイルして、MapForce Server で実行して確認することができます（次を参照してください： MapForce Server 実行可能ファイルにマッピングをコンパイルする）
- インライン XBRL ファイルのタクソノミは、URL アドレス、またはローカルファイルへの参照である場合があります。後者の場合、マッピングが MapForce Server により作動する場合、以下の追加条件に関して注意してください：
  - タクソノミファイルは、.mfx（マッピング実行）ファイルと同じディレクトリに存在する必要があります。

(C) 2015-2021 Altova GmbH

  - インライン XBRL ファイルは、相対的パスを使用するタクソノミファイルを指す必要があります、例: `<link:schemaRef xlink:type="simple" xlink:href="taxonomy.xsd" />`。
  - コンポーネント設定 ダイアログボックスは、相対的パスを使用するタクソノミファイルを指す必要があります（例えば、"taxonomy.xsd"）（次を参照: XBRL コンポーネント設定）。
- ソース XBRL ファイルは、単一の XBRL インスタンスを含む必要があり、複数のインスタンスは、サポートされていません。

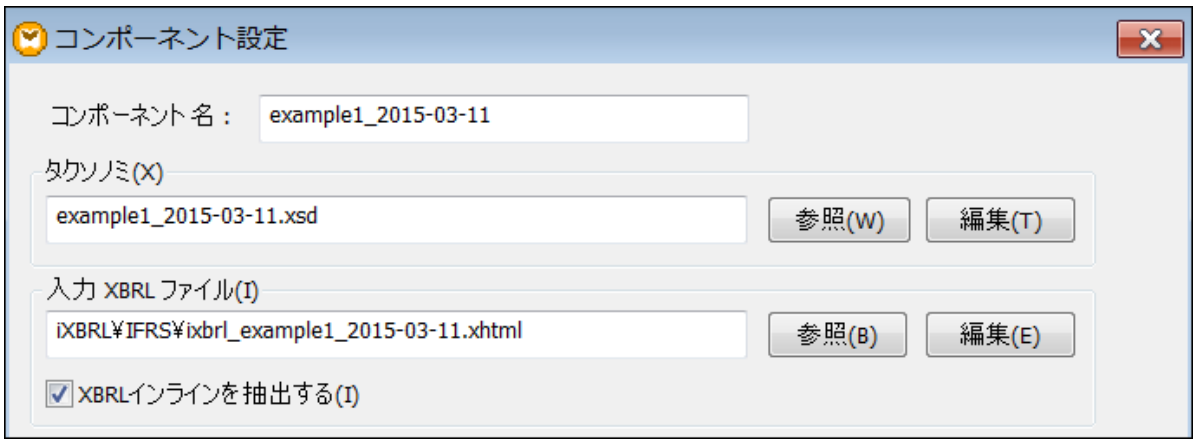
インライン XBRL を含むインスタンスファイルをマッピングに追加する

1. 「挿入」メニューから、「XBRL ドキュメント」をクリックします。
2. 次のうちの 1 つを行います:
   a. XBRL タクソノミがある場合、XBRL タクソノミ（通常、.xsd 拡張子を持つファイル）を参照します。これは、インライン XBRL インスタンス ファイルにとって妥当なタクソノミである必要があります。URL をタクソノミとして指定します。これをおこなうには、ブラウザーダイアログボックス上の「URL に切り替える」ボタンをクリックします（URL からコンポーネントを追加する も参照してください）。
   b. XBRL タクソノミ ファイルまたは URL が無い場合、インライン XBRL インスタンス ファイルを代わりに参照してください。この場合、MapForce は、自動的に対応する XBRL タクソノミを決定しロードします。
3. インスタンスを提供するようにプロンプトされると、インライン XBRL を含む HTML ファイルを参照します（通常、.html、.htm または .xhtml 拡張子を持つファイル）。
4. 「入力 XBRL ファイル」の横のインライン XBRL を含むファイルを参照します（通常の場合、ファイルには .html または .xhtml 拡張子が付いています）。
5. 「OK」をクリックします。

この時点では、インライン XBRL コンテンツの構成は、コンポーネント内で使用することができます。検証エラーが存在する場合、メッセージウィンドウに表示されます。それ以外の場合、通常の XBRL コンポーネントと同様にインライン XBRL コンポーネントから XML、テキスト、データベース などのターゲット構造に接続を描きます。

「インライン XBRL の抽出」チェックボックス

XBRL コンポーネント 設定 ダイアログボックス上の「インライン XBRL の抽出」チェックボックスが選択されると、.html または .xhtml ファイルが、インライン XBRL として扱われる入力として指定されます。この結果、MapForce は、インライン XBRL コンテンツを、入力ファイルから抽出し、ダイアログボックスが閉じられると検証します。このチェックボックスは、マッピングにインライン XBRL インスタンス ファイルを追加すると通常自動的に選択されます。オプションが有効化されているかを確認するには、マッピング上の XBRL コンポーネント ヘッダーをダブルクリックします（または、代わりに、XBRL コンポーネント ヘッダーを右クリックして、「プロパティ」を選択します）。



6.8.3 XBRL コンポーネントアイテムについて

以下のテーブルは、MapForce の一部である XBRL コンポーネント アイテム(ノード)の概要です。ノードの一部は、対応する構造ビューが選択されている場合のみ使用することができます (構造ビューの選択を参照してください)。例: の場合、プレゼンテーション定義リンクベースからのみデータを表示することが選択されている場合、テーブルリンクベースに特有のノードは、表示されません。

コンポーネントアイテム	アイコン	説明
抽象アイテム		抽象アイテムは関連するファクトを編成するために使用されます。表示リンクベース、または定義リンクベース内にあるドメインメンバーネットワークにて定義することができます。これらが XBRL インスタンスファイルにて実際に記述されることはありませんが、直感的な方法でファクトのグループ化を行うことができます。
ブレークダウン		テーブルリンクベース 仕様 1.0、 §5.4 内で定義されているとおり。
コンテキスト		コンテキストノードは、関連する全てのビジネスファクトを収めるコンテナーです。ハイパーキューブ内部にあるコンテキストノードにより、コンテキスト要素や次元が自動的に管理されます。XBRL ルート要素の子である xbrli:context ノードは、次元を手動で管理する際に使用されます。
次元		次元 (XBRL Dimensions 仕様 1.0, §2.5) はビジネスファクトのためのコンテキスト情報を構造するために使用されます。次元はハイパーキューブ内のタクソノミにより定義されます。
明示的なメンバー		明示的なメンバーは QName 値の次列挙により定義される次元メンバーです (コンポーネント内の次元の表示も参照)。
明示的なメンバーの値		明示的なメンバーの値
ファクト		ファクトとは(XBRL 仕様 2.1, §1.4) XBRL アイテムの値のことです。以下の型があります: • 通貨アイテム( ) • 文字列アイテム( ) • 数値アイテム( ) • 一般アイテム( ) • 株式アイテム( )
フットノート		フットノートでは、概念に対してテキスト情報を新たに加えることができます。
ハイパーキューブ		ハイパーキューブでは(XBRL Dimensions 仕様 1.0, §2.2) 、定義リンクベースや表示リンクベースから得られた情報を使って、階層構造の次元、概念、そして関連する XBRL 概念が構成されます。XBRL ハイパーキューブと作業する を参照してください。
ルート要素		xbrli:xbrl 要素が XBRL コンポーネントのインスタンスルート要素となります。
ルールノード		テーブルリンクベース 仕様 1.0、 §6.6 内で定義されているとおり。
構造ノード		テーブルリンクベース 仕様 1.0、 §5.5 内で定義されているとおり。

(C) 2015-2021 Altova GmbH

コンポーネントアイテム	アイコン	説明
テーブル		テーブルリンクベース仕様 1.0、§5.1 内で定義されているとおり。MapForce は、テーブル構造を階層に表示します（XBRL テーブルと作業するを参照してください）。
テーブルセット		テーブルリンクベース仕様 1.0、§5.2 内で定義されているとおり。
タプル		Tuples（XBRL 仕様 2.1，§1.4）タプルはファクトやその他のタプルをメンバーとして含んだ複合型の要素です
型指定されたメンバー		型指定されたメンバーは XML スキーマ要素により定義されます（http://www.xbrl.org/specification/dimensions/rec-2012-01-25/dimensions-rec-2006-09-18+corrected-errata-2012-01-25-clean.html#sec-typed-dimensions を参照してください）。
ユニット		ユニット要素 xbrli:unit にはXBRL アイテムが参照するユニットが含まれます。データのマッピングを行う際には、xbrl:unit 要素に対して値を定義する必要があります。例えば、米ドルに対して UnitID = usd ならびに Measure = iso4217:USD という値が使用されます。XBRL デフォルトと作業するを参照してください。
表示		表示は、XBRL タクソノミの定義とプレゼンテーションの拡張されたリンクロールを示します。

アイコン は、ジェネリックで、XBRL ルート要素を含む多種のノードの型に使用されます。

更に、以下のグラフィカルユーザーインターフェイス要素は、コンテキストで必要とされる場合、XBRL コンポーネントで表示されます。

GUI 要素	アイコン	説明
追加情報		このアイコンは、エラーなどのメッセージに関連したアイテムと共に表示されます。このアイコンをクリックして、メッセージペインに追加情報を表示することができます。
エラー		このアイコンは、エラーメッセージを示します。
コンテキストメニューの表示		このアイコンは、このセクションで使用することのできる追加コンテキストメニューオプションと共に表示されます。
未知		このアイコンは、無効な名前空間への参照、または参照されたアイテムがタクソノミ内に存在しない場合、MapForce により解決できないアイテムと共に表示されます。

6.8.4 構造ビューの選択

XBRL ドキュメントをマッピングエリアに追加する際、コンポーネント上で XBRL 構造ビューが表示されるように、（XBRL ファイルをマッピングコンポーネントとして追加するを参照してください）を選択することができます。または後に選択することもできます。使用することのできる構造ビューは、以下のとおりです。

XBRL コンポーネント内での構造ビューの表示の選択:

1. XBRL コンポーネントのルート要素から「**コンテキストメニューの表示**」( ) ボタンクリックして、「**構造 ビュー**」を選択します。

2. 1つまたは複数の構造ビューを選択し、「OK」をクリックします。

使用することのできる構造ビューは、以下のとおりです。

構造の表示	説明
テーブルリンクベースからのテーブル	タクソノミのテーブルリンクベース内で定義されたテーブルを表示します。タクソノミがテーブル定義を含まない場合、このオプションは無効化されます。 このビューは任意です。続行するには、このチェックボックスを選択するか、または「**プレゼンテーションおよび定義リンクベースからのビュー**」チェックボックスを選択します。
プレゼンテーションおよび定義リンクベースからのビュー	プレゼンテーション定義リンクベースからのデータを表示します。プレゼンテーションリンクベースはハイパーキューブと次元を持つ拡張されたリンクロールを含む階層構造と定義リンクベースを含みます。 このビューは任意です。続行するには、このチェックボックスを選択するか、または「**テーブルリンクベースからのテーブル**」チェックボックスを選択します。
全てのコンセプト (コンテキストの管理付き)	階層構造内の追加ノード"全てのコンセプト" を表示します。2つのデフォルトのディメンションの識別子とピリオドを使用したハイパーキューブ内でレポートされるかに関わらず、このノードは、タクソノミのすべてのコンセプトのマッピングを有効化する、ハイパーキューブ"無次元"を含みます。自動的なコンテキストの処理は、タクソノミのすべてのXBRL 概念を子として含む、**コンテキスト**ノードにより提供されます。抽象アイテムは、表示されていません。

(C) 2015-2021 Altova GmbH

構造の表示	説明
	このビューは任意で表示されます。
全てのコンセプト（未フォーマット）	自動的なコンテキストの処理、または次元の処理のサポートを持たないのすべてのXBRL インスタンスのファクトへのアクセスを与える"全てのコンセプト（未フォーマット）" を表示します。これは、インスタンスファイル内の概念のフォーマットされていないXML 構造 をモデルとしていて、これらのアイテムをマッピングする場合、`contextRef` 属性を手動でマップしなければなりません。 このビューは任意で表示されます。

メモ

- XBRL コンポーネントからデータを読み込む場合、（すなわち、XBRL コンポーネントがソースコンポーネントの場合）、コンテキストメニュー から構造ビューの組み合わせを選択することができます。XBRL にデータを書き込む場合、**テーブルリンクベースからのテーブル** ビューが選択されている場合、技術的な理由から、これ以外のビューにデータを書き込むことはできません。
- 「**テーブルリンクベースからのテーブル**」ビューは"ビルドイン"が変換言語として選択設定されている必要があります（ 変換言語の選択も参照）。

6.8.5 XBRL タクソノミマネジャー

XBRL タクソノミ マネジャーはMapForce を含むAltova XBRL-が有効化されているアプリケーション全体で使用するためにXBRL タクソノミをインストールし管理する中央化した方法を提供するツールです。W indows ではXBRL タクソノミ マネジャーには視覚的なユーザーインターフェイスがあり、コマンドラインでも使用可能です。Linux とMac* 上ではコマンドラインでのみツールを使用することができます。

＊Linux とmacOS オペレーティングシステムをこれらのオペレーティングシステム上でXBRL タクソノミ マネジャーをMapForce Server、StyleVision Server、またはRaptorXML+XBRL Server などのAltova クロスプラットフォームサーバーと共に実行している場合に適用可能です。

XBRL タクソノミ マネジャーは以下の機能を搭載しています:

- 使用中のコンピューターにインストールされているXBRL タクソノミをビューして、新規のバージョンをダウンロード可能か確認してください。
- Altova 製品のリリースサイクルに関わらずXBRL タクソノミの新しいバージョンをダウンロードします。全てのタクソノミはXBRL タクソノミ マネジャーによりアクセス可能なオンラインベースのパッケージストレージ上でAltova により管理されており、使用可能になるとすぐにダウンロードすることが可能です。
- タクソノミの複数のバージョン（または、必要な場合全てのバージョン）をインストールまたはアンインストールする
- 単一のXBRL タクソノミは［パッケージ］を表しており、他のタクソノミ上に依存関係が存在する場合があります。特定のタクソノミをインストールまたはアンインストールすることを選択すると、依存関係を持つタクソノミが検知され、自動的にインストールまたはアンインストールされます。グラフィカルなユーザーインターフェイス（または適用できる場合コマンドラインは）依存が追加または削除されることを通知します。
- XBRL タクソノミ マネジャーにより管理されるXBRL タクソノミはインスタンス、またはスキーマドキュメントインのURI レファレンスがインターネットからファイルを抽出することとは異なり、ローカルファイルから解決されることを有効化するXML カタログ メカニズムから利益を得ることができます。通常パフォーマンス上の理由からリモート URI からのスキーマのレゾルーションが実質的ではない場合、大きなXBRL タクソノミの場合これはとても重要です。

XBRL タクソノミ マネジャーはXBRL タクソノミの管理者がAltova XBRL-有効化されたアプリケーションの一つを使用していることを必要

とします。欧州銀行監督局レポーティングフレームワークタクソノミ、US-GAAP 金融レポーティングタクソノミ、他の多種の国別、または、ドメイン固有のXBRL タクソノミが含まれます。完全なリストを確認するにはXBRL タクソノミ マネジャーを実行または `list` コマンドをコマンドラインで実行します 。インストール済みパッケージをビューを参照してください。

カスタム XBRL タクソノミ

XBRL タクソノミ マネジャーに含まれるカスタムXBRL タクソノミと作業する場合、これらを以下のように有効化します:

- Altova XBRL-有効化済みのデスクトップアプリケーションから **ツール | オプション** メニューコマンドを実行し、**XBRL | タクソノミ パッケージ** 設定ページに移動します。設定ページからカスタムXBRL タクソノミの .zip パッケージを参照してください。詳細に関しては XBRL タクソノミ パッケージ を参照してください。
- サポートするコマンドまたはAPI メソッドを実行する場合 Altova XBRL-有効化済みサーバーアプリケーションから― `taxonomy-package` または `--taxonomy-package-config-file` オプションを提供してください。例えば、これらのオプションは **`valxbrl`** または **`valxbrltaxonomy`** などのRaptorXML+XBRL Server 内のXBRL 検証コマンドまたは MapForce Server 内の **`run`** コマンド（または関連するAPI メソッド）によりサポートされています。

Altova XBRL-有効化済みアプリケーション

以下のAltova アプリケーションはXBRL-有効化済みで、XBRL タクソノミ マネジャーにより提供される機能によりメリットを受けています:

- Excel のためのAltova XBRL アドイン（EBA, Solvency II）
- MapForce Enterprise Edition
- MapForce Server
- MapForce Server Advanced Edition
- RaptorXML+XBRL Server
- StyleVision Server
- StyleVision Enterprise Edition
- XMLSpy Enterprise Edition

> XBRL タクソノミ マネジャーを使用したXBRL タクソノミ内での変更は同じコンピューターにインストールされている場合、上にリストされたすべてのアプリケーションで反映されます。

インストール

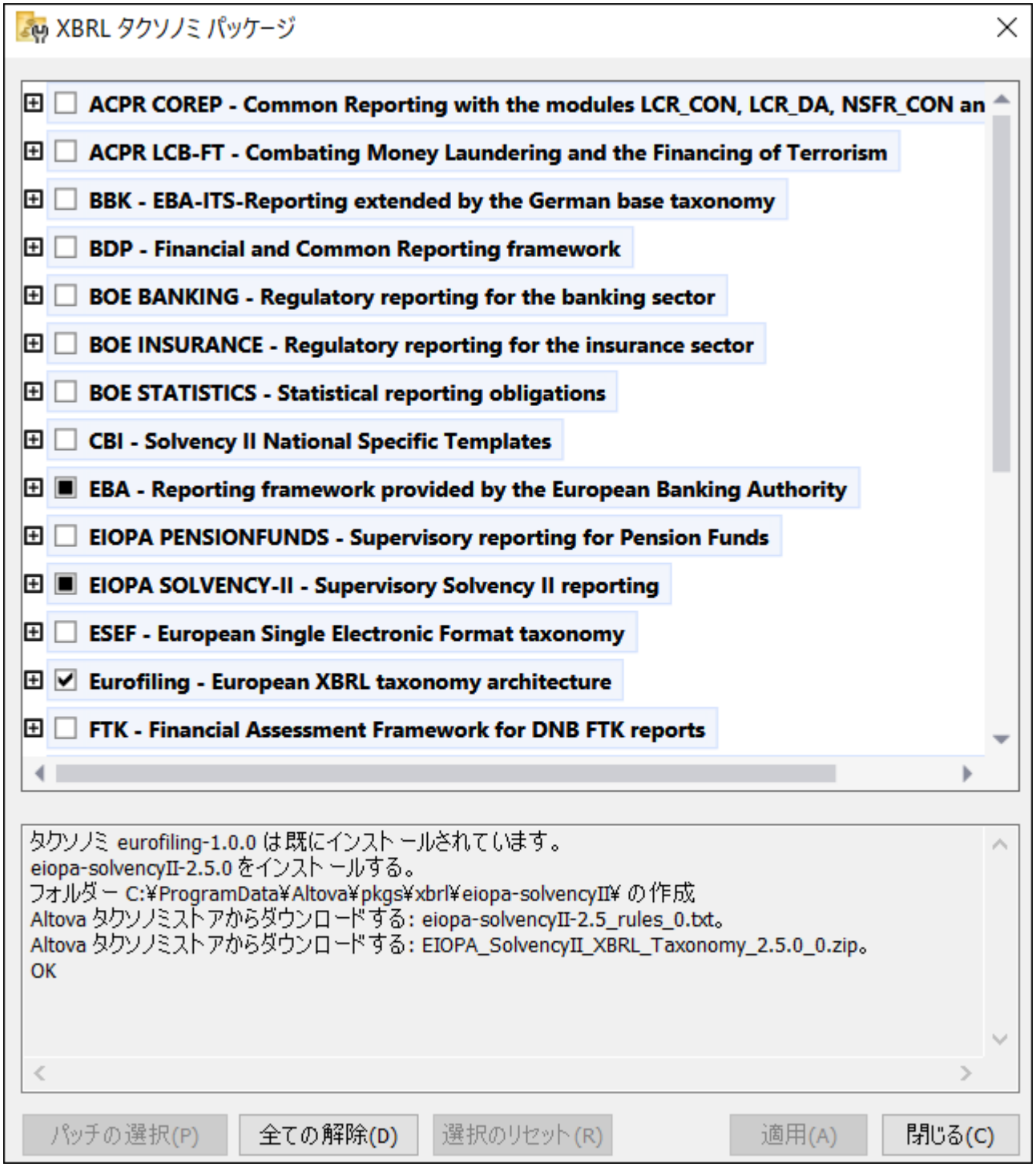
Altova XBRL-有効化済みアプリケーション、または **Altova Mission Kit Enterprise Edition** をインストールするとXBRL タクソノミ マネジャーは自動的にインストールされます。同様に、Altova XBRL-有効化済みアプリケーション、または **Altova Mission Kit Enterprise Edition** をコンピューターからアンインストールすると自動的にアンインストールされます。

しくみ

Altova は製品内で使用されている全てのXBRL タクソノミを保管するオンラインタクソノミストレージを管理しています。このタクソノミストレージは定期的に更新されます。例えば、発行機関により関連するタクソノミの新規のバージョンがリリースされた後などに更新されます。

XBRL タクソノミ マネジャーをグラフィカルなユーザーインターフェイスで実行すると、タクソノミをインストール、更新またはアンインストールすることができるダイアログボックス内に使用可能な最新のタクソノミの情報が表示されます。コマンドラインインターフェイスで同じアクションを実行することができます。

(C) 2015-2021 Altova GmbH



Altova W eb サイト (https://www.altova.com/download) からダウンロードされた running .altova_ taxonomies ファイルを実行してタクソノミをインストールすることができます。更にXBRL タクソノミをロードするアクションを取る場合 MapForce は不足するXBRL タクソノミをインストールするようにプロンプトします。

タクソノミのインストール方法に関わらず、インストール済みのタクソノミに関する情報はコンピューター内で中央化された場所でトラックされてい

ます。これはローカルキャッシュディレクトリとして知られています。ローカルキャッシュディレクトリには(オンデマンドでインストールされる実際のタクソノミファイルを除き) Altova パッケージに関する情報が含まれています。ローカルキャッシュディレクトリは以下のパスで見つけることができます:

Linux	`/var/opt/Altova/pkgs`
macOS	`/var/Altova/pkgs`
Windows	`C:\ProgramData\Altova\pkgs`

ローカルコンピューターにオンラインストレージの最新の状態を作成するため、ローカルキャッシュディレクトリは時々自動的に更新されます。更に具体的には、キャッシュは以下のように更新されます:

- XBRL タクソノミマネジャーを実行 する場合。
- 同じカレンダーの日付で MapForce を実行する場合。
- MapForce が既に実行中の場合、キャッシュディレクトリは 24 時間ごとに更新されます。
- コマンドラインインターフェイスから更新を実行しローカルキャッシュをオンラインストレージからオンデマンドで更新することができます。

タクソノミをインストールまたはアンインストールすると、ローカルキャッシュディレクトリは使用可能な情報、およびインストールされているタクソノミの情報、およびタクソノミファイル自身と共に自動的に更新されます。

ローカルキャッシュディレクトリはインストールまたはアンインストールするタクソノミをベースに自動的に管理されます。手動で変更または削除されてはなりません。XBRL タクソノミ マネジャーを元の状態にリセットする必要がある場合、コマンドラインインターフェイスの `reset` コマンドを実行してから `initialize` コマンドを実行します。または `reset` コマンドを `-i` オプションと共に実行してください。

6.8.5.1 XBRL タクソノミ マネジャー の実行

XBRL タクソノミ マネジャーを開始するには、以下の内の1つを行ってください:

- MapForce の[ツール] メニュー内で、[XBRL タクソノミマネージャー]を選択します。
- Altova W eb サイトからダウンロードされた.altova_ taxonomies 拡張子を持つファイルを(ダブルクリックして) 実行します。
- W indows コントロールパネルから Altova タクソノミマネジャー エントリを右クリックして、変更 または アンインストール をコンテキストメニューから選択します。

更に、MapForce のインストール後チェックボックス Altova タクソノミマネージャーの呼び出し がインストールウィザードの最後のページで使用可能になります。

コマンドライン インターフェイス

XBRL タクソノミ マネジャーをコマンドインターフェイスから実行する方法:

1. コマンドプロンプトを開きディレクトリを **C:\ProgramData\Altova\SharedBetweenVersions** に変更します。
2. コマンドラインでヘルプを表示するには、以下を実行します:

```
TaxonomyManager.exe --help
```

6.8.5.2 タクソノミのインストール

タクソノミをインストールする方法:

1. 実行 する。
2. インストールするタクソノミまたはタクソノミ バージョンの横のチェックボックスを選択し [適用]をクリックします。

または**.altova_taxonomies** 拡張子を持つファイルを Altova W eb (https://www.altova.com/taxonomy-manager)サイトからダウンロードし、**.altova_taxonomies** ファイルを実行するためにダブルクリックします。**.altova_taxonomies** ファイルを実行する際に XBRL タクソノミ マネジャーが開かれます。

以下の点に注意してください:

- XBRL タクソノミ マネジャーからタクソノミをインストールまたはアンインストールすると同じコンピューター上のすべてのユーザーアカウントに影響を及ぼします。
- XBRL タクソノミ マネジャーからタクソノミをインストールまたはアンインストールすると同じコンピューター上にインストールされている Altova XBRL-有効化済みのすべてのアプリケーションに影響を与えます。
- 現在のタクソノミは他のタクソノミ上に依存関係があり、依存するタクソノミもインストールされます(または適用可能な場合アンインストールされます)。

コマンドライン インターフェイス

タクソノミをインストールするには、以下を実行します:

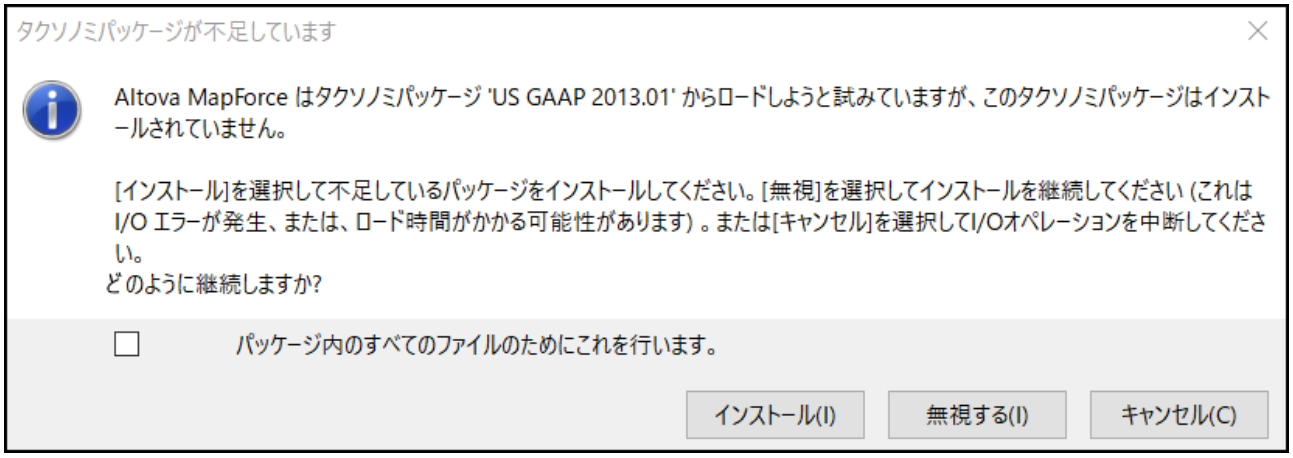
```
TaxonomyManager.exe install FILTER...
```

`FILTER` が以下を意味する箇所:

1. フォーマット `<name>-<version>` 内のタクソノミ識別子の例: **eba-2.10**、**us-gaap-2020.0**。使用可能なすべてのタクソノミ識別子をビューするには `list` コマンドを実行します。
2. Altova W eb サイトからダウンロードされた**altova_taxonomies** ファイル。

オンデマンドでタクソノミをインストールする

MapForce が特定のXBRL タクソノミを検知すると使用中のコンピューターにインストールされる必要があります。オンデマンドでタクソノミをインストールするようにプロンプトされる場合があります。例えば、下のようなダイアログボックスはXBRL タクソノミのロードを必要とするアクションを実行すると表示される場合があります:



不足するタクソノミをインストールする場合 XBRL タクソノミ マネジャーはインストールとこの情報をトラックするためにローカルキャッシュディレクトリの更新を実行します。**ツール | XBRL タクソノミ マネジャー**から XBRL タクソノミ マネジャーを実行して以前にインストールされたすべてのタクソノミを常に確認することができます。

無視 を選択すると、MapForce は乏しいパフォーマンスまたはエラーを引き起こす可能性のあるローカルファイルの代わりに URI を公開されたオリジンから解決します。

6.8.5.3 パッチの適用

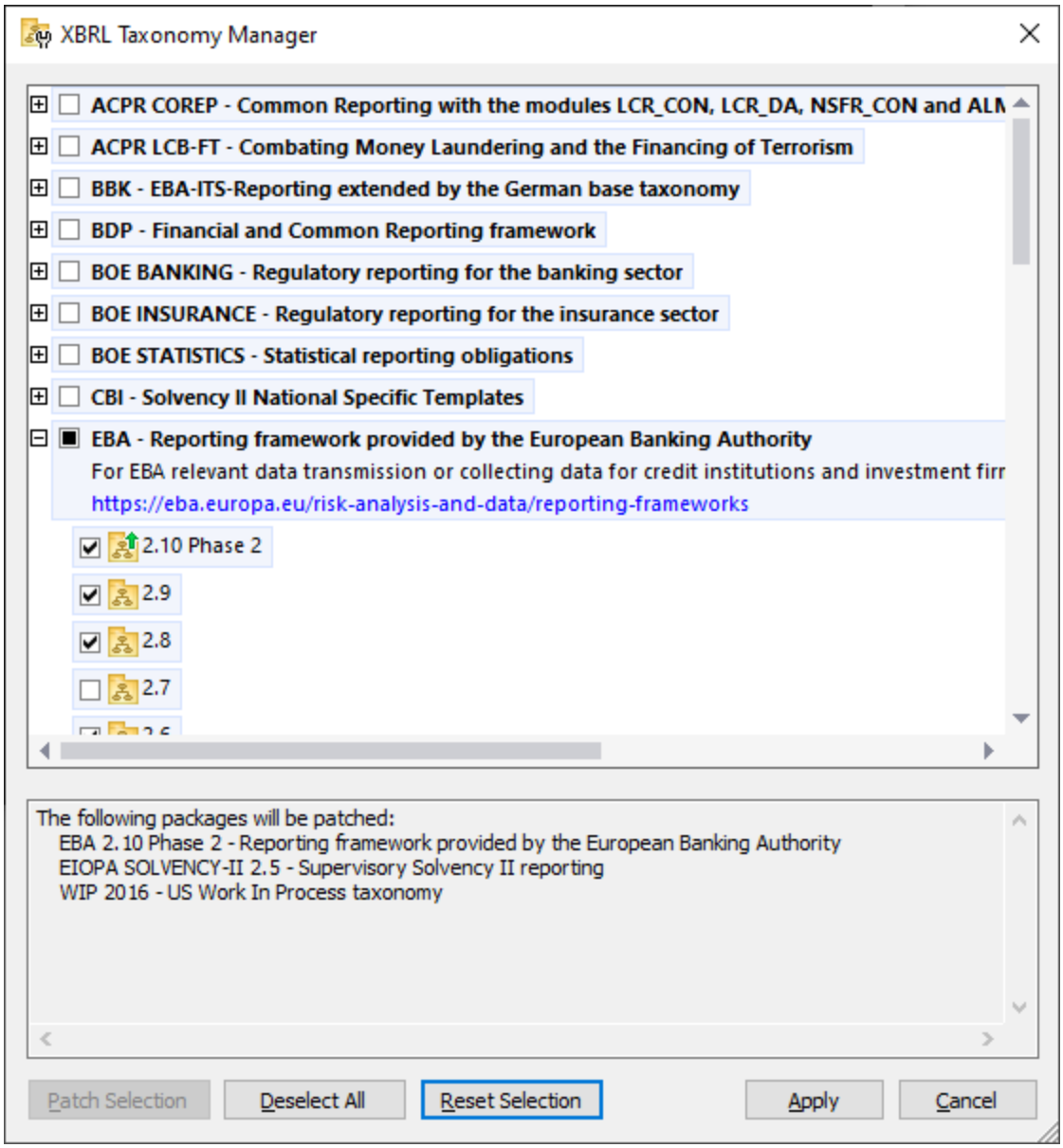
不定期に XBRL タクソノミは発行元からパッチを受け取ります。XBRL タクソノミマネジャーがパッチが使用可能であることを検知すると、以下が表示されます:

- Windows の視覚的なユーザーインターフェイスを介して XBRL タクソノミマネジャーを使用すると、対応する XBRL タクソノミは アイコンと共に表示されます。
- コマンドラインまたは Linux/macOS システムを使用すると、`list -u` コマンドと共に実行可能ファイルを実行する際にリストされているパッチを持つ XBRL タクソノミがリストされます。

Windows 上でパッチを適用する方法:

1. **[パッチの選択]** ボタンをクリックします。 から への変更を修飾する各 XBRL タクソノミのアイコンとダイアログボックスは適用されるパッチについて通知します。例:

(C) 2015-2021 Altova GmbH



メモ **パッチの選択** ボタンは現在インストールされているXBRL タクソノミのためにパッチが使用な箇所でのみ有効化されています。

2. [適用]をクリックします。

コマンドラインインターフェイスでパッチを適用する方法:

1. `list -u` コマンドを実行します。パッチ更新が使用可能なタクソノミをリストします。
2. `upgrade` コマンドを実行してパッチをインストールします。

更に詳しい情報に関しては コマンドラインインターフェイス。

6.8.5.4 インストール済みのタクソノミをビューする

視覚的なユーザーインターフェイスからインストール済みのすべてのタクソノミをビューするにはXBRL タクソノミ マネジャーを実行します。タクソノミ（または、タクソノミのバージョン）の横の選択済みのチェックボックスはそのタクソノミがインストール済みであることを示しています。

コマンドライン インターフェイス

コマンドラインインターフェイスからすべての使用可能なタクソノミをビューするには、以下を実行します:

```
TaxonomyManager.exe list
```

インストール済みのタクソノミのみをビューするには、以下を実行します:

```
TaxonomyManager.exe list -i
```

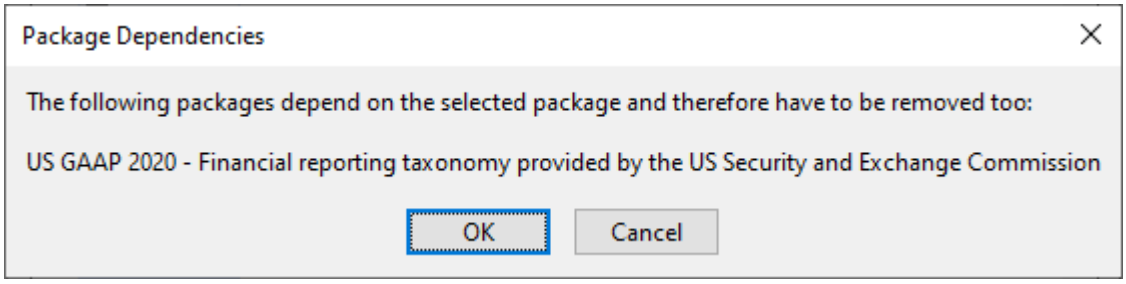
新規のバージョンが使用可能なタクソノミのみをビューするには、以下を実行します:

```
TaxonomyManager.exe list -u
```

6.8.5.5 パッケージのアンインストール

タクソノミをアンインストールする方法:

1. 実行する。
2. アンインストールするタクソノミまたはタクソノミバージョンの横のチェックボックスのチェックを解除します。選択されたタクソノミが他のタクソノミパッケージに依存する場合、依存関係も削除されることを通知するダイアログボックスが開かれます。例:



3. [適用] をクリックしてタクソノミとその依存関係を削除します。

コマンドライン インターフェイス

タクソノミをアンインストールするには、以下を実行します:

```
TaxonomyManager.exe uninstall FILTER...
```

`FILTER` が以下を意味する箇所:

1. フォーマット `<name>-<version>` 内のタクソノミ識別子の例: **eba-2.10**、**us-gaap-2020.0**。使用可能なすべてのタクソノミ識別子をビューするには `list` コマンドを実行します。

2. Altova W eb サイトからダウンロードされた**altova_taxonomies** ファイル。

6.8.5.6 コマンドラインインターフェイス

XBRL タクソノミ マネジャーをコマンドラインで呼び出すには場合実行可能ファイルのパスを知っている必要があります。デフォルトで XBRL タクソノミ マネジャー実行可能ファイルを以下のパスで見つけることができます:

*Linux**	/opt/Altova/<%APPNAME-UL%>2021/bin/taxonomymanager
*macOS**	/usr/local/Altova/<%APPNAME-UL%>2021/bin/taxonomymanager
Windows	C:\ProgramData\Altova\SharedBetweenVersions\TaxonomyManager.exe

＊Linux とmacOS パスをこれらのオペレーティングシステム上で XBRL タクソノミ マネジャーを MapForce Server、StyleVision Server、または RaptorXML+XBRL Server などの Altova クロスプラットフォームサーバーと共に実行している場合に適用可能です。

規則としては、このドキュメンテーションは特定のコマンドを説明する際に実行可能ファイルのフルパスを省略し、実行可能ファイルのなめを使用する代わりに**<exec>** を使用します。例:

```
<exec> help
```

<exec> は実行可能ファイルのパスまたは名前です。

XBRL タクソノミ マネジャーをインストールディレクトリから呼び出す方法

実行可能ファイルをフルパスを入力することなく呼び出すには、現在のディレクトリを以下の1つに変更します:

Linux	cd /opt/Altova/<%APPNAME-UL%>2021/bin
macOS	cd /usr/local/Altova/<%APPNAME-UL%>2021/bin
Windows	cd "C:\ProgramData\Altova\SharedBetweenVersions"

相対的なパスを使用してコマンドを実行し実行可能ファイルを呼び出すことができます。

Linux	sudo ./taxonomymanager help
macOS	sudo ./taxonomymanager help
Windows	TaxonomyManager.exe help

メモ Linux とmacOS システム上では、プレフィックス **./** は実行可能ファイルが現在のディレクトリ内にあることを示しています。**sudo** はコマンドがルート特権と共に実行されなければならないことを示しています。

XBRL タクソノミ マネジャーをディレクトリから呼び出す方法

実行可能ファイルをディレクトリから呼び出すには絶対パスを使用して参照してください。または、プログラムを実行可能ファイル名を入力する

だけで呼び出す場合、使用中のオペレーティングシステムのPATH 環境変数を編集することにより、実行可能ファイルのディレクトリへのフルパスが含まれるようになります。PATH 環境変数を変更する方法に関しては、使用中のオペレーティングシステムのドキュメンテーションを参照してください。

メモ

- PATH 環境変数の変更後、ターミナルウィンドウを閉じ、変更の効果が反映されるように新規のウィンドウを開く必要がある場合があります。
- Linux とmacOS 上では、 `sudo` の使用はユーザーのPATH を考慮していません。

コマンドライン構文

コマンドラインを使用するための一般的な構文は以下の通りです:

```
<exec> -h | --help | --version | <command> [options] [arguments]
```

上のリスト内で垂直バー `|` は相互に排他的なアイテムを分割します。角かっこは `[]` は任意のアイテムを示しています。基本的には、 `--h`, `--help`、または `--version` オプションまたはコマンドが後に続く実行可能ファイルパスを入力することができます。各コマンドはオプションと引数を持つ場合があります。コマンドのリストは以下のセクションで説明されています:

6.8.5.6.1 help

このコマンドはXBRL タクソノミ マネジャー実行可能ファイルに関連するコマンドに関するコンテキストのヘルプを提供します。

構文

```
<exec> help [command]
```

[command] が有効なコマンド名を指定するオプションの引数である個所。

リマーク

`-h` または `--help` が後に続くコマンドを入力してヘルプを呼び出すことができます。例:

```
<exec> list -h
```

実行可能ファイルの後に `-h` または `--help` を入力して一般的なヘルプを呼び出すことができます。例:

```
<exec> --help
```

サンプル

次のコマンドは `list` コマンドに関するヘルプを表示します:

Linux、macOS	`./taxonomymanager help list`
Windows	`TaxonomyManager.exe help list`

(C) 2015-2021 Altova GmbH

6.8.5.6.2 info

このコマンドは引数として与えられたタクソノミのそれぞれの詳細を表示します。詳細にはタイトル、バージョン、詳細、後悔元、依存レファレンスが含まれています。

構文

```
<exec> info [options] FILTER...
```

複数のタクソノミを指定するには `FILTER` を必要な回数繰り返します。

引数

`FILTER`	名前内にこの文字列を含むタクソノミをリストします（例えば、**eba** または **us-gaap-2020.0**）。この引数を複数回指定することができます。 識別子のフルリストをビューするには `list` コマンドを実行します。

オプション

`--help, --h`	コマンドラインでこのコマンドに関するヘルプを表示します。

例

以下のコマンドは **eba-2.10** と **us-gaap-2020.0** タクソノミに関する詳細を表示します:

Linux、macOS	`./taxonomymanager info eba-2.10 us-gaap-2020.0`
Windows	`TaxonomyManager.exe info eba-2.10 us-gaap-2020.0`

6.8.5.6.3 initialize

このコマンドはXBRL タクソノミ マネジャー環境を初期化します。タクソノミに関するすべての情報が保管されている個所でキャッシュディレクトリを作成します。XBRL タクソノミ マネジャーを含むAltova アプリケーションをインストールすると初回に初期化が実行されるため、通常このコマンドを実行する必要はありません。`reset` コマンドを実行後に通常このコマンドを 実行する必要があります。

構文

```
<exec> initialize [options]
```

このコマンドのエイリアスは `init` です。

オプション

`--help, --h`	コマンドラインでこのコマンドに関するヘルプを表示します。

--silent, --s	エラーメッセージのみを表示します。デフォルトの値は**false** です。
--verbose, --v	実行中の追加情報を表示します。デフォルトの値は**false** です。

6.8.5.6.4 install

このコマンドは1つまたは複数のタクソノミをインストールします。以下の点に注意してください:

- XBRL タクソノミ マネジャーからタクソノミをインストールまたはアンインストールすると同じコンピューター上のすべてのユーザーアカウントに影響を及ぼします。
- XBRL タクソノミ マネジャーからタクソノミをインストールまたはアンインストールすると同じコンピューター上にインストールされている Altova XBRL-有効化済みのすべてのアプリケーションに影響を与えます。
- 現在のタクソノミは他のタクソノミ上に依存関係があり、依存するタクソノミもインストールされます(または適用可能な場合アンインストールされます)。

構文

```
<exec> install [options] FILTER...
```

複数のタクソノミを指定するには FILTER を必要な回数繰り返します。

引数

FILTER	FILTER が以下を意味する箇所: 1. フォーマット <name>-<version> 内のタクソノミ識別子の例: **eba-2.10**、**us-gaap-2020.0**。使用可能なすべてのタクソノミ識別子をビューするには list コマンドを実行します。 2. Altova W eb サイトからダウンロードされた**altova_taxonomies** ファイル。 一意の場合省略された識別子を使用することができます、例えば、**eba**。省略された識別子を使用する場合、そのタクソノミの使用可能な最新バージョンがインストールされます。

オプション

--help, --h	コマンドラインでこのコマンドに関するヘルプを表示します。
--silent, --s	エラーメッセージのみを表示します。デフォルトの値は**false** です。
--verbose, --v	実行中の追加情報を表示します。デフォルトの値は**false** です。

サンプル

以下のコマンドは最新の**eba** (欧州銀行監督局) と**us-gaap** (米国会計基準) タクソノミをインストールします:

Linux、macOS	./taxonomymanager install eba us-gaap
Windows	TaxonomyManager.exe install eba us-gaap

6.8.5.6.5 list

このコマンドを使用して以下の方法の1つによりコマンドラインでタクソノミをリストします。

- 使用可能なタクソノミを全てリストします
- 特定のタクソノミをリストします
- インストール済みのタクソノミのみをリストします
- アップグレードが必要なタクソノミのみをリストします

構文

```
<exec> list [options] [FILTER...]
```

このコマンドは `ls` を使用して省略することができます。

引数

FILTER	名前内にこの文字列を含むタクソノミのみをリストします。この引数を複数回指定することができます。

オプション

--help, --h	コマンドラインでこのコマンドに関するヘルプを表示します。
--installed, --i	インストール済みのタクソノミのみをリストしますデフォルトの値は **false** です。
--upgradeable, --u	パッチ更新が使用可能なタクソノミのみをリストします。デフォルトの値は **false** です。

例

使用可能な全てのタクソノミをリストするには、以下を実行します:

Linux、macOS	`./taxonomymanager list`
Windows	`TaxonomyManager.exe list`

インストール済みのタクソノミのみをリストするには、以下を実行します:

Linux、macOS	`./taxonomymanager list -i`
Windows	`TaxonomyManager.exe list -i`

名前内に"eba"または"us-gaap"を含むすべてのタクソノミをリストするには、以下を実行します:

Linux、macOS	`./taxonomymanager list eba us-gaap`
Windows	`TaxonomyManager.exe list eba us-gaap`

6.8.5.6.6 reset

このコマンドはすべてのインストールされたタクソノミとキャッシュディレクトリを削除します。

> 警告: このコマンドはすべてのインストールされたタクソノミとその情報を削除します。

このコマンドの実行後、キャッシュディレクトリを再作成するために `initialize` コマンドを実行してください。または `reset` コマンドを `-i` オプションと共に実行してください。

`reset -i` は製品オリジナルのインストールを復元するため、リセット後に更に `update` コマンドを実行することが奨励されます。または、`-i` と `-u` オプションを使用してコマンドをリセットします。

構文

```
<exec> reset [options]
```

オプション

`--help, --h`	コマンドラインでこのコマンドに関するヘルプを表示します。
`--init, --i`	XBRL タクソノミマネジャー環境をリセット後初期化してください。有効な値は**true** または**false** です。デフォルトの値は**false** です。
`--silent, --s`	エラーメッセージのみを表示します。デフォルトの値は**false** です。
`--update, --u`	XBRL タクソノミマネジャー環境をリセット後に初期化し更新してください。有効な値は**true** または**false** です。デフォルトの値は**false** です。
`--verbose, --v`	実行中の追加情報を表示します。デフォルトの値は**false** です。

例

XBRL タクソノミ マネジャーをリセットするには、以下を実行します:

Linux、macOS	`./taxonomymanager reset`
Windows	`TaxonomyManager.exe reset`

6.8.5.6.7 setdeflang

このコマンドはXBRL タクソノミ マネジャーの言語を設定します。

構文

```
<exec> setdeflang language
```

`language` が言語コードを提供する必須の引数の箇所。このコマンドのエイリアスは `sdl` です。

引数

language	設定する言語有効な言語は英語 (en)、フランス語、(fr)、ドイツ語 (de)、日本語 (ja)、およびスペイン語 (es) です。

オプション

--help, --h	コマンドラインでこのコマンドに関するヘルプを表示します。

サンプル

言語をスペイン語に設定するには、以下を実行します:

Linux、macOS	`./taxonomymanager setdeflang es`
Windows	`TaxonomyManager.exe setdeflang es`

6.8.5.6.8 uninstall

このコマンドは 1 つまたは複数のタクソノミをアンインストールします。デフォルトでは現在のタクソノミにより参照されるタクソノミもアンインストールされます。現在のタクソノミだけをアンインストールし、参照されたタクソノミを保つにはオプション `--k` を設定します。

構文

```
<exec> uninstall FILTER...
```

複数のタクソノミを指定するには `FILTER` を必要な回数繰り返します。

引数

FILTER	`FILTER` が以下を意味する箇所: 1. フォーマット `<name>-<version>` 内のタクソノミ識別子の例: **eba-2.10**、**us-gaap-2020.0**。使用可能なすべてのタクソノミ識別子をビューするには `list` コマンドを実行します。 2. Altova Web サイトからダウンロードされた **altova_taxonomies** ファイル。

オプション

--help, --h	コマンドラインでこのコマンドに関するヘルプを表示します。
--keep-references, --k	このオプションが設定されると、参照されたタクソノミはアンインストールされません。デフォルトの値は **false** です。
--silent, --s	エラーメッセージのみを表示します。デフォルトの値は **false** です。
--verbose, --v	実行中の追加情報を表示します。デフォルトの値は **false** です。

サンプル

以下のコマンドは**eba-2.10** と**us-gaap-2020.0** タクソノミとその依存関係をアンインストールします:

Linux、*macOS*	`./taxonomymanager uninstall eba-2.10 us-gaap-2020.0`
Windows	`TaxonomyManager.exe uninstall eba-2.10 us-gaap-2020.0`

6.8.5.6.9 update

このコマンドクエリはオンラインストレージから使用可能なタクソノミのリストをクエリし、ローカルのキャッシュディレクトリを更新します。この情報の更新は明示的に行われ、`reset` と`initialize` を実行した場合を除いてこのコマンドを実行する必要はありません。

構文

```
<exec> update [options]
```

オプション

`--help, --h`	コマンドラインでこのコマンドに関するヘルプを表示します。
`--silent, --s`	エラーメッセージのみを表示します。デフォルトの値は**false** です。
`--verbose, --v`	実行中の追加情報を表示します。デフォルトの値は**false** です。

サンプル

以下のコマンドはタクソノミの情報を明示的に更新します。

Linux、*macOS*	`./taxonomymanager update`
Windows	`TaxonomyManager.exe update`

6.8.5.6.10 upgrade

このコマンドは全ての対象のタクソノミを使用可能な最新 *patch* バージョンにアップグレードします。すなわち、特定のリリースのパッチレベルでのみ更新を行います。このコマンドを実行することは使用可能なタクソノミを更新する場合のみ意味があります。`list -u` コマンドを実行して更新可能なタクソノミを識別することができます。

構文

```
<exec> upgrade [options]
```

オプション

`--help, --h`	コマンドラインでこのコマンドに関するヘルプを表示します。

(C) 2015-2021 Altova GmbH

--silent, --s	エラーメッセージのみを表示します。デフォルトの値は**false** です。
--verbose, --v	実行中の追加情報を表示します。デフォルトの値は**false** です。

6.8.6 カスタム XBRL タクソノミ パッケージ

XBRL *タクソノミ パッケージ* は、オフラインのXBRL *タクソノミ*のコピーを含むZIP *アーカイブ*です。XBRL *タクソノミ パッケージ*の利点は(*タクソノミ プロバイダ*から提供されている場合) スキーマなどのXBRL *リソース*がインターネットから抽出される代わりにローカルの(オフラインの)*タクソノミ*ファイルから抽出されます。XBRL *タクソノミ パッケージ*の構造とフォーマットは*タクソノミ パッケージ* 1.0 勧告で説明されています (https://www.xbrl.org/Specification/taxonomy-package/REC-2016-04-19/taxonomy-package-REC-2016-04-19.html)。

使用中のXBRL *タクソノミ* が上記の勧告に準拠するXBRL *タクソノミ パッケージ*のダウンロードを提供する場合、このような*パッケージ*をMapForce に追加することができます。これはXBRL 検証と処理に影響を与えます(XBRL *スキーマ*レファレンス等の特定のURL が*タクソノミ パッケージ*に含まれるローカル ファイルにリダイレクトされるなどと同じ意味です)。*タクソノミ パッケージ*は、XBRL *タクソノミ*の全てのエントリポイントをMapForce から直接確認することができ、XBRL 検証の*パフォーマンス*を向上します。

以下の点に注意してください:

1. 多くのXBRL *タクソノミ*はAltova によりホストされており XBRL *タクソノミ マネジャー* を使用して簡単に管理することができます. 使用しようとしているXBRL *タクソノミ*は既にXBRL *タクソノミマネジャー*により管理されており、カスタムXBRL *タクソノミ パッケージ*をダウンロードして有効化する必要はありません。
2. 次のAltova *デスクトップアプリケーション*はXBRL *タクソノミ パッケージ*の追加をサポートしています: XMLSpy、MapForce、および StyleVision。XBRL *タクソノミ パッケージ*リストはこれらすべてのアプリケーションで共通です。1つのアプリケーション内でリストを編集すると他のアプリケーション内でも変更されたリストが表示されます(そしてXBRL 検証と処理に影響が出ます)。

しくみ

各 XBRL *タクソノミ パッケージ*は内部の*カタログファイル*(**catalog.xml**)を含みます。この*カタログファイル*は、ローカル*リソース*にリダイレクトされる(マップし直される) URI を定義します。*カタログファイル*は、制限されたXML *カタログ*仕様のサブセットに準拠します (https://www.oasis-open.org/committees/download.php/14809/xml-catalogs.html)。同様のメカニズムがMapForceにも搭載されており、をローカル*リソース*にURI をマップし直すことができます。*カタログファイル*を参照してください。

1 つ、または複数のXBRL *タクソノミ パッケージ*を追加した後、(下の手順を参照してください)、MapForce がどのようにXBRL 検証のために*リソース*を解決するかを影響します。例えば、XBRL *インスタンス*をマッピングに追加すると、スキーマが、インターネットから抽出される代わりに、ローカル*タクソノミ パッケージ*からロードされます。(*タクソノミ パッケージ*がアクティブでその**catalog.xml** ファイルがスキーマURI をローカル*リソース*にマップし直すと想定します)。この結果、これは以降のすべてのXBRL 検証と処理に影響を与えます。

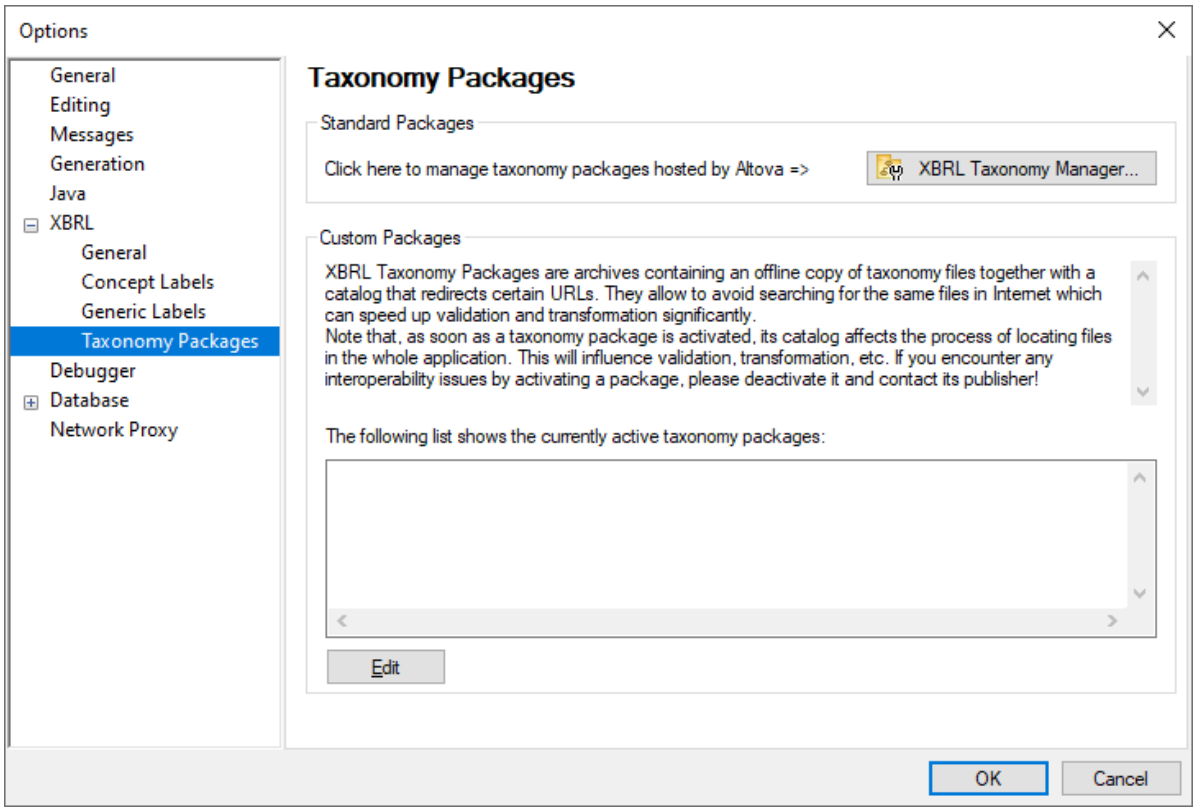
注意点: パッケージ カタログは、互換性の無い*リソース*にリダイレクトされる可能性があります
アクティブな*パッケージ*のカタログファイルにより指される*リソース*は、その*リソース*を必要とするすべてのMapForce オペレーションで使用されます。このような*リソース*のサンプルは、XML 検証 とXBRL 検証に使用されるXML スキーマです。*パッケージ*のカタログファイルによりロケートされるオフラインの*リソース*は、使用中の既存の環境と互換性が無いためにエラーが発生する場合があります。この場合、*タクソノミ パッケージ*を無効化して、エラー情報を*パッケージ*の作成者に伝えてください。

タクソノミ パッケージの追加

1. XBRL *タクソノミ パッケージ*(.zip *アーカイブ*) をプロバイダからダウンロードし、頻繁に変更される予定のないローカルディレクトリに保存します。。ダウンロードされた*アーカイブ*を解凍しないでください

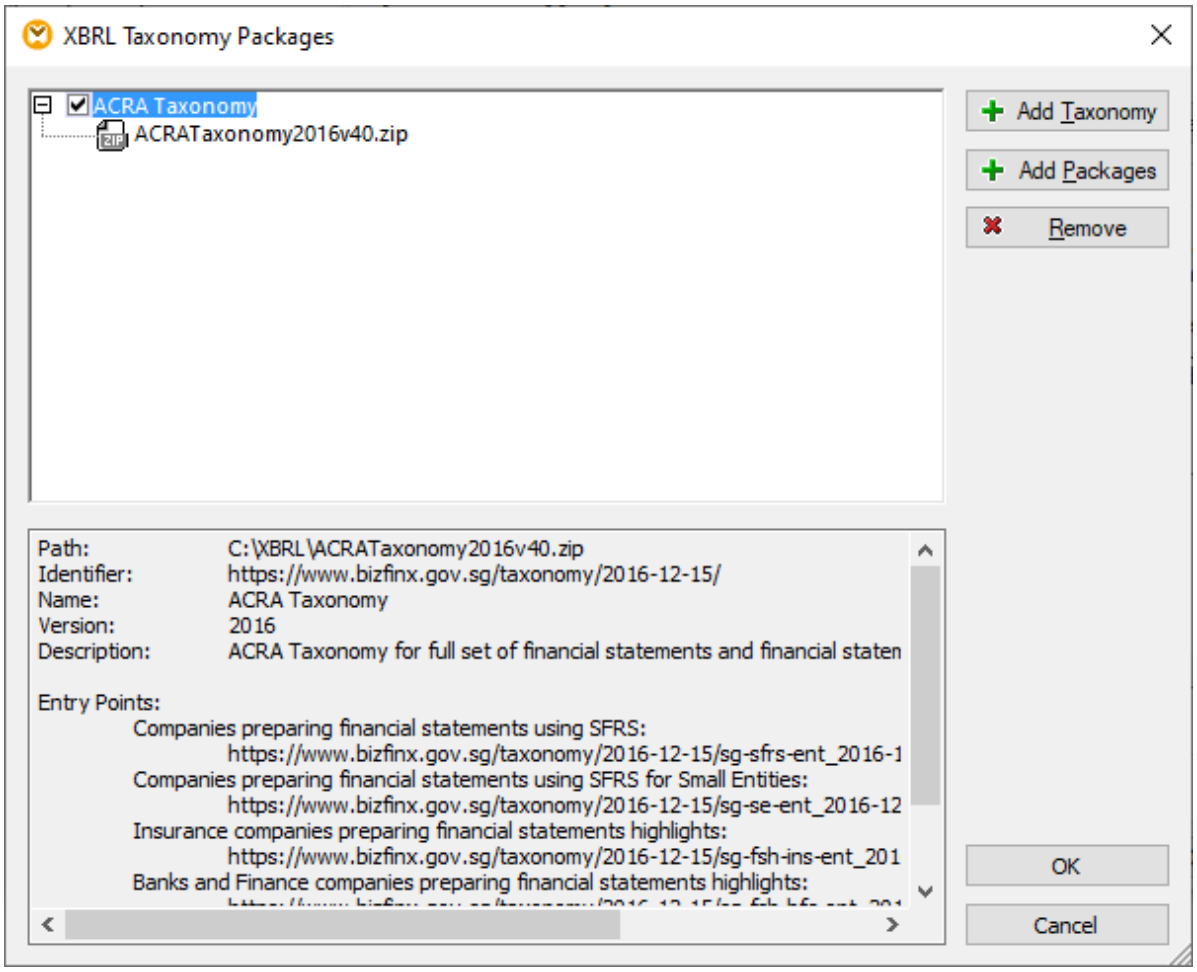
メモ ダウンロードされた.zip アーカイブは、タクソノミ パッケージ1.0 勧告に準拠する必要があります (https://www.xbrl.org/Specification/taxonomy-package/REC-2016-04-19/taxonomy-package-REC-2016-04-19.html)。

2. [ツール] メニューから[オプション | XBRL | タクソノミ パッケージ]をクリックします。



3. [編集]をクリックします。
4. [タクソノミの追加]をクリックして、.zip タクソノミ パッケージを参照します。複数のパッケージを選択する場合は **[Shift]** キーを押します。

追加されたXBRL タクソノミ パッケージはダイアログボックス内に2つのレベルで表示されます。最初のレベルはタクソノミを表し、2番目のレベルは、そのタクソノミのパッケージを表します。XBRL タクソノミ パッケージ ダイアログ内でタクソノミ パッケージが選択されると、(オフラインの場所も含まれる)詳細がダイアログの下のペインに表示されます。



タクソノミの左横のチェックボックスのエントリは、タクソノミがアクティブであるか表示します。デフォルトでは、新規に追加されたタクソノミがアクティブです。タクソノミを無効化するには、チェックボックスのチェックを解除してください。無効化は、タクソノミの2つのバージョン間を切り替える際に問題が発生すると、役に立つ機能です。

追加パッケージを既存のタクソノミに追加する場合、タクソノミを選択し、入力して[パッケージの追加]をクリックします。追加されたパッケージは、そのタクソノミの2番目のレベルに表示されます。パッケージを削除するには、選択して[削除]をクリックします(これは、ディスクから実際のタクソノミを削除しないことにご注意してください)。

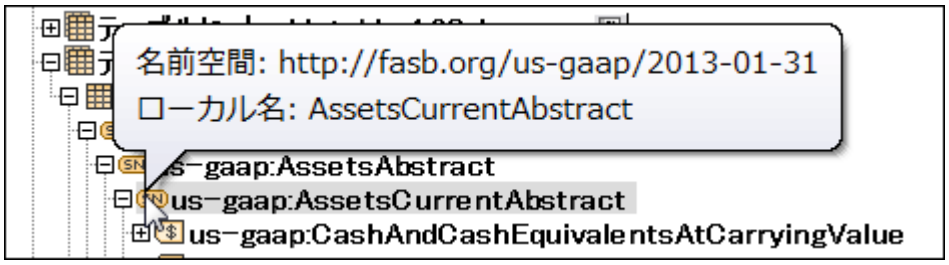
6.8.7 XBRL 設定と優先順位

MapForce では、XBRL 特有の設定を以下のように構成することができます:

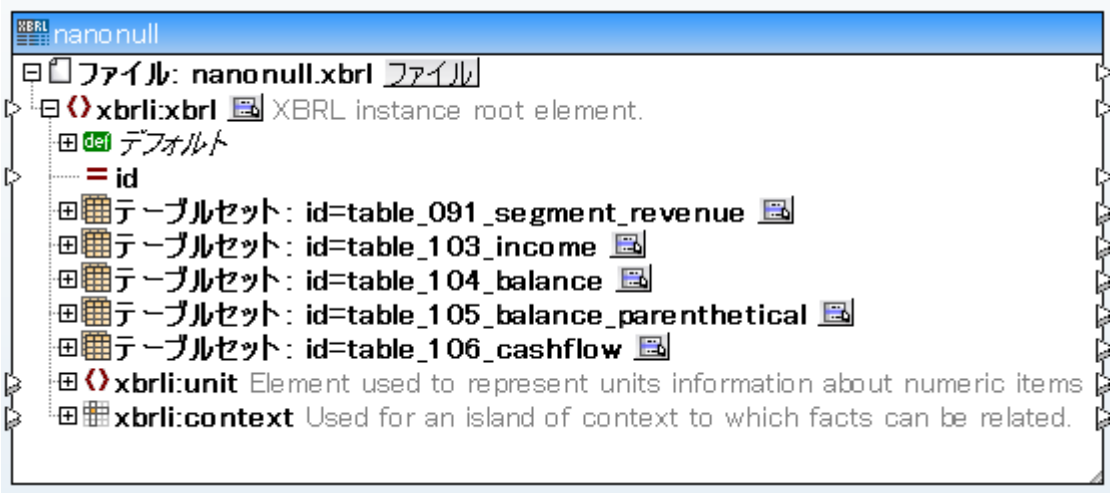
- マッピングレベルでコンポーネント アイテムに関する追加情報を確認することができます ヒントと注釈の有効化を参照してください)。
- マッピング上の各 XBRL コンポーネントのために適用可能な設定を変更することができます。XBRL コンポーネント設定を参照してください。
- プログラムレベルでは全般 XBRL オプションを参照してください。

6.8.7.1 ヒントと注釈の有効化

ヒントが有効化されていると、各コンポーネントアイテムについての追加情報を、マウスのカーソルをポイントすることにより、以下のように確認することができます。



(存在する場合)注釈テキストをオンとオフに切り替えることができます。有効化されている場合、注釈は、コンポーネント内でアイテムの右側に表示されます。例: 下のスクリーンショットでは、`xbrli:xbrl` ルート要素の注釈は、は"XBRL インスタンス ルート要素" です。



ヒントを有効化、または無効化するには、以下を行います:

- 「表示」メニューから「ヒントの表示」をクリックします。
- 「ヒントの表示」( ) ツールバーボタンをクリックします。

注釈を有効化、または無効化するには、以下を行います:

- 「表示」メニューから「注釈の表示」をクリックします。
- 「注釈の表示」( ) ツールバーボタンをクリックします。

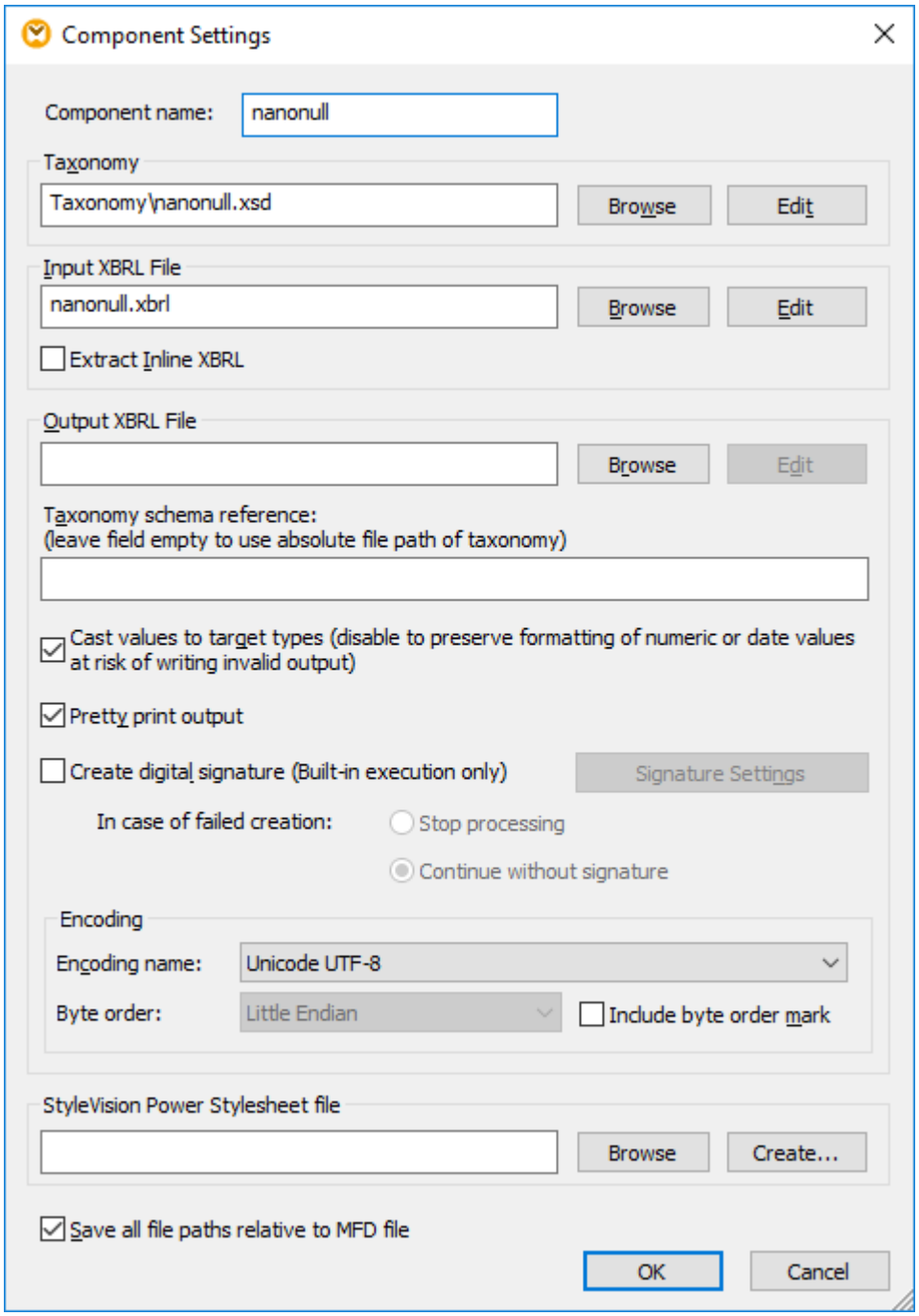
6.8.7.2 XBRL コンポーネント設定

マッピングエリアにXBRL コンポーネントを追加すると、コンポーネント設定ダイアログボックスから適用することのできる設定を構成することができます。

(C) 2015-2021 Altova GmbH

コンポーネント設定ダイアログボックスの開きかた

- XBRL コンポーネントのタイトルバーを右クリックし、[プロパティ] を選択します。



コンポーネント設定ダイアログボックスには以下の設定が含まれます。

コンポーネント名	コンポーネント名はコンポーネントを作成する際に自動的に生成されます。ですが、後に名前を変更することが可能です。 コンポーネント名はスペースを含むことができ、またフルストップ文字も含むことができます。コンポーネント名は、スラッシュ、バックスラッシュ、コロン、二重引用符、行頭と末尾スペースを含む場合があります。全般的に、コンポーネントの名前を変更する際、以下の影響を考慮してください: • マッピングを FlowForce Server にデプロイする場合、コンポーネント名は一意である必要があります。 • コマンドラインに入力できる文字のみを使用することが奨励されます。コマンドラインと Windows 内では、それぞれの文字で異なったエンコードを使用する場合があります。
タクソノミ	メインのタクソノミファイルのファイル名とパスを指定します。 ファイルのロケーションを変更するには[参照]をクリックして新規のファイルを選択します。 XBRL エディター(例: XMLSpy)内のファイルを編集するには、[編集] をクリックします。
入力 XBRL ファイル	現在選択されている XBRL コンポーネントのための入力 XBRL インスタンスのファイル名を指定します。このフィールドは XBRL コンポーネント を挿入して、XBRL インスタンスファイルを割り当てると自動的に入力されます。 ファイルのロケーションを変更するには[参照]をクリックして新規のファイルを選択します。 XBRL エディター(例: XMLSpy)内のファイルを編集するには、[編集] をクリックします。
インライン XBRL の抽出	このチェックボックスが選択されている場合 XBRL コンテンツは入力として指定されている .html、.htm または .xhtml ファイルから抽出されます。
出力 XBRL ファイル	コンポーネントがマッピングターゲットとして使用されている場合、XBRL ターゲットインスタンスファイルが保存される場所のファイル名とパスを指定します。 入力 XBRL インスタンスファイルを割り当てると、入力 XBRL ファイルフィールドからのエントリが自動的にこのフィールドにコピーされます。入力 XBRL インスタンスファイルをコンポーネントに割り当てない場合は、このフィールドはタクソノミファイルのファイル名とパスと拡張子 "xml"を含みます。 ファイルのロケーションを変更するには[参照]をクリックして新規のファイルを選択します。 XBRL エディター(例: XMLSpy)内のファイルを編集するには、[編集] をクリックします。
タクソノミ スキーマレファレンス	MFD ファイルに相対した参照された関連したタクソノミ スキーマファイルのパスです。このフィールドを検証のために異なるタクソノミを指定するために使用します。タクソノミ参照は `link:schemaRef` 要素の `href` 属性内に書き込まれます。

	`<link:schemaRef xlink:type="simple" xlink:href="..\..\nanonull.xsd"/>`
ターゲット 型に値をキャストする	ターゲット XML スキーマ型がマッピングする際に使用されるかを定義することを許可します。またすべてのデータがターゲットコンポーネントにマップされ文字列の値として処理されるかを定義することができます。デフォルトではこの設定 が有効化されています。
再フォーマットされた出力	構成された外観を与えるために、出力 XBRL ドキュメントを再フォーマットします。各子ノードは、単一タブ文字のそれぞれの親のオフセットです。
デジタル署名の作成	XBRL 出力インスタンスファイルにデジタル署名を追加することを許可します。"ビルドイン" を変換言語として選択している場合、デジタル署名を追加することができます。デジタル署名も参照してください。
エンコード	出力インスタンスファイルの次の設定を指定することを許可します: • エンコード名 • バイトオーダー • バイトオーダーマーク(BOM) 文字が含まれるか否か。
StyleVision Power スタイルシートファイル	このオプションは、Altova StyleVision スタイルシートファイルを選択または作成することを許可します。このようなファイルは XBRL インスタンスファイルからのデータをレポートのために HTML、RTF などの適切なフォーマットで出力することを有効化します。
MFD ファイルに相対的なすべてのファイルパスを保存する	このオプションが有効化されると、MapForce は、コンポーネント設定ダイアログボックスに表示された、MapForce Design (.mfd) ファイルの場所に相対したファイルパスを保存します。この設定 は以下のファイルに影響を与えます: • XBRL タクソノミファイル • XBRL 入力ファイル • XBRL 出力ファイル • StyleVision スタイルシートファイル Uコンポーネント上で相対パスを使用するを参照してください。

6.8.7.3 全般 XBRL オプション

MapForce は、以下の全般的な(アプリケーション全体の) XBRL 設定を構成することができます:

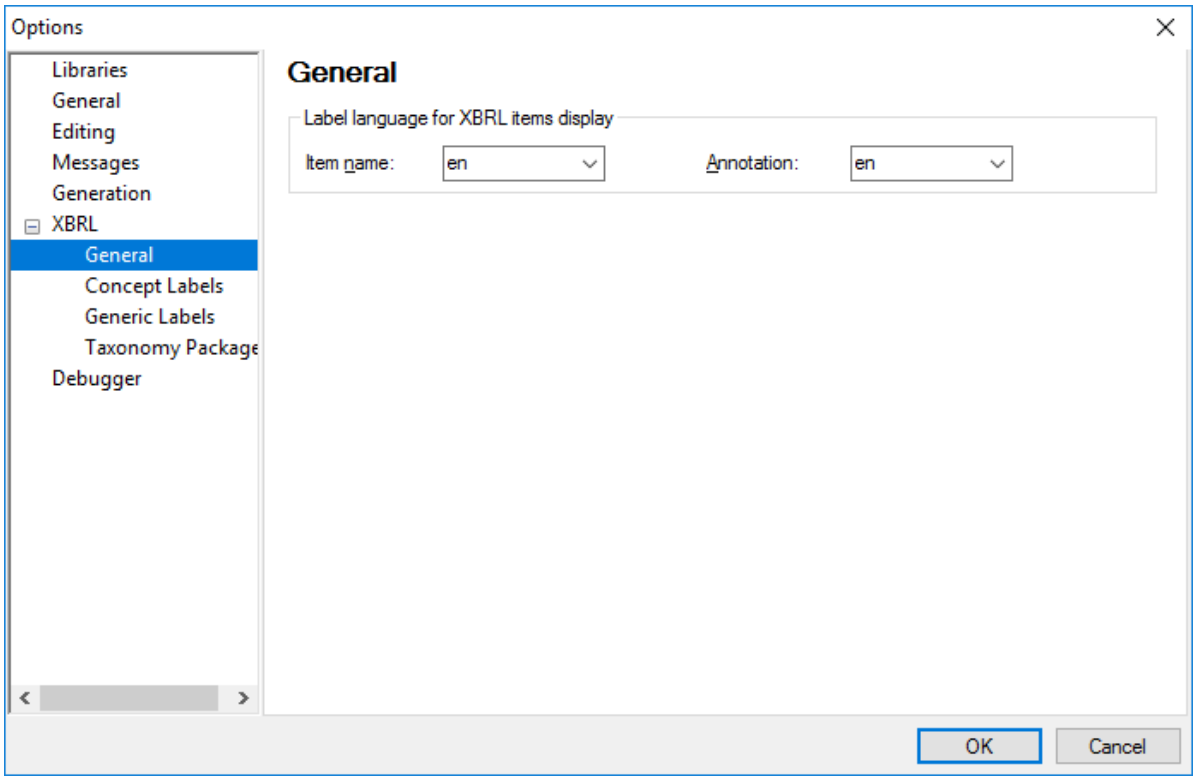
- XBRL アイテムのラベル言語と注釈を設定します。
- XBRL アイテム名のための優先されるラベルロールを設定します。
- XBRL アイテムのための注釈のラベルロールの特定の型設定します。
- カスタム XBRL タクソノミパッケージを追加します。

XBRL セクションのオプションダイアログボックスから上の設定すべてを構成することができます。

全般 XBRL オプションを変更する方法

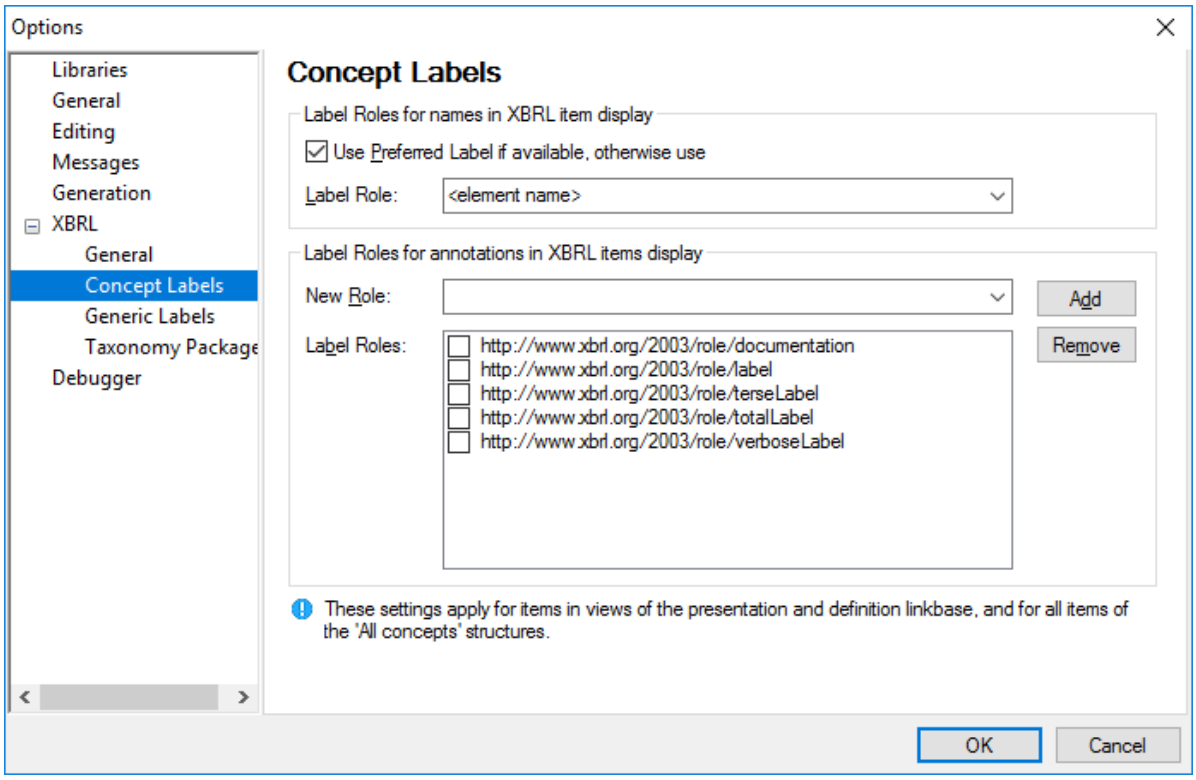
1. [表示] メニューから[XBRL 表示オプション]を開きます。
2. 以下の内の 1 つを行ってください：

   - XBRL アイテムのラベル言語と注釈を変更するには[XBRL | 全般]をクリックします。

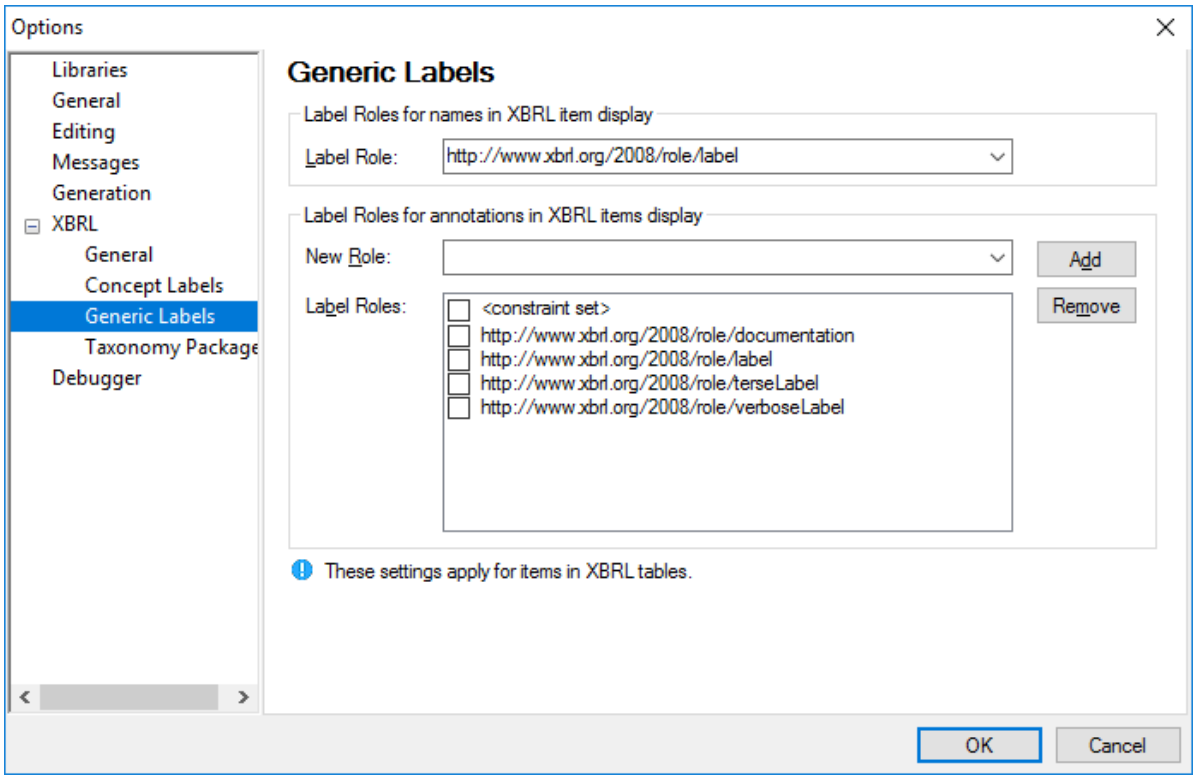


   - 以下の構成ビュー内のアイテムに適用することのできる優先順位を有するラベルロールと注釈を変更するには、[XBRL | 概念ラベル]:
     - *プレゼンテーション定義リンクベースからのビュー*
     - *全てのコンセプト（コンテキストの管理付き）*
     - *全てのコンセプト（未フォーマット）*

(C) 2015-2021 Altova GmbH



- *テーブルリンクベースからのテーブル内のアイテムに適用可能な優先ラベルロールと注釈を変更するには[XBRL | 一般ラベル]をクリックします*

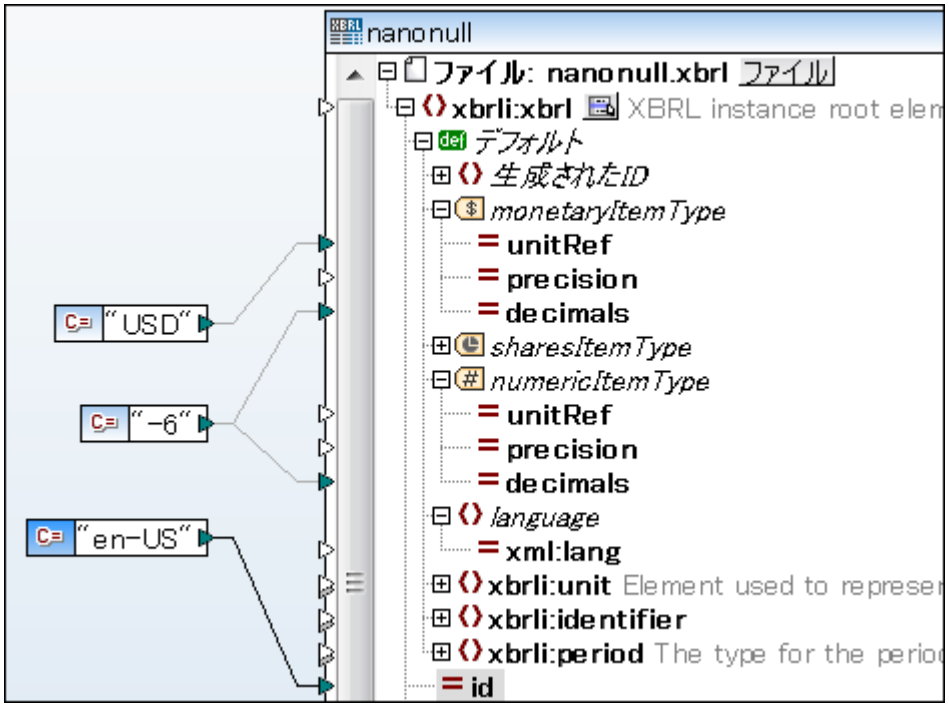


- XBRL タクソノミ パッケージを追加、または変更するには[XBRL | タクソノミ パッケージ] をクリックします(XBRL タクソノミ パッケージを参照してください)。

6.8.8 XBRL デフォルトと作業する

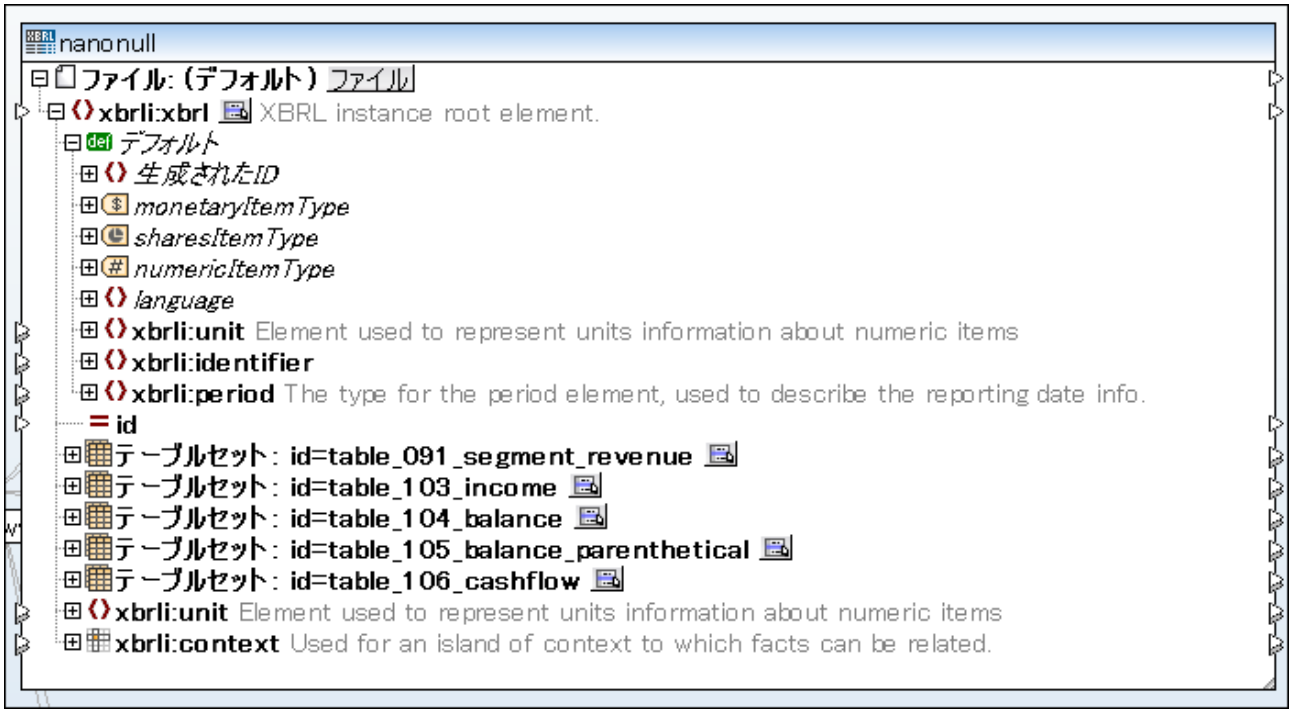
デフォルトは、XBRL コンポーネントの属性全てに対して明示的にマッピング接続することなく、に属性の値を割り当てることのできる強力な方法です。 例: （以下のスクリーンショットに表示されるように）定数ユニット識別子をデフォルトの階層内の`monetaryItemType` アイテムの属性 `unitRef` に割り当てると、入力が明示的に他の値にマップされている箇所を除いて、各金融 XBRL アイテムユニット識別子にデフォルトを割り当てます。下のスクリーンショットも、フットノートの言語を定義する`xml:lang` 属性のためのデフォルトの値の使用を表示しています。

(C) 2015-2021 Altova GmbH



デフォルトでは、"デフォルト" ノードは、新規のXBRL コンポーネント内で表示されています。しかしながら、「デフォルト」ノードを(XBRL ドキュメント全体のために) ルートレベルで、またはXBRL コンポーネント. の全ての階層レベルで表示することもできます。XBRL 構造内の異なるレベルで定義することができるため、異なるサブツリーは異なるデフォルトの値を持つことができます。

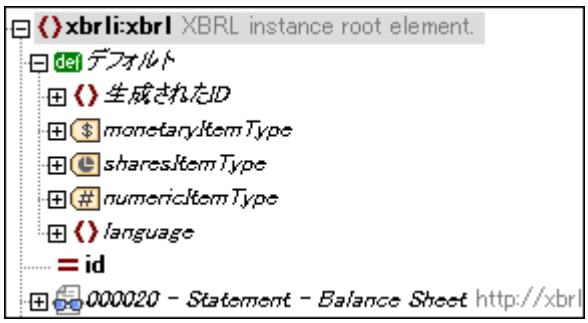
XBRL コンポーネントがプレゼンテーション/定義 リンクベースの代わりにテーブルリンクベースビューを使用する場合、テーブルリンクベースから、アスペクト値のデフォルトをマップすることができます。下のスクリーンショットでは、 `xbrli:unit`, `xbrli:identifier` と `xbrli:period` 要素は、XBRL テーブルにマッピングする際に使用することのできるデフォルトのアスペクト値です。



特定のアイテムのための"デフォルト"ノードを表示する

1. デフォルトのユニットを表示するアイテムを右クリックし、「XBRL | **デフォルトの表示**」を選択します（または、ノードのために使用できる場合は「**コンテキストメニューの表示**」（ ）ボタンクリックして、「 **デフォルトの表示**」）をクリックします。

   これは、複数のアイテムの型のためのデフォルトの値に接続することのできるデフォルトのアイテムを挿入します。

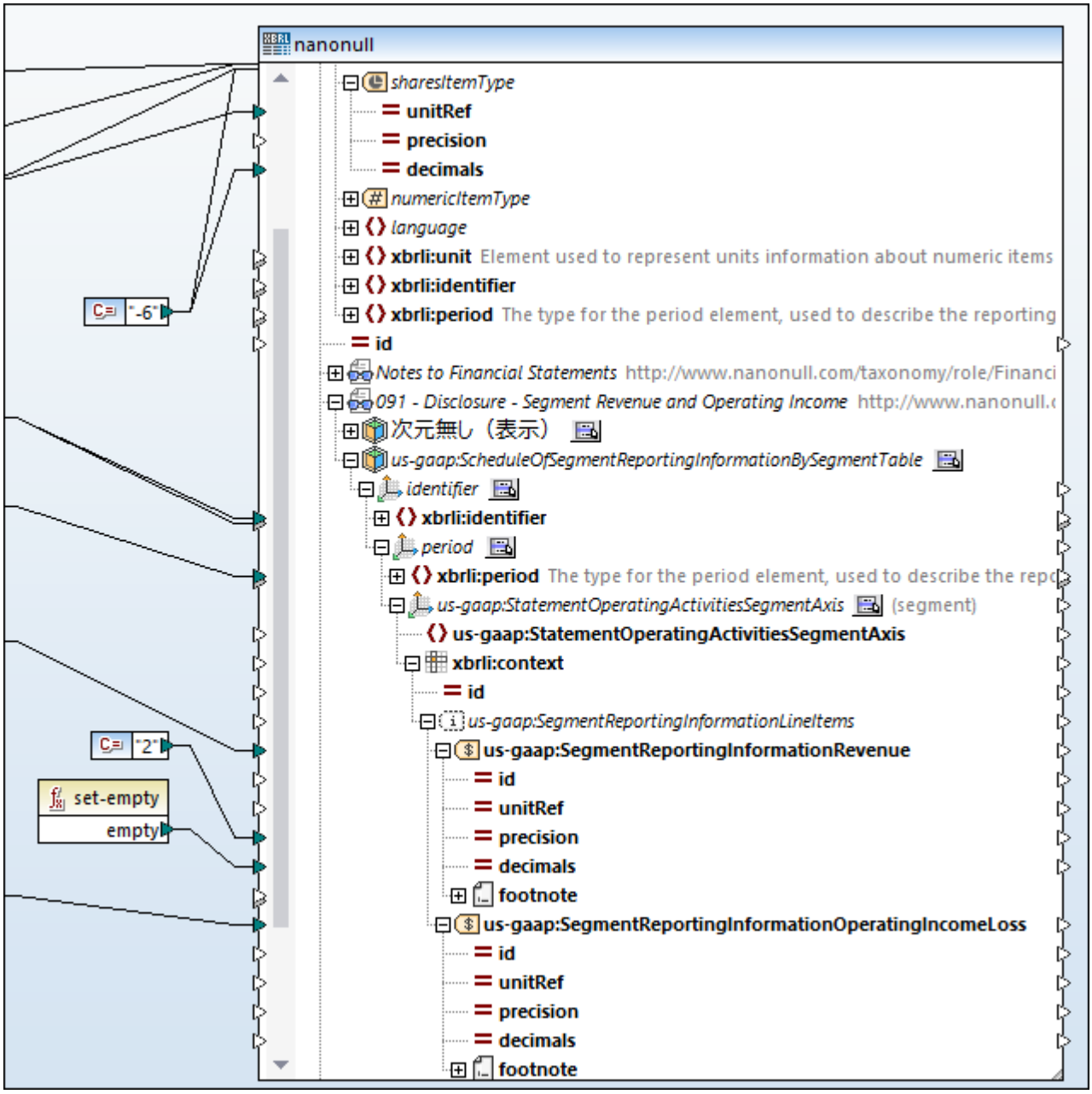


デフォルトの値を置き換える、または無効化する

デフォルトの値が一部のコンセプト属性、例えば、**decimals** のために定義されている場合、関数 **set-empty** を使用して、それぞれのコンセプトのためにこの設定を削除することが可能です。関数を参照してください。

デフォルトの値「-6」を金融アイテム「Segment Reporting Information, Revenue」のために無効化します。他のアイテムはのデフォルトの数値と共にレポートされますが、このアイテムは「precision」属性にマップされている「2」によりレポートされています。「Segment Reporting Information, Operating Income Loss」は「-6」のデフォルトの数値を使用してレポートされますが、このアイテムは属性にマップされているの精度によりレポートされます。

(C) 2015-2021 Altova GmbH



この結果、マッピングの出力内では、これらのアイテムは下で示されているようになります:

```
<us-gaap:SegmentReportingInformationRevenue contextRef="ctx1" unitRef="USD"
precision="2">23000988</us-gaap:SegmentReportingInformationRevenue>
<us-gaap:SegmentReportingInformationOperatingIncomeLoss contextRef="ctx1" unitRef="USD"
 decimals="-6">5605900</us-gaap:SegmentReportingInformationOperatingIncomeLoss>
```

コンテキストの扱い方

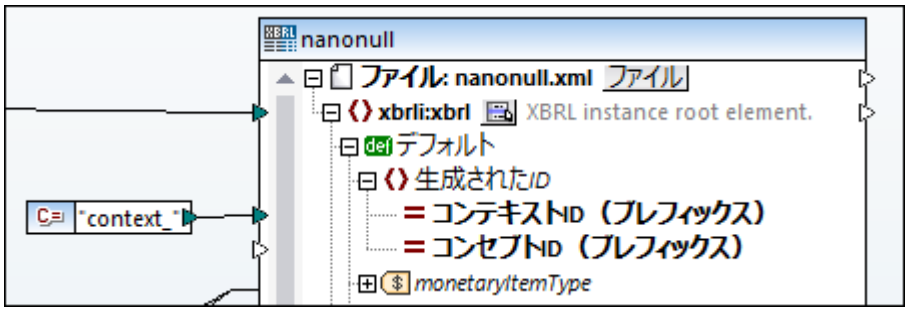
XBRL コンポーネント内の階層構造は、自動的なコンテキストの処理を許可します。XBRL 入力インスタンス内の `xbrli:context` の生成は、関連した概念をレポートする際、自動的に行われます。

XBRL インスタンス内の特定のコンテキストの属性 `id` の値は、XBRL 概念内の属性 `contextRef` の値です。MapForce は自動的に出力インスタンス内に作成された全てのコンテキストに番号を与えます。

"生成されたID "の下のノード "context id (prefix)"にテキストをプレフィックスとして割り当てることによりカスタマイズが可能になります。

例: 定数値 "**context_**" をデフォルトのプレフィックスとしてマッピングすると、値 "context_1"、"context_2"、"context_3" などの値を持つ連番のコンテキスト ID を出力インスタンス内に作成します。

デフォルトの値が定義されていない場合、MapForce はプレフィックス "**ctx_**"を持つコンテキスト ID を作成します。



出力 XBRL インスタンスが フットノート を含む場合、関連した概念は、自動的に生成されたフットノートのリンクをリンクするために概念 ID を持つ必要があります。これらの属性は自動的に生成されます。アイテム "**concept id (prefix)**" は、このようなプレフィックスを決定するためにポップされることができます。

XBRL コンポーネント内でプレフィックスがマップされない場合、MapForce は、プレフィックス "**fact_**" を持つ全てのコンセプト ID を作成します。

6.8.9 XBRL ハイパーキューブとの作業

XBRL ハイパーキューブアイテムにより、自動コンテキスト処理が有効になります。XBRL ハイパーキューブには以下のような種類があります:

- タクソノミで定義されたもの。例: Statement [Table]
- MapForce により生成されたもので、表示リンクベースから派生したデフォルトの次元である identifier と period を簡略化して表示します。例: Dimensionless (presentation)
- 全てのコンセプトノード内で、MapForce により生成されたもの。例: Dimensionless

ハイパーキューブには、identifier と period という 2 つのデフォルト次元が含まれており、各コンテキストにあるこれら 2 つの要素に対して簡単に読み書きを行うことができるようになります。更に、タクソノミ内で定義された次元が、コンテキスト要素の xbrli:segment ならびに xbrli:scenario に対して自動的に関連付けられます。

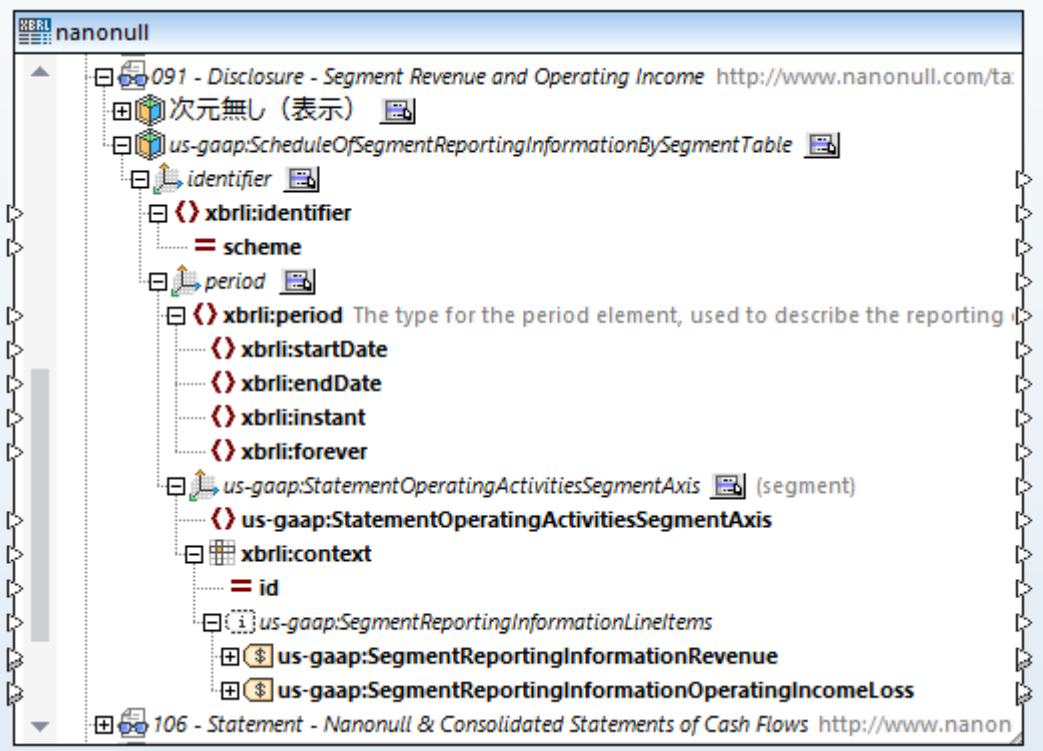
ハイパーキューブを表す「次元なし(表示)」では両方の次元が使用されます。コンテキストノード内にて階層構造で表示されるコンセプトの順序は、表示リンクベースから得られます。

(C) 2015-2021 Altova GmbH

「次元なし」ハイパーキューブアイテムでも両方のデフォルト次元が使用されますが、階層構造のコンセプトの順番は無く、タクソノミ内で定義された全てのコンセプトの元リストだけが表示されます。

その他のハイパーキューブは全てタクソノミ内で定義されており、タクソノミのラベルリンクベース内で定義された名前に従い表示されます。

ハイパーキューブとその次元(または軸)には、それぞれ小さなアイコン ic-dim-icon が表示されており、コンポーネント内にある各次元の表示を定義することができます。以下にあるイメージは、タクソノミファイルを開いた際のデフォルト状態を示します。



メモ 報告可能なコンセプトを含む全てのハイパーキューブがMapForce では表示されます。関連するハイパーキューブ次元がドメインを持たない場合、その次元はXBRL コンポーネントにて表示されません。

6.8.9.1 コンポーネント内で次元を表示する

XBRL 内にある次元アイテムは、インスタンス内部にて明示的または型付の値を参照します。各次元アイテムの注釈により、コンテキスト要素のxbrli:segment またはxbrli:scenario にて報告されるかを確認することができます。

型付次元により、XML スキーマ型の要素が表示されます。これらの値は直接マッピングされます。

XBRL タクソノミ内の明示的次元には特定のドメインから得られたxs:QName 型の値が与えられます。XBRL ドメインメンバー値と、XBRL ドメインアイテムそのものの値によりこの値は構成されます。明示的な次元は、マッピング、ならびにXBRL へマッピングしているその他のコンポーネント/構造間における必須要件に合わせて 2 つのモードで表示することができます。

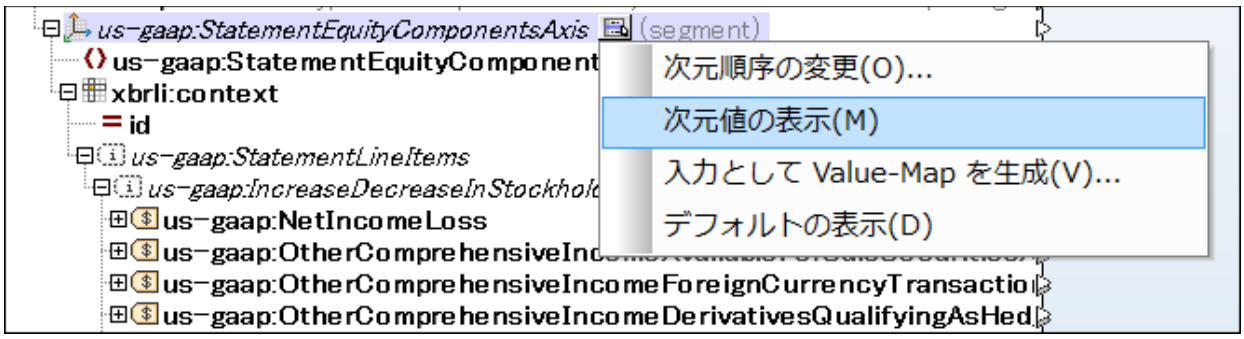
初期段階では、明示的な次元が単一の子ノードとともに表示され、この子へ直接マッピングすることができます(例: "Statement, Equity Components [Axis]")。

この方法は、データベースフィールドや、Excel テーブルのカラムといったソースデータ内にあるフィールドから、次元の値が派生するような場合に（XBRL ターゲットコンポーネントに対して）使用することができます。ソースデータには通常必要な QName データ型が含まれていないため、Value-Map 関数を使用してそれらを自動的に作成することができます（ハイパーキューブ次元のためにValue-Maps を生成するを参照）。

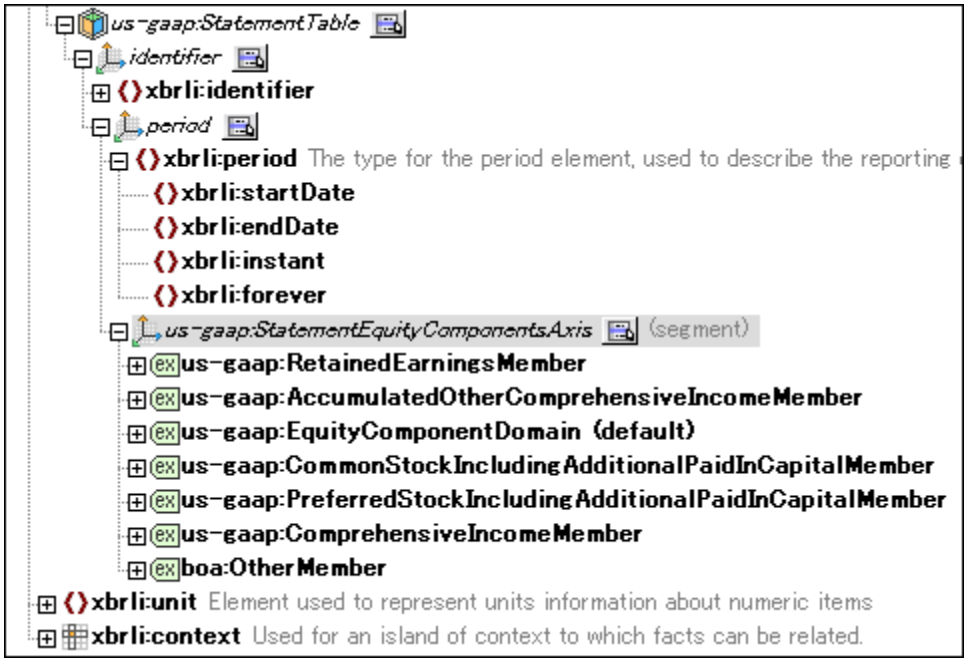
各次元メンバーに関連したファクトに対して異なるマッピングを行うため、次元ドメインにある個々の値に対して独立したノードを表示することができます。

コンポーネント内にある次元値を表示する:

1. 値を表示する次元のアイコン をクリックして、「次元値の表示」を選択します。

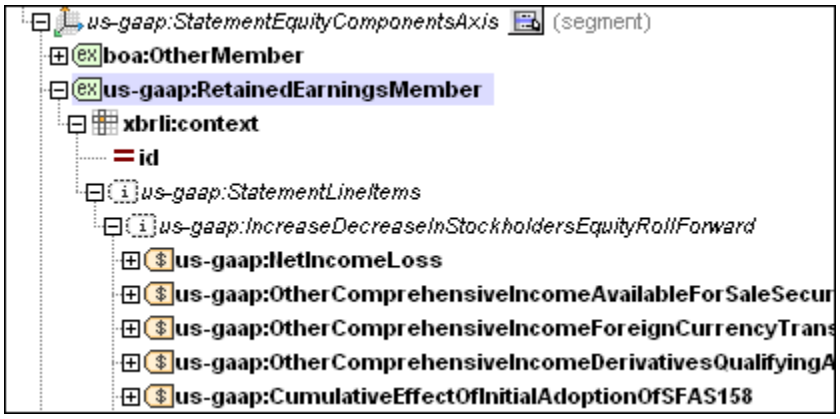


この操作により、次元名以下のアイテムが表示されます。次元のドメインならびにメンバーアイテムが表示され、それぞれ緑色のアイコンで表示されます。これらは全てドメインの明示的なメンバーで、アイテムアイコンには "ex" プレフィックスが表示されます。



明示的な各メンバーには、同じサブ構造が含まれており、個々に対して異なるマッピングを行うことが可能です。

(C) 2015-2021 Altova GmbH



コンセプトの出力に対してマッピングを行うと、これらの値が、適切な次元値を持つ関連するコンテキスト要素に対して使用されます。例えば、"Statement, Equity Components [Axis]" 次元において "Comprehensive Income [Member]" 次元値を含むコンテキストに対してのみ、インスタンス内にある "Net Income (Loss)" という値がマッピングされます。フィルタリングを行う必要はありません。

XBRL 出力インスタンスを記述する際には、コンテキスト内部における適切な次元値の自動生成がサポートされます。例えば、必須報告アイテムの "Net Income (Loss)" に対して、コンテキストノードの xbrli:context が、コンテキスト要素 (xbrli:segment) 内部にて "Comprehensive Income [Member]" という値を含む明示的な要素の "Statement, Equity Components [Axis]" を取得します。

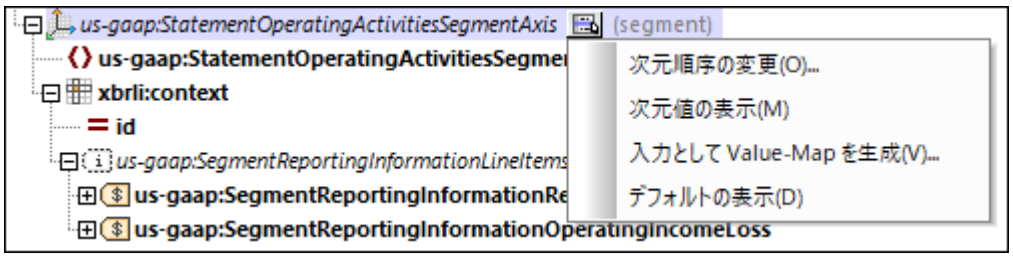
6.8.9.2 次元の順序を変更する

初期段階では、ネストされた子ノードというかたちでハイパーキューブにある全ての次元が表示され、階層構造が形成されます。ハイパーキューブ内部における次元の階層構造順序を変更することで、(非 XBRL の) マッピングに対応することができます。

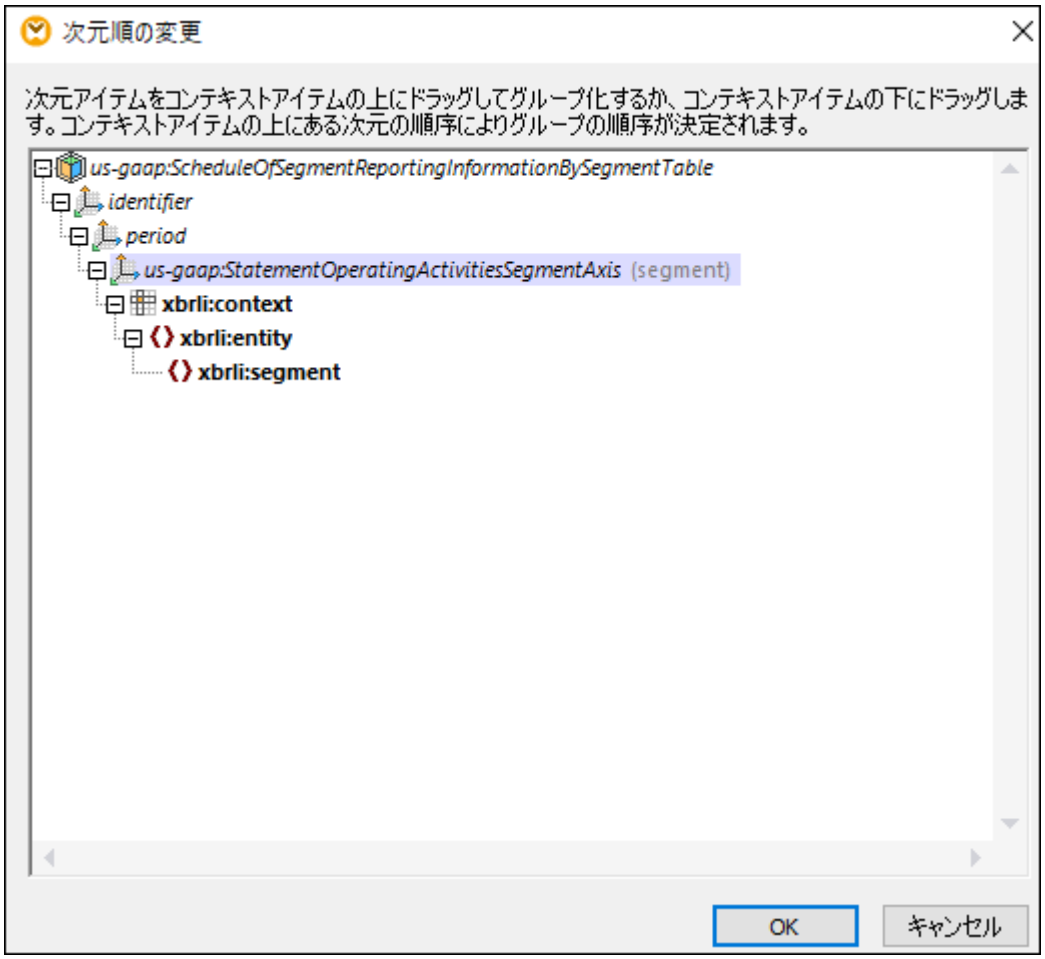
更に、特定のコンセプトに対して次元の値をセットする必要がある場合、MapForce では階層構造無しに次元を表示し、コンテキストノードの子要素として表示することができます。

次元アイテムの順序を変更する:

1. をクリックして、ポップアップメニューから次元順の変更エントリーを選択します。



この操作によりダイアログボックスが表示され、ハイパーキューブの次元位置を変更することができます。
ハイパーキューブには **identifier** と **period** という、2 つのデフォルト次元があり、ハイパーキューブ内部におけるこれら次元の位置も変更することができます。
2. ハイパーキューブ次元をクリックして、ドラッグアンドドロップによりその次元を移動します。次元をドロップすることができる位置にラインが表示されます。

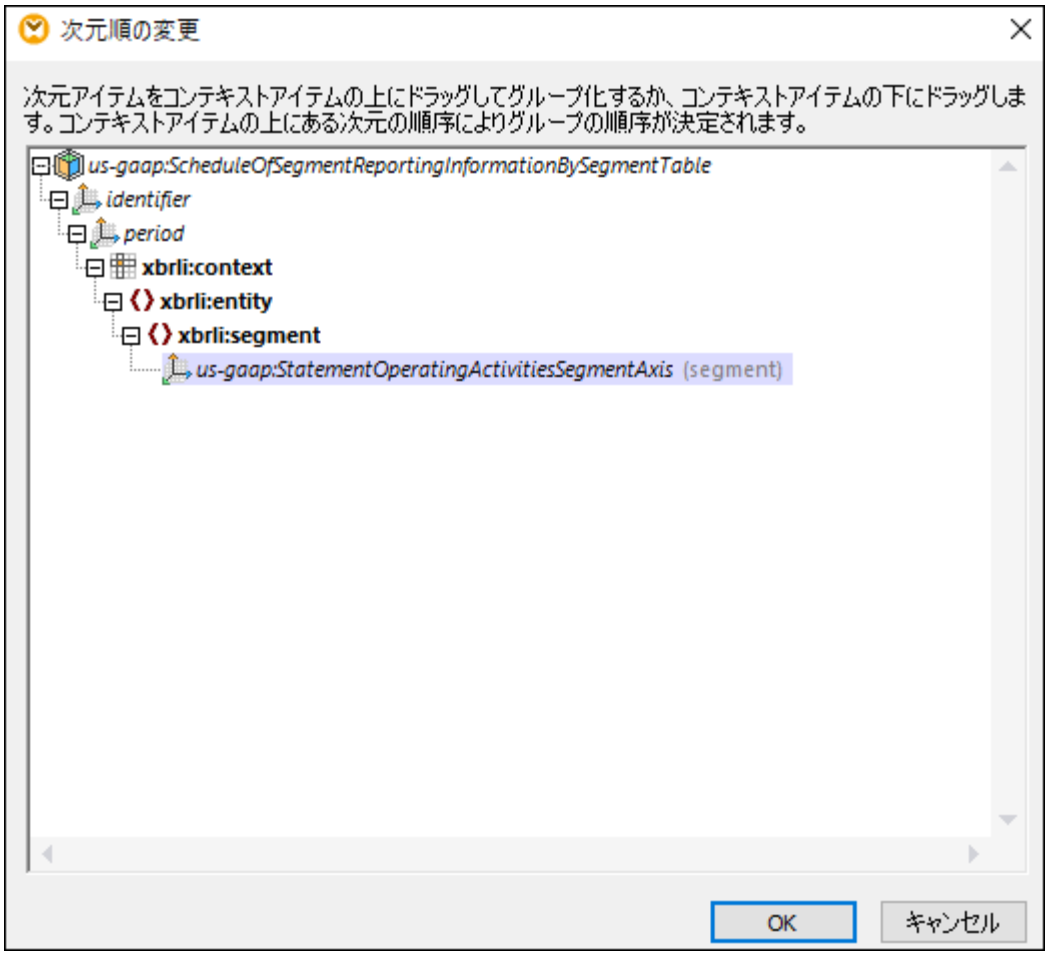


3. 「OK」をクリックすると、ダイアログが閉じられ、コンポーネント内の次元位置が変更されます。

階層構造から次元を除外する

- 次元を **xbrli:context** ライン以下にドラッグすると、次元がコンテキストアイテムに挿入されます。

(C) 2015-2021 Altova GmbH

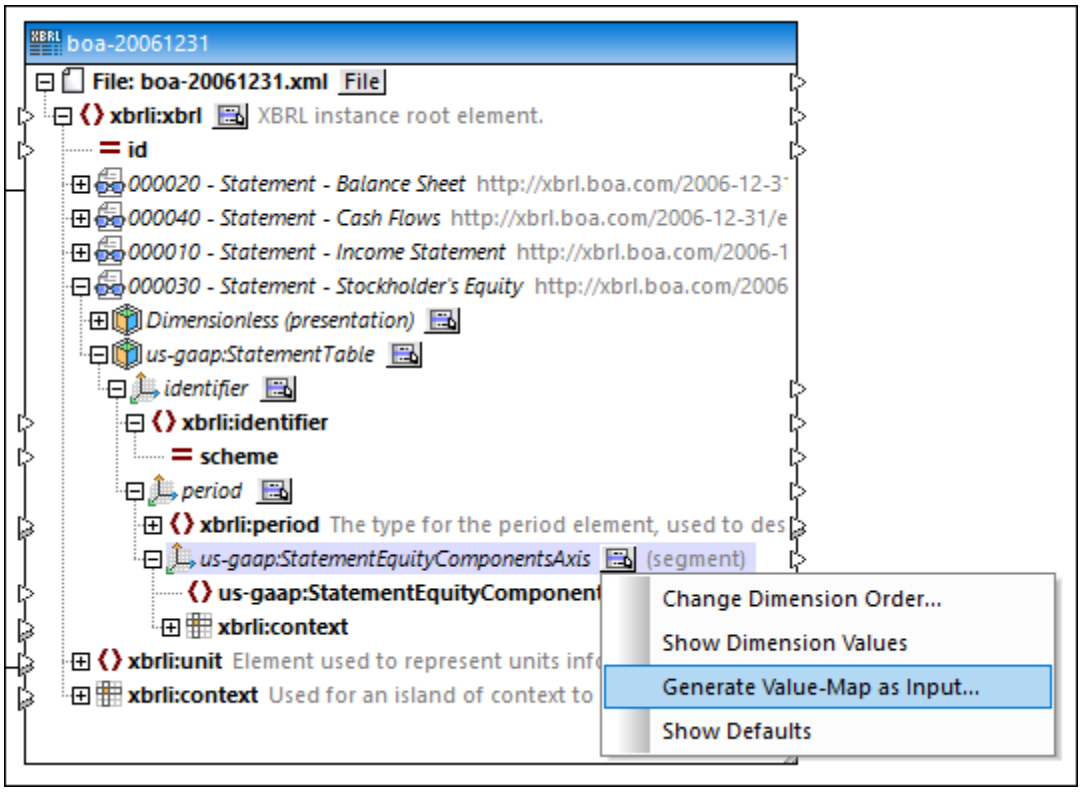


6.8.9.3 ハイパーキューブの明示的な次元のために Value-Maps を生成する

このオプションにより、任意の型の入力データをターゲットコンポーネント内における妥当な QName へ変換します。つまり、入力文字列をプレフィックス付きの名前（QName）へ変換します。

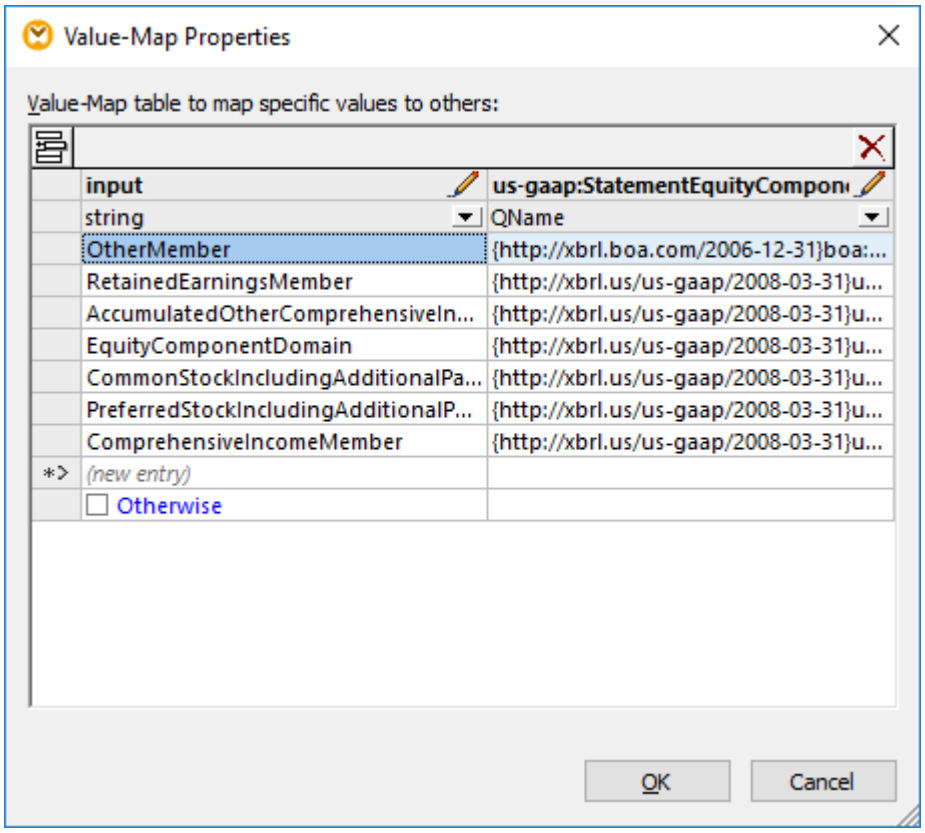
選択されたハイパーキューブ次元に対して Value-Map ルックアップテーブルを生成する

1. 次元値が特定のハイパーキューブ次元からは見えないようになっていることを確認してください。
2. 目的の次元にあるアイコン をクリックして、ポップアップメニューから入力として Value-Map を生成オプションを選択します。



この操作により、Value-Map プロパティダイアログボックスが開かれ、タクソノミにて定義された通り、次元のデフォルトドメインならびにドメインメンバーをベースに、自動的に生成された入力ならびに出力値が表示されます。

(C) 2015-2021 Altova GmbH



3. 必要に応じて入力値を編集した後、「OK」をクリックして Value-Map コンポーネントを挿入します。カラムのヘッダーテキストを（ヘッダーをダブルクリックした後に）編集して、必要に応じて更に短い、または詳細な名前に変更することもできます。

4. この操作により Value-Map 関数が挿入され、入力ならびに出力パラメーター名が表示されます。出力コネクタがターゲットのドメイン要素に対して自動的に接続されます。

5. 変換される入力データを含むソースアイテムを関数の入力パラメーターへ接続します。

コンポーネントの詳細に関しては、Value-Map の使用を参照ください。

6.8.10 XBRL テーブルと作業する

XBRL タクソノミがテーブルリンクベース（Table Linkbase Specification 1.0 を参照してください）を参照する場合、MapForce は、このテーブルリンクベース（テーブルまたはブレークダウンなど）に適用することのできるノードの型を表示するように構成することができます。これにより、表示されるテーブルにデータをマップすることができます

XBRL コンポーネント上で "テーブルリンクベースからのテーブル" ビューを表示する方法:

1. 「挿入」メニューから「XBRL ドキュメント」をクリックし、XBRL インスタンスファイルを参照します。XBRL ファイルをマッピングコンポーネントとして追加するを参照してください。

2. XBRL コンポーネントのルート要素上で、「**コンテキスト メニューの表示**」( ) ボタンを押し、「**構造ビューの選択**」をクリックします。
3. 「**テーブルリンクベースからのテーブル**」チェックボックスを選択し、「OK」をクリックします。

ビューに関する詳細に関しては、構造ビューの選択を参照してください。

メモ

- XBRL コンポーネントからデータを読み込む場合、(すなわち、XBRL コンポーネントがソースコンポーネントの場合)、コンテキスト メニュー から構造ビューの組み合わせを選択することができます。XBRL にデータを書き込む場合、**テーブルリンクベースからのテーブル** ビューが選択されている場合、技術的な理由から、これ以外のビューにデータを書き込むことはできません。
- 「**テーブルリンクベースからのテーブル**」ビューは"ビルドイン"が変換言語として選択設定されている必要があります(変換言語の選択も参照)。

Table Linkbase からの情報を表示するには、MapForce は通常、以下の例外を除いて、構造モデルを使用します:

- マージされたルートノードは表示されます
- 兄弟を持たないロールアップノードは、個別の階層レベルを作成しません。

構造モデルは、Table Linkbase Specification 1.0、§4 で定義されている3つのデータモデルうちの1つです。マッピングにより参照される構造モデルの一部を識別するために、MapForce は、定義モデルからの情報も使用します。

MapForce コンポーネント 構造は一次元的ですが、テーブルは3つ(x、y、z)までの次元を持つことができ、MapForce は、テーブルの次元を全てのブレークダウンを共にネストすることにより、以下のデフォルトの順番(z、y、x) で表示します。MapForce内のブレークダウンノードの目的は、各ブレークダウンの開始点に関する情報を提供するためです。

このセクションは、次のテーブルリンクベースからのエンティティとの作業方法の命令を含みます:

- ブレークダウンの表示または非表示

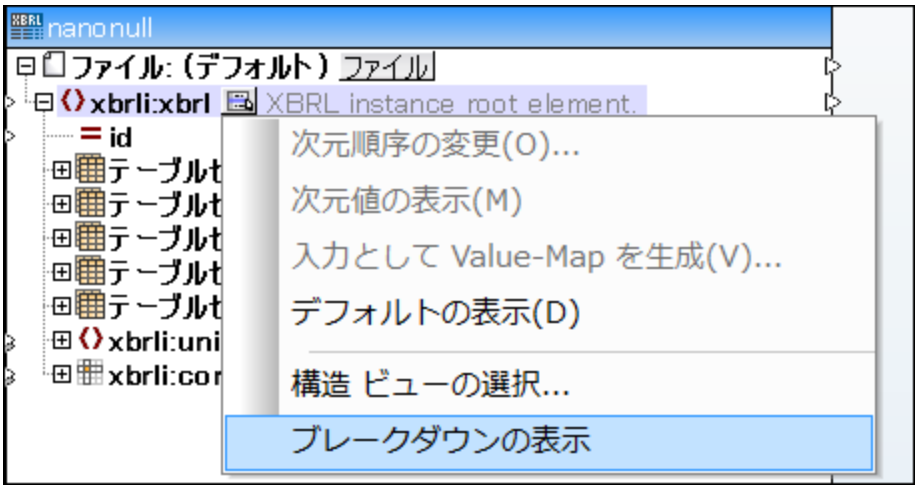
- ブレークダウンの順序の変更
- パラメーターと作業

6.8.10.1 ブレークポイントを表示または非表示

テーブルリンクベースを参照するXBRLドキュメントでは、コンポーネント内でブレークダウンを表示、または非表示するかを選択することができます。

ブレークポイントを表示または非表示

1. XBRLコンポーネント内で、「**コンテキストメニューの表示**」( ) ボタンが右横に表示されているノードを探します(例: root node)。



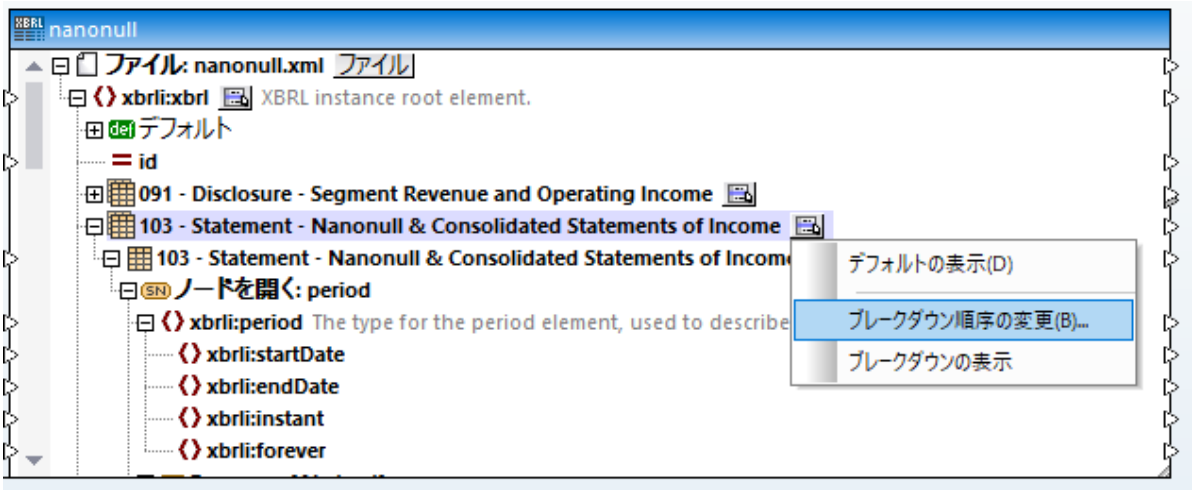
2. 「**コンテキストメニューの表示**」( ) ボタンをクリックして、「**ブレークダウンの表示**」をクリックします。

6.8.10.2 ブレークダウンの順序の変更

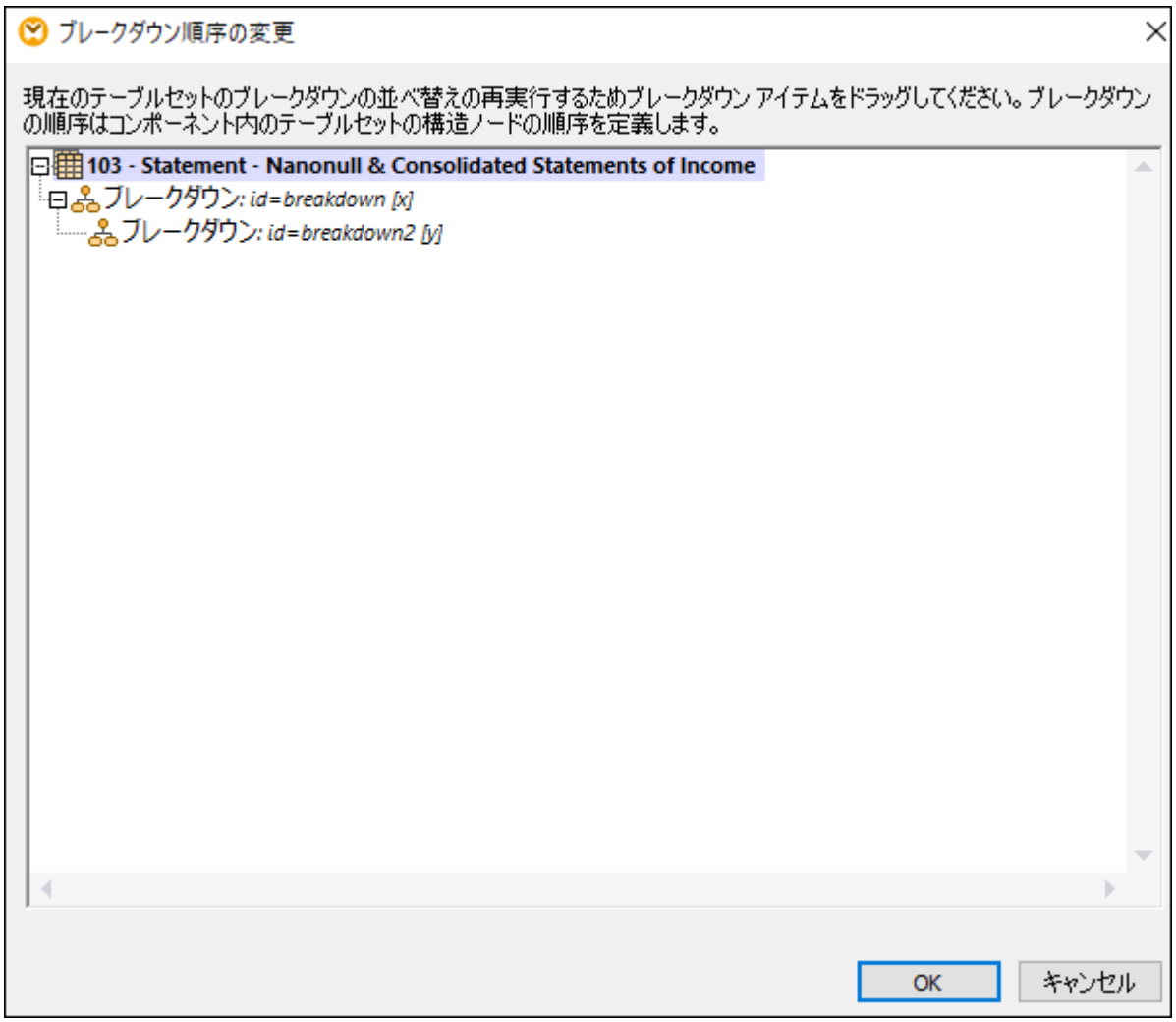
テーブルリンクベースを参照するXBRLドキュメントではブレークダウンの順序を変更することができます。

ブレークダウンの順序の変更方法

1. XBRLコンポーネント内でブレークダウンの順序を変更するノードをロケートします。
2. ノードの右にある「**コンテキストメニューの表示**」( ) ボタンをクリックして、「**ブレークダウンの順序の変更**」をクリックします。(「**ブレークダウンの順序の変更**」メニューオプションが使用することができない場合、これは選択されたノードにとにこのオプションが有効でないことを意味します)。



3. ブレークダウンを希望する場所にドラッグします（例: 以下のスクリーンショットでは、*y* ブレークダウンを *x* ブレークダウンにドラッグします）。



4. 「OK」をクリックします。

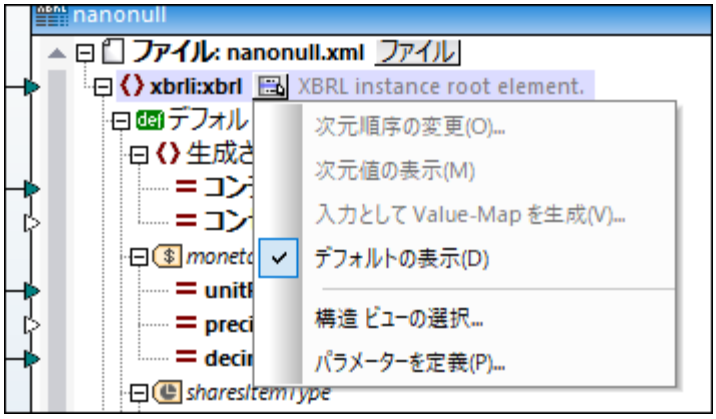
6.8.10.3 パラメーターと作業

XBRL ドキュメントがテーブルリンクベースを参照し、パラメーター（Table Linkbase Specification 1.0, §5.3）を含む場合、パラメーターの値を変更することができます。コンテキストにサポートされる場合、新しいパラメーター値を既存のパラメーターに追加することができます。

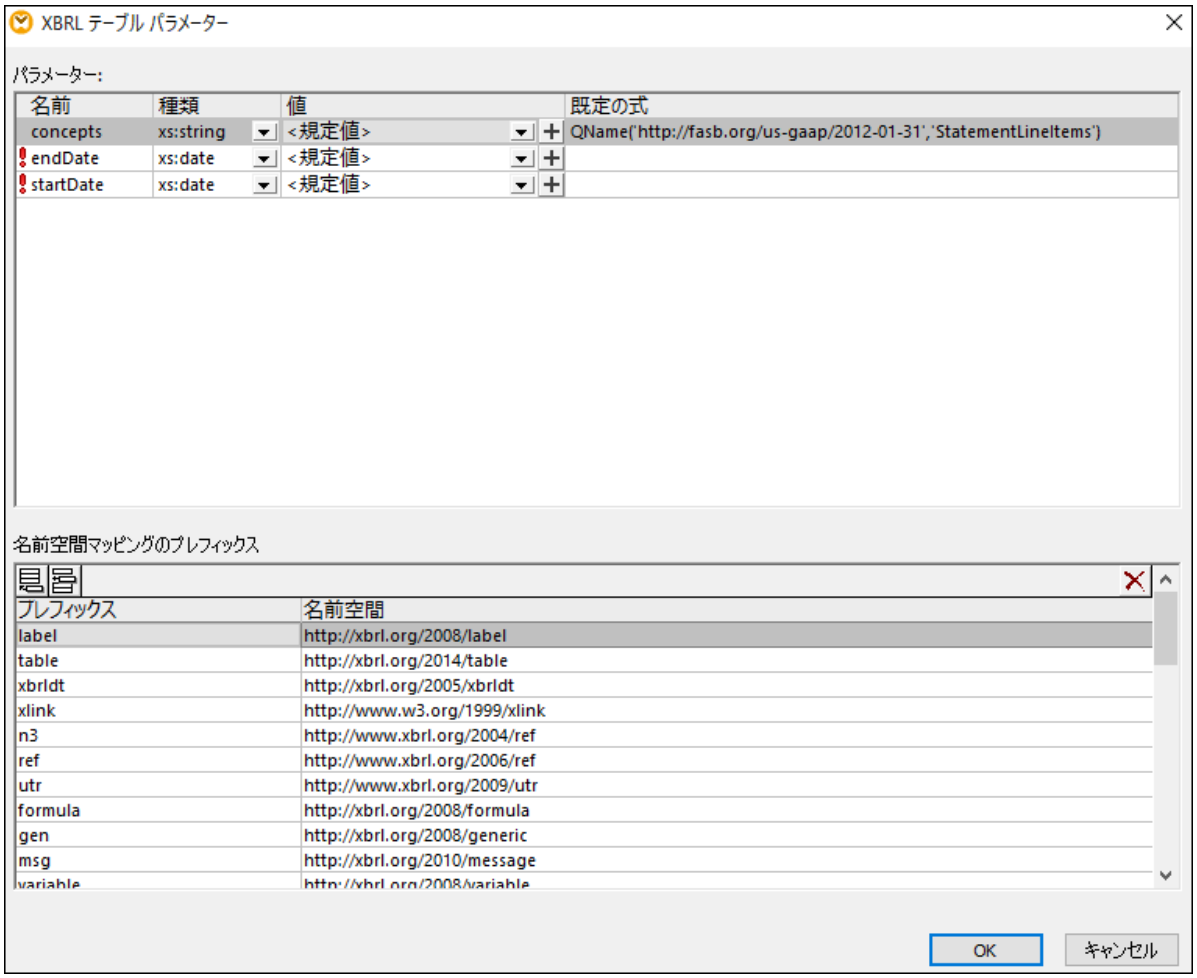
XBRL テーブルパラメーターダイアログボックスから、パラメーター値を変更することができます（下の命令を参照）。このダイアログボックスは、グローバルレベル（検知できるタクソノミセット内どこでも）と個別のテーブル内で定義されるパラメーターを表示します。パラメーターの値を変更し、ダイアログボックスを閉じると、MapForce は、コンポーネント内のすべてのテーブルを再生成し、構造ツリーノードをマージしようと試みます。パラメータが不足するために再生成が行われない場合、コンポーネントの対応するノード内にエラーノードが表示されます。エラーに関する詳しい情報は、 メッセージウィンドウのチェックボタンをチェックします。

パラメーターの値を変更する方法:

1. ルートノードの右の「**コンテキストメニューの表示**」（ ）ボタンをクリックして、「**パラメータの定義**」 をクリックします。「**パラメータの定義**」メニューオプションが使用できない場合、テーブルリンクベースはパラメーターを含みません。



2. 新しい列内の新しい値のパラメーターレコードを選択します。必須のパラメーターは感嘆符（ ）アイコンと共に表示されます。



既存のパラメーターにパラメーター値を追加する

パラメーターレコードを選択して、値 列の下の「**値の追加**」( ) ボタンをクリックします。パラメーターの型が複数の値の追加をサポートする場合のみ値の追加が可能になることご注意してください。

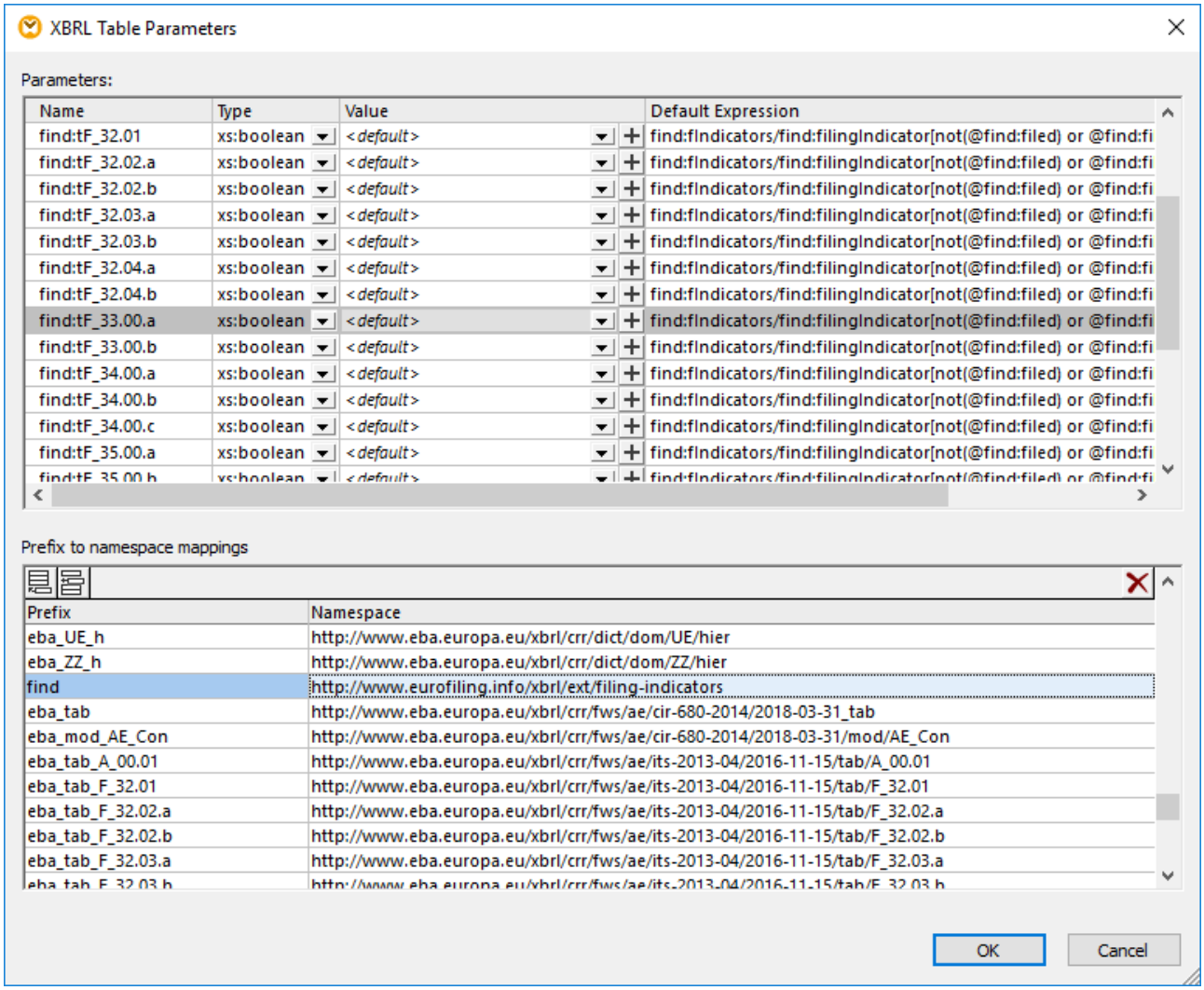
追加された値を削除する

削除する値の横の「**値の削除**」( ) ボタンクリックします。

名前空間マッピングのプレフィックス

テーブルパラメータ名は任意の名前空間とローカル名により構成されています(すなわち、これらはQNames で、書式 *namespace_prefix:localname* を取ります)。MapForce はこのようなマッピングを自動的に検知し、XBRL テーブルパラメーターダイアログボックスの下のエリアに表示します。例えば、下のイメージでは、 プレフィックス **find** は **http://eurofiling.info/xbrl/ext/filling-indicators** にマップされます。

(C) 2015-2021 Altova GmbH



カスタム名前空間に従い、MapForce がパラメータープレフィックスを解析する必要がある場合、既存のプレフィックス名前空間ペアを変更、または カスタムのものを追加することができます。名前プレフィックスマップを作成するには、「**入力**」 または「**追加**」 ボタンをクリックします。マッピングが XBRL インスタンスにデータを書き込む場合、これは XBRL インスタンスアイテムの名前プレフィックスに影響を与えないことにご注意してください。これはダイアログが閉じられた後 MapForce がどのように解析し、XBRL 構造を生成するかの表示としての役割を果たすだけです。

6.8.11 XBRL マッピングのサンプル

このセクションは、以下の XBRL マッピングのサンプルを含みます:

- Microsoft Access から XBRL へマッピング (Microsoft Access データベースからのデータが XBRL タクソノミにマップされる方法を示し、有効な XBRL インスタンスファイルを生成します)。
- Microsoft Excel から XBRL へマッピング (Microsoft Excel スプレッドシートからのデータが XBRL タクソノミにマップされる方法を示し、有効な XBRL インスタンスファイルを生成します)。

6.8.11.1 Microsoft Access から XBRL へマッピング

データベースのデータを抽出するための様々な関数やフィルターを使用したこのサンプルは ...\MapForceExamples フォルダー内に収められている **DB_to_XBRL.mfd** ファイルにて確認することができます。

タクソノミの **nanonull.xsd** は US:GAAP から派生したものです。マッピングにより、1 つの Disclosure と 3 つの Statement に対して、全てのエンテキスト、コンセプト、ユニット、そしてフットノートを含む XBRL 出力インスタンスが作成されます。

レポート **"091 - Disclosure - Segment Revenue and Operating Income"** では、MapForce で次元の値をマッピングする方法が示されます。ハイパーキューブの "us-gaap:ScheduleOfSegmentReportingInformationBySegmentTable" には明示的な次元の "us-gaap:StatementOperatingActivitiesSegmentAxis" が含まれています。

このドメインは、"nanonull:USA"、"nanonull:Europe"、そして "nanonull:Asia" という 3 つの次元値によりタクソノミ内にて拡張されます。マッピングでは、Value-Map によりデータベーステーブルの "Region" にある "Name" カラムの値が、QName 型で目的の次元値へ変換されます。

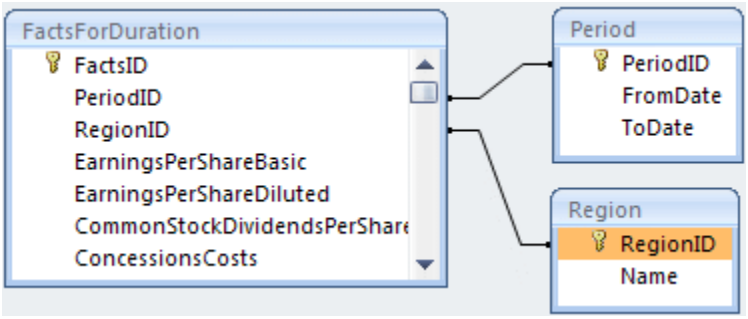
レポート **"106 - Statement - Nanonull and Consolidated Statement of Cash Flows"** では、MapForce で期間とインスタンス期間の両方に関連した出力インスタンス内へ、ファクトを書き込む方法が示されます。

マッピングに示される通り、階層構造における period アイテムを複製することで "Cash and cash equivalents at beginning (end) of period" というファクトが作成されます。

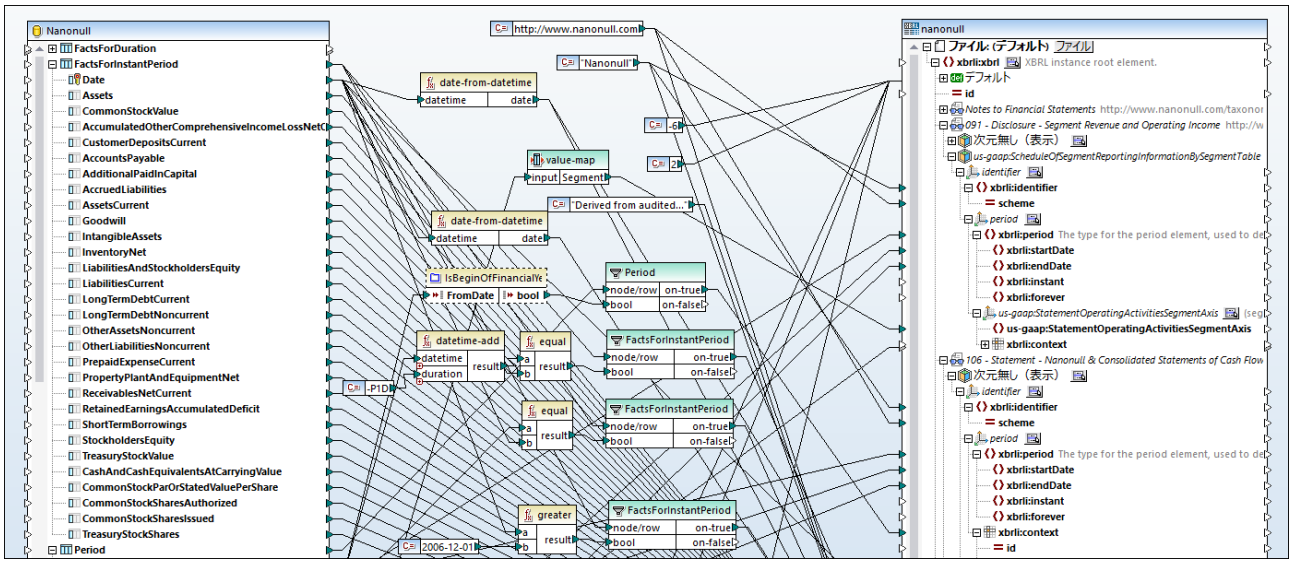
マッピングにより出力インスタンス内にて、"USD" と "perShareItemType" という 2 つのユニットが定義されます。このためには、xbrli:unit 要素を複製する必要があります。XBRL ライブラリから "xbrl-measure-currency" と "xbrl-measure-shares" 関数を使用することで、関連する measure 要素が作成されます。

instant または duration 期間に関連するかにより、データベーステーブル内のファクトが分割されます。

- **FactsForInstantPeriod** テーブルは、フラットな値のテーブルになります。
- **FactsForDuration** テーブルは階層構造になっていて、中に含まれる各ファクトが、特定の PeriodID ならびに RegionID に関連付けられます。



Period テーブルでは FromDate ならびに ToDate により、期間の開始ならびに終了日が定義されます。Region テーブルにより、ファクトが各地域(Asia、Europe、または USA)へ関連付けられます。



DB_ to_ XBRL.mfd マッピング

XBRL インスタンスファイルでは、以下のXBRL アイテムが必要になります:

- **unitRef** と、monetary コンセプト内の**decimals** または**precision** のどちらか
- 次元のスキームとxbrli:**identifier**
- xbrli:**period** と、xbrli:instant または xbrli:**startDate**/xbrli:**endDate** 要素のどちらか
- xbrli:**unit** 要素内における xbrli:**measure** ならびにxbrli:**id**

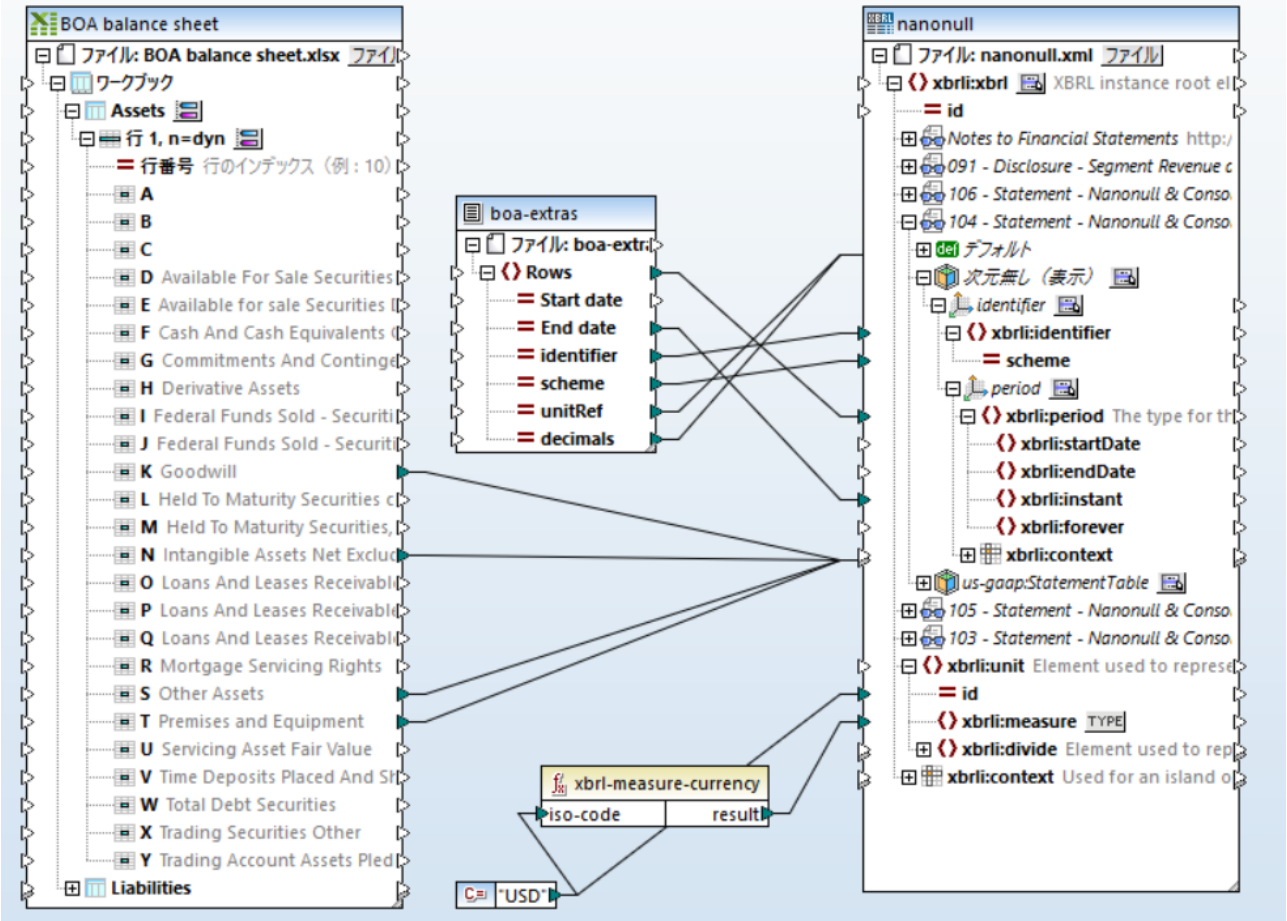
Microsoft Excel からXBRL へマッピング を参照してください。

6.8.11.2 Excel から XBRL へのサンプル

XBRL インスタンスファイルを生成するためにExcel シートからデータがXBRL タクソノミにマップされている状態をこのサンプルは表示しています。実際のマッピングデザインファイルは次のパスで検索することができます: **<マイドキュメント>\Altova\MapForce2021\MapForceExamples\Tutorial\boa-balance-sheet.mfd**。

マッピングをシンプルに保つため、"Assets" ワークシートから得られたデータだけがマッピングされています。Dec. 31st 2012 というインスタンスに対して Assets を含んでいる、妥当な XBRL インスタンスがマッピングの結果として生成されます。

ターゲットコンポーネントのためのXBRL タクソノミは次のパスで見つけることができます: **<マイドキュメント>\Altova\MapForce2021\MapForceExamples\\Taxonomy\Nanonull.xsd**。これは内部が簡素化されたデモタクソノミです。実際のタクソノミとは異なり、マップが可能なくつかのXBRL アイテムが含まれています。ターゲット XBRL コンポーネントは、プレゼンテーションと定義リンクベース（ハイパーキューブ）を表示するように設定されています。構成ビューの選択を参照してください。



boa-balance-sheet.mfd マッピング

このサンプルでは、アルファベット順で表示されているExcel の列名を、タクソノミコンポーネント内にあるXBRL アイテムファクト（Cast ならびにCash Equivalents）へ直接マッピングする方法について説明します。

ソースコンポーネントとターゲットコンポーネント内にあるアイテムの名前を一致させる必要は無いという点に留意してください。ターゲットタクソノミアイテムの名前により、結果 XBRL インスタンス内で使用される名前が決定されます。ソースとターゲットアイテムの名前が一致している場合、一致した名前の子アイテムを自動的に接続する、**子の自動接続機能**を利用することができます（一致する子を接続）を参照してください

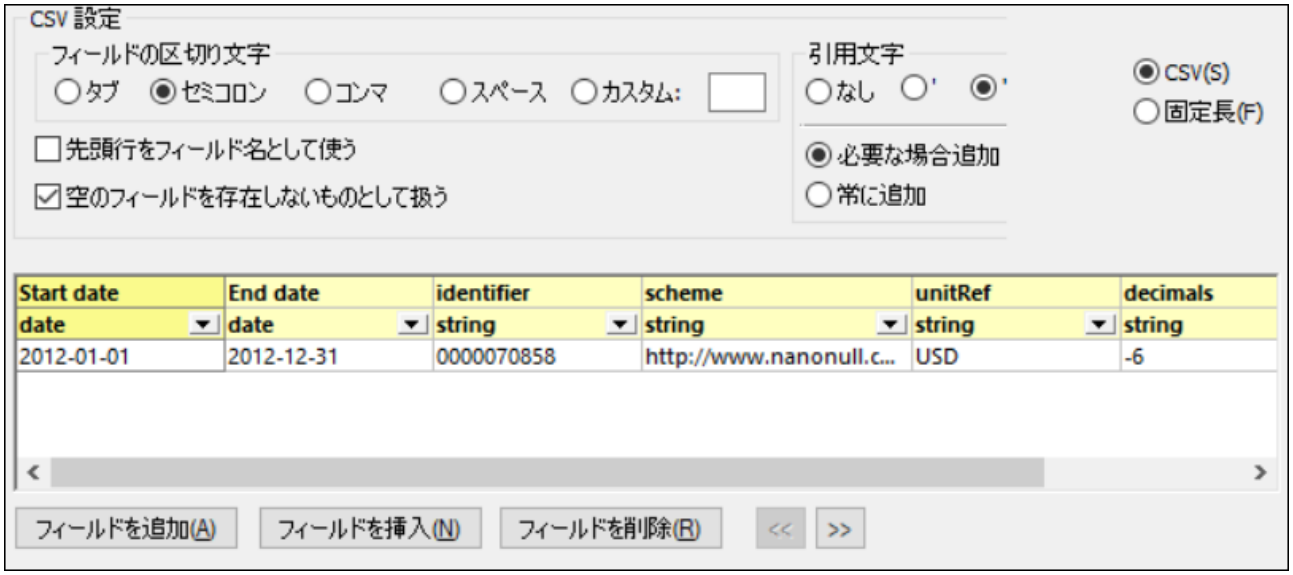
有効な XBRL インスタンスファイルを生成可能にするためにマップされる必要のあるアイテムが多種存在します。テキストファイルにより **Dimensionless (presentation)** ハイパーキューブ内の**デフォルト** アイテムの子要素にフィールドがマップされる必須ファイルの一部が提供されます（boa-extras.txt はマッピング上の中間のコンポーネントです）。

ハイパーキューブ次元を右クリックして、「XBRL | **デフォルトの表示**」を選択するか、アイコン をクリックして表示される同名のオプションを選択することで、「**デフォルトの**」アイテムを挿入することができます。詳細に関しては、XBRL デフォルトとの作業を参照してください。

次のXBRL アイテムは有効な XBRL インスタンスファイルを生成するためにマップされています:

- **unitRef** と、monetary コンセプト内にある **decimals** または **precision**
- identifier 次元のスキーマならびにxbrli:**identifier**
- xbrli:**period** と、xbrli:instant またはxbrli:**startDate**/xbrli:**endDate** 要素のどちらか
- xbrli:**unit** 要素内にあるxbrli:**id** ならびにxbrli:**measure**

必須アイテムの一部のために テキストファイルはデータを提供します。"boa-extras" コンポーネントを右クリックして、値を確認、または変更することができ、コンテキストメニューから「プロパティ」を選択します。



定数により、**id** ならびに**measure** アイテムのデータが与えられ、タクソノミコンポーネントのベースに使用されます。`xbrl-measure-currency` 関数は、**xbrli:measure** フィールドのために適切な値に為替コードを変換します。MapForce 関数に関する詳細は、関数 チャプターを参照してください。

出力ボタンをクリックして、XBRL インスタンスファイルを確認します。

- ツールバーにある出力の検証ボタン をクリックすることで、XBRL インスタンスの検証を行うことができます。メッセージや警告がメッセージウィンドウにて表示されます。
- テキストビュー設定ボタン をクリックすることで、生成された出力に適用できる多種の設定を設定することができます。

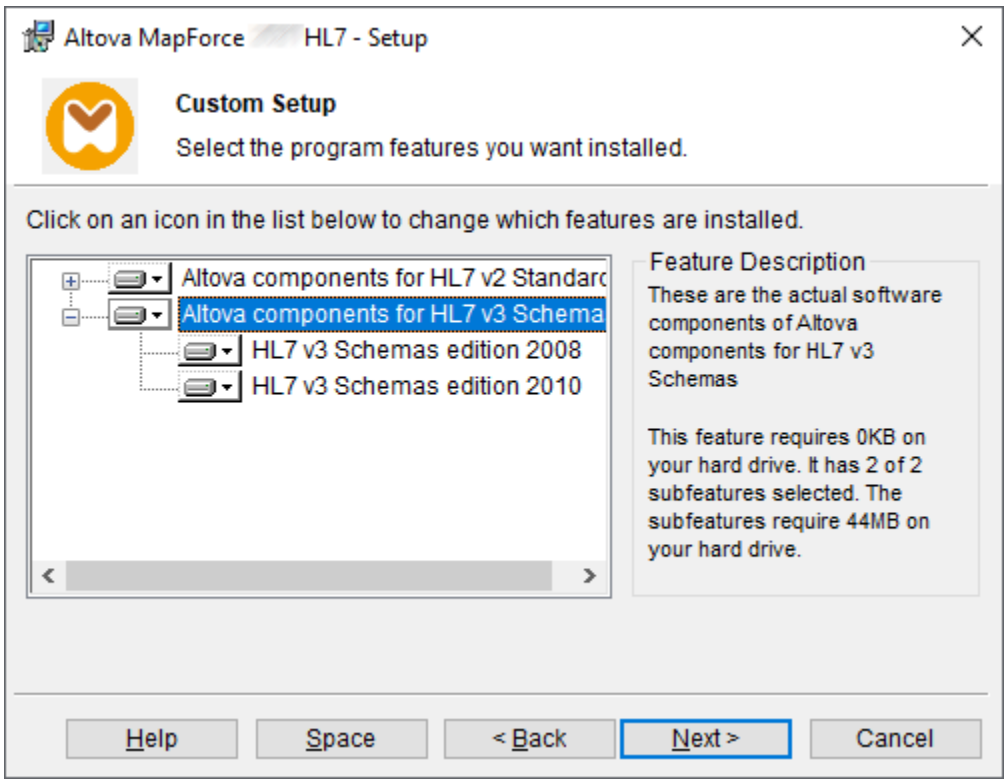
6.9 HL7 Version 3

MapForce にはHL7 バージョン3 ドキュメントからのデータ、またはドキュメントへのデータのマッピングがサポートされています。具体的にはMapForce のエディションによりサポートされているコンポーネントが他からデータをHL7 3.x インスタンスファイルへマップ、または、その逆を行うことができます。

The HL7 バージョン3 標準はHL7 バージョン2.とは異なりXML をベースにしています。このためHL7 バージョン3 コンポーネントを他のXML スキーマに追加する方法と同じ方法でマッピングに追加することができます。MapForce はHL7 バージョン3 スキーマをデフォルトでは追加しないことにご注意してください。しかしながら以下の手順に従いHL7 バージョン3 スキーマを個別にダウンロードしインストールすることができます。

HL7 バージョン3 XML スキーマのインストール方法:

1. [HL7 EDI 構成ファイル] と呼ばれるパッケージをAltova ダウンロードセンター https://www.altova.com/mapforce/download#mapforce_libraries で[MapForce ライブラリ]ページからダウンロードします。このパッケージは(EDI-ベースである) HL7 バージョン2 構成ファイルとHL7 バージョン3 XML スキーマを含んでいます。
2. ダウンロード済みの実行可能ファイルをダブルクリックしインストールウィザードステップに従います。
3. セットアップ型を選択するようにプロンプトされると**カスタム** を選択し、**次へ**をクリックします。
4. HL7 バージョン3 スキーマのインストールを選択します。例:



インストール後、HL7 バージョン3 XML スキーマを以下のディレクトリで使用できるようになります: **C:\Program Files\Altova\Common2021\Schemas\hl7v3**.

メモ 32-ビット MapForce を64-ビット オペレーティングシステム上で作動している場合、パスを**C:\Program Files(x86)\Altova\Common2021\Schemas\hl7v3** に調整してください。

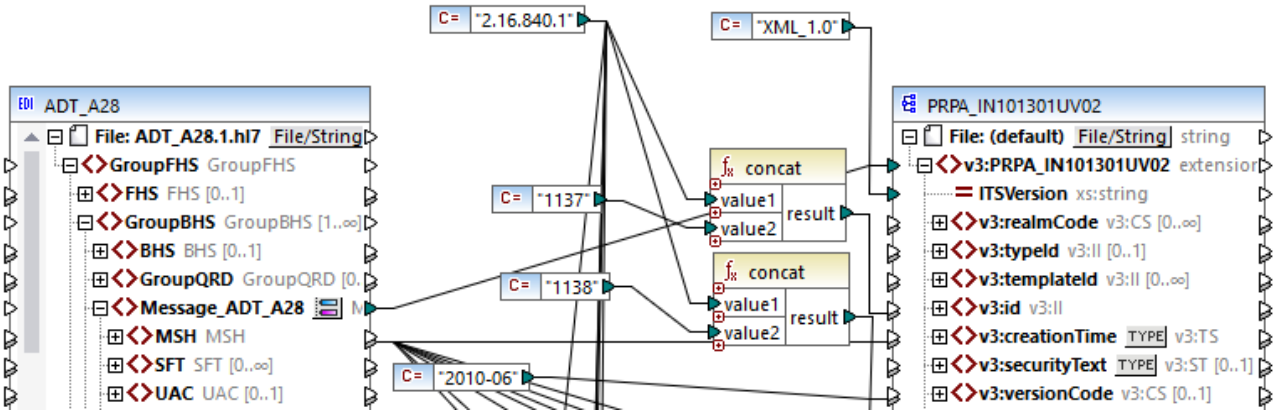
(C) 2015-2021 Altova GmbH

HL7 バージョン 3 XML スキーマをインストールすると、マッピングを以下のように開始することができます:

1. MapForce を実行する
2. 「挿入」メニューから「XML スキーマ/ファイル」をクリックします。
3. スキーマファイルを参照します。例えば、**PRPA_IN101301UV02.xsd file** を上のインストールディレクトリから選択することができます。
4. データを読み取り他のフォーマットにマップする場合、インスタンスファイルを提供するようにプロンプトされると前に選択されているメッセージ型に対応する HL7 バージョン 3 インスタンスファイルを選択します。それ以外の場合(データを読み取るのではなく)インスタンスファイルを生成する場合 スキップしてください。

マッピングに HL7 バージョン 3 コンポーネントを追加すると実際のマッピングの接続のデザインを継続することができます。MapForce を使用したマッピングのデザインの紹介に関しては チュートリアルを参照してください。

HL7 バージョン 2.6 ファイルから HL7 バージョン 3 ファイル へデータを変換するマッピングのサンプルを次のパスから使用することができます: <マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥**HL7V260_To_HL7V3.mfd**。このマッピングのフラグメントは以下で示されています。



上で示されるマッピング内ではソースコンポーネントは HL7 バージョン 2.6 インスタンスファイルです。マッピングにこのようなコンポーネントを追加するにはメニューコマンド [入力 | EDI] を選択して、必要とされるメッセージ型を [HL7] グループから選択します。これは EDI コンポーネントです。コンポーネントのタイトルバーをダブルクリックしてインスタンスファイルとこのコンポーネントに関する他の設定をビューまたは変更することができます。EDI コンポーネント設定も参照してください。

ターゲットコンポーネントは HL7 バージョン 3 インスタンスファイルです。これは上記の XML コンポーネントです。このようなコンポーネントをマッピングにメニューコマンド[入力 | XML スキーマ/ファイル]を使用して追加することができます。XML コンポーネントのために変更可能なすべての設定の詳細に関しては XML コンポーネント設定を参照してください。

6.10 プロトコルバッファー

MapForce はプロトコル バッファー書式 (https://developers.google.com/protocol-buffers/) でエンコードされるバイナリファイルからのデータのマッピング、またはバイナリファイルへのデータのマッピングをサポートするようになりました。具体的には、MapForce (または MapForce Server) は、このようなバイナリファイルからデータを読み取り、MapForce によりサポートされる(例えば、XML、CSV、データベースなどの)他の書式に変換することができるようになりました。同様に、MapForce によりサポートされる書式からデータを読み取ることができ、プロトコル バッファー バイナリファイルとして書き込むことができます。更に、Web サービスリクエストとレスポンス内のプロトコル バッファーファイルを使用することができます。Web サービス呼び出しの追加 (REST-Style). を参照してください。

以下に注意してください:

- プロトコル バッファー書式のバイナリファイルを読み取り、書き込むには、マッピングの変換言語は "Built-In" である必要があります。C#、C++、Java、XSLT、または XQuery でのコード生成はサポートされていません。詳細に関しては、変換言語の選択 を参照してください。
- データの転送を可能にするには、プロトコル バッファー書式内のバイナリファイルは、.proto ファイルが付随しています。MapForce はプロトコル バッファー .proto ファイル バージョン 2 と 3 をサポートします。

.proto ファイルは、XML スキーマが XML インスタンス ファイルの構成の構成を説明することと同様にエンコード済みのバイナリデータの構成を定義します。 例えば、次のコードリストは個人のオブジェクトを説明する .proto ファイルを表しています:

```
syntax="proto3";

message Person {
  string name = 1;
  int32 id = 2;
  string email = 3;
}
```

サンプル .proto ファイル

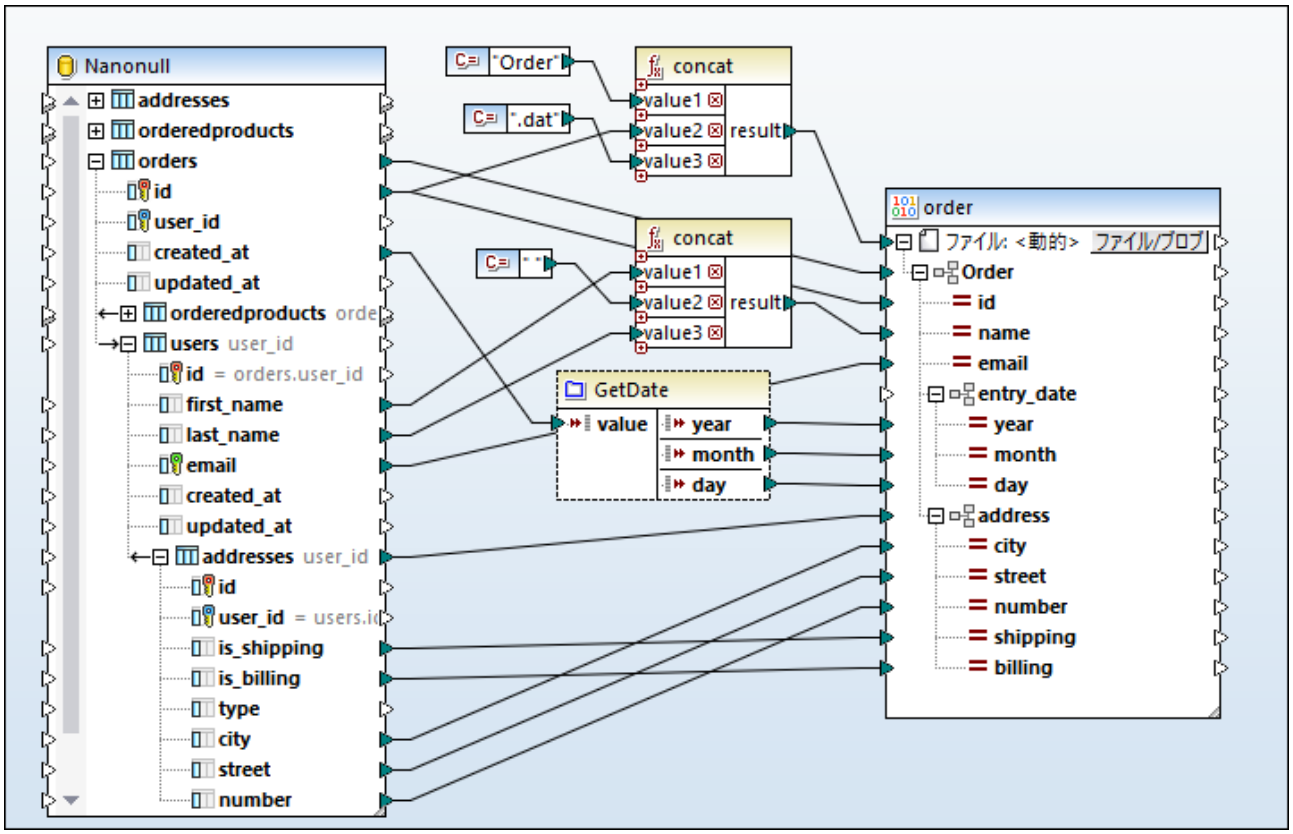
.proto ファイルは対応するバイナリファイルを解釈し、処理することができます。このため、MapForce がプロトコル バッファー書式でデータを読み取るには、以下を行う必要があります:

1. ソース バイナリファイル。このファイルには .bin または .dat などの拡張子、または他の拡張子を持つ場合があります。
2. .proto ファイル。

MapForce を使用して、プロトコル バッファー書式 でデータを書き込むためには、.proto ファイルのみが必要とされます。.proto ファイルに手動で書き込む、またはデータを取得する対応するフォーマットのデータから取得します。この場合、マッピングを実行する際、MapForce (または MapForce Server) は .proto ファイルをベースにしたバイナリファイルが生成されます。

通常の状況では、MapForce の外部のプロトコル バッファー ファイルからデータを作成または読み取る場合、プログラムコードを生成し書き込む必要があります。MapForce では、しかしながら、からコードを書き込む、または生成する必要なく、.proto ファイルをマッピングに追加し、必要とされる接続を視覚的に描くだけです。マッピングを実行すると、MapForce (または、適用できる場合、MapForce Server) はマッピングデザインに従い、ソース バイナリファイルからデータを読み取る、またはバイナリファイルを生成します。例えば、下のイメージは、データベースからデータを読み取り、プロトコル バッファー書式で複数のバイナリファイルとして書き込むマッピングを示しています。

(C) 2015-2021 Altova GmbH



GenerateOrders.mfd

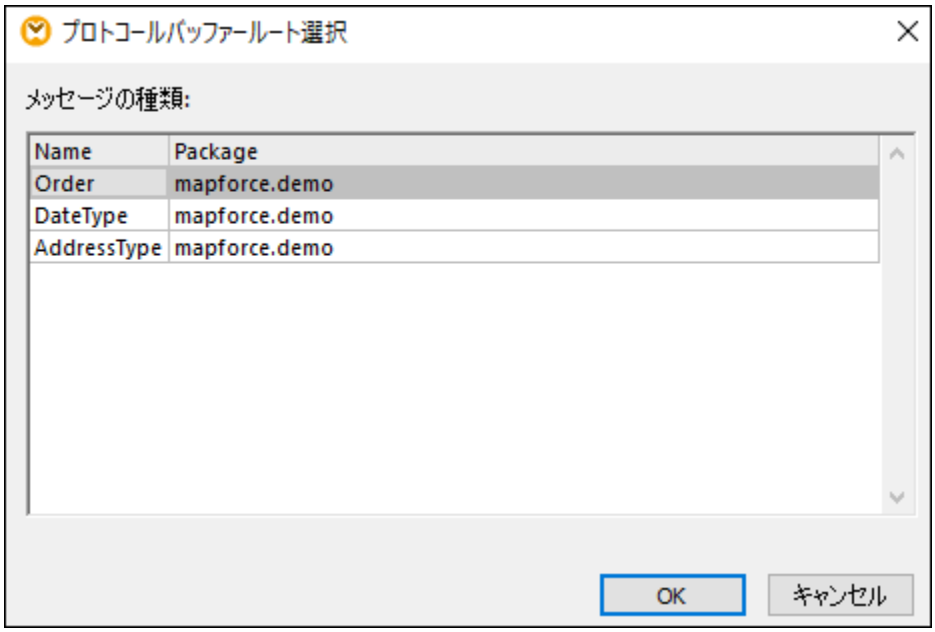
例: プロトコルバッファーにデータを書き込む で上記のマッピングは詳しく説明されています。

6.10.1　バイナリファイルをマッピングに追加する

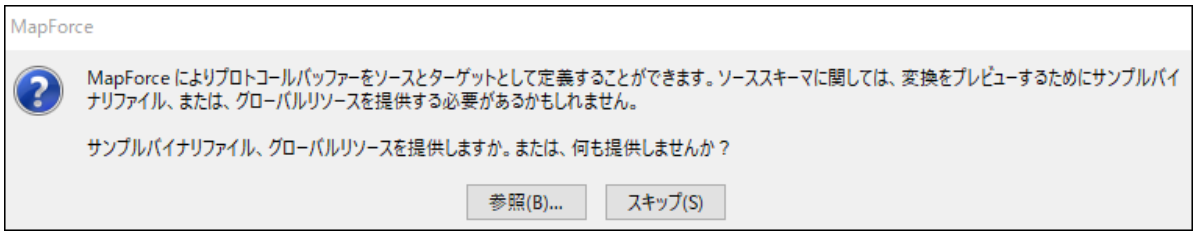
プロトコルバッファー書式でエンコードされているバイナリファイルから、またはバイナリファイルへデータを読み取るには、MapForce マッピングに .proto ファイルを以下のように追加する必要があります。

マッピングコンポーネントとして .proto ファイルを追加する方法:

1. 「入力」メニューで「プロトコルバッファー」をクリックします。または、「プロトコルバッファーファイルの挿入」ツールバーボタンをクリックします。
2. 次のいずれかを実行してください:
   a. ローカルディスク上の保管されている .proto ファイルを開くには、「開く」を開きます。
   b. URL から .proto ファイルを開くには、「URL に切り替える」をクリックして、ファイルの URL をダイアログボックスに入力します。URL からコンポーネントを追加するを参照してください。
   c. 以前にグローバルリソースとして定義されている .proto ファイルを開くには、「グローバルリソースに切り替える」をクリックします。グローバルリソースに関する情報は、Altova グローバルリソースを参照してください。
3. .proto ファイルは通常複数のメッセージ型を含んでいます。プロンプトされると、(通常「ルート」と呼ばれる)階層の一番上のオブジェクトを選択します。

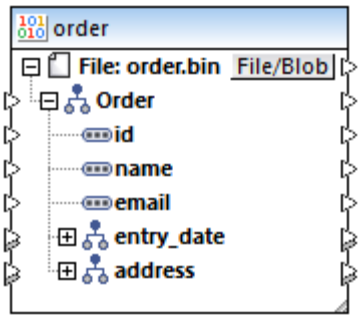


4. 「OK」をクリックします。ダイアログボックスが表示されます。



5. プロトコルバッファーバイナリファイルからデータを読み取る場合、「参照」をクリックして、データを読み取るバイナリインスタンスを選択します。プロトコルバッファーファイルにデータを書き込む場合、または後にインスタンスを選択する場合、「スキップ」をクリックします。

   .proto ファイルの読み取りに成功すると、MapForce は他のコンポーネント型に行われるようにマッピング上に構造を表示します。例:



バイナリコンポーネントがマッピングに追加されました。しかしながら、まだデータ読み取り、書き込まれていないため、マッピングはまだ完成していません。完成されたマッピングを表すサンプルに関しては、次を参照してください:

- 例: プロトコルバッファーからデータを読み取る

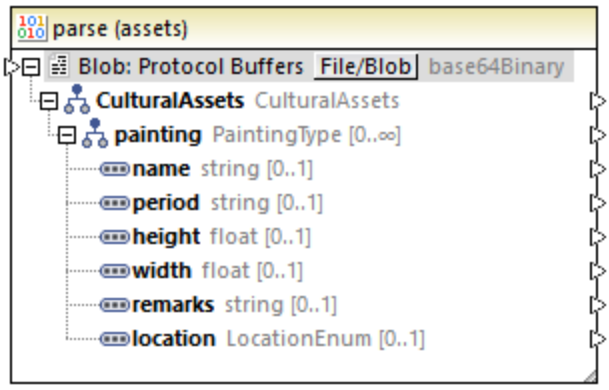
(C) 2015-2021 Altova GmbH

- 例: プロトコルバッファーにデータを書き込む

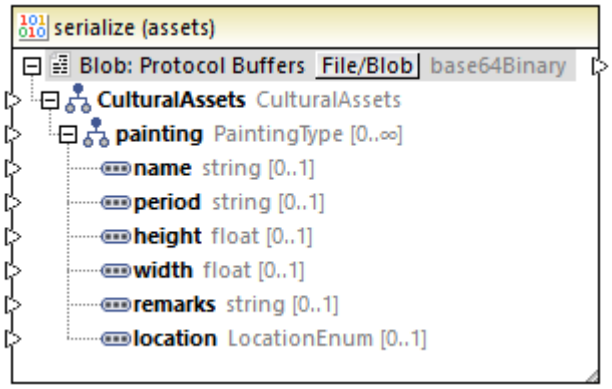
「ファイル/ブロブ」ボタン

バイナリコンポーネント上の「ファイル/ブロブ」ボタンは、次のシナリオで役に立ついくつかのコマンドが搭載されたコンテキストメニューを開きます:

- **コンポーネント設定からファイル名を使用する** – これはデフォルトで選択されるオプションで、マッピングが単一のバイナリファイルを読み取る、または生成する際に意味を成します。このオプションが設定されていると、ファイル名を「」ダイアログボックスから変更することができます。このオプションが設定されていると、「コンポーネント設定」ダイアログボックスからファイルの名前を変更することができます。バイナリコンポーネントの設定を参照してください。
- **マッピングにより提供された動的なファイル名を使用する** – このコマンドはマッピングが複数のバイナリファイルを読み取る、または生成する場合意味を成します。このオプションが設定されていると、「コンポーネント設定」ダイアログボックスからファイル名を変更することはできません。代わりに、マッピング自身からの情報をベースにファイル名を設定することができます。例: プロトコルバッファーにデータを書き込むを参照してください。
- **プロトコルバッファーにブロブを解析する** – このコマンドは、コンポーネントを解析コンポーネントに変換します。これにより、マッピングが可能なツリー構造にすることにより、プロトコルバッファー書式のバイナリの値を解析することができます。これは文字列の解析とシリアル化に類似した方法で作動します。文字列の解析とシリアル化を参照してください。この場合、文字列ではなくバイナリバイト文字列とのみ作業することができます。 例えば、下のコンポーネントでは、バイナリ型のアイテムを左側の「ブロブ」入力に接続することができます。マッピングのランタイムでは、個のバイナリの値は解析され、データを右側のマップ可能な構造にパスします。



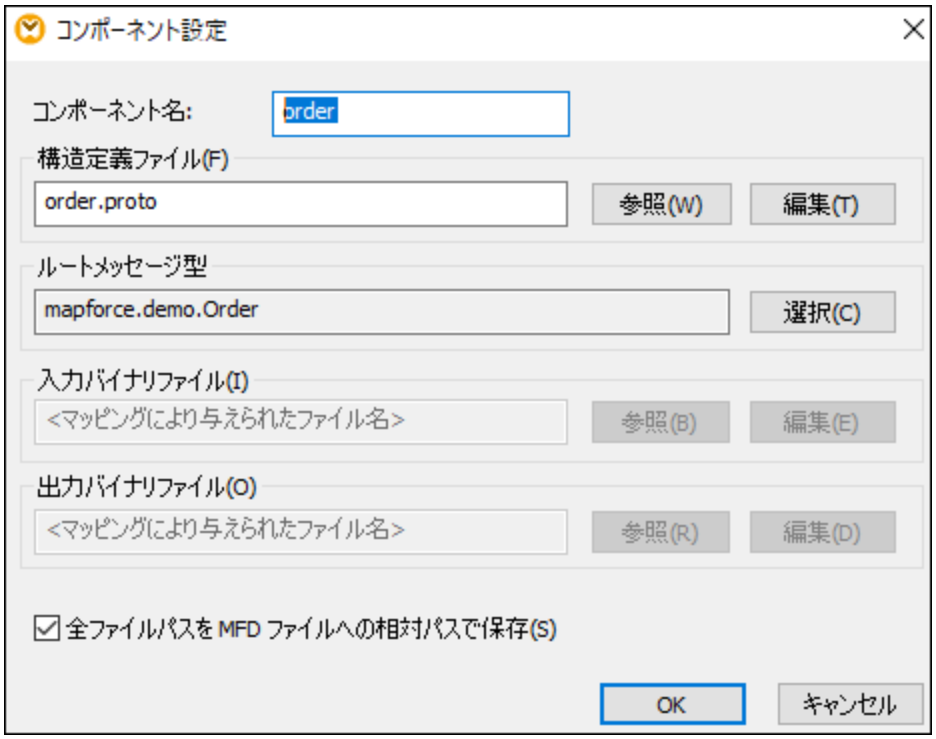
- **プロトコルバッファーをブロブにシリアル化する** – このコマンドは、コンポーネントをシリアル化コンポーネントに変換することができます。これにより、マップ可能なツリー構造をプロトコルバッファー書式内のバイナリ値にシリアル化を手助けすることができます。例えば、下のコンポーネントでは、適切な型のアイテムを左側のマップ可能構造に接続することができます。マッピングランタイムでは、構造はシリアル化され、(バイナリ型の)単一の出力を右側に生成し、更にターゲットコンポーネントに接続できるようになります。これはマッピングがBLOB (Binary Large Object) 型としてデータをターゲットデータベースに書き込む際に役に立ちます。他の使用方法は、データをターゲットプロトコルバッファーコンポーネントの(「バイト」型の) バイナリフィールドに書き込む場合です。



6.10.2 プロトコルバッファーコンポーネントの設定

バイナリコンポーネントをマッピングエリアに追加した後、コンポーネント設定 ダイアログボックスから適用することのできる設定を構成することができます。コンポーネント設定 ダイアログボックスを次の方法で開くことができます:

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



コンポーネント設定 ダイアログボックス

使用することのできる設定は以下の通りです。

(C) 2015-2021 Altova GmbH

コンポーネント名	コンポーネント名はコンポーネントを作成する際に自動的に生成されます。ですが、後に名前を変更することが可能です。 コンポーネント名はスペースを含むことができ、またフルストップ文字も含むことができます。コンポーネント名は、スラッシュ、バックスラッシュ、コロン、二重引用符、行頭と末尾スペースを含む場合があります。全般的に、コンポーネントの名前を変更する際、以下の影響を考慮してください: • マッピングを FlowForce Server にデプロイする場合、コンポーネント名は一意である必要があります。 • コマンドラインに入力できる文字のみを使用することが奨励されます。コマンドラインと Windows 内では、それぞれの文字で異なったエンコードを使用する場合があります。
プロトコルバッファー ファイル	実際のバイナリデータをマップするために、MapForce により使用される構成 定義 (.proto) ファイルの名前、またはパスを指定します。 構成定義ファイルを変更するには、「参照」をクリックして、新規のファイルを選択します。「編集」をクリックします。
ルートメッセージの型	.proto ファイルは通常複数のメッセージ型を含みます。「選択」をクリックし、階層構造内のトップ(「ルート」)に表示されているメッセージ型を選択します。
入力バイナリファイル	MapForce がデータを読み取るバイナリ インスタンスファイルを指定します。このフィールドはソースコンポーネントにとり重要です。 ファイルの場所を変えるには、「参照」をクリックして、新規のファイルを選択します。「編集」をクリックします。
出力バイナリファイル	MapForce がデータを書き込むバイナリ インスタンスファイルを指定します。このフィールドはターゲットコンポーネントにとり重要です。 ファイルの場所を変えるには、「参照」をクリックして、新規のファイルを選択します。「編集」をクリックします。
MFD ファイルに相対的なすべてのファイル パスを保存する	オプションが有効化されると、MapForce は、MapForce Design (.mfd) ファイルのロケーションに対して相対的なコンポーネント設定 ダイアログボックス上に表示されるファイル パスを保存します。コンポーネント上の相対的なパスの使用 を参照してください。

6.10.3　例: プロトコルバッファーからデータを読み取る

このサンプルは、プロトコル バッファー書式でエンコードされているバイナリファイルからデータを読み取る方法について示しています。ソース バイナリファイルを次のパスで見つけることができます: **<マイドキュメント>\Altova\MapForce2021\MapForceExamples\Tutorial\assets.bin**。バイナリファイルを説明する .proto ファイルは同じディレクトリで使用することができ、以下のようになります:

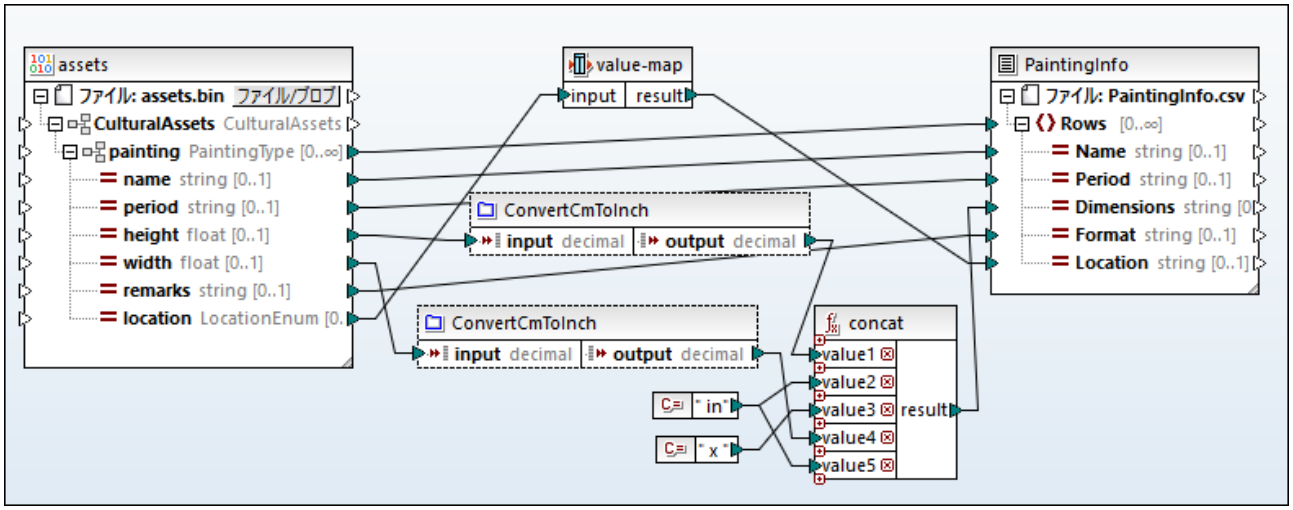
```
syntax="proto3";

package mapforce.demo;
```

```
message CulturalAssets {
  repeated PaintingType painting = 1;
}

message PaintingType {
       string name = 1;
       string period = 2;
       float height = 3; // in cm
       float width = 4; // in cm
       string remarks = 5;
       LocationEnum location = 6;

       enum LocationEnum {
              MUSEUM = 0;
              MUSEUM = 1;
              MUSEUM = 2;
              MUSEUM = 3;
       }
}
```

assets.proto

ソースバイナリファイルには、多種の文化財に関する情報が含まれています（このサンプルでは、絵画のコレクションが使用されています）。上記の `.proto` ファイルにより表示されているとおり、各絵画の「高さ」と「幅」はセンチメートルで表示されています。絵画についての説明が、（美術館、私蔵、寺院などの）場所を指定する一覧表などを含む、他のフィールドで説明されています。コンマで区切られた（CSV）ファイルでこの情報を抽出することが、業務の必要条件です。また、絵画の寸法はセンチメートルからインチに変換され、「24 インチ x 56.8 インチ」などのフォーマットのように単一の文字列で表示される必要があります。

上記の必要条件にしたがデータを抽出し、希望する出力を生成するするマッピングは下に表示される通りです。このマッピングは次のディレクトリで使用することができます: **<マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥Tutorial¥**。マッピングを開き直接実行、または下のステップに従い始めから作成することができます。



ReadProtocolBuffers.mfd

上記のとおり、マッピングは、ソースバイナリコンポーネント（**assets**）とターゲット CSV コンポーネント（**PaintingInfo**）により構成されています。また、次の中間コンポーネントが含まれています:

- 0、1、2、3 などの列挙の値を人間が判読できる文字列表記（例えば、`0 = UNKNOWN`、`1 = MUSEUM` など）に変換する value-map テーブル。

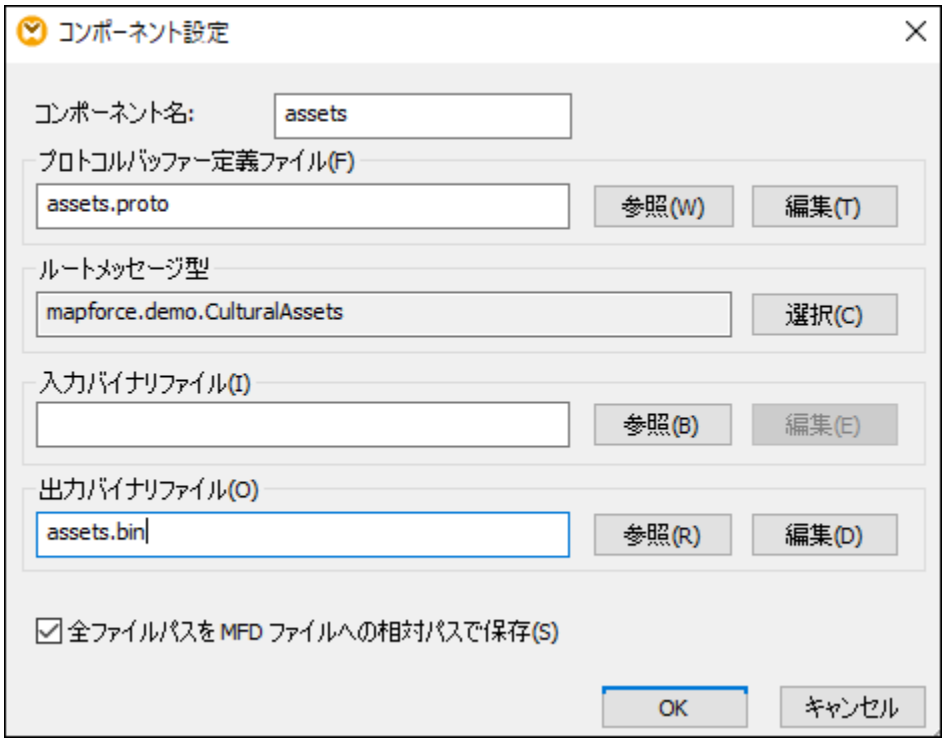
(C) 2015-2021 Altova GmbH

- `ConvertCmToInch` 関数。タイトルが示す通り、これは、センチメートルをインチに変換するユーザー定義関数です。この関数はマッピング上で（それぞれ、「高さ」と`width`「幅」のために）2回呼び出されます。
- `concat` 関数。この関数は複数の値を連結し「# in x # in」のフォーマットで文字列を返します。# は絵画の実際の高さと幅をインチで示しています。

次のステップはマッピングを最初から作る方法を表示しています。

ステップ 1: ソースバイナリファイルの追加

1. 「**挿入**」メニューで「**プロトコルバッファーファイル**」をクリックし、次のファイルを参照します: **<マイドキュメント>\Altova\MapForce2021\MapForceExamples\Tutorial\assets.proto** 。
2. プロンプトされると、「CulturalAssets」をルートメッセージとして選択し、「OK」をクリックします。
3. 新しいコンポーネントのタイトルバーをダブルクリックします。「バイナリコンポーネント設定」ダイアログボックスが開かれます。
4. 「**入力バイナリファイル**」の横の「**参照**」をクリックし、次のファイル: **<マイドキュメント>\Altova\MapForce2021\MapForceExamples\Tutorial\assets.bin**。



バイナリファイルをマッピングに追加するも参照してください。

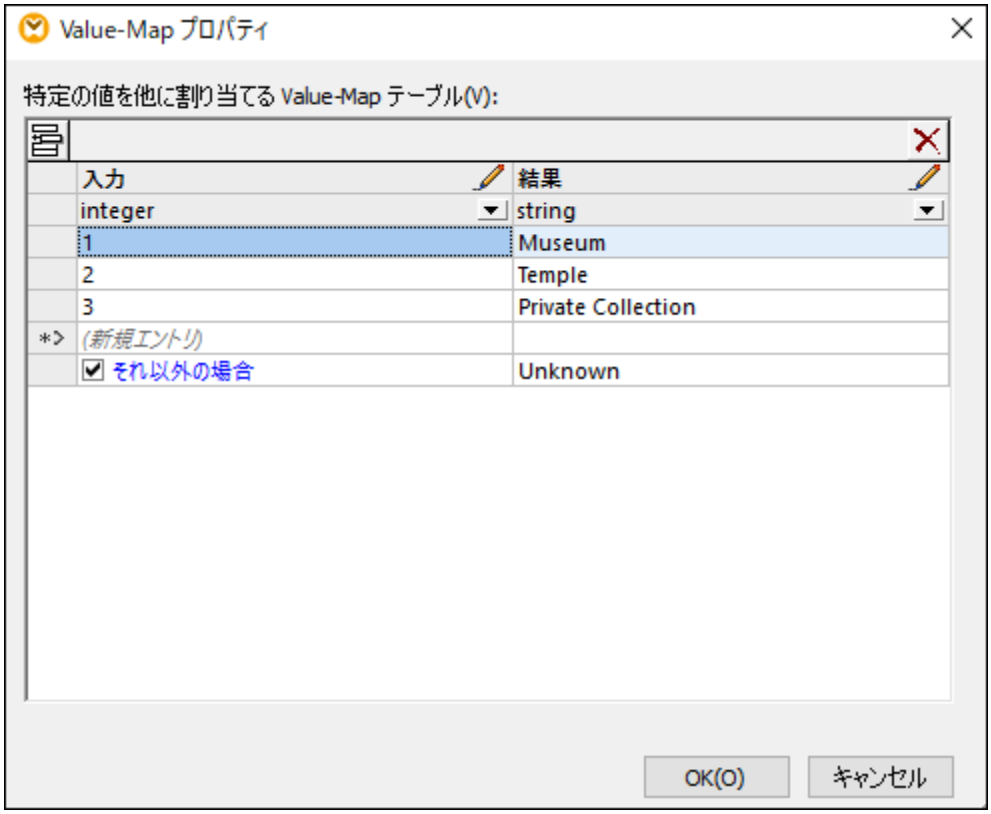
ステップ 2: ターゲット CSV コンポーネントの追加

1. 「**挿入**」メニューから「**テキストファイル**」をクリックします。
2. 「**標準 CSV に単純な処理を使用**」チェックボックスを選択し、「**継続**」をクリックします。「CSV コンポーネント設定」ダイアログボックスが開かれます。
3. 「**フィールドの追加**」ボタンを複数回クリックし、希望するデータを保管するために必要とされる5つのターゲットフィールドを作成します。
4. 各フィールドのヘッダーをダブルクリックし、詳細を説明する名前を与えます。

詳細に関しては、CSV オプションの設定 を参照してください。

ステップ 3: value-map の追加

1. 「挿入」メニューで「Value-Map」をクリックします。
2. 下に示されるように各マップされたレコードのためにエントリを作成します。

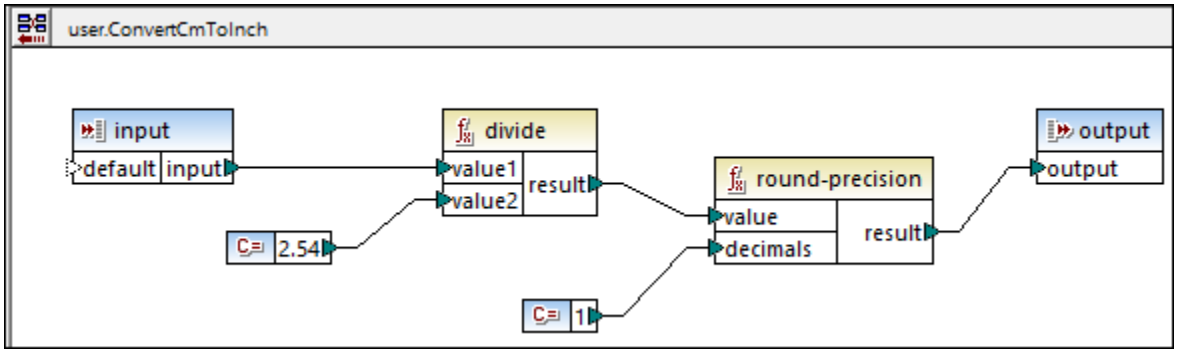


「Otherwise」条件は、バイナリのファイルが `Location` フィールドのために値が含まれない場合を処理します。このイベントが発生すると、マッピングは、結果を生成しない代わりに、値「Unknown」をターゲットに書き込みます。詳細に関しては、Value-Map の使用を参照してください。

ステップ 4: ユーザー定義関数の作成

1. 関数 メニューで「**ユーザー定義関数の作成**」をクリックし、"ConvertCmToInch" と名前を付けます。他のオプションをそのままにします。
2. **`divide`** と **`round-precision`** ビルトイン関数をライブラリウィンドウからドラッグし、関数のマッピングウィンドウにドロップします。マッピングに関数を追加するも参照してください。
3. 下に示されるとおり2つの定数を追加し、マッピングに定数を追加するを参照してください。最初の定数は、書式に従いセンチメートルをインチに変換するための小数定数を提供します。2番目の定数は、四捨五入の精度を提供します。

基本的には、以下に示される通り関数は、入力値を 2.54 で割り、結果を小数点の一桁まで四捨五入します:

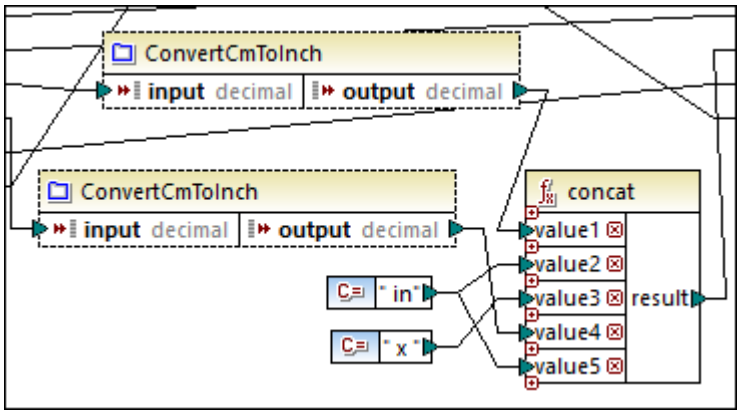


エラーを回避するために、入力と出力コンポーネントのデータ型を小数に設定します。データ型の変更を確認するには、コンポーネントのタイトルバーをダブルクリック、またはタイトルバーを右クリックし、コンテキストメニューから「プロパティ」を選択します。

ユーザー定義関数が準備されると、「メインマッピングに戻る」をクリックし、呼び出すためにライブラリウィンドウから関数をマッピングにドラッグします。このサンプルでは、関数は2度呼び出されます。詳細に関しては、ユーザー定義関数 を参照してください。

ステップ 5: 結合関数の追加

ライブラリウィンドウに「concat」を入力し、マッピングに`concat` 関数 をドラッグします。必要とされる入力引数をすべて追加するには、「パラメーターの追加」(⊙) を複数回クリックします。関数引数の追加または削除を参照してください。定数 「in」と「x」は文字列の値です。基本的には、関数は、絵画の「高さ」を「 x 」が後に続く文字列「 in 」と共に返し、「高さ」を「 in 」と共に返します。



マッピングの実行

MapForce 内で生成されたファイルをプレビューするには、「出力」タブをクリックします。三番目のCSV 列には必要とされたフォーマットのディメンションが含まれていることに注意してください。

```
Red and White Plum Blossoms,18th century,61.4 in x 67.8 in,Ink and color on paper,Museum
Red and white hibiscuses,1197,10 in x 9.9 in,Color on silk,Museum
Pigeon on a peach branch,1108,11.2 in x 10.3 in,Color on silk,Private collection
Pine trees in snow,1773,142.5 in x 61.2 in,Light color on paper,Museum
Fishing village in the evening glow,13th century,13 in x 44.3 in,Ink on paper,Museum
Five Guardian Kings,Kamakura period,76.3 in x 49.7 in,Color on silk,Temple
Snowclad houses in the night,Edo period,11 in x 51.2 in,Ink and light color on paper,Private
```

マッピングの出力

マッピングの出力をディスクに保存する方法:

- 「出力」メニューで「出力ファイルを保存」をクリックします。

MapForce Server を使用した自動化

ライセンスされたMapForce Server を使用する場合、Linux、macOS、またはW indows マシン上のコマンドラインでマッピングを実行することができます:

1. メニューコマンド「ファイル | MapForce Server 実行ファイルにコンパイルする」、を使用してマッピングをMapForce Server 実行ファイル(.mfx)にコンパイルします。MapForce Server 実行可能ファイルにマッピングをコンパイルするを参照してください。
2. .mfx ファイルをサーバーマシンにコピーします。
3. MapForce Server を下のコマンドを使用して実行します。

```
mapforceserver run ReadProtocolBuffers.mfx
```

メモ

- **mapforceserver** は、使用中のオペレーティングシステムに適用することのできるMapForce Server 実行可能へのパスです。
- .mfx ファイルへのパスを適用できるように変更、または実行可能と同じフォルダー内に.mfx をコピーします。
- .mfx ファイルは内蔵されており、.proto ファイルはサーバー実行のために必要ありません。

サーバー実行の際は、API 呼出、またはFlowForce Server ジョブとしてマッピングを実行することができます。詳細に関しては、MapForce Serve を使用した自動化 を参照してください。

6.10.4 例: プロトコルバッファーにデータを書き込む

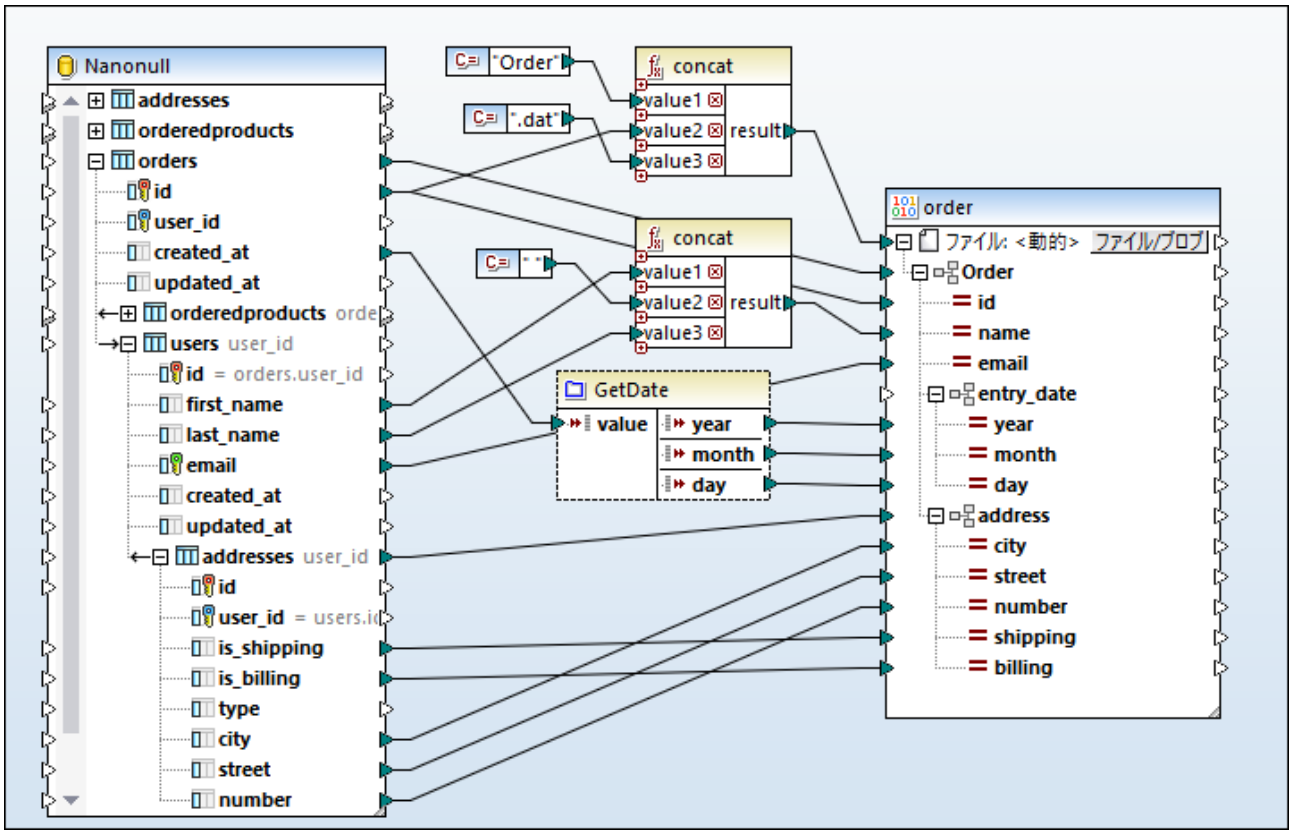
このサンプルはプロトコルバッファー書式でエンコードされているバイナリファイルにデータを書き込む方法を示しています。このデモ内では、SQLite データベース(**Nanonull.sqlite**) からデータソースが来ていますが、同様のアプローチを使用して、MapForce がサポートするXML、EDI、JSON などの多種の書式からデータを取得することもできます。

ソースデモデータベースは、個人、住所、ストアから注文した製品のデータを保管します。業務の必要条件は、データベースから注文情報を抽出し、プロトコルバッファー書式で複数のバイナリファイルを生成します。.dat 拡張子を持つ1つのバイナリファイルは、各注文のために生成される必要があります。生成されたファイル名は**Order1.dat**、**Order2.dat** などの注文の一意のID を含む必要があります。例えば、各バイナリファイルは、注文ID 、個人の名前、電子メールアドレス、日付、および、配達先の住所に関する情報を含む必要があります。具体的には、注文の構成を説明する .proto ファイルは以下のようになります:

```
syntax="proto3";

package mapforce.demo;

message Order {
  int32 id = 1;
  string name = 2;
  string email = 3;
  DateType entry_date = 4;
  // An order can have multiple addresses (shipping, billing)
  repeated AddressType address = 5;
}

message DateType {
  // Must be from 1 to 9999
  int32 year = 1;
  // Must be from 1 to 12
  int32 month = 2;
  // Must be from 1 to 31, and valid for the year and month
  int32 day = 3;
}

message AddressType {
  string city = 1;
  string street = 2;
  int32 number = 3;
  bool shipping = 5;
  bool billing = 4;
}
```

orders.proto

Nanonull.sqlite データベースと`orders.proto`ファイルの両方は、次のディレクトリで見つけることができます: **<マイドキュメント>\Altova\MapForce2021\MapForceExamples\Tutorial**。変換を行うマッピングデザインファイルは **GenerateOrders.mfd** と呼ばれ、同じディレクトリで見つけることができます。マッピングを開き直接実行、または下のステップに従い始めから作成することができます。



GenerateOrders.mfd

上記のとおり、マッピングソースコンポーネント（SQLite データベース）とターゲットコンポーネント（バイナリコンポーネント）により構成されています。ソース データベース内では、このマッピングに関連するメインのテーブルは**orders** です。

MapForce は2 つのテーブル間で外部キーリレーションシップを自動的に検知するため、テーブル**users** はテーブル**orders** の下にネストされていることに注意してください。同様に、テーブル**addresses** は**users** の下にネストされます。このようなネストされているテーブルから描かれるすべての接続は、キーリレーションシップがマッピングにより保管されることを保証します。詳細に関しては、データベースリレーションシップの処理 を参照してください。

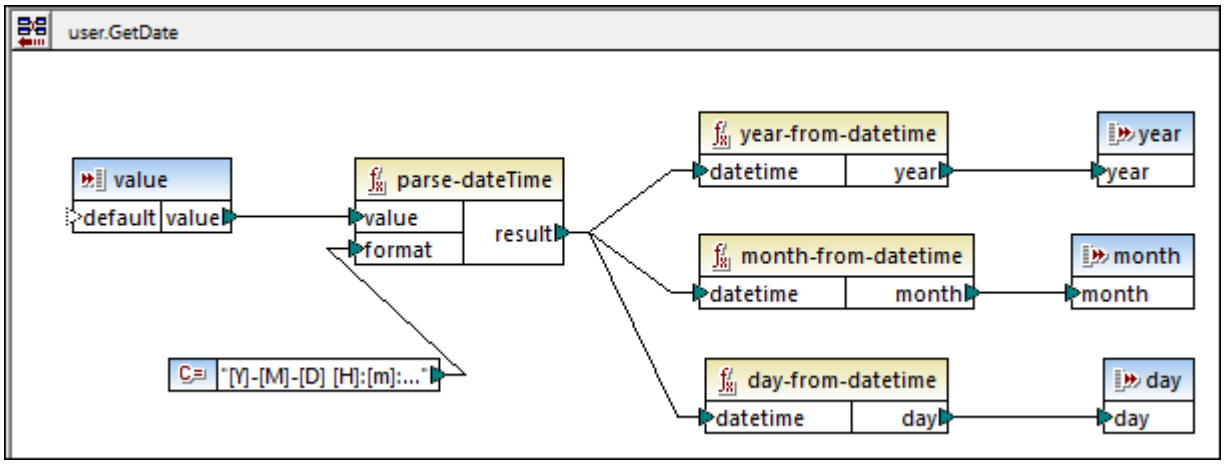
ターゲット側では、**ファイル:＜動的＞** アイテムは、マッピングからの情報をベースにしてバイナリコンポーネントがインスタンスファイルを動的に生成することを示しています。ターゲットコンポーネントにファイルを動的に生成させるには、 ファイル/プロプ ボタンを押して、**「マッピングにより提供される動的なファイルの使用」**をコンテキストメニューから選択します。動的なファイル名の詳細に関しては、複数の入力または出力ファイルを動的に処理を参照してください。

このマッピングでは、**orders** データベーステーブル内の各 **id** では、**ファイル:＜動的＞** アイテムへの接続で示されるように新規のファイルが生成されます。各生成されたファイルの名前は、最初の `concat` 関数の助けと共に作成されます。この関数は、データベースからの注文票の一意のID を持つ文字列 "Order" と、拡張子 ".dat" をジョインします。このため、生成されたファイルは**Order1.dat**、**Order2.dat** などの名前を取得します。

`concat` 関数はマッピングから2 度目の呼び出しを受けます。今回は、氏名を連結し、空白文字を間に入力して、個人の名前を返します。

最後に、中央の"GetDate" コンポーネントは、バイナリファイル内の日付フィールドを作成するユーザー定義関数です。この関数は、入力として文字列で表示される日付を取り、年度、月、日を個別に整数の値として返します。この変換は、 `.proto` ファイル内で定義されている通り日付フォーマットを適応させるために必要とされます。マッピング上の関数のタイトルバーをダブルクリックすると、この関数の定義を確認または変更することができます。

(C) 2015-2021 Altova GmbH



"GetDate" ユーザー定義関数

上記のとおり、"GetDate" ユーザー定義関数は`parse-dateTime` ビルトイン関数を使用して入力値を解析します。マスク `[Y]-[M]-[D] [H]:[m]:[s]` は、データベースに保存されている書式に日付書式を一致させます。例えば、`2017-10-19 08:09:54`。書式マスクの詳細に関しては、`parse-dateTime` 関数の説明を参照してください。次に、多種のビルドイン関数が呼び出され、解析された日付から重要な日付の部分が抽出され返されます。

マッピングの作成方法

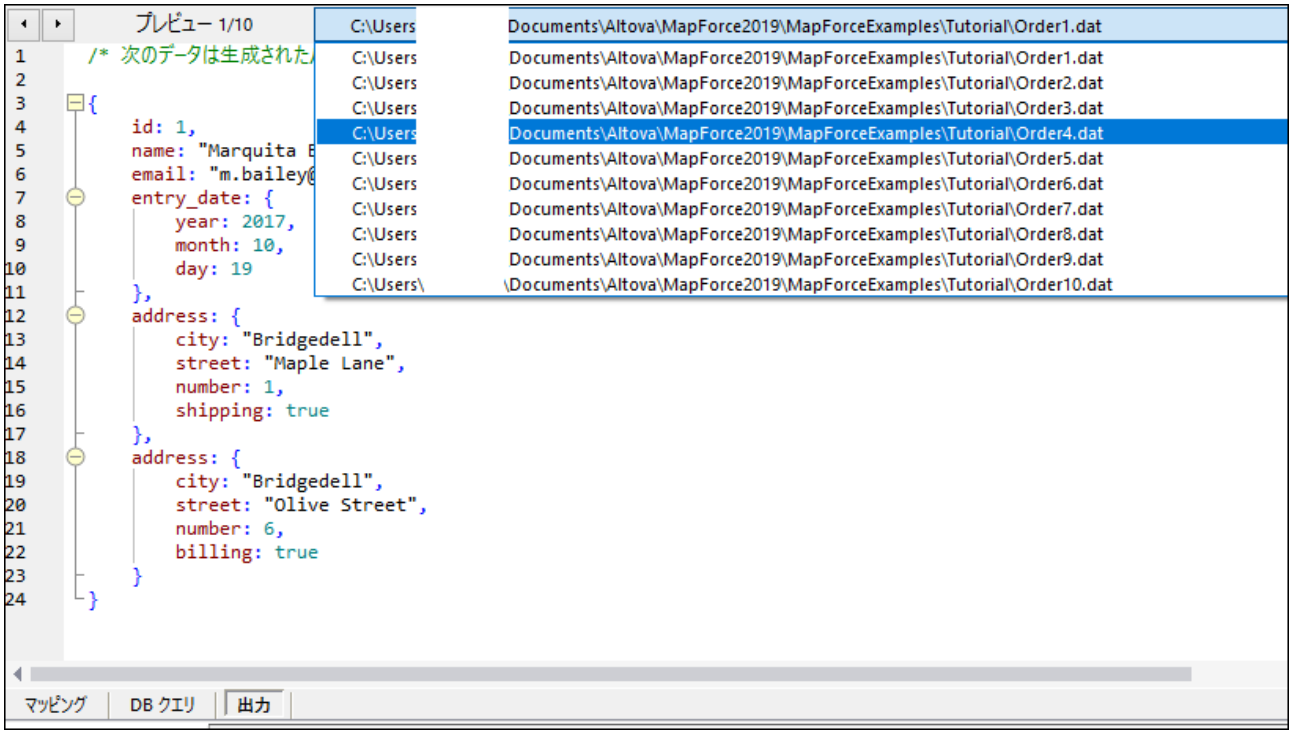
上記のマッピング（または類似するマッピング）を作成するには次のステップに従ってください：

1. マッピングにデータベースを追加するで説明されているとおり、ソースデータベースを挿入します。
2. ターゲット `.proto` ファイルを入力します。バイナリファイルをマッピングに追加するを参照してください。
3. ファイル名を動的に生成するために、ターゲットコンポーネントを構成します。これを行うには、ファイル/プロプ ボタンをクリックして、**「マッピングにより提供される動的なファイル名の使用」**をコンテキストメニューから選択します。ファイル名を提供する入力に**ファイル<動的>** アイテムを接続します。このサンプルでは、ファイル名は上記の通り、`concat` 関数を使用して生成されます。動的なファイル名に関する詳細は、複数の入力または出力ファイルを動的に処理を参照してください。
4. 任意でMapForce ビルトイン関数を追加し、入力を接続します。このサンプルでは、`concat` 関数が2回使用されています。詳細に関しては、マッピングに関数を追加する を参照してください。`concat` などのビルトイン関数を呼び出す場合、文字列定数から入力データの一部が来る可能性があります。このような定数の情報に関しては、マッピングに定数を追加するを参照してください。
5. 必要な場合、ユーザー定義関数を追加します。これらにより、マッピングをあいまいにし、大きな概要に集中することができます。このサンプルでは、「GetDate」ユーザー定義関数は、文字列を、上記のとおり、日付の部分に変換し、メインマッピングから実装ロジックは隠されます。カスタム関数の詳細に関しては、ユーザー定義関数を参照してください。

MapForce の使用の初心者は、シンプルなステップバイステップのサンプルに関してはチュートリアルを参照してください。

マッピングの実行

このサンプルの最初に説明されているビジネスの必要条件の通り、マッピングは複数の.dat ファイルを生成することが期待されています。生成されたファイルをプレビューするには、**「出力」**タブをクリックします。



上記のとおり、「出力」ペインはすべての生成されたファイルを表示し、方向ボタンをクリックして、またはリストからアイテムをピックしてナビゲートすることができます。生成された出力を保存するには、次のいずれかを実行してください:

- 「出力」メニューで「全ての生成ファイルを保存する」( )をクリックします。
- 「生成されたすべての出力を保存する」( ) ツールバーボタンをクリックします。

MapForce により生成された出力バイナリファイルをプレビューする場合、個のフォーマットはバイナリ書式とは異なり人間が判読できるため、構造はJSON として表示されます。JSON としてマッピングを確認することは、出力をテストする手助けをします。出力 ファイルを保存する場合、バイナリファイルとして保存されます。

MapForce Serve を使用した自動化

ライセンスされたMapForce Server を使用する場合、Linux、macOS、またはW indows マシン上のコマンドラインでマッピングを実行することができます:

1. メニューコマンド「ファイル| MapForce Server 実行ファイルにコンパイルする」、を使用してマッピングをMapForce Server 実行ファイル(.mfx)にコンパイルします。MapForce Server 実行可能ファイルにマッピングをコンパイルするを参照してください。
2. .mfx ファイルをサーバーマシンにコピーします。
3. このマッピングはSQLite データベースファイルからデータを読み取るため、サーバー上の.mfx ファイルと同じディレクトリに **Nanonull.sqlite** データベースファイルをコピーします。他のデータベースの種類に関しては、多種の実行環境でのデータベースマッピングを参照してください。
4. MapForce Server を下のコマンドを使用して実行します。

```
mapforceserver run GenerateOrders.mfx
```


(C) 2015-2021 Altova GmbH

メモ

- **mapforceserver** は使用中のオペレーティングシステムのために適用できる MapForce Server 実行可能ファイルへのパスです。
- 適用できる通り、.mfx ファイルへのパスを変更、または実行可能ファイルと同じフォルダーに .mfx をコピーします。

サーバー実行の際は、API 呼出、または FlowForce Server ジョブとしてマッピングをオンデマンド、または再帰的なベースで実行することができます。 詳細に関しては、MapForce Serve を使用した自動化 を参照してください。

6.11 バイナリファイル

MapForce はBLOB（バイナリラージオブジェクト）データをバイナリファイルからマッピングに読み取り、バイナリデータ（生）の内部構造を変更することなく消費する機能を搭載しています。例えば、バイナリデータをデータベースBLOB フィールド、XML ファイル内の `xs:base64Binary` のフィールドに保存、または、W eb サービスに送信することができます*。

＊W eb サービス呼び出しは MapForce Enterprise Edition のみでサポートされています。

（データベース内のBLOB フィールド、XML ファイル内の型 `xs:base64Binary` のフィールド、W eb サービスなど）一部のソースからバイナリデータを読み取るマッピングを作成し、ディスクにバイナリファイルを書き込むこともできます。

以下はバイナリファイルの読み取り、またはバイナリファイルへの書き込み機能の利点を示したサンプルシナリオです:

- ベース-64 データとしてエンコードされたバイナリを XML ファイルから抽出しディスクに（例えばPDF ファイルとして）保存します。
- ディスク上に保管されたイメージファイルを処理して ベース-64 エンコードされたバイナリコンテンツとして W eb サービスに送信します。
- BLOB コンテンツをデータベーステーブルから抽出し、ディスクにイメージファイル（データベーステーブル内の各行に対して1 つのイメージファイル）として保存します。
- ディスクからイメージファイルを読み取り、BLOB データフィールドとしてデータベーステーブルに保存します。

メモ　バイナリファイルからデータをマッピングする、または、バイナリファイルにデータをマッピングするには、変換言語として BUILT-IN が必要になります。MapForce 内でマッピングをプレビュー（し存在する場合は出力ファイルを保存）、または（個別にライセンス供与されている）MapForce Server を使用して異なるコンピューターまたはプラットフォームで実行することを選択できます。実行可能な C#, C++, または Java プログラムをバイナリファイルを読み取るまたはバイナルファイルに書き込むマッピングから生成することはサポートされていません。

バイナリファイルをマッピングに追加する方法

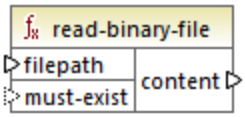
XML テキスト、またはJSON ファイル同様にMapForce 内でバイナリファイルに関連するコンポーネントは存在しません。代わりに、上記のような目的を達成するためにMapForce ビルドイン関数を使用することができます:

- read-binary-file
- write-binary-file

これらの関数をライブラリウィンドウで見つけることができます（具体的には[lang] ライブラリで見つけることができます）。マッピング内でどちらかの関数を使用するには、ライブラリウィンドウからマッピングにドラッグ、または、マッピングの空のエリアをダブルクリックし、関数名の入力を開始して選択します。詳細に関しては、マッピングに関数を追加するを参照してください。

read-binary-file

この関数は型 `xs:base64Binary` のBLOB（バイナリラージオブジェクト）として指定されたファイルのコンテンツを返します。データ型が[base64Binary] と呼ばれる場合でも内部の表記はBLOB です。関数の結果を型 `xs:base64Binary` のXML ノードにマップする場合のみ、ベース64 エンコードになります。関数の結果を `xs:hexBinary`、データベースブロブ、またはプロトコールバッファー構造内のバイナリフィールドにマップすることもできます。



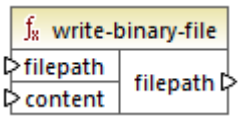
バイナリファイルをマッピングに読み込むには**filepath** 引数に対しての入力としてパスを提供します。**filepath** が相対的な場合 MapForce はマッピングと同じディレクトリ内でファイルを探します。**must-exist** 引数が任意です。ファイルを開くことはできず、このパラメー

(C) 2015-2021 Altova GmbH

ターは**true**，になり、マッピングはエラーをスローします。ファイルを開くことができなく、このパラメーターが**false** の場合、からのバイナリが返されます。

write-binary-file

この関数はバイナリコンテンツを指定されたファイルパスに書き込み、書き込まれたファイルのパスを返します。バイナリファイルが唯一の希望する出力の場合 単純型出力 コンポーネントに関数の結果を接続します。出力がマッピング内で使用される都度この関数はファイルに書き込むため、間に他の処理を使用することなく、関数の結果を直接ターゲットコンポーネントに接続することが奨励されます。



バイナリファイルを書き込むには**filepath** 引数に対しての入力としてパスを提供します。**filepath** が相対的な場合 MapForce はマッピングと同じディレクトリ内でファイルを生成します。The **content** 引数は実際のバイナリコンテンツ（例えば、データベース内のBLOB フィールド）に接続されている必要があります。

MapForce 内でマッピングをプレビューすると、ディスクにファイルを直接書き込む代わりに関数は一時的なファイルをデフォルトで生成します。一時的なファイルをディスクに保存するには、最初に**出力** タブをクリックし、適用可能な場合 **生成された出力を保存** または **全ての生成済みの出力** ツールバーボタンをクリックします。

MapForce が出力を一時的なファイルではなくファイナルファイルに直接書き込むように構成するには[**ツール | オプション** メニューコマンド]を選択し [**全般**]をクリックして [**最終出力ファイルに直接書き込む**] オプションを選択します。このオプションは同じ名前を持つ既存のファイルを上書きすることにご注意してください。

この関数はファイルがディスクに保存されていない場合（マッピングをプレビューし [**最終出力ファイルに直接書き込む**] オプションが無効化されている場合）でも常に最終（一時的ではない）ファイル名を返します。

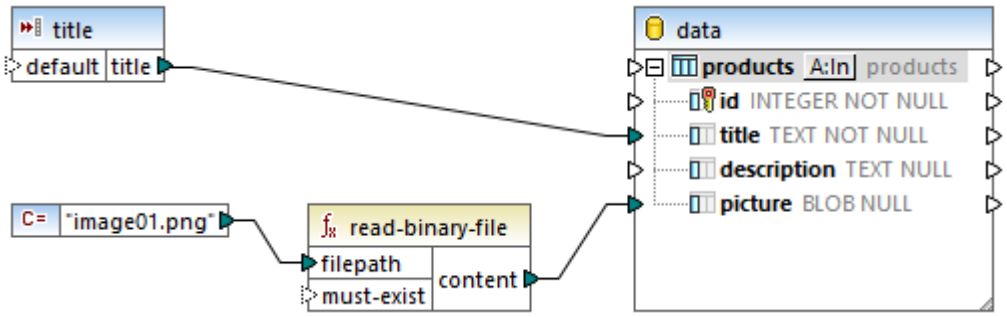
> マッピングが自身の出力ファイルを読み取ることはサポートされていないことにご注意してください。

例

サンプル: バイナリの読み取り と 例: バイナリファイルの書き込み を参照してください。

6.11.1　例: バイナリファイルの読み取り

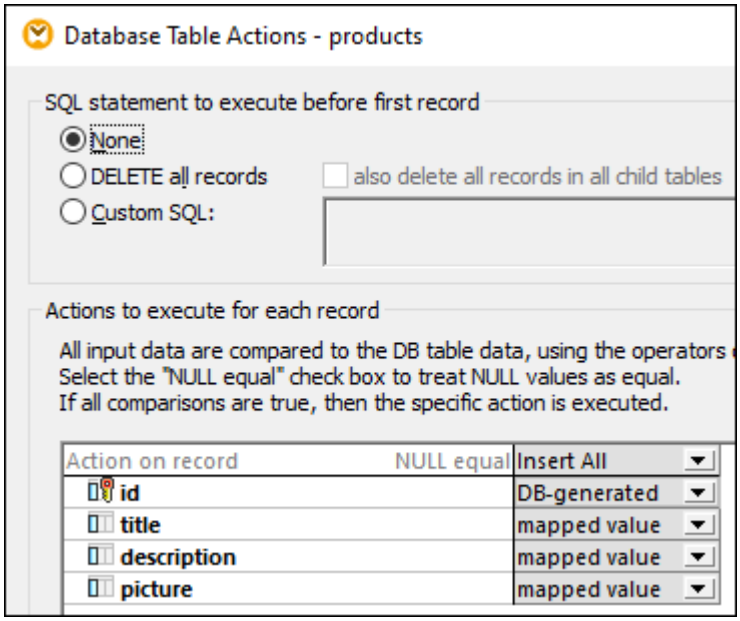
下で説明されるマッピングはイメージファイルからデータを読み取りデータベーステーブルに書き込みます。ターゲットデータベースはSQLite です。**picture** データベースフィールドのデータ型はBLOB であることにご注意してください。.



ファイルからバイナリコンテンツを抽出するために `read-binary-file` 関数が使用されました。このサンプルでは最初の引数 **filepath** が関数により提供されています。パスが相対的なためMapForce はマッピングとして同じディレクトリ内のイメージファイルを検索します。

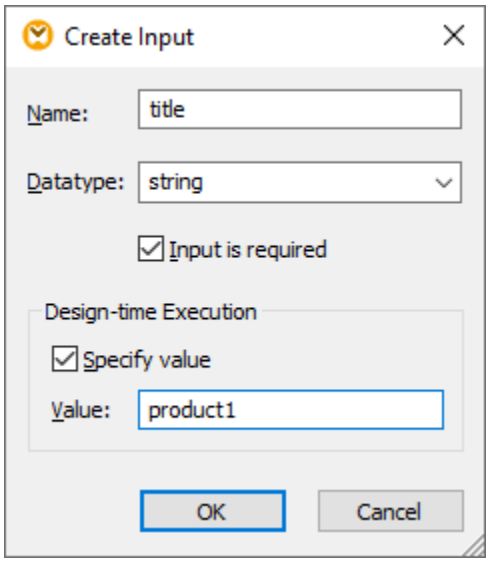
マッピングはターゲットデータベースに次のフィールドを作成します:

- **id** – このサンプルでは **id** がデータベースコンポーネントがマッピングにより提供されるよりもデータベースにより生成されるように構成されています。データベースにより生成されるように構成されています。詳細に関しては[データをテーブルに挿入する]を参照してください。



- **title** – この値は単純型入力コンポーネントにより提供されています。デザイン時の実行値は("product1") に設定されておりマッピング内でプレビューできるようになっています。詳細に関しては、[マッピングにパラメーターを与える]を参照してください。

(C) 2015-2021 Altova GmbH



- **picture** – このフィールドは`read-binary-file` 関数の直接の出力を受け取ります。

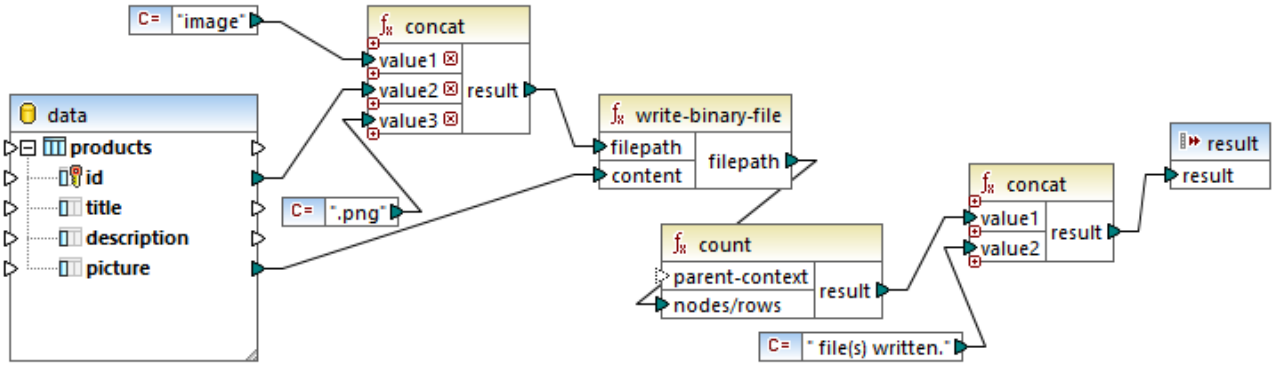
ターゲットコンポーネントはデータベースであるため、マッピングのプレビューは疑似 SQL スクリプトを生成しますがデータベースに変更を送信しません。実際のスクリプトをデータベースに対して実行するには「**出力 | SQL-スクリプト の実行**」を選択します。

6.11.2　例: バイナリファイルの書き込み

下で説明されているマッピングはBLOB 値をSQLite データベースから読み込み ディスクにイメージファイルを書き込みます。データベースには以下の列（フィールド）を持つ**products** と呼ばれるテーブルが存在します:

- **id** (*整数*、レコードの一意の永続的なシリアル番号です)
- **title** (*テキスト*、製品のタイトル)
- **description** (*テキスト*、製品の説明)
- **picture** (*ブロブ*、製品のイメージ)

このサンプルのスコープのために**id** と**picture** フィールドは関連性があります。

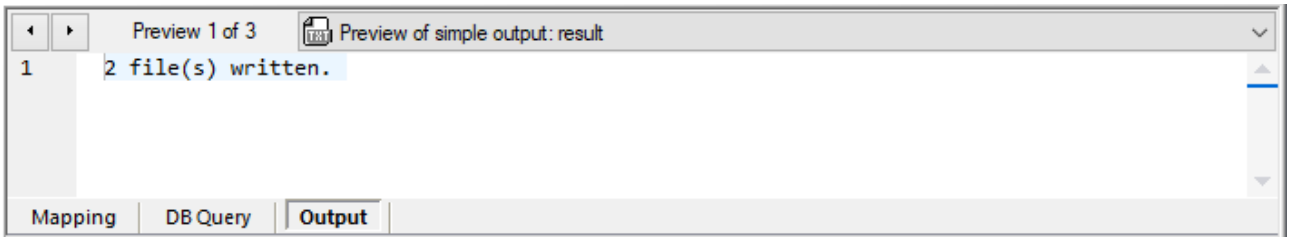


このマッピングのゴールは**products** テーブルからすべてのピクチャを抽出しディスクにファイルとして書き込みます。上記のとおり`write-binary-file` 関数が目的を達成するために使用されています。最初の引数 **filepath** はファイルパスを各イメージのために受け取りま

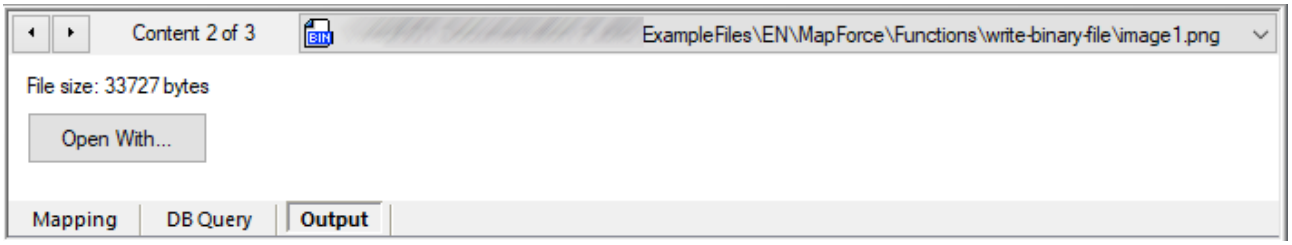
す。ファイルの上書きを回避するためにパスは一意である必要があります。ですから用語 [image] を持つ各レコードの一意のデータベース **id** を連結し、[.png] ファイル拡張子を追加することにより生成されています（全てのピクチャはPNG の書式を持っていることが想定されます）。パスはマッピングと同じディレクトリに保存されるようにパスは相対的です。

2番目の引数 **content** はデータベース内に保存されているバイナリコンテンツを受け取ります。

`count` 関数はすべての生成されたファイルのカウントを返し、この数値を文字列 [書き込まれたファイル] と連結します。これはマッピングにより生成されたファイル数に関するレポートを提供します。このサンプルでは、データベースは2つのレコードのみを含んでおり、マッピングの出力はこれを反映しています:



前記の通り、マッピングがプレビューの実行で作動する場合この関数は一時的なファイルを生成します。各ファイルをプレビューする場合、矢印を使用して興味のあるレコードをナビゲートし、**以下で開く** ボタンをクリックし、イメージエディターを選択します。



全てのファイルを保存するには **生成された出力を保存** または **全ての生成された出力を保存** ツールバーを適用できる箇所でクリックしてください。

(C) 2015-2021 Altova GmbH

7 関数

MapForce 内では必要に応じてデータを変換するために以下の関数のカテゴリを使用することができます:

- **MapForce ビルドイン関数** – MapForce 内でこれらの関数は定義済みで、これらを文字列、数値、日付、および他のデータ型を扱う広範囲の処理を行い、マッピング内で使用することができます。グループ分け、集計、自動付番、および他の多種のタスクを行うために使用することができます。使用することのできるすべてのビルドイン関数への参照に関しては、関数ライブラリの参照を参照してください。
- **ノード関数とデフォルト** – これらはマッピングコンポーネント上の1つまたは複数の子孫ノードにカスタム処理ロジックを作成し適用する更に特化された関数です。マッピング構造のデータに到達する *前* または到達した *直後* にデータを処理できるようになります。詳細に関してはデフォルトとノード関数を参照してください。
- **ユーザー定義関数** (UDF) – これらはMapForce 内で使用が可能なネイティブなコンポーネント型とビルドイン関数のベースとする、自身で作成が可能な MapForce 関数です。ユーザー定義関数を参照してください。
- **カスタム関数** – これらはXSLT ライブラリ、XQuery ライブラリモジュール、Java .class ファイル、.NET .dll ファイルなどの外部ソースからインポートし MapForce に適用することが可能な関数です。MapForce 内で再利用可能にするには、カスタム関数は(文字列または整数) 単純型のデータを返し、単純型のパラメーターを取る必要があります。詳細に関してはカスタムXSLT関数のインポート、カスタムXQuery 1.0 関数のインポート とカスタムJava と.NET ライブラリのインポートを参照してください。

メモ (構成を必要とすることなく)関数のカスタム外部ライブラリを直接、またはMapForce により認識されているMFF (MapForce 関数ファイル) を構成することによりインポートすることができます。後者のアプローチを使用する場合 Java クラスと.NET アセンブリに加え C++ ライブラリをインポートすることができます。.mff ファイルによりインポートされたライブラリはJava、C# とC++ ライブラリを手動で参照する方法 で説明されている必要条件を満たす必要があります。

以下のロードマップを使用して 特定のタスクに関連した関数にアクセスしてください:

...を実行するには	以下を参照してください...
MapForce 関数、または定数をマッピングに追加する方法	• マッピングに関数を追加する • マッピングに定数を追加する • 関数の検索 • 関数の型と詳細を確認する • 関数引数の追加、または削除
MapForce 内で自身の関数を作成し、同じマッピングまたは他のマッピングで後に使用する方法	• ユーザー定義関数
カスタムXSLT 関数を MapForce にインポートする方法	• カスタムXSLT 関数のインポート
カスタム.NET とJava ライブラリを MapForce に追加する方法	• カスタムJava、C# とC++ 関数ライブラリの追加
MapForce 関数ファイル(.mff) を使用して適応することによりカスタムJava、C# とC++ ライブラリを MapForce にインポートします。このアプローチは上記のアプローチとは異なり手動の構成を必要としますがC++ ライブラリのインポートを追加でサポートしています。	• Java、C# およびC++ ライブラリに手動的に参照する方法

7.1 使用方法

7.1.1 マッピングにビルトイン関数を追加する

MapForce には下記の通り、マッピングに追加することのできるすぐに使用可能な多数のビルドイン関数が含まれています。使用することのできるすべてのビルドイン関数への参照に関しては、関数ライブラリの参照 を参照してください。

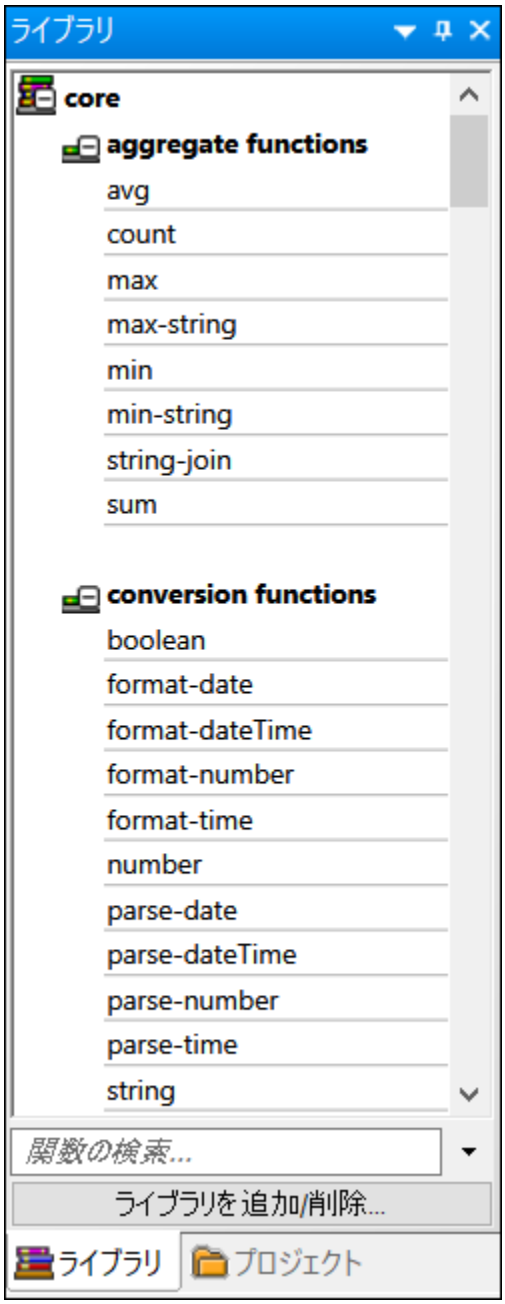
以下を前提条件として下で説明されるアプローチと同様ユーザー定義関数（UDF）をマッピングに追加することができます:

- 同じマッピング内でUDF が既に作成されている、または
- UDF を含むマッピングをローカルまたはグローバルライブラリとして既にインポートしている。

マッピング内で関数を使用する:

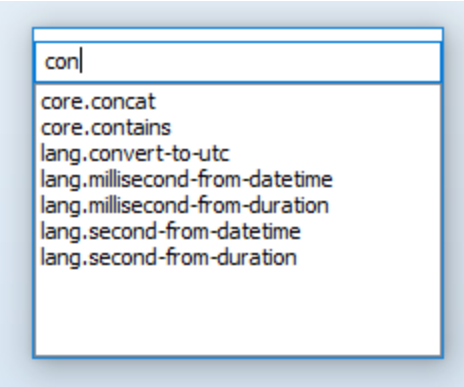
1. 変換言語を選択します。ライブラリウィンドウ内の必要な関数をクリックし、マッピングエリアにドラッグします。
2. 名前別に関数をフィルターするには、ウィンドウの下の部分内にあるテキストボックス内に関数名を入力します: ライブラリウィンドウの下の部分内にあるテキストボックス内で関数名の入力を開始します。

(C) 2015-2021 Altova GmbH



代わりに、定数を次のように追加することができます:

1. マッピングの空白のエリアをダブルクリックし、関数名を開始します。入力されるテキストによりフィルターされるライブラリウィンドウと同じ関数を持つコンボボックスが表示されます。各関数に関する詳細を表示するヒントを確認するには、リスト内から関数を選択してください。



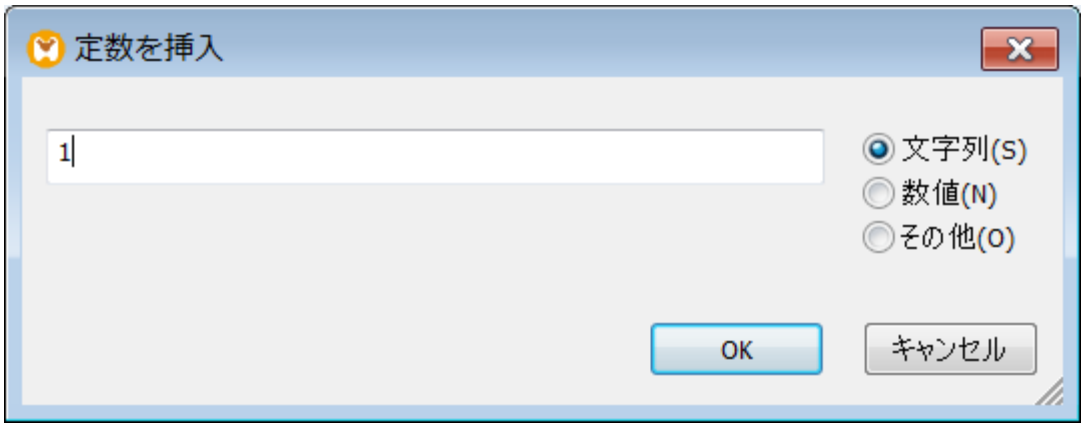
2. 必要とされる関数を選択し、マッピングに追加するために **Enter** を押します。関数を選択することなく、コンボボックスを閉じるには、**Escape** を押し、ボックス外のエリアをクリックします。

7.1.2 マッピングに定数を追加する

定数を使用することによりマッピングにカスタムのテキスト、または数値を与えることができます。定数の値は、その名前が示すように、マッピングのライフタイムの期間中、変化せず同じままの状態でいます。

マッピングに定数を追加する方法:

1. 以下の1つを行います:
   a. 「**挿入**」メニューから、「**定数**」をクリックします。
   b. マッピングを右クリックし、コンテキストメニューから「**定数を挿入する**」を選択します。

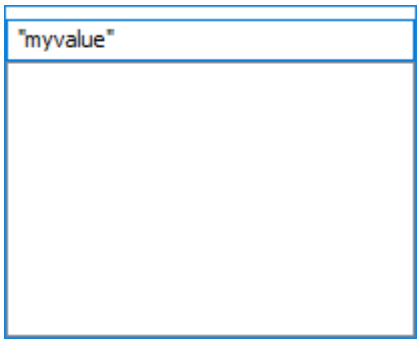


2. 定数の値を入力し、データ型(「文字列」、「数値」、「その他」)を選択し、「OK」をクリックします。

代わりに、定数を次のように追加することができます:

1. 空のマッピングエリア内をダブルクリックします。
2. 以下の1つを行います:
   a. 文字列の定数を追加するには、二重引用符の後に定数を入力します。終わりの二重引用符は任意です。

(C) 2015-2021 Altova GmbH



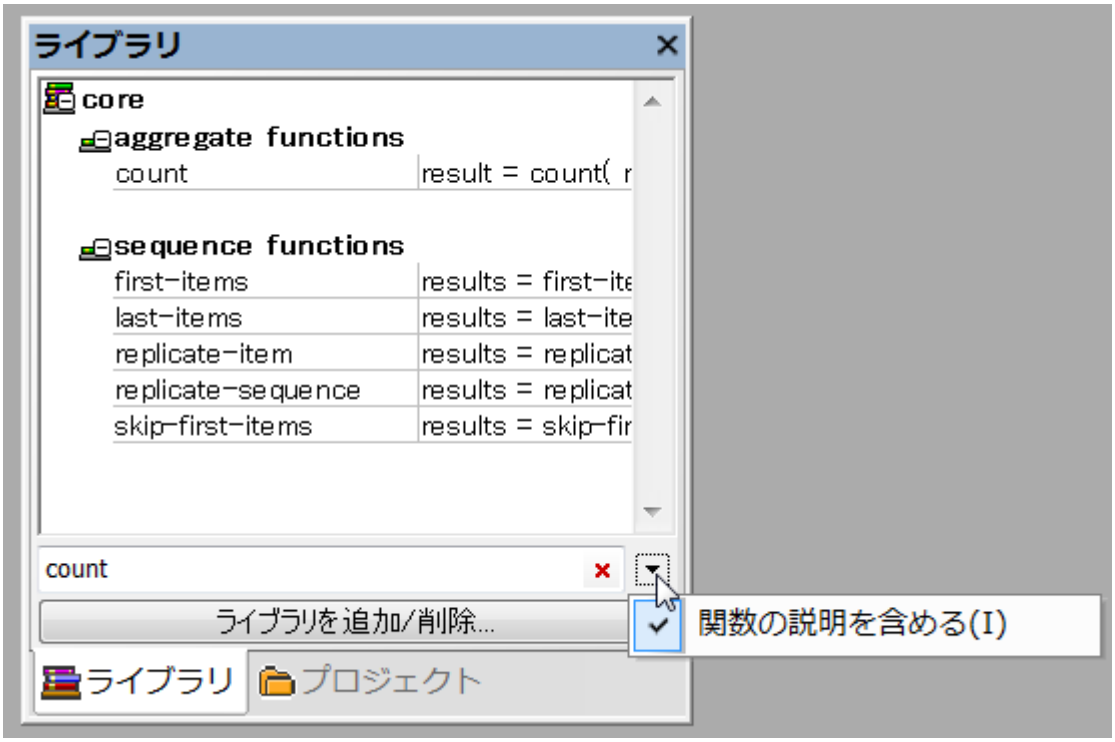
b. 数値の定数を追加するには、数値を入力します。

3. 「Enter」を押します。

7.1.3 関数の検索

ライブラリウィンドウ内で関数を検索するには、ライブラリ ウィンドウの下の部分内にあるテキスト ボックス内で関数名の入力を開始します。

デフォルトでは、MapForce は関数名、または説明テキストにより検索することができます。検索から関数の説明を除外するには、下向きの矢印をクリックし、「関数の説明を含む」オプションを無効化します。



検索をキャンセルするには、「Esc」キーを押す、または「 × 」をクリックします。

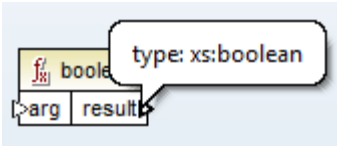
ライブラリ ウィンドウ内で使用することのできる関数は、現在選択されている変換言語によりことなります。次を参照してください: 変換言語の選択。

現在アクティブなマッピング内での関数のすべての発生を検索するには、ライブラリ ウィンドウ内の関数名を右クリックし、「全ての呼び出しを検索する」をコンテキスト メニューから選択します。検索結果 がメッセージ ウィンドウ内に表示されます。

7.1.4 関数の型と詳細を確認する

入力、または出力の引数のデータ型を確認する

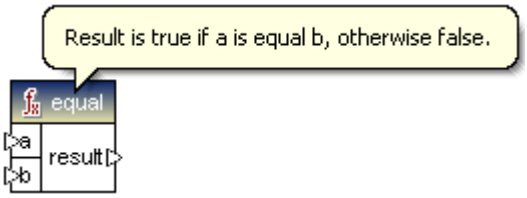
1. 「ヒントの表示」 ツール バーボタンが有効化されていることを確認してください。
2. 関数の引数の部分にマウスを移動させ、ポイントします。



(C) 2015-2021 Altova GmbH

関数の説明を確認する:

1. 「**ヒントの表示**」 ツールバーボタンが有効化されていることを確認してください。
2. 関数のマウスを移動します（マッピングエリア上のライブラリペイン内でこれらの機能を使用することができます）。

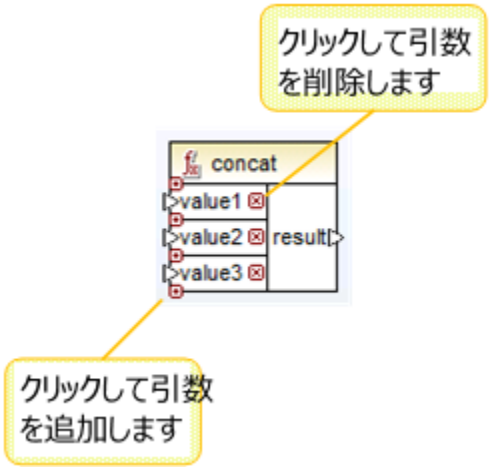


7.1.5 関数引数の追加、または、削除

コンテキストにより必要とされる通り必要な数量のパラメーターを追加できるという意味で MapForce ビルトイン関数の一部は拡張可能です。このような関数のよい例は全ての文字列を連結を必要とする `concat` です。

（このような振る舞いをサポートする関数のための）関数の引数の追加、または削除方法：

- パラメーターの横の**パラメーターの追加**（ ）または**パラメーターの削除**（ ）を使用して、パラメーターを追加、または削除します。



シンボル上に接続をドロップすると、パラメーターが自動的に追加され、接続されます。

7.2 デフォルトとノード関数

MapForce がデータを読み取り、または データを書き込む場合、ソース、または目的ファイル、または データベースに空、または null フィールドが存在する場合があります。このような場合を取り扱うために、MapForce には多数の内蔵の関数、if-else 条件、不足している、または null データを置き換える、または不足しているフィールドが検出されると例外をスローする他のメカニズムが搭載されています。

更に、複数のアイテムのためのデフォルトの値を同時に設定する必要がある場合があります（例えば、XML 要素の全ての子）。代わりに、空の値をテキスト（例えば、"n/a"）で置き換えるシンプルな関数を作成し、複数のアイテムにこの関数を適用する必要がある可能性があります。通常の環境では、これを行うために、マッピング上で同じ関数を複数回コピーして張り付ける必要があります。これを行うと、マッピングを乱雑にするだけではなく、理解し難くします。簡単な代替とし、このチャプターの主題であるデフォルトとノード関数を使用することができます。

メモ マッピングのターゲット言語がBUILT-IN の場合、デフォルトとノード関数はサポートされています。このようなマッピングを生成された C#、C++、Java プログラムコードから、または生成された XSLT/XQuery から実行する場合、変換はサポートされません。サーバー側では、MapForce Server Advanced Edition を使用してこのようなマッピングを実行することができます。

用語 "ノード関数" は、関数をノードレベル（XMLノード、またはCSV、JSON、EDI、またはデータベースフィールド）で適用することができることを意味します。ノード関数を単一のアイテム、または複数のアイテムに1度に適用することができます。同様に、用語 "default" は、単一のアイテム、または複数のアイテムのために ノードレベルに適用するデフォルトの値を意味します。

同じ処理ロジックを構造の中で子孫アイテムに複数回適用する場合、デフォルトとノード関数は特に役に立ちます。例:

- 空、または null 値が発生すると、他の値と置換え、全ての子孫アイテムに対して再帰的に行うことができます。
- 特定の値が発生すると、（例えば、"N/A"）、他の値と（または空の文字列と）置き換えることができます。全ての子孫アイテムに対して再帰的に行うことができます。
- データベーステーブルから読み取る際に、データベース null 値を空の文字列、またはカスタムのテキスト（数値のフィールドの場合は、0 の値とと置き換えることができます。
- ソースデータベースから来る全ての値のために末尾の空白を削除することができます。
- ターゲットファイル、またはデータベースに書き込まれる全ての値にカスタムプレフィックス、またはサフィックスを追加します。
- 特定の値が発生する都度、null 値を作成します。

メモ "null" と "空の" 値を区別することは重要です。null 値は"nothing"（値が不足している状態）を意味し、空の値は空の文字列（''）を意味します。MapForce はノード関数 を含む両方のケースに対応する多種の方法を提供しています。値は通常からの文字列（''）です。MapForce は、ノード関数とデフォルトを含む複数の方法を与えます。

7.2.1 デフォルトとノード関数の作成方法

マッピング上のほぼすべてのアイテム（ノード）のためにノード関数、またはデフォルトを作成することができます。この過程を*ルール* と呼ぶことにしましょう。

デフォルト、またはノード関数を作成するために、デフォルト、またはノード関数を定義する箇所でアイテム（ノード、またはフィールド）を決定します。これは（子孫を持たない）"リーフ" アイテム、または子孫を持たないアイテムであることができます。後者の場合、関数、またはデフォルトを全ての子孫アイテムにも適用することができます。条件を満たすアイテムを型、または名前別に正規表現を使用して検索することができます。

必要条件:

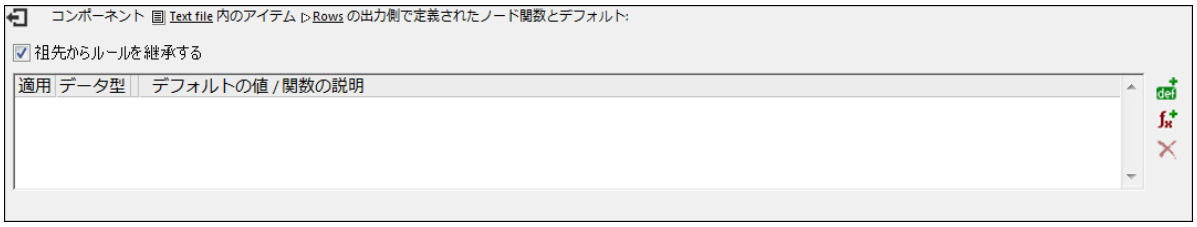
- デフォルト、またはノード関数をターゲットコンポーネントの入力側、またはソースコンポーネントの出力側に作成することができます。作成する側を確立するには、次を参照してください: 入力と出力サイドの選択。
- デフォルトとノード関数は、ソースとターゲット間の接続型は、"ソースを優先型" または ターゲットを優先型"です。"全てコピー" 接続はサポートされません。具体的には、ノード関数とデフォルトは"全てコピー接続"に対して適用することができません。"全てコピー" 接続を持つノードは、単純な値を持つノード関数とデフォルトに対してのみ適用することができます。例えば、単純型のコンテンツと

属性を持つ要素など。このため、関数を定義、または子孫を持つノード上でデフォルトを設定する場合、ソースとターゲット間の接続型はで"全てコピー"あってはなりません。接続型を確認、または変更する場合、接続を右クリックし、コンテキストメニューからターゲット優先(**標準マッピング**) を選択します。詳細に関しては、次を参照してください: 接続の種類。

- デフォルトの作成、またはルートアイテム上のノード関数はファイルノードのためにサポートされません。このノードを使用すると、ファイルの作成と読み取りを動的に行うことができます。次を参照してください: 複数の入力または出力ファイルを動的に処理。

デフォルト、またはノード関数の作成方法:

1. アイテム(ノード) を右クリックし、**デフォルトとノード関数 | 入力デフォルトとノード関数** をコンテキストメニューから選択します。(または、場合によっては、**出力デフォルトとノード関数**)。**マッピング** ペインはグリッドを上に表示します。または、この場合、コネクタを右クリックすると、MapForce はそのサイドのためにノード関数を表示します。**マッピング**ペインは、上の部分に表示されます。例:



ルールを定義する箇所に親が存在する場合、親に反対するルール(ノード関数、またはデフォルト) が存在する場合があります。このようなルールを継承する場合、**祖先からルールを継承** チェックボックスを選択します。継承の詳細に関しては、次を参照してください: デフォルトとノード関数のしくみ。

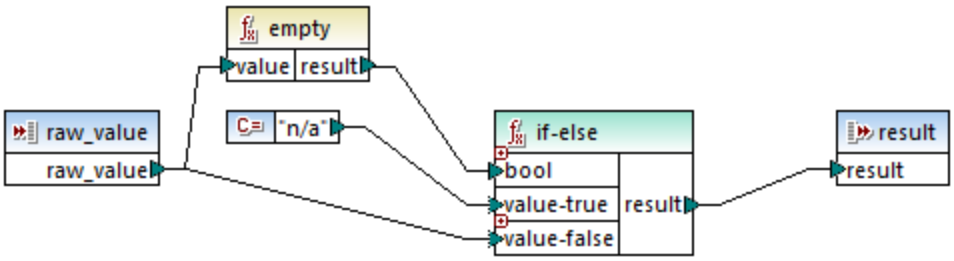
2. 以下の内の1つを行ってください:
   a. デフォルトを追加するには、**デフォルトを追加** ( ) をクリックします。
   b. 関数を追加するには、**関数を追加** ( ) をクリックします。

これにより新しいルーツが作成されます(**マッピング** ペインの上のグリッド内の行でこのルールのために条件を選択することができます)。ルールを以下のように構成します:

適用	現在のアイテム、または全ての子孫アイテムに、深さに関係なく、または直接の子のみにルールが適用されるかを決定します。 ステップ1 で選択したアイテムに子孫が存在しない場合、"現在のアイテム" が唯一の選択となります。
データ型	**省略記号** ボタン をクリックし、ダイアログボックスから データ型を選択します。ルール(デフォルト、またはノード関数) はこのデータ型(または、生成されたデータ型)を持つアイテムのみに適用されます。詳細に関しては、ノード関数とデフォルトを条件付きで適用するを参照してください。 ステップ1 で選択されたアイテムに子孫が存在しない場合、アイテムのデータ型は選択された項目のみに適用されます。
デフォルトの値 / 関数の説明	デフォルトの( ) 型を定義する場合、選択されたアイテムのために設定するデフォルトの値(適用できる場合、は全ての子孫)をここに入力します。デフォルトとして空の文字列を設定する場合、このフィールドを空のままにしておいてください。

	関数（ ）を定義する場合、このフィールドは情報のためのみに使用されます。このフィールドは関数の概要を表示します。次のステップ内の関数を定義することができます。

3. 関数を定義する場合、マッピングエリアは関数の入力（"raw_value" の下に示される）と出力（"result"）を表示を表示するために変更されます。このマッピングエリアは、ミニマッピングであり、同じ一般的なルールが通常のマッピングを定義する時にも適用されます。例えば、関数のボディは以下のようになります:



上記のノード関数は、空の値を値 "n/a" と置き換えます。このサンプルの詳細に関しては、次を参照してください: 例: 空のCSV フィールドの置換。

以下の点に注意してください:

- ノード関数内では、ビルトイン関数、編集、if-else 条件などのこのコンテキスト内で意味を成す特定のMapForce コンポーネントがサポートされます。XML、JSON、EDI、またはデータベースなどの複雑な構成はサポートされません。インラインのユーザー定義関数、またはジョインコンポーネントをノード関数に追加することもサポートされません。
- ノード関数は、最多で1つの入力パラメーターを持つ、またはパラメーターを持たないでいることができます。入力パラメータは常に「raw_value」と呼ばれます。関数のために必要ない場合でも、関数入力コンポーネント（"raw_value"）を削除しないでください。それ以外の場合、マッピングの実行時にエラーが表示されます。編集の出力も同様です。メニューコマンドを使用して「**関数 | 入力の挿入**」偶然に削除された入力コンポーネントを復元する必要があります。
- 入力パラメーター（"raw_value"）が既存の場合、後に追加される入力パラメーターはメタデータパラメーターになります。このようなメタデータを使用して現在のノードに関する追加情報を取得し、関数の内部で使用することができます。ノードメタデータをノード関数に提供するを参照してください。
- 一部の場合では、ノード関数を以下の方法で作成すると便利です: **ライブラリ** ウィンドウから入力、または出力 コネクタ ▷ にノード関数をドラッグする。これは `right-trim` または `uppercase` などの単純型の関数を使用する場合便利です。

ノード関数を終了する方法:

- **マッピング** ペインの左上の**戻る** ボタンをクリック、または「Escape」を押します。

既存のルールの編集と削除も参照してください。

前に定義されたノード関数を確認、または変更する方法:

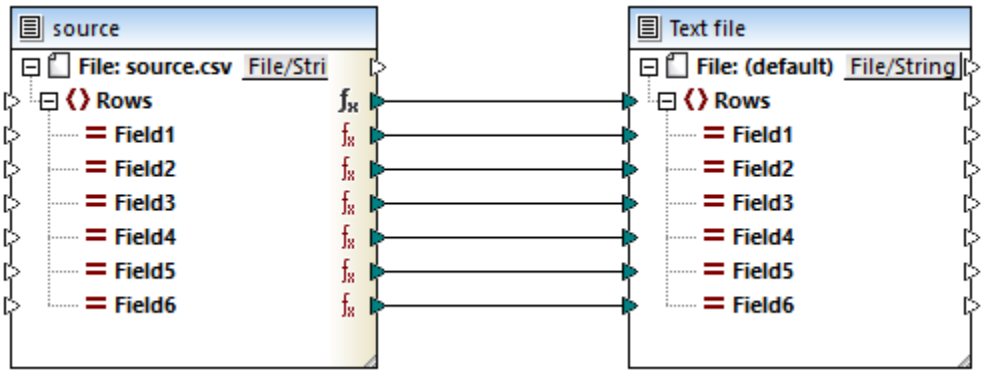
1. メインマッピング内で、注目するノードの横の アイコン（黒色または赤色）をクリックします。このアイコンはノード関数を以前に定義した箇所に存在します。ノード関数アイコンの意味に関する詳細は次を参照してください: デフォルトとノード関数のしくみ。
2. **マッピング** ペインの上のグリッドから関数を選択し、Edit をクリックします。Edit ボタンが存在しない場合、関数は、現在のアイテムではなく、祖先により定義されている可能性があります（前のステップを参照してください）。

(C) 2015-2021 Altova GmbH

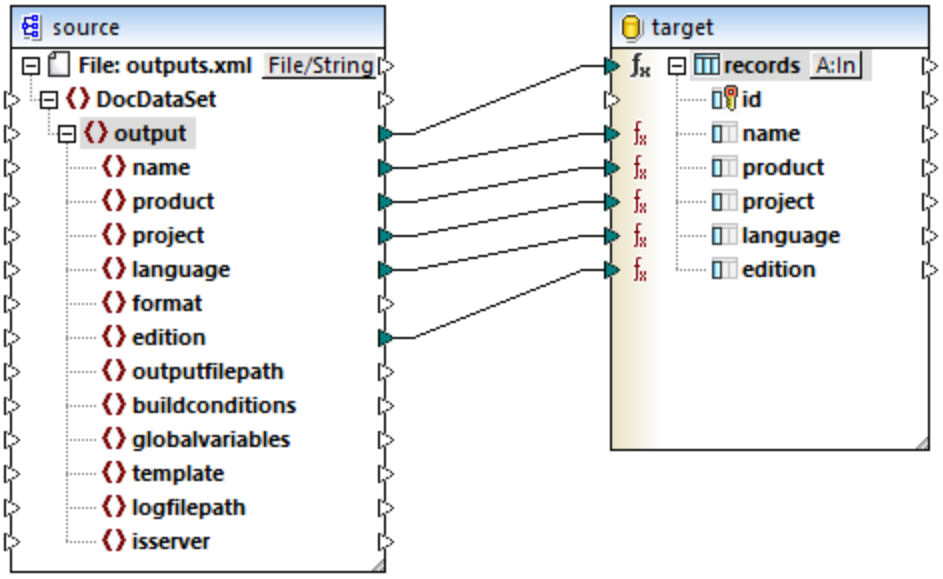
7.2.1.1 入力と出力サイドの選択

MapForce コンポーネント は常に入力側と出力側が存在し、必要に応じてどちらかのサイドでノード関数とデフォルトの値を定義することができます。更に理解を深めるために、マッピングの仕組みについて思い出してください！ 最初にソースコンポーネントからデータを読み取り（例えば、データベース、またはファイル）、（例えば、関数、またはフィルターを使用して）データを任意で処理し（例えば、データベース、またはファイルなどの）ターゲットコンポーネントにデータを書き込みます。この点を考慮して、ノード関数を適用し、異なるステージでデフォルトを設定します:

- ソールファイル、またはデータベースからデータが読み取られたすぐ後（マッピングにより処理される前）。例えば、下のマッピングでは、関数、またはデフォルトは、ソースコンポーネントの出力側で定義されています（ノード関数、またはデフォルトを示す fx アイコンが存在することに注目してください）:



- データがターゲットファイル、またはデータベースに書き込まれるすぐ前（そして、全ての中間の処理が終わった後）。例えば、下のマッピングでは、関数、またはデフォルトがターゲットコンポーネントの入力側で定義されています:



- マッピング処理内の中間のステージで、例えば、マッピングが複雑型の中間関数（XML 構造など）を含む場合、XML 構造に提供される前、または構造により返されたすぐ後にすべての値を削除することができます）。

上記を要約すると、コンポーネント上の入力、または出力、のどちらかでノード関数を定義することができます。入力側の関数（またはデフォルト）は、コンポーネント上の対応するアイテムを入力する前にデータを処理します。他方、出力側で定義される場合、データを対応するアイテ

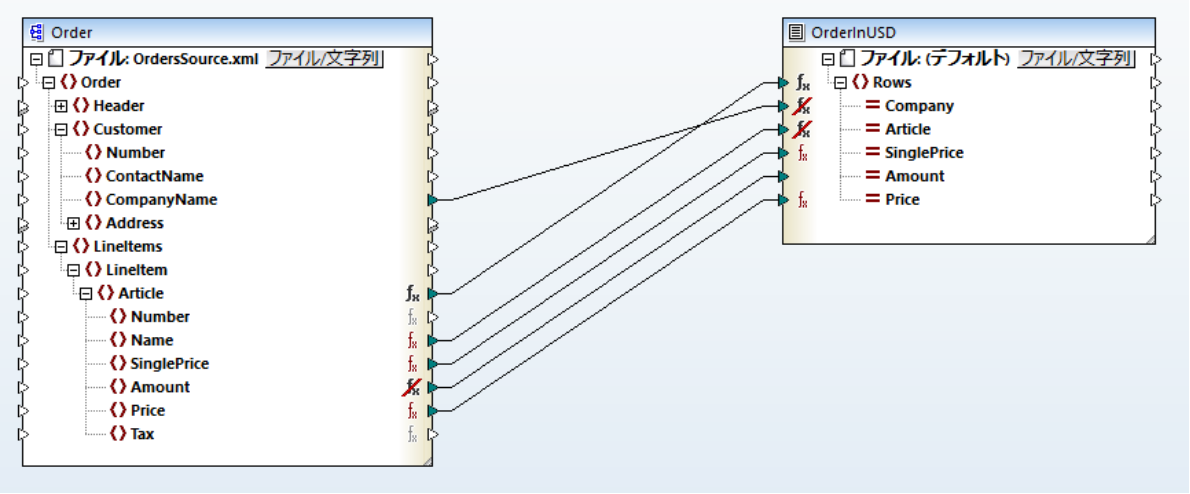
ムから返された後すぐに処理します。関数が定義されている箇所でアイテムが子アイテムを持つ場合、デフォルトの関数を全ての子アイテムに対して適用することもできます。

7.2.2 既存のルールの編集と削除

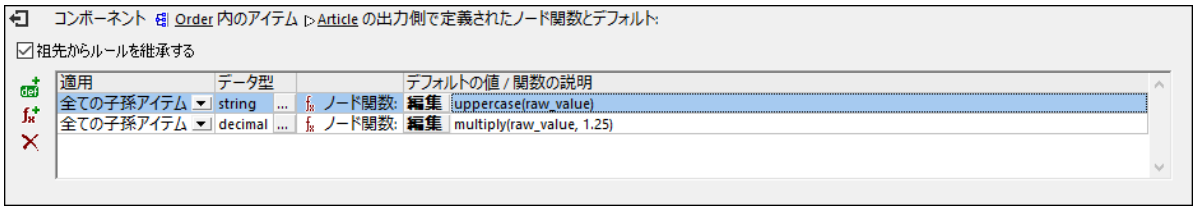
デフォルト、またはノード関数のためにルールがマッピングに含まれている場合、注目されているノードの横に fx アイコン（黒色、または赤色）と共に表示されています。以下に表示されているように、ルールを確認、変更、または削除することができます。

確認、または編集のためにデフォルト、またはノード関数を開く方法:

1. メインマッピング内で、注目されているノードの横の fx アイコン（黒色、または赤色）をクリックします。このアイコンは、デフォルト、またはノード関数のためのルールが依然定義されている個所に存在するノードを表しています。例えば、下のマッピング内で、ソースコンポーネントの **Articles** アイテム、および、ターゲットコンポーネントの **Rows** アイテム上でルールは定義されます。このデモマッピングは以下のパスで見つけることができます: **<マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥OrderInUSD.mfd**。



2. fx アイコンをクリックした後、このノード上で定義されている各 デフォルト、またはノード関数のための行が含まれている上のグリッドを観察してください。対応する行を選択し、必要に応じて変更を加えます。



3. 関数の場合、関数の実行（マッピング）を変更する場合、Edit ボタンをクリックします。Edit ボタンが存在しない場合、関数は以前の祖先に定義されている可能性が高いです。この場合、ルールが定義されている個所のアイテム上の fx アイコンがクリックされていることを確認してください（下のメモを参照してください）。

> fx アイコンはルールが定義されている個所を示しており、定義されている個所ではありません。アイコンの意味の詳細に関しては、デフォルトとノード関数のしくみを参照してください。

ルールを削除する方法:

- 上にあるグリッドからルールを選択して、「関数 / デフォルトの削除」 ボタンをクリックします。

メインマッピングを完了する方法:

- 「マッピング」ペインの左上の「戻る」 ボタンをクリックして、「Escape」を押します。

7.2.3 デフォルトとノード関数のしくみ

デフォルトとノード関数の作成方法 で説明されているとおり、ノード関数、またはデフォルトをマッピング上のほぼ全てのアイテム(ノード)のために作成することができます。この過程をルールの定義と呼ぶことにしましょう。柔軟性のためにルールには以下の重要な特徴があります:

- *継承*: 子孫を持つアイテム上のルールを定義する場合、このオプションを無効化しない限り、デフォルトで、ルールは子孫に継承されます。関数を定義する箇所でアイテムがネストされている子アイテムを持つ場合、直接子アイテムのみにルールを適用する、または全ての子孫アイテムにルールを適用するかを決定することができます。
- *データ型のフィルタリング*: MapForce は各アイテムのデータ型に応じてルールを条件に従い適用することができます。これにより、例えば、文字列の型の全てのアイテムのための特定のデフォルトの値(または関数)を適用、または数値型の全てのアイテムのために異なるデフォルト(または関数) を適用することができます。

上記の振る舞いは特定の影響を与えます。具体的には、ルールの定義と、ルールの適用を区別して理解する必要があります。一部のアイテムのためにルールを定義しても、そのルールがアイテムに影響を与えるとは限りません。例えば、型 "文字列" のアイテム上でルールを定義し、ルールの条件としてデータ型 "数値" を選択すると、ルールは適用されません。ルールの条件(データ型と継承)が満たされた場合アイテム、またはその子孫にルールが適用されます。

定義し適用するルールに関して理解するには、MapForce マッピングのための視覚的なヒントに注意してください:

アイコン	詳細
	このアイコンは、(黒色) はこのアイテムのためにルールが定義され、全ての子孫に影響を与える可能性があることを示しています。アイコンをクリックして、ルールを変更、または削除します。
	このアイコンは、(赤茶色) はアイテムが祖先レベルで定義されているルールは見込みがある(適用可能である) ことを示しています。すなわち、このアイテムに適用(または、影響) することができるルールは存在します。
	このアイコンは、(太字、赤茶色) は、このアイテムのためにルールが定義されている、またこのアイテムにルールが適用されることを示しています。祖先からルールを継承するアイテムの場合この詳細があてはまり、同時に、アイテムのためにルールが定義されていることを指します。
	このアイコンは、このアイテムに適用することのできるルールが意図的にブロックされていることを指します。ルールを適用しない箇所で特定のアイテムのためにこの操作を行うことができます。
	このアイコンは、(グレーアウト) は、このアイテムに適用されるアイテムは無効化されています。例えば、このアイコンマッピングに接続されていないアイテムのために表示されます。

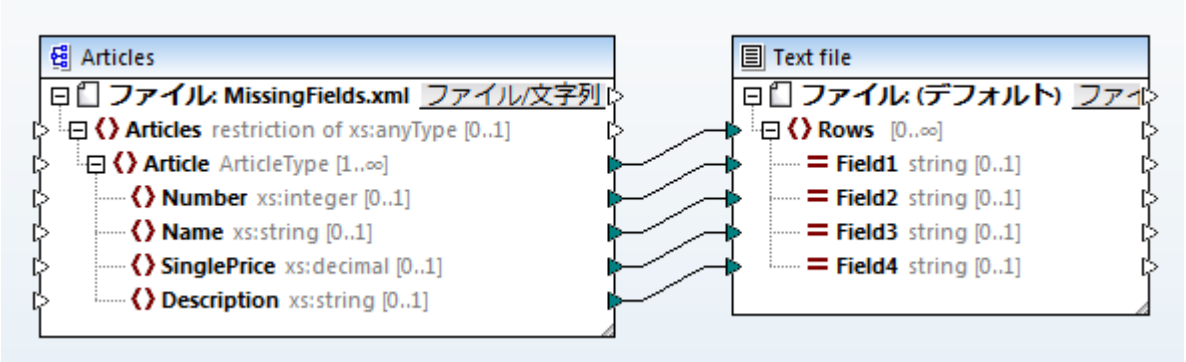
通常、複数のノード関数、またはデフォルトがつ、および、同じアイテムのために存在する場合、次のルールに注意してください:

アイテムに適用することのできるノード関数とデフォルトの数に関わらず、マッピング上の単一のアイテムのために、MapForce は1つのノード関数、および、1つのデフォルトのみを適用します。

通常、以下のようにルールを解釈することができます:

- 同じアイテムのために複数のルールが存在する場合、MapForce はそのアイテムに近いルールを適用します。例えば、3回ノード関数を定義したと仮定します: **Company** と呼ばれるルート XML ノードでは、**Department** と呼ばれる子ノード、および、孫 **Employee**。この場合、MapForce は近いため、**Employee** アイテム上で定義されている **Employee** アイテム関数を適用します。関数が存在しない場合、直接の祖先 **Department** の関数を検索します。**Department** のために関数が存在しない場合、この場合 **Company** である ルートノードを更に検索します。継承は任意です。継承を無効化するには、**祖先からルールを継承** チェックボックスを無効化します。このチェックボックスがクリアされていると、アイテムには "ブロック済みのルール" アイコンが表示されます。
- 同じアイテムに複数のルールが存在する場合、MapForce はマッピングペイン上の上のグリッドから最初のルールを適用します。このグリッドのルールの順序を変更するには、ルールをクリックし、グリッド内でルールをクリックし、グリッド内の新しい箇所にドラッグアンドドロップします。現在のアイテムのために定義されている場合のみグリッドのルールをドラッグできることに注意してください。継承されたルールの位置は変更することができず、継承を有効化、または無効化することはできます。

仕組みを説明するにあたり、次のパスで使用することのできるマッピングを使用します: **<マイドキュメント>\Altova\MapForce2021\MapForceExamples\Tutorial\MissingFields.mfd**.



MissingFields.mfd

上記のとおり、このマッピングはソース XML ファイルからデータを読み取りターゲットテキストファイル(固定長フィールド)に書き込みます。ソース XML ファイル内では、要素 **Article** には異なる型の次の子要素が存在します: "integer"、"string"、および "decimal"。かそれぞれの子要素は任意です(`minOccurs="0"`)。このため、これらの要素がソース XML 内で存在しない場合、デフォルトの値を与える必要がある可能性があります。それ以外の場合、ターゲット CSV ファイル内で空のフィールドが存在する場合があります。例:

```
   T-Shirt   25    Available in all sizes
2            2.3
3  Pants           Limited stock
4  Jacket    57.5
```

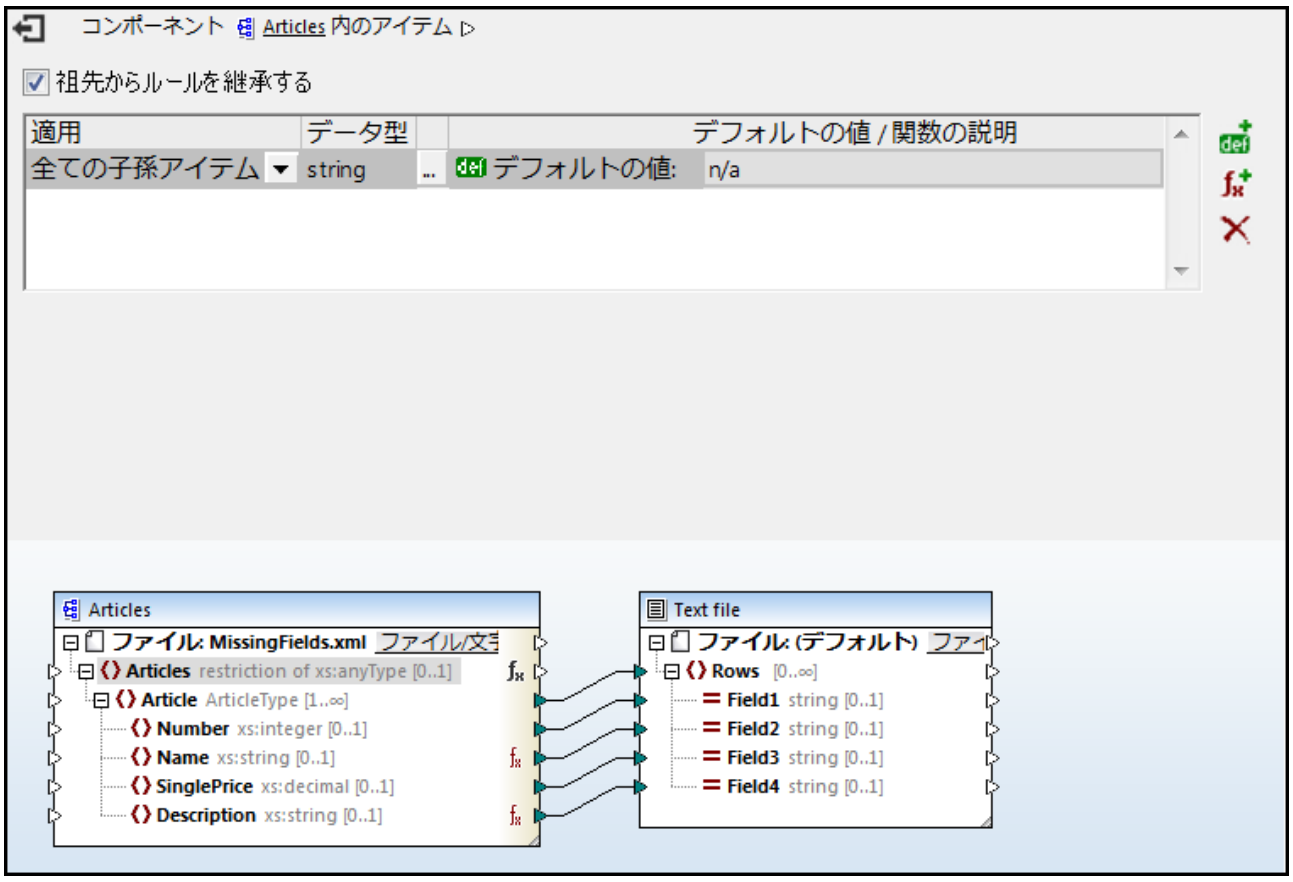
下記にルールを使用して不足しているデータを扱う異なる方法、および、マッピング結果をルールがどのように影響するかについて説明されています。特定のアイテムに対して複数のルールが存在する場合、どのルールを管理するかについて理解することができます。

サンプル1: 全ての文字列アイテムのためにデフォルトを提供する方法

マッピング **MissingFields.mfd** 内で、次の条件が存在すると仮定します: ***Article** の子が型 "string" であり、不足している場合、デフォルトの値として "n/a" を使用します。*

この条件を満たすためには、以下のステップに従います:

1. **Article** アイテムを右クリックし、**デフォルトとノード関数 | 出力デフォルトとノード関数** をコンテキストメニューから選択します。
2. **デフォルトを追加** ( ) をクリックします。
3. **デフォルトの値** の下に、"n/a" を入力し、「Enter」を押します。



上のマッピングでは、ルールの条件を以下のように設定することができます: ***Article** の全ての子孫アイテムで、データ型が "string" で、ソース XML 要素が不足している場合、デフォルトの値として "n/a" を使用します。*このサンプルでは、型 "string" の2つのアイテムである **Name** と **Discription** が存在し、ルールは両方に適用されます。

上記のとおり、ルールが定義される箇所で、アイテムの横に アイコンが表示されます。ルールが定義される箇所で、アイテムの横に アイコンが表示されます。この時点でマッピングをプレビューすると、出力内で不足していた文字列は、"n/a" と置き換えられていることに気が付きます:

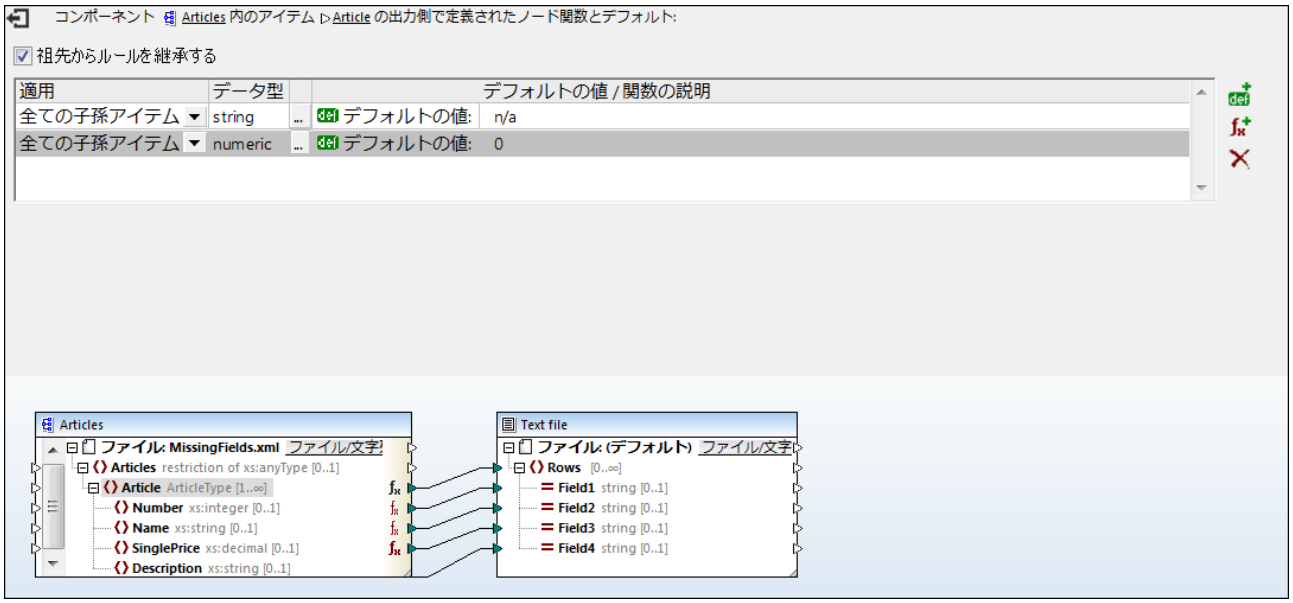
```
   T-Shirt   25    Available in all sizes
2  n/a       2.3   n/a
3  Pants           Limited stock
4  Jacket    57.5  n/a
```

サンプル2: データ型をベースにデフォルトを条件付きで提供する方法

文字列アイテムのためのデフォルトに加え、数値型のアイテムにデフォルトの値 **0** を提供すると仮定します。この条件を満たすためには、以下のステップに従います:

1. **Article** アイテムをクリックします。

2. **デフォルトを追加**（ ）をクリックして、次の条件を持つ2番目のルールを追加します:



上のマッピングでは、ルールの条件は以下のように設定することができます:

- ***Article** の全ての子孫アイテムのためにデータ型が "string" で、ソース XML 要素が不足している場合、デフォルトの値 "n/a" を使用します。*
- ***Article** の全ての子孫アイテムのためにデータ型が数値で、ソース XML 要素が不足している場合、デフォルトの値 "0" を使用します。*

結果的には、概要は以下のようになります:

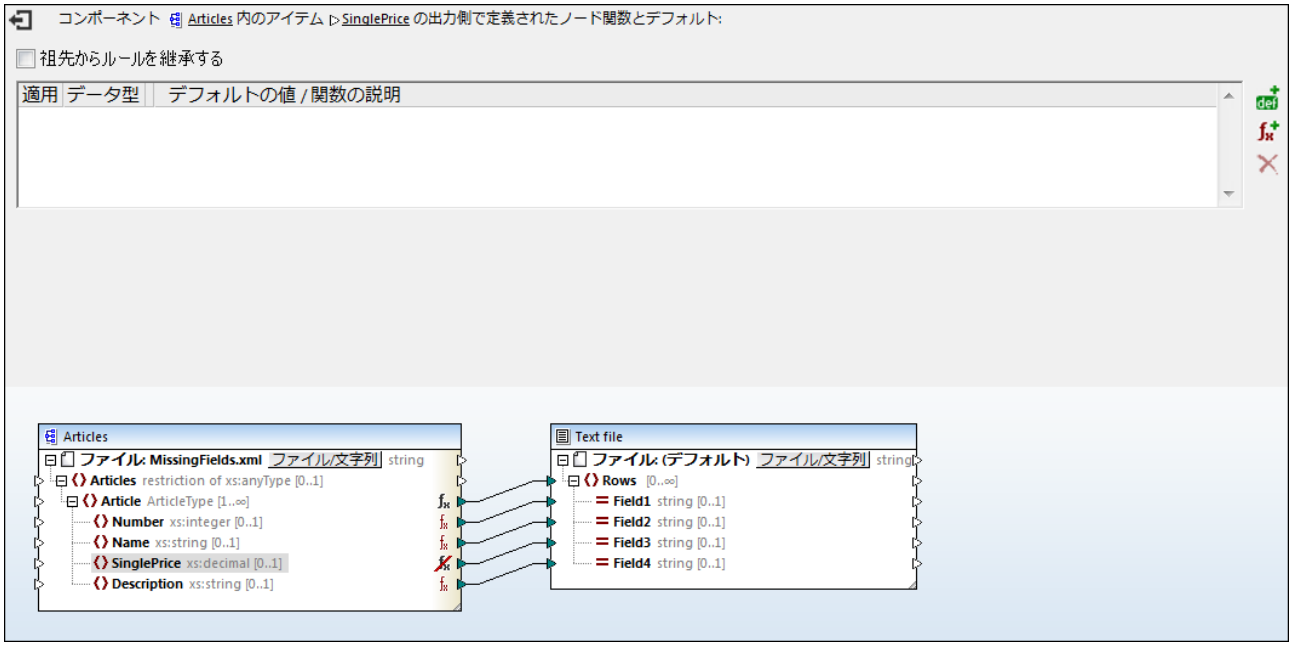
```
0  T-Shirt   25    Available in all sizes
2  n/a       2.3   n/a
3  Pants     0     Limited stock
4  Jacket    57.5  n/a
```

メモ "integer" と "decimal" データ型の両方を含むため、データ型 "numeric" は実際は型のカテゴリです。このため、ルールは **Number** と **SinglePrice** 要素の両方に適用されます。データ型として "decimal" を選択すると、XML スキーマ型階層では、型 "integer" は型 "decimal" からが派生するため、ルールは **Number** と **SinglePrice** の両方に適用されます。("XML Schema Part 2: Datatypes Second Edition"、https://www.w3.org/TR/xmlschema-2 の §3 を参照してください)。データ型 として "integer" を選択する場合、ルールは **Number** のみに適用されます。

サンプル 3: 特定のアイテムのためにルールをブロックする方法

前のアイテム同様全ての文字列と数値アイテムにデフォルトを適用すると仮定します。しかしながら、**SinglePrice** アイテムにデフォルトを適用しないとする場合以下を行います。

この条件を満たすためには、アイテム **SinglePrice** をクリックして、チェックボックス 祖先からルールを継承をクリアします。



上のマッピングでは、アイテム **SinglePrice** は親 **Article** からルールを継承しません。このため、"ブロック済みのルール" アイコン が表示されます。

この結果、対応するフィールドは結果内で空白のまま表示されます:

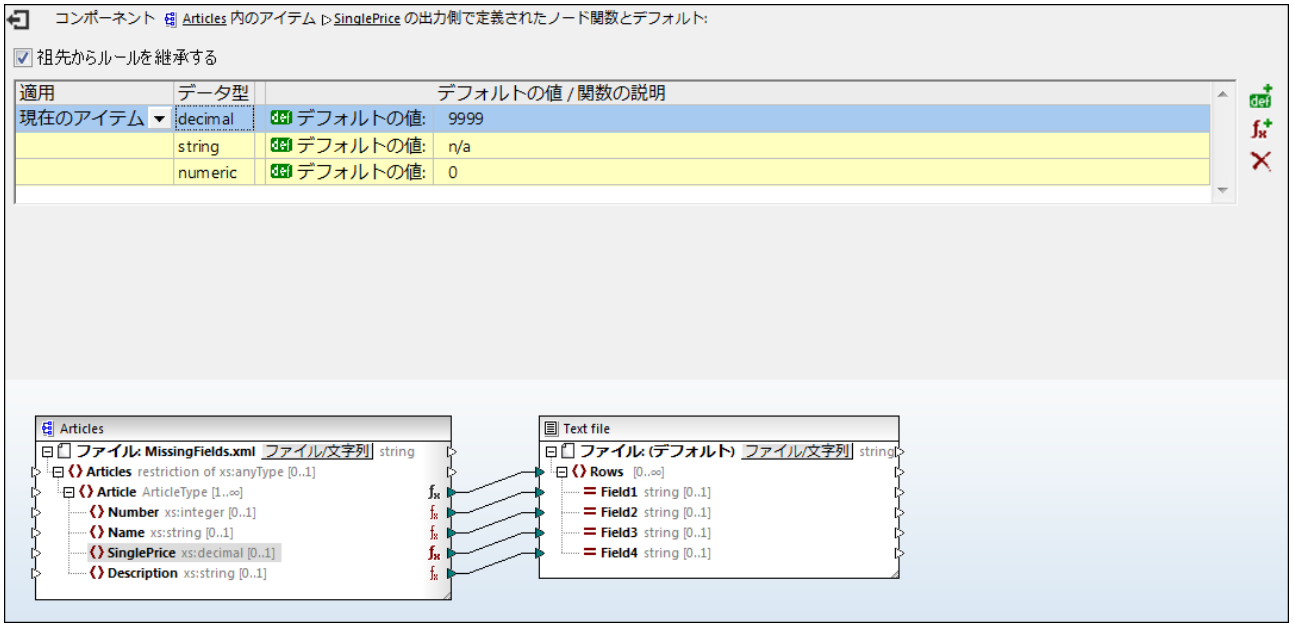
```
0  T-Shirt   25    Available in all sizes
2  n/a       2.3   n/a
3  Pants           Limited stock
4  Jacket    57.5  n/a
```

サンプル 4: 特定のアイテムのための継承されたルールをオーバーライドする方法

全ての文字列に数値を提供すると仮定します。しかしながら、アイテム **SinglePrice** のみ、**9999** を提供するとします。

この条件を満たすために、以下のステップに従ってください:

1. アイテム **SinglePrice** をクリックします。
2. **デフォルトを追加** ( ) をクリックし、**9999** のデフォルトの値を入力します。
3. 任意で、**祖先からルールを継承** チェックボックスを選択します。



メモ 継承されたルールの背景は黄色です。

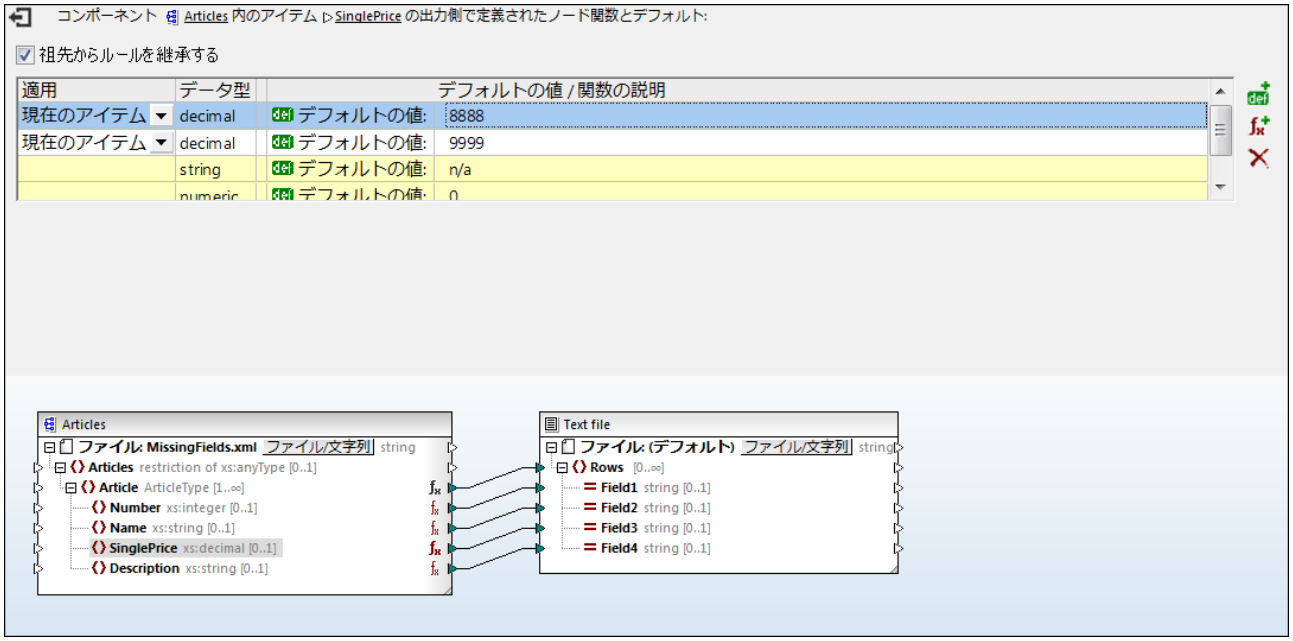
上のマッピングでは、アイテム **SinglePrice** にて起用することのできる 3 つのルールが存在するとします：2 つの継承されたルールと、1 つの直接的なルールが存在します。この場合、アイテム上で直接定義されているルールが優先されます。継承されたルールは無視されるため、概要は以下のようになります:

```
0  T-Shirt   25    Available in all sizes
2  n/a       2.3   n/a
3  Pants     9999  Limited stock
4  Jacket    57.5  n/a
```

サンプル 5: 適用するルールを手動で選択する方法

前のサンプルを更に展開し、アイテム **SinglePrice** のために 1 つアイテムを定義します。**8888** がデフォルトの値です。上記のとおり、現在のアイテムに直接定義されているルールが優先されます。しかしながら、（継承されたルールに加え）2 つのルールが現在のアイテムに存在します。問題は 2 つのデフォルト **8888** または **9999** のどちらが適用されるかです。

(C) 2015-2021 Altova GmbH

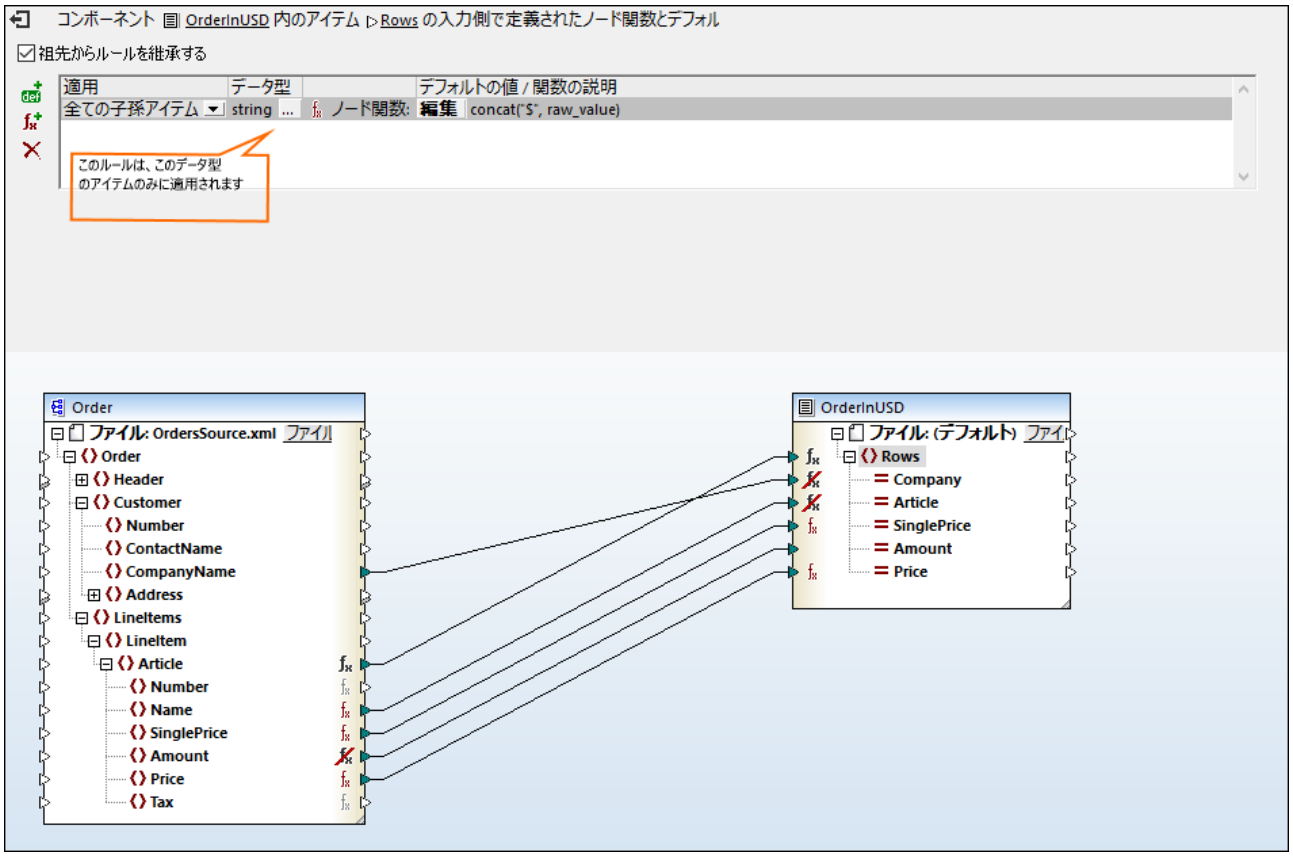


上のマッピングのように同じアイテムに対して複数のルールが存在する場合、優先されるルールをグリッドの上に手動でドラッグします。一番上に表示されているルールが優先されます。このため、**SinglePrice** のデフォルトの値は、グリッドの上に表示されている**8888** です:

```
0  T-Shirt   25    Available in all sizes
2  n/a       2.3   n/a
3  Pants     8888  Limited stock
4  Jacket    57.5  n/a
```

7.2.4 ノード関数とデフォルトを条件付きで適用する

ノード関数、またはデフォルトの(いわゆる「ルール」)を作成すると、子孫を持つアイテム上、または「リーフ」アイテム上で定義することができます。重要な点は、アイテムに子孫が存在すると、ルールのためにデータ型を指定しなければなりません。これは、このデータ型の子孫のみがルールにより影響を受けることを意味します。この概念を更によく理解するために、次のデモマッピングを開いてください:<マイドキュメント>**\Altova\MapForce2021\MapForceExamples\OrderInUSD.mfd**。ターゲットコンポーネント内の「行」アイテムの横の fx アイコンをクリックしてください。



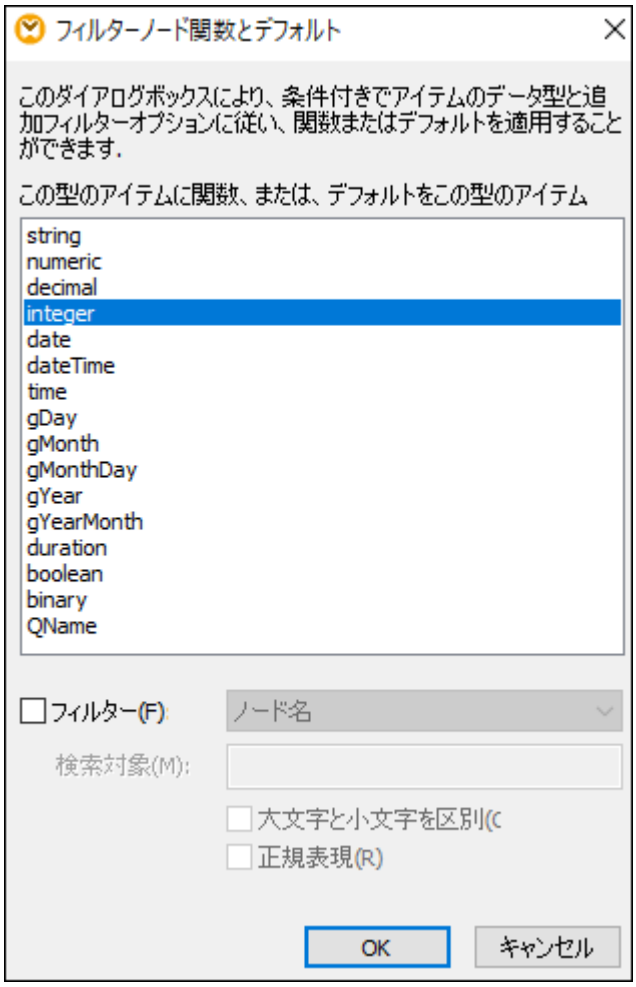
OrderInUSD.mfd

上のマッピングは、XML ファイルからデータを読み取り、ターゲット CSV ファイルに書き込みます。ターゲット CSV コンポーネント内では、**Rows** アイテムの入力側でルールが定義されています。**Rows** アイテムには、型 `string` と `integer` の子が存在します。ルールのデータ型は `string` であるため、ルールはこの型のアイテムのみ影響します。すなわち、 アイコンは型 `string` のアイテムのためにのみ存在しますが、型 `integer` のためには存在しません。 アイコンは、一部のアイテムのためにはルールがブロックされていることを意味し、このファクトは現時点では無効です。[デフォルトとノード関数のしくみ]に関する詳細は、を参照してください。

ヒント: 上のマッピング内で各アイテムの横にデータ型を表示するには、「**データ型の表示**」 ツールバーボタンをオンに切り替え、「**注釈の表示**」 ツールバーボタンをオフに切り替えます。「**表示**」メニュー内でこれらの切り替えコマンドを使用することができます。

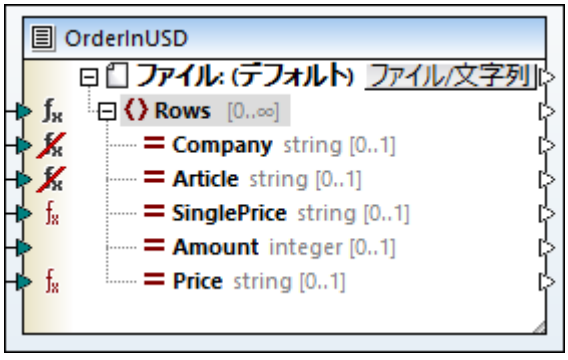
ルールのデータ型を変更するには、マッピングの上のグリッド内の「**省略記号**」ボタン をクリックします。これにより、必要とする型を選択することのできるダイアログボックスが開かれます。

(C) 2015-2021 Altova GmbH



「フィルター ノード関数をデフォルト」ダイアログボックス

例えば、データ型として `integer` を選択する場合、(型 `integer` の唯一のアイテムであるため) ルールはアイテム **Amount** のみに適用されます。このファクトを示すためにコンポーネントは再度描かれます。**Amount** 以外の他のアイテムは、型 `string` であるため ルールにより影響を受けていないことにご注意してください。



ツールのデータ型をノード関数内で意味が成す場合のみ変更します。デモ **OrderInUSD.mfd** マッピング内で、ノード関数は 2つ

の文字列を結合する) 数値ではなく、string データを処理するようにデザインされています。この結果、このマッピングはノード関数のデータ型を integer に変更すると、失敗します。

型の継承

MapForce はXML データだけではなく、データ型および「ノード関数とデフォルトをフィルター」ダイアログボックス上での選択のために使用できるデータ型とも作業することができ、XML スキーマ型に正確に対応する必要はありません。一部は型のカテゴリである場合もあり、これは、型の広範囲の選択に一致することを意味します。例えば、型 string は、 normalizedString、token、NCName、NMTOKEN、IDREF、ENTITY などから派生した string 多種の他のデータ型に一致します。同様に型 decimal は、派生した型 integer、long、short などに一致します。

型の階層構造はXML スキーマW 3C 勧告に従います。階層構造を説明するダイアグラムに関しては、 §3 in "XML スキーマパート 2: データ型第2エディション"、https://www.w3.org/TR/xmlschema-2 を参照してください。

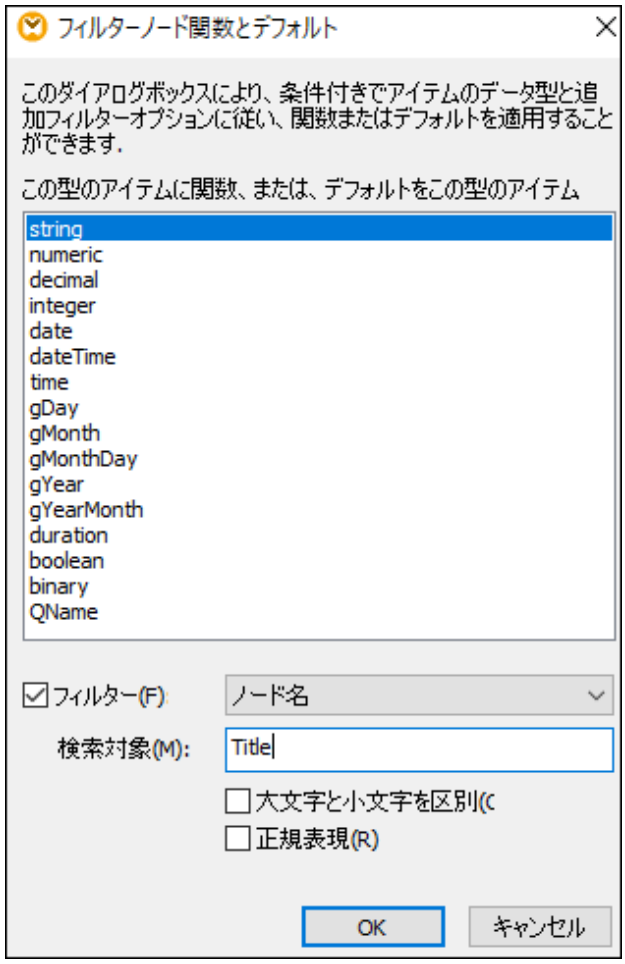
高度なフィルタリング

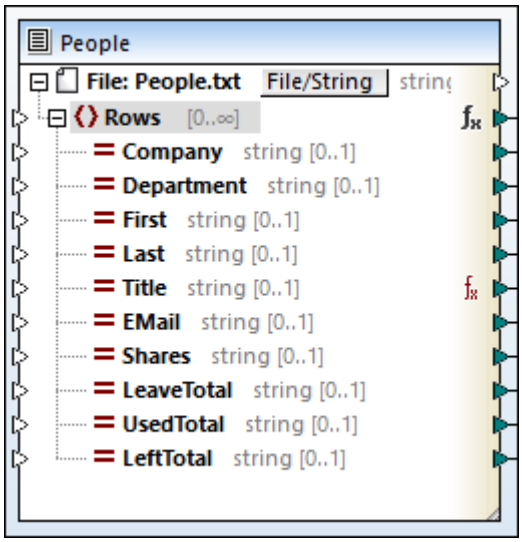
任意で、更に高度な条件をベースにしたルールを適用することもできます:

- 特定の型 AND であり、ノード名がカスタムのテキストの一部に一致する場合、ルールを適用します。
- 特定の型 AND であり、型名がカスタムのテキストの一部に一致する場合、ルールを適用します。

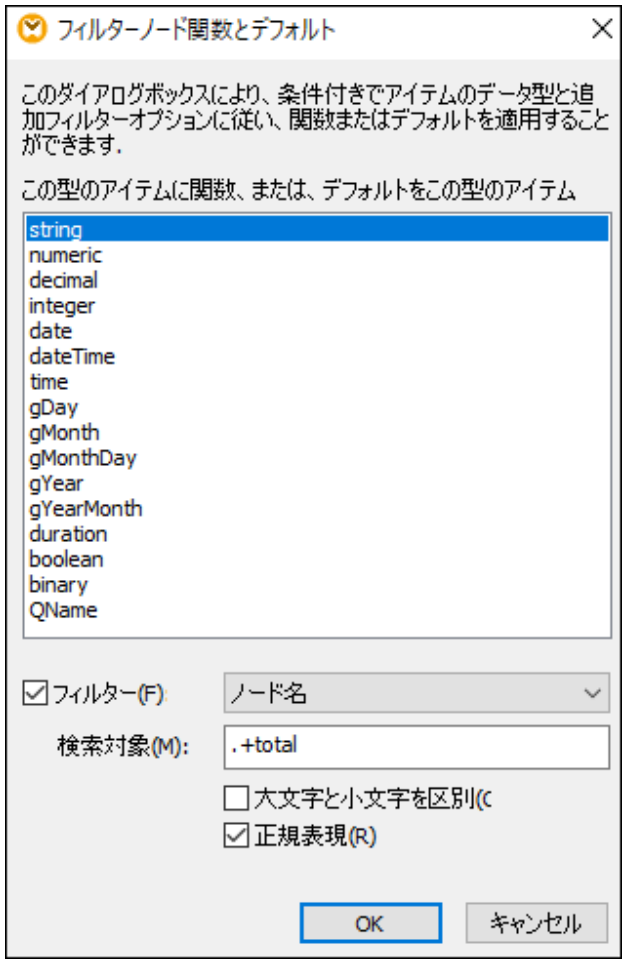
例えば、下の設定は、ノード型が string AND であり、ノード名が「Title」の場合のみルールが適用されます。右のイメージは、(ノード「Title」のみが一致し、このために fx アイコンが取得されるなど)このようなルールが役に立つ場合のサンプル構成を示しています:

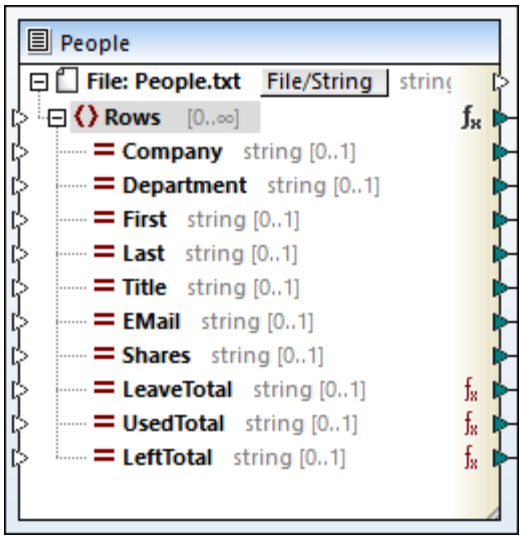
(C) 2015-2021 Altova GmbH





または、適するする複数のノード名または型名に一致する正規表現を使用することができます。例えば、下の設定は、名前が"total" で終わるすべての型 `string` アイテムに適用されます。正規表現が使用されていることご注意してください。正規表現 `.+total` 文字通り、以下を意味します: テキスト "total" に続く1回または複数回の文字の発生を一致させる。また、検索で大文字と小文字を区別することができ、"SOMETOTAL" と"SomeTotal" の双方が一致することを意味します。右のイメージは、(具体的には、3つの"Total" アイテムが同様にルールに適し fx アイコンを取得するなど)このようなルールが役に立つかもしれないサンプルの構成を示しています:





正規表現 大アレクト は、XML スキーマのものと同じです。正規表現 を参照してください。以下の点に注意してください:

- アンカー ^ と $ は暗黙であり、「検索対象」ボックス内に入力する必要はありません。
- 「大文字と小文字を区別」チェックボックスにより大文字と小文字は区別されます。ですから *i* フラグはサポートされません。
- ノードフィルタリングのために複数の行を一致させることは意味をなしません。ですから *m* フラグはサポートされません。

7.2.5 ノードメタデータをノード関数に提供する

現在のノードに関する情報をベースにして何かを行うために、ノード関数が必要な場合があります(「ノード メタデータ」とこの情報を呼ぶことにしましょう)。例えば、次のロジックを持つノード関数が必要と仮定します: ノード名に「Total」という単語が含まれている場合、ノードの値にドルのシンボルが追加されます。それ以外の場合、ノードの値がそのまま返されます。

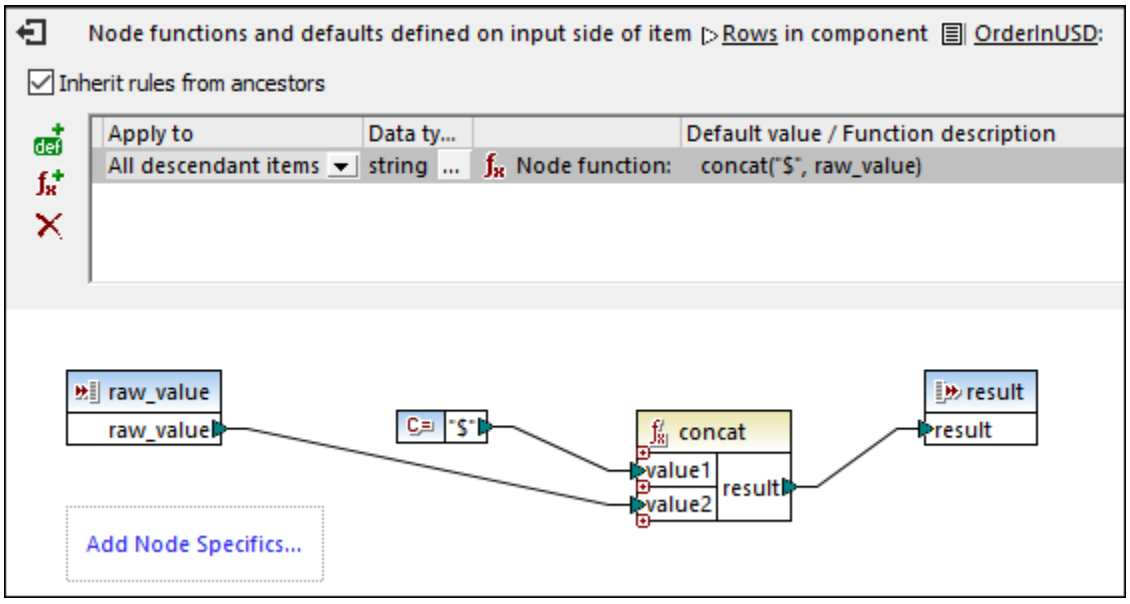
サンプルでも記述されていますが、「ノード名」はノードメタデータのサンプルです。一般的には、「メタデータ」はデータ自身を説明する何かを意味します。すなわち、これは「データに関するデータ」を意味します。「ノードメタデータ」により、数値のデータ型や他の場合、ノード名、値の長さ、または精度など関数が適用されるノードに関する他の情報を理解することができます。

次のテーブルは、ノード関数内で使用することのできるすべてのメタデータをリストしています。下にリストされるメタデータは、特定の種類のノードにだけのみ意味があります(例えば、XML、またはデータベースフィールド)。この結果、現在のノードと互換性の内メタデータを使用しようとすると、MapForce は警告を表示します。

node_name	現在のノードの名前を提供します。このメタデータをすべてのノードに適用することができます。XML の場合、現在の要素または属性の名前です。CSV の場合、これはCSV フィールドの名前です。データベースの場合、テーブル列の名前です。
node_annotation	「**注釈の表示**」 ツールバーボタンをクックすると、アイテムの横に表示される注釈テキストを提供します。このメタデータをすべてのノードに適用することができます。
node_minLength	ノードのデータ型の`minLength` ファセットの値を提供します。適切な型を持つXMLとテキストノードに適用することができます。
node_maxLength	ノードのデータ型の`maxLength` ファセットの値を提供します。適切な型を持つXMLとテキストノードに適用することができます。
node_totalDigits	ノードのデータ型の`totalDigits` ファセットの値を提供します。適切な型を持つXMLノードに適用することができます。
node_fractionDigits	ノードのデータ型の`fractionDigits` ファセットの値を提供します。適切な型を持つXMLノードに適用することができます。
node_length	ノードのデータ型の長さを提供します。適切な型を持つフィールドに適用することができます。
node_precision	ノードのデータ型の精度を提供します。適切な型を持つフィールドに適用することができます。
node_scale	ノードのデータ型のスケールを提供します。適切な型を持つフィールドに適用することができます。

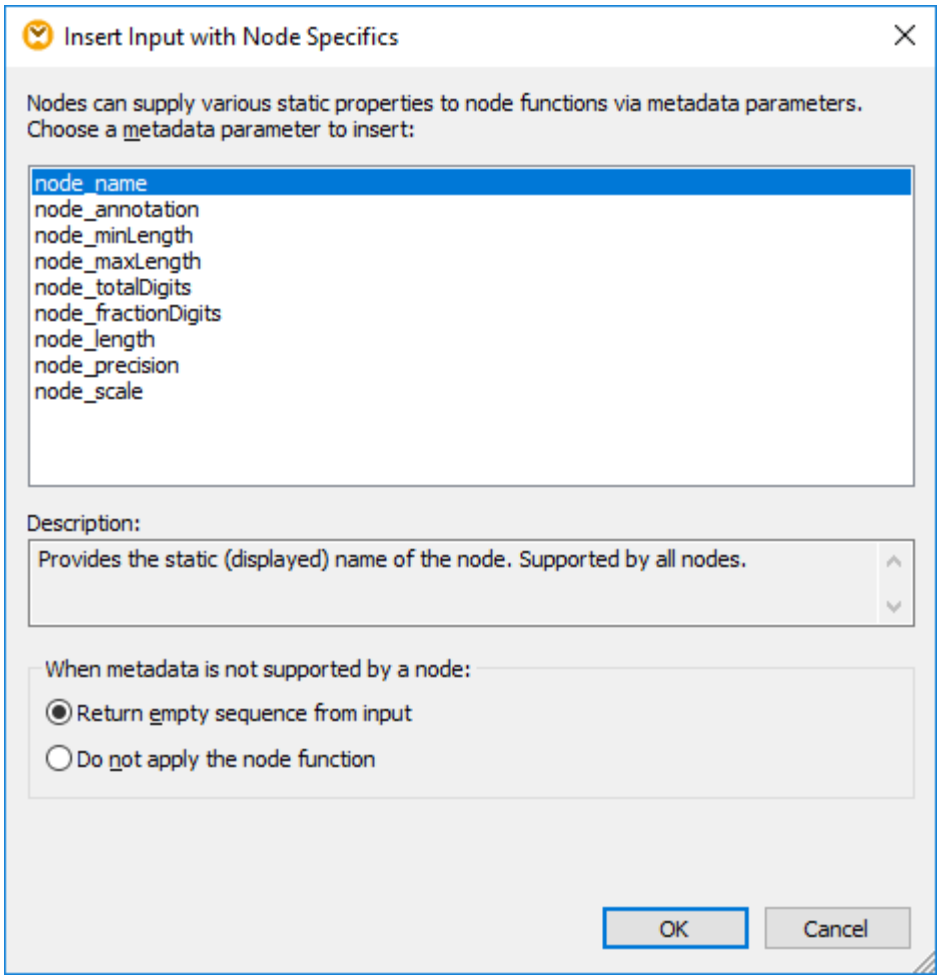
ノード関数にメタデータを提供する方法:

1. 新規のノードの作成を開始（デフォルトとノード関数の作成方法を参照）、または既存のノードを編集（既存のルールの編集と削除を参照）のために開きます。例えば、下で示されている関数は、ノード値を持つ文字列 "$" を連結して、外部のマッピングの結果を返します。この関数のマッピングを次のパスを使用して見つけることができます: **<マイドキュメント>\Altova\MapForce2021\MapForceExamples\OrderInUSD.mfd**。関数のマッピングを開くには、ターゲットコンポーネントの**Rows** アイテムの横の f_x アイコンをクリックし、グリッド上の Edit ボタンをクリックします。



2. 次のいずれかを実行してください:

   - 「**ノードの指定の追加**」をクリックします。
   - マッピング内の空のエリアを右クリックし、「**入力の挿入**」をコンテキストメニューから選択します。
   - 「**入力の挿入**」 ツールバーボタンをクリックします。
   - 「**関数**」メニューから、「**入力の挿入**」をクリックします。



3. ダイアログボックスから必要とされるメタデータ（例えば"static_node_name"）を選択します。

メモ 挿入するメタデータパラメータを選択すると、MapForce は、マッピング上のノード関数が既に適用する条件を満たしている個所で現在展開されているノードを分析し、メタデータパラメータがこれらのノードによりサポートされるかを決定します。サポートされていない場合、ダイアログボックスは「選択されたメタデータパラメータはこの関数のスコープ内の既存のノードによりサポートされていません。」に類似した警告を表示します。メモリを保存しユーザーエクスペリエンスを構造するために デフォルト では、深くネストされた構造はフルにスキャンされません。ノード関数を適用する箇所でコンポーネントがこのような深くネストされた構造を持つ場合、MapForce が認識できるようにマッピング上で関連するノードを展開することができます。この場合、新規のメタデータパラメータを追加する際に、MapForce は展開されたノードを考慮し、警告が表示される可能性があります。ノード関数を適用するには接続が存在する必要がありますが、接続されていないアイテムの展開は関連性がありません。

4. メタデータが現在のノードにサポートされていない場合、関数の振る舞いを次のように決定することができます:
   a. ノード関数を適用する場合、チェックボックス「**入力から空のシーケンスを返す**」を選択し、メタデータパラメータに空のシーケンスを返すようにすることができます。空のシーケンスは空の文字列と区別される必要があります。[substitute-missing]

または **exists** などのシーケンス関数、または他のコンポーネントを処理するために使用する必要が通常あります。警告: 空のシーケンスが扱われる必要があります。それ以外の場合、ノード関数は値を全く返さない場合があります。

b. このメタデータがノードにサポートされなくノード関数を適用しない場合、チェックボックス「**ノード関数を適用しない**」を選択します。

5. 「OK」をクリックします。デフォルトの**raw_value** に加え、関数のマッピングに新規の入力パラメータが追加されました。新規のパラメータの出力のコネクタをこのメタデータ通常は(関数の入力コネクタ)を必要とするターゲットアイテムに接続します。

7.2.6 例: 空の CSV フィールドの置換

このサンプルは、CSV ファイルからデータを読み取り他のCSV ファイルにデータを書き込むMapForce マッピングの作成方法を説明しています。カスタムの値(″n/a″) を持つソースCSV ファイルからすべての空のフィールドを置き換えることがこのサンプルの目的です。ソース CSV ファイルは以下のようになると仮定します:

```
H,111,332.1,22537.7,,Container ship,,,
D,111,A-1579-227,10,3,400,Microtome,,
D,111,B-152-427,7,6,1200,Miscellaneous,,
H,222,978.4,7563.1,,Air freight,,,
D,222,ZZ-AW56-1,10,5,10000,Gas Chromatograph,,
```

希望するマッピングの出力は以下のようになります:

```
H,111,332.1,22537.7,n/a,Container ship,n/a,n/a,n/a
D,111,A-1579-227,10,3,400,Microtome,n/a,n/a
D,111,B-152-427,7,6,1200,Miscellaneous,n/a,n/a
H,222,978.4,7563.1,n/a,Air freight,n/a,n/a,n/a
D,222,ZZ-AW56-1,10,5,10000,Gas Chromatograph,n/a,n/a
```

次のパスでこのサンプル内で作成されたマッピングを見つけることができます: **<マイドキュメント >\Altova\MapForce2021\MapForceExamples\Tutorial\ReplaceEmptyFields.mfd**。マッピングのためのソースCSV ファイルは、**Orders.csv** と呼ばれ、同じファイル内にあります。ターゲット CSV ファイルはMapForce により生成されます。

マッピングの目的を達成するために、存在するそれぞれの空の値を″n/a″ と置き換える単一のノード関数を作成します。下に示されるように、この関数は1 度のみ定義されますが複数の子孫 CSV フィールドに適用されます。

ステップ 1: ソース CSV ファイルをマッピングへ追加する

ソースCSV ファイルをマッピングに以下のように追加することができます:

1. 「**挿入**」メニューから、「**テキストファイル**」をクリックします。
2. (MapForce Enterprise Edition のみ) オプション「**標準 CSV(カンマ区切り)またはFLF(固定長)のファイルに単純な処理を使用 フィールド**」を選択して、「**継続**」をクリックします。
3. 「**入力 ファイル**」をクリックして、次のファイルを選択します: **<マイドキュメント >\Altova\MapForce2021\MapForceExamples\Tutorial\Orders.csv**。

   チェックボックス「**空のフィールドを存在しないものとして扱う**」が選択されている場合、クリアしてください。チェックされている場合、このチェックボックスは、空の値を抑圧し、ノード関数が作動しません。詳細に関しては、次を参照してください: CSV オプションの設定.

4. 「OK」をクリックします。
5. ″Orders″ に対するコンポーネントを変更するようにプロンプトされると、希望するオプションをクリックします(例えば、**コンポーネント名を変更しない**)。

MapForce のCSV コンポーネント詳細に関しては次を参照してください: CSV とテキストファイル.

ステップ 2: ターゲット CSV ファイルをマッピングへ追加する

ターゲット CSV ファイルをマッピングに以下のように追加することができます:

1. 「入力」メニューから、「テキストファイル」をクリックします。
2. (MapForce Enterprise Edition のみ) オプション「標準 CSV(カンマ区切り)またはFLF(固定長)のファイルに単純な処理を使用 フィールド」を選択して、「継続」をクリックします。
3. ターゲット ファイルはソースと同じ数のフィールドを有する必要があります。このため、9つのフィールドを追加するために「フィールドの追加」ボタンを複数回クリックしてください。
4. 「OK」をクリックします。

ステップ 3: マッピング 接続の描写

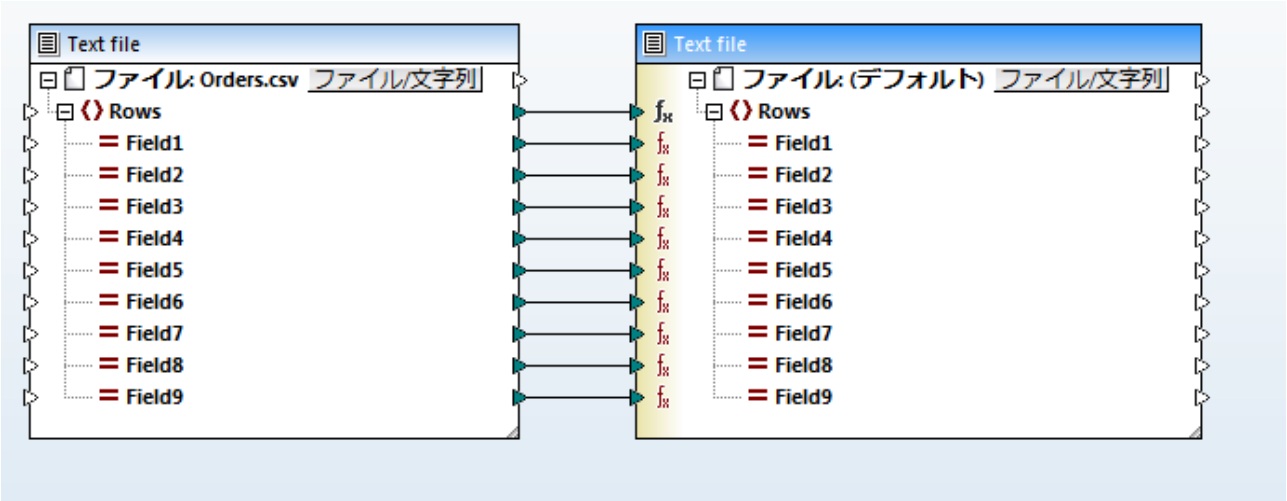
この時点では、マッピングには次の2つのコンポーネントが含まれています: ソースCSV ファイルとターゲット。ソースコンポーネント上の **Rows** アイテムの横の出力コネクタ ▷ クリックし、ターゲットコンポーネント上の**Rows** アイテムの入力コネクタ ▷ にドラッグします。これを行うと、設定に応じてMapForce は自動的に全ての子孫アイテムを接続し、いわゆる、"全てコピー" 接続を作成する可能性があります。

一致する子要素を自動接続する ツールバーオプションが有効化されている場合のみ上記が行われます。前記のとおり、ノード関数は"全てコピー" 接続の子孫に適用されません。このため、"全てコピー" 接続は最初に通常の接続に変更される必要があります。これを行うには、ソースの**Field1** から、ターゲットの**Field1** に接続します。プロンプトされると、**接続の置換**をクリックし、**全てコピー接続を解決する**をクリックします。

一致する子要素を自動接続する オプションが有効化されていない場合、ソースとターゲット出力間の接続を以下のように作成します:

1. ソース内の**Rows** アイテムをターゲット内の**Rows** アイテムと接続します。
2. 接続を右クリックし、**一致する子要素を接続する** をコンテキストメニューから選択します。
3. **全てコピー接続を作成する** チェックボックスをクリアします。
4. 「OK」をクリックします。

マッピングは、以下のようになります:

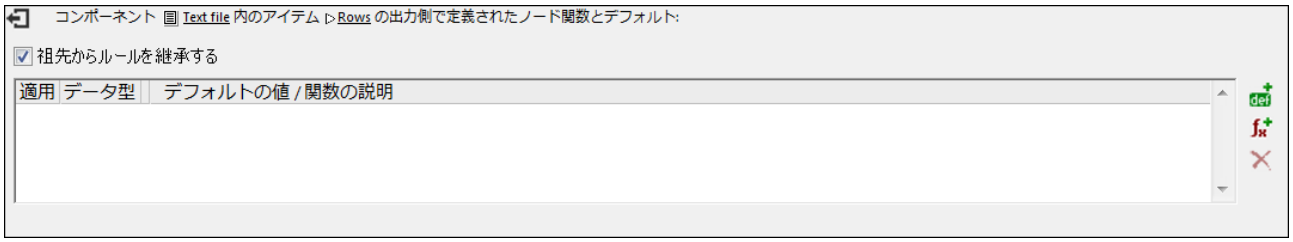


(C) 2015-2021 Altova GmbH

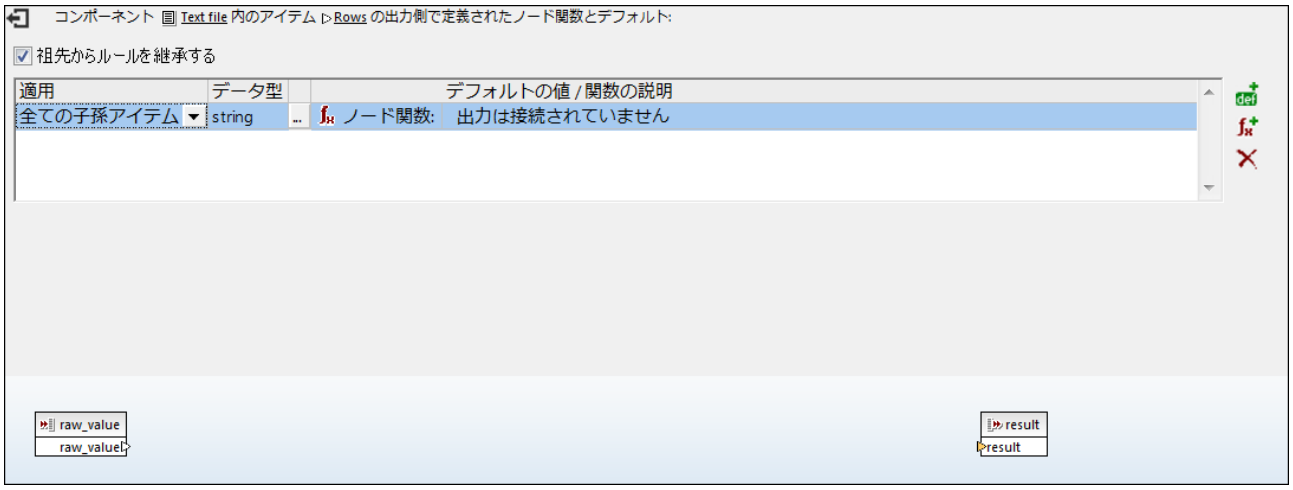
ステップ 4: ノード関数の作成

データがソースを離れてすぐに、またはターゲットに書き込まれる前に、ノード関数を作成することができます。このサンプルの目的のため、ターゲットコンポーネントの入力側でノード関数を作成します。これは、基本的には、「ターゲットにデータが書き込まれる前」を意味します。詳細に関しては、次を参照してください: 入力と出力サイドの選択.

ターゲットコンポーネント上のRows アイテムを右クリックし、「**デフォルトとノード関数 | 入力デフォルトとノード関数**」をコンテキストメニューから選択します。「**マッピング**」ペインの上に空のグリッドが表示されます。



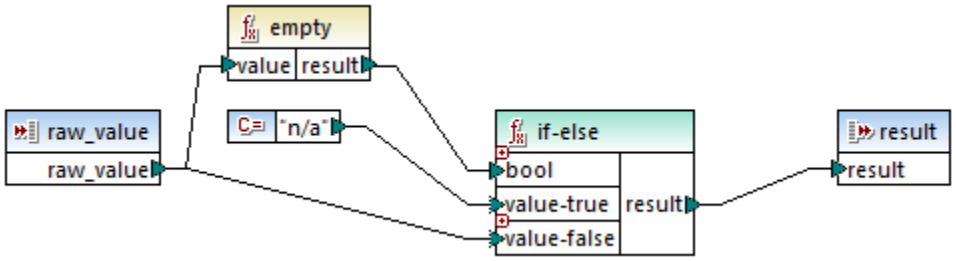
次に、グリッドの左の**関数を追加** ボタンをクリックします。マッピングは関数の入力（"raw_ value"）出力（"result"）を表示します。



前記のとおり、関数の目的は、空の値を文字列 "n/a" に置き換えることです。これを達成するために、マッピングに追加コンポーネントを追加します:

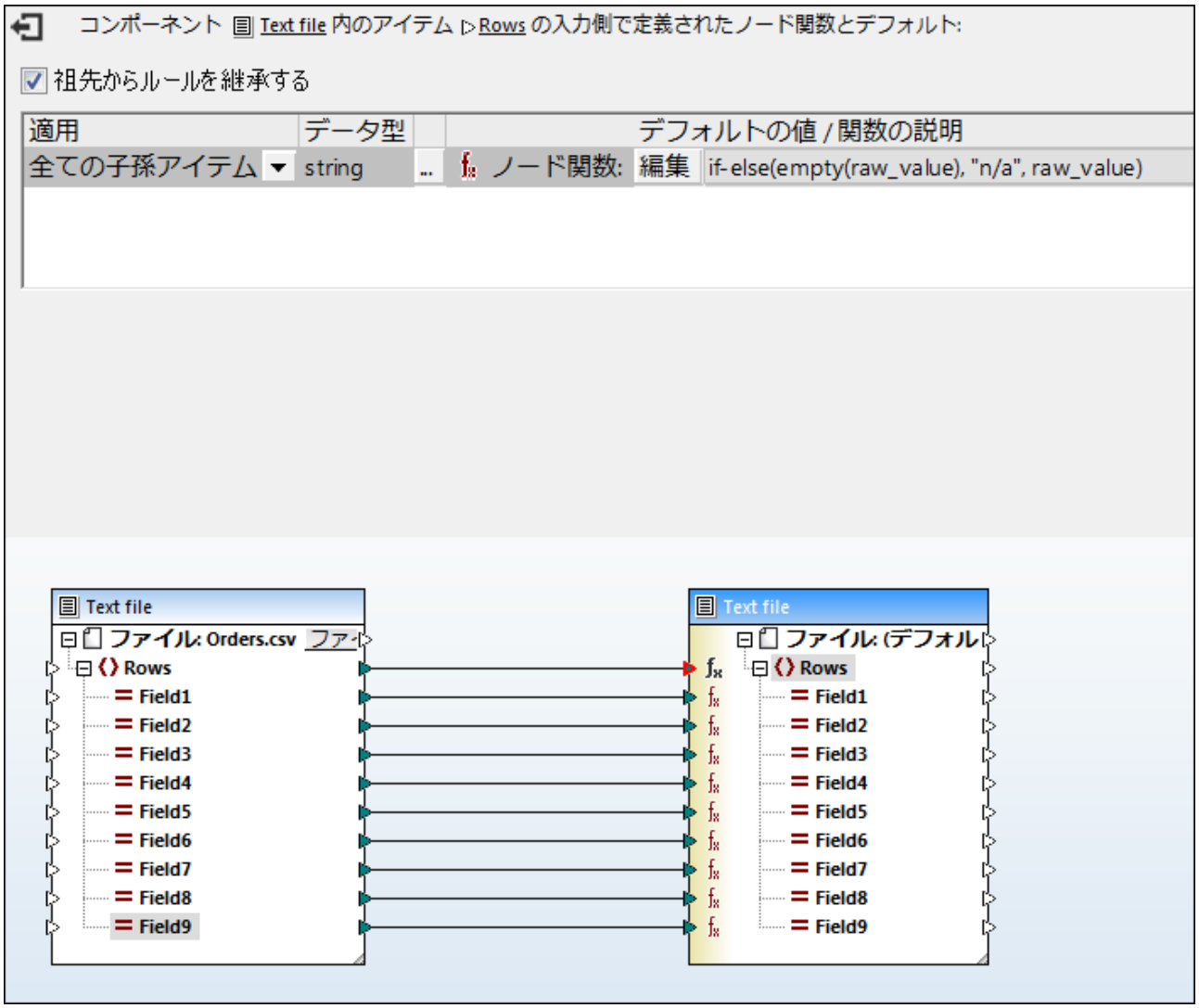
- MapForce ビルトイン関数 **empty**。引数として提供された値が空の場合、この関数はtrue を返します。それ以外の場合はfalse を返します。ライブラリウィンドウから関数をマッピングにドラッグ、またはマッピングをダブルクリックして、"empty" を入力します。 マッピングに関数を追加するも参照してください。
- テキストの値 "n/a"。値を追加するには、マッピング上の空のエリアをダブルクリックし、2重引用符に囲まれた "n/a" を入力します。次も参照してください: マッピングに定数を追加する。
- **If-Else 条件**。マッピングに追加するには、**If-Else 条件**（ ）ツールバーボタンをクリックします。このようなコンポーネントの詳細に関しては、次を参照してください: 例: 条件付で値を返す。

これらのコンポーネントを使用して、関数をデザインをします:



上記のデザインは、以下の用に作動します: 最初に、外側のマッピングからの入力値は、**raw_value** 入力を使用して関数を入力します。行の値は、`empty` 関数に入力として提供されます。**If-Else コンポーネント** は`empty` 関数により返されたブール関数の結果を評価します（**true** または**false**）。結果が**true** の場合、定数 "n/a" は関数の結果になります。結果が**false** の場合、関数の行入力の値は、関数の結果になります。関数の**result**（"n/a" または**raw_value**）が外側のマッピングから返されます。

Exit をクリックして（または、**Escape** を押し）関数の編集のエリアを終了します。



(C) 2015-2021 Altova GmbH

上記のマッピング内で、以下の点に注意してください:

- ウィンドウの上の部分のテキストは関数が定義された箇所を示します。これは同じコンポーネントのために複数のノード関数が定義されている状況では特に役にたちます。
- グリッド内の**適用** オプションは **全ての子孫** アイテムに設定されています。このサンプルでは、これは期待される振る舞いです。すなわち、**Rows** の全ての子孫アイテムが条件を満たす場合影響を受けます。関数は親アイテムに1度のみ定義されていますが、マッピング上で確認できるように、ターゲットコンポーネントの左側(入力)は複数の アイコンを表示します。
- **データ型** オプションは"string"に設定されています。このサンプルでは、テキストデータを作業しているため、これは期待される振る舞いでありまた、デフォルトの振る舞いでもあります。
- **編集** ボタンにより、必要に応じて関数の定義に戻り変更することもできます。このボタンが表示されていない場合、 アイコンを最初にクリックしてください。

ステップ 5: マッピングの実行

MapForce 内でマッピングの結果をプレビューするには、**出力** タブをクリックします。検証エラーが発生すると、**メッセージ** ウィンドウ内に表示されます。成功するとCSV 結果が**出力**ペイン内に表示されます。次を参照してください: マッピングの検証。

MapForce Server Advanced Edition を使用して、次の方法でサーバーマシン上でこのようなマッピングを実行することもできます:

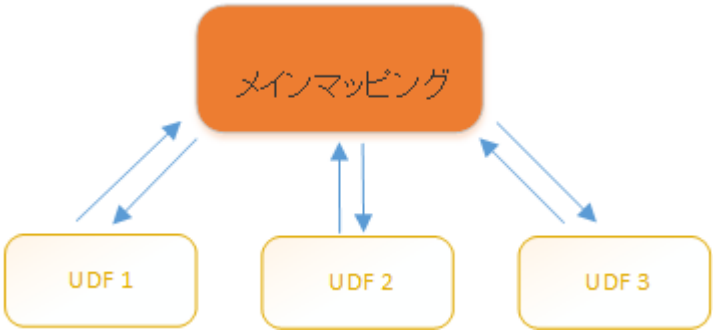
- MapForce Server Advanced Edition スタンドアロンを使用する場合、マッピングを実行可能ファイルにコンパイルして、ターゲットマシンにコピーしてください。次を参照してください: MapForce Server 実行可能ファイルにマッピングをコンパイルする
- FlowForce Server とMapForce Server Advanced Edition にライセンスが供与されている場合、FlowForce Server に直接マッピングをデプロイし、予定通り、またはオンデマンドとしてジョブが作動するように構成します。次を参照してください: マッピングを FlowForce Server にデプロイする。

7.3 ユーザー定義関数

ユーザー定義関数（UDF）は一度定義されるカスタム関数で、同じマッピング、または複数のマッピング内で複数回使用することができます。ユーザー定義関数自身は小さなマッピングと類似しています。通常、一つまたは複数の入力パラメーター、データを処理するための中間パラメーター、および、と呼び出し元にデータを返すための出力により構成されています。呼び出し元はメインマッピング、または他のユーザー定義関数であることができます。

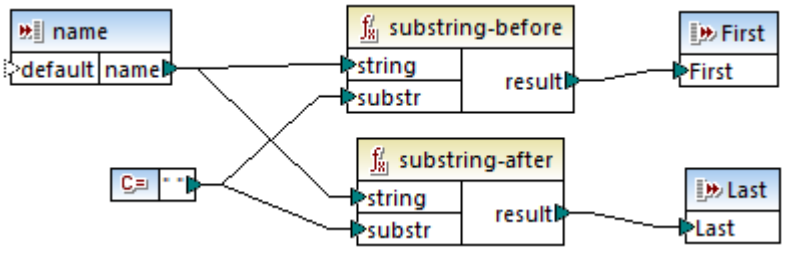
ヒント: 複数の出力を返すユーザー定義関数を作成することは可能です。関数が「インライン」と定義されているとこの機能はサポートされます。インラインと正規ユーザー定義関数を参照してください。

再使用の可能性とは別に、マッピングの部分を小さいコンポーネントにパッケージ、および、実装の詳細を無視し、結果、マッピングを簡単に理解するためにユーザー定義関数は役に立ちます。



通常、文字列、数値、日付、および、他のデータを内蔵のMapForce 関数を拡張しカスタム化して処理するためにユーザー定義関数を作成します。例えば、特定の方法で結合、または分割する、または高度な計算を実行、時刻を操作、再利用できるようにマッピングの一部をパッケージするなどの場合が挙げられます。ユーザー定義関数の他の一般的な使用方法は、MapForce によりサポートされる他のファイル、データベース、またはデータストア内のフィールドを検索することです。

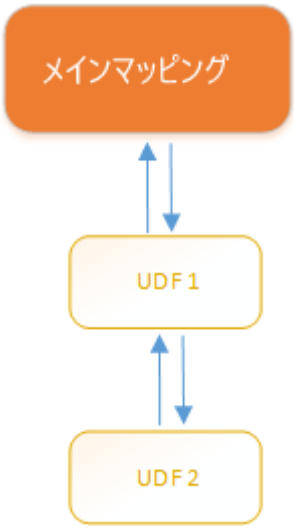
以下のマッピングは、文字列を2つの異なる文字列に分割する、ユーザー定義関数のサンプルです。このユーザー定義関数は次のデモマッピングの一部です: <マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥**ContactsFromPO.mfd**。パラメーター（例えば、"Helen Smith"）としての名前を取り、
ビルトイン関数 `substring-before` と `substring-after` を適用し、2つの結果する値（"Helen" と "Smith"）を返します。



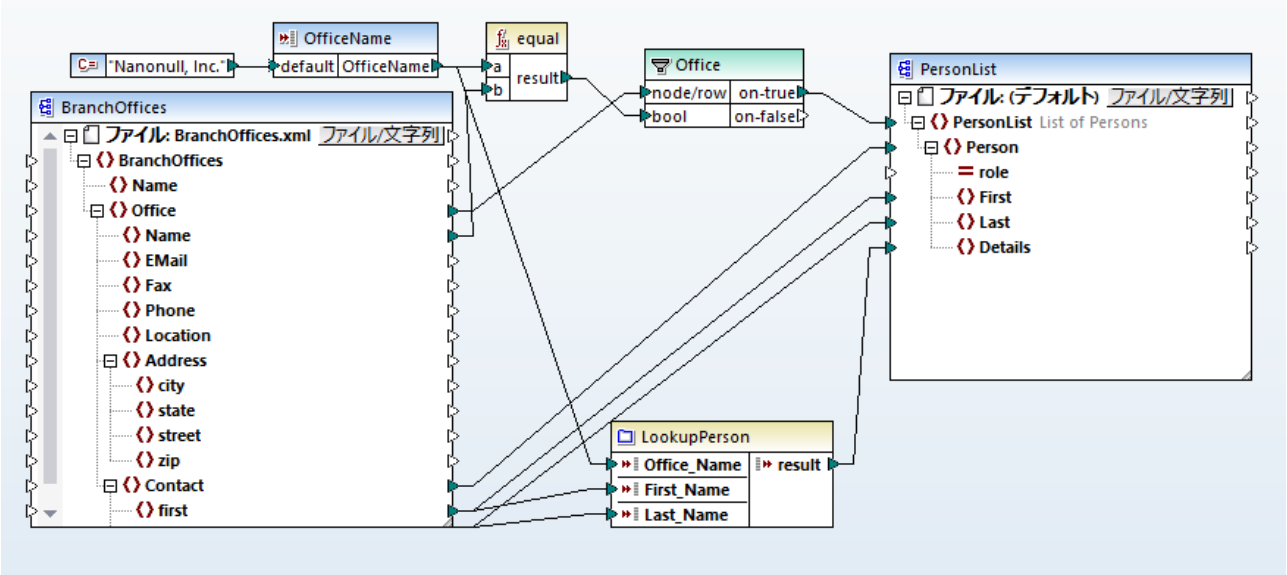
ContactsFromPO.mfd

上記のとおり、ユーザー定義関数をメインのマッピング、またはユーザー定義関数から呼び出すことができます。すなわち、ユーザー定義関数を必要であれば、以下に示されるように、ネストすることができます。

(C) 2015-2021 Altova GmbH

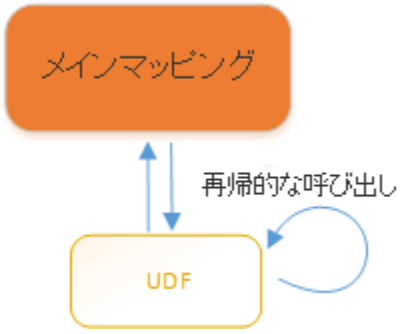


例えば、下のマッピングはユーザー定義関数（「LookupPerson」）を呼び出し、XML ファイル内の個人の名前を検索します。「LookupPerson」コンポーネントのヘッダーをダブルクリックすると、マッピングウィンドウ内の定義が開かれ、この関数は次のユーザー定義関数を呼び出していることに気が付くはずです: "EqualAnd" と "Person2Details"。このマッピングはデモとして使用することができ、次のパスで見つけることができます: **<マイドキュメント>\Altova\MapForce2021\MapForceExamples\PersonListByBranchOffice.mfd**.



PersonListByBranchOffice.mfd

ユーザー定義関数を再帰的に呼び出すことができます（すなわち、ユーザー定義関数が自身を呼び出します）。これは、ユーザー定義関数を（インラインではなく）関数正規関数として定義する必要があることを意味します。インラインと正規ユーザー定義関数を参照してください。



再帰的なユーザー定義関数により、*N* が事前に既知ではなく、*N* 個の子の深さを持つデータの構想を反復するなど、多種の高度なマッピングの必要条件を解決することができます。例: 再帰的な検索を参照してください。

ユーザー定義関数の作成後、作成した同じマッピング内に保存されます。しかしながら、他のマッピングにインポートし、そこから呼び出すこともできます。詳細に関しては、ユーザー定義関数の呼び出しとインポートを参照してください。

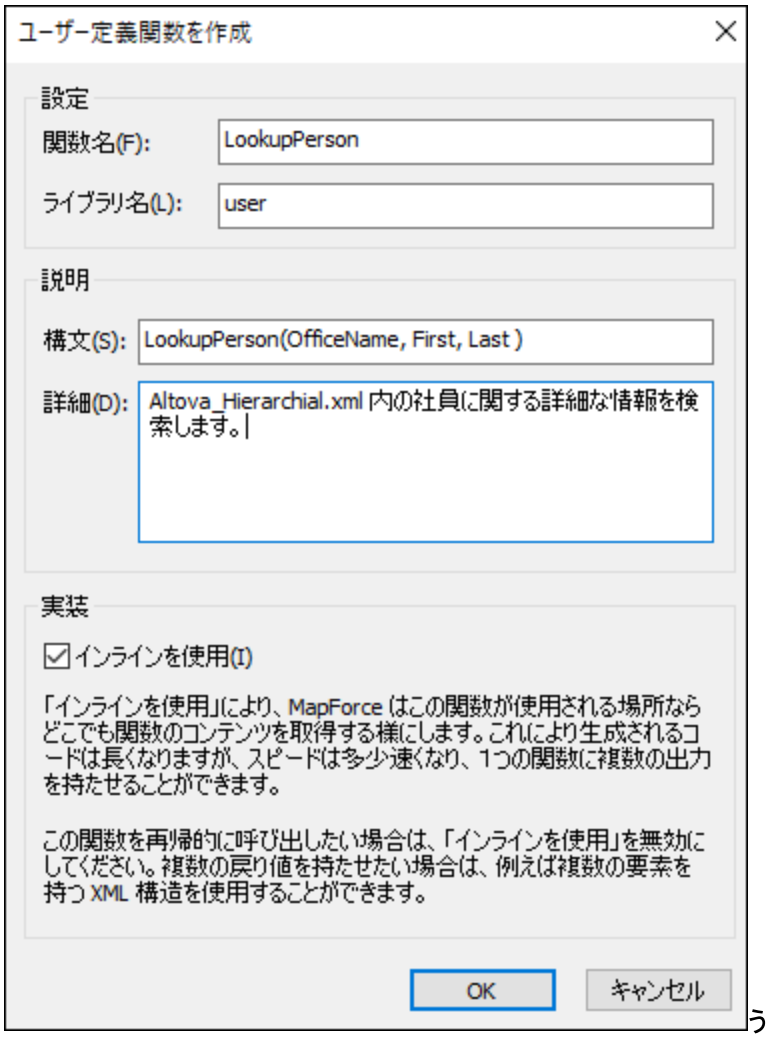
7.3.1 ユーザー定義関数の作成

ユーザー定義関数の作成を始めから作成、またはマッピング上に既存のコンポーネントの選択から作成することができます。

ユーザー定義関数を最初から作成する方法:

1. 「関数」メニューから、「**ユーザー定義関数の作成**」をクリックします。または、代わりに、「**ユーザー定義関数の作成**」( ) ツールバーボタンをクリックします。

(C) 2015-2021 Altova GmbH



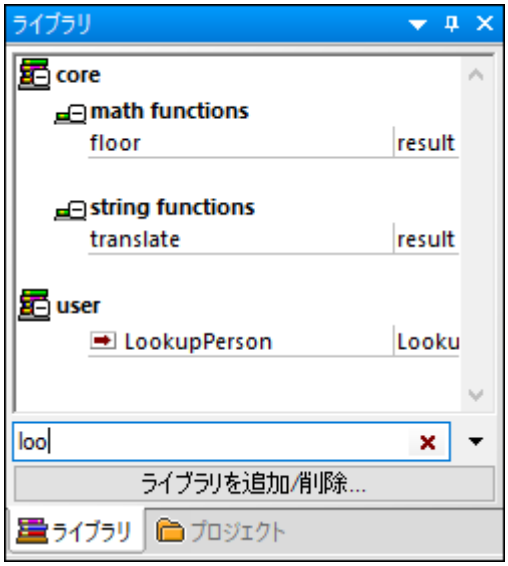
う

2. 必要なフィールドに情報を挿入します（下のテーブルを参照してください）。

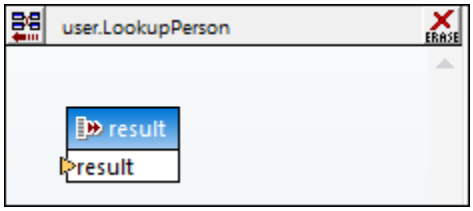
関数名	必須のフィールド。作成するユーザー定義関数のために名前を入力します。有効な文字: • 英数字文字（a-z、A-Z、0-9） • 下線（ _ ） • ハイフン/点線（ - ） • コロン（ : ）
ライブラリ名	必須のフィールド。関数が属するべきライブラリ名を入力します。関数はライブラリウィンドウ内でこのライブラリ名で表示されます。ライブラリを指定しない場合、関数は "user" という名前のデフォルトのライブラリ内に配置されます。
構文	任意のフィールド。関数の構文を的確に詳細するテキストを入力します。（例えば、期待されるパラメーター）。このテキストは、ライブラリウィンドウ内の関数の横に表示され、関数の実行には影響を与えません。

詳細	任意のフィールド。関数の任意のテキストの詳細を入力してください。このテキストは、ライブラリウィンドウ内で、または他のコンテキスト内でカーソルを動かすと表示されます。
インラインされた使用	関数がインライン関数として作成される場合、このチェックボックスを選択してください。正規関数を作成する場合は、チェックを解除してください。詳細に関しては、インラインと正規ユーザー定義関数を参照してください。

3. 「OK」をクリックします。上記で指定されているライブラリ名の下のライブラリウィンドウ内で関数がすぐに表示されます。例:



また、マッピングウィンドウが再度描き直され、新規の関数を作成することができます(これは"関数のマッピング"として参照されるスタンドアロンマッピングです)。関数は出力を必要とするため、関数のマッピングにはデフォルトで出力コンポーネントが含まれています。



左上の角の「**メインマッピングに戻る**」( ) ボタンにより、関数のマッピングからメインマッピングに戻ることができます。関数のマッピングを開くためには、ライブラリウィンドウ内の関数をダブルクリックします。詳細に関しては、ユーザー定義関数の呼び出しとインポートとユーザー定義関数のナビゲートを参照してください。

4. 関数の定義により必要とされるすべてのコンポーネントを関数のマッピングに追加します。標準のマッピングと同じ方法でこれを行うことができます。例えば、入力または出力パラメーターを追加するには、次のいずれかを実行してください:

- メニューコマンド「**関数 | 入力の挿入**」、または「**関数 | 出力の挿入**」をそれぞれ実行します。
- マッピングエリアを右クリックし、「**入力の挿入**」または「**出力の挿入**」をコンテキストメニューから選択します。
- 「**入力の挿入**」( ) または「**出力の挿入**」( ) ツールバーボタンをクリックします。

少なくとも、関数はデータが接続される1 つの出力コンポーネントを必要とします。入力パラメーターに関しては、関数は、ゼロ、または1 つ以上の入力を持つことができます。入力または出力パラメーターは、単純型（文字列、または整数など）、または複合型（構成）であることができます。単純型、および、複合型のパラメーターに関する詳細は、ユーザー定義関数内のパラメーターを参照してください。

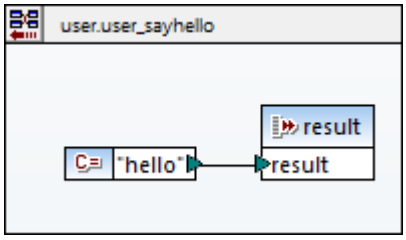
ユーザー定義関数が作成されましたが、まだ使用されていません。関数をマッピング内で使用するには、ライブラリウィンドウから関数をメインマッピング エリアにドラッグします。 ユーザー定義関数の呼び出しとインポートを参照してください。

ユーザー定義関数を既存のコンポーネントから作成する方法:

1. マウスを使用して正方形を作成し、マッピング上で複数のコンポーネントを選択します。「Ctrl」 キーを押したまま、それぞれをクリックして複数のコンポーネントを選択することができます。
2. 「関数」メニューから、「選択からユーザー定義関数を作成する」をクリックします。代わりに、「選択からユーザー定義関数を作成する」（ ）ツールバーボタンをクリックします。
3. 上記のステップ 2-4 を繰り返します。

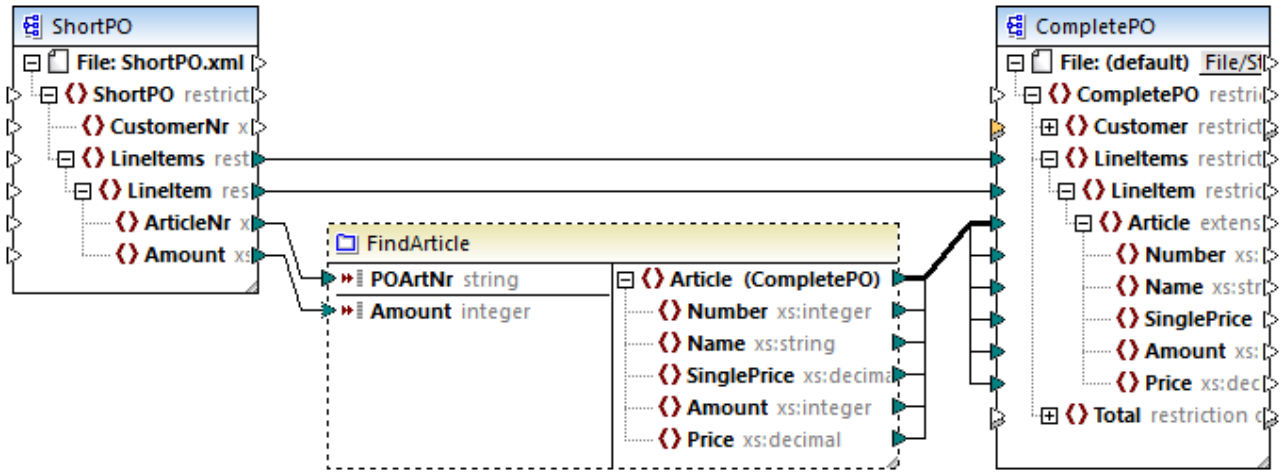
7.3.2 ユーザー定義関数内のパラメーター

ユーザー定義関数を作成する場合、（必要な場合）どの入力パラメーターが取られるか、および、出力が返されるかを指定する必要があります。入力パラメーターが必要とされない場合がありますが、出力パラメーターはすべての場合に必要とされます（すなわち、関数は常に何かを返す必要があります）。例えば、下の関数には入力は存在しませんが、呼出元にテキスト「hello」を返す1 つの出力が存在します:



関数パラメーターは、単純型（文字列、または整数などの）または複合型の構成です。例えば、下に表示されるユーザー定義関数 "FindArticle" では、2 つの入力と1 つの出力パラメーターが存在します。

- **POArtNr** は単純型 "string" の入力パラメーターです。
- **Amount** は単純型 "integer" の入力パラメーターです。
- **CompletePO** は複合型 XML の出力パラメーターです。



LookupArticle.mfd

この上のマッピングはデモとして次のパスで見つけることができます: **<マイドキュメント>\Altova\MapForce2021\MapForceExamples\LookupArticle.mfd**。

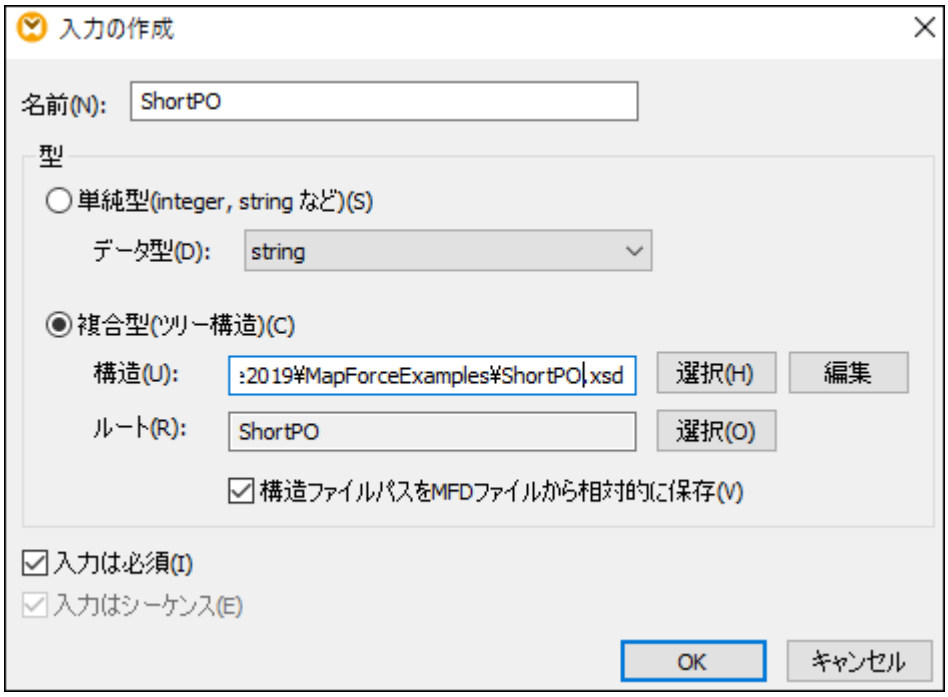
パラメーターの追加

入力または出力パラメーターを追加する方法:

1. ユーザー定義関数 マッピングを作成、または既存のものを開きます（ユーザー定義関数の作成を参照してください。）または、（ユーザー定義関数の編集を参照してください）。
2. 次のいずれかを実行してください:

   - メニューコマンド「**関数 | 入力の挿入**」または「**関数 | 出力の挿入**」を実行します。
   - **入力の挿入** または **出力の挿入** ツールバーボタンをクリックします。

(C) 2015-2021 Altova GmbH



3. 上のダイアログボックスで、入力または出力パラメーターが(文字列、または整数などの) 単純型 (XML 構成などの) 複合型の構成であるかを選択してください。複合型のXML 型であるパラメーターを作成するためには、「構造」の横の「選択」をクリックし、必要とされた構成を説明するXML スキーマを参照してください。

関数のマッピングはXML スキーマを既に含む場合、構成として選択のために使用することができます。それ以外の場合、パラメーターの構成を適用する新規のスキーマを選択することができます。 MapForce のエディションによりサポートされる場合、データベース、または他の複合型の構成も同様に扱われます。

XML 構成を使用すると、XML スキーマが許可する場合、構成のためにルート要素を選択することができます。ルート要素を指定するためには、「Root」の横の「選択」し、開かれるダイアログボックスからルート要素を選択します。

選択された場合、チェックボックス「MFD ファイルから相対的に構成ファイルパスを保存」はファイルを保存する際、構成ファイルの絶対パスを現在のマッピングに対して相対的なパスに変更します。詳細に関しては、コンポーネント上で相対パスを使用するを参照してください。

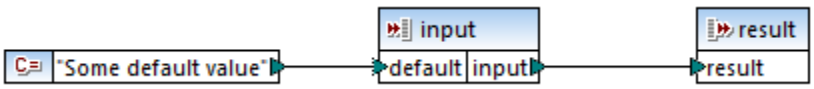
「入力は必須」と「入力はシーケンス」チェックボックスは次のセクションで説明されています。

必須のパラメーター

ユーザー定義関数内でパラメーターを必須にするには、「入力は必須」チェックボックスを選択します。パラメーターが必須の場合、入力が接続されていない場合検証エラーが発生します。

パラメーターを任意にするには、「入力は必須」チェックボックスのチェックを解除します。メインマッピングでは、任意のパラメーターには若干異なる外観(入力コネクタ(小さな三角形) の点線の境界線で表示)が存在します。

パラメーターの「default」入力に接続することにより、デフォルトのパラメーターの値を指定する必要があります。例:

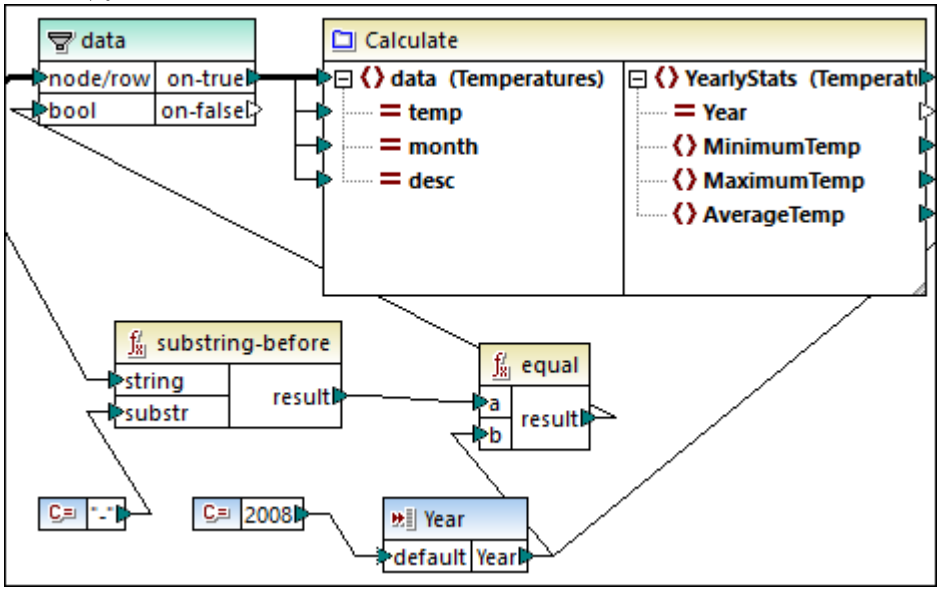


デフォルトの値はその他の値が存在しない場合のみ適用されます。関数が呼び出される際に、任意のパラメーターが値を受け取ると、この値はデフォルトの値に対し優先されます。

シーケンスパラメーター

関数のパラメーターが単一の値（デフォルトの振る舞い）、またはシーケンスとして扱われるように任意で指定することができます。パラメータの入力値を単一の値と異なり、シーケンスとして扱うには、「**入力はシーケンス**」チェックボックスを選択します。このチェックボックスは役に立ち、ユーザー定義関数が型「正規」に設定されている場合のみ有効化されます。[インラインと正規ユーザー定義関数](#)を参照してください。それ以外の場合、チェックボックスは無効化されます。

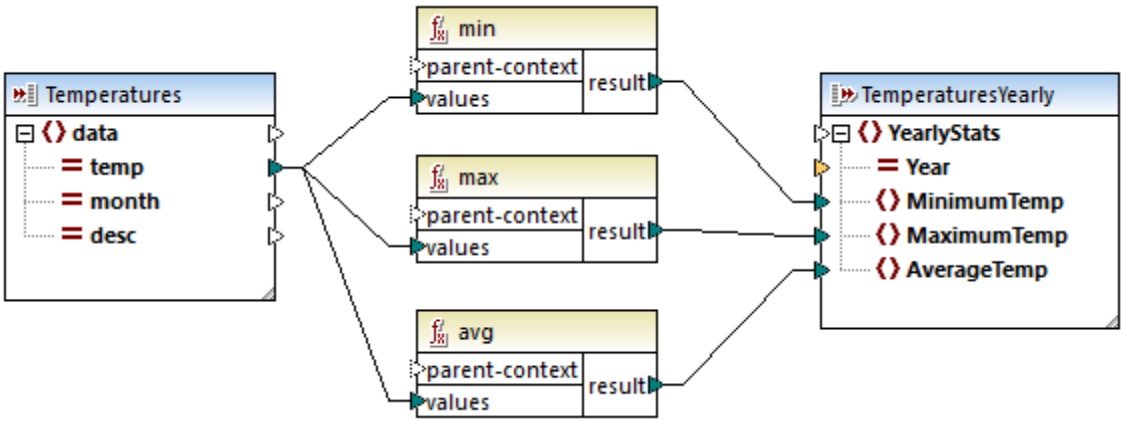
シーケンスはゼロ、またはゼロ以上の値の範囲です。ユーザー定義関数が入力データをシーケンスとして期待する場合、そのシーケンス内で値の集計を行うには、パラメーターの入力値をシーケンスとして扱う必要がある場合があります（例えば、`avg`, `min`, `max` などの関数を呼び出す場合などが挙げられます）。例えば、デモマッピングを開いてください：**<マイドキュメント>\Altova\MapForce2021\MapForceExamples\InputIsSequence.mfd**。このマッピングでは、「data」フィルターはユーザー定義関数「Calculate」に接続されています。フィルターの出力は、アイテムのシーケンスで、関数の入力パラメーターはシーケンスに設定されています。



InputIsSequence.mfd

内部では、「Calculate」関数はすべてのシーケンス値が集計されます（下で示されるように`min`、`max` と`avg` を実行し入力シーケンス上の関数を集計します）。

(C) 2015-2021 Altova GmbH



一般的には、シーケンス、または非シーケンスに関わらず、入力データはどのように関数が呼び出されるかを決定します。

- **sequence** パラメーターに入力データが接続されている場合、ユーザー定義関数1度のみ呼び出され、完全なシーケンスがユーザー定義関数内にパスされます。
- **non-sequence** パラメーターに入力データが接続されている場合、ユーザー定義関数はシーケンス内の各単数のアイテムのためにそれぞれ呼び出されます。

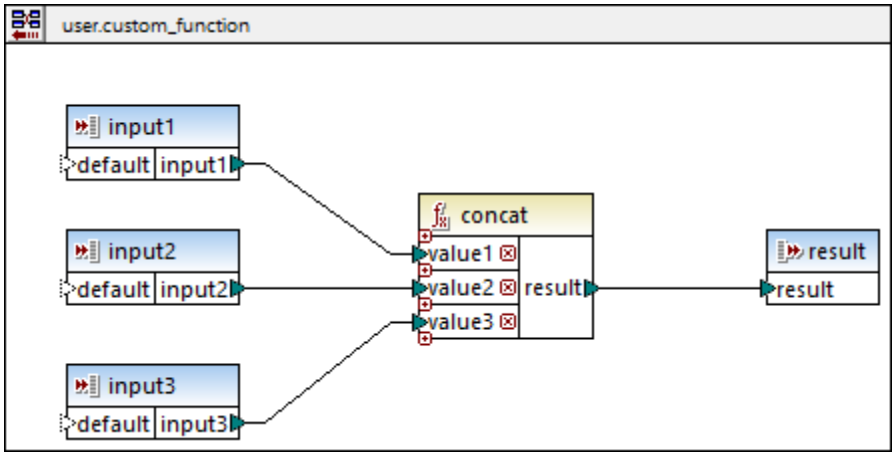
空のシーケンスを非シーケンスパラメーターに接続すると、パラメーターは関数が呼び出されないままの結果を取得します。

これは、ソース構成が任意のアイテムを持つ場合、またはフィルターの条件が一致するアイテムを返さない場合に発生します。これを回避するには、シーケンスが空にならないことを保証するために、関数を入力する前にsubstitute-missing 関数を使用し、パラメーターをシーケンスに設定し、関数内の空のシーケンスのために処理を追加します。

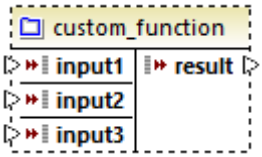
「出力はシーケンス」チェックボックスは、出力パラメータのために必要とされています。関数が複数の値のシーケンスを出力コンポーネントにパスし、出力コンポーネントがシーケンスに設定されていない場合、関数は、シーケンス内の最初のアイテムのみを返します。

パラメーターの順序

ユーザー定義関数が複数の入力または出力パラメーターを持つ場合、この関数の呼出元に表示されるパラメーターの順序を変更することができます。例えば、下の関数には3つのパラメーター（**input1**、**input2**、および**input3**）が存在します。



関数のマッピング内の(上から開始する)パラメーターの順序は、この関数の送信元が表示される順序を指定します:



以下の点に注意してください:

- 入力と出力パラメーターは上から下に位置が並べ替えられます。このため、パラメーター**input3** を関数のマッピングに移動すると、この関数のパラメーターは最初になります。
- 2つのパラメーターが同じ垂直の位置にある場合、左側のパラメーターが優先されます。
- 2つのパラメーターが同じ位置にある場合、内部コンポーネント ID が自動的に使用されます。

7.3.3 インラインと正規ユーザー定義関数

ユーザー定義関数には2つの種類が存在します: インラインと正規関数。関数を作成する際に、関数がインライン、または正規関するであるかを指定することができます。 ユーザー定義関数の作成を参照してください。インラインと正規関数は、コード生成、再帰性、複数のパラメータを持つことができる能力などで異なる振る舞いをします。

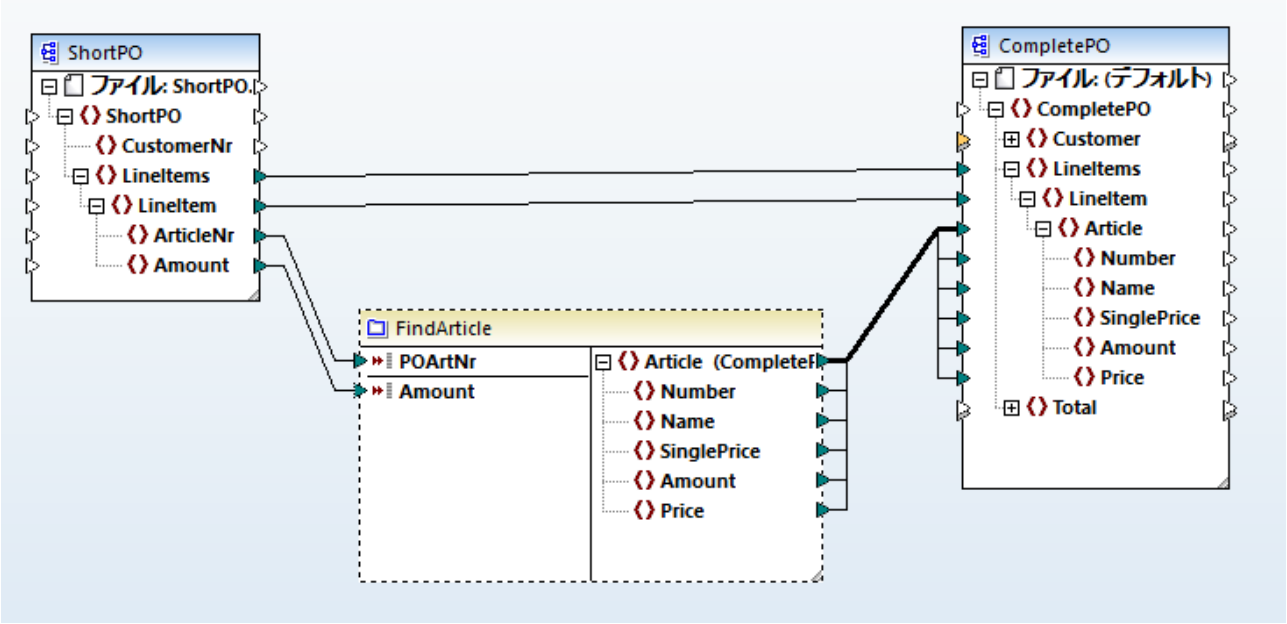
インライン関数	正規関数
インライン関数は、生成されたコード内で発生する箇所ですべてのインスタンス内から抽出することができ、コードは長くなりますが、若干速くなります。 インライン関数は、大幅に生成されたプログラムコードを増加することができます。ユーザー定義関数コードは、関数が呼び出されるすべての場所で実際に挿入されます。このため、正規表現とは異なり、コードのサイズが増加します。	各ユーザー定義関数コンポーネントは、入力がパラメーターにパスされる個所で、関数の呼び出しのためにコードを生成し、出力は値を返す関数(コンポーネント) ランタイムでは、すべての入力パラメーターが最初に評価され、関数が入力データの各発生のために呼び出されます。
インライン関数は、複数の出力を持つことができ、このため、複数の値を返します。	正規関数は1つの出力のみを持つことができます。複数の値を返すには、複数の値を発信者にパスすることを許可する出力を複合型(例えば、XML 構成)に宣言することができます。

(C) 2015-2021 Altova GmbH

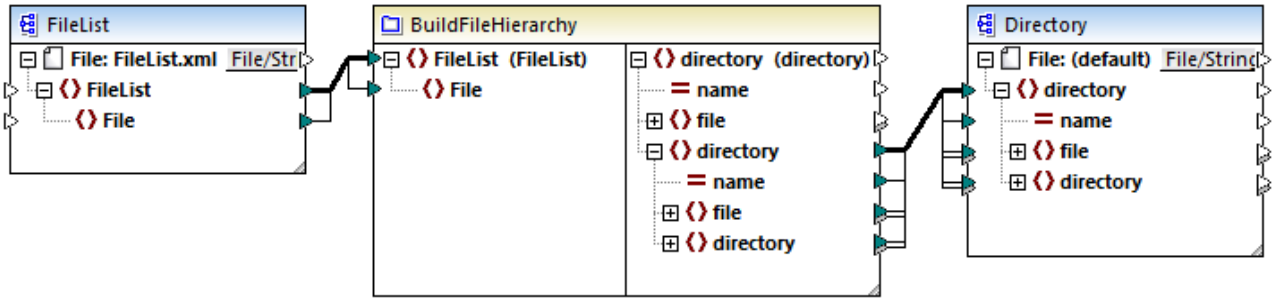
インライン関数	正規関数
インライン関数を再帰的に呼び出すことはできません。	正規関数を再帰的に呼び出すことは可能です。
インライン関数はパラメーター上で優先コンテキスト[優先コンテキスト]を設定することをサポートしません。	正規関数は、パラメーター上で優先コンテキストの設定をサポートします。

ユーザー定義関数を[インライン]から[正規]に切り替える、またはその逆の場合、[マッピングコンテキスト]に影響を与える場合があり、これはマッピングが異なる結果を生成する可能性を引き出します。

マッピング上で、インライン ユーザー定義関数は、点線の囲いにより表示されています。例えば、下のマッピング内の中間コンポーネントはインライン ユーザー定義関数です。



正規関数は、実線の囲いにより表示されています。例えば、下のマッピングの中間コンポーネントは正規のユーザー定義関数です。

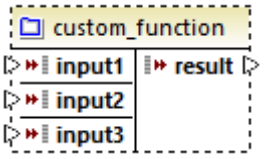


7.3.4 ユーザー定義関数のナビゲート

マッピングにユーザー定義関数が含まれている場合、各ユーザー定義関数とメインマッピングの間を下に示されているように簡単にナビゲートすることができます。

ユーザー定義関数を表示、または編集のために開く方法:

- マッピング上のユーザー定義関数のタイトルバーをダブルクリックします。



- ライブラリウィンドウの特定のユーザー定義関数をダブルクリックします。

ライブラリウィンドウの名前をダブルクリックして関数を編集することができます。現在アクティブなドキュメント内の関数をこの方法で開くことができます。他のマッピング内で作成されたユーザー定義関数をダブルクリックすると、新規のウィンドウ内にそのマッピングを開きます。

メモ 複数のマッピングにインポートされたユーザー定義関数を編集または削除する場合、この変更によりすべてのインポートされるマッピングが影響を受けます。

メインマッピングに戻る方法:

- メインマッピングに戻るには、マッピングウィンドウの左角の「メインマッピングに戻る」( ) ボタンをクリックします。

更に、ユーザー定義関数を含む多種のMapForce タブをナビゲートするとその履歴が保存されます。使用したタブ間を前後に移動するには、「戻る」( ) と「次へ」( ) ツールバーボタンをクリックします。これらのボタンに対応するキーボードのショートカットは、「Alt+Left」と「Alt+Right」です。

7.3.5 ユーザー定義関数の編集

ユーザー定義関数を編集する方法:

1. ユーザー定義関数を含むマッピングを開きます。
2. マッピング上のユーザー定義関数のタイトルバーをダブルクリックします。コンテンツを表示するために、必要に応じてコンポーネントを追加、編集、または削除することのできるマッピングウィンドウが変更されます。
3. (名前または詳細などの) 関数のプロパティを変更するために、以下の一つを行ってください:
   - マッピングの空白の部分を右クリックし、コンテキストメニューから「関数の設定」を選択します。
   - 「ユーザー定義関数設定」( ) ツールバーボタンをクリックします。

ライブラリウィンドウの名前をダブルクリックして関数を編集することができます。現在アクティブなドキュメント内の関数をこの方法で開くことができます。他のマッピング内で作成されたユーザー定義関数をダブルクリックすると、新規のウィンドウ内にそのマッピングを開きます。

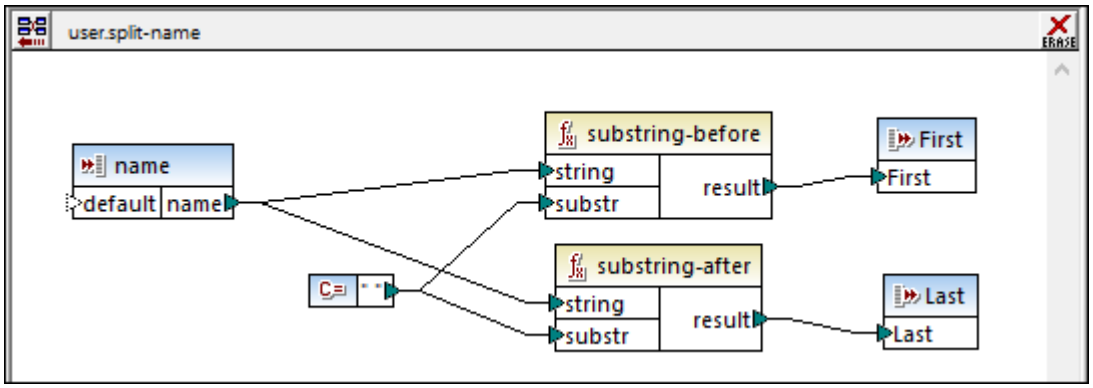
(C) 2015-2021 Altova GmbH

メモ 複数のマッピングにインポートされたユーザー定義関数を編集または削除する場合、この変更によりすべてのインポートされるマッピングが影響を受けます。

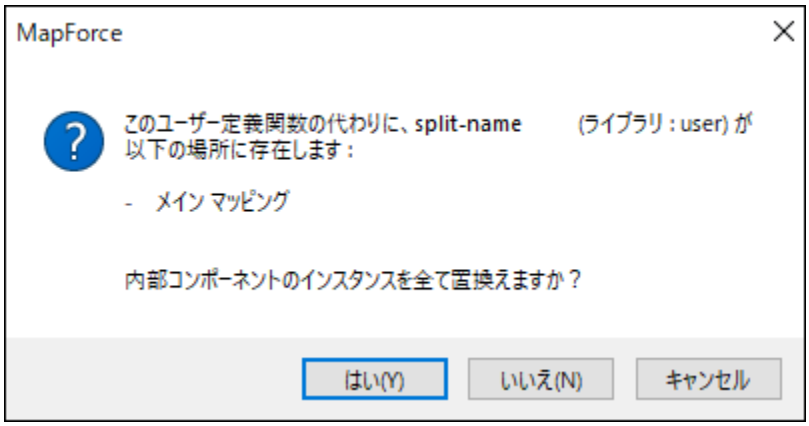
7.3.6 ユーザー定義関数の削除

マッピング上のユーザー定義関数のタイトルバーをダブルクリックしてユーザー定義関数を削除します。

1. マッピングウィンドウの右角の「削除」 ボタンをクリックします。



2. 現在開かれているマッピング内で関数が使用されている場合、ダイアログボックスが表示されます。



関するを削除し、関数のコンポーネントと共に呼び出されているすべてのインスタンスを置き換える場合、「はい」をクリックします。これにより、関数が削除される場合でも、メインマッピングを有効に保つことができます。削除された関数が他の外部のマッピングで使用されている場合、これらは無効になります。

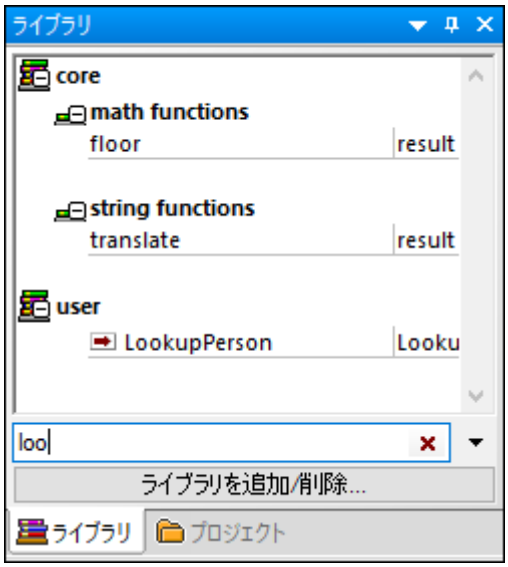
関数とその内部コンポーネントを永久に削除する場合は、「いいえ」をクリックします(この場合、関数が使用されるすべてのマッピングは無効になります)。

7.3.7 ユーザー定義関数の呼び出しとインポート

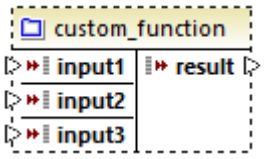
ユーザー定義関数の作成後、作成した同じマッピングから呼び出す、または他のMapForce マッピングから呼び出すことができます。

ユーザー定義関数を同じマッピングから呼び出す方法:

1. ライブラリウィンドウ内で関数を検索します。関数の作成時に指定したライブラリの下に関数が表示されます。デフォルトの「ユーザー」ライブラリを作成した場合、「ユーザー」ライブラリ内で関数を検索してください。名前別に関数を素早く検索するには、ライブラリウィンドウ内に名前を入力します。



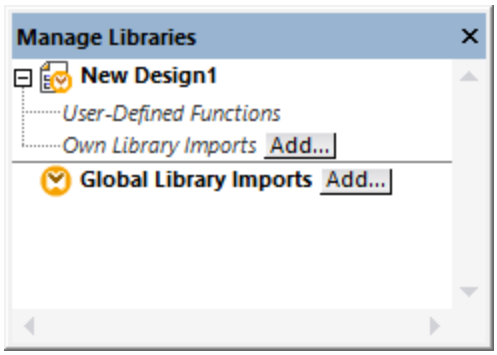
2. ライブラリウィンドウからメインマッピングに関数をドラッグします。必要とされるすべてのパラメーターを接続することができます。関数の結果は出力パラメーター(または、適用可能な場合複数のパラメーター)により適用されます。



ユーザー定義関数を他のマッピングからインポートする方法:

1. ライブラリ ウィンドウのベースのライブラリの追加/削除 ボタンをクリックします。ライブラリの管理ウィンドウが開かれます。

(C) 2015-2021 Altova GmbH

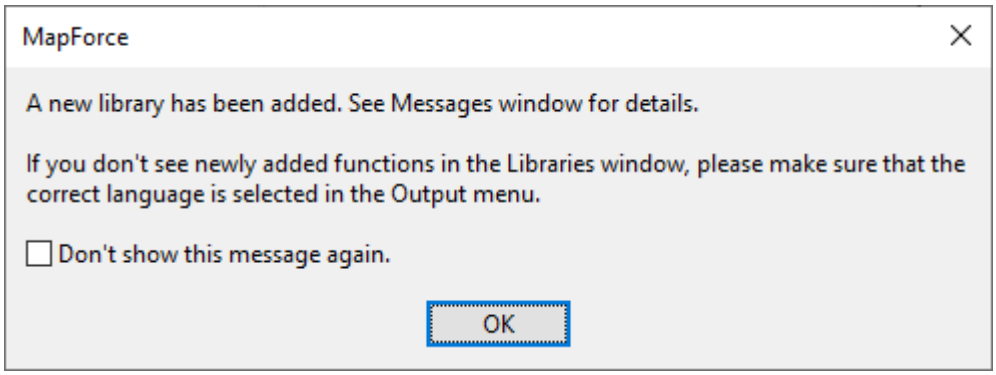


2. 以下の内の1つを行います:

   - (現在のマッピング内のスコープのみで) *ローカル*ライブラリとして関数をインポートする場合現在のマッピング名の下の「**追加**」ボタンをクリックします。
   - (プログラムレベルで) *グローバル*ライブラリとして関数をインポートする場合 **グローバルライブラリのインポート** の横の「**追加**」ボタンをクリックします。

メモ ライブラリを*ローカル*でインポートする場合、ライブラリファイルのパスをマッピングに対して相対的に設定することが可能です。グローバルにインポートされたライブラリでは、インポートされたライブラリへのパスは常に絶対的なパスです。

3. ユーザー定義関数が含まれるマッピングファイル(.mfd) を参照し、「**開く**」をクリックします。新規のライブラリが追加されたことを通知するメッセージボックスが表示され、新規のライブラリがライブラリウィンドウに表示されます。



ライブラリウィンドウからマッピングにドラッグして、現在のマッピング内のインポート済みの関数を使用することができます。関数をマッピングに追加するも参照してください。

ライブラリウィンドウはビルトイン関数と現在のマッピングファイル内のユーザー定義関数を表示します。上に示されるように他の.mfd ファイルをライブラリとして現在のマッピングにインポートすると他のインポート済みのファイルからのユーザー定義関数も表示されます。複数の*.mfd ファイルまたはカスタムライブラリ全体で同じライブラリ名を使用すると、ライブラリウィンドウ内で同じライブラリ名の下に表示される使用可能なすべてのソースからの関数が表示されます(カスタム Java と.NET ライブラリのインポートを参照してください)。

インポートされた.mfd ファイルが資格情報を含む場合、これらは資格情報マネージャー内で黄色い背景色で インポート済みと表示されます。デフォルトでは、インポートされた資格情報はメインマッピングに保存されませんが、任意でローカルコピーを作成し、メインマッピングに保存することができます。ユーザー定義関数内の資格情報 も参照してください。

関数ライブラリの表示と整理に関する詳細は関数ライブラリの管理を参照してください。

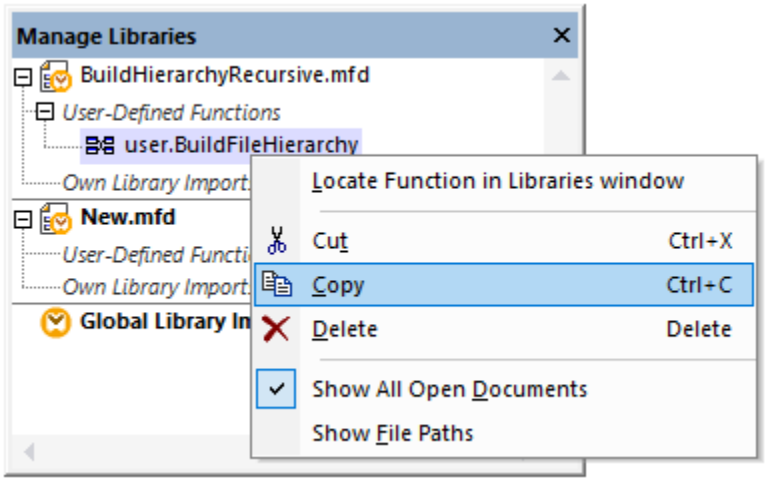
7.3.8 マッピング間で UDF をコピーし張り付ける用法

以下のようにマッピング間でユーザー定義関数を簡単にコピーして貼り付けることができます:

1. ライブラリウィンドウのベースのライブラリの追加/削除 をクリックします。ライブラリの管理ウィンドウが開かれます。
2. ライブラリウィンドウ内の空のエリアを右クリックしオプション「全ての開かれているドキュメントを表示」を選択します。
3. ソースと目的マッピングを両方開きます。例えば、下のイメージではソースは**BuildHierarchyRecursive.mfd** で、目的のマッピングは**New.mfd** です。

メモ ソースとターゲットマッピングが既にディスクに保存されていることを確認してください。これにより正確なパスが保証されます。相対的なパスのコピーと張り付けを参照してください。

4. ソースマッピングファイルからのユーザー定義関数を右クリックし、コンテキストメニューからコピー を選択します。(または「Ctrl+C」を押します)。

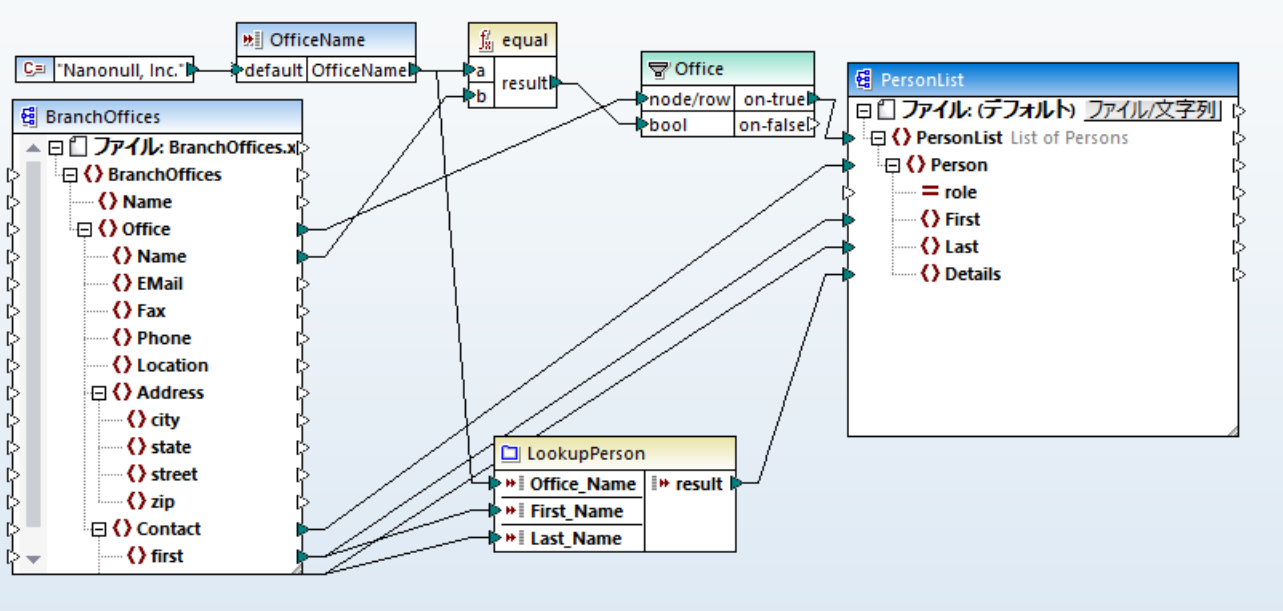


5. ターゲットマッピングファイルの「ユーザー定義関数」エントリを右クリックし、コンテキストメニューから「張り付け」 を選択します。

関数ライブラリの管理も参照してください。

7.3.9 例: ルックアップと連結

ユーザー定義関数の一般的な使用方法を説明したいくつかのデモマッピングがMapForce で使用することができます。**PersonListByBranchOffice.mfd** ファイルのいずれは<マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥フォルダー内で見つけることができます。



PersonListByBranchOffice.mfd

マッピングには以下の必要条件があります:

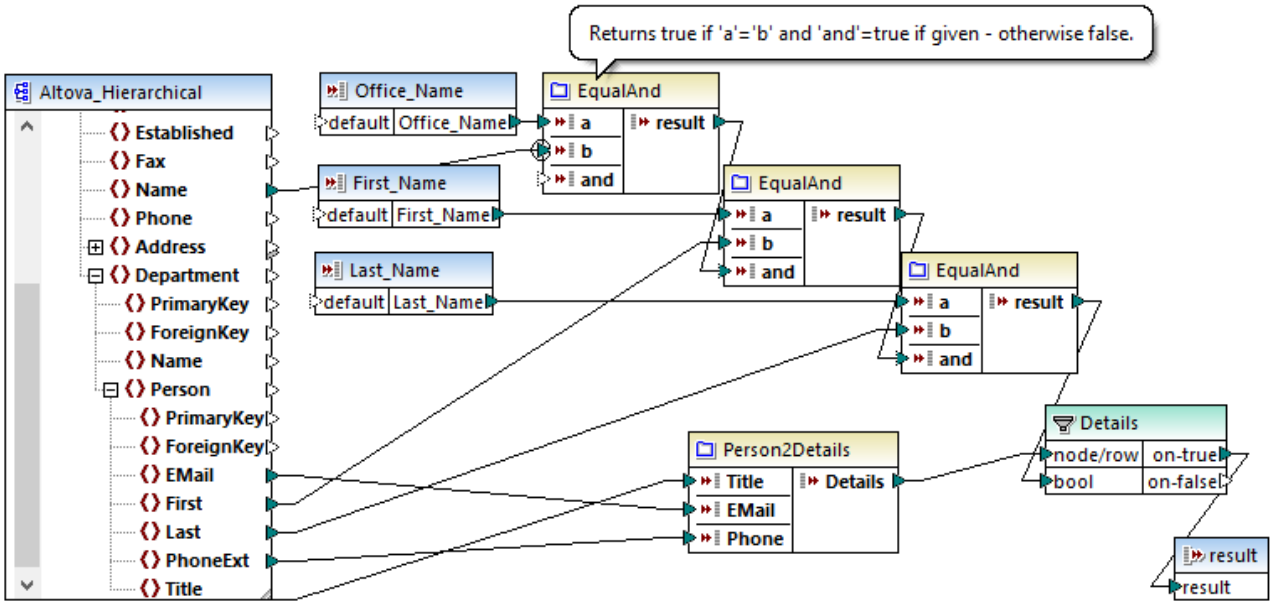
- XML ファイルからデータを抽出し、ターゲット XML ファイルに書き込みます。データは氏名など従業員の詳細から構成されています。
- 個別のXML ファイル(電話、電子メールアドレス、職位) 内で個別の従業員に関するデータを検索します。
- ターゲットに書き込む前に希望する方法でデータを処理します。具体的には、電話番号、電子メール、職位が単一の文字列で表示されており、ターゲット XML の **詳細** 要素に書き込まれています。
- XML 要素、この場合、特定の部署からの従業員に関する情報のみを抽出します。マッピングの発信者はオフィス名をコマンドラインでパラメーターとして指定できる必要があります。例えば、マッピングがMapForce Server により実行されている場合など。

上記の必要条件を実装するコンポーネントを検証してみましょう:

- マッピング("OfficeName") の入力パラメーターはシンプルな入力コンポーネントです。デフォルトの値("Nanonull, Inc.") は定数により提供されます。この値はマッピングの呼出元がパラメーターの値を提供しない場合使用されます。マッピングにパラメーターを提供するを参照してください。
- 特定のオフィスに属する従業員をフィルターするために、マッピングはフィルターコンポーネント ("Office") を使用します。基本的には、フィルターはパラメーターにより提供されているオフィス名がソースXML ファイル内のオフィス名と等価かどうかをチェックします。等価の場合、フィルターはソース **Office** アイテムからデータをターゲットコンポーネントにパスします。フィルターに関する詳細はフィルターと条件を参照してください。.
- 2番目のソースXML ファイルからの情報を検索するには、マッピングはユーザー定義関数 "LookupPerson" を呼び出します。この関数のロジックは以下で詳細に説明されています。
- データを処理するために"LookupPerson" 関数は内部で適切な方法で各従業員に関する情報を抽出し連結する他の関数を呼び出します。これらすべての操作は巻数の自身のマッピング内にあり、メインマッピング内では表示されません。これは一般的なカプセル化の例です。"LookupPerson" 関数は **Details** 要素をターゲット XML 内で作成します。

ルックアップの実行

ルックアップ機能は機能について以下で説明されている"LookupPerson" 関数により提供されています。

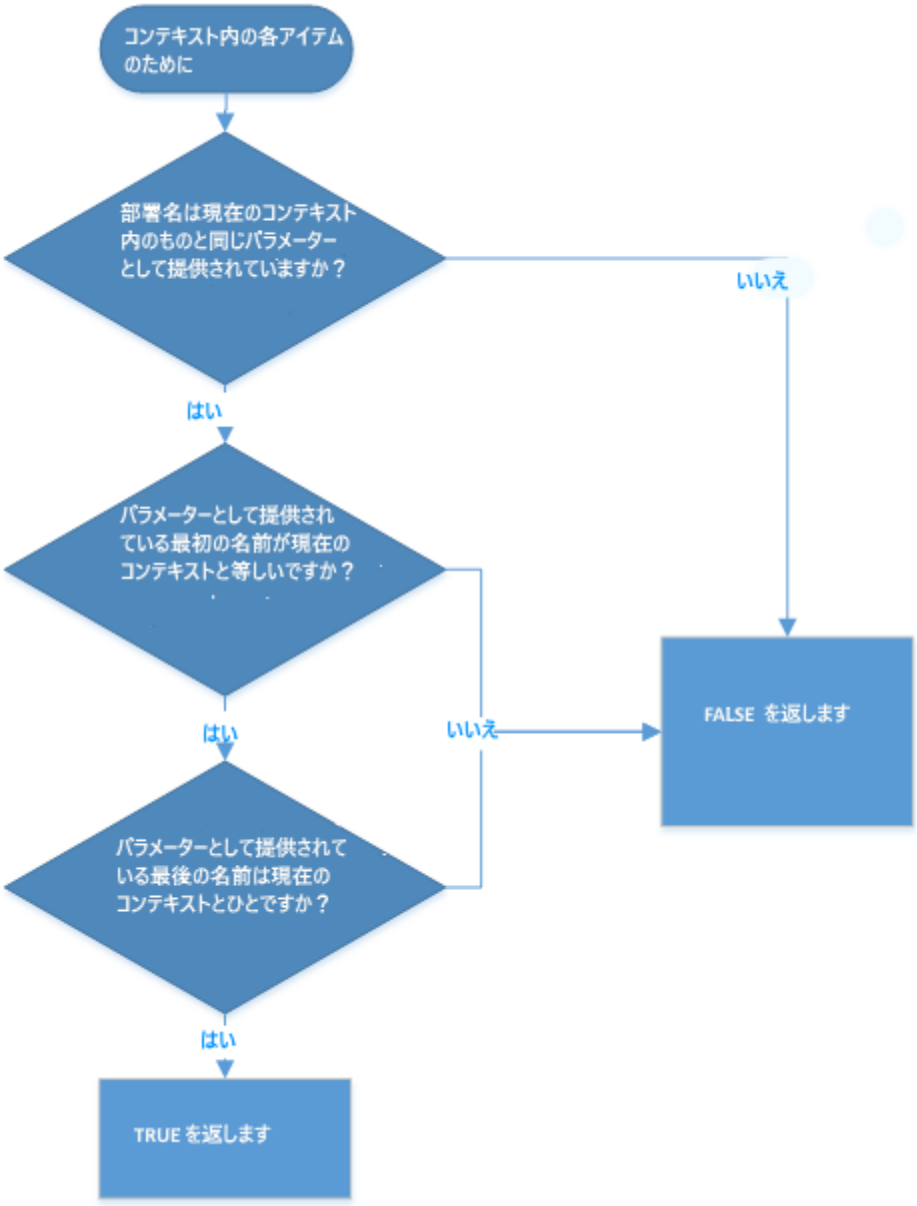


"LookupPerson" function

上記の通り、関数にはデータを抽出することのできるソースXML ファイルが含まれています。次に、ルックアップ値を提供する3つの入力パラメーターが存在します: **Office_Name**、**First_Name**、および**Last_Name**。捨てべの入力パラメーターは必須として設定されます（すなわち、チェックボックス「入力が必要とされています」がプロパティダイアログボックス内で選択されています）。

"EqualAnd 関数" は現在のものに囲まれている個別のユーザー定義関数です。この機能はブール値を返します。上記のシーケンス内でのこの関数の呼び出しは以下のブールのロジックを説明しています:

(C) 2015-2021 Altova GmbH

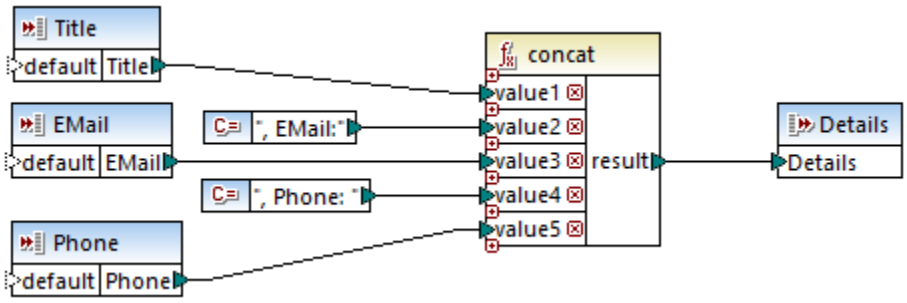


関数の値（TRUE または FALSE）は新規のアイテムが処理される都度フィルターにパスされます。フィルターが値 TRUE を得ると、ルックアップオペレーションは成功し、従業員の詳細が抽出され外部のマッピングに返されます。それ以外の場合、コンテキスト内の次のアイテムが検査され、ループが完了するまで継続されます。

さいしょの "EqualAnd" 関数の発生では、コネクタ **b** は丸枠で囲まれており、これはこのパラメーターが優先コンテキストとして設定されていることを示しています。優先コンテキストはマッピングの実行を最適化する任意の機能です。具体的には、特定のオフィスの個人情報が入力パラメータ **a** により提供され、最初に処理されます。優先コンテキストとしてパラメーターを設定するには、右クリックして、コンテキストメニューから **優先** を選択します。 詳細に関しては 優先コンテキスト を参照してください。

実行の連結

The "Person2Details" 関数は "LookupPerson" 関数にネストされている他の関数です。この関数は単一の値を返します。以下で説明されているとおり、パラメーターとして受け取った3つの値と2つのテキスト定数を連結します:



"Person2Details" function

concat 関数はビルトイン関数です。必要とする数量のパラメーターを取ることのできるMapForce ビルトイン関数です。関数の引数の追加または削除を参照してください。

マッピングの実行

MapForce 内のマッピングの実行をプレビューするには「出力」タブをクリックします。マッピングはデフォルトの入力パラメーター（「Nanonull, Inc.」）を使用して実行され、この結果このオフィスからのみ従業員の情報を抽出します。他のオフィスからデータを抽出するには、入力パラメーターに背景されている定数を「Nanonull, Inc.」から「Nanonull Partners, Inc.」に変更しマッピングを再度実行します。

ライセンス供与済みのMapForce Server を使用することができる場合、Linux、macOS、またはW indows マシン上のコマンドライン上でマッピングを実行することができます。最初に、メニューコマンド「ファイル | MapForce Server 実行可能ファイルにコンパイル」を使用してMapForce Server 実行ファイル(.mfx) にマッピングをコンパイルします。MapForce Server 実行可能ファイルにマッピングをコンパイルするを参照してください。次に、.mfx ファイルをサーバーマシンにコピーし、以下のコマンドでMapForce Server を実行します。名前が付けられたパラメーター -p=OfficeName は入力パラメーターを提供します:

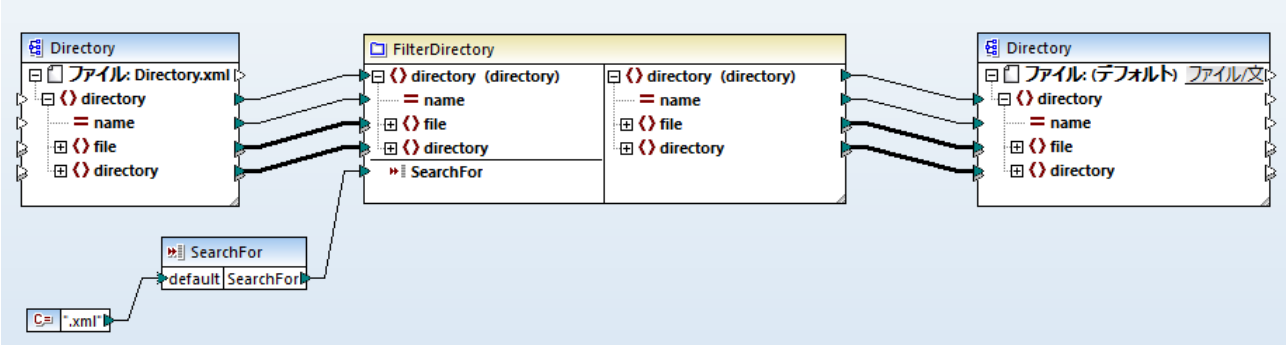
```
mapforceserver run PersonListByBranchOffice.mfx -p=OfficeName:"Nanonull, Inc."
```

メモ

- **mapforceserver** はMapForce Server 実行可能ファイルのパスで、使用中のオペレーティングシステムに適用可能です。
- 適用できる場合 .mfx ファイルにパスを変更します。または、.mfx を実行可能ファイルと同じフォルダーにコピーします。

7.3.10 例:再帰的な検索

このサンプルは、再帰的なユーザー定義関数の助けを使用してソースXML ファイル内でデータを検索するマッピングを表しています。マッピングファイルは次のパスで見つけることができます: **<マイドキュメント>\Altova\MapForce2021\MapForceExamples\RecursiveDirectoryFilter.mfd**。



(C) 2015-2021 Altova GmbH

RecursiveDirectoryFilter.mfd

ソースXMLファイルには、下のコードリストに示されているとおり、ファイルとディレクトリに関する情報が含まれています(リスティングではデータの一部が省略されていることにご注意ください):

```
<?xml version="1.0" encoding="UTF-8"?>
<directory name="Examples">
   <directory name="ExampleSite">
      <file name="blocks.sps" size="7473"/>
      <file name="block_file.xml" size="992"/>
      <directory name="output">
         <file name="examplesite1.css" size="3174"/>
         <directory name="images">
            <file name="blank.gif" size="88"/>
            <file name="block_file.gif" size="13179"/>
         </directory>
      </directory>
   </directory>
</directory>
```

ソースXMLファイル

ソースとターゲットXMLファイルは同じスキーマ**Directory.xsd**を使用します。ファイルシステム上で、ディレクトリはファイルまたは他のディレクトリを含むことができ、これはスキーマ内で反映されます。重要な点は、スキーマが「ディレクトリ」要素が*再帰的*であることを指定していることです(ライン`<xs:element ref="directory"/>`を参照してください)。

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified"
attributeFormDefault="unqualified">
   <xs:element name="directory">
      <xs:complexType>
         <xs:choice minOccurs="0" maxOccurs="unbounded">
            <xs:element name="file">
               <xs:complexType>
                  <xs:attribute name="name" type="xs:string"/>
                  <xs:attribute name="size" type="xs:unsignedLong"/>
               </xs:complexType>
            </xs:element>
            <xs:element ref="directory"/>
         </xs:choice>
         <xs:attribute name="name"/>
      </xs:complexType>
   </xs:element>
</xs:schema>
```

Directory.xsd

マッピングのビジネスの必要条件は、特定の拡張子を持つファイルの*み*をフィルターします。すべてのディレクトリのネストされた構造は、保存される必要があります。例えば、拡張子が".xml"の場合、(ソースXMLファイルが以前にリストされているため)期待される出力は以下に類似します:

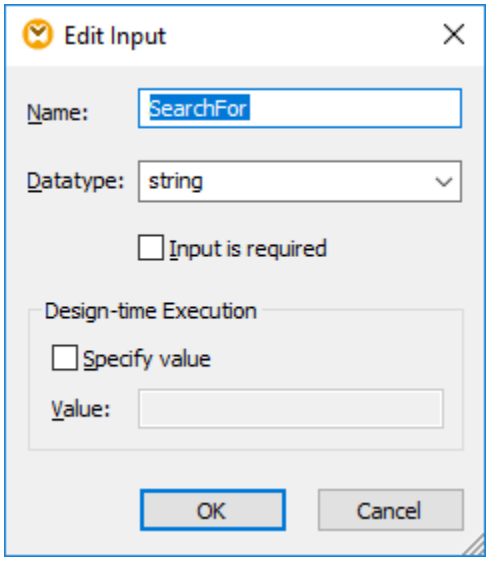
```
<?xml version="1.0" encoding="UTF-8"?>
<directory name="Examples">
   <directory name="ExampleSite">
      <file name="block_file.xml" size="992"/>
```

```
      <directory name="output">
         <directory name="images"/>
      </directory>
   </directory>
</directory>
```

期待される XML 出力

2番目に、マッピングの発信者は、パラメーターとしてファイル拡張子を提供する必要があります。デフォルトで、発信者がパラメーターの値を提供しない場合、マッピングは拡張子を持つ .xml ファイルをフィルターアウトします。

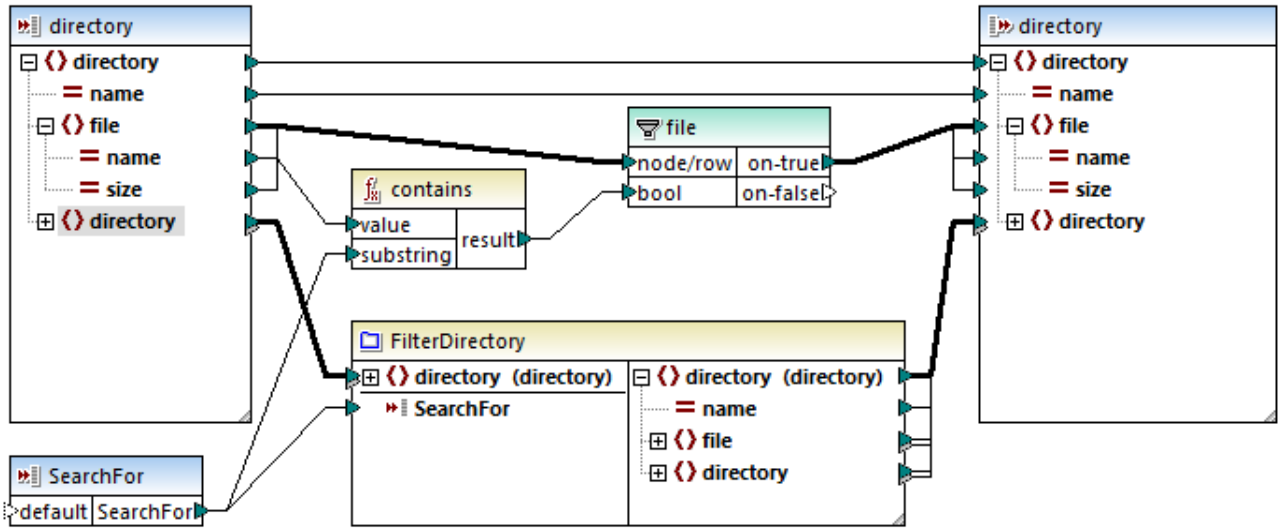
上記の必要条件を対処するために、マッピングにはテキスト定数を介してデフォルトのファイルの拡張子を提供する単純型入力パラメーター、"SearchFor"、が含まれています。このパラメーターは任意です（「**入力は必須**」チェックボックスはプロパティダイアログボックス内で選択されていません）:



入力パラメーターに関する詳細は、マッピングにパラメーターを与えるを参照してください。

次に、マッピングにはユーザー定義関数、"FilterDirectory" が含まれています。この関数は再帰的で、自身への呼び出しも含んでいます。再帰的な要素の「**ディレクトリ**」に接続されているため、この関数はソース XML インスタンス内のネストされた「**ディレクトリ**」要素を必要なだけ呼び出すことができます。再帰的な呼び出しをサポートするために、この関数は、インライン関数ではなく、正規関数として作成されています（「**インラインの使用**」オプションは、関数のプロパティ内で選択されていません）。関数のプロパティを確認するには、マッピング内の空のエリアを右クリックし、コンテキストメニューから「**関数の設定**」を選択します。 ユーザー定義関数の編集を参照してください。

(C) 2015-2021 Altova GmbH



上記で示されるように、関数は入力として2つのパラメーターを取ります:

1. 検索されるXML構成を定義する複合型パラメーター、**Directory**、(このパラメーターは「haystack」です)。
2. 検索するファイル拡張子を指定する文字列パラメーター、**SearchFor**、(このパラメーターは「needle」です)。

マッピング上の入力または出力パラメーターのタイトルバーを右クリックして、設定を確認します。

関数には、MapForce ビルトイン関数 `contains` が接続されているフィルターコンポーネントが含まれています。`contains` 関数は、検索の値がソース構成内の"name"属性(ファイル名)に一致する場合、**true** を返します。**true** の値は、フィルターに現在のアイテムを出力にコピーするように命令します。それ以外の場合、スキップされます。フィルターに関する詳細は、フィルターと条件を参照してください。

関数の**directory** パラメーター(入力と出力の双方)同様、マッピングの、ソースとターゲットファイルは同じスキーマ**Directory.xsd** を持ちます。MapForce は、これらの型の割り当ては整合性があると検知し、入力パラメーターと関数間の接続の型は「全てコピーする」です。全てコピー接続を参照してください。

マッピングの実行

MapForce 内で、マッピングの実行をプレビューするには、「出力」タブをクリックします。デフォルトの入力パラメーター(".xml")を使用してマッピングは実行され、この結果、この検索結果に一致する結果のみが抽出されます。異なる検索の条件を提供するには、入力パラメーターへ接続されている定数を".xml"から".sps"に変更し、マッピングを再度実行します。

ライセンスが提供されたMapForce Server を使用できる場合、Linux、macOS、またはW indows マシン上のコマンドラインでマッピングを実行することができます。最初に、メニューコマンド「ファイル | MapForce Server 拡張ファイルにコンパイルする」を使用して、MapForce Server 拡張ファイル(.mfx)にマッピングをコンパイルします。また、MapForce Server 拡張ファイルにマッピングをコンパイルするを参照してください。.mfx ファイルをサーバーマシンにコピーし、下のコマンドを使用してMapForce Server を実行します。名前付けられたパラメーター `-p=SearchFor` は入力値を提供します:

```
mapforceserver run RecursiveDirectoryFilter.mfx -p=SearchFor:".sps"
```

メモ

- **mapforceserver** は、使用中のオペレーティングシステムに適用することのできるMapForce Server 実行可能へのパスです。
- .mfx ファイルへのパスを適用できるように変更、または実行可能と同じフォルダー内に.mfx をコピーします。

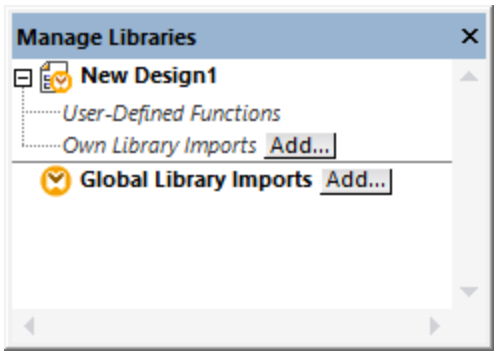
7.4 カスタム XSLT 1.0 または 2.0 関数のインポート

XSLT 1.0、XSLT 2.0 とXSLT3.0 関数ライブラリを MapForce 内で使用することのできる単純型を返すカスタム関数として拡張することができます。

> 簡単なデータ型（例えば、文字列）を返すカスタム関数のみがサポートされます。

XSLT ファイルから関数をインポートする:

1. ライブラリ ウィンドウのベースの ライブラリの追加/削除 ボタンをクリックします。ライブラリの管理ウィンドウが開かれます。



2. 以下の内の1つを行います:

   - （現在のマッピング内のスコープのみで） ローカル ライブラリとして関数をインポートする場合現在のマッピング名の下の「追加」ボタンをクリックします。
   - （プログラムレベルで） グローバル ライブラリとして関数をインポートする場合 グローバルライブラリのインポート の横の「追加」ボタンをクリックします。

メモ ライブラリをローカルでインポートする場合、ライブラリファイルのパスをマッピングに対して相対的に設定することが可能です。グローバルにインポートされたライブラリでは、インポートされたライブラリへのパスは常に絶対的なパスです。

3. 関数が含まれる .xsl ファイルを参照し、「開く」をクリックします。メッセージボックスが表示され、新規のライブラリが追加されていることを通知します。

インポートされたXSLT ファイルは、ライブラリウィンドウ内のライブラリに表示され、全ての名前を持つテンプレートを関数として、ライブラリ名の下に表示されます。インポート済みのライブラリが表示されていない場合、XSLT が変換言語として選択されていることを確認してください。関数ライブラリの管理も参照してください。

以下の点に注意してください:

- MapForce にインポートが許可されている場合、関数はXSLT ファイル内でXSLT 仕様に準拠している名前が付けられたテンプレートとして宣言されている必要があります。XSLT 2.0 ドキュメント内で発生する関数を `<xsl:function name="MyFunction">` の書式でインポートすることもできます。インポートされたXSLT ファイル内で、他のXSLT ファイルをインポートまたは含むことができ、これらのXSLT ファイルと関数もインポートされることができます。
- インポートされたカスタム関数のマップすることのできる入力コネクタはテンプレート呼び出し内で使用されるパラメーターの数により異なります。任意のパラメーターもサポートされています。
- 名前空間はサポートされています。

(C) 2015-2021 Altova GmbH

- MapForce に既にインポートされている XSLT ファイルの更新を行う場合、変更は自動的に検知され、MapForce がファイルの再ロードを促します。
- 名前付きのテンプレートに書き込む場合、テンプレート内で使用されている XPath ステートメントが正しい名前空間を持つことを確認してください。マッピングの名前空間にバインドする名前空間を確認するには、生成された XSLT コードをプレビューしてください。

XPath 2.0 内のデータ型

XML ドキュメントから XML スキーマが参照されており、スキーマに対してドキュメントが妥当である場合、演算にて目的のデータ型へ暗示的に変換されないデータ型を明示的に構築またはキャストする必要があります。

Altova XSLT 2.0 エンジンで使用される XPath 2.0 データモデルでは、XML ドキュメントから原子化された全てのノード値に `xs:untypedAtomic` データ型が割り当てられます。`xs:untypedAtomic` 型は、以下にあるように暗示的な型変換に使用されます:

例えば、

- `xdt:untypedAtomic` 値が加算演算子により `xs:double` に*暗黙的に*変換されるため式 `xs:untypedAtomic("1") + 1` は2の値を結果とします。
- 四則演算では、オペランドが暗黙的に `xs:double` へ変換されます。
- 値の比較を行う場合、比較の前にオペランドが `xs:string` へ変換されます。

次も参照してください

例: カスタム XSLT 関数の追加
例: ノードの値の集計
XSLT 1.0 エンジンの実装
XSLT 2.0 エンジンの実装

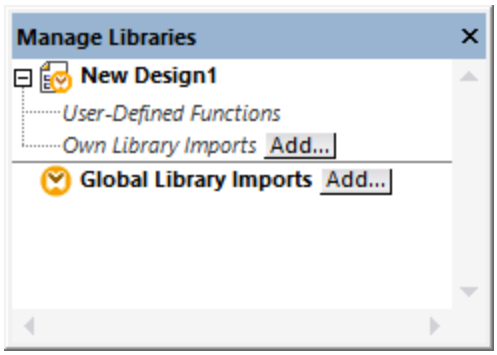
7.4.1 例: カスタム XSLT 関数の追加

この例は、カスタム XSLT 1.0 関数を MapForce にインポートする方法を説明しています。以下に示される単純なサンプルに必要なファイルは <マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥ ディレクトリに収められています。

- **Name-splitter.xslt**. この XSLT ファイルは"string" のパラメーターを伴う "**tokenize**" という名前付きテンプレートを定義しています。テンプレートは入力文字列と作業し、それぞれの大文字の文字を空白文字で区切ります。
- **Name-splitter.xml** (処理されるソース XML インスタンスファイル)
- **Customers.xsd** (ソース XML スキーマ)
- **CompletePO.xsd** (ターゲット XML スキーマ)

カスタム XSLT 関数の追加方法:

1. ライブラリ ウィンドウのベースのライブラリの追加/削除 ボタンをクリックします。ライブラリの管理ウィンドウが開かれます。

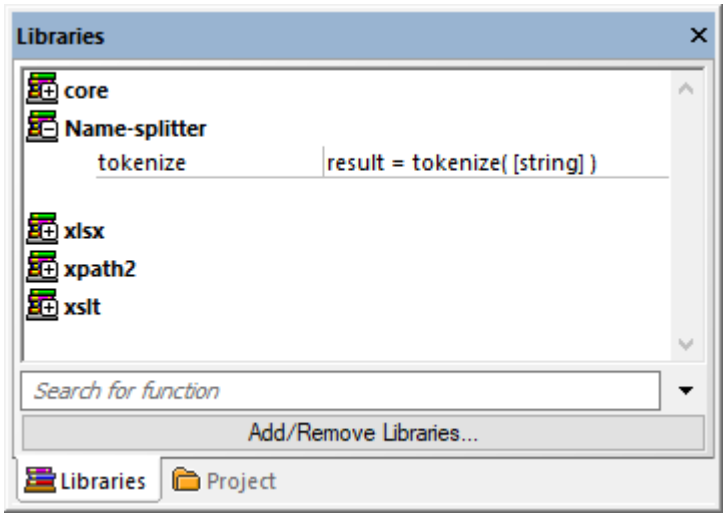


2. 以下の内の1つを行います:

   - (現在のマッピング内のスコープのみで) *ローカル*ライブラリとして関数をインポートする場合現在のマッピング名の下の「**追加**」ボタンをクリックします。
   - (プログラムレベルで) *グローバル*ライブラリとして関数をインポートする場合 **グローバルライブラリのインポート** の横の「**追加**」ボタンをクリックします。

メモ ライブラリを*ローカル*でインポートする場合、ライブラリファイルのパスをマッピングに対して相対的に設定することが可能です。グローバルにインポートされたライブラリでは、インポートされたライブラリへのパスは常に絶対的なパスです。

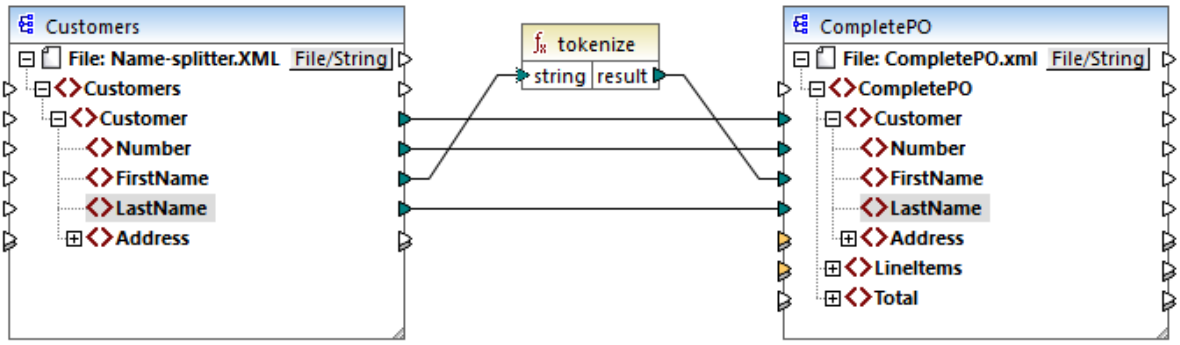
3. 関数として振る舞う名前の付けられたテンプレートを含む.xsl または.xslt ファイルを参照します。この場合は **Name-splitter.xslt** です。「開く」をクリックします。新規のライブラリが追加されたことを通知するメッセージボックスが表示され、ライブラリウィンドウ内に名前の付けられたテンプレートとして定義された関数と共にXSLT ファイル名が表示されます。(この場合は、`tokenize` 関数を持つ **Name-splitter**)。



XSLT 関数を MapForce から使用する方法:

1. `tokenize` 関数をマッピングウィンドウにドラッグして、以下に示されるようにアイテムをマップします。

(C) 2015-2021 Altova GmbH



2. **XSLT** タブをクリックして生成されたXSLTコードを確認します。

```
1   <?xml version="1.0" encoding="UTF-8"?>
2   <!--...-->
11  <xsl:stylesheet version="2.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform" xmlns:xs="http
12      <xsl:include href="file:///C:/Users/altova/Documents/Altova/MapForce2020/MapForceExamples
13      <xsl:output method="xml" encoding="UTF-8" byte-order-mark="no" indent="yes"/>
14      <xsl:template match="/">
15          <CompletePO>
16              <xsl:attribute name="xsi:noNamespaceSchemaLocation" namespace="http://www.w3.org/
    CompletePO.xsd'"/>
17              <xsl:for-each select="Customers/Customer">
18                  <Customer>
19                      <Number>
20                          <xsl:sequence select="xs:string(xs:integer(fn:string(Number)))"/>
21                      </Number>
22                      <FirstName>
23                          <xsl:call-template name="tokenize">
24                              <xsl:with-param name="string" select="FirstName" as="item()"/>
25                          </xsl:call-template>
26                      </FirstName>
27                      <LastName>
28                          <xsl:sequence select="fn:string(LastName)"/>
29                      </LastName>
30                  </Customer>
31              </xsl:for-each>
32          </CompletePO>
33      </xsl:template>
34  </xsl:stylesheet>
35
```

メモ マッピング内で名前の付けられたテンプレートが使用されると、名前の付けられたテンプレートを含むXSLTファイルは生成されたXSLTコード（**xsl:include href...**）内に含まれます。そしてコマンド **xsl:call-template** を使用して呼び出されます。

3. 「出力」タブをクリックして、マッピングの結果を確認します。

MapForce からカスタム XSLT ライブラリを削除する方法：

1. ライブラリ ウィンドウのベースのライブラリの追加/削除 をクリックします。ライブラリの管理ウィンドウが開かれます。
2. 削除するライブラリの横のライブラリの削除 ✕ をクリックします。

7.4.2 例: ノードの値の集計

このセクションではXML インスタンスドキュメント内にある複数のノードを処理し、その結果をターゲットアイテムにある単一のアイテムへマッピングする方法について記述します。この例で使用されるファイルは以下のとおりで、具体的には、マッピングの目的は、ソースXML ファイル内の全ての製品の値段を計算し、出力 XML ファイルに単一の値として書き込むことです。この例で使用されるファイルは以下のとおりで、<マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥Tutorial¥フォルダーに収められています:

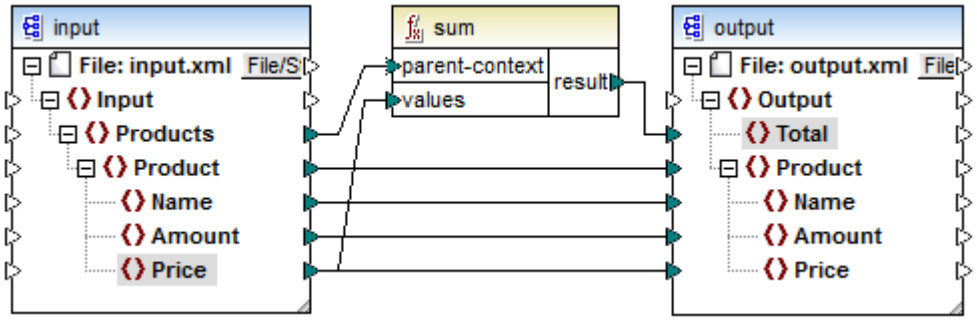
- **Summing-nodes.mfd** – マッピングファイル
- **input.xml** – ソースXML ファイル
- **input.xsd** – ソースXML スキーマ
- **output.xsd** – ターゲット XML スキーマ
- **Summing-nodes.xslt** – カスタムXSLT スタイルシートには個別のノードを集計する名前が付けられたテンプレートが含まれています。

以下の2 つの方法により、集計関数を使用することができます:

- sum 関数を使用します。このMapForce ビルトイン関数はライブラリウィンドウ内で使用することができます。
- カスタムXSLT スタイルシートをMapForce にインポートします。

ソリューション1: "sum" 集計関数の使用

sum 関数をマッピング内で使用するには、関数をライブラリウィンドウからマッピングにドラッグします。ライブラリウィンドウ内で使用することのできる関数は、選択されたXSLT 言語バージョン(XSLT 1 またはXSLT 2)により異なります。次に、以下に表示されるようにマッピング接続を作成します。



コアライブラリの集計関数に関する情報に関しては core | 集計関数を参照してください。

ソリューション2: カスタムXSLT スタイルシートの使用

上記の通りにこのサンプルの目的はソースXML ファイル内の製品の Price フィールドを集計することです。この場合製品 A とB です。

```
<?xml version="1.0" encoding="UTF-8"?>
<Input xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="input.xsd">
   <Products>
      <Product>
         <Name>ProductA</Name>
         <Amount>10</Amount>
         <Price>5</Price>
      </Product>
```


(C) 2015-2021 Altova GmbH

```
      <Product>
         <Name>ProductB</Name>
         <Amount>5</Amount>
         <Price>20</Price>
      </Product>
   </Products>
</Input>
```

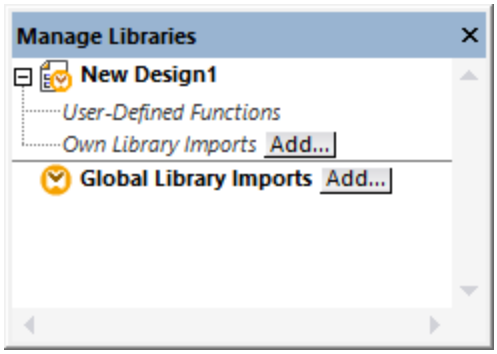
下のコードリスティングは、名前をつけられたテンプレート "Total" と単一パラメーター `string` を使用するカスタム XSLT スタイルシートを表示しています。テンプレートは、XML 入力ファイルを介して作動し、XPath 式 `/Product/Price` により取得された全ての値を合計します。

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
   <xsl:output method="xml" version="1.0" encoding="UTF-8" indent="yes"/>

   <xsl:template match="*">
      <xsl:for-each select=".">
      <xsl:call-template name="Total">
         <xsl:with-param name="string" select="."/>
      </xsl:call-template>
      </xsl:for-each>
   </xsl:template>

   <xsl:template name="Total">
   <xsl:param name="string"/>
      <xsl:value-of select="sum($string/Product/Price)"/>
   </xsl:template>
</xsl:stylesheet>
```

メモ XSLT 2.0 内のノードを合計するには、スタイルシートの宣言を `version="2.0"` に変更してください。

XSLT スタイルシートを MapForce にインポートする前に XSLT 1.0 を変換言語として選択してください。カスタム関数をインポートする準備が整いました。以下を行ってください:

1. ライブラリ ウィンドウのベースのライブラリの追加/削除 ボタンをクリックします。ライブラリの管理ウィンドウが開かれます。

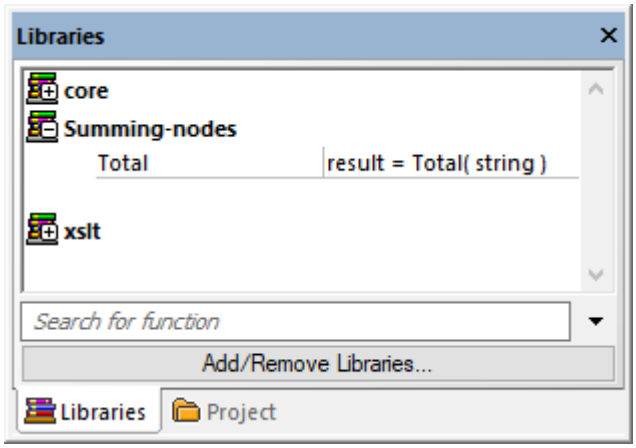


2. 以下の内の 1 つを行います:

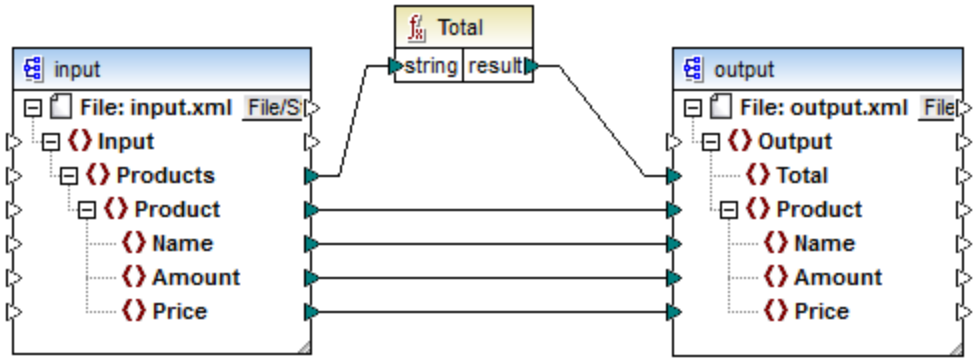
   - (現在のマッピング内のスコープのみで) ローカルライブラリとして関数をインポートする場合現在のマッピング名の下の「追加」ボタンをクリックします。
   - (プログラムレベルで) グローバルライブラリとして関数をインポートする場合 グローバルライブラリのインポート の横の「追加」ボタンをクリックします。

メモ ライブラリをローカルでインポートする場合、ライブラリファイルのパスをマッピングに対して相対的に設定することが可能です。グローバルにインポートされたライブラリでは、インポートされたライブラリへのパスは常に絶対的なパスです。

3. <マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥Tutorial¥Summing-nodes.xslt を参照して「開く」をクリックします。新規のライブラリが追加されたことを通知するメッセージボックスが表示され、新規のライブラリがライブラリウィンドウに表示されます。



4. ライブラリから `Total` 関数をマッピングにドラッグし、下に表示されるとおりマッピング接続を描きます。



マッピングの結果をプレビューするには、「出力」タブをクリックします。2つの `Price` フィールドの合計が `Total` フィールドに表示されます。

```
<?xml version="1.0" encoding="UTF-8"?>
<Output xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="output.xsd">
   <Total>25</Total>
   <Product>
      <Name>ProductA</Name>
      <Amount>10</Amount>
      <Price>5</Price>
   </Product>
   <Product>
      <Name>ProductB</Name>
      <Amount>5</Amount>
      <Price>20</Price>
   </Product>
</Output>
```


(C) 2015-2021 Altova GmbH

7.5 カスタム XQuery 1.0 関数のインポート

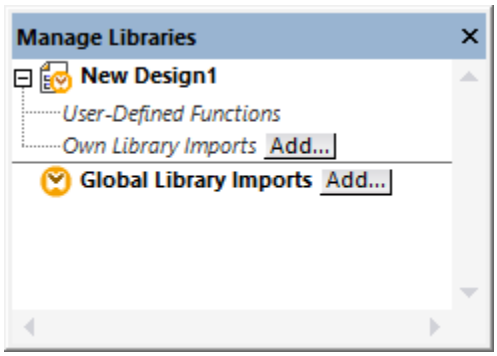
XQuery がマッピングの変換言語として選択されている場合、MapForce は、XQuery のために使用することのできるビルトイン関数ライブラリをライブラリウィンドウ内に表示します。必要であれば、このリストを XQuery 関数を使用して、カスタム XQuery 1.0 ライブラリモジュールを MapForce にインポートして拡張することができます

MapForce にインポートするためには、XQuery ファイルは以下の条件を満たす必要があります:

- XQuery 仕様に従う有効なライブラリモジュールである必要があります。すなわち、`module namespace <prefix>="<namespace name」` などのモジュール宣言と共に開始する必要があります。
- インポートされたライブラリモジュール内で宣言された全ての関数は、動的なデータ型を返す必要があります(例えば、`xs:string`、`xs:boolean`、`xs:integer` など)。関数パラメーターも動的な型を有する必要があります。

XQuery ライブラリモジュールをインポートする:

1. ライブラリ ウィンドウのベースのライブラリの追加/削除 ボタンをクリックします。ライブラリの管理ウィンドウが開かれます。



2. 以下の内の1つを行います:

   - (現在のマッピング内のスコープのみで) *ローカル*ライブラリとして関数をインポートする場合現在のマッピング名の下の「**追加**」ボタンをクリックします。
   - (プログラムレベルで) *グローバル*ライブラリとして関数をインポートする場合 **グローバルライブラリのインポート** の横の「**追加**」ボタンをクリックします。

メモ ライブラリを*ローカル*でインポートする場合、ライブラリファイルのパスをマッピングに対して相対的に設定することが可能です。グローバルにインポートされたライブラリでは、インポートされたライブラリへのパスは常に絶対的なパスです。

3. .xq または.xquery ライブラリファイルを参照し「**開く**」をクリックします。

インポートされたライブラリもモジュールはライブラリウィンドウに表示され、特定の関数をマッピングエリアにドラッグし、他の関数コンポーネントと同様に使用することができます。関数をマッピングに追加するも参照してください。

インポート済みの XQuery ライブラリモジュールが表示されていない場合、XQuery が変換言語として選択されていることを確認してください。

次も参照:

XQuery エンジンの実装

7.5.1 例: カスタム XQuery 関数のインポート

このサンプルは、デモXQuery ライブラリモジュールをMapForce にインポートし、マッピングから関数を呼び出す方法について説明しています。このサンプルのデモモジュールは20%としての小数で示される税金を計算する関数により構成されています。生産のシナリオについては、XQuery モジュールには複数の関数が含まれている可能性があります。

> XQuery モジュール内で宣言されているすべての関数は、アトミックな型を返し、彼らのパラメーターもアトミックデータ型である必要があります。それ以外の場合、モジュールはMapForce にインポートすることはできません。

デモXQuery モジュールファイルをMapForce がインストールされているコンピューターのドキュメントフォルダーに対して相対的なパスで見つけることができます: <マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥Tutorial¥module.xq.

```
xquery version "1.0";

module namespace demo="http://www.altova.com/mapforce/demo";

declare function demo:calculatetax($val as xs:decimal) as xs:decimal {
   $val*0.2
};
```

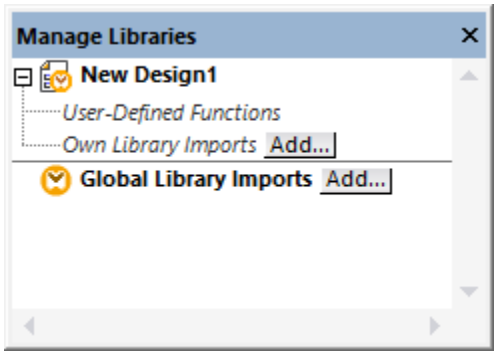
module.xq

MapForce にXQuery モジュールファイルをインポートした後、`demo:calculatetax` 関数をマッピングから呼び出せるようになります。XQuery 関数を使用して税金を計算することは、デモの目的です。同じ結果をMapForce ビルドイン関数を使用して達成することもできます。

`demo:calculatetax` 関数を呼び出すデモマッピングは、次のパスで見つけることができます:: <マイドキュメント>**\Altova\MapForce2021\MapForceExamples\Tutorial\CalculateTax_XQuery.mfd**。このマッピングを最初に開くと、MapForce はXQuery 内で使用することのできない1つまたは複数のコンポーネントが含まれていることを通知する警告を表示します。この警告は通常で、マッピングがまだインポートされていないカスタムXQuery ライブラリから関数をマッピングが参照すると発生します。この警告を削除し、マッピングを実行するには、以下の通り、不足するXQuery モジュールをMapForce にインポートします。

XQuery モジュールをMapForce にインポートする方法:

1. ライブラリ ウィンドウのベースの**ライブラリの追加/削除** ボタンをクリックします。ライブラリの管理ウィンドウが開かれます。



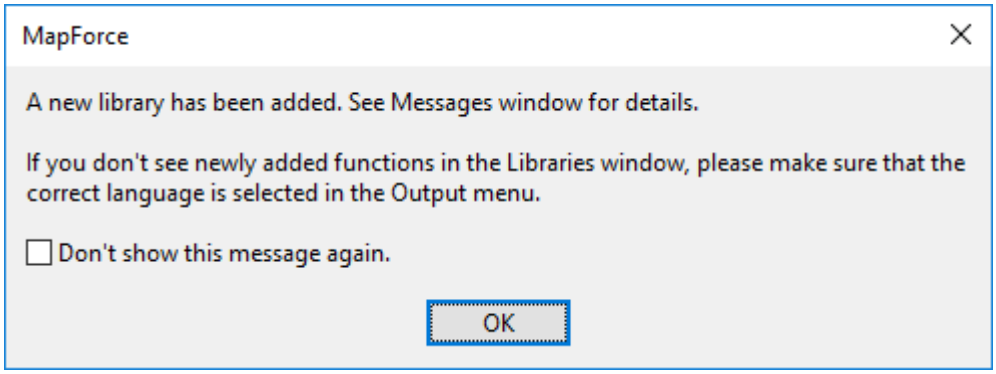
2. 以下の内の1つを行います:

   - (現在のマッピング内のスコープのみで) *ローカル*ライブラリとして関数をインポートする場合現在のマッピング名の下の「**追加**」

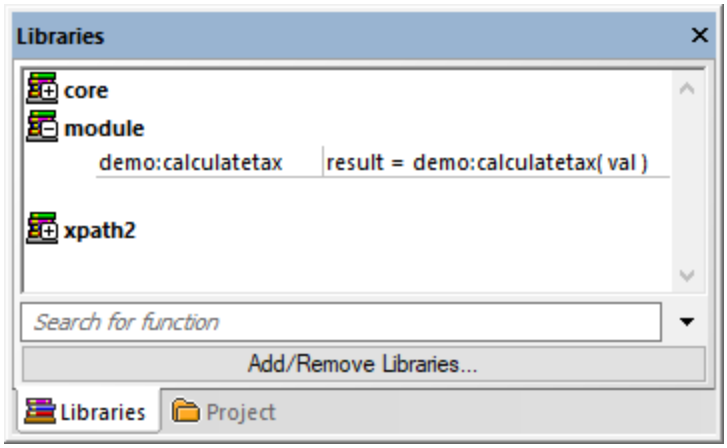
ボタンをクリックします。
- (プログラムレベルで) グローバルライブラリとして関数をインポートする場合 **グローバルライブラリのインポート** の横の「**追加**」ボタンをクリックします。

メモ　ライブラリをローカルでインポートする場合、ライブラリファイルのパスをマッピングに対して相対的に設定することが可能です。グローバルにインポートされたライブラリでは、インポートされたライブラリへのパスは常に絶対的なパスです。

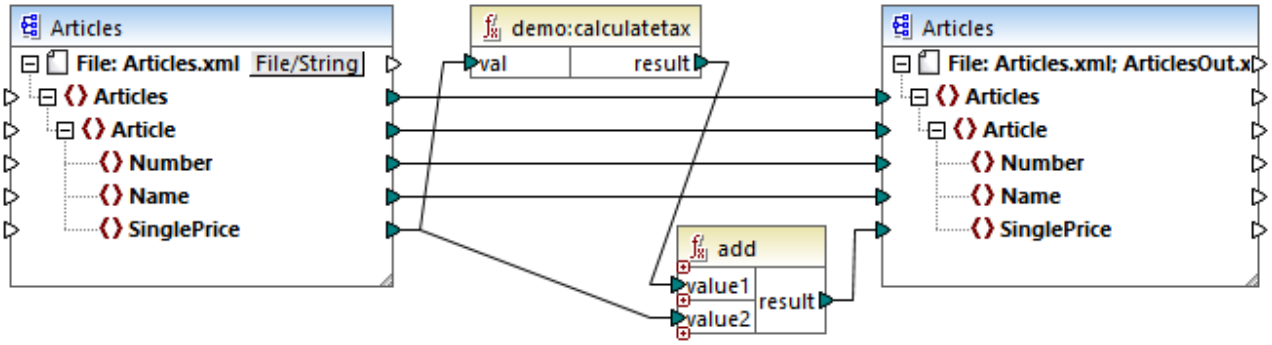
3. <マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥Tutorial¥module.xq を参照して「**開く**」をクリックします。メッセージボックスが表示され、新規のライブラリが追加されていることを通知します。



インポートされたライブラリと `demo:calculatetax` 関数はライブラリウィンドウで確認することができます。



また、マッピングは検証され、警告なしで実行されます示されている `demo:calculatetax` 関数はインポートされたXQuery モジュールから来ており、下のイメージで表示されています。また、他のビルトイン関数同様マッピングに追加することができます。マッピングに関数を追加するを参照して下さい。



CalculateTax_XQuery.mfd

マッピングの説明

CalculateTax_XQuery.mfd マッピングは、商品を保管するXMLファイルを入力としてとります。各商品には単一の価格が小数として表示されています。例：

```
<Articles>
   <Article>
      <Number>1</Number>
      <Name>T-Shirt</Name>
      <SinglePrice>25</SinglePrice>
   </Article>
   <Article>
      <Number>2</Number>
      <Name>Socks</Name>
      <SinglePrice>2.30</SinglePrice>
   </Article>
   <Article>
      <Number>3</Number>
      <Name>Pants</Name>
      <SinglePrice>34</SinglePrice>
   </Article>
   <Article>
      <Number>4</Number>
      <Name>Jacket</Name>
      <SinglePrice>57.50</SinglePrice>
   </Article>
</Articles>
```

Articles.xml

XMLファイルとして同じスキーマにつくXMLファイルをマッピングは作成します。このために、ソースとターゲットのコンポーネントは、マッピング上で同じ構成を持ちます。マッピングの接続が提案するとおり、ほとんどすべての要素が率直な方法でターゲットからソースにマップされます。例、ソース内の各 **Article** に対して、ターゲット内に**Article** が存在します。全てのアイテムは逐語的に、**SinglePrice** を除いてソースXMLにコピーされます。**SinglePrice** の値は2つの関数の助けを借りて計算されます:

- `demo:calculatetax` XQuery 関数は、基の**SinglePrice** を入力としてとり税金を計算します。
- MapForce ビルトイン `add` 関数は税金と基の**SinglePrice** 価格を足して最終の価格が返されます。

(C) 2015-2021 Altova GmbH

重要な点は 入力パラメーター型に対応し、XQuery関数の型を返す**SinglePrice** アイテムのデータ型がis `xs:decimal`ー であることです。

下記の **出力** タブをクリックすると、マッピングにより生成された出力が表示されます。ソースXML.に比べ各価格は20%の価格情報が適用されていることにご注意してください。

```
<?xml version="1.0" encoding="UTF-8"?>
<Articles xmlns:xsi="http://www.w3.org/2001/XMLSchema-
    <Article>
        <Number>1</Number>
        <Name>T-Shirt</Name>
        <SinglePrice>30</SinglePrice>
    </Article>
    <Article>
        <Number>2</Number>
        <Name>Socks</Name>
        <SinglePrice>2.76</SinglePrice>
    </Article>
    <Article>
        <Number>3</Number>
        <Name>Pants</Name>
        <SinglePrice>40.8</SinglePrice>
    </Article>
    <Article>
        <Number>4</Number>
        <Name>Jacket</Name>
        <SinglePrice>69</SinglePrice>
    </Article>
</Articles>
```

7.6 カスタム Java と .NET ライブラリのインポート

コンパイルされた Java クラスファイルや .NET アセンブリファイル(.NET 4.0 アセンブリを含む)はライブラリペインに追加して、MapForce にあるその他の関数と同じように使用することができます。これら Java や .NET 関数の出力は、出力ペインで確認することができるほか、関数は生成されたコードでも利用することができます。

メモ

- カスタム Java または .NET 関数をインポートするには、 Java クラス(.class) または .NET dll アセンブリファイルをコンパイルする必要があります。.NET アセンブリではない、Java .jar ファイルまたは .dll ファイルのインポートはサポートされていません。
- コンパイルされた Java クラス(.class) ファイルは、マッピング言語が Java に設定されているサポートされています。Java Runtime Environment 7 または以降が使用するマシンにインストールされている必要があります。特定の型とメンバーがサポートされています(以下を参照してください: Java 関数サポート)。
- .NET アセンブリファイルは、マッピング言語が C# に設定されているとサポートされています。パラメーターと戻り値の型としてシステムアセンブリから基本データ型 .NET アセンブリは、C# 以外の .NET 言語で書かれている可能性があります。(例えば、C++.NET または VB.NET)。のみを使用することを考慮すると、(以下を参照してください: .NET 関数サポート)。
- マッピングがインポートされた Java .class または .NET DLL アセンブリを使用する場合、マッピング言語を C++ に設定することは、サポートされていません。
- ネイティブの C++ DLL 関数をからインポートすることは、制限されており、特別なアプローチが必要です。詳細に関しては、以下を参照してください Java、C# と C++ ライブラリに手動で参照する
- マッピングがインポートされた Java .class または .NET DLL アセンブリを使用する場合、マッピング言語を XSLT に設定することは、サポートされていません。(アダプターの役割をするカスタム XSLT 関数が書き込まれる必要があります)。
- MapForce マッピングから呼び出される全ての関数が“べき等”である必要があります(これは、同じ入力パラメーターを使用して関数が呼び出される都度、同じ値が返されることを意味します)。 MapForce により呼び出される正確な順序および回数は定義されていません。
- Java の場合、インポートされたクラスファイルとパッケージは、CLASSPATH 変数に追加される必要はありません。これは、ビルトイン実行エンジンが、生成された Java コードと同様、Java エンジンのクラスパスまたは Ant の自動的にインポートされたパッケージに追加されるからです。しかしながら、インポートされたクラスファイルとパッケージの依存関係は、自動的には処理されません。ですから、インポートされた Java クラスファイルまたはパッケージが他のクラスファイルに依存する場合、CLASSPATH 環境変数の全ての依存パッケージの親ディレクトリが追加されていることを確認してください。

Java 関数サポート

トップレベルのクラス、静的メンバークラス、そして非静的メンバークラスがサポートされます::

- `new <classname>(<arg1>, <arg2>, ...)`
- `<object>.new <member-class>(<arg1>, <arg2>, ...)`

メンバー関数と静的関数がサポートされます:

- `<function>(<arg1>, <arg2>, ...)`
- `<object>.<method>(<arg1>, ...)`

XML スキーマと Java のデータ型間におけるサポートされている変換を以下に示します:

スキーマ型	Java 型
xs:string	String
xs:byte	byte
xs:short	short

スキーマ型	Java 型
xs:int	int
xs:long	long
xs:boolean	boolean
xs:float	float
xs:double	double
xs:decimal	java.math.BigDecimal
xs:integer	java.math.BigInteger

両方向への接続を行うことができます。その他の Java データ型はサポートされていません。配列型はサポートされていません。配列を使った引数や戻り値は無視されます。.

Object 型はコンストラクター呼び出しまたはメソッドの戻り値としてサポートされています。これらの型はその他の Java メソッドへマッピングすることもできます。MapForce にてオブジェクトを操作することはできません。

.NET 関数サポート

トップレベルのクラス、静的メンバークラス、そして非静的メンバークラスがサポートされます:

- `new <classname>(<arg1>, <arg2>, ...)`

メンバー関数と静的関数がサポートされます:

- `<function>(<arg1>, <arg2>, ...)`
- `<object>.<method>(<arg1>, ...)`

以下に示される XML スキーマ型と .NET/C# 型との間の接続がサポートされます:

スキーマ型	.NET 型	C# 型
xs:string	System.String	string
xs:byte	System.SByte	sbyte
xs:short	System.Int16	short
xs:int	System.Int32	int
xs:long	System.Int64	long
xs:unsignedByte	System.Byte	byte
xs:unsignedShort	System.UInt16	ushort
xs:unsignedInt	System.UInt32	uint
xs:unsignedLong	System.UInt64	ulong
xs:boolean	System.Boolean	bool

スキーマ型	.NET 型	C# 型
xs:float	System.Single	float
xs:double	System.Double	double
xs:decimal	System.Decimal	decimal

両方向への接続を行うことができます。その他の.NET/C# 型はサポートされていません。配列型はサポートされていません。配列を使った引数や戻り値は無視されます。

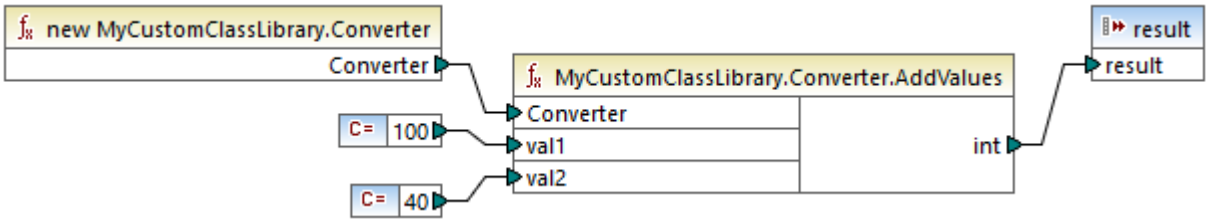
Object 型はコンストラクター呼び出しまたはメソッドの戻り値としてサポートされています。これらの型はその他の.NET メソッドへマッピングすることもできます。MapForce にてオブジェクトを操作することはできません。

データの型の問題と解決方法

カスタムライブラリ内の関数が整数型を期待する場合、型 "Number" の定数を関数の引数に接続すると以下に類似した型不一致のエラーを引き起こす場合があります:

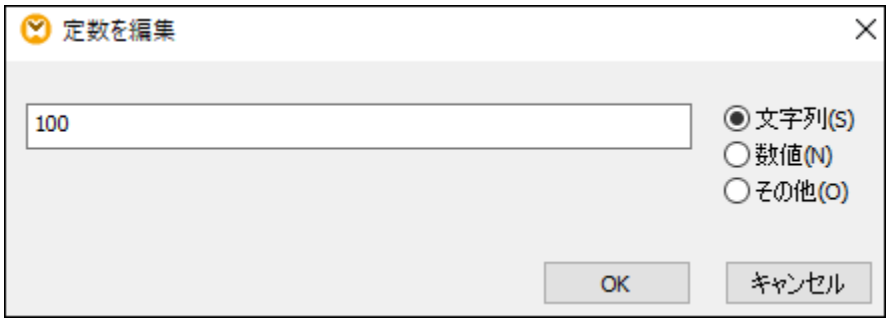
```
No match for MyCustomClassLibrary, Version=1.0.0.0, Culture=neutral,
PublicKeyToken=null.MyCustomClassLibrary.Converter.AddValues(MyCustomClassLibrary.Convert
er, xs:decimal, xs:decimal). Check argument types.
```

この問題は型 "Number" の定数に固有の問題です。このエラーを引き起こすマッピングの例は以下に示されています。このマッピングでは、型 "Number" の2つの定数は(型 "Integer" の)関数の引数に接続されています。



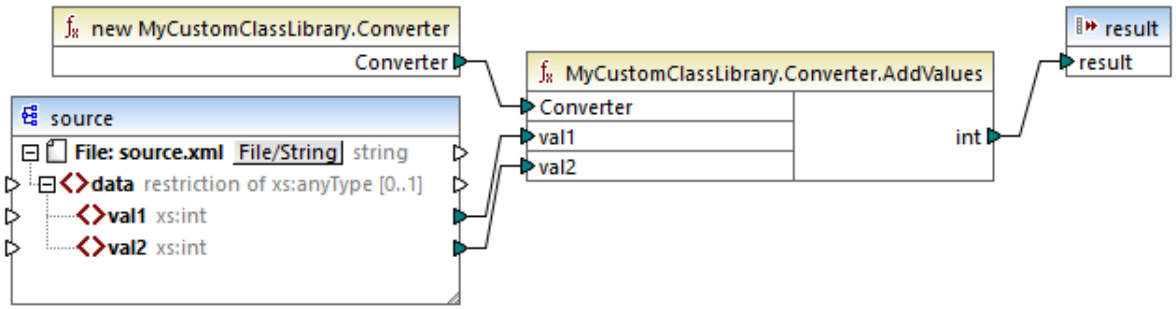
この問題にはいくつかの解決方法があります:

1. 変数を「数値」から「その他」に変更します(定数コンポーネントのタイトルバーをダブルクリックしてこれを行うことができます。)



2. 定数の代わりに関数により期待されるデータ型の値を提供する(例えば、XML ファイルなどの)ソースコンポーネントを使用します。

(C) 2015-2021 Altova GmbH



3. 外部コード内で、整数の値を返す小数の値を受け入れるラッパー関数を作成します。ラッパーソリューションは異なるライブラリとしてインポートされることができます。このため、ターゲット関数のオリジナルのソースコードをこのアプローチに変更する必要はありません。

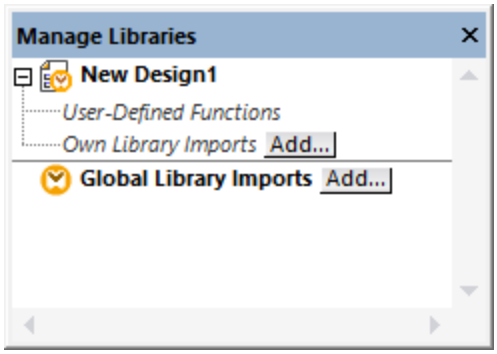
7.6.1 例: カスタム Java クラスのインポート

この例は、カスタム Java クラスファイルを MapForce にインポートする方法を説明しています。

メモ このサンプルを完了するには Java SE 7 Runtime Environment または以降が必須です。

変換言語を Java に設定します(次を参照してください:

1. ライブラリ ウィンドウのベースのライブラリの**追加/削除** ボタンをクリックします。ライブラリの管理ウィンドウが開かれます。

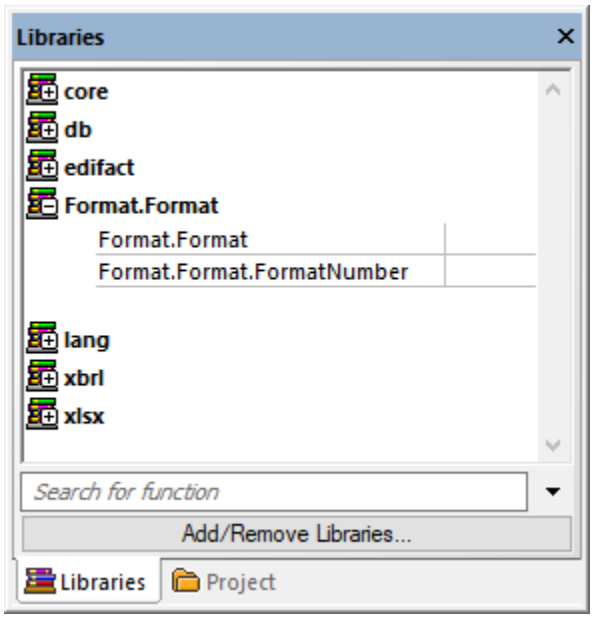


2. 以下の内の 1 つを行います:

   - (現在のマッピング内のスコープのみで) *ローカル* ライブラリとして関数をインポートする場合現在のマッピング名の下の「**追加**」ボタンをクリックします。
   - (プログラムレベルで) *グローバル* ライブラリとして関数をインポートする場合 **グローバルライブラリのインポート** の横の「**追加**」ボタンをクリックします。

メモ ライブラリを*ローカル*でインポートする場合、ライブラリファイルのパスをマッピングに対して相対的に設定することが可能です。グローバルにインポートされたライブラリでは、インポートされたライブラリへのパスは常に絶対的なパスです。

3. 以下のファイルを参照します: **<マイドキュメント>\Altova\MapForce2021\MapForceExamples\Java\Format\Format.class**。メッセージが表示され、新規のライブラリが追加されていることを通知します。インポートされたライブラリはライブラリウィンドウで確認することができます。



インポートされた新規のライブラリがライブラリウィンドウに表示されていない場合、変換言語がJava に設定されていることを確認してください。関数をマッピングに追加するにはライブラリウィンドウからマッピングにドラッグします。関数をマッピングに追加するも参照してください。

MapForce 内でマッピングの出力をプレビューする方法:

1. 次のマッピングを開きます: <マイドキュメント>**\Altova\MapForce2021\MapForceExamples\Java\FormatNumber.mfd**。これは上記のJava .class ライブラリをインポートする完全なマッピングです。
2. 出力ボタンをクリックして、マッピングの結果を確認します。

```
 1  Start date,End date,Region,Amount
 2  2008-01-01,2008-01-31,CA,"110.400,00"
 3  2008-01-01,2008-01-31,MA,"75.300,00"
 4  2008-02-01,2008-02-29,CA,"114.300,00"
 5  2008-02-01,2008-02-29,MA,"65.200,00"
 6  2008-03-01,2008-03-31,CA,"134.200,00"
 7  2008-03-01,2008-03-31,MA,"86.100,00"
 8  2008-04-01,2008-04-30,CA,"107.300,00"
 9  2008-04-01,2008-04-30,MA,"112.100,00"
10  2008-05-01,2008-05-31,CA,"114.400,00"
11  2008-05-01,2008-05-31,MA,"93.800,00"
```

Java 内でマッピングを実行する:

1. 「ファイル」メニューから「コードの生成 | Java」をクリックします。
2. コードが生成されるターゲットディレクトリを選択し、「OK」をクリックします。
3. 生成されたライブラリを Java プロジェクトにインポートし Java アプリケーションをビルドします 例: Java コードの生成と実行を参照してください。

(C) 2015-2021 Altova GmbH

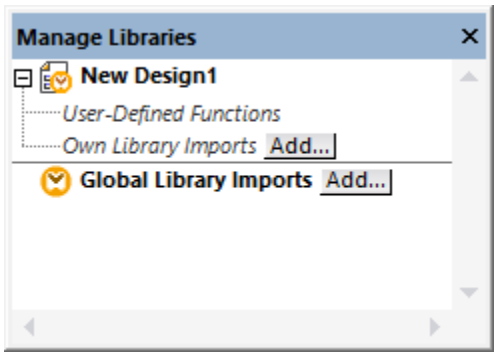
7.6.2 例: カスタム .NET DLL アセンブリのインポート

この例は、C# で作成された.NET DLL アセンブリを MapForce にインポートする方法を説明しています。このサンプルのソースコードは以下のパスで見つけることができます: <マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥C#¥Format。MapForceにインポートされる.dll アセンブリは**..\bin\Debug** ディレクトリ内にあります。Visual Studio 内でソリューション.sln ファイルを開き最新の.NET Framework バージョンを使用して新規の.dll ファイルをコンパイルすることができます。

メモ このサンプルを実行するには.NET Framework 2.0 または以降がコンピューターにインストールされている必要があります。有効化されていない場合 **コントロール パネル** > **プログラムと機能** > **Windows 機能の有効化または無効化** > .NET Framework <バージョン> から有効化することができます。

.NET アセンブリファイルをインポートする方法:

1. ライブラリ ウィンドウのベースの **ライブラリの追加/削除** ボタンをクリックします。ライブラリの管理ウィンドウが開かれます。

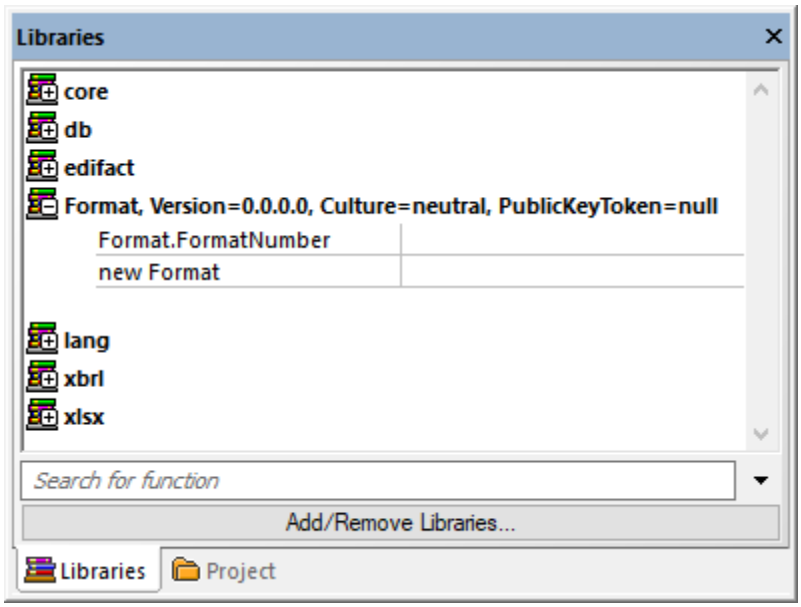


2. 以下の内の1つを行います:

   - (現在のマッピング内のスコープのみで) *ローカル*ライブラリとして関数をインポートする場合現在のマッピング名の下の「**追加**」ボタンをクリックします。
   - (プログラムレベルで) *グローバル*ライブラリとして関数をインポートする場合 **グローバルライブラリのインポート** の横の「**追加**」ボタンをクリックします。

メモ ライブラリを*ローカル*でインポートする場合、ライブラリファイルのパスをマッピングに対して相対的に設定することが可能です。グローバルにインポートされたライブラリでは、インポートされたライブラリへのパスは常に絶対的なパスです。

3. **Format.dll** ファイルを **...\MapForceExamples\C#\Format\bin\Debug** ディレクトリから参照します。メッセージが表示され、新規のライブラリが追加されていることを通知します。インポートされたライブラリはライブラリウィンドウで確認することができます。



インポートされた新規のライブラリがライブラリウィンドウに表示されていない場合、変換言語がC# に設定されていることを確認してください。関数をマッピングに追加するにはライブラリウィンドウからマッピングにドラッグします。関数をマッピングに追加するも参照してください。

マッピング出力をプレビューする方法:

1. **...\MapForceExamples\C#** フォルダー内にある **FormatNumber.mfd** ファイルを開きます。これは既に上記の .dll ライブラリをインポートしているサンプルマッピングです。
2. 出力ボタンをクリックして、マッピングの結果を確認します。

```
 1   Start date,End date,Region,Amount
 2   2008-01-01,2008-01-31,CA,"110.400,00"
 3   2008-01-01,2008-01-31,MA,"75.300,00"
 4   2008-02-01,2008-02-29,CA,"114.300,00"
 5   2008-02-01,2008-02-29,MA,"65.200,00"
 6   2008-03-01,2008-03-31,CA,"134.200,00"
 7   2008-03-01,2008-03-31,MA,"86.100,00"
 8   2008-04-01,2008-04-30,CA,"107.300,00"
 9   2008-04-01,2008-04-30,MA,"112.100,00"
10   2008-05-01,2008-05-31,CA,"114.400,00"
11   2008-05-01,2008-05-31,MA,"93.800,00"
```

カスタム C# アプリケーションからマッピングを実行する方法:

1. 「ファイル」メニューから「コードの生成 | C#」をクリックします.
2. コードが生成されるターゲットディレクトリを選択し、「OK」をクリックします。
3. Visual Studio を使用してアプリケーションをビルドし、コンソールアプリケーションを実行します（ C# コードの生成を参照してください）。

(C) 2015-2021 Altova GmbH

7.7 Java、C# および C++ ライブラリを手動で参照する

MapForce に直接カスタムライブラリをインポートする代替のオプションとしては、MapForce により識別されているカスタム .mff ファイル (MapForce 関数ファイル) を使用してレファレンスを作成することができます。mff ライブラリファイルは、カスタムコード内のクラス定義と MapForce のリンクを手動で定義する XML ファイルにより必須です。カスタム .mff ファイルを作成すると、.NET DLL または Java クラスファイルをインポートすることと同様 MapForce にインポートすることができます。

メモ

- MapForce にインポートするために関数を修飾するには、戻り値とパラメーターが単純型である必要があります。各言語のために使用することのできるデータ型のリストに関しては、データ型マッピングを参照してください。
- 関数ライブラリをカスタム .mff ファイルからインポートすると、(「出力」ボタンをクリックして) MapForce 内で直接マッピングをプレビューすることは制限されます。C++ で書かれているライブラリに関しては、MapForce 内のマッピングのプレビューは、サポートされていません。Java と C# の場合は、ライブラリがネイティブ言語の型を使用する場合は、プレビューを使用することができますが、ライブラリが Altova 生成されたクラスをインポートする場合は、使用することができません。しかしながら、ライブラリによりターゲットされている特定の言語でコードを生成することができます。カスタム関数は、生成されたコード内で使用することができ、生成されたコードからマッピングを実行することが可能になります。
- 生成されたマッピングコードにより呼び出される関数の順序は定義されていません。MapForce ではキャッシュされた計算結果を再利用するか、任意の順序で条件式を再評価する必要があります。ユーザー定義関数とカスタム関数ライブラリを区別する必要があります。
- ユーザー定義関数は、マッピング内で視覚的に作成され、マッピング .mfd ファイルと共に保存されるため *.mff ファイルには保存することができません。ユーザー定義関数は、マッピング内で視覚的に作成され、マッピング .mfd ファイルと共に保存されるため *.mff ファイルには保存することができません。詳細に関しては、以下を参照してください。詳細に関してはユーザー定義関数の呼び出しとインポートを参照してください。
- 2010 より以前の MapForce バージョンをアップグレードする場合、カスタム関数内で使用されているデータ型を更新する必要がある可能性があります。データ型マッピングを参照してください。

カスタム .mff ファイルの作成と構成の順序に関する詳細は、 .mff ファイルの構成方法を参照してください。例に関しては、以下を参照してください:

- 例: C# を .mff に適用する方法
- 例: C++ を .mff に適用する方法
- 例: Java を .mff に適用する方法

7.7.1 .mff ファイルの構成

MapForce 関数ファイル (.mff) は、MapForce にカスタム Java、C#、または C++ ライブラリから関数の適用を可能にし、ライブラリウインドウ内で表示される XML フォーマットの構成ファイルです。.mff ファイルは、基本的にカスタムライブラリと MapForce の媒介であり、(a) カスタム関数にインターフェイスし、また、(b) 生成されたコード内に実装が存在する箇所を指定するために構成されていなければなりません。このトピックは上記を行う方法について説明しています。

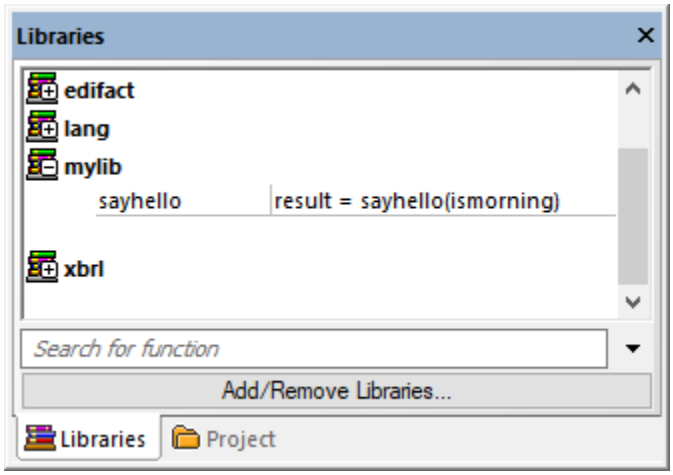
以下の点に注意してください:

- *.mff *.mff ライブラリファイルは以下のスキーマファイルに対して妥当である必要があります: **..\Program Files\MapForceLibraries\mff.xsd**。**mff.xsd** スキーマは内部的な使用のためだけのものです。Altova GmbH は、新規リリースにて、このファイルフォーマットを変更する権利を有しています。
- 1 つの C#、C++、または Java クラスを .mff ファイルごとに定義することが可能です。

以下のコードリストは C++ のためのサンプル .mff を説明しています。

```
<?xml version="1.0" encoding="UTF-8"?>
<mapping version="9" library="mylib" xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="mff.xsd">
   <implementations>
      <implementation language="cpp">
         <setting name="namespace" value="mylib"/>
         <setting name="class" value="Greetings"/>
         <setting name="path" value="C:\Libraries\cpp"/>
         <setting name="include" value="Greetings.h"/>
         <setting name="source" value="Greetings.cpp"/>
      </implementation>
   </implementations>
   <group name="greetings">
      <component name="sayhello">
         <sources>
            <datapoint name="ismorning" type="xs:boolean"/>
         </sources>
         <targets>
            <datapoint name="result" type="xs:string"/>
         </targets>
         <implementations>
            <implementation language="cpp">
               <function name="SayHello"/>
            </implementation>
         </implementations>
         <description>
            <short>result = sayhello(ismorning)</short>
            <long>Returns "Good morning" or "Good day", depending on the input
parameter.</long>
         </description>
      </component>
   </group>
</mapping>
```

mff ファイルがMapForce で表示される例が以下のイメージで示されています。新たなライブラリ"mylib" が"sayhello" 文字列関数を含む(アルファベット順に並べ替えられた)ライブラリエントリーに表示されます。



ニーズに合うかたちで mff ファイルを適応させるステップを以下に示します:

(C) 2015-2021 Altova GmbH

ライブラリ名の構成方法

以下に示される mff ファイルのセグメントにてライブラリ名が宣言されます。慣例として、MapForce 内ではライブラリ名は全て小文字で記述されますが、大文字も使用することができます。

```
<mapping version="9" library="mylib" xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="mff.xsd">
```

上のサンプルでは、ライブラリウィンドウに表示されるエントリは **mylib** と呼ばれます。

言語の実装の構成

<implementations> 要素はライブラリがサポートする言語を指定する必須要素であり、<mapping> の子として追加される必要があります。例:

```
<!-- ... -->
<mapping version="9" library="mylib" xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="mff.xsd">
   <implementations>
      <implementation language="cpp">
         <setting name="namespace" value="mylib"/>
         <setting name="class" value="Greetings"/>
         <setting name="path" value="C:\Libraries\cpp"/>
         <setting name="include" value="Greetings.h"/>
         <setting name="source" value="Greetings.cpp"/>
      </implementation>
   </implementations>
<!-- ... -->
```

各 **<implementation>** の設定により、生成されたコードから Java、C++、C# にて定義された関数を呼び出すことができるようになります。

.mff ファイルは、1 つ以上の言語をターゲットすることができるように作成することができます。この場合、追加言語ごとに追加 **<implementation>** 要素が含まれる必要があります。各プログラミング言語の設定を以下に示します。

Java ライブラリレファレンス

```
<!-- ...-->
<implementation language="java">
   <setting name="package" value="com.hello.functions"/>
   <setting name="class" value="Greetings"/>
</implementation>
<!-- ... -->
```

生成されたファイルが **Greetings.class** ファイルを検索できることは重要です。このためクラスへの参照は Java クラスパスに追加されていることを確認してください。

C# ライブラリレファレンス

```
<!-- ...-->
```

```
      <implementation language="cs">
         <setting name="namespace" value="MyLibrary" />
         <setting name="class" value="Greetings" />
         <setting name="reference" value="C:
\Libraries\cs\MyLibrary\bin\debug\MyLibrary.dll" />
      </implementation>
<!-- ... -->
```

C# に関しては、コード内の名前空間が、.mff ファイル内で定義されている名前空間に一致することが重要です。(上のコードリストでの名前空間は**MyLibrary** です)。クラス名に関しても同様です(上のコードリストでの名前は`Greetings` です)。第3の設定である、**reference** は、生成されたコードへリンクする dll のパスを提供します。

C++ ライブラリレファレンス

```
<!-- ...-->
   <implementation language="cpp">
         <setting name="namespace" value="MyLibrary"/>
         <setting name="class" value="Greetings"/>
         <setting name="path" value="C:\Libraries\cpp"/>
         <setting name="include" value="Greetings.h"/>
         <setting name="source" value="Greetings.cpp"/>
   </implementation>
<!-- ... -->
```

C++ のために以下に注意してください:

- **namespace** は、`Greetings` クラスが定義される名前空間です。**mapping** 要素内の**library** 属性に等しくある必要があります。
- **path** は、ソールファイルを見つけることができるパスです。
- マッピングのためのコードが生成されると、ソースファイルは「**ファイル | コードを生成 | C++**」メニューコマンドおよびディレクトリを選択する際に定義される) ディレクトリ**targetdir/libraryname** にコピーされ、プロジェクトファイルに含まれます

提供するファイルはすべて生成されるアルゴリズムに含まれます。

コンポーネントの追加

MapForce のライブラリウィンドウのグラフィカルユーザーインターフェイスでは、各関数は、関数グループの下にネストされ表示されます。例えば、「文字列関数」など。.mff ファイル内では、関数は`<component>` 要素に対応します。一方、各 `<component>` は、`<group>` 要素の下でネストされる必要があります。例:

```
<!-- ...-->
<group name="string functions">
   <component name="sayhello">
   <!-- ... -->
   </component>
</group>
<!-- ... -->
```

下に表示されるコードは、**sayhello** と呼ばれるサンプル関数(コンポーネント)を定義しています。

```
<!-- ...-->
<component name="sayhello">
   <sources>
```

```
      <datapoint name="ismorning" type="xs:boolean"/>
   </sources>
   <targets>
      <datapoint name="result" type="xs:string"/>
   </targets>
   <implementations>
   <!-- ... -->
   </implementations>
   <description>
     <short>result = sayhello(ismorning)</short>
     <long>Returns "Good morning" or "Good day", depending on the input
parameter.</long>
   </description>
</component>
<!-- ... -->
```

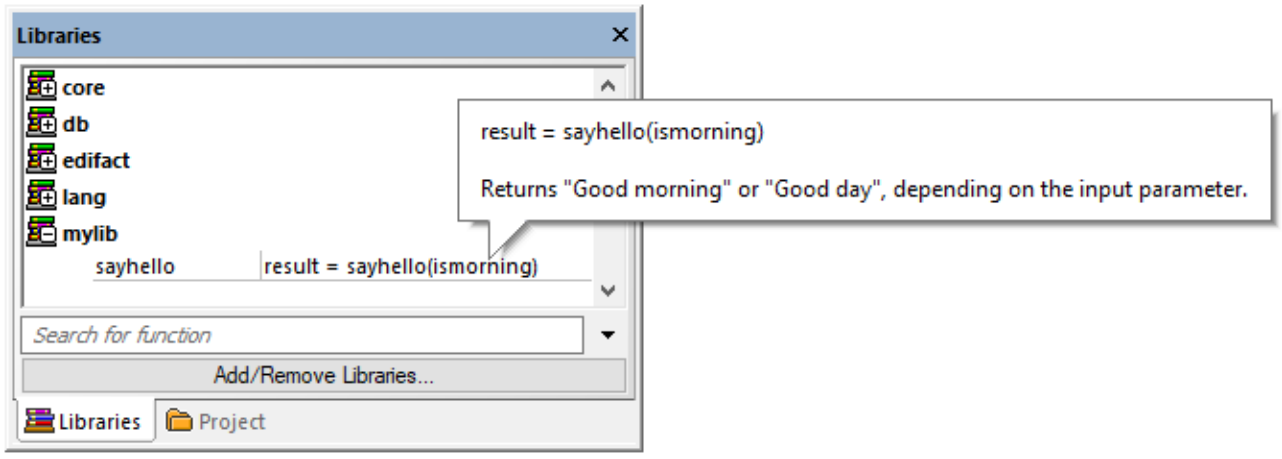
MapForce 内でコンポーネントは以下のように表示されます:

上のコードリスティングでは、<datapoint> は、大まかに入力または出力コネクタとして知られる関数の入力または出力パラメーターとして定義することができます。<datapoint> の**type** 引数は、パラメーターのデータ型（または、戻り値のデータ型）を指定します。

各関数のために、ターゲットデータポイントのみが許可されています。定義することのできるソースデータポイントの数に制限はありません。

各データポイントのデータ型は、XML スキーマ型（例えば、xs:string、xs:integer など）の 1 つである必要があります。これらのデータ型は、Java、C++ または C# ライブラリ内出定義された関数のパラメーターの型のデータ型に対応します。XML スキーマデータ型を言語型にマッピングするには、データ型のマッピングを参照してください。

ライブラリウィンドウ内では関数は短い、および、長い説明と共に表示されています。短い説明は常に、関数名の右側に表示されますが、長い説明は、マウスを短い説明の上にポイントするとヒントとして表示されます。



言語の実装の定義

ライブラリウィンドウ内でカスタム Java、C# または C++ クラスの関数間の接続を作成します。これは、**<implementation>** 要素を使用することにより達成することができます。

上記の通り、1 つの関数が、サポートされるプログラミング言語のために 1 つずつ、複数の実装要素を持つ場合があります。関数は、Java 内では、"Hello" と呼ばれ、C++ 内では "SayHello" と呼ばれる場合があります。このため、各プログラミング言語のために個別の関数名を指定する必要があります。3 つのプログラミング言語のための関数は以下のようになります:

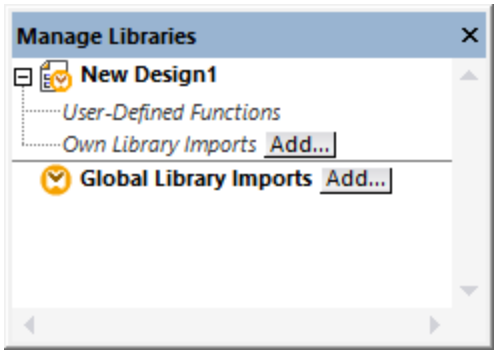
```
<!-- ...-->
<component name="sayhello">
<!-- ... -->
   <implementations>
      <implementation language="cs">
         <function name="HelloFunction"/>
      </implementation>
      <implementation language="java">
         <function name="Hello"/>
      </implementation>
      <implementation language="cpp">
         <function name="SayHello"/>
      </implementation>
   </implementations>
<!-- ... -->
</component>
<!-- ... -->
```

関数名として提供される値は、Java、C# または C++ クラス内のメソッドの名前に正確に一致する必要があります。

7.7.2 .mff ファイルを MapForce にインポートする

カスタム .mff ファイルの作成後 MapForce に以下のようにインポートすることができます:

1. ライブラリ ウィンドウのベースのライブラリの追加/削除 ボタンをクリックします。ライブラリの管理ウィンドウが開かれます。



2. 以下の内の 1 つを行います:

   - (現在のマッピング内のスコープのみで) *ローカル* ライブラリとして関数をインポートする場合現在のマッピング名の下の「**追加**」ボタンをクリックします。
   - (プログラムレベルで) *グローバル* ライブラリとして関数をインポートする場合 **グローバルライブラリのインポート** の横の「**追加**」

ボタンをクリックします。

メモ ライブラリをローカルでインポートする場合、ライブラリファイルのパスをマッピングに対して相対的に設定することが可能です。グローバルにインポートされたライブラリでは、インポートされたライブラリへのパスは常に絶対的なパスです。

3. カスタム.mff ファイルを参照し「開く」をクリックします。

マッピング言語をカスタムライブラリによりターゲットにされた言語に設定すると、インポートされたライブラリがライブラリウィンドウ内で表示されます。

*.mff ファイルを プログラムファイル (または プログラムファイル (x86) フォルダー) に相対する..
\Altova\MapForce2021\MapForceLibraries フォルダーに保存すると、MapForce の起動時にライブラリは自動的にライブラリウィンドウにロードされます。対応するライブラリファイル (*.mff) を削除または追加することにより、ライブラリとその関数の表示の切り替えを行うことができます。

7.7.3 データ型のマッピング

次のテーブルは MapForce に Java、C#、および C++ ライブラリを適応する、カスタム.mff ファイルを作成する際に関数戻り値の型とパラメーターの型としてサポートされるデータ型をリストしています。テーブルは、ネイティブとネイティブでは内データ型をリストしています。Altova date などのネイティブでは内データ型をサポートする場合、カスタム Java と C# ライブラリは Altova ライブラリへのレファレンスを含む必要があります。C++ の場合、Altova ライブラリは常にインポートされる必要があります。Altova ライブラリの生成方法に関しての詳細は、次を参照してください: コード ジェネレーター。

XML スキーマ型	Java 型	C# 型	C++ 型
anyAtomicType	String	string	string_type
anySimpleType	String	string	string_type
anyURI	String	string	string_type
base64Binary	byte[]	byte[]	altova::mapforce::blob
boolean	boolean	bool	bool
byte	int	int	int
date	com.altova.types.DateTime	Altova.Types.DateTime	altova::DateTime
dateTime	com.altova.types.DateTime	Altova.Types.DateTime	altova::DateTime
dayTimeDuration	com.altova.types.Duration	Altova.Types.Duration	altova::Duration
decimal	java.math.BigDecimal	decimal	double
double	double	double	double
duration	com.altova.types.Duration	Altova.Types.Duration	altova::Duration
ENTITIES	String	string	string_type

XML スキーマ型	Java 型	C# 型	C++ 型
ENTITY	String	string	string_type
float	double	double	double
gDay	com.altova.types.DateTime	Altova.Types.DateTime	altova::DateTime
gMonth	com.altova.types.DateTime	Altova.Types.DateTime	altova::DateTime
gMonthDay	com.altova.types.DateTime	Altova.Types.DateTime	altova::DateTime
gYear	com.altova.types.DateTime	Altova.Types.DateTime	altova::DateTime
gYearMonth	com.altova.types.DateTime	Altova.Types.DateTime	altova::DateTime
hexBinary	byte[]	byte[]	altova::mapforce::blob
ID	String	string	string_type
IDREF	String	string	string_type
IDREFS	String	string	string_type
int	int	int	int
integer	java.math.BigInteger	decimal	__int64
language	String	string	string_type
long	long	long	__int64
Name	String	string	string_type
NCName	String	string	string_type
negativeInteger	java.math.BigInteger	decimal	__int64
NMTOKEN	String	string	string_type
NMTOKENS	String	string	string_type
nonNegativeInteger	java.math.BigInteger	decimal	unsigned __int64
nonPositiveInteger	java.math.BigInteger	decimal	__int64
normalizedString	String	string	string_type
NOTATION	String	string	string_type
positiveInteger	java.math.BigInteger	decimal	unsigned __int64

(C) 2015-2021 Altova GmbH

XML スキーマ型	Java 型	C# 型	C++ 型
QName	javax.xml.namespace.QName	Altova.Types.QName	altova::QName
short	int	int	int
string	String	string	string_type
time	com.altova.types.DateTime	Altova.Types.DateTime	altova::DateTime
token	String	string	string_type
unsignedByte	long	ulong	unsigned __int64
unsignedInt	long	ulong	unsigned __int64
unsignedLong	java.math.BigInteger	ulong	unsigned __int64
unsignedShort	long	ulong	unsigned __int64
untypedAtomic	String	string	string_type
yearMonthDuration	com.altova.types.Duration	Altova.Types.Duration	altova::Duration

7.7.4 例: C# を .mff に適用する方法

このサンプルはサンプル C# ライブラリを作成し MapForce ライブラリとしてインポート可能な MapForce 関数ファイル(.mff) に適用する方法について説明しています。

メモ C# ライブラリの .mff 書式への適用は C# ライブラリを MapForce にインポートする多くの方法の一つです。簡単な代替法は .NET アセンブリを直接インポートすることです。例: カスタム .NET DLL アセンブリのインポートを参照してください。

構成プロセスは以下のステップにより構成されています:

1. Visual Studio 内に新規クラスライブラリを作成します。関数は、`public static` として定義されていることに注意してください。

```
namespace MyLibrary
{
    public class Greetings
    {
        public static string SayHello(bool isMorning)
        {
            if (isMorning)
                return "Good morning!";
            return "Good Day!";
        }
    }
}
```

2. 特別な XML スキーマ型 (日付および期間) が必要な場合は Visual Studio プロジェクトから **Altova.dll** ライブラリにレファレンスを追加します。このライブラリを取得するには、カスタム関数を使用しないマッピングから C# コードを生成します。**Altova.dll**

ファイルはコードが生成されたディレクトリに相対する **..\Altova\bin\debug** ディレクトリ内にあります。Visual Studio 内の **Altova.dll** にレファレンスを追加するには、「**プロジェクト**」メニューから「**レファレンスを追加**」をクリックして、**Altova.dll** ファイルを参照します。また、コードに以下のラインを追加してください: `using Altova.Types;` 。XML スキーマ型を C# 型にマップする方法に関する情報はデータ型のマッピングを参照してください。

3. Visual Studio プロジェクトを構築します。**MyLibrary.dll** がプロジェクト出力ディレクトリに生成されます。
4. XML エディターを使用して、新規の .mff ファイルを作成し、**..¥Program Files¥MapForceLibraries¥mff.xsd** スキーマに対して検証します。`implementation language="cs"` 下のすべての参照が正確な C# メンバーと前に作成されているパスを指していることを確認してください。また `function name="SayHello"` は C# 内で定義されている通りの関数名を参照している必要があります。.mff ファイルの構成を参照してください。

```
<?xml version="1.0" encoding="UTF-8"?>
<mapping version="9" library="mylib" xmlns:xs="http://www.w3.org/2001/XMLSchema"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="mff.xsd">
   <implementations>
      <implementation language="cs">
         <setting name="namespace" value="MyLibrary" />
         <setting name="class" value="Greetings" />
         <setting name="reference" value="C:
\Libraries\cs\MyLibrary\bin\debug\MyLibrary.dll" />
      </implementation>
   </implementations>
   <group name="string functions">
      <component name="sayhello">
         <sources>
            <datapoint name="ismorning" type="xs:boolean"/>
         </sources>
         <targets>
            <datapoint name="result" type="xs:string"/>
         </targets>
         <implementations>
            <implementation language="cs">
               <function name="SayHello"/>
            </implementation>
         </implementations>
         <description>
            <short>result = sayhello(ismorning)</short>
            <long>Returns "Good morning" or "Good day", depending on the input
parameter.</long>
         </description>
      </component>
   </group>
</mapping>
```

MapForce に適応するカスタムライブラリと .mff ファイルの作成が完了しました。カスタム .mff ファイルを MapForce で使用することができるようになりました。.mff ファイルを MapForce にインポートするを参照してください。

7.7.5 例: C++ を .mff に適用する方法

このサンプルはサンプル C++ ライブラリを作成し MapForce ライブラリとしてインポート可能な MapForce 関数ファイル(.mff) に適用する方法について説明しています。

1. ヘッダー(.h) ファイルをクラスライブラリのために作成します。以下のコードリストは **Greetings.h** という名前のサンプルヘッダーファイルを表しています。

(C) 2015-2021 Altova GmbH

```
#ifndef MYLIBRARY_GREETINGS_H_INCLUDED
#define MYLIBRARY_GREETINGS_H_INCLUDED

#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000

using namespace altova;

namespace mylib {

class ALTOVA_DECLSPECIFIER Greetings
{
   public:
    static string_type SayHello(bool isMorning);
};

} // namespace mylib

#endif // MYLIBRARY_GREETINGS_H_INCLUDED
```

関数は静的として宣言されており、名前空間 `altova` がインポートされています。`ALTOVA_DECLSPECIFIER` をクラス名の前に書き込んでください。これにより、動的または静的なリンケージの使用に関わらず、次に生成されるコード内でクラスが正確にコンパイルされます。

2. .cpp ヘッダーファイルと同じ名前のファイルを作成します。.cpp ファイルは、.h ファイルと同じディレクトリに存在する必要があります。次のコード リスティングは、前に作成された **Greetings.h** ファイルを含む、**Greetings.cpp** と呼ばれるサンプル .cpp ファイルを示しています:

```
#include "StdAfx.h"
#include "../Altova/Altova.h"
#include "../Altova/AltovaException.h"
#include "../Altova/SchemaTypes.h"

#include "Greetings.h"

namespace mylib {

   string_type Greetings::SayHello(bool isMorning)
   {
      if( isMorning )
         return _T("Good morning!");
      return _T("Good day!");
   }
}
```

StdAfx.h と一部の Altova ライブラリをインポートするラインに注意してください。これらのラインは変更されてはなりません。ライブラリへのパスは正確です。生成されたコード内では、これらのパスはそれぞれのファイルを指します。

> Java または C# と対象的に、ソース C++ ファイルをコンパイルする必要はありません。ファイルは生成されたコードにコピーされ、生成されたマッピングコードの残りと共にコンパイルされます。

3. XML エディターを使用して、新規 .mff ファイルを作成し、**..\Program Files\MapForceLibraries\mff.xsd** スキーマに対して検証します。.mff ファイルの構成方法で説明されている通り名前空間、関数名、データ型が内の C++ コードのものに対応していることを確認してください。データ型へのサポートに関する詳細はデータ型のマッピングを参照してください。

```
<?xml version="1.0" encoding="UTF-8"?>
<mapping version="9" library="mylib" xmlns:xs="http://www.w3.org/2001/XMLSchema"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="mff.xsd">
   <implementations>
      <implementation language="cpp">
         <setting name="namespace" value="mylib"/>
         <setting name="class" value="Greetings"/>
         <setting name="path" value="C:\Libraries\cpp"/>
         <setting name="include" value="Greetings.h"/>
         <setting name="source" value="Greetings.cpp"/>
      </implementation>
   </implementations>
   <group name="greetings">
      <component name="sayhello">
         <sources>
            <datapoint name="ismorning" type="xs:boolean"/>
         </sources>
         <targets>
            <datapoint name="result" type="xs:string"/>
         </targets>
         <implementations>
            <implementation language="cpp">
               <function name="SayHello"/>
            </implementation>
         </implementations>
         <description>
            <short>result = sayhello(ismorning)</short>
            <long>Returns "Good morning" or "Good day", depending on the input
parameter.</long>
         </description>
      </component>
   </group>
</mapping>
```

MapForce に適応するカスタムライブラリと .mff ファイルの作成が完了しました。カスタム .mff ファイルをライブラリとして MapForce にインポートする準備が整いました。 .mff ライブラリのインポートを参照してください。C++ コードの生成で説明されているとおり、ネイティブの C++ ライブラリを使用するマッピングを実行するには、C++ コードを生成し、C++ コードまたはアプリケーションから実行する必要があります。

C++ コンパイルエラーが発生した場合

以下のラインでコンパイラーエラーが発生した場合:

```
#import "msado15.dll" rename("EOF", "EndOfFile")
```

msado15.dll ファイルへの参照を含むプロジェクトプロパティを変更します（通常は **C:\Program Files\Common Files\System\ADO**）。

7.7.6 例: Java を .mff に適用する方法

このサンプルはサンプル Java ライブラリを作成し MapForce ライブラリとしてインポート可能な MapForce 関数ファイル（.mff）に適用する方法について説明しています。

(C) 2015-2021 Altova GmbH

メモ Java ライブラリの .mff 書式への適用は Java ライブラリを MapForce にインポートする多くの方法の一つです。簡単な代替法は Java .class ファイルを直接インポートすることです。例: カスタム Java クラスのインポートを参照してください。

構成プロセスは以下のステップにより構成されています:

1. (例えば、Eclipse などの) 優先する開発環境で新規の Java プロジェクトを作成します。
2. `Greetings` と呼ばれるクラスから構成される `com.mylib` と呼ばれる新規のパッケージをプロジェクトに追加します。下のコードリスティングでは、`SayHello` 関数は、`public static` として定義されていることに注意してください。

```
package com.mylib;

public class Greetings {

   public static String SayHello ( boolean isMorning ) {
      if( isMorning )
         return "Good Morning!";
      return "Good Day!";
   }

}
```

3. 任意で、 日付、時間、または期間などの特別なスキーマ型をサポートする必要がある場合、 com.altova.types パッケージをインポートしてください。このパッケージを取得するには、Java コードをカスタム関数を持たないマッピングから生成してください。

```
import com.altova.types.*;
```

4. クラスファイルへカスタムライブラリをコンパイルし、Java クラスパスに追加します。
5. XML エディターを使用して、新規 .mff ファイルを作成し、**..\Program Files\MapForceLibraries\mff.xsd** スキーマに対して検証します。`implementation language="java"` 下のすべての参照が正確な Java メンバーと前に作成されているパスを指していることを確認してください。また `function name="SayHello"` は Java 内で定義されている通りの関数名を参照している必要があります。.mff ファイルの構成 を参照してください。

```
<?xml version="1.0" encoding="UTF-8"?>
<mapping version="9" library="custom"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="mff.xsd">
   <implementations>
      <implementation language="java">
         <setting name="package" value="com.mylib"/>
         <setting name="class" value="Greetings"/>
      </implementation>
   </implementations>
   <group name="greetings">
      <component name="sayhello">
         <sources>
            <datapoint name="ismorning" type="xs:boolean"/>
         </sources>
         <targets>
            <datapoint name="result" type="xs:string"/>
         </targets>
         <implementations>
            <implementation language="java">
               <function name="SayHello"/>
            </implementation>
```

```
         </implementations>
         <description>
            <short>result = sayhello(ismorning)</short>
            <long>Returns "Good morning" or "Good day", depending on the input
parameter.</long>
         </description>
      </component>
   </group>
</mapping>
```

MapForce に適応する Java ライブラリと .mff ファイルの作成が完了しました。カスタム .mff ファイルをライブラリとして MapForce にインポートする準備が整いました。 .mff ライブラリのインポートを参照してください。

(C) 2015-2021 Altova GmbH

7.8 関数ライブラリの管理

MapForce では以下のライブラリの種類をインポートしマッピング内で使用することができます:

- ユーザー定義関数 (UDF) を含むマッピングデザインファイル(*.mfd)。これはビルドイン関数を使用しコンポーネントをビルディングブロックとして使用する MapForce を使用して作成されたユーザー定義関数を含むマッピングファイルを具体的に指しています。詳細に関しては、ユーザー定義関数の作成 を参照してください。
- 関数を含む XSLT ファイル。カスタム XSLT 関数のインポート 内で説明されているとおり MapForce にインポートするために MapForce 外部で作成された XSLT 関数を指しています。
- 関数を含むカスタム XQuery 1.0 ファイル。カスタム XSLT 1.0 関数のインポート 内で説明されているとおり MapForce にインポートするために MapForce 外部で作成された XQuery 関数を指しています。
- カスタム Java と .NET ライブラリのインポート 内で説明されている通りにインポートするための Java .class ファイルと .NET .dll ライブラリ。

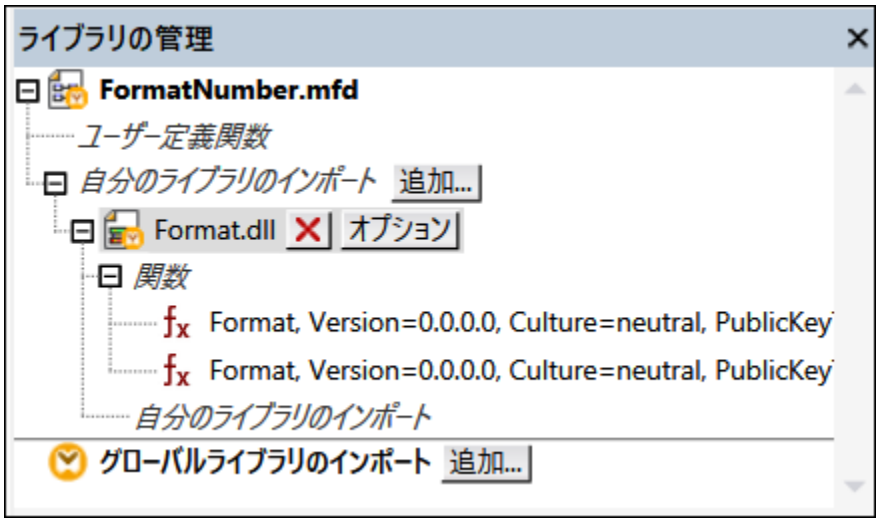
メモ (構成を必要とすることなく) 関数のカスタム外部ライブラリを直接、または MapForce により認識されている MFF (MapForce 関数ファイル) を構成することによりインポートすることができます。後者のアプローチを使用する場合 Java クラスと .NET アセンブリに加え C++ ライブラリをインポートすることができます。.mff ファイルによりインポートされたライブラリは Java、C# と C++ ライブラリを手動で参照する方法 で説明されている必要条件を満たす必要があります。

ライブラリの管理ウィンドウ

マッピングファイルにより使用されたすべてのライブラリをライブラリの管理ウィンドウからビューして管理することができます。これには UDF とカスタムライブラリも含まれています。

デフォルトではライブラリの管理ウィンドウは表示されていません。表示するには以下の一つを行うことができます:

- 「表示」メニューから「ライブラリの管理」をクリックします。
- ライブラリ ウィンドウのベースの ライブラリの追加/削除 をクリックします。



UDF とライブラリを現在アクティブなマッピングのためにのみ、またはすべての開かれているマッピングのために表示することを選択できます。現在開かれているすべてのマッピングのためにインポートされた関数とライブラリを表示するには、ウィンドウ内を右クリックし、コンテキストメニューから「開かれているドキュメントの表示」選択します。

名前の代わりに開かれているマッピングのパスを表示するには、コンテキストメニューから「ファイルパスの表示」を選択します。

ライブラリの管理ウィンドウ内に表示されているデータはツリー階層に以下のように整理されています:

- 現在開かれているマッピングドキュメントはトップレベルエントリとして表示されています。各エントリには2つのブランチが存在します: **ユーザー定義関数** と **自身のライブラリインポート**。
  - **ユーザー定義関数** ブランチはそのドキュメント内に含まれているユーザー定義関数を表示しています。
  - **自身のライブラリのインポート** ブランチは現在のマッピングドキュメントに *ローカルで* インポートされたライブラリを表示しています。「ライブラリ」という用語は他のマッピングドキュメント(ユーザー定義関数を含む.mfd) またはXSLT 1,0 XSLT 2.0、XQuery 1.0*、Java*、C#* で作成されたカスタム外部ライブラリ、または前述の.mff ファイルを指します。マッピングは他のマッピングドキュメントをライブラリとしてインポートする可能性があるため **自身のライブラリのインポート** 構造は複数のレベルの深さになります。
- **グローバルライブラリのインポート** エントリはアプリケーションレベルで *グローバルに* インポートしたカスタムライブラリを含みます。.mfd ファイルの場合、構造は上記の理由から複数のレベルの深さになります。

** MapForce Professional または Enterprise エディション内でのみこれらの言語はサポートされています*

メモ XSLT、XQuery、C#、および Java ライブラリは自身の依存関係を持つ場合があります。このような依存関係はライブラリウィンドウに表示されません。

コンテキストメニューコマンド

オブジェクトを右クリックし、次のコンテキストメニューオプションを選択することにより、ライブラリの管理ウィンドウ内のオブジェクトに対して多種の操作を行うことができます:

コマンド	説明	適用可能範囲
開く	マッピングを開きます。	マッピング
追加	関数のカスタムライブラリを参照するダイアログボックスを開きます。	自身のライブラリのインポート
ライブラリウィンドウ内で関数をロケートする	ライブラリウィンドウのフォーカスを変更し、関数を選択します。	関数
切り取り、コピー、削除	これらの標準 Windows コマンドは MapForce ユーザー定義関数に対してのみ適用が可能です。外部 XSLT ファイルまたは他のライブラリの種類から関数をコピーして貼り付けることはできません。	ユーザー定義関数
貼り付け	以前にクリップボードにコピーされたユーザー定義関数を現在のライブラリに貼り付けることができます。	ライブラリ(UDF)
オプション	現在のライブラリのためにオプションを設定または変更することができるダイアログボックスを開きます。	ライブラリ
開かれているすべてのドキュメントを表示する	このオプションがオンに切り替えられていると、ライブラリの管理ウィンドウは現在開かれているすべてのマッピングを開きます。マッピング間で関数をコピーして貼り付ける場合この機能は通常役に立ちます。それ以外の場合、現在焦点の当たっているマッピングのみが表示されます。	常に
Windows ファイルパスの表示	このオプションがオンに切り替えられていると、ライブラリの管理ウィンドウ内のオブジェクトがフルパスと共に表示されています。それ以外の場合、オブジェクト名のみが表示されています。	常に

(C) 2015-2021 Altova GmbH

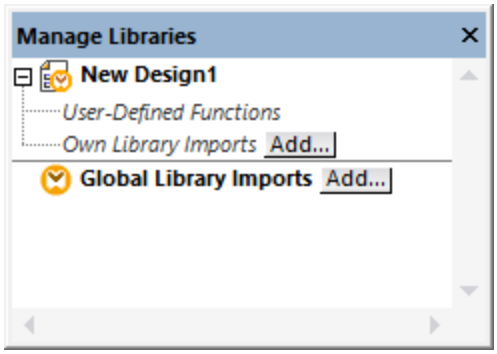
7.8.1 ローカルとグローバルライブラリ

ライブラリを *ローカル* または *グローバル* にインポートすることができます。グローバルなインポートはアプリケーションレベルです。ライブラリがグローバルにインポートされると、全てのマッピングからの関数を使用することができます。

ローカルなインポートはマッピングファイルレベルです。例えば、マッピング **A.mfd** で作業中にマッピング **B.mfd** からすべてのユーザー定義関数をインポートすると仮定します。この場合マッピング **B.mfd** は **A.mfd** にローカルなライブラリとしてインポートされると考えられ、**A.mfd** 内の **B.mfd** からも関数を使用することができます。同様にXSLT ファイルから関数を **A.mfd** にインポートすると、これもローカルなインポートと考えられます。

ライブラリの管理ウィンドウからすべてのローカルとグローバルのインポートを確認し管理することができます。ライブラリをインポートするには、以下を行います:

1. ライブラリ ウィンドウのベースの **ライブラリの追加/削除** ボタンをクリックします。ライブラリの管理ウィンドウが開かれます。



2. 以下の内の1つを行います:

   - (現在のマッピング内のスコープのみで) *ローカル* ライブラリとして関数をインポートする場合現在のマッピング名の下の「**追加**」ボタンをクリックします。
   - (プログラムレベルで) *グローバル* ライブラリとして関数をインポートする場合 **グローバルライブラリのインポート** の横の「**追加**」ボタンをクリックします。

メモ ライブラリを*ローカル*でインポートする場合、ライブラリファイルのパスをマッピングに対して相対的に設定することが可能です。グローバルにインポートされたライブラリでは、インポートされたライブラリへのパスは常に絶対的なパスです。

競合する関数名

以下のレベルで同じ名前を持つ関数が発生するシチュエーションに遭遇する可能性があります:

- メインマッピング内
- ライブラリがローカルでインポートとされている場合
- ライブラリがグローバルでインポートとされている場合

このようなシチュエーションに遭遇した場合、曖昧さを回避するためにMapForce は上記の順序で関数を呼び出す試みを行います。すなわち、ローカルでインポートされたライブラリ内に同じ名前の関数が存在する場合、マッピング内で直接定義されている関数は前例を取ります。また、(両方の関数が同じ名前を持つことを想定して)ローカルでインポートされている関数はグローバルでインポートされている関数よりも前例を優先します。

同じ名前が存在する複数の関数では、上記のルールに従い「優位な」関数が呼び出されます。その他の曖昧な関数名はブロックされます。このようなブロック関数はライブラリウィンドウ内で灰色表示され、マッピング内で使用することはできません。

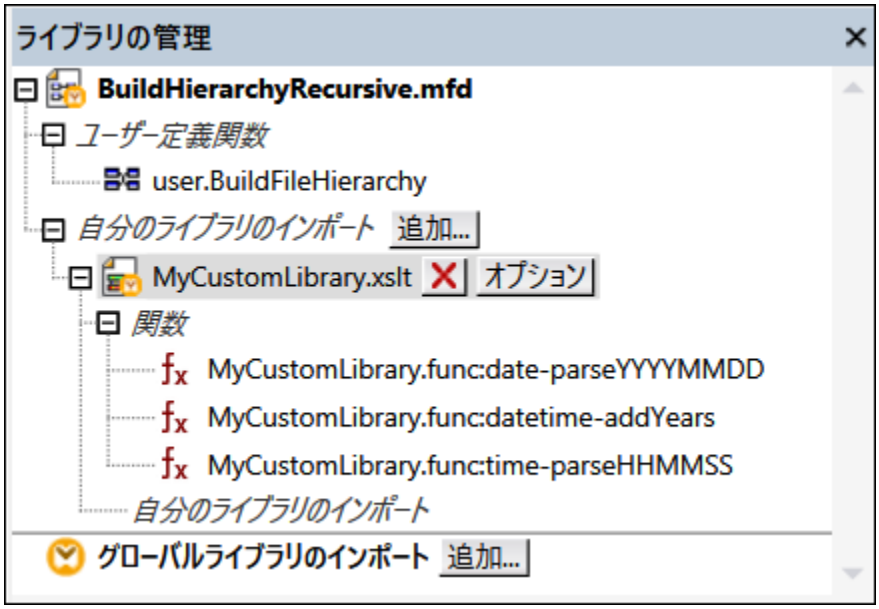
7.8.2 相対的なライブラリパス

ローカルとグローバルなライブラリで説明されているとおりライブラリが(グローバルではなく)ローカルにインポートされていることを前提として、インポートされたライブラリファイルのパスをマッピング デザインファイル(.mfd)に対して相対的に設定することができます。

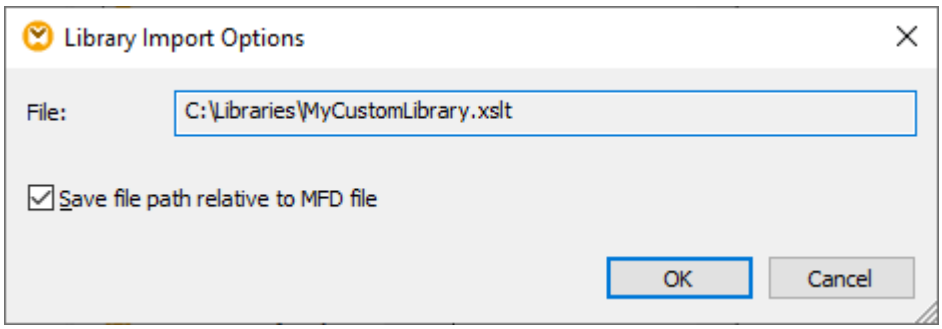
> ライブラリパスを相対的に設定するオプションはドキュメントレベルで *ローカル*にインポートされたライブラリに対してのみ適用することができます。マッピングがプログラムレベルで *グローバル*にインポートされる場合、パスは常に絶対的です。

マッピングデザインファイルに対して相対的にライブラリパスを設定する方法:

1. ライブラリ ウィンドウのベースの**ライブラリの追加/削除** をクリックします。ライブラリの管理ウィンドウが開かれます。



2. 注目するライブラリの横の「**オプション**」をクリックします。(または ライブラリを右クリックして、コンテキスト メニューから を「**オプション**」選択します 、



3. 「 MFD ファイルに対してファイル パスを相対的に保存する」チェックボックスを選択します。

メモ チェックボックスが灰色表示されている場合、ライブラリがグローバルではなくローカルにインポートされていることを確認してください。

(C) 2015-2021 Altova GmbH

チェックボックスが有効化されている場合、「名前を付けて保存」メニューコマンドを使用してマッピングファイルを新規のディレクトリに保存する際にMapForce は参照されるライブラリファイルへのパスを記録し更新します。また、ライブラリファイルがマッピングファイルと同じディレクトリに存在する場合、ディスク上の新しい場所にディレクトリ全体を移動してもパスの参照が壊されることはありません。コンポーネント上で相対的なパスを使用するを参照してください。

「MFD ファイルに対してファイルパスを相対的に保存する」チェックボックスはパスが*マッピングファイルに対して相対的に*なることを指定し、生成されたコードには影響を与えません。生成されたコード内でどのようにライブラリ参照が処理されるかについては、多種の実行環境内のパスを参照してください。

7.9 正規表現

MapForce マッピングのデザイン時、以下のコンテキストで正規表現（"regex"）を使用することができます:

- match-pattern と tokenize-regexp 関数
- ノード関数が適用されるノードをフィルターするため。詳細に関しては ノード関数とデフォルトの適用を参照してください。
- MapForce FlexText テンプレートを作成する際にテキストをパターンをベースに分割するには FlexText と正規表現を参照してください。

XSLT とXquery のため正規表現構文は XML スキーマパート 2: データベースセカンドエディションの付属 F データベースセカンドエディションの付属 F 内で定義されています。

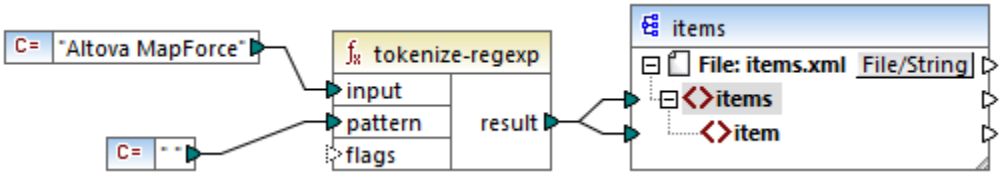
メモ C++、C#、またはJava コードを生成する際には、正規表現構文の高度な機能が異なる可能性があります。詳細に関しては各言語の正規表現ドキュメントを確認してください。

用語

tokenize-regexp 関数を基本の分析正規表現用語を検証してみましょう。この関数は正規表現の助けを借りてテキストを文字列のシーケンスに分割します。これを達成するには、関数は次の入力パラメーターを取ります:

input	関数により処理される入力文字列。正規表現はこの文字列上で操作されます。
pattern	適用される実際の正規表現パターン。
flags	これは正規表現がどのように解釈されるかを決定する追加オプション（フラグ）を定義する任意のパラメーターです。下の"フラグ"を確認してください。

下のマッピングでは、入力文字列は[Altova MapForce] です。**pattern** パラメーターは空白文字で、正規表現フラグは使用されていないことに注意してください。



これは空白文字が発生する都度テキストの分割を引き起こします。マッピングの出力:

```
<items>
   <item>Altova</item>
   <item>MapForce</item>
</items>
```

tokenize-regexp 関数は一致した文字を結果から除外することに注意してください。つまり、このサンプルの空白文字は出力から除外されています。

上のサンプルは基本で、同じ結果を正規表現を使用せず、tokenize 関数を使用して達成することができます。実際的なシナリオでは **pattern** パラメーターは更に複雑な正規表現を含んでいます。正規表現は次のどれかにより構成されることができます:

- リテラル

(C) 2015-2021 Altova GmbH

- 文字クラス
- 文字範囲
- 否定クラス
- メタ文字
- 限定子

リテラル

リテラルを使用して書かれている通り（文字通り）に文字を一致します。例えば、入力文字列が`abracadabra` の場合、**pattern** はリテラル`br` で出力は以下の通りです:

```
<items>
   <item>a</item>
   <item>acada</item>
   <item>a</item>
</items>
```

リテラル`br` は 入力文字列 `abracadabra`内に2個の一致を持っています。出力から一致する文字を削除した後、上記の3個の文字列のシーケンスが生成されます。

文字クラス

角かっこ（`[` と`]`）内に文字のセットが含まれており、これが文字クラスを作成します。文字クラス内の文字の1つのみが一致します。例:

- パターン `[aeiou]` は小文字の母音アルファベットに一致します。
- パターン `[mj]ust` は"must" と"just" に一致します。

メモ パターンに含まれる値は大文字と小文字で区別されるという点に注意してください。小文字の"a" は大文字の"A" にマッチしません。大文字と小文字を区別しないようにするには、**i** フラグを使用してください。以下を参照してください。

文字範囲

`[a-z]` を使用して2つの文字間の範囲を作成します。一度にマッチする文字は一文字だけである点に注意してください。例えば、パターン `[a-z]` は"a" から"z" までの小文字アルファベットに一致します。

否定クラス

角括弧の開始直後にハット記号（`^`）を使用することで、その文字クラスを否定したことになります。例えば、パターン `[^a-z]` は改行文字を含む文字列クラスに含まれていない文字に一致します。

任意の文字を一致する方法

改行文字を除く任意の単一文字に一致させるためにドット（`.`）メタ文字を使用します。例えば、パターン `.` は全ての単一文字に一致します。

限定子

正規表現内で限定子は一致が発生するために許可される文字またはサブ式の数量を定義します。

?	直前のアイテムのゼロまたは1度の発生に一致します。例えば、パターン `mo?` は"m" と"mo" に一致します。

+	直前のアイテムの1度または複数の発生に一致します。例えば、パターン `mo+` は "mo"、"moo"、"mooo" などに一致します。
*****	直前のアイテムのゼロまたは複数の発生に一致します。
{min,max}	*min* と *max* の間の発生数を一致させます。例えば、パターン `mo{1,3}` は "mo"、"moo"、および "mooo" に一致します。

かっこ

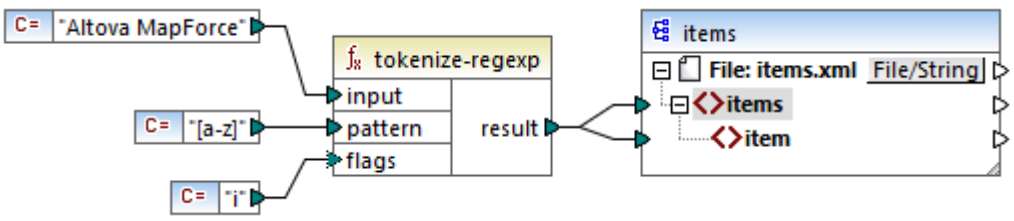
かっこ `(` と `)` は正規表現のパーツをグループ分けするために使用されます。(1文字とは異なり)または代替のある(以下参照)サブ式に限定式を適用するために使用されます。

代替

垂直バー(パイプ) 文字 `|` は "or" を意味します。`|` により区切られたサブ式の一部に一致するための使用することができます。例えば、パターン `(horse|make) sense` は "horse sense" と "make sense" に一致します。

フラグ

以下に示されるのはオプションのパラメーターで、正規表現の解釈方法を指定するために使用されます。各フラグは文字に対応します。文字はどの順序でも使用することができ、繰り返して使用することもできます。

s	このフラグが存在する場合、「全てドット」モード内で一致プロセスが操作されます。 **s** フラグがセットされていれば、入力文字列に "hello" と "world" が2 つの異なる行にある場合でも、`hello*world` という正規表現によりマッチが行われます。
m	このフラグが存在する場合、「複数行」モード内で一致プロセスが操作されます。 複数行ノードでは、ハット記号 `^` が全ての行頭(文字列全体の開始位置と改行文字の直後に来る文字)に対してマッチするようになります。 更にドル記号 `$` が全ての行末(文字列全体の終了位置と改行文字の直前に来る文字)に対してマッチするようになります。 改行に使用される文字は **#x0A** となります。
i	このフラグが存在する場合、「大文字と小文字を区別する」モード内で一致プロセスが操作されます。例えば、正規表現 `[a-z]` と **i** フラグはすべての文字 a-z と A-Z に一致します。 
x	このフラグが存在する場合、空白文字はマッチング処理が行われる前に正規表現から削除されます。空白文字は **#x09**、**#x0A**、**#x0D** および **#x20** です。

(C) 2015-2021 Altova GmbH

	メモ 文字クラス内の空白文字は削除されません。例、`[#x20]`。

7.10 関数ライブラリリレファレンス

このレファレンスチャプターでは、ライブラリ別に整理された「ライブラリ」ウィンドウ内で使用することのできるMapForce 内蔵の関数に関して説明されています。

core ライブラリは、C++、C#、Java 言語で以下のうち1つで使用することのできる関数のコレクションです: XQuery、XPath、またはXSLT。 **lang** ライブラリは、言 C++、C#、とJava 語内で使用することのできる関数です。他のライブラリは、出力の異なる型に関連した関数を含みます。

メモ

- ライブラリウィンドウ内の関数のライブラリはマッピングの変換言語により異なります。詳細に関しては、下のテーブルを参照してください。
- **XQuery**、**C#**、**C++**、**Java**、および**Built-In** はMapForce Professional またはEnterprise Edition を必要とします。
- シーケンスを扱うXPath 2.0 関数の一部は現在使用することができません。

下のテーブルは各言語内でサポートされている関数をライブラリ別にリストしています。

core | aggregate functions (集計関数)

関数	XSLT 1.0	XSLT 2.0	XSLT 3.0	XQuery 1.0	C#	C++	Java	Built-In
avg		●	●	●	●	●	●	●
count	●	●	●	●	●	●	●	●
max		●	●	●	●	●	●	●
max-string		●	●	●	●	●	●	●
min		●	●	●	●	●	●	●
min-string		●	●	●	●	●	●	●
string-join		●	●	●	●	●	●	●
sum	●	●	●	●	●	●	●	●

core | conversion functions (変換関数)

関数	XSLT 1.0	XSLT 2.0	XSLT 3.0	XQuery 1.0	C#	C++	Java	Built-In
boolean	●	●	●	●	●	●	●	●
format-date		●	●		●	●	●	●
format-dateTime		●	●		●	●	●	●

(C) 2015-2021 Altova GmbH

関数	XSLT 1.0	XSLT 2.0	XSLT 3.0	XQuery 1.0	C#	C++	Java	Built-In
format-number	●	●	●		●	●	●	●
format-time		●	●		●	●	●	●
number	●	●	●	●	●	●	●	●
parse-date					●	●	●	●
parse-dateTime					●	●	●	●
parse-number					●	●	●	●
parse-time					●	●	●	●
string	●	●	●	●	●	●	●	●

core | file path functions (ファイルパス関数)

関数	XSLT 1.0	XSLT 2.0	XSLT 3.0	XQuery 1.0	C#	C++	Java	Built-In
get-fileext	●	●	●	●	●	●	●	●
get-folder	●	●	●	●	●	●	●	●
main-mfd-filepath	●	●	●	●	●	●	●	●
mfd-filepath	●	●	●	●	●	●	●	●
remove-fileext	●	●	●	●	●	●	●	●
remove-folder	●	●	●	●	●	●	●	●
replace-fileext	●	●	●	●	●	●	●	●
resolve-filepath	●	●	●	●	●	●	●	●

core | generator functions (ジェネレーター関数)

関数	XSLT 1.0	XSLT 2.0	XSLT 3.0	XQuery 1.0	C#	C++	Java	Built-In
auto-number	●	●	●	●	●	●	●	●

core | logical functions (論理関数)

関数	XSLT 1.0	XSLT 2.0	XSLT 3.0	XQuery 1.0	C#	C++	Java	Built-In
equal	●	●	●	●	●	●	●	●
equal-or-greater	●	●	●	●	●	●	●	●
equal-or-less	●	●	●	●	●	●	●	●
greater	●	●	●	●	●	●	●	●
less	●	●	●	●	●	●	●	●
logical-and	●	●	●	●	●	●	●	●
logical-not	●	●	●	●	●	●	●	●
logical-or	●	●	●	●	●	●	●	●
not-equal	●	●	●	●	●	●	●	●

core | math functions (数学関数)

関数	XSLT 1.0	XSLT 2.0	XSLT 3.0	XQuery 1.0	C#	C++	Java	Built-In
add	●	●	●	●	●	●	●	●
ceiling	●	●	●	●	●	●	●	●
divide	●	●	●	●	●	●	●	●
floor	●	●	●	●	●	●	●	●
modulus	●	●	●	●	●	●	●	●
multiply	●	●	●	●	●	●	●	●
round	●	●	●	●	●	●	●	●
round-precision					●	●	●	●
subtract	●	●	●	●	●	●	●	●

core | node functions (ノード関数)

関数	XSLT 1.0	XSLT 2.0	XSLT 3.0	XQuery 1.0	C#	C++	Java	Built-In
is-xsi-nil	●	●	●	●	●	●	●	●

(C) 2015-2021 Altova GmbH

関数	XSLT 1.0	XSLT 2.0	XSLT 3.0	XQuery 1.0	C#	C++	Java	Built-In
local-name	●	●	●	●	●	●	●	●
node-name		●	●	●	●	●	●	●
set-xsi-nil		●	●	●	●	●	●	●
static-node-annotation	●	●	●	●	●	●	●	●
static-node-name	●	●	●	●	●	●	●	●
substitute-missing-with-xsi-nil		●	●	●	●	●	●	●

core | QName functions (QName 関数)

関数	XSLT 1.0	XSLT 2.0	XSLT 3.0	XQuery 1.0	C#	C++	Java	Built-In
QName		●	●	●	●	●	●	●
local-name-from-QName		●	●	●	●	●	●	●
namespace-uri-from-QName		●	●	●	●	●	●	●

core | sequence functions (シーケンス関数)

関数	XSLT 1.0	XSLT 2.0	XSLT 3.0	XQuery 1.0	C#	C++	Java	Built-In
distinct-values		●	●	●	●	●	●	●
exists	●	●	●	●	●	●	●	●
first-items		●	●	●	●	●	●	●
generate-sequence		●	●	●	●	●	●	●
group-adjacent		●	●		●	●	●	●
group-by		●	●		●	●	●	●
group-ending-with		●	●		●	●	●	●
group-into-blocks		●	●		●	●	●	●
group-starting-with		●	●		●	●	●	●
item-at		●	●	●	●	●	●	●
items-from-till		●	●	●	●	●	●	●

関数	XSLT 1.0	XSLT 2.0	XSLT 3.0	XQuery 1.0	C#	C++	Java	Built-In
last-items		●	●	●	●	●	●	●
not-exists	●	●	●	●	●	●	●	●
position	●	●	●	●	●	●	●	●
replicate-item		●	●	●	●	●	●	●
replicate-sequence		●	●	●	●	●	●	●
set-empty		●	●	●	●	●	●	●
skip-first-items		●	●	●	●	●	●	●
substitute-missing	●	●	●	●	●	●	●	●

core | string functions (文字列関数)

関数	XSLT 1.0	XSLT 2.0	XSLT 3.0	XQuery 1.0	C#	C++	Java	Built-In
char-from-code		●	●	●	●	●	●	●
code-from-char		●	●	●	●	●	●	●
concat	●	●	●	●	●	●	●	●
contains	●	●	●	●	●	●	●	●
normalize-space	●	●	●	●	●	●	●	●
starts-with	●	●	●	●	●	●	●	●
string-length	●	●	●	●	●	●	●	●
substring	●	●	●	●	●	●	●	●
substring-after	●	●	●	●	●	●	●	●
substring-before	●	●	●	●	●	●	●	●
tokenize		●	●	●	●	●	●	●
tokenize-by-length		●	●	●	●	●	●	●
tokenize-regexp		●	●	●	●	●	●	●
translate	●	●	●	●	●	●	●	●

(C) 2015-2021 Altova GmbH

db | null processing functions （NULL 処理関数）

関数	XSLT 1.0	XSLT 2.0	XSLT 3.0	XQuery 1.0	C#	C++	Java	Built-In
is-not-null					●	●	●	●
is-null					●	●	●	●
set-null					●	●	●	●
substitute-null					●	●	●	●

edifact | datetime functions（日付時刻関数）

関数	XSLT 1.0	XSLT 2.0	XSLT 3.0	XQuery 1.0	C#	C++	Java	Built-In
auto-format					●	●	●	●
to-date					●	●	●	●
to-datetime					●	●	●	●
to-duration					●	●	●	●
to-time					●	●	●	●

lang | datetime functions（日付時刻関数）

関数	XSLT 1.0	XSLT 2.0	XSLT 3.0	XQuery 1.0	C#	C++	Java	Built-In
age					●	●	●	●
convert-to-utc					●	●	●	●
date-from-datetime					●	●	●	●
datetime-add					●	●	●	●
datetime-diff					●	●	●	●
datetime-from-date-and-time					●	●	●	●
datetime-from-parts					●	●	●	●
day-from-datetime					●	●	●	●
day-from-duration					●	●	●	●

関数	XSLT 1.0	XSLT 2.0	XSLT 3.0	XQuery 1.0	C#	C++	Java	Built-In
duration-add					●	●	●	●
duration-from-parts					●	●	●	●
duration-subtract					●	●	●	●
hour-from-datetime					●	●	●	●
hour-from-duration					●	●	●	●
leapyear					●	●	●	●
millisecond-from-datetime					●	●	●	●
millisecond-from-duration					●	●	●	●
minute-from-datetime					●	●	●	●
minute-from-duration					●	●	●	●
month-from-datetime					●	●	●	●
month-from-duration					●	●	●	●
now					●	●	●	●
remove-timezone					●	●	●	●
second-from-datetime					●	●	●	●
second-from-duration					●	●	●	●
time-from-datetime					●	●	●	●
timezone					●	●	●	●
weekday					●	●	●	●
weeknumber					●	●	●	●
year-from-datetime					●	●	●	●
year-from-duration					●	●	●	●

lang | file functions (ファイル関数)

関数	XSLT 1.0	XSLT 2.0	XSLT 3.0	XQuery 1.0	C#	C++	Java	Built-In
read-binary-file								●

(C) 2015-2021 Altova GmbH

関数	XSLT 1.0	XSLT 2.0	XSLT 3.0	XQuery 1.0	C#	C++	Java	Built-In
write-binary-file								●

lang | generator functions (ジェネレーター関数)

関数	XSLT 1.0	XSLT 2.0	XSLT 3.0	XQuery 1.0	C#	C++	Java	Built-In
create-guid					●	●	●	●

lang | logical functions (論理関数)

関数	XSLT 1.0	XSLT 2.0	XSLT 3.0	XQuery 1.0	C#	C++	Java	Built-In
logical-xor					●	●	●	●
negative					●	●	●	●
numeric					●	●	●	●
positive					●	●	●	●

lang | math functions (数学関数)

関数	XSLT 1.0	XSLT 2.0	XSLT 3.0	XQuery 1.0	C#	C++	Java	Built-In
abs					●	●	●	●
acos					●	●	●	●
asin					●	●	●	●
atan					●	●	●	●
cos					●	●	●	●
degrees					●	●	●	●
divide-integer					●	●	●	●
exp					●	●	●	●
log					●	●	●	●
log10					●	●	●	●
max					●	●	●	●

関数	XSLT 1.0	XSLT 2.0	XSLT 3.0	XQuery 1.0	C#	C++	Java	Built-In
min					●	●	●	●
pi					●	●	●	●
pow					●	●	●	●
radians					●	●	●	●
random					●	●	●	●
sin					●	●	●	●
sqrt					●	●	●	●
tan					●	●	●	●
unary-minus					●	●	●	●

lang | QName functions (QName 関数)

関数	XSLT 1.0	XSLT 2.0	XSLT 3.0	XQuery 1.0	C#	C++	Java	Built-In
QName-as-string					●	●	●	●
string-as-QName					●	●	●	●

lang | string functions (文字列関数)

関数	XSLT 1.0	XSLT 2.0	XSLT 3.0	XQuery 1.0	C#	C++	Java	Built-In
capitalize					●	●	●	●
charset-decode								●
charset-encode								●
count-substring					●	●	●	●
empty					●	●	●	●
find-substring					●	●	●	●
format-guid-string					●	●	●	●
left					●	●	●	●
left-trim					●	●	●	●

(C) 2015-2021 Altova GmbH

関数	XSLT 1.0	XSLT 2.0	XSLT 3.0	XQuery 1.0	C#	C++	Java	Built-In
lowercase					●	●	●	●
match-pattern					●		●	●
pad-string-left					●	●	●	●
pad-string-right					●	●	●	●
repeat-string					●	●	●	●
replace					●	●	●	●
reversefind-substring					●	●	●	●
right					●	●	●	●
right-trim					●	●	●	●
string-compare-ignore-case					●	●	●	●
string-compare-ignore-case					●	●	●	●
uppercase					●	●	●	●

mime | mime

関数	XSLT 1.0	XSLT 2.0	XSLT 3.0	XQuery 1.0	C#	C++	Java	Built-In
content-encoding								●
content-type								●
decode-mime-entity								●
mime-entity								●

xbrl | Unit helpers

関数	XSLT 1.0	XSLT 2.0	XSLT 3.0	XQuery 1.0	C#	C++	Java	Built-In
xbrl-measure-currency					●	●	●	●
xbrl-measure-pure					●	●	●	●
xbrl-measure-shares					●	●	●	●

xlsx | datetime functions (日付時刻関数)

関数	XSLT 1.0	XSLT 2.0	XSLT 3.0	XQuery 1.0	C#	C++	Java	Built-In
columnname-to-index		●	●		●		●	●
date-to-xlsx		●	●		●		●	●
datetime-to-xlsx		●	●		●		●	●
index-to-columnname		●	●		●		●	●
time-to-xlsx		●	●		●		●	●
xlsx-to-date		●	●		●		●	●
xlsx-to-datetime		●	●		●		●	●
xlsx-to-time		●	●		●		●	●

xpath2 | accessors

関数	XSLT 1.0	XSLT 2.0	XSLT 3.0	XQuery 1.0	C#	C++	Java	Built-In
base-uri		●	●	●				
node-name		●	●	●				
string		●	●	●				

xpath2 | anyURI functions (anyURI 関数)

関数	XSLT 1.0	XSLT 2.0	XSLT 3.0	XQuery 1.0	C#	C++	Java	Built-In
resolve-uri		●	●	●				

xpath2 | boolean functions (ブール値関数)

関数	XSLT 1.0	XSLT 2.0	XSLT 3.0	XQuery 1.0	C#	C++	Java	Built-In
false		●	●	●				
true		●	●	●				

(C) 2015-2021 Altova GmbH

xpath2 | constructors

関数	XSLT 1.0	XSLT 2.0	XSLT 3.0	XQuery 1.0	C#	C++	Java	Built-In
xs:ENTITY		●	●	●				
xs:ID		●	●	●				
xs:IDREF		●	●	●				
xs:NCName		●	●	●				
xs:NMTOKEN		●	●	●				
xs:Name		●	●	●				
xs:QName		●	●	●				
xs:anyURI		●	●	●				
xs:base64Binary		●	●	●				
xs:boolean		●	●	●				
xs:byte		●	●	●				
xs:date		●	●	●				
xs:dateTime		●	●	●				
xs:dayTimeDuration		●	●	●				
xs:decimal		●	●	●				
xs:double		●	●	●				
xs:duration		●	●	●				
xs:float		●	●	●				
xs:gDay		●	●	●				
xs:gMonth		●	●	●				
xs:gMonthDay		●	●	●				
xs:gYear		●	●	●				
xs:gYearMonth		●	●	●				
xs:hexBinary		●	●	●				

関数	XSLT 1.0	XSLT 2.0	XSLT 3.0	XQuery 1.0	C#	C++	Java	Built-In
xs:int		●	●	●				
xs:integer		●	●	●				
xs:language		●	●	●				
xs:long		●	●	●				
xs:negativeInteger		●	●	●				
xs:nonNegativeInteger		●	●	●				
xs:nonPositiveInteger		●	●	●				
xs:normalizedString		●	●	●				
xs:positiveInteger		●	●	●				
xs:short		●	●	●				
xs:string		●	●	●				
xs:time		●	●	●				
xs:token		●	●	●				
xs:unsignedByte		●	●	●				
xs:unsignedInt		●	●	●				
xs:unsignedLong		●	●	●				
xs:unsignedShort		●	●	●				
xs:untypedAtomic		●	●	●				
xs:yearMonthDuration		●	●	●				

xpath2 | context functions (コンテキスト関数)

関数	XSLT 1.0	XSLT 2.0	XSLT 3.0	XQuery 1.0	C#	C++	Java	Built-In
current-date		●	●	●				
current-dateTime		●	●	●				
current-time		●	●	●				
default-collation		●	●	●				

(C) 2015-2021 Altova GmbH

関数	XSLT 1.0	XSLT 2.0	XSLT 3.0	XQuery 1.0	C#	C++	Java	Built-In
implicit-timezone		●	●	●				
last		●	●	●				

xpath2 | durations, date and time functions (期間、日付、および時刻関数)

関数	XSLT 1.0	XSLT 2.0	XSLT 3.0	XQuery 1.0	C#	C++	Java	Built-In
adjust-date-to-timezone		●	●	●				
adjust-date-to-timezone		●	●	●				
adjust-dateTime-to-timezone		●	●	●				
adjust-dateTime-to-timezone		●	●	●				
adjust-time-to-timezone		●	●	●				
adjust-time-to-timezone		●	●	●				
day-from-date		●	●	●				
day-from-dateTime		●	●	●				
days-from-duration		●	●	●				
hours-from-dateTime		●	●	●				
hours-from-duration		●	●	●				
hours-from-time		●	●	●				
minutes-from-dateTime		●	●	●				
minutes-from-duration		●	●	●				
minutes-from-time		●	●	●				
month-from-date		●	●	●				
month-from-dateTime		●	●	●				
months-from-duration		●	●	●				
seconds-from-dateTime		●	●	●				
seconds-from-duration		●	●	●				
seconds-from-time		●	●	●				

関数	XSLT 1.0	XSLT 2.0	XSLT 3.0	XQuery 1.0	C#	C++	Java	Built-In
subtract-dateTimes		●	●	●				
subtract-dates		●	●	●				
subtract-times		●	●	●				
timezone-from-date		●	●	●				
timezone-from-dateTime		●	●	●				
timezone-from-time		●	●	●				
year-from-date		●	●	●				
year-from-dateTime		●	●	●				
years-from-duration		●	●	●				

xpath2 | node functions (ノード関数)

関数	XSLT 1.0	XSLT 2.0	XSLT 3.0	XQuery 1.0	C#	C++	Java	Built-In
lang		●	●	●				
local-name		●	●	●				
local-name		●	●	●				
name		●	●	●				
name		●	●	●				
namespace-uri		●	●	●				
namespace-uri		●	●	●				
number		●	●	●				
number		●	●	●				

xpath2 | numeric functions (数値関数)

関数	XSLT 1.0	XSLT 2.0	XSLT 3.0	XQuery 1.0	C#	C++	Java	Built-In
abs		●	●	●				
round-half-to-even		●	●	●				

(C) 2015-2021 Altova GmbH

xpath2 | string functions (文字列関数)

関数	XSLT 1.0	XSLT 2.0	XSLT 3.0	XQuery 1.0	C#	C++	Java	Built-In
codepoints-to-string		●	●	●				
compare		●	●	●				
compare		●	●	●				
ends-with		●	●	●				
ends-with		●	●	●				
lower-case		●	●	●				
matches		●	●	●				
normalize-unicode		●	●	●				
replace		●	●	●				
starts-with		●	●	●				
string-to-codepoints		●	●	●				
substring-after		●	●	●				
substring-before		●	●	●				
upper-case		●	●	●				

xpath3 | external information functions (外部情報関数)

関数	XSLT 1.0	XSLT 2.0	XSLT 3.0	XQuery 1.0	C#	C++	Java	Built-In
available-environment-variables			●					
environment-variable			●					
unparsed-text			●					
unparsed-text-available			●					
unparsed-text-lines			●					

xpath3 | formatting functions (書式設定関数)

関数	XSLT 1.0	XSLT 2.0	XSLT 3.0	XQuery 1.0	C#	C++	Java	Built-In
format-date			●					
format-dateTime			●					
format-integer			●					
format-time			●					

xpath3 | math functions (数学関数)

関数	XSLT 1.0	XSLT 2.0	XSLT 3.0	XQuery 1.0	C#	C++	Java	Built-In
acos			●					
asin			●					
atan			●					
atan2			●					
cos			●					
exp			●					
exp10			●					
log			●					
log10			●					
pi			●					
pow			●					
sin			●					
sqrt			●					
tan			●					

xpath3 | URI functions (URI 関数)

関数	XSLT 1.0	XSLT 2.0	XSLT 3.0	XQuery 1.0	C#	C++	Java	Built-In
encode-for-uri			●					

(C) 2015-2021 Altova GmbH

関数	XSLT 1.0	XSLT 2.0	XSLT 3.0	XQuery 1.0	C#	C++	Java	Built-In
escape-html-uri			●					
iri-to-uri			●					

xslt10 | xpath functions (xpath 関数)

関数	XSLT 1.0	XSLT 2.0	XSLT 3.0	XQuery 1.0	C#	C++	Java	Built-In
lang	●							
last	●							
local-name	●	●	●					
name	●	●	●					
namespace-uri	●	●	●					
position	●							

xslt10 | xslt functions (xslt 関数)

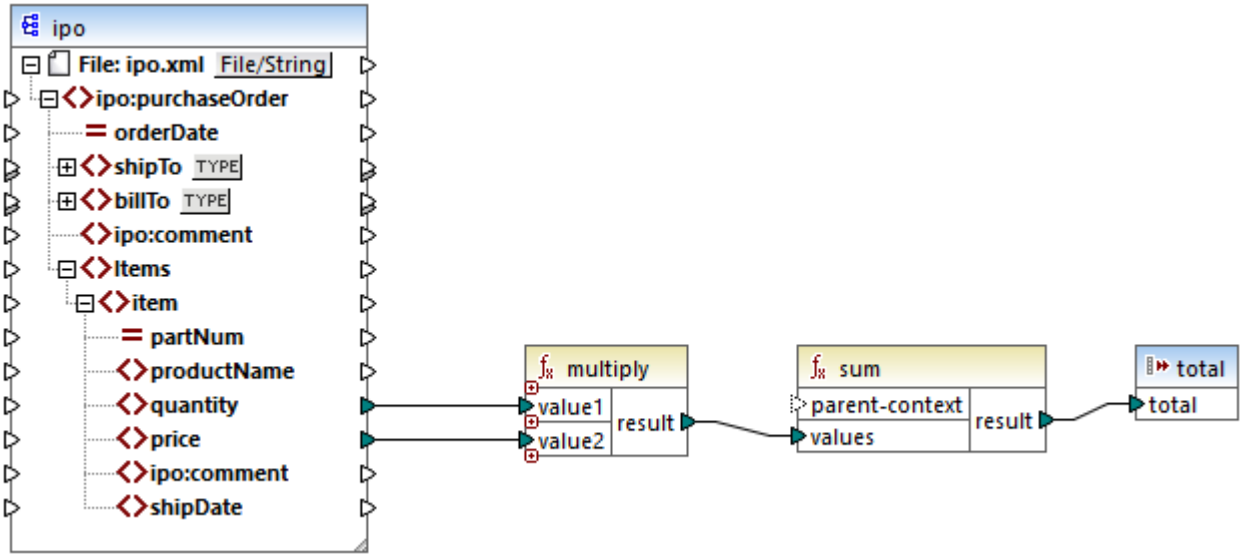
関数	XSLT 1.0	XSLT 2.0	XSLT 3.0	XQuery 1.0	C#	C++	Java	Built-In
currrent	●							
document	●							
element-available	●							
function-available	●							
generate-id	●	●	●					
system-property	●	●	●					
unparsed-entity-uri	●							

7.10.1 core | aggregate functions (集計関数)

「集計」は合計、カウント、平均など1つの結果を取得するために同じ型の複数の値を処理することを意味します。MapForce 内で `avg`、`count`、`max`、および他の集計関数を使用してデータの集計を行うことができます。

以下の2つの引数はすべての集計関数に共通します:

1. **parent-context**。この引数はオプションです。これによりデフォルトのマッピングコンテキストをオーバーライドすることができます（この結果、関数のスコープが、または、関数が反復する値が変更されます）。成功例に関しては例: 親コンテキストの変更で詳細に説明されています。
2. **values**。この引数は、処理する値を与えるソースアイテムに接続されている必要があります。例えば、下に示されるマッピング内で `sum` 関数はソースXMLファイルを基にする数値のシーケンスを入力としてとります。ソースXMLファイル内の各アイテムのために `multiply` 関数はアイテムの価格を個数と掛けた値を取得し `sum` 関数にパスします。`sum` 関数はすべての入力の値を集計し、マッピングの出力である総計結果を生成します。**...\MapForceExamples\Tutorial** ディレクトリ内でこのマッピングを見つけることができます。

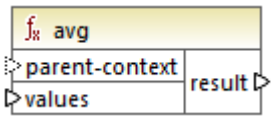


SimpleTotal.mfd

`min`、`max`、`sum` と `avg`、などの集計関数の一部は排他的に数値とのみ作業することができます。これらの関数の入力データが **decimal** データ型に変換され、処理されます。

7.10.1.1 avg

入力シーケンスから得られた値の平均値を返します。空のセットの平均は結果は空のセットになります。



言語

Built-in、C++、C#、Java、XQuery、XSLT 2.0、XSLT 3.0。

パラメーター

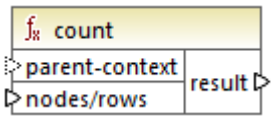
引数	説明
parent-context	オプションの引数。親コンテキストを提供します。サンプル: 親コンテキストの変更で詳細に説明されています。
values	この引数は実際のデータを与えるソースアイテムに接続されている必要があります。与えられる引数の値は数値である必要があります。

サンプル

サンプル: キー別にレコードをグループ分けする方法を参照してください

7.10.1.2 count

入力シーケンスを構成する個別のアイテムの数量を返します。入力が空セットのカウントは0 になります。



言語

Built-in、C++、C#、Java、XQuery、XSLT 1.0、XSLT 2.0、XSLT 3.0。

この関数はXSLT 1.0 内では機能が制限されています。

パラメーター

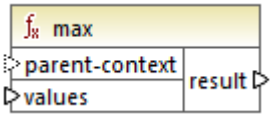
引数	説明
parent-context	オプションの引数。親コンテキストを提供します。サンプル: 親コンテキストの変更で詳細に説明されています。
nodes/rows	この引数は数えられるソースアイテムに接続されている必要があります。

サンプル

サンプル: 親コンテキストの変更、サンプル: テーブル行のカウントを参照してください。

7.10.1.3 max

入力シーケンス内の全ての数値の最小値を返します。空のセットの最大値は空のセットになります。



言語

Built-in、C++、C#、Java、XQuery、XSLT 2.0、XSLT 3.0。

パラメーター

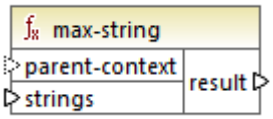
引数	説明
parent-context	オプションの引数。親コンテキストを提供します。サンプル: 親コンテキストの変更で詳細に説明されています。
values	この引数は実際のデータを与えるソースアイテムに接続されている必要があります。与えられる引数の値は数値である必要があります。文字列のシーケンスから最大値を得るには max-string 関数を使用してください。

サンプル

サンプル: キー別にレコードをグループ分けする方法を参照してください

7.10.1.4 max-string

入力シーケンス内の全ての文字列の値の最大値を返します。例えば、`max-string("a","b","c")` は `"c"` を返します。**strings** 引数が空のセットの場合、関数は空のセットを返します。



言語

Built-in、C++、C#、Java、XQuery、XSLT 2.0、XSLT 3.0。

パラメーター

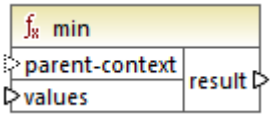
引数	説明
parent-context	オプションの引数。親コンテキストを提供します。サンプル: 親コンテキストの変更で詳細に説明されています。

(C) 2015-2021 Altova GmbH

引数	説明
strings	この引数は実際のデータを与えるソースアイテムに接続されている必要があります。与えられた引数の値は `xs:string` のシーケンス(ゼロまたは複数)である必要があります。

7.10.1.5 min

入力シーケンス内の全ての数値の最小値を返します。空のセットの最小値は空のセットになります。



言語

Built-in、C++、C#、Java、XQuery、XSLT 2.0、XSLT 3.0。

パラメーター

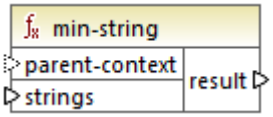
引数	説明
parent-context	オプションの引数。親コンテキストを提供します。サンプル: 親コンテキストの変更で詳細に説明されています。
values	この引数は実際のデータを与えるソースアイテムに接続されている必要があります。与えられる引数の値は数値である必要があります。文字列のシーケンスから最小値を得るには min-string 関数を使用してください。

サンプル

サンプル: キー別にレコードをグループ分けする方法を参照してください

7.10.1.6 min-string

入力シーケンス内の全ての文字列の値の最小値を返します。例えば、`min-string("a","b","c")` は `"c"` を返します。**strings** 引数が空のセットの場合、関数は空のセットを返します。



言語

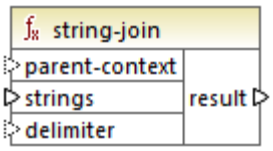
Built-in、C++、C#、Java、XQuery、XSLT 2.0、XSLT 3.0。

パラメーター

引数	説明
parent-context	オプションの引数。親コンテキストを提供します。サンプル: 親コンテキストの変更で詳細に説明されています。
strings	この引数は実際のデータを与えるソースアイテムに接続されている必要があります。与えられた引数の値は `xs:string` のシーケンス(ゼロまたは複数)である必要があります。

7.10.1.7 string-join

入力シーケンスのすべての値を区切り文字として使用することを選択した文字列により区切られた一つの文字列に連結します。**strings** 引数が空に設定されている場合関数は空の文字列を返します。



言語

Built-in、C++、C#、Java、XQuery、XSLT 2.0、XSLT 3.0。

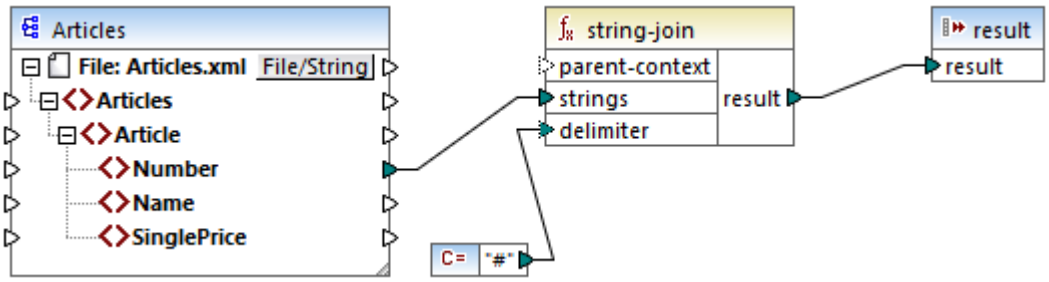
パラメーター

引数	説明
parent-context	オプションの引数。親コンテキストを提供します。サンプル: 親コンテキストの変更で詳細に説明されています。
strings	この引数は実際のデータを与えるソースアイテムに接続されている必要があります。与えられた引数の値は `xs:string` のシーケンス(ゼロまたは複数)である必要があります。
delimiter	オプションの引数。2つの連続した文字列の間に区切り文字を挿入するように指定します。

サンプル

以下に示されるサンプルではソースXMLファイルには以下の番号を持つ4つの**Article**アイテムが含まれています: 1、2、3 と4。

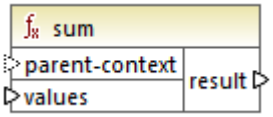
(C) 2015-2021 Altova GmbH



定数は"#"を区切り文字として提供しています。マッピングの結果はこのため`1#2#3#4`です。区切り文字を提供しない場合、結果は`1234`になります。

7.10.1.8 sum

入力シーケンス内のすべての値の合計値を返します。空のセットの合計は0になります。



言語

Built-in、C++、C#、Java、XQuery、XSLT 1.0、XSLT 2.0、XSLT 3.0。

パラメーター

引数	説明
parent-context	オプションの引数。親コンテキストを提供します。サンプル: 親コンテキストの変更で詳細に説明されています。
values	この引数は実際のデータを与えるソースアイテムに接続されている必要があります。与えられる引数の値は数値である必要があります。

サンプル

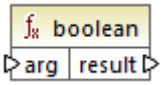
サンプル: ノードの値の集計を参照してください

7.10.2 core | conversion functions (変換関数)

明示的なデータ型変換をサポートするには、**conversion** ライブラリ内で複数の変換関数を使用することができます。多くの場合MapForceは必要な変換を自動的に作成するため、変換関数は常に必要ではありません。変換関数は日付および時刻の値を書式設定する場合、または値を比較する場合において通常役に立ちます。例えば、マッピングアイテムの一部が(整数と文字列など)異なる型の場合 number 変換関数を使用して数値の比較を強制することができます。

7.10.2.1 boolean

arg の値をブール値の値として変換します。(**equal**、**greater** や フィルターとif-else 条件 などの理論関数と作業する際に役に立ちます。ブール値 **false** を得るには空の文字列、または数値 0 を引数として与えます。ブール値 **true** を得るには空の文字列、または数値 1 を引数として与えます。



言語

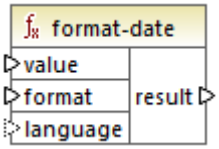
Built-in、C++、C#、Java、XQuery、XSLT 1.0、XSLT 2.0、XSLT 3.0。

パラメーター

引数	説明
arg	必須の引数。変換される値を提供します。

7.10.2.2 format-date

型 `xs:date` の日付の値を特定のオプションに従い文字列と書式に変換します。



言語

Built-in、C++、C#、Java、XSLT 2.0、XSLT 3.0。

パラメーター

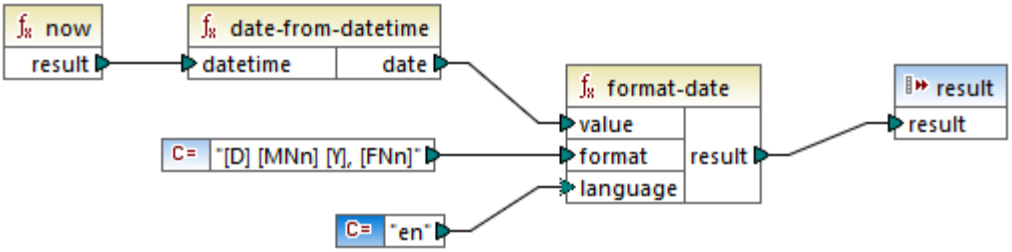
引数	説明
value	変換する `xs:date` 値です。
format	日付がフォーマットされる方法を識別する文字列をフォーマットします。この引数はformat-dateTime 関数内の **format** 引数と同じ方法で使用されます。
language	オプションの引数。与えられると、月の名前と週の曜日が特定の言語で返されます。有効な値:

(C) 2015-2021 Altova GmbH

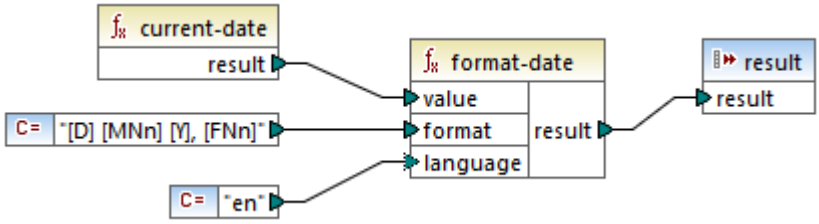
引数	説明	
	de	ドイツ語
	en (default)	英語
	es	スペイン語
	fr	フランス語
	ja	日本語

サンプル

以下のマッピングは以下のようなフォーマットで現在の日付を出力します: "25 March 2020, W ednesday"。この値をスペイン語に翻訳するには、**language** 引数の値を **es** に設定します。

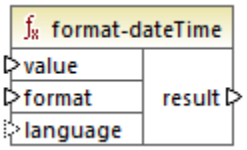


上のマッピングはBuilt-in、C++、C#、またはJava 変換言語のためにデザインされています。XSLT 2.0 では、同じ結果を以下のマッピングで得ることができます:



7.10.2.3 format-dateTime

型 `xs:dateTime` の値を文字列に変換します。日時の文字列の表示は **format** 引数の値に従いフォーマットされます。



言語

Built-in、C++、C#、Java、XSLT 2.0、XSLT 3.0。

パラメーター

引数	説明
value	変換する `xs:dateTime` 値です。
format	**value** がフォーマットされる方法を識別する文字列をフォーマットします。下の「説明」を確認してください。
language	オプションの引数。与えられると、月の名前と週の曜日が特定の言語で返されます。有効な値: **de** ドイツ語 **en (default)** 英語 **es** スペイン語 **fr** フランス語 **ja** 日本語

メモ 値はターゲットの型へキャストされるため、関数の出力 (result) が文字列以外のアイテムへ接続されている場合、書式が失われる可能性があります。自動的なキャストを無効化するには ターゲットコンポーネント内の[コンポーネント設定] 内の**ターゲット型にターゲット値をキャストする** チェックボックスをクリアします。

リマーク

format 引数は、`[Y]/[M]/[D]` などの角括弧に囲まれた変数マーカーと呼ばれる文字列により構成されます。角括弧外の文字はリテラルな文字です。角括弧を定数文字として使用する場合、括弧を重ねあわせて記述する必要があります。

各変数マーカーは、表示する日付や時刻を表す識別子、オプションの書式修飾子、プレゼンテーション修飾子、幅修飾子により構成され、これらオプションの修飾子はコンマにより分割されます。

```
format := (literal | argument)*
argument := [component(format)?(presentation)?(width)?]
width := , min-width ("-" max-width)?
```

コンポーネントは以下のとおりです:

識別子	説明	デフォルトの表示
Y	年(絶対値)	4桁の数値 (2010)
M	その年の月	1-12
D	その月の日付	1-31
d	その年の日付	1-366

(C) 2015-2021 Altova GmbH

識別子	説明	デフォルトの表示
F	その週の曜日	曜日の名前(言語に依存)
W	その年の週	1-53
w	その月の週	1-5
H	時(24時間)	0-23
h	時(12時間)	1-12
P	A.M. またはP.M.	アルファベット(言語に依存)
m	その時間の分	00-59
s	その分の秒	00-59
f	小数点以下の秒	数値、小数点1つ
Z	UTC からのタイムゾーンオフセット	+08:00
z	GMT からのタイムゾーンオフセット	GMT+n

書式修飾子は以下であることができます:

文字	説明	サンプル
1	最初に0 を伴わない数値書式:	1, 2, 3
01	2 桁の数値書式:	01, 02, 03
N	コンポーネントの名前、大文字[1]	MONDAY, TUESDAY 1)
n	コンポーネントの名前、小文字[1]	monday, tuesday 1)
Nn	コンポーネントの名前、頭文字が大文字[1]	Monday, Tuesday

脚注:

1.**N**、**n**、および**Nn** 修飾子は以下のコンポーネントのみによりサポートされています: **M**、**d**、**D**。

幅の修飾子は必要な場合最小の幅を示す小数が次に続くコンマにより表示されます。オプションで、最大の幅を示す他の小数が次に表示されるダッシュ文字を追加することもできます。例:

- `[D,2]` は2桁で表示された先頭が0 で表示された月の日です。
- `[MNn,3-3]` は*Jan*、*Feb*、*Mar* など3文字で表示された月の名前です。

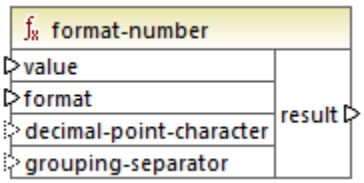
サンプル

format-dateTime 関数を使用して、下のテーブルは`xs:dateTime` 値のフォーマットのサンプルを表示しています。"「値」列は、**value** 引数に与えられる値を指定します。「書式」列は、**format** 引数の値を指定します。「結果」列は、関数により返される内容を表示しています。

値	書式	結果
2003-11-03T00:0:00	`[D]/[M]/[Y]`	2003/03/11
2003-11-03T00:0:00	`[Y]-[M,2]-[D,2]`	2003/11/03
2003-11-03T00:0:00	`[Y]-[M,2]-[D,2] [H,2]:[m]:[s]`	2003-11-03 00:00:00
2010-06-02T08:02	`[Y] [MNn] [D01] [F,3-3] [d] [H]:[m]:[s].[f]`	2010 June 02 Wed 153 8:02:12.054
2010-06-02T08:02	`[Y] [MNn] [D01] [F,3-3] [d] [H]:[m]:[s].[f] [z]`	2010 June 02 Wed 153 8:02:12.054 GMT+02:00
2010-06-02T08:02	`[Y] [MNn] [D1] [F] [H]:[m]:[s].[f] [Z]`	2010 June 2 Wednesday 8:02:12.054 +02:00
2010-06-02T08:02	`[Y] [MNn] [D] [F,3-3] [H01]:[m]:[s]`	2010 June 2 Wed 08:02:12

7.10.2.4 format-number

数値を文字列に変換し、特定のオプションに従い書式設定します。



言語

Built-in、C++、C#、Java、XSLT 1.0、XSLT 2.0、XSLT 3.0。

パラメーター

引数	説明
value	必須の引数。フォーマットする数値を提供します。
format	必須の引数。数値がフォーマットされる方法を識別するフォーマット文字列を提供します。下の「説明」を確認してください。
decimal-point-format	オプションの引数。小数点文字として使用される文字を提供します。デフォルトの値はフルストップ（.）文字です。
grouping-separator	オプションの引数。数値グループを分割するために使用されるセパレーター/デミタ。デフォルトの値はコンマ（,）文字です。

(C) 2015-2021 Altova GmbH

メモ 値はターゲットの型へキャストされるため、関数の出力 (result) が文字列以外のアイテムへ接続されている場合、書式が失われる可能性があります。自動的なキャストを無効化するには ターゲットコンポーネント内の[コンポーネント設定] 内の**ターゲット型にターゲット値をキャストする** チェックボックスをクリアします。

リマーク

format 引数は以下の書式を取ります:

```
format := subformat (;subformat)?
  subformat := (prefix)? integer (.fraction)? (suffix)?
  prefix := any characters except special characters
  suffix := any characters except special characters
  integer := (#)* (0)* ( allowing ',' to appear)
  fraction := (0)* (#)* (allowing ',' to appear)
```

最初の *サブフォーマット* は正の数を書式設定するために使用され、2番目のサブフォーマット は負の数をフォーマットするために使用されます。1つの *サブフォーマット* のみが指定されている場合、同じサブフォーマットが負の数のためにも使用されますが、マイナスのシンボルが*プレフィックス*に追加されます。

特別な文字	デフォルト	説明
0 桁	0	結果内のこの時点で桁が常に表示されます。
桁	#	余剰な先行または最終的に0 になる場合を除き、結果の文字列にてこの場所に常に数値が表示されます。
小数点	。	数値の整数部と小数点以下を分けるために使用されます。
grouping-separator	,	桁のグループを区切ります。
パーセント記号	%	数値を100倍して、パーセンテージとして表示します。
パーミル	‰	数値を1000倍して、パーミルとして表示します。

下のテーブルでは書式設定文字列とその結果のサンプルが表示されています。

メモ `format-number` 関数により使用される端数処理メソッドは「ハーフアップ」です。これは値が0.5 よりも大きいまたは等しい場合丸められることを意味します。0.5 未満であれば切り捨てが行われます。このメソッドは、生成されたコードとビルトイン実行エンジンでのみ使用することができます。XSLT 1.0 では四捨五入のモードが定義されていません。XSLT 2.0 では端数処理モードは偶数への丸めです。

数値	書式設定文字列	結果
1234.5	#,##0.00	1,234.50
123.456	#,##0.00	123.46
1000000	#,##0.00	1,000,000.00
-59	#,##0.00	-59.00
1234	###0.0###	1234.0

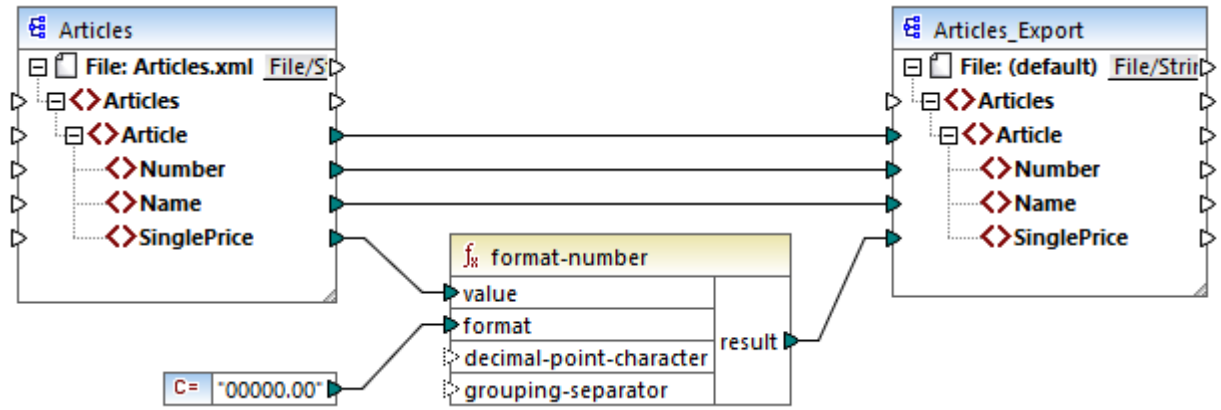
数値	書式設定文字列	結果
1234.5	###0.0###	1234.5
.00025	###0.0###	0.0003
.00035	###0.0###	0.0004
0.25	#00%	25%
0.736	#00%	74%
1	#00%	100%
-42	#00%	-4200%
-3.12	#.00;(#.00)	(3.12)
-3.12	#.00;#.00CR	3.12CR

サンプル

下で説明されるマッピングはソースXML からデータを読み取りXML ターゲットに書き込みます。ソース内には以下の小数値を含む複数の **SinglePrice** 要素が存在します: **25**, **2.30**, **34**, **57.50**. このマッピングには2 つのゴールがあります:

1. 全ての値の左側をゼロでパディし、重要な部分が5 桁になるようにします。
2. 全ての値の左側をゼロでパディし、重要な部分が2 桁になるようにします。

これを達成するために、フォーマット文字列 `00000.00` が `format-number` 関数の引数として提供されています。



PreserveFormatting.mfd

この結果、ターゲット内の値は以下のようになります:

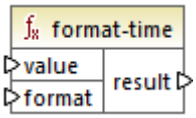
```
00025.00
00002.30
00034.00
00057.50
```


(C) 2015-2021 Altova GmbH

マッピングデザインを次のパスを使用して見つけることができます: **<マイドキュメント>\Altova\MapForce2021\MapForceExamples\PreserveFormatting.mfd**。

7.10.2.5 format-time

`xs:time` 入力値を文字列へ変換します。



言語

Built-in、C++、C#、Java、XSLT 2.0、XSLT 3.0。

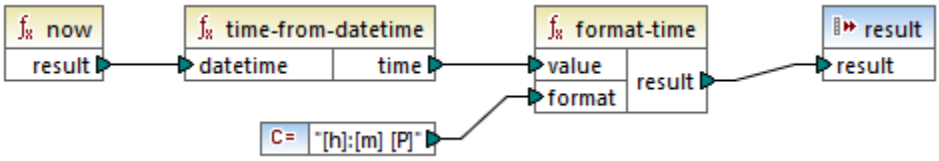
パラメーター

引数	説明
value	必須の引数。フォーマットする `xs:dateTime` 値を提供します。
format	必須の引数。フォーマット文字列を提供します。この引数はformat-dateTime 関数内の **format** 引数と同じ方法で使用されます。

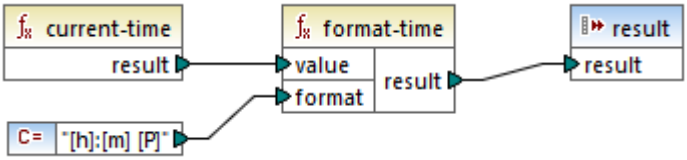
サンプル

以下のマッピングは以下の `2:15 p.m.` ような書式で現在の時刻を出力します。これを達成するには以下の書式のフォーマット文字列 `[h]:[m] [P]` を使用します:

- **[h]** は12 時間書式の現在の時間数です
- **[m]** は現在の分数です
- **[P]** は"a.m." または"p.m." の部分です。

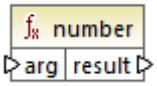


上のマッピングはBuilt-in、C++、C#、またはJava 変換言語のためにデザインされています。XSLT 2.0 では、同じ結果を以下のマッピングで得ることができます:



7.10.2.6 number

arg が文字列または**ブール値**である個所で **arg** の値を数値に変換します。**arg** は文字列で、MapForce は数値として解析を試みます。例えば、`"12.56"` などの文字列が小数の値 `12.56` に変換されます。**arg** が**ブール値 true** の場合、数値 **1** に変換されます。**arg** が**ブール値 false** の場合、数値 **0** に変換されます。



言語

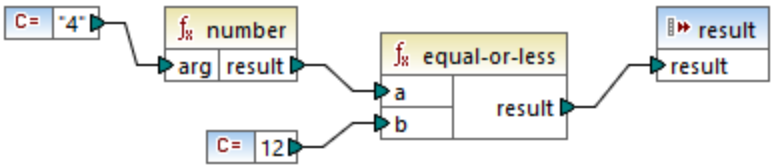
Built-in、C++、C#、Java、XQuery、XSLT 1.0、XSLT 2.0、XSLT 3.0。

パラメーター

引数	説明
arg	必須の引数。変換される値を提供します。

サンプル

下のサンプルでは、最初の定数は型 `string` で、文字列 "4" を含んでいます。2番目の定数は、定数 12 を含んでいます。2つの値が数値として比較されるためには型が一致する必要があります。

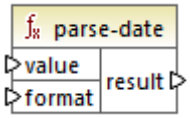


number 関数を最初の定数に追加するには、文字列 "4" の定数を数値 4 に変換します。比較の結果は "true" になります。**number** 関数が使用されない場合結果が "false" の文字列の比較が発生します。(すなわち "4" が直接 **a** に接続されています)。

7.10.2.7 parse-date

文字列を日付に変換します。この関数は parse-dateTime 関数をベースにしたもので、時刻コンポーネントが無視されます。出力は `xs:date` 型になります。

(C) 2015-2021 Altova GmbH



言語

Built-in、C++、C#、Java。

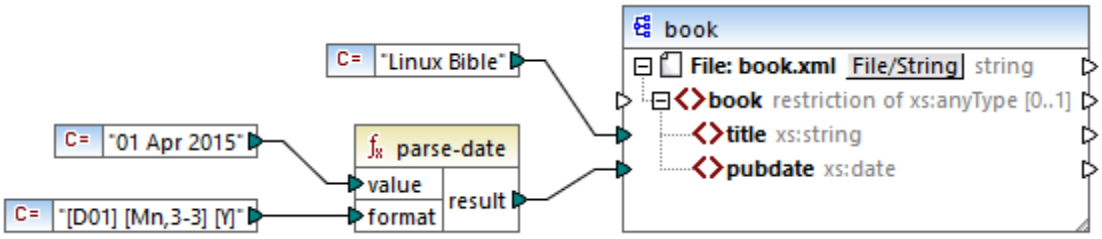
パラメーター

引数	説明
value	必須の引数。変換される文字列の値を提供します。
format	必須の引数。フォーマット文字列を提供します。この引数は parse-dateTime 関数内の **format** 引数と同じ方法で使用されます。

サンプル

下のマッピングは `"01 Apr 2015"` を解析し、日付に変換し結果を型 `xs:date` のターゲットアイテム（**pubdate**）に書き込みます。これは書式 `[D01] [MNn,3-3] [Y]` を使用して達成することができます:

- `[D01]` は2桁で表示された月の日です。
- `[MNn,3-3]` は月の名前です。最小と最大の幅は3文字です。
- `[Y]` は年数です



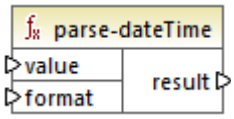
（XML と名前宣言を除く）結果は以下の通りです:

```
<book>
   <title>Linux Bible</title>
   <pubdate>2015-04-01</pubdate>
</book>
```

サンプル: RSS フィードからデータをマップするを参照してください。

7.10.2.8 parse-dateTime

文字列として表されている日付と時刻を型 `xs:dateTime` の値に変換します。



言語

Built-in、C++、C#、Java。

パラメーター

引数	説明
value	変換される文字列の値
format	**value** に適用される書式マスクを指定します。

リマーク

書式マスクは以下のコンポーネントにより構成されることができます:

コンポーネント	説明	デフォルトの表示
Y	年(絶対値)	4桁の数値(2010)
M	その年の月	1-12
D	その月の日付	1-31
d	その年の日付	1-366
H	時(24時間)	0-23
h	時(12時間)	1-12
P	A.M. または P.M.	アルファベット(言語に依存)
m	その時間の分	00-59
s	その分の秒	00-59
f	小数点以下の秒	数値、小数点1つ
Z	UTC からのタイムゾーンオフセット	+08:00
z	GMT からのタイムゾーンオフセット	GMT+n

(C) 2015-2021 Altova GmbH

上記のコンポーネントの一部は識別子を取ります（例えば、日付を1桁、または2桁として解釈するために使用されます）:

識別子	説明	サンプル
1	最初に0 を伴わない数値書式: 1, 2, 3, …	1, 2, 3
01	2 桁の数値書式: 01, 02, 03, …	01, 02, 03
N	コンポーネント名、大文字	FEBRUARY, MARCH
n	コンポーネント名、小文字	february, march
Nn	コンポーネント、頭文字が大文字	February, March

メモ *N*、*n*、と*Nn* 識別子はコンポーネント *M*（月）のみをサポートします。

幅の修飾子は必要な場合最小の幅を示す小数が次に続くコンマにより表示されます。オプションで、最大の幅を示す他の小数が次に表示されるダッシュ文字を追加することもできます。例:

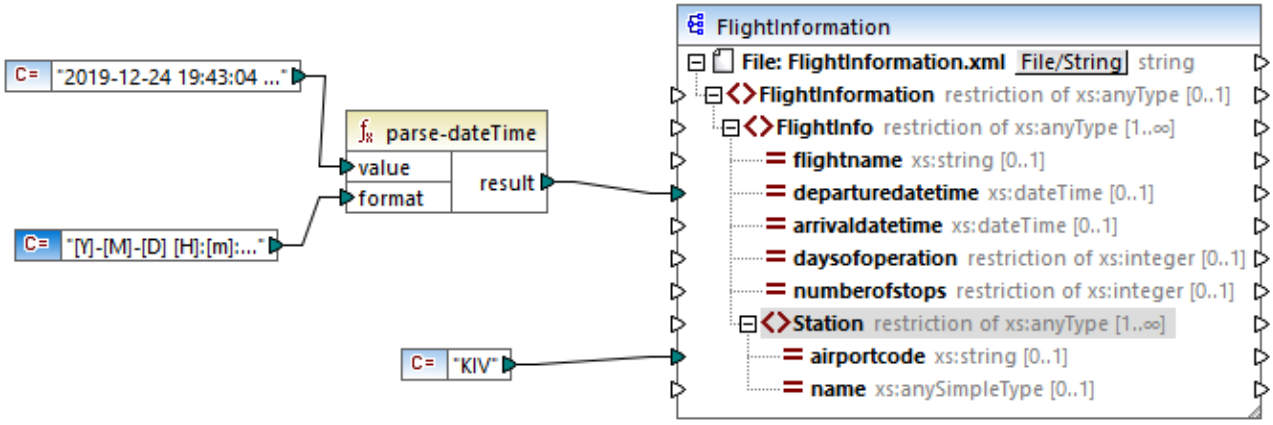
- `[D,2]` は2桁で表示された先頭が0 で表示された月の日です。
- `[MNn,3-3]` は *Jan*、*Feb*、*Mar* など3文字で表示された月の名前です。

下のテーブルは書式のサンプルの一部をリストしています:

値	書式	結果
21-03-2002 16:21:12.492 GMT+02:00	`[D]-[M]-[Y] [H]:[m]:[s].[f] [z]`	2002-03-21T16:21:12.492+02:00
315 2004 +01:00	`[d] [Y] [Z]`	2004-11-10T00:00:00+01:00
1.December.10 03:2:39 p.m. +01:00	`[D].[MNn].[Y,2-2] [h]:[m]:[s] [P] [Z]`	2010-12-01T15:2:39+01:00
20110620	`[Y,4-4][M,2-2][D,2-2]`	2011-06-20T00:0:00

サンプル

文字列の値 `2019-12-24 19:43:04 +02:00` はフォーマットマスク `[Y]-[M]-[D] [H]:[m]:[s] [Z]` を適用することにより `dateTime` の等価に変換されます。



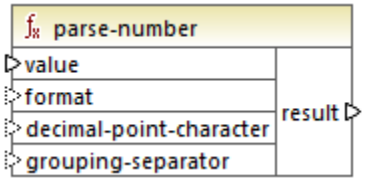
(XML と名前宣言を除く)結果は以下の通りです:

```
<FlightInformation>
   <FlightInfo departuredatetime="2019-12-24T19:43:04+02:00">
      <Station airportcode="KIV"/>
   </FlightInfo>
</FlightInformation>
```

サンプル: プロトコルバッファーにデータを書き込む を参照してください。

7.10.2.9 parse-number

指定されたフォーマットに従い文字列を小数に変換します。



言語

Built-in、C++、C#、Java。

パラメーター

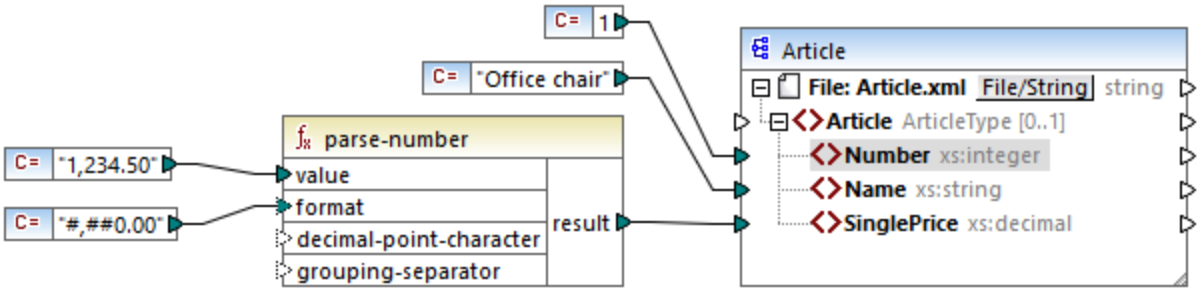
引数	説明
value	数値として変換される文字列。
format	オプションの引数。数値が現在書式処理されている方法を識別する書式文字列書式文字列は format-number 内で使用されたものと同じになります。

(C) 2015-2021 Altova GmbH

	デフォルトは"#,##0.#"になります。
decimal-point-character	オプションの引数。小数点文字として使用される文字を指定します。デフォルトは'.' 文字です。
grouping-separator	オプションの引数。数値グループを分割するために使用されるセパレータ/デミタを指定します。デフォルトは"," 文字です。

サンプル

以下のマッピングは書式マスク `#,##0.00` を使用して文字列の値 `"1,234.50"` を小数の等価に解析します。このマッピング内ではデフォルトの値が入力文字列の書式と一致するため**decimal-point-character** と**grouping-separator** 引数を接続する必要はありません。

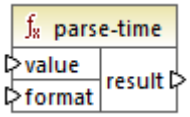


(XML と名前宣言を除く) マッピングの結果は以下の通りです:

```
<Article>
   <Number>1</Number>
   <Name>Office chair</Name>
   <SinglePrice>1234.5</SinglePrice>
</Article>
```

7.10.2.10 parse-time

文字列を `xs:time` 値に変換します。この関数は parse-dateTime 関数をベースにしたもので、その中でも日付コンポーネントが無視されます。



言語

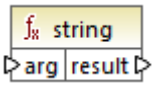
Built-in、C++、C#、Java。

パラメーター

引数	説明
value	必須の引数。変換される文字列の値を提供します。
format	必須の引数。フォーマット文字列を提供します。この引数は parse-dateTime 関数内の **format** 引数と同じ方法で使用されます。

7.10.2.11 string

入力値を文字列に変換します。ノードのテキストコンテンツを抽出するために関数を使用することもできます。XML 複合型が入力ノードとなる場合、全ての子要素も単一の文字列として出力されます。



言語

Built-in、C++、C#、Java、XQuery、XSLT 1.0、XSLT 2.0、XSLT 3.0。

パラメーター

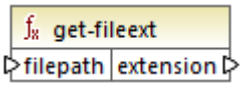
引数	説明
arg	必須の引数。変換される値を提供します。

7.10.3 core | file path functions (ファイルパス関数)

file path 関数によりマッピング内で更に処理するためにフォルダー、ファイル名、および拡張子などのファイルパスデータに直接アクセス、または操作することができます。これらの関数は MapForce にて使用可能な全ての言語で使用することができます。

7.10.3.1 get-fileext

ドット "." 文字を含むファイルパスの拡張子が削除されます。



言語

Built-in、C++、C#、Java、XQuery、XSLT 1.0、XSLT 2.0、XSLT 3.0。

(C) 2015-2021 Altova GmbH

パラメーター

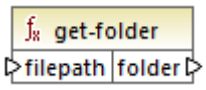
引数	説明
filepath	必須の引数。処理されるファイル パスを提供します。

サンプル

"c:¥data¥Sample.mfd" を引数として提供すると結果は `.mfd` です。

7.10.3.2 get-folder

ファイル パスから、スラッシュやバックスラッシュ(日本語環境では円マーク)を含むフォルダー名が返されます。



言語

Built-in、C++、C#、Java、XQuery、XSLT 1.0、XSLT 2.0、XSLT 3.0。

パラメーター

引数	説明
filepath	必須の引数。処理されるファイル パスを提供します。

サンプル

"c:¥data¥Sample.mfd" を引数として提供すると結果は `c:\data\` です。

7.10.3.3 main-mfd-filepath

マッピングデザインファイル(.mfd) のフル パスを返します。mfd ファイルが現在保存されている場合、空の文字列が返されます。

言語

Built-in、C++、C#、Java、XQuery、XSLT 1.0、XSLT 2.0、XSLT 3.0。

7.10.3.4 mfd-filepath

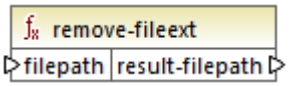
関数がメインマッピング内で呼び出された場合、main-mfd-filepath関数と同一の値(メインマッピングが含まれている.mfd ファイルのフルパス)が返されます。mfd ファイルが現在保存されていない場合、空の文字列が返されます。.mfd ファイルによりインポート済みのユーザー定義関数を呼び出すと、ユーザー定義関数の定義を含むインポート済みの.mfd ファイルのフルパスが返されます。

言語

Built-in、C++、C#、Java、XQuery、XSLT 1.0、XSLT 2.0、XSLT 3.0。

7.10.3.5 remove-fileext

ドット文字を含むファイルパスの拡張子が削除されます。



言語

Built-in、C++、C#、Java、XQuery、XSLT 1.0、XSLT 2.0、XSLT 3.0。

パラメーター

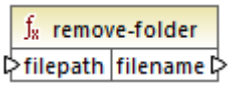
引数	説明
filepath	必須の引数。処理されるファイルパスを提供します。

サンプル

"c:¥data¥Sample.mfd" を引数として提供すると結果は`c:\data\Sample` です。

7.10.3.6 remove-folder

ファイルパスから、スラッシュやバックスラッシュ(日本語環境では円マーク)を含むフォルダー名が削除されます。



(C) 2015-2021 Altova GmbH

言語

Built-in、C++、C#、Java、XQuery、XSLT 1.0、XSLT 2.0、XSLT 3.0。

パラメーター

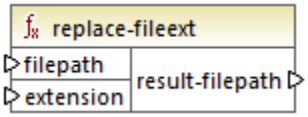
引数	説明
filepath	必須の引数。処理されるファイル パスを提供します。

サンプル

"c:¥data¥Sample.mfd" を引数として提供すると結果は `Sample.mfd` です。

7.10.3.7 replace-fileext

filepath パラメーターにより提供されたファイル パスの拡張子を **extension** パラメーターへの接続により提供されたものと置き換えます。



言語

Built-in、C++、C#、Java、XQuery、XSLT 1.0、XSLT 2.0、XSLT 3.0。

パラメーター

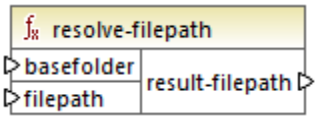
引数	説明
filepath	必須の引数。処理されるファイル パスを提供します。
extension	必須の引数。使用する新規の拡張子を提供します。

サンプル

"c:¥data¥Sample.log" を **filepath** として、".txt" を **extension** として提供すると、結果は `c:\data\Sample.txt` になります。

7.10.3.8 resolve-filepath

ベースフォルダーに対して相対的なファイル パスを解決します。関数は '.'(現在のディレクトリ) と '..'(親ディレクトリ) をサポートします。



言語

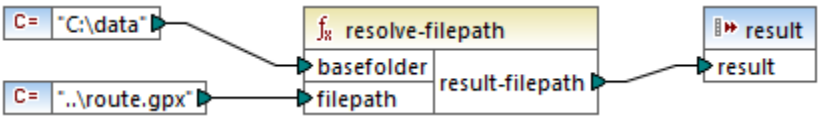
Built-in、C++、C#、Java、XQuery、XSLT 1.0、XSLT 2.0、XSLT 3.0。

パラメーター

引数	説明
basefolder	必須の引数。パスが解決される相対的なベースディレクトリを提供します。これは絶対的または相対的なパスであることができます。
filepath	必須の引数。解決される相対的なファイル パスを提供します。

サンプル

下のマッピングでは相対的なファイル パス **..\route.gpx** は **C:\data** ディレクトリに対して解決されます。



マッピングの結果は `C:\route.gpx` です。

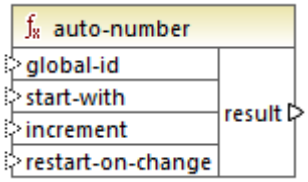
7.10.4 core | generator functions (ジェネレーター関数)

core / generator 関数ライブラリには値を生成する関数が含まれています。

7.10.4.1 auto-number

(例えば、1,2,3,4, ... などの) シーケンスで整数を生成します。インクリメント値である最初の整数と他のオプションをパラメーターを使用して設定することが可能です。

(C) 2015-2021 Altova GmbH



生成されたマッピングコードにより呼び出される関数の順序は定義されていません。MapForce ではキャッシュされた計算結果を再利用するか、任意の順序で条件式を再評価する必要があります。同じ入力パラメーターを使用して複数回呼び出されると他の関数と異なり `auto-number` 関数は異なる結果を返します。このため `auto-number` 関数は注意して使用することが奨励されます。一部の場合、同じ結果を達成するために position 関数を代わりに使用することができます。

言語

Built-in、C++、C#、Java、XQuery、XSLT 1.0、XSLT 2.0、XSLT 3.0。

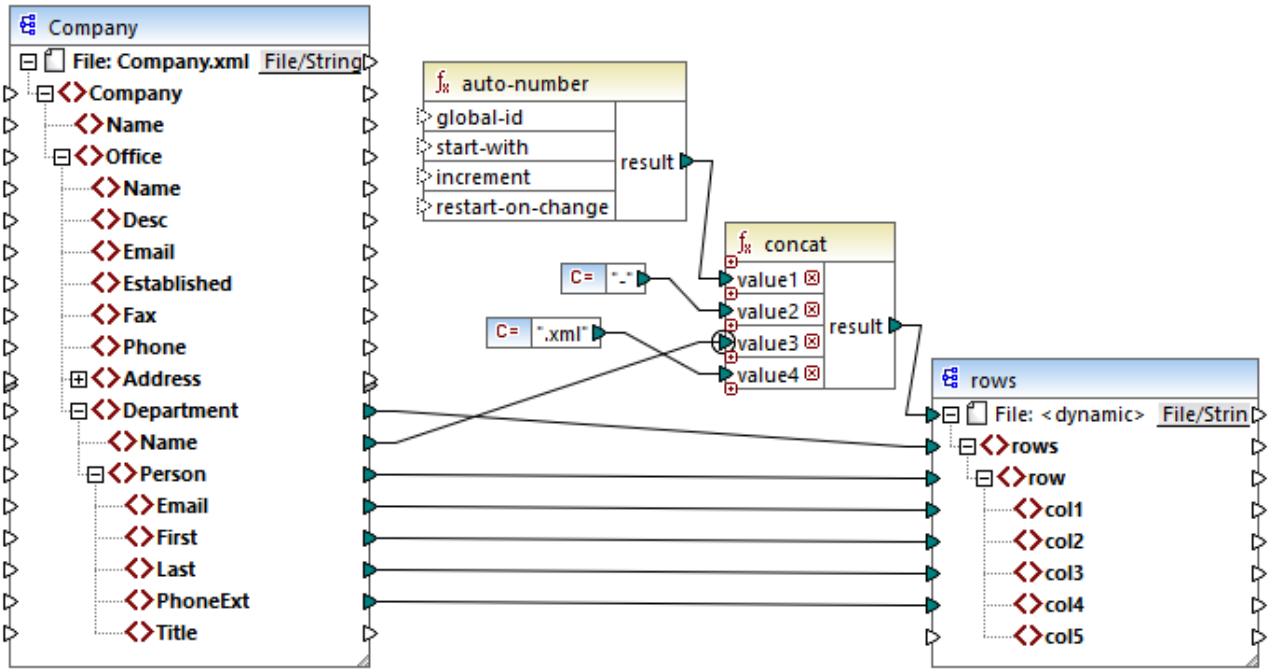
パラメーター

引数	説明
global-id	任意のパラメーターです。マッピングデザインが複数の `auto-number` 関数を含む場合複製（重複する）番号と共にシーケンスを生成します。全ての `auto-number` 関数がお互いを考慮するようにし、重複しないシーケンスを生成するには、各 `auto-number` 関数の（例えば、定数などの）**global-id** 入力に共通の文字列を接続します。
start-with	任意のパラメーターです。生成されたシーケンスが開始する整数を指定します。デフォルト の値は**1** です。
increment	任意のパラメーターです。インクリメント値を指定します。デフォルト の値は**1** です。
restart-on-change	任意のパラメーターです。接続されているアイテムのコンテンツが変更されると、**start-with** へのカウンターをリセットします。

サンプル

以下のマッピングは**ParentContext.mfd** マッピングのバリエーションで、サンプル: 親コンテキストの変更で詳細に説明されています。

下で説明されているマッピングの目的はソース XML ファイル内でそれぞれの部署に つずつ複数の XML ファイルを生成することです。同じ名前を持つ部署が存在します（これはそれぞれが異なる親部署に属するためです）。このために、生成されたそれぞれのファイル名は連番で始まる必要があります、例えば、**1-Administration.xml**、**2-Marketing.xml** などです。



マッピングの目的を達成するには `auto-number` 関数が使用されます。この関数の結果はダッシュ文字の後に部署名および ".xml" 文字列が表示されている形で連結されています。重要な点は `concat` 関数の3番目のパラメーター（部署名）に priority context が適用されていることです。これには `auto-number` 関数が各部署のコンテキスト内で呼び出され、必要とされる連番の値を生成する効果があります。優先コンテキストが使用されない場合 `auto-number` 関数が（コンテキストが存在しない場合）番号 1 を生成し続け、複製ファイル名が結果として生成されます。

7.10.5 core | logical functions（論理関数）

logical 関数は（通常）入力されたデータを比較してブール値 [true] または [false] を返すために使用されます。通常、フィルターを使用してサブセットを渡す前段階で、データを評価するために使用されます。ほぼすべての理論関数には以下の構造が存在します:

入力パラメーター= **a** | **b** または **value1** | **value2**
出力パラメーター= 結果

評価結果は、入力の値と比較に使用されたデータ型により異なります。例: 整数の値 **4** と **12** の [より小さい] 比較は、4 が 12 より小さいため、ブール値 [true] を返します。2 つの入力パラメーターが文字列の値 `[4]` と `[12]` を含む場合、[4] はアルファベットの順序でオペランド（12）の最初の文字 [1] よりも大きいため、構文的分析はの出力値 "false" を返します。

全ての入力値が同じデータ型の場合、比較は共通の型のために行われます。入力値が異なる型の場合、（例: 整数と文字列 または 文字列と日付）そして、使用されるデータ型は、2 つの入力型のうち 最も一般的な（最も制限のない）入力データ型です。

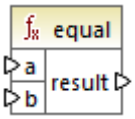
具体的には、異なる型の2 つの値を比較する前に、すべての入力の値は、共通のデータ型に変換されます。前のサンプルで説明すると、データ型 文字列 は整数より制限が少ないことがわかります。整数の値 **4** を文字列 `[12]` と比較し、整数の値 **4** を文字列 `[4]` に変換し、文字列 `[12]` と比較します。

メモ Logical 関数を null 値の存在をテストするために使用することはできません。null 値を logical 関数の引数として与えると、null 値が返されます。null 値の扱い方に関しての詳しい情報は、Nil 値 / Nillable を参照してください。

(C) 2015-2021 Altova GmbH

7.10.5.1 equal

a が *b* と等価の場合、ブール値 **true** が返され、それ以外の場合 **false** となります。



言語

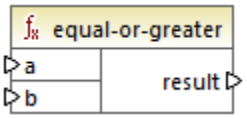
Built-in、C++、C#、Java、XQuery、XSLT 1.0、XSLT 2.0、XSLT 3.0。

パラメーター

引数	説明
a	必須のパラメーターです。比較する最初の値を提供します。
b	必須のパラメーターです。比較する2番目の値を提供します。

7.10.5.2 equal-or-greater

a が *b* と等価または大きい場合、ブール値 **true** が返され、それ以外の場合 **false** となります。



言語

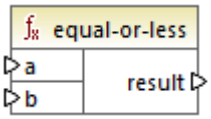
Built-in、C++、C#、Java、XQuery、XSLT 1.0、XSLT 2.0、XSLT 3.0。

パラメーター

引数	説明
a	必須のパラメーターです。比較する最初の値を提供します。
b	必須のパラメーターです。比較する2番目の値を提供します。

7.10.5.3 equal-or-less

a が *b* と等価または小さい場合、ブール値 **true** が返され、それ以外の場合 **false** となります。



言語

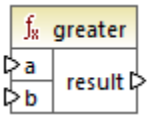
Built-in、C++、C#、Java、XQuery、XSLT 1.0、XSLT 2.0、XSLT 3.0。

パラメーター

引数	説明
a	必須のパラメーターです。比較する最初の値を提供します。
b	必須のパラメーターです。比較する2番目の値を提供します。

7.10.5.4 greater

a が *b* より大きい場合、ブール値 **true** が返され、それ以外の場合 **false** となります。



言語

Built-in、C++、C#、Java、XQuery、XSLT 1.0、XSLT 2.0、XSLT 3.0。

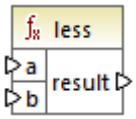
パラメーター

引数	説明
a	必須のパラメーターです。比較する最初の値を提供します。
b	必須のパラメーターです。比較する2番目の値を提供します。

(C) 2015-2021 Altova GmbH

7.10.5.5 less

a が *b* より小さい場合、ブール値 **true** が返され、それ以外の場合 **false** となります。



言語

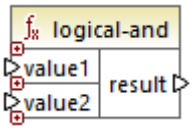
Built-in、C++、C#、Java、XQuery、XSLT 1.0、XSLT 2.0、XSLT 3.0。

パラメーター

引数	説明
a	必須のパラメーターです。比較する最初の値を提供します。
b	必須のパラメーターです。比較する2番目の値を提供します。

7.10.5.6 logical-and

各入力値が true の場合のみブール値 **true** を返し、それ以外の場合は **false** を返します。結果を他の `logical-and` 関数に接続することができ、このために全てが **true** を返すようにテストするために、論理 AND と任意の数の条件をジョインすることができます。また、この関数は追加の引数を取るように拡張することができます。関数の引数の追加および削除を参照してください。



言語

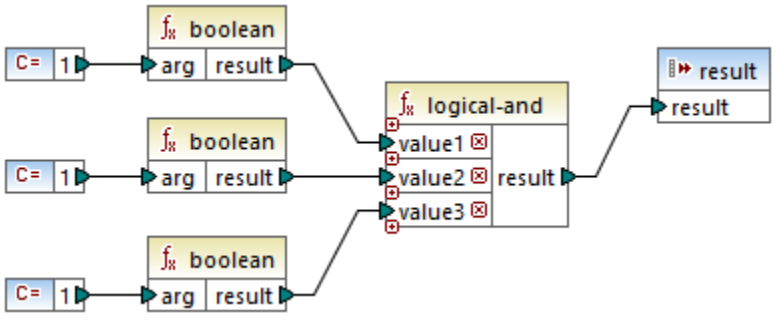
Built-in、C++、C#、Java、XQuery、XSLT 1.0、XSLT 2.0、XSLT 3.0。

パラメーター

引数	説明
value1	必須のパラメーターです。比較する最初の値を提供します。
value2	必須のパラメーターです。比較する2番目の値を提供します。

サンプル

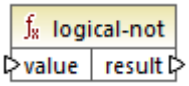
下で説明されているマッピングは`logical-and` への全ての入力値が**true** であるため**true** を返します。入力値のいずれかが**false** の場合、マッピングの結果は**false** になります。



サンプル: ルックアップと連結を参照してください。

7.10.5.7 logical-not

論理状態/結果を反転します。例えば *値* が**true** の場合、関数の結果はfalse です。*値* が**false** の場合、関数の結果はtrue です。



言語

Built-in、C++、C#、Java、XQuery、XSLT 1.0、XSLT 2.0、XSLT 3.0。

パラメーター

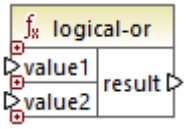
引数	説明
value	必須のパラメーターです。入力パラメーターを提供します。

7.10.5.8 logical-or

両方の入力値をブール型に揃える必要があります。少なくとも入力値の1つが**true** の場合、結果は**true** になります。それ以外の場合結果は**false** になります。

この関数は追加の引数を取るように拡張することができます。関数の引数の追加または削除を参照してください。

(C) 2015-2021 Altova GmbH



言語

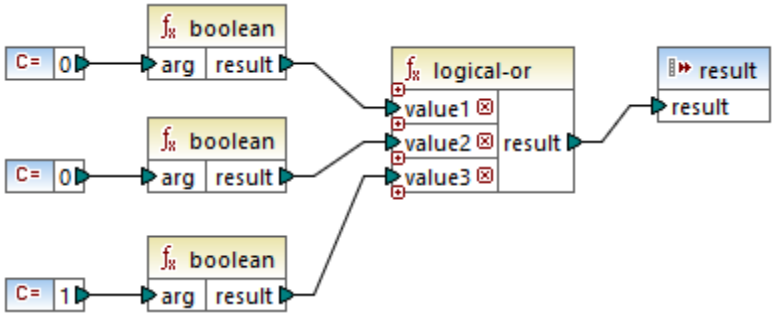
Built-in、C++、C#、Java、XQuery、XSLT 1.0、XSLT 2.0、XSLT 3.0。

パラメーター

引数	説明
value1	必須のパラメーターです。比較する最初の値を提供します。
value2	必須のパラメーターです。比較する2番目の値を提供します。

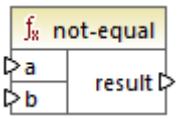
サンプル

少なくとも1つの関数の引数が**true** のため、下のマッピングの結果は**true** です。



7.10.5.9 not-equal

a が *b* と等価ではない場合、ブール値 **true** が返され、それ以外の場合 **false** となります。



言語

Built-in、C++、C#、Java、XQuery、XSLT 1.0、XSLT 2.0、XSLT 3.0。

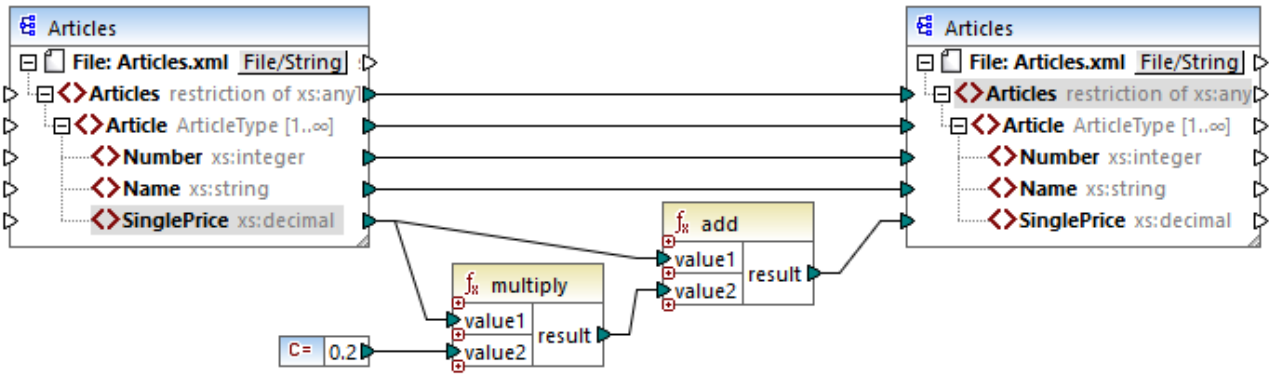
パラメーター

引数	説明
a	必須のパラメーターです。比較する最初の値を提供します。
b	必須のパラメーターです。比較する2番目の値を提供します。

7.10.6　core | math functions（数学関数）

データを使った初歩的な計算を行うために数学関数を利用することができます。これら関数を使用して期間や`datetime`（日時）の計算はできないという点に注意してください。

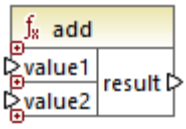
ほぼすべての数学関数は数学オペレーションのオペランドである2つの入力パラメーター（**value1**、**value2**）を取ります。更に処理するために入力値は`decimal`へ自動的に変換され、処理が行われます。数学関数の結果も`decimal`型です。



上に示されるサンプルでは、ターゲットコンポーネントにマッピングされた各製品の価格に対して20%の消費税が上乗せされます。

7.10.6.1　add

value1を**value2**に追加して結果を小数の値で返します。この関数は追加の引数を取るように拡張することができます。関数の引数の追加または削除を参照してください。



言語

Built-in、C++、C#、Java、XQuery、XSLT 1.0、XSLT 2.0、XSLT 3.0。

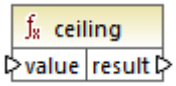
(C) 2015-2021 Altova GmbH

パラメーター

引数	説明
value1	必須のパラメーターです。最初のオペランドを提供します。
value2	必須のパラメーターです。2番目のオペランドを提供します。

7.10.6.2 ceiling

value より大きい、または、等しい一番小さい整数を返します。



言語

Built-in、C++、C#、Java、XQuery、XSLT 1.0、XSLT 2.0、XSLT 3.0。

パラメーター

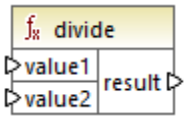
引数	説明
value	必須のパラメーターです。関数の入力パラメーターを提供します。

サンプル

入力値は**11.2** で`ceiling` 関数を適用して結果を**11.2** より大きい一番小さい整数である**12** にします。

7.10.6.3 divide

value1 を**value2** で割り、結果を小数として返します。除算結果の精度はターゲット言語により変化します。round-precision 関数を使用することで、結果の精度を定義することができます。



言語

Built-in、C++、C#、Java、XQuery、XSLT 1.0、XSLT 2.0、XSLT 3.0。

パラメーター

引数	説明
value1	必須のパラメーターです。最初のオペランドを提供します。
value2	必須のパラメーターです。2番目のオペランドを提供します。

7.10.6.4 floor

value より小さい、または、等しい一番大きい整数を返します。

言語

Built-in、C++、C#、Java、XQuery、XSLT 1.0、XSLT 2.0、XSLT 3.0。

パラメーター

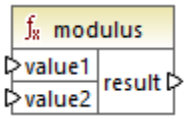
引数	説明
value	必須のパラメーターです。関数の入力パラメーターを提供します。

サンプル

入力値は**11.7** で`floor` 関数を適用して結果を**11.7** より小さい一番大きい整数である**11** にします。

7.10.6.5 modulus

value1 を**value2** で割った値の余りが返されます。



言語

Built-in、C++、C#、Java、XQuery、XSLT 1.0、XSLT 2.0、XSLT 3.0。

(C) 2015-2021 Altova GmbH

パラメーター

引数	説明
value1	必須のパラメーターです。最初のオペランドを提供します。
value2	必須のパラメーターです。2番目のオペランドを提供します。

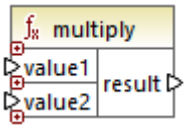
サンプル

入力値が**1.5** と**1** の場合、`modulus` 関数の結果は**0.5** です。説明は`1.5 / 1` はあまりとして**0.5** を返します。

入力値が**9** と**3** の場合 `9 / 3` が結果のため、余りはなく結果は**0** です。

7.10.6.6 multiply

value1 を**value2** と乗算し結果を小数として返します。



言語

Built-in、C++、C#、Java、XQuery、XSLT 1.0、XSLT 2.0、XSLT 3.0。

パラメーター

引数	説明
value1	必須のパラメーターです。最初のオペランドを提供します。
value2	必須のパラメーターです。2番目のオペランドを提供します。

7.10.6.7 round

一番近い整数に値を丸め処理します。値が2つの整数の中央にある場合、[正の無限大に向けて半分に丸める]アルゴリズムが使用されます。例えば、値 [10.5] は [11] に丸められ端数処理され、値 [-10.5] は [-10] に丸められ端数処理されます。

言語

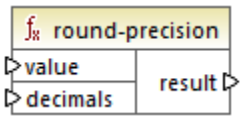
Built-in、C++、C#、Java、XQuery、XSLT 1.0、XSLT 2.0、XSLT 3.0。

パラメーター

引数	説明
value	必須のパラメーターです。関数の入力パラメーターを提供します。

7.10.6.8 round-precision

N が**小数**の引数である箇所で、入力値を *N* 小数の桁に丸め端数処理します。



言語

Built-in、C++、C#、Java。

パラメーター

引数	説明
value	必須のパラメーターです。関数の入力パラメーターを提供します。
decimals	必須のパラメーターです。丸めて端数処理する小数の桁を指定します。

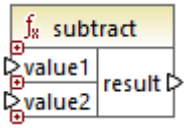
サンプル

値 **2.777777** を 2 桁の小数に丸めると **2.78** になります。値 **0.1234** を 3 桁の小数に丸めると **0.123** になります。

サンプル: プロトコルバッファーからデータを読み取るを参照してください。

7.10.6.9 subtract

value2 を **value1** から減算し、結果を小数値として返します。



(C) 2015-2021 Altova GmbH

言語

Built-in、C++、C#、Java、XQuery、XSLT 1.0、XSLT 2.0、XSLT 3.0。

パラメーター

引数	説明
value1	必須のパラメーターです。最初のオペランドを提供します。
value2	必須のパラメーターです。2番目のオペランドを提供します。

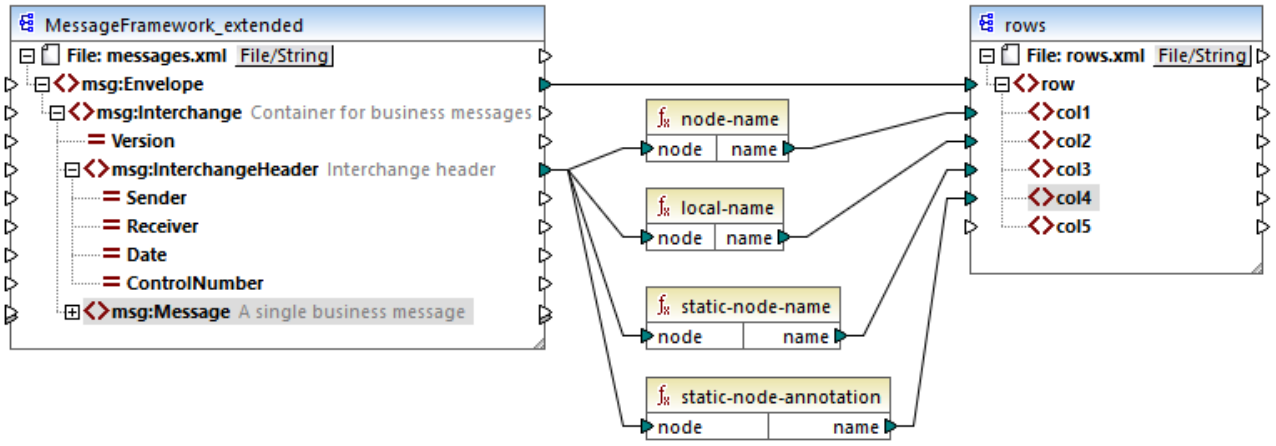
7.10.7 core | node functions (ノード関数)

core | node functions ライブラリからの関数は、名前または注釈などのマッピングコンポーネント上のノードに関する情報についてのアクセスまたはnillable 要素の処理を許可します。Nil 値 / Nillable も参照してください。

ノード関数を必要とせずにノード名にアクセスする代替法が存在することに注意してください。ノード名のマッピングを参照してください。

下で表示されているマッピングはソースXML ファイルの**msg:InterchangeHeader** ノードから情報を取得するノード関数のいくつかを表示しています。具体的には、以下の情報が抽出されます:

1. `node-name` 関数はノードプレフィックスを含むノードの修飾名を返します。
2. `local-name` 関数はローカルのパートのみを返します。
3. `static-node-name` 関数は`node-name` 関数に類似したものを返します。XSLT 1.0 でも使用することができます。
4. `static-node-annotation` 関数はスキーマ内で定義された通り要素の注釈を取得します。

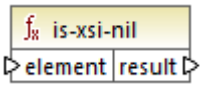


(XML と名前宣言を除く) マッピングの出力は以下の通りです:

```
<row>
   <col1>msg:InterchangeHeader</col1>
   <col2>InterchangeHeader</col2>
   <col3>msg:InterchangeHeader</col3>
   <col4>Interchange header</col4>
</row>
```

7.10.7.1 is-xsi-nil

element 要素の`xsi:nil` 属性が**[true]** に設定されている場合 **[true]** を返します。



言語

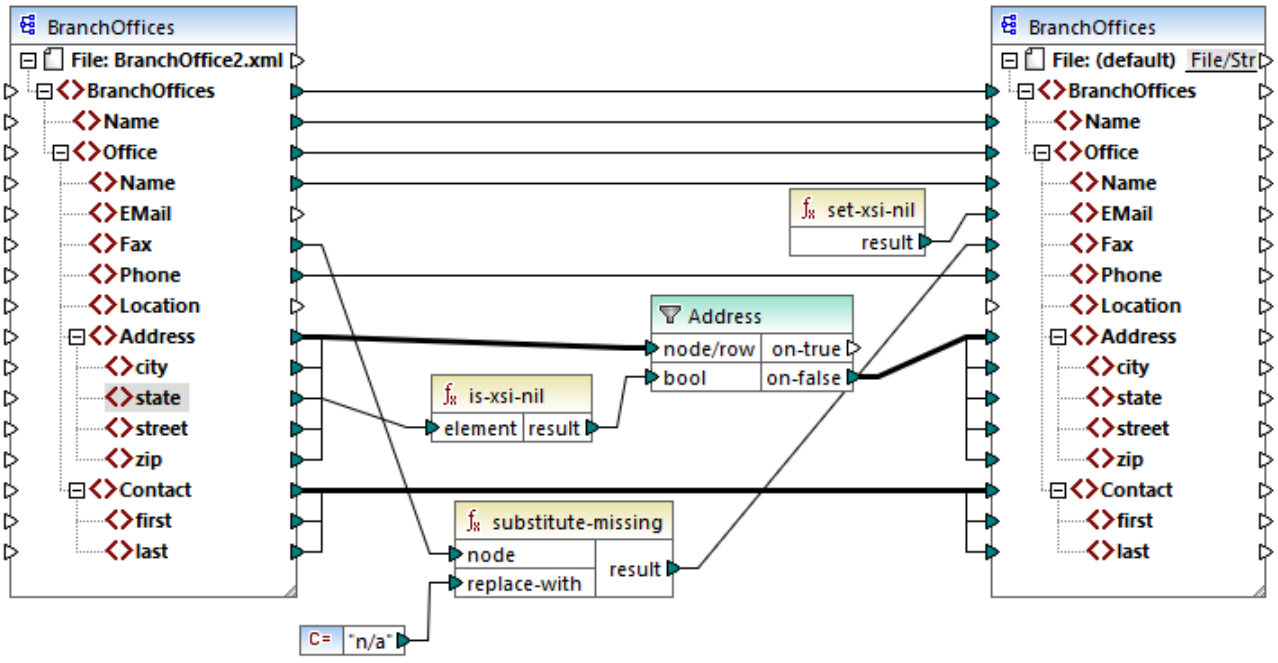
Built-in、C++、C#、Java、XQuery、XSLT 1.0、XSLT 2.0、XSLT 3.0。

パラメーター

引数	説明
element	必須のパラメーターです。チェックされるソースノードに接続されている必要があります。

サンプル

下に表示されるマッピングデザインはデータをソースからターゲット XML ファイルに条件付きでコピーし `is-xsi-nil` を含む複数の関数の使用法を表示しています。このマッピングは**HandlingXsiNil.mfd** と呼ばれくマイドキュメント **>\Altova\MapForce2021\MapForceExamples** ディレクトリ内で見つけることができます。



上で説明される `is-xsi-nil` 関数は`xsi:nil` 属性がソースファイル内の**state** アイテムのために[true] かどうかをチェックします。この属性が[false] の場合フィルターは親 **Address** 要素をターゲットにコピーします。(XML と名前宣言を除く)ソース XML ファイルは以下の通りです:

(C) 2015-2021 Altova GmbH

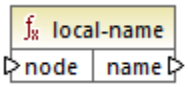
```
<BranchOffices>
   <Name>Nanonull</Name>
   <Office>
      <Name>Nanonull Research Outpost</Name>
      <EMail>sp@nanonull.com</EMail>
      <Fax xsi:nil="true"/>
      <Phone>+8817 3141 5926</Phone>
      <Address>
         <city>South Pole</city>
         <state xsi:nil="true"/>
         <street xsi:nil="true"/>
         <zip xsi:nil="true"/>
      </Address>
      <Contact>
         <first>Scott</first>
         <last>Amundsen</last>
      </Contact>
   </Office>
</BranchOffices>
```

ソース内に**Address** が一つのみ存在し、`xsi:nil` 属性が**state** 要素のために[true] に設定されているために、にマッピングの結果は**Address** はターゲットにコピーされません。この結果、マッピングの出力は以下の通りです:

```
<BranchOffices>
   <Name>Nanonull</Name>
   <Office>
      <Name>Nanonull Research Outpost</Name>
      <EMail xsi:nil="true"/>
      <Fax>n/a</Fax>
      <Phone>+8817 3141 5926</Phone>
      <Contact>
         <first>Scott</first>
         <last>Amundsen</last>
      </Contact>
   </Office>
</BranchOffices>
```

7.10.7.2 local-name

ノードのローカル名を返します。node-name 関数とは異なり **local-name** はノードのプレフィックスを返しません。ノードがプレフィックスを持たない場合 **local-name** と**node-name** は同じ値を返します。



言語

Built-in、C++、C#、Java、XQuery、XSLT 1.0、XSLT 2.0、XSLT 3.0。

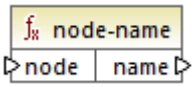
パラメーター

引数	説明
node	必須のパラメーターです。この入力をノード名を取得するノードに接続してください。

7.10.7.3 node-name

接続されているノードの修飾名（QName）を返します。ノードがXML **text()** ノードの場合、空のQname が返されます。この関数は名前付きのノードに対してのみ使用することができます。（ `fn:node-name` を呼び出す）XSLT 2.0 をターゲット言語としている場合、関数は名前を持っていないノードに対して空のシーケンスを返します。

メモ ノード名の取得はノード、データベーステーブル、またはフィールド、XBRL、Excel、JSON またはプロトコールバッファーフィールドは[ファイル入力]フィールドのためにはサポートされていません。



言語

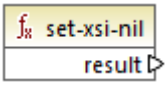
Built-in、C++、C#、Java、XQuery、XSLT 2.0、XSLT 3.0。

パラメーター

引数	説明
node	必須のパラメーターです。この入力をノード名を取得するノードに接続してください。

7.10.7.4 set-xsi-nil

ターゲットノードを xsi:nil にセットします。



言語

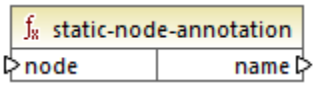
Built-in、C++、C#、Java、XQuery、XSLT 2.0、XSLT 3.0。

7.10.7.5 static-node-annotation

接続されたノードの注釈と共に文字列を返します。入力は以下である必要があります: (i) ソースコンポーネントノード、または(ii) 呼び出しマッピング内のノードに直接接続されているパラメーターに直接接続されている型 インラインのユーザー定義関数。

(C) 2015-2021 Altova GmbH

接続は直接的である必要があります。フィルター、または標準（非[インライン]）ユーザー定義関数をパススルーすることはできません。これは、接続されたノードから取得されたテキストの生成時に置き換えられた擬似関数で、全ての言語で使用することができます。



言語

Built-in、C++、C#、Java、XQuery、XSLT 1.0、XSLT 2.0、XSLT 3.0。

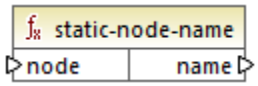
パラメーター

引数	説明
node	必須のパラメーターです。この入力を注釈を取得するノードに接続してください。

7.10.7.6 static-node-name

接続されたノードの名前と共に文字列を返します。入力は以下である必要があります: (i) ソースコンポーネントノード、または(ii) 呼び出しマッピング内のノードに直接接続されているパラメーターに直接接続されている型 インラインのユーザー定義関数。

接続は直接的である必要があります。フィルター、または 非インラインユーザー定義関数をパススルーすることはできません。これは、接続されたノードから取得されたテキストの生成時に置き換えられた擬似関数で、全ての言語で使用することができます。



言語

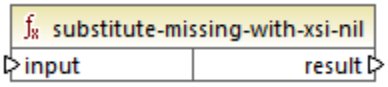
Built-in、C++、C#、Java、XQuery、XSLT 1.0、XSLT 2.0、XSLT 3.0。

パラメーター

引数	説明
node	必須のパラメーターです。この入力をノード名を取得するノードに接続してください。

7.10.7.7 substitute-missing-with-xsi-nil

単純型コンテンツを持つノードでは、この関数はソースコンポーネントの不足する値（またはNULL 値）をターゲットノード内で `xsi:nil` 属性と置き換えます。



言語

Built-in、C++、C#、Java、XQuery、XSLT 2.0、XSLT 3.0。

パラメーター

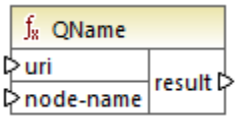
引数	説明
input	必須のパラメーターです。この入力をノード名を取得するノードに接続してください。

7.10.8 core | QName functions (QName 関数)

QName 関数は、XML ドキュメント内の修飾名 (QName) を操作する方法を提供します。

7.10.8.1 QName

名前空間 URI とローカルパートから QName を構築します。Qname をターゲットコンポーネント内で作成するために この関数を使用します。定数関数により **uri** と **node-name** パラメーターを提供することができます。



言語

Built-in、C++、C#、Java、XQuery、XSLT 2.0、XSLT 3.0。

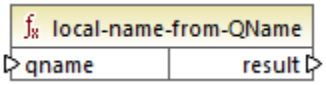
パラメーター

名前	説明
uri	必須。URI を提供します。
node-name	必須。ノードの名前を提供します。

7.10.8.2 local-name-from-QName

型 `xs:QName` の値からローカル名の部分が抽出されます。*node* のローカル名を返す local-name 関数とは異なり、この関数は **qname** 入力に接続されているアイテムの *content* を処理します。

(C) 2015-2021 Altova GmbH



言語

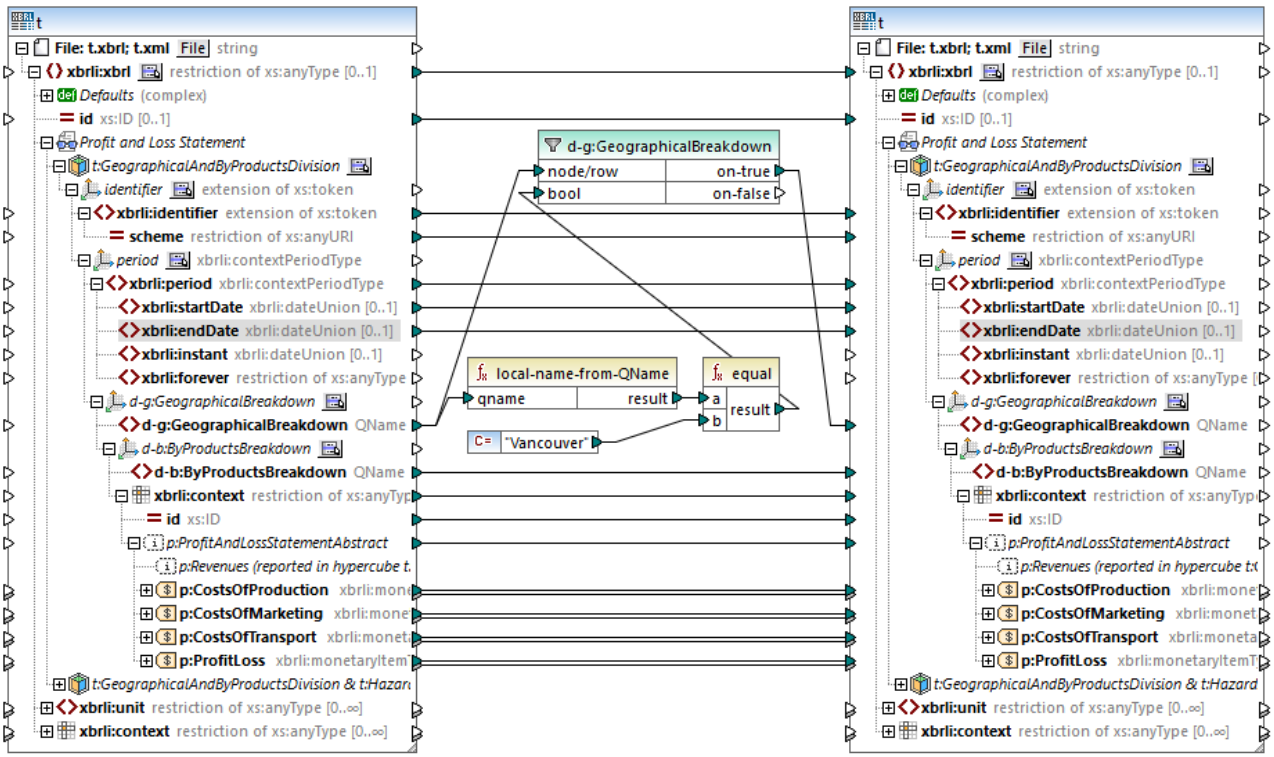
Built-in、C++、C#、Java、XQuery、XSLT 2.0、XSLT 3.0。

パラメーター

名前	説明
qname	必須。型 `xs:QName` の関数入力値を提供します。

サンプル

ハイパーキューブを含むXBRL インスタンスドキュメントをマップする際この関数は便利です。



上のマッピングでは、明示的なメンバー**d-g:Vancouver** のコンテンツのローカル名が[Vancouver] に等しい箇所で、これらのファクトがフィルターされています。下のコードリスティングで表示されているとおりメンバーのコンテンツはQname 自身であることに注意してください！

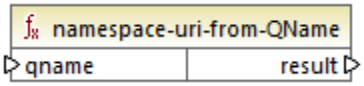
```
<xbrli:entity>
    <xbrli:identifier scheme="http://nasdaq.com/ticker">ACME2</xbrli:identifier>
```

```
    <xbrli:segment>
        <xbrldi:explicitMember dimension="d-b:ByProductsBreakdown">d-
b:Water</xbrldi:explicitMember>
        <xbrldi:explicitMember dimension="d-g:GeographicalBreakdown">d-
g:Vancouver</xbrldi:explicitMember>
    </xbrli:segment>
</xbrli:entity>
```

マッピングの実行後、ディメンション **GeographicalBreakdown** に属する全てのファクトはフィルターされターゲットコンポーネントに パスされます。

7.10.8.3 namespace-uri-from-QName

引数として与えられている Qname 値の名前空間 URI 部分が返されます。



言語

Built-in、C++、C#、Java、XQuery、XSLT 2.0、XSLT 3.0。

パラメーター

名前	説明
qname	必須。関数の入力パラメーターを提供します。

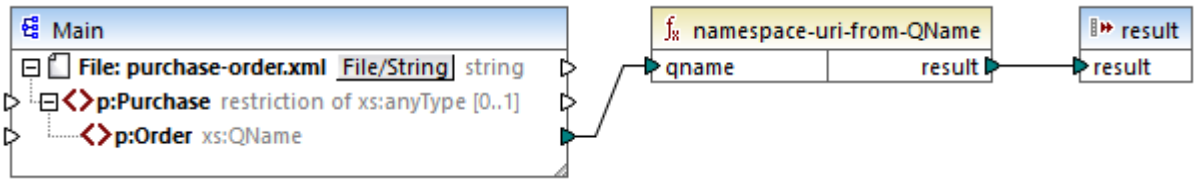
サンプル

次のXML ファイルにはQName 値 **o:name** が含まれています。プレフィックス [o] は名前空間 `http://NamespaceTest.com/Order` にマップされています。

```
<?xml version="1.0" encoding="utf-8"?>
<p:Purchase xsi:schemaLocation="http://NamespaceTest.com/Purchase Main.xsd"
          xmlns:p="http://NamespaceTest.com/Purchase"
          xmlns:o="http://NamespaceTest.com/Order"
          xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
          <p:Order>o:name</p:Order>
</p:Purchase>
```

QName 値を処理し、名前空間 URI を取得するマッピングは下で表示されています:

(C) 2015-2021 Altova GmbH



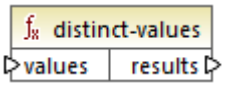
このマッピングの出力は`http://NamespaceTest.com/Order`です。

7.10.9 core | sequence functions（シーケンス関数）

シーケンス関数は入力 シーケンス の処理とそのコンテンツのグループ化を許可します。

7.10.9.1 distinct-values

values 入力に接続されている値のシーケンスを処理し、シーケンスとして一意の値のみを返します。これはシーケンスにて重複した値を取り除き、ユニークなアイテムをターゲットコンポーネントへコピーする際に役に立ちます。



言語

Built-in、C++、C#、Java、XQuery、XSLT 2.0、XSLT 3.0。

パラメーター

名前	説明
values	この入力はゼロまたは他の値を提供するマッピングアイテムからの接続を受け取る必要があります。例えば、接続はソースXML アイテム、CSV フィールド、データベースレコードなど . から始まっている場合があります。

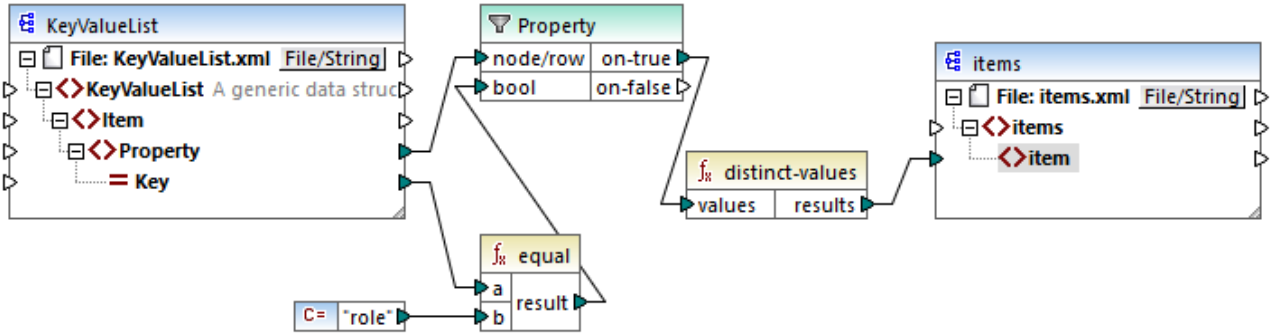
サンプル

次のXML ファイルには架空の会社の社員に関する情報が含まれています。社員の一部は同じロールを有しているため、[ロール]属性ロールには複数の値が含まれています。例えば[Loby Matise] と[Susi Sanna] は同じロール[Support] を持っています。

```
<?xml version="1.0" encoding="UTF-8"?>
<KeyValueList xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="KeyValueList.xsd">
   <Item>
      <Property Key="role">Manager</Property>
      <Property Key="First">Vernon</Property>
      <Property Key="Last">Callaby</Property>
```

```
   </Item>
   <Item>
      <Property Key="role">Programmer</Property>
      <Property Key="First">Frank</Property>
      <Property Key="Last">Further</Property>
   </Item>
   <Item>
      <Property Key="role">Support</Property>
      <Property Key="First">Loby</Property>
      <Property Key="Last">Matise</Property>
   </Item>
   <Item>
      <Property Key="role">Support</Property>
      <Property Key="First">Susi</Property>
      <Property Key="Last">Sanna</Property>
   </Item>
</KeyValueList>
```

このXML ファイル内で発生するすべての 一意のロール名のリストを抽出する必要があると仮定します。これは以下のマッピング内で達成することができます:



上記のマッピングでは次が発生します:

- ソースXML ファイルからの各 **Property** 要素はフィルターにより処理されます。
- フィルターの**bool** 入力への接続は**Key** 属性が[role] に等しい個所のみで **Property** 要素がターゲットコンポーネントに提供されることを保証します。文字列 [role] は定数により与えられます。(フィルターの条件を満たす [Support] プロパティが2つ存在するために) フィルターの出力はまだこの時点では複製を作成することに注意してください。
- フィルターにより作成されるシーケンスは複製値を除外する `distinct-values` 関数により処理されます。

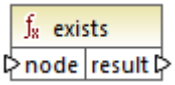
この結果 (XML と名前宣言を除く) マッピングの出力は以下の通りです:

```
<items>
   <item>Manager</item>
   <item>Programmer</item>
   <item>Support</item>
</items>
```


(C) 2015-2021 Altova GmbH

7.10.9.2 exists

接続済みのノードが存在する場合は**true** を返します。それ以外の場合は**false** を返します。ブール値を返すためこの関数は子要素または属性を持つ（または持たない）レコードのみをフィルターするために通常 フィルター と共に使用されます。



言語

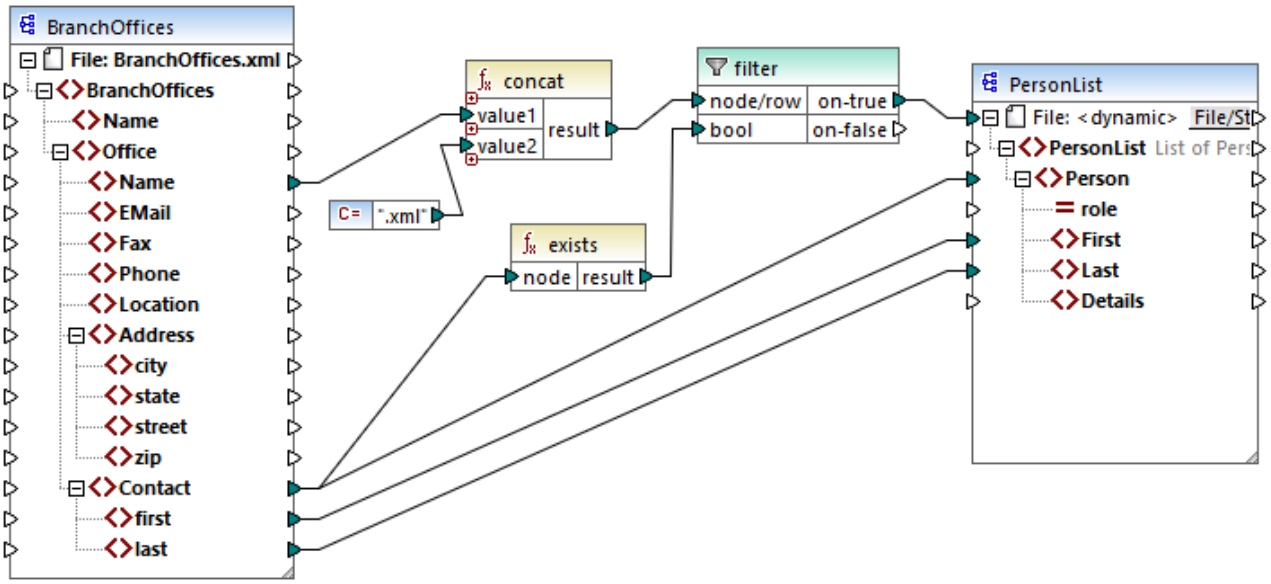
Built-in、C++、C#、Java、XQuery、XSLT 1.0、XSLT 2.0、XSLT 3.0。

パラメーター

名前	説明
node	存在を確認するノード。

サンプル

次のマッピングは`exists` 関数を使用してデータをフィルターする過程を表示しています。このマッピングは**PersonListsForAllBranchOffices.mfd** と呼ばれ＜マイドキュメント＞¥Altova¥MapForce2021¥MapForceExamples¥ディレクトリ内で見つけることができます。



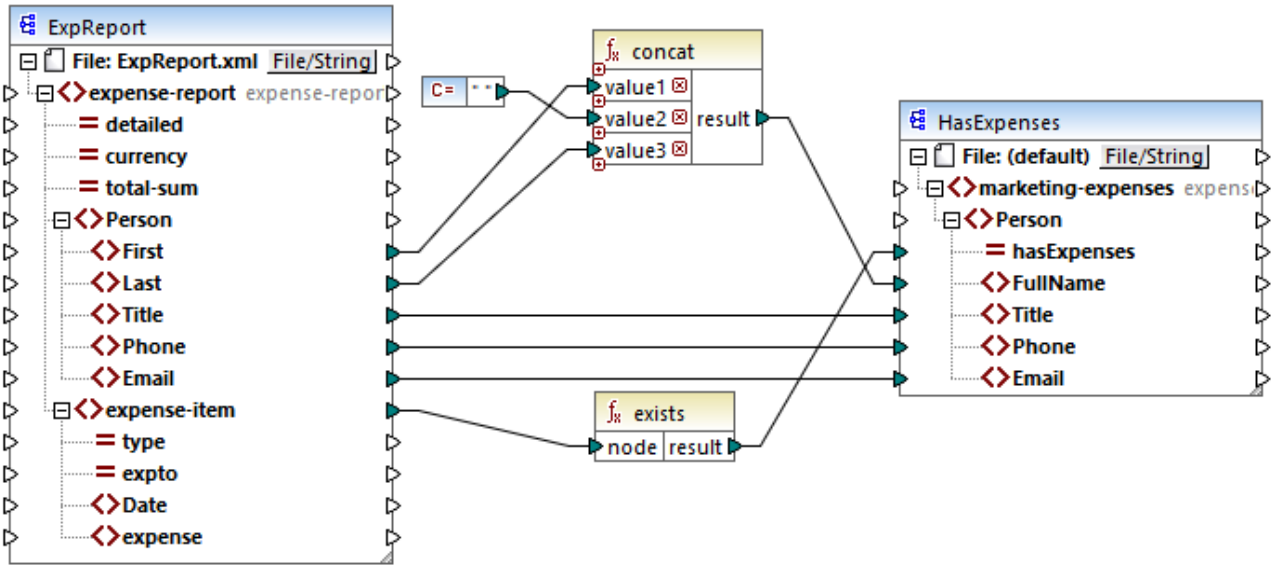
PersonListsForAllBranchOffices.mfd

ソースファイル**BranchOffices.xml** 内には3つの**Office** 要素が存在します。オフィスの一つに**Contact** 子要素が存在しないことに気が付きます。マッピングのゴールは複数存在します:

a) 各オフィスのために、そのオフィス内に存在する連絡先のリストを抽出する
b) 各オフィスのためにオフィスと同じ名前を持つ個別のXML ファイルを作成する
c) オフィスに連絡先が存在しない場合 XML ファイルを生成しない

この目的を達成するために、フィルターがマッピングに追加されました。**Contact** アイテムが少なくとも1つが存在する箇所で フィルターは **Office** アイテムをターゲットに パスします。ブール値条件は`exists` 関数により提供されます。関数の結果がtrue の場合、ターゲットファイル名を生成するためにオフィスの名前は文字列 `.xml` と連結されます。マッピングからのファイル名の生成の詳細に関しては、複数の入力または出力ファイルを動的に処理を参照してください。

もう一つの例は次のマッピングです: **<マイドキュメント>\Altova\MapForce2021\MapForceExamples\HasMarketingExpenses.mfd**。**expense-item** がソースXML にて存在する場合、ターゲット XML/スキーマファイルの**hasExpenses** 属性が**true** となります。

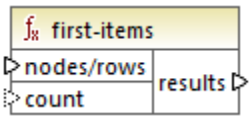


HasMarketingExpenses.mfd

例: ノードが存在しない場合の例外を参照してください。

7.10.9.3 first-items

N が**count** パラメーターにより提供される個所で、入力シーケンスの最初の *N* 個のアイテムを返します。



言語

Built-in、C++、C#、Java、XQuery、XSLT 2.0、XSLT 3.0。

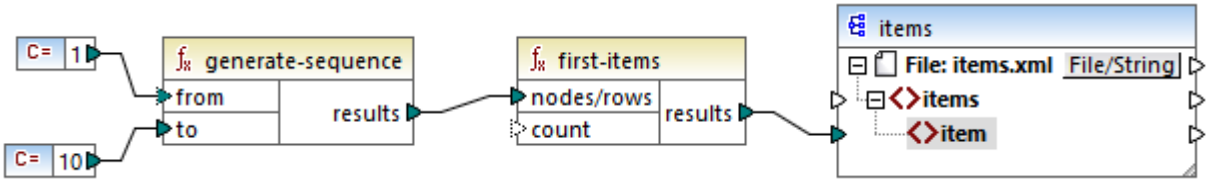
(C) 2015-2021 Altova GmbH

パラメーター

名前	説明
nodes/rows	この入力はゼロまたは他の値を提供するマッピングアイテムからの接続を受け取る必要があります。例えば、接続はソースXML アイテム、CSV フィールド、データベースレコードなど.から始まっている場合があります。
count	任意のパラメーターです。入力シーケンスから抽出されるアイテムの数量を指定します。デフォルトの値は**1** です。

サンプル

次の模擬マッピングは10 の値のシーケンスを生成します。`first-items` 関数によりシーケンスは処理され、結果はターゲット XML ファイルに書き込まれます。



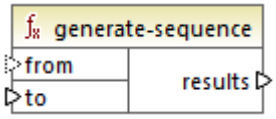
count 引数に値が存在しないため、**1** のデフォルトの値が適用されます。個の結果、シーケンスからの最初の値のみがマッピング出力内に生成されます:

```
<items>
   <item>1</item>
</items>
```

更に現実的なサンプルに関しては、マッピングにパラメーターを提供する内の**FindHighestTemperatures.mfd** マッピングを参照してください。

7.10.9.4 generate-sequence

[from] と[to] パラメーターを境界として使用し、整数のシーケンスを作成します。



言語

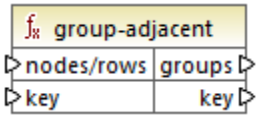
Built-in、C++、C#、Java、XQuery、XSLT 2.0、XSLT 3.0。

パラメーター

名前	説明
from	任意のパラメーターです。シーケンスが開始する整数(下限の数量)を指定します。デフォルト の値は**1** です。
宛先	必須のパラメーターです。シーケンスが終了する整数(上限の数量)を指定します。

7.10.9.5 group-adjacent

`group-adjacent` 関数は、**key** 入力に接続されているキーにより**nodes/rows** 入力に接続されているアイテムをグループ分けします。この関数は、アイテムが隣接しない場合、同じキー情報を共有するアイテムを個別のグループに分けます。複数の連続した(隣接した)アイテムが同じキーを共有する場合、これらのアイテムは同じグループに配置されます。



例えば、下記の抽象的な変換に示されるように、グループキーは「Department」です。このダイアグラムの左側は入力データを示し、右側はグループ分けの後の出力データを示しています。次は、変換の実行中に発生します:

- 最初は、最初のキー「Administration」は新規のグループを作成します。
- 次のキーは異なるため、2番目のグループである「Marketing」が作成されます。
- 3番目のキーもまた異なるための、「Engineering」と呼ばれる他のグループが作成されます。
- 4番目のキーは3番目と同じキーです。このため、レコードは既存のグループに配置されます。
- 最後に、5番目のキーは4番目のキーと異なるため、最後のグループを作成します。

下に説明されるとおり、「Michelle Butler」と「Fred Landis」は隣接する同じキーを持つために一緒にグループ分けされています。しかしながら、「Vernon Callaby」と「Frank Further」は同じキーを所有しているにもかかわらず隣接していないため、異なるグループに分けれています。

Department	Name
Administration	Vernon Callaby
Marketing	Susi Sanna
Engineering	Michelle Butler
Engineering	Fred Landis
Administration	Frank Further

group-adjacent()

Department	Name
Administration	Vernon Callaby

Department	Name
Marketing	Susi Sanna

Department	Name
Engineering	Michelle Butler
Engineering	Fred Landis

Department	Name
Administration	Frank Further

(C) 2015-2021 Altova GmbH

言語

Built-in、C++、C#、Java、XSLT 2.0、XSLT 3.0。

パラメーター

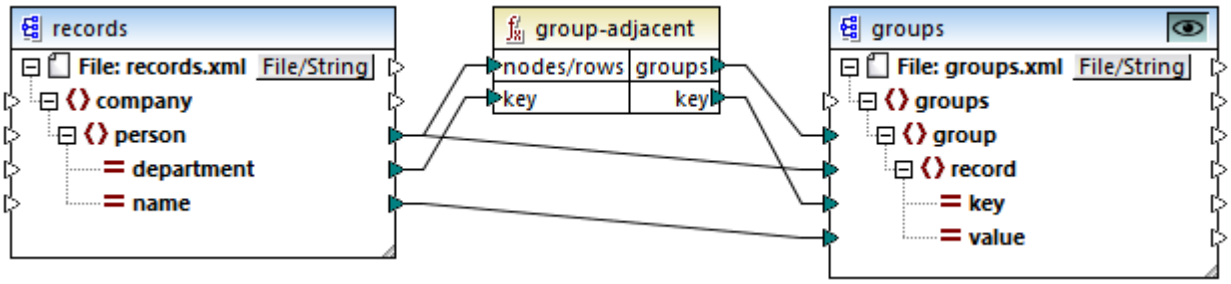
名前	説明
nodes/rows	この入力はゼロまたは他の値を提供するマッピングアイテムからの接続を受け取る必要があります。例えば、接続はソースXMLアイテム、CSVフィールド、データベースレコードなど.から始まっている場合があります。
key	アイテムをグループするキー

サンプル

使用するソースデータは次のコンテンツを持つXMLファイルです（下のコードリストでは名前空間とXML宣言は簡素化のために削除されていることにご注意してください）。

```
<company>
   <person department="Administration" name="Vernon Callaby"/>
   <person department="Marketing" name="Susi Sanna"/>
   <person department="Engineering" name="Michelle Butler"/>
   <person department="Engineering" name="Fred Landis"/>
   <person department="Administration" name="Frank Further"/>
</company>
```

ビジネスの条件は、隣接する部署の個人レコードをグループ分けします。これを達成するには、次のマッピングが`group-adjacent`関数を呼び出し、**department**を**key**として提供します。



マッピングの結果は以下のとおりです:

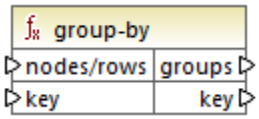
```
<groups>
   <group>
      <record key="Administration" value="Vernon Callaby"/>
   </group>
   <group>
      <record key="Marketing" value="Susi Sanna"/>
   </group>
   <group>
      <record key="Engineering" value="Michelle Butler"/>
```

```
      <record key="Engineering" value="Fred Landis"/>
   </group>
   <group>
      <record key="Administration" value="Frank Further"/>
   </group>
</groups>
```

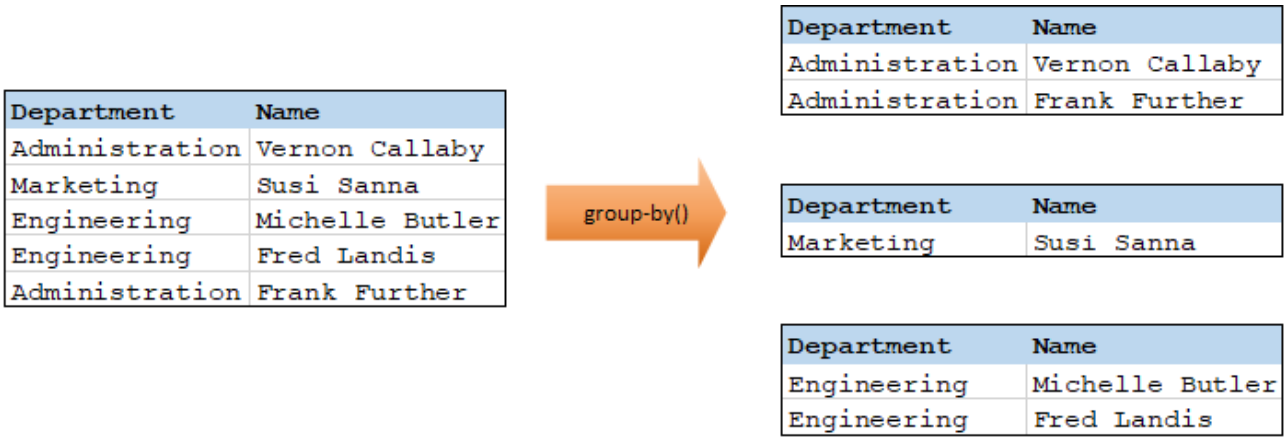
他のグループサンプルと共に、このサンプルは、次のマッピングファイルの一部です: **<マイドキュメント>\Altova\MapForce2021\MapForceExamples\Tutorial\GroupingFunctions.mfd**。出力 タブをクリックする前に、プレビューする関数に適用することのできる プレビュー ボタンをクリックします。

7.10.9.6 group-by

group-by 関数は、指定するグループキーに従いレコードのグループを作成します。



例えば、下で示される抽象的な変換では、グループキーは、「Department」です。トータルで3つの一意の部署が存在するため、**group-by** を適用すると3つのグループが作成されます:



言語

Built-in、C++、C#、Java、XSLT 2.0、XSLT 3.0。

パラメーター

名前	説明
nodes/rows	この入力はゼロまたは他の値を提供するマッピングアイテムからの接続を受け取る必要があります。例えば、接続はソースXML アイテム、CSV フィールド、データベースレコードなど.から始まっている場合があります。

(C) 2015-2021 Altova GmbH

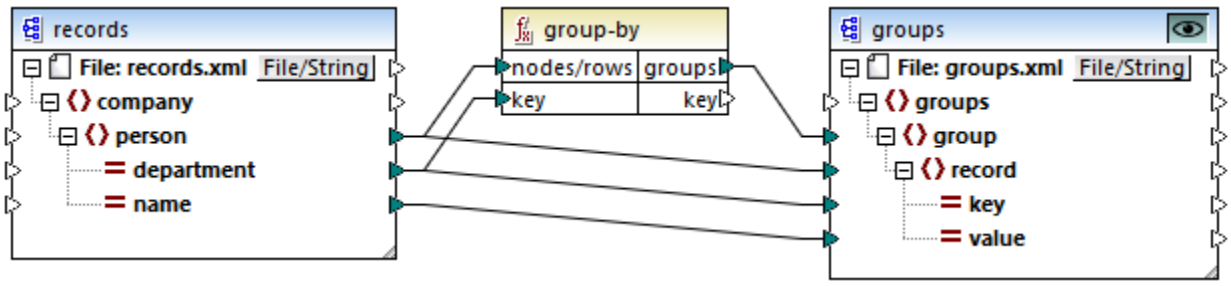
名前	説明
key	アイテムをグループするキー

サンプル

使用するソースデータは次のコンテンツを持つXML ファイルです（下のコードリストでは名前空間とXML 宣言は簡素化のために削除されていることにご注意してください）。

```
<company>
   <person department="Administration" name="Vernon Callaby"/>
   <person department="Marketing" name="Susi Sanna"/>
   <person department="Engineering" name="Michelle Butler"/>
   <person department="Engineering" name="Fred Landis"/>
   <person department="Administration" name="Frank Further"/>
</company>
```

ビジネスの条件は、部署別に個人のレコードをグループ分けします。これを達成するには、次のマッピングが`group-by` 関数を呼び出し **department** を提供します。



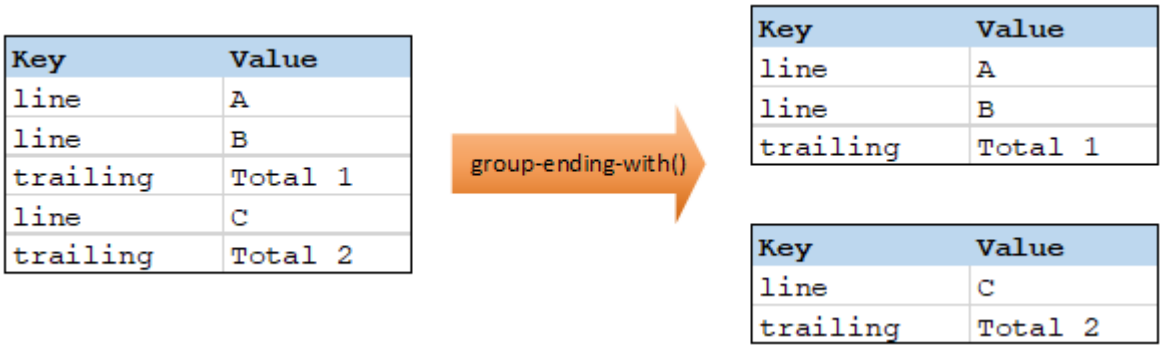
マッピングの結果は以下のとおりです:

```
<groups>
   <group>
      <record key="Administration" value="Vernon Callaby"/>
      <record key="Administration" value="Frank Further"/>
   </group>
   <group>
      <record key="Marketing" value="Susi Sanna"/>
   </group>
   <group>
      <record key="Engineering" value="Michelle Butler"/>
      <record key="Engineering" value="Fred Landis"/>
   </group>
</groups>
```

他のグループサンプルと共に、このサンプルは、次のマッピングファイルの一部です: **<マイドキュメント >\Altova\MapForce2021\MapForceExamples\Tutorial\GroupingFunctions.mfd**。出力 タブをクリックする前に、プレビューする関数に適用することのできる プレビュー ボタンをクリックします。

7.10.9.7 group-ending-with

group-ending-with 関数はブール式の条件を引数として取ります。関数はブール式の条件がtrue の場合、条件を満たすレコードが最後になるように、新規のグループが作成されます。下のサンプルでは、「Key」が「trailing」と等価であることが条件です。3番目と5番目のレコードのために、この条件がtrue の場合、2つのグループが結果として作成されます:



メモ 条件を満たす最後のレコードの前にレコードが存在する場合、追加のグループが作成されます。例えば、「line」レコードが最後の「heading」レコードより先多く存在する場合、これらは新しいグループ内に配置されます。

言語

Built-in、C++、C#、Java、XSLT 2.0、XSLT 3.0。

パラメーター

名前	説明
nodes/rows	この入力はゼロまたは他の値を提供するマッピングアイテムからの接続を受け取る必要があります。例えば、接続はソースXML アイテム、CSV フィールド、データベースレコードなど.から始まっている場合があります。
bool	**true** の場合、新規のグループを開始するブール式を提供します。

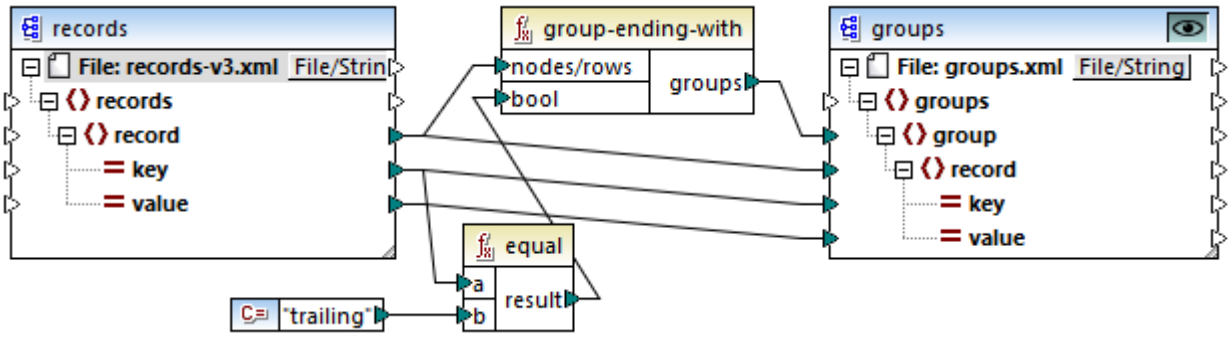
サンプル

使用するソースデータは次のコンテンツを持つXML ファイルです(下のコードリストでは名前空間とXML 宣言は簡素化のために削除されていることにご注意してください)。

```
<records>
   <record key="line" value="A"/>
   <record key="line" value="B"/>
   <record key="trailing" value="Total 1"/>
   <record key="line" value="C"/>
   <record key="trailing" value="Total 2"/>
</records>
```

ビジネスの条件は、各[行末]レコードのためにグループを作成することです。各グループは[行末]レコードの前にある[line]レコードを含む必要があります。これを達成するには、次のマッピングが**group-ending-with** 関数を呼び出します。下のマッピングでは、**key** 名が[行末]に等しい場合、**bool** に与えられている引数が**true** になり、新規のグループが作成されます。

(C) 2015-2021 Altova GmbH



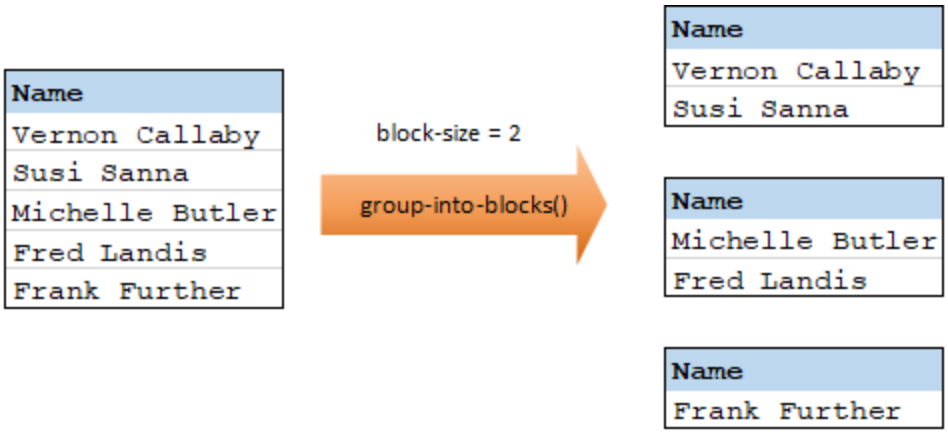
マッピングの結果は以下のとおりです:

```
<groups>
   <group>
      <record key="line" value="A"/>
      <record key="line" value="B"/>
      <record key="trailing" value="Total 1"/>
   </group>
   <group>
      <record key="line" value="C"/>
      <record key="trailing" value="Total 2"/>
   </group>
</groups>
```

他のグループサンプルと共に、このサンプルは、次のマッピングファイルの一部です: **<マイドキュメント>\Altova\MapForce2021\MapForceExamples\Tutorial\GroupingFunctions.mfd**。出力 タブをクリックする前に、プレビューする関数に適用することのできる プレビュー ボタンをクリックします。

7.10.9.8 group-into-blocks

group-into-blocks 関数は、N が`block-size` 引数に与えられる値である、N アイテムを含む等価のグループを作成します。最後のグループは、ソース内のアイテムの数に従い、N つのアイテム、または少ない数のアイテムを含むことができます。下のサンプルでは、`block-size` は2 です。総数で5 つのアイテムが存在し、最後のアイテムを除き、それぞれのグループは2 つのアイテムを含んでいます。



言語

Built-in、C++、C#、Java、XSLT 2.0、XSLT 3.0。

パラメーター

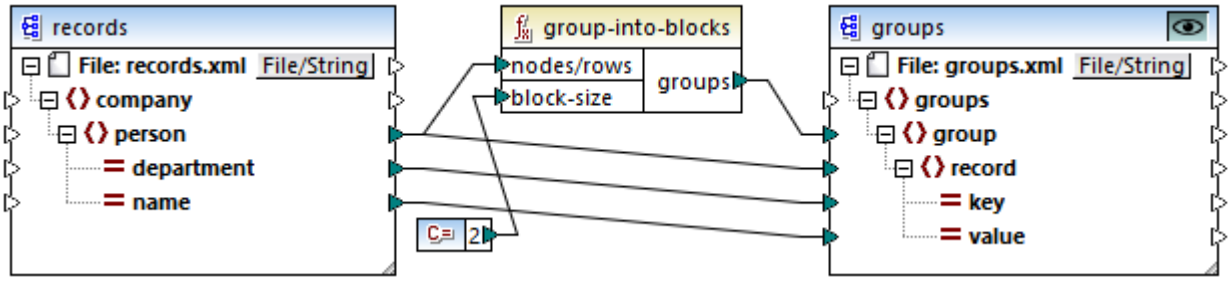
名前	説明
nodes/rows	この入力はゼロまたは他の値を提供するマッピングアイテムからの接続を受け取る必要があります。例えば、接続はソースXML アイテム、CSV フィールド、データベースレコードなど.から始まっている場合があります。
block-size	各グループのサイズを指定します。

サンプル

使用するソースデータは次のコンテンツを持つXML ファイルです(下のコードリストでは名前空間とXML 宣言は簡素化のために削除されていることにご注意してください)。

```
<company>
   <person department="Administration" name="Vernon Callaby"/>
   <person department="Marketing" name="Susi Sanna"/>
   <person department="Engineering" name="Michelle Butler"/>
   <person department="Engineering" name="Fred Landis"/>
   <person department="Administration" name="Frank Further"/>
</company>
```

個人のレコードを2 つのアイテムを持つブロックにグループ分けすることがビジネスの必要条件とします。これを達成するために、以下のマッピングは`group-into-blocks` 関数を呼び出し **block-size** として整数 [2] を提供します。



マッピングの結果は以下のとおりです:

```
<groups>
   <group>
      <record key="Administration" value="Vernon Callaby"/>
      <record key="Marketing" value="Susi Sanna"/>
   </group>
   <group>
      <record key="Engineering" value="Michelle Butler"/>
      <record key="Engineering" value="Fred Landis"/>
   </group>
```


(C) 2015-2021 Altova GmbH

```
    <group>
      <record key="Administration" value="Frank Further"/>
    </group>
</groups>
```

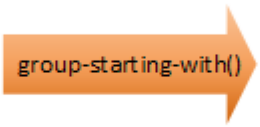
アイテムの総数（5）を2 で割ることができないため、最後のグループに1つのアイテムのみ含まれていることにご注意してください。

他のグループサンプルと共に、このサンプルは、次のマッピングファイルの一部です: **<マイドキュメント>\Altova\MapForce2021\MapForceExamples\Tutorial\GroupingFunctions.mfd**。出力 タブをクリックする前に、プレビューする関数に適用することのできる プレビュー ボタンをクリックします。

7.10.9.9 group-starting-with

group-starting-with 関数はブール式の条件を引数として取ります。関数はブール式の条件がtrue の場合、条件を満たすレコードから、新規のグループが作成されます。下のサンプルでは、「Key」が「heading」と等価であることが条件です。最初と4番目のレコードのために、この条件がtrue の場合、2つのグループが結果として作成されます:

Key	Value
heading	Intro
line	A
line	B
heading	Body
line	C



Key	Value
heading	Intro
line	A
line	B

Key	Value
heading	Body
line	C

メモ 条件を満たす最初のレコードの前にレコードが存在する場合、追加のグループが作成されます。例えば、「line」レコードが最初の「heading」レコードよりも多く存在する場合、これらは新しいグループ内に配置されます。

言語

Built-in、C++、C#、Java、XSLT 2.0、XSLT 3.0。

パラメーター

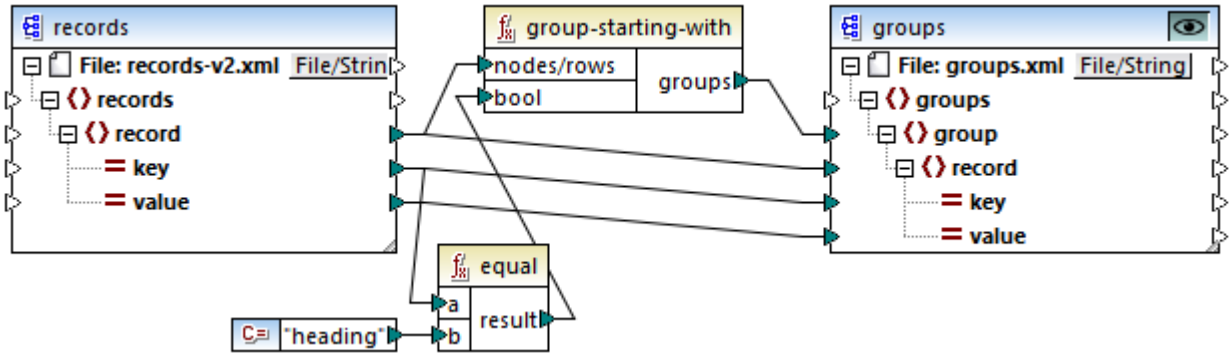
名前	説明
nodes/rows	この入力はゼロまたは他の値を提供するマッピングアイテムからの接続を受け取る必要があります。例えば、接続はソースXML アイテム、CSV フィールド、データベースレコードなど.から始まっている場合があります。
bool	**true** の場合、新規のグループを開始するブール式を提供します。

サンプル

使用するソースデータは次のコンテンツを持つXML ファイルです（下のコードリストでは名前空間とXML 宣言は簡素化のために削除されていることにご注意してください）。

```
<records>
   <record key="heading" value="Intro"/>
   <record key="line" value="A"/>
   <record key="line" value="B"/>
   <record key="heading" value="Body"/>
   <record key="line" value="C"/>
</records>
```

ビジネスの条件は、各 [heading] レコードのためにグループを作成することです。各グループは [heading] レコードの後にある [line] レコードを含んでいる必要があります。これを達成するには、次のマッピングが`group-starting-with` 関数を呼び出します。下のマッピングでは、**key** 名が「heading」に等しい場合、**bool** に与えられている引数が**true** になり、新規のグループが作成されます。



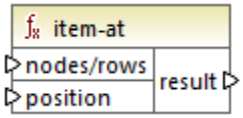
マッピングの結果は以下のとおりです:

```
<groups>
   <group>
      <record key="heading" value="Intro"/>
      <record key="line" value="A"/>
      <record key="line" value="B"/>
   </group>
   <group>
      <record key="heading" value="Body"/>
      <record key="line" value="C"/>
   </group>
</groups>
```

他のグループサンプルと共に、このサンプルは、次のマッピングファイルの一部です: **<マイドキュメント >\Altova\MapForce2021\MapForceExamples\Tutorial\GroupingFunctions.mfd**。**出力** タブをクリックする前に、プレビューする関数に適用することのできる **プレビュー** ボタンをクリックします。

7.10.9.10 item-at

position 引数により提供されたポジションで引数として提供された **nodes/rows** のシーケンスからアイテムを返します。最初のアイテムはポジション **1** にあります。



言語

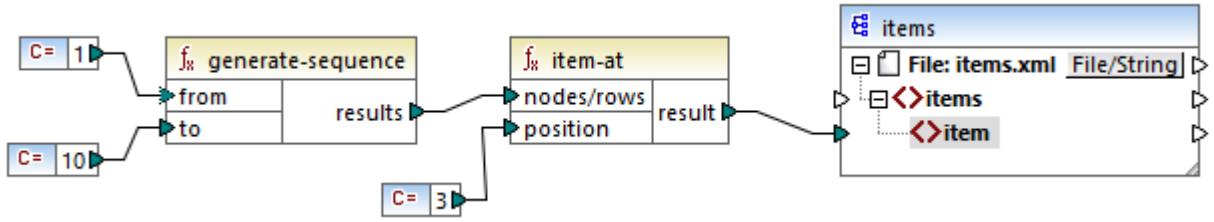
Built-in、C++、C#、Java、XQuery、XSLT 2.0、XSLT 3.0。

パラメーター

名前	説明
nodes/rows	この入力はゼロまたは他の値を提供するマッピングアイテムからの接続を受け取る必要があります。例えば、接続はソースXML アイテム、CSV フィールド、データベースレコードなど.から始まっている場合があります。
position	個の整数はアイテムのシーケンスからどのアイテムが返されるかを指定します。

サンプル

次の模擬マッピングは10 の値のシーケンスを生成します。`item-at` 関数によりシーケンスは処理され、結果はターゲット XML ファイルに書き込まれます。

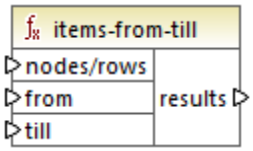


position 引数が**3** に設定されているため、シーケンスからの3番目の値がターゲットにパスされます。そのため、(XML と名前宣言を除く) マッピングの出力は以下の通りです:

```
<items>
   <item>3</item>
</items>
```

7.10.9.11 items-from-till

[from] と[till] パラメーターを境界として使用し、**nodes/rows** のシーケンスを返します。最初のアイテムはポジション **1** にあります。



言語

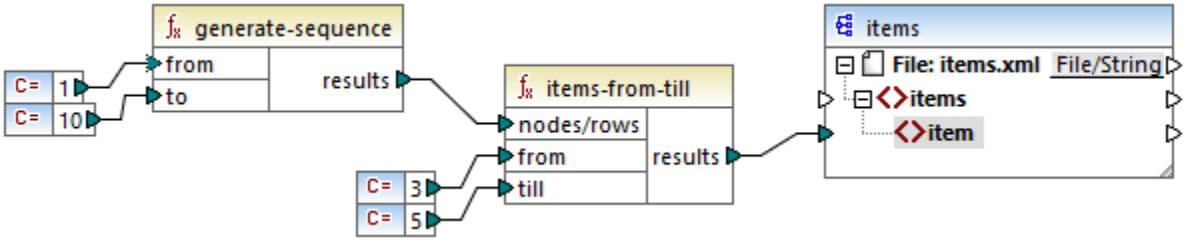
Built-in、C++、C#、Java、XQuery、XSLT 2.0、XSLT 3.0。

パラメーター

名前	説明
nodes/rows	この入力はゼロまたは他の値を提供するマッピングアイテムからの接続を受け取る必要があります。例えば、接続はソースXML アイテム、CSV フィールド、データベースレコードなど.から始まっている場合があります。
from	この整数は抽出されるアイテムの開始のポジションを指定します。
till	この整数は抽出されるアイテムまでのポジションを指定します。

サンプル

次の模擬マッピングは10 の値のシーケンスを生成します。`items-from-till` 関数によりシーケンスは処理され、結果はターゲット XML ファイルに書き込まれます。



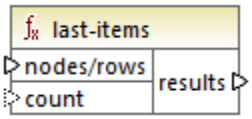
from と**till** 引数は**3** と**5** にそれぞれ設定されているため、**3** から**5** からの値のサブセットのみがターゲットにパスされます。そのため、(XML と名前宣言を除く)マッピングの出力は以下の通りです:

```
<items>
   <item>3</item>
   <item>4</item>
   <item>5</item>
</items>
```


(C) 2015-2021 Altova GmbH

7.10.9.12 last-items

N が**count** パラメーターにより提供される個所で、入力シーケンスの最後の *N* 個のアイテムを返します。最初のアイテムはポジション [1] にあります。



言語

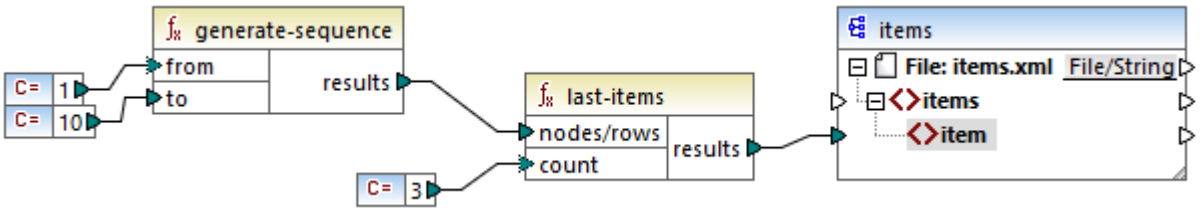
Built-in、C++、C#、Java、XQuery、XSLT 2.0、XSLT 3.0。

パラメーター

名前	説明
nodes/rows	この入力はゼロまたは他の値を提供するマッピングアイテムからの接続を受け取る必要があります。例えば、接続はソースXML アイテム、CSV フィールド、データベースレコードなど.から始まっている場合があります。
count	任意のパラメーターです。入力シーケンスから抽出されるアイテムの数量を指定します。デフォルトの値は**1** です。

サンプル

次の模擬マッピングは10 の値のシーケンスを生成します。`last-items` 関数によりシーケンスは処理され、結果はターゲット XML ファイルに書き込まれます。

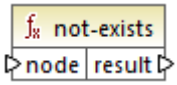


カウント引数が**3** に設定されているため、シーケンスからの最後の3 つの値がターゲットにパスされます。そのため、（XML と名前宣言を除く）マッピングの出力は以下の通りです：

```
<items>
   <item>8</item>
   <item>9</item>
   <item>10</item>
</items>
```

7.10.9.13 not-exists

接続済みのノードが存在する場合は**false** を返します。それ以外の場合は**true** を返します。この関数はexists 関数の逆ですが、使用方法は同じです。



言語

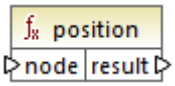
Built-in、C++、C#、Java、XQuery、XSLT 1.0、XSLT 2.0、XSLT 3.0。

パラメーター

名前	説明
node	不在を確認するノード。

7.10.9.14 position

現在処理されているアイテムのシーケンス内のアイテムのポジションを返します。これは、例えば、アイテムに自動的に付番する場合に使用することができます。



言語

Built-in、C++、C#、Java、XQuery、XSLT 1.0、XSLT 2.0、XSLT 3.0。

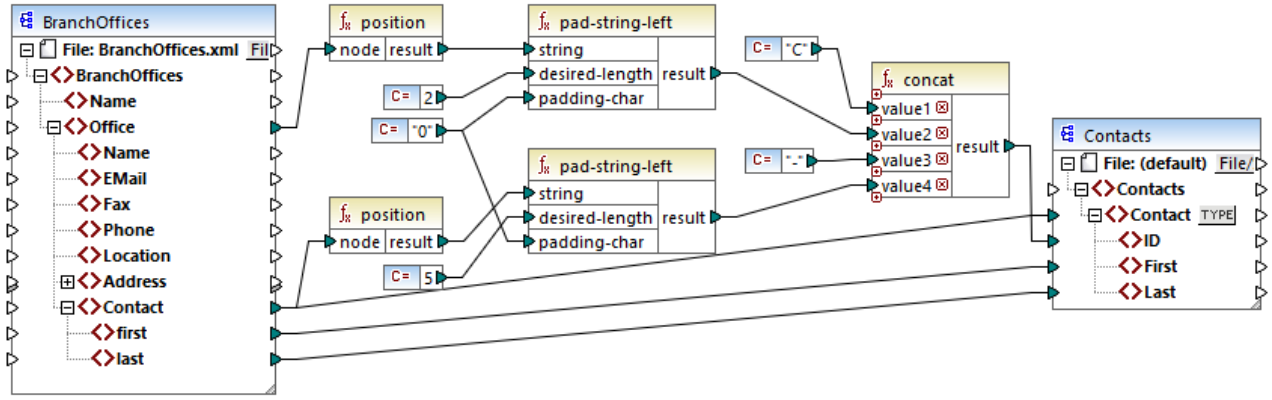
パラメーター

名前	説明
node	この入力はゼロまたは他の値を提供するマッピングアイテムからの接続を受け取る必要があります。例えば、接続はソースXML アイテム、CSV フィールド、データベースレコードなど.から始まっている場合があります。

サンプル

次のマッピングではマッピングにより生成されたデータ内で一意の識別地を生成するために`position` 関数が使用されています。このマッピングには以下のパスで見つけることができるマッピングデザインファイルが存在します: **<マイドキュメント>\Altova\MapForce2021\MapForceExamples\ContactsFromBranchOffices.mfd**.

(C) 2015-2021 Altova GmbH



ContactsFromBranchOffices.mfd

上のマッピングではソースXML ファイル内に3 つのブランチオフィスが含まれています。ブランチオフィスには**Contact** 子アイテムの任意の番号を含む場合があります。マッピングの目的は以下の通りです:

- ソースXML ファイルからすべての**Contact** アイテムを抽出し、ターゲット XML ファイルに書き込みます。
- 各連絡先に一意の識別番号が割り当てられる必要があります(ターゲット XML 内の**ID** アイテム)。
- 各連絡先のID はX がオフィス番号を識別し、X が連絡先番号を識別するフォーム`CXX-YYYYY` を取る必要があります。オフィス番号が2 文字以下の場合、左側はゼロで埋め込まれる必要があります。同様に、連絡先番号が5 文字以下を取る場合、左側はゼロで埋め込まれる必要があります。この結果、最初のオフィスからの最初の連絡先の有効な識別番号は`C01-00001` のようになります。

マッピングの目的を達成するには`position` 関数を含むいくつかのMapForce 関数が使用されます。上の`position` 関数は各オフィスのポジションを取得します。下の関数は各オフィスのコンテキストで各連絡先のポジションを取得します。

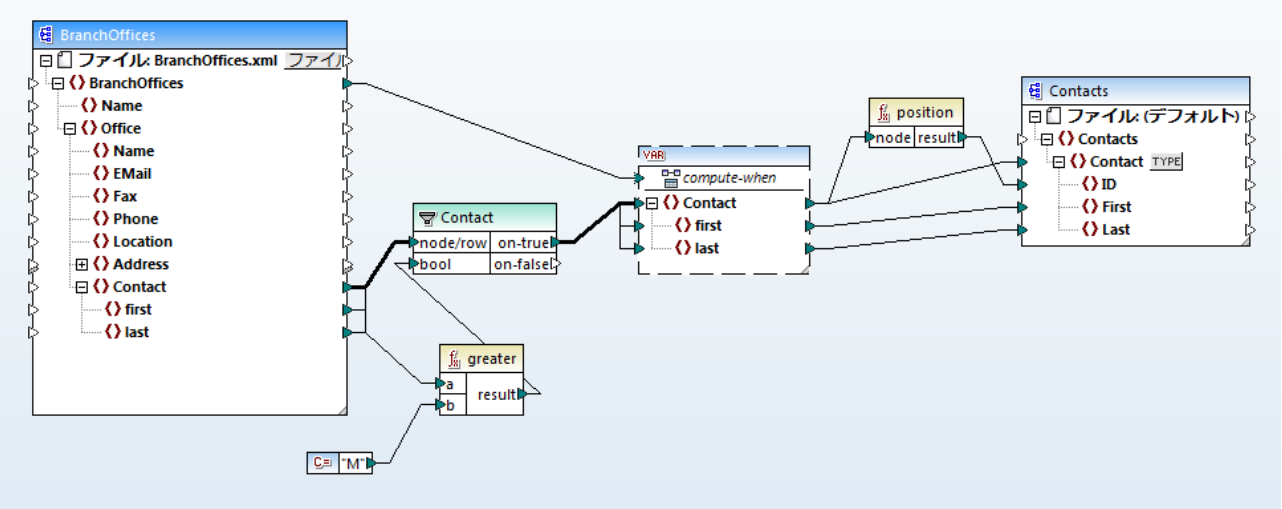
`position` 関数を使用する場合、現在のマッピングコンテキストを考慮してください。具体的には、マッピングの実行中に関数を介して間接的に最初のマッピングコンテキストが接続されるソースアイテムのターゲットコンポーネントのルートアイテムから確率されます。このサンプル内では上の`position` 関数は*全てのオフィスのシーケンス*を処理し、シーケンス内の最初のオフィスに対応する値 1 を最初に生成します。下の`position` 関数は(1、2、3 などの) *そのオフィスのコンテキスト* で連絡先のポジションに対応するシーケンシャル番号を生成します。次のオフィスが処理されるとこの内部シーケンスが設定され(このため、再度 1 から開始します) 。前に記述された必要条件に従い`pad-string-left` 関数の両方はパディングを生成された番号に適用します。(ターゲット **Contact** へのソースからの親接続のため) `Concat` 関数は*各連絡先のコンテキスト*で作動します全ての値をジョインし、各連絡先の一意の識別番号を返します。

上のマッピングから生成有れた出力は以下に表示される通りです(いくつかのレコードは読みやすさのために削除されています):

```
<Contacts>
   <Contact>
      <ID>C01-00001</ID>
      <First>Vernon</First>
      <Last>Callaby</Last>
   </Contact>
   <Contact>
      <ID>C01-00002</ID>
      <First>Frank</First>
      <Last>Further</Last>
   </Contact>
   <!-- ...-->
   <Contact>
      <ID>C02-00001</ID>
      <First>Steve</First>
```

```
      <Last>Meier</Last>
   </Contact>
   <Contact>
      <ID>C02-00002</ID>
      <First>Theo</First>
      <Last>Bone</Last>
   </Contact>
   <!-- ...-->
</Contacts>
```

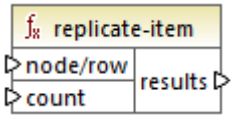
フィルターの適用後にアイテムのポジションを取得する必要がある場合があります。フィルターコンポーネントはシーケンス関数ではないため、フィルターされたアイテムのポジションを検索する関数と共には`position` 関数と共に*直接*使用することはできません非間接的にこれは変数コンポーネントを追加することにより可能になります。例えば、下のマッピングは前のマッピングを簡素化したものです。マッピングデザインファイルは次のパスで検索することができます: **<マイドキュメント >\Altova\MapForce2021\MapForceExamples\PositionInFilteredSequence.mfd**.



MapForce 内の変数の結果は常にシーケンスです。このために上にマッピングでは`position` 関数は変数により作成されたシーケンスを反復し、そのシーケンス内の各アイテムのポジションを返します。サンプル: ノードのフィルターと番号付けで詳しく説明されています。

7.10.9.15 replicate-item

入力シーケンス内の全てのアイテムを **count** 引数内で指定された回数繰り返します。単一のアイテムを **node/row** 引数に接続すると、関数は、*N* が **count** 引数の値である箇所の *N* アイテムを返します。アイテムのシーケンスを **node/row** 引数に接続すると、1度に1つのアイテムを処理し、関数は、シーケンス内のそれぞれのアイテムを **count** 回繰り返します。例えば、カウントが **2** の場合、シーケンス `1,2,3` は `1,1,2,2,3,3` を生成します。下のサンプルで説明されている通り異なる **count** 値を入力シーケンス内の各アイテムに与えることができます。



(C) 2015-2021 Altova GmbH

言語

Built-in、C++、C#、Java、XQuery、XSLT 2.0、XSLT 3.0。

パラメーター

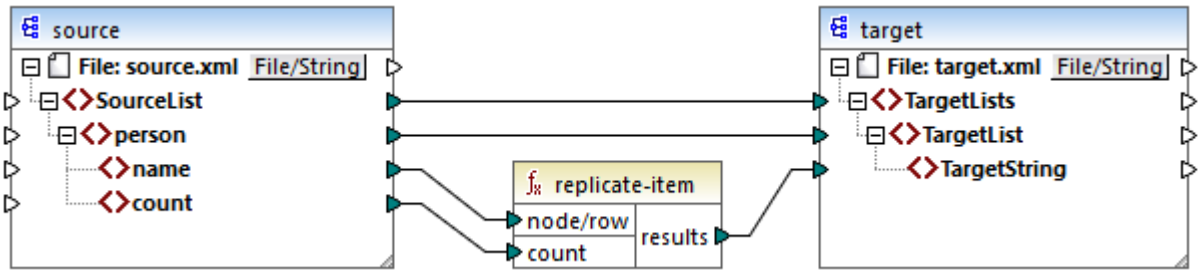
名前	説明
nodes/rows	この入力はゼロまたは他の値を提供するマッピングアイテムからの接続を受け取る必要があります。例えば、接続はソースXMLアイテム、CSVフィールド、データベースレコードなど.から始まっている場合があります。
count	**node/row** に接続されている各アイテムまたはシーケンスを複数する回数を指定します。

サンプル

以下の構造を持つソースXMLファイルが存在すると想定します:

```
<SourceList>
   <person>
      <name>Michelle</name>
      <count>2</count>
   </person>
   <person>
      <name>Ted</name>
      <count>4</count>
   </person>
   <person>
      <name>Ann</name>
      <count>3</count>
   </person>
</SourceList>
```

`replicate-item` 関数を使用して各個人名を異なる回数ターゲットコンポーネント内で繰り返すことができます。これを達成するために各個人の**count** ノード を関数の `replicate-item`関数の**count** 入力に接続します:



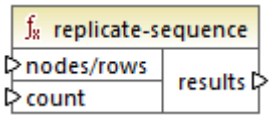
出力は以下のようになります:

```
<TargetLists>
   <TargetList>
      <TargetString>Michelle</TargetString>
```

```
      <TargetString>Michelle</TargetString>
   </TargetList>
   <TargetList>
      <TargetString>Ted</TargetString>
      <TargetString>Ted</TargetString>
      <TargetString>Ted</TargetString>
      <TargetString>Ted</TargetString>
   </TargetList>
   <TargetList>
      <TargetString>Ann</TargetString>
      <TargetString>Ann</TargetString>
      <TargetString>Ann</TargetString>
   </TargetList>
</TargetLists>
```

7.10.9.16 replicate-sequence

入力シーケンス内の全てのアイテムを **count** 引数内で指定された回数繰り返します。例えば、カウントが **2** の場合、シーケンス `1,2,3` は `1,2,3,1,2,3` を生成します。



言語

Built-in、C++、C#、Java、XQuery、XSLT 2.0、XSLT 3.0。

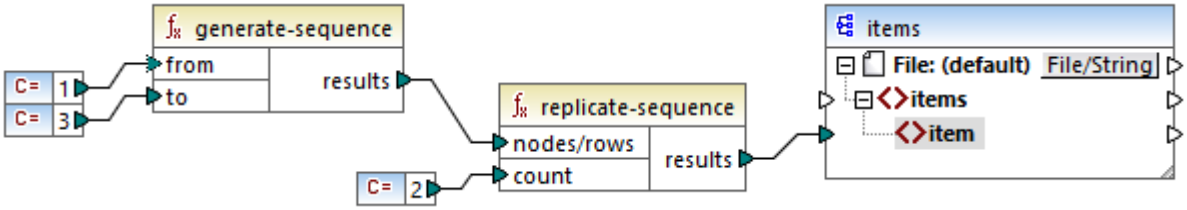
パラメーター

名前	説明
nodes/rows	この入力はゼロまたは他の値を提供するマッピングアイテムからの接続を受け取る必要があります。例えば、接続はソース XML アイテム、CSV フィールド、データベースレコードなど .から始まっている場合があります。
count	接続されているシーケンスを複数する回数を指定します。

サンプル

次の模擬マッピングはシーケンス `1,2,3` を生成します。`replicate-sequence` 関数によりシーケンスは処理され、結果はターゲット XML ファイルに書き込まれます。

(C) 2015-2021 Altova GmbH

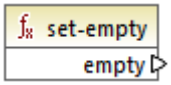


count 引数が**2** に設定されているため、シーケンスは2度複製されターゲットにパスされます。そのため、（XML と名前宣言を除く）マッピングの出力は以下の通りです：

```
<items>
   <item>1</item>
   <item>2</item>
   <item>3</item>
   <item>1</item>
   <item>2</item>
   <item>3</item>
</items>
```

7.10.9.17 set-empty

空のシーケンスを返します。例えば、XBRL コンポーネント/タクソノミの上位で定義されたXBRL ドキュメントのデフォルトの値 をキャンセルすることができます。

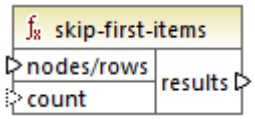


言語

Built-in、C++、C#、Java、XQuery、XSLT 2.0、XSLT 3.0。

7.10.9.18 skip-first-items

count 引数によりN が提供される個所で入力シーケンスの最初のN 個のアイテムをスキップし、残りのシーケンスを返します。



言語

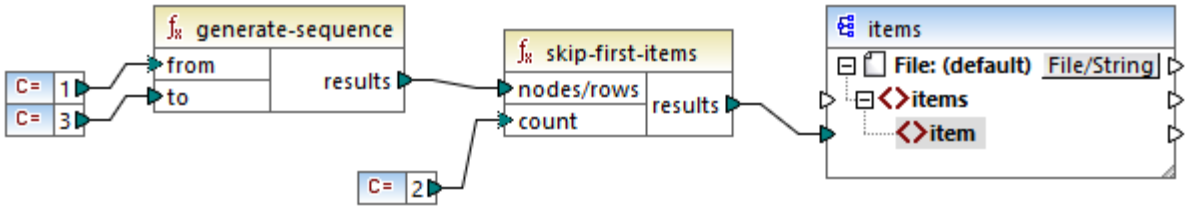
Built-in、C++、C#、Java、XQuery、XSLT 2.0、XSLT 3.0。

パラメーター

名前	説明
nodes/rows	この入力はゼロまたは他の値を提供するマッピングアイテムからの接続を受け取る必要があります。例えば、接続はソースXMLアイテム、CSVフィールド、データベースレコードなど.から始まっている場合があります。
count	オプションの引数。スキップするアイテムの数量を指定します。デフォルトの値は**1**です。

サンプル

次の模擬マッピングはシーケンス`1,2,3`を生成します。`skip-first-items`関数によりシーケンスは処理され、結果はターゲットXMLファイルに書き込まれます。

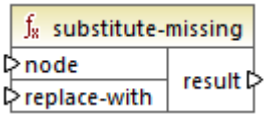


count引数が**2**に設定されているため、最初の2つのアイテムはスキップされ、残りのアイテムはターゲットにパスされます。そのため、(XMLと名前宣言を除く)マッピングの出力は以下の通りです:

```
<items>
   <item>3</item>
</items>
```

7.10.9.19 substitute-missing

この関数はexistsとif-else 条件の組み合わせにより構成されます。**node**入力に接続されているアイテムが存在すると、コンテンツはターゲットにコピーされます。それ以外の場合、**replace-with**入力に接続されているアイテムのコンテンツはターゲットにコピーされます。



言語

Built-in、C++、C#、Java、XQuery、XSLT 1.0、XSLT 2.0、XSLT 3.0。

パラメーター

名前	説明
node	この入力はゼロまたは他の値を提供するマッピングアイテムからの接続を受け取る必要があります。例

(C) 2015-2021 Altova GmbH

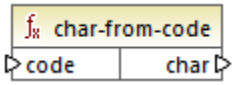
名前	説明
	えば、接続はソースXML アイテム、CSV フィールド、データベースレコードなど.から始まっている場合があります。
replace-with	この入力は置換値を提供するマッピングアイテムから接続を受け取る必要があります。

7.10.10 core | string functions (文字列関数)

文字列関数はデータの一部分を取得、サブ文字列のテスト、文字列からの情報の取得、文字列の分割などを行うために文字列データを操作することができます。

7.10.10.1 char-from-code

引数として提供されている小数 Unicode 値 (コード) の文字表示を返します。ヒント: 文字のUnicode 小数コードを検索するには `code-from-char` 関数を使用することができます。



言語

Built-in、C++、C#、Java、XQuery、XSLT 2.0、XSLT 3.0。

パラメーター

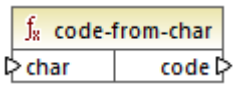
名前	説明
code	小数としてのUnicode 値。

サンプル

Unicode W eb サイトで使用可能なチャートによると(https://www.unicode.org/charts/) 感嘆符文字は `0021` の16 進数値を持っています。小数フォーマットの対応する値は**33** です。このため、引数として `33` を提供すると、 `char-from-code` 関数は `!` 文字を返します。特殊文字の置換も参照してください。

7.10.10.2 code-from-char

文字列として提供されている文字の小数 Unicode 値 (コード) が返されます。文字列として提供されている文字列が複数の文字を持つ場合、最初の文字のコードが返されます。



言語

Built-in、C++、C#、Java、XQuery、XSLT 2.0、XSLT 3.0。

パラメーター

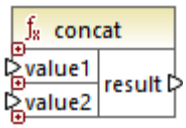
名前	説明
char	入力文字列の値。

サンプル

入力 **char** は **$**（ドルサイン）文字です。関数はこの文字のための小数 Unicode 値である **36** を返します。

7.10.10.3 concat

2 個または以上の値を連結（追加）して、単一の結果文字列を生成します。全ての入力値は自動的に文字列型へ変換されます。デフォルトでは、この関数には2 つのパラメータのみ存在しますが、更にパラメーターを追加することができます。**パラメーターの追加**（⊡）または**パラメーターの削除**（⊠）をクリックしてパラメーターを追加または削除します。関数の引数の追加、または削除も参照してください。



メモ `concat` 関数へのすべての入力は値を必要とします。入力の 1 つが値を持たない場合、関数が呼び出されず、エラーが発生します。空の文字列は有効な入力値ですが、（`set-empty` 関数の結果など）空のシーケンスは無効な値で、この結果、関数は失敗します。これを回避するには substitute-missing 関数を使用して値を最初に処理し `concat` 関数に結果を入力として提供します。

言語

Built-in、C++、C#、Java、XQuery、XSLT 1.0、XSLT 2.0、XSLT 3.0。

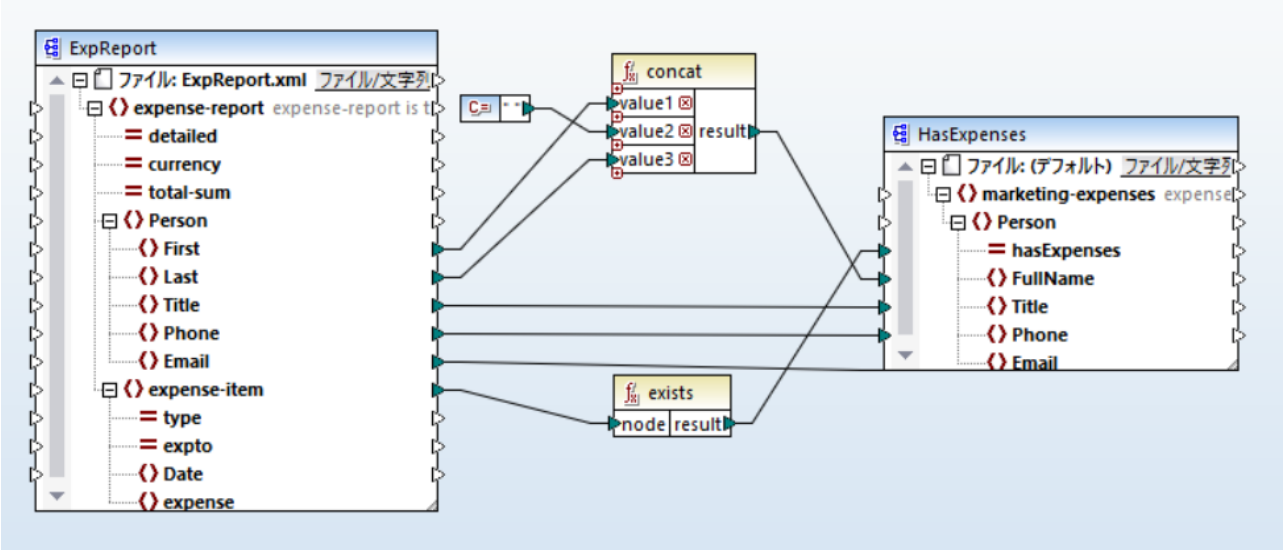
パラメーター

名前	説明
value1	最初の入力値。
value2	2番目の入力値。
valueN	*N* の入力値。

サンプル

下で説明されるマッピングでは、`concat` 関数は最初の名、定数 " "、そして姓を結合します。返される値は、**FullName** ターゲットアイテムに書き込まれます。この関数のマッピングは次のパスで見つけることができます: **<マイドキュメント>\Altova\MapForce2021\MapForceExamples\HasMarketingExpenses.mfd**。

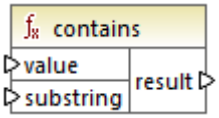
(C) 2015-2021 Altova GmbH



HasMarketingExpenses.mfd

7.10.10.4 contains

引数の値として提供されている文字列の値に引数として提供されているサブ文字列が含まれている場合、ブール値 **true** を返します。



言語

Built-in、C++、C#、Java、XQuery、XSLT 1.0、XSLT 2.0、XSLT 3.0。

パラメーター

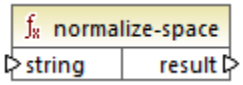
名前	説明
value	入力値（すなわち [haystack]）。
substring	（すなわち [needle]）を検索するサブ文字列。

サンプル

入力 **value** が[category] で **substring** が[cat] の場合、関数は **true** を返します。

7.10.10.5 normalize-space

正規化された入力文字列を返します。正規化は行頭および行末スペースが削除され、複数の連続した空白文字の各シーケンスが単一の空白スペースと置き換えられていることを意味します。空白スペースのためのUnicode 文字は(U+0020) です。



言語

Built-in、C++、C#、Java、XQuery、XSLT 1.0、XSLT 2.0、XSLT 3.0。

パラメーター

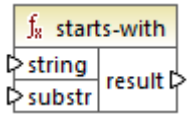
名前	説明
string	正規化するための入力文字列。

サンプル

入力文字列が `The quick  brown fox` の場合、関数は`The quick brown fox` を返します。

7.10.10.6 starts-with

引数として提供されているサブ文字列から開始する引数として文字列が提供されている場合、ブール値 **true** を返します。それ以外の場合 **false** を返します。



言語

Built-in、C++、C#、Java、XQuery、XSLT 1.0、XSLT 2.0、XSLT 3.0。

パラメーター

名前	説明
string	入力文字列。
substr	チェックするサブ文字列。

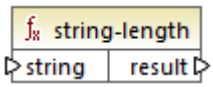
(C) 2015-2021 Altova GmbH

サンプル

入力 **string** が`category` で、**substr** が`cat` の場合、関数は**true** を返します。

7.10.10.7 string-length

引数として与えられている文字列内の文字の数量を返します。



言語

Built-in、C++、C#、Java、XQuery、XSLT 1.0、XSLT 2.0、XSLT 3.0。

パラメーター

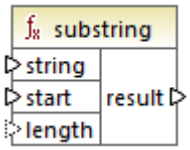
名前	説明
string	入力文字列。

サンプル

入力文字列が`car` の場合、関数は**3** を返します。入力文字列が空白の場合、関数は**0** を返します。

7.10.10.8 substring

start と**length** パラメーターにより指定されている文字列の部分を返します。



言語

Built-in、C++、C#、Java、XQuery、XSLT 1.0、XSLT 2.0、XSLT 3.0。

パラメーター

名前	説明
string	入力文字列。

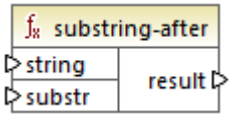
名前	説明
start	サブ文字列が抽出される開始位置（インデックス）を指定します。最初のインデックスは**1** です。
length	任意。抽出されるアイテムの数量を指定します。**length** パラメーターが指定されていない場合、結果は**start** から開始する文字列の最後までのフラグメントです。

サンプル

入力文字列が`MapForce` の場合、開始は**1** で、長さは**3** で、関数は`Map` を返します。入力文字列が`MapForce` の場合、開始は**4** で、長さが提供されていない場合、関数は`Force` を返します。

7.10.10.9 substring-after

substr の最初の発生後に発生する文字列の部分を返します。**substr** が**string** で発生しない場合、関数は空の文字列を返します。



言語

Built-in、C++、C#、Java、XQuery、XSLT 1.0、XSLT 2.0、XSLT 3.0。

パラメーター

名前	説明
string	入力文字列。
substr	サブ文字列。**substr** の最初の発生の後の文字は関数の結果です。

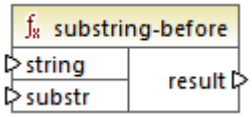
サンプル

入力文字列が`MapForce` で**substr** が`Map` の場合、関数は`Force` を返します。入力文字列が`2020/01/04` で**substr** が`/` の場合、関数は`01/04` を返します。例: RSS フィードからデータをマップするを参照してください。

7.10.10.10 substring-before

substr の最初の発生前に発生する文字列の部分を返します。**substr** が**string** で発生しない場合、関数は空の文字列を返します。

(C) 2015-2021 Altova GmbH



言語

Built-in、C++、C#、Java、XQuery、XSLT 1.0、XSLT 2.0、XSLT 3.0。

パラメーター

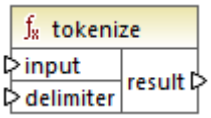
名前	説明
string	入力文字列。
substr	サブ文字列。**substr** の最初の発生の前の文字は関数の結果です。

サンプル

入力文字列が`MapForce` で**substr** が`Force` の場合、関数は`Map` を返します。入力文字列が`2020/01/04` で**substr** が`/` の場合、関数は`2020` を返します。

7.10.10.11 tokenize

引数として与えられた区切り文字を使用して入力文字列を文字列の*シーケンス*に分割します。



言語

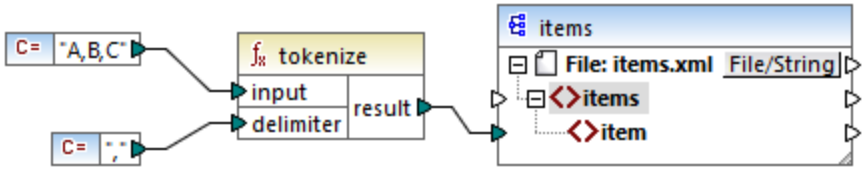
Built-in、C++、C#、Java、XQuery、XSLT 2.0、XSLT 3.0。

パラメーター

名前	説明
input	入力文字列。
delimiter	使用する区切り文字。

サンプル

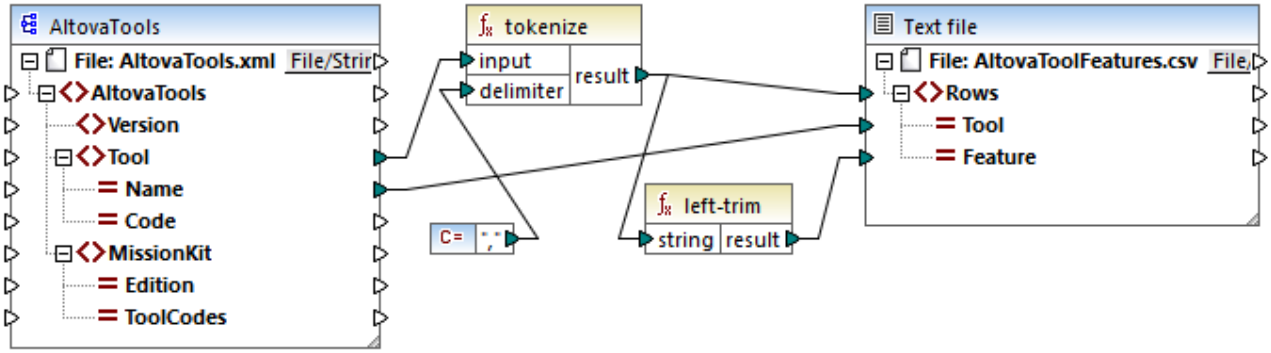
入力文字列が`A,B,C` の場合、区切り文字が`,` の場合、関数は3つの文字列の*シーケンス*を返します: `A`、`B`、および`C`。



上の模擬マッピングでは関数の結果は文字列のシーケンスです。一般的なマッピング ルールによると、ソースシーケンス内の各アイテムのために新規 **item** がターゲットコンポーネント内に作成されます。この結果、マッピングの出力は以下のようになります:

```
<items>
   <item>A</item>
   <item>B</item>
   <item>C</item>
</items>
```

更に詳細なサンプルに関しては<**マイドキュメント**>¥Altova¥MapForce2021¥MapForceExamples¥フォルダー内の **tokenizeString1.mfd** マッピングを参照してください。



tokenizeString1.mfd

ソースXMLファイルは以下で表示されています。**Tool** 要素には2つの属性が存在します: **Name** と**Code**。**Tool** 要素データはコンマで区切られたテキストで構成されています。

```
<?xml version="1.0" encoding="UTF-8"?>
<AltovaTools xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="AltovaTools.xsd">
   <Version>2010</Version>
   <Tool Name="XMLSpy" Code="XS">XML editor, XSLT editor, XSLT debugger, XQuery editor,
XQuery debugger, XML Schema / DTD editor, WSDL editor, SOAP debugger</Tool>
   <Tool Name="MapForce" Code="MF">Data integration, XML mapping, database mapping,
text conversion, EDI translator, Excel mapping, XBRL mapping, Web services</Tool>
   <Tool Name="StyleVision" Code="SV">Stylesheet designer, electronic forms, XSLT
design, XSL:FO design, database reporting, XBRL rendering</Tool>
   <Tool Name="UModel" Code="UM">UML modeling tool, code generation, reverse
engineering, UML, BPMN, SysML, project documentation, XMI interchange</Tool>
   <Tool Name="DatabaseSpy" Code="DS">Multi-database tool, SQL auto-completion,
graphical database design, table browser, content editor, database comparison
tool</Tool>
```


(C) 2015-2021 Altova GmbH

```
   <!-- ...-->
</AltovaTools>
```

マッピングは以下を行います:

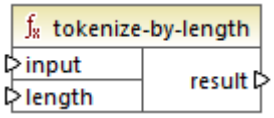
- **Tokenize** 関数は**Tool** ソースアイテムからデータを受け取り、コンマ , 区切り文字を使用してデータを個別のチャンクに分割します。最初のチャンクは[XML editor] および2番目のチャンクは[XSLT editor] などになります。
- **tokenize** 関数から結果する各チャンクのために、ターゲット内に新規の行が生成されます。これは関数の結果間の接続とターゲットコンポーネント内の**Rows** アイテムにより発生します。
- **tokenize** 関数の結果は各チャンクの先頭の空白文字を削除する **left-trim** 関数にもマップされます。
- **left-trim** 関数の結果（各チャンク）はターゲットコンポーネントの**Feature** アイテムに書き込まれます。
- ターゲットコンポーネント出力ファイルはフィールド区切り文字がセミコロンでCSV ファイル（**AltovaToolFeatures.csv**）として定義されています（設定を確認するためにコンポーネントをダブルクリックしてください）。

マッピングの結果は、 **tokenize** 関数により作成された各チャンクのためにターゲット CSV ファイル内で新規の行が作成されます。マッピングの出力のフラグメントは以下のようになります:

```
Tool;Feature
XMLSpy;XML editor
XMLSpy;XSLT editor
XMLSpy;XSLT debugger
XMLSpy;XQuery editor
XMLSpy;XQuery debugger
XMLSpy;XML Schema / DTD editor
XMLSpy;WSDL editor
XMLSpy;SOAP debugger
MapForce;Data integration
MapForce;XML mapping
MapForce;database mapping
MapForce;text conversion
MapForce;EDI translator
MapForce;Excel mapping
```

7.10.10.12 tokenize-by-length

入力文字列を文字列のシーケンスに分割します。結果する各文字列のサイズは**length** パラメーターにより決定されます。



言語

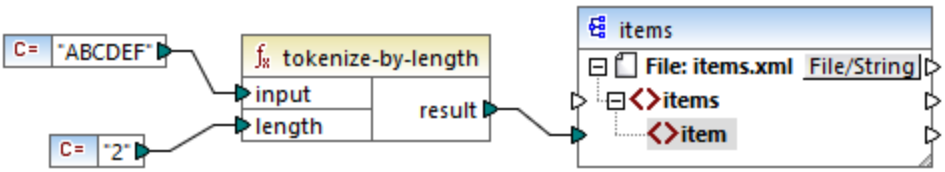
Built-in、C++、C#、Java、XQuery、XSLT 2.0、XSLT 3.0。

パラメーター

名前	説明
input	入力文字列。
length	文字列の生成されたシーケンス内の各文字列の長さを決定します。

サンプル

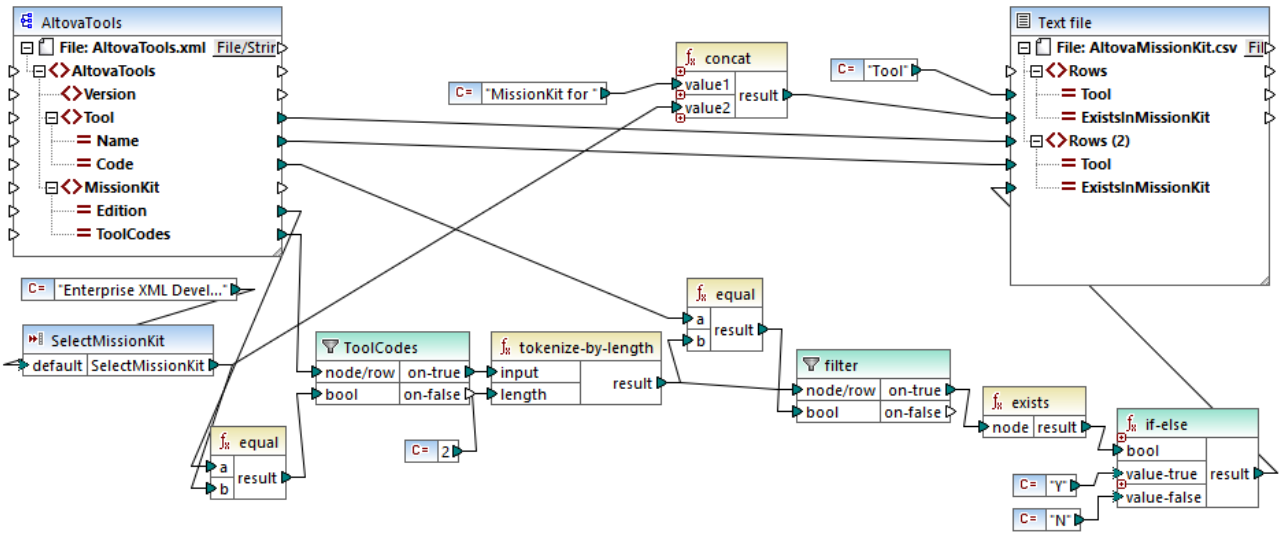
入力文字列が`ABCDEF` の場合、長さが`2` の場合、関数は3つの文字列のシーケンスを返します: `AB`、`CD`、および`EF`。



上の模擬マッピングでは関数の結果は文字列のシーケンスです。一般的なマッピング ルールによると、ソースシーケンス内の各アイテムのために新規 **item** がターゲットコンポーネント内に作成されます。この結果、マッピングの出力は以下のようになります:

```
<items>
   <item>AB</item>
   <item>CD</item>
   <item>EF</item>
</items>
```

更に詳細なサンプルに関しては<マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥フォルダー内の **tokenizeString2.mfd** マッピングを参照してください。



tokenizeString2.mfd

(C) 2015-2021 Altova GmbH

前例で使用したものと同じ XML ソースファイルを以下に示します。**MissionKit** 要素には2つの属性が存在します: **Edition** と **ToolCodes** ですがMissionKit 要素コンテントではありません。このサンプルに関連しない XML コンテンツは削除されていることに注意してください。

```
<?xml version="1.0" encoding="UTF-8"?>
<AltovaTools xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="AltovaTools.xsd">
   <Version>2010</Version>
   <Tool Name="XMLSpy" Code="XS"><!--...--></Tool>
   <Tool Name="MapForce" Code="MF"><!--...--></Tool>
   <Tool Name="StyleVision" Code="SV"><!--...--></Tool>
   <Tool Name="UModel" Code="UM"><!--...--></Tool>
   <Tool Name="DatabaseSpy" Code="DS"><!--...--></Tool>
   <Tool Name="DiffDog" Code="DD"><!--...--></Tool>
   <Tool Name="SchemaAgent" Code="SA"><!--...--></Tool>
   <Tool Name="SemanticWorks" Code="SW"><!--...--></Tool>
   <Tool Name="Authentic" Code="AU"><!--...--></Tool>
   <MissionKit Edition="Enterprise Software Architects" ToolCodes="XSMFSVUMDSDDSASW"/>
   <MissionKit Edition="Professional Software Architects" ToolCodes="XSMFSVUMDS"/>
   <MissionKit Edition="Enterprise XML Developers" ToolCodes="XSMFSVDDSASW"/>
   <MissionKit Edition="Professional XML Developers" ToolCodes="XSMFSV"/>
</AltovaTools>
```

どのAltova 製品が、MissionKit の各エディションに含まれているかを表すリストを生成します。

マッピングのしくみ:

- **SelectMissionKit** 入力コンポーネントはマッピングへの パラメーターとしての役割を果たします。定数からデフォルトの値を受け取ります。この場合は[Enterprise XML Developers] です。
- `Equal` 関数はソース XML ファイルからの**Edition** アイテムと共に パラメータとして提供されているエディションを比較し、結果を **ToolCodes** フィルターの**bool** パラメーターに パスします。
- **ToolCodes** フィルターの**node/row** 入力がソースコンポーネントの**ToolCodes** アイテムから接続されています。[Enterprise XML Developers] エディションの値: `XSMFSVDDSASW`.
- `XSMFSVDDSASW` 値は**on-true** パラメーターに パスされ、更に`tokenize-by-length` 関数の**input** パラメーターに パスされます。
- `tokenize-by-length` 関数は値 `XSMFSVDDSASW` をそれぞれ2文字ずつ持つ複数のチャンクに分割します。**length** パラメーターは**2** です。このために6 個のチャンクが結果として作成されます。
- 各チャンクは(総合して9 個のアイテムからなる)ソースファイルからの2-文字 **Code** 値に対して比較されます。比較の結果がフィルター(filter) の**bool** パラメーターへ渡されます。`tokenize-by-length` 関数から生成された全てのの要素がフィルターの **node/row** パラメーターへ渡されている点に注意してください。
- `exists` 関数はフィルターコンポーネントの**on-true** パラメーターにより パスされた既存/非既存ノードをチェックします。既存するノードとは **ToolCodes** チャンクと**Code** 値の間で一致するものです。非既存のノードは**Code** 値に一致する**ToolCodes** チャンクが存在しないものです。
- `exists` 関数の各 **bool** 結果 は**if-else** コンポーネントに パスされます。ノードが存在する場合、または[Y] を、ノードが存在しない場合 [N] をターゲット内で生成する。

マッピングの結果は以下のようになります:

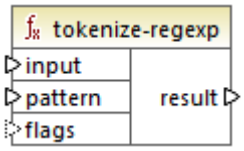
```
Tool;MissionKit for Enterprise XML Developers
XMLSpy;Y
MapForce;Y
StyleVision;Y
UModel;N
DatabaseSpy;N
```

```
DiffDog;Y
SchemaAgent;Y
SemanticWorks;Y
Authentic;N
```

7.10.10.13 tokenize-regexp

入力文字列を文字列のシーケンスに分割します。引数として与えられている正規表現 **pattern** に一致するサブ文字列はセパレーターを定義します。一致する（セパレーター）文字列は関数により返された結果に含まれていません。

メモ C++、C#、またはJava コードを生成する際には、正規表現構文の高度な機能が異なる可能性があります。詳細に関しては各言語の正規表現ドキュメントを確認してください。



言語

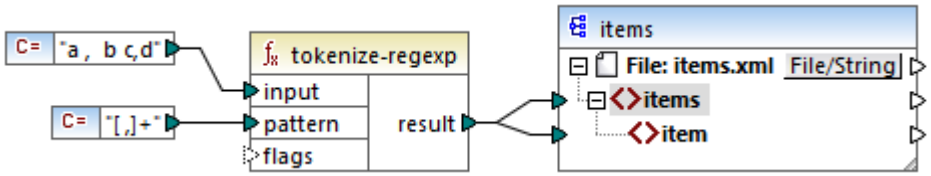
Built-in、C++、C#、Java、XQuery、XSLT 2.0、XSLT 3.0。

パラメーター

名前	説明
input	入力文字列。
pattern	正規表現パターンを提供します。パターンに一致するサブ文字列は区切り文字として扱われます。正規表現に関する更に詳しい情報については正規表現のセクションを参照ください。
flags	任意のパラメーターです。使用される正規表現 flags を提供します。例えば、フラグ "i" はマッピングに大文字と小文字を区別して処理するように命令します。

サンプル

下で紹介されているマッピングのゴールは文字列 `a 、 b c、d` を各アルファベット文字がシーケンス内のアイテムである個所で文字列のシーケンスに分割することです。冗長空白スペースまたはコンマを削除する必要があります。



このゴールを達成するには正規表現パターン `[ ,]+` が`tokenize-regexp` 関数のパラメーターとして提供される必要があります。このパターンは以下を意味します:

(C) 2015-2021 Altova GmbH

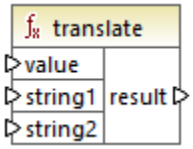
- 文字クラス `[ ,]` 内の文字のいずれかに一致します。このために、入力文字列内でコンマまたは空白文字が発生すると分割が行われます。
- 量指定子 `+` は一致する先頭の文字クラスの1つまたは複数の発生を指定します。量指定子がない場合は文字列の結果シーケンス内に各空白文字またはコンマが個別のアイテムを作成します。これは期待されない結果です。

マッピングの出力は以下のようになります:

```
<items>
   <item>a</item>
   <item>b</item>
   <item>c</item>
   <item>d</item>
</items>
```

7.10.10.14 translate

文字の置換による文字を実行します。**string1** 内に含まれている文字のために **value** を確認し **string2** 内の同じポジションにあるもののために各文字を置き換えます。**string2** に対応する文字が無い場合、その文字は削除されます。



言語

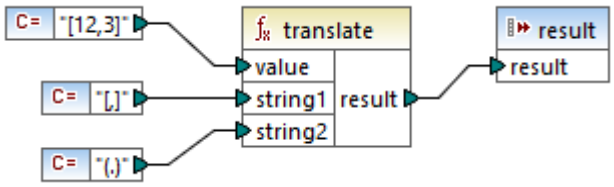
Built-in、C++、C#、Java、XQuery、XSLT 1.0、XSLT 2.0、XSLT 3.0。

パラメーター

名前	説明
value	入力文字列。
string1	検索文字のリストを提供します。文字列内の各文字のポジションは重要です。
string2	置換文字のリストを提供します。各置換文字のポジションは **string1** 内のものに対応する必要があります。

サンプル

文字列 `[12,3]` を `(12.3)` に変換すると仮定します。具体的には、角かっこが丸かっこと置き換えられる必要があり、コンマはピリオドと置き換えられる必要があります。これを達成するには `translate` 関数を以下のように呼び出します:



上のマッピングでは、最初の定数は処理される入力文字列を提供します。2番目と3番目の定数はそれぞれ**string1** と**string2** として文字のリストを提供します。

string1 `[,]`

string2 `(.)`

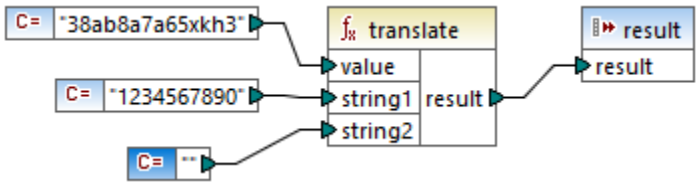
string1 と**string2** には同じ数の文字が存在することに注意してください。**string1** 内の各文字のために、同じポジションの**string2** からの等価の文字が置換として使用されます。この結果、次の置換が発生します:

- 各 `[` は `(` と置き換えられます
- 各 `,` は `.` と置き換えられます
- 各 `]` は `]` と置き換えられます

マッピングの出力は以下のようになります:

```
(12.3)
```

この関数は文字列から特定の文字を削除するために使用することができます。これを達成するには**string1** パラメーターを削除する文字に設定し、を空の文字列を**string2** に設定します。例えば下のマッピングは文字列 `38ab8a7a65xkh3` からすべての小数を削除します。



マッピングの出力は以下のようになります:

```
abaaxkh
```

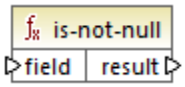
7.10.11 db

db ライブラリは、データベース内にNULL フィールドが発生すると、マッピングの結果を定義することのできる関数を含みます。

(C) 2015-2021 Altova GmbH

7.10.11.1 is-not-null

フィールドがnull の場合 **false** を返します。それ以外の場合は**true** を返します。



言語

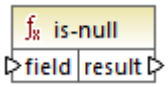
Built-in、C++、C#、Java。

パラメーター

名前	説明
field	データベースフィールド

7.10.11.2 is-null

フィールドがnull の場合 **true** を返します。それ以外の場合は**false** を返します。



言語

Built-in、C++、C#、Java。

パラメーター

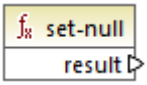
名前	説明
field	データベースフィールド

7.10.11.3 set-null

データベースフィールドをNULL をセットします。この関数はデフォルト値をNULL に上書きします。データベースフィールド以外のアイテムへ接続されている場合、空のシーケンスとして振る舞います。以下の点にご注意してください：

- 他の関数へ`set-null` を接続しても、通常その関数は呼ばれません。（例えば`count` のような）シーケンス関数へ`set-null` を接続すると、空のシーケンスが呼び出しに使用されます。
- `set-null` をフィルターに接続するとif-else 条件は予想通り作動します。フィールドはNULL に設定されます。フィルターには[node/row] 入力を意味します。

- `set-null` から `simpleType` への接続を行っても、ターゲットコンポーネント内に要素が作成されることはありません。

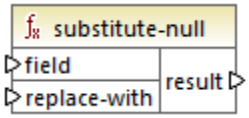


言語

Built-in、C++、C#、Java。

7.10.11.4 substitute-null

NULL ではない場合フィールド自身を返します。それ以外の場合 **replace-with** が返されます。



言語

Built-in、C++、C#、Java。

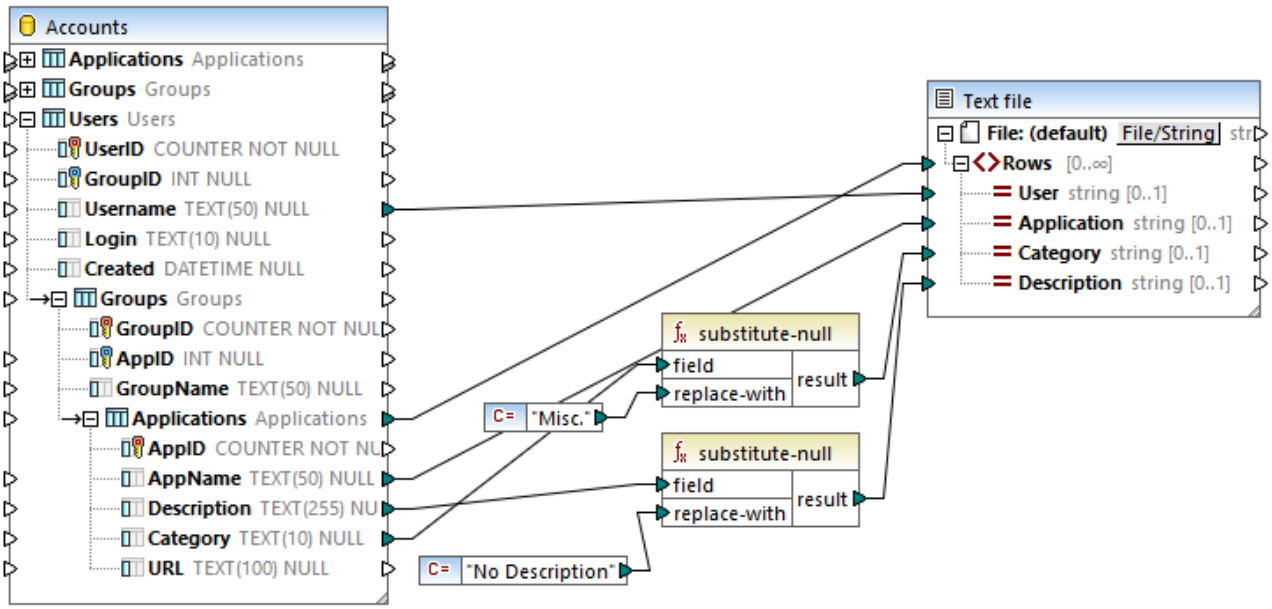
パラメーター

名前	説明
field	データベースフィールド
replace-with	代替の値。

サンプル

`substitute-null` 関数が実際に使用されている例が以下に示されています。このマッピングは **DB-ApplicationList.mfd** と呼ばれ＜マイドキュメント＞¥Altova¥MapForce2021¥MapForceExamples¥フォルダー内で見つけることができます。

(C) 2015-2021 Altova GmbH



[Applications] テーブルを含むAccess データベースからのデータを読み取ります。

	AppID	AppName	Description	Category	URL
1	1	Altova MapForce	The premier data mapping tool.	IDE	www.altova.com/xmlspy
2	2	Notepad	*[NULL]*	*[NULL]*	*[NULL]*

最初の関数により、[Applications] テーブル内の**Category** フィールドがNULL であるかチェックされます。このフィールドはNotepad アプリケーションのためにNULL のため、代替の値 [Misc] がテキストファイルの**Category** アイテムにマップされます。

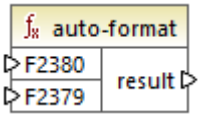
2番目の関数は**Description** フィールドがNULL であるかチェックします。このフィールドはNotepad アプリケーションのためにNULL のため、代替の値 [No description] がターゲットファイルの**Description** アイテムにマップされます。

7.10.12 edifact

edifact ライブラリ内の関数はUN/EDIFACT コンポーネントに関連するマッピングに適用可能です。UN/EDIFACT を参照してください。関数はEDI 日付、時刻、期間の値を `xs:date` フォーマットに変換するために使用することができます。

7.10.12.1 auto-format

データ要素 **F2379** ([日付または時刻または期間フォーマットコード]) を使用してデータ要素 **F2380** ([日付、時刻、期間の値]) から抽出された `date`、`time`、または `datetime` 値を返します。



言語

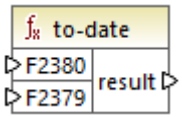
Built-in、C++、C#、Java。

パラメーター

名前	説明
F2380	この入力はUN/EDIFACT コンポーネントの**F2380** アイテムに接続されている必要があります。
F2379	この入力はUN/EDIFACT コンポーネントの**F2379** アイテムに接続されている必要があります。

7.10.12.2 to-date

データ要素 **F2379** ([日付または時刻または期間フォーマットコード]) を使用してデータ要素 **F2380** ([日付、時刻、期間の値]) から抽出された `date` 値を返します。



言語

Built-in、C++、C#、Java。

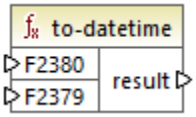
パラメーター

名前	説明
F2380	この入力はUN/EDIFACT コンポーネントの**F2380** アイテムに接続されている必要があります。
F2379	この入力はUN/EDIFACT コンポーネントの**F2379** アイテムに接続されている必要があります。

7.10.12.3 to-datetime

データ要素 **F2379** ([日付または時刻または期間の値]) を使用して書式設定されたデータ要素 **F2380** ([日付または時刻または期間フォーマットコード]) から抽出された `datetime` 値を返します。

(C) 2015-2021 Altova GmbH



言語

Built-in、C++、C#、Java。

パラメーター

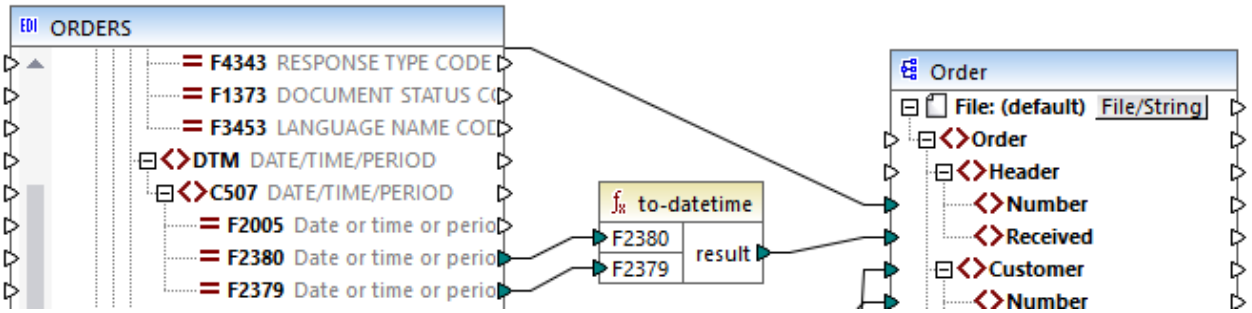
名前	説明
F2380	この入力はUN/EDIFACT コンポーネントの**F2380** アイテムに接続されている必要があります。
F2379	この入力はUN/EDIFACT コンポーネントの**F2379** アイテムに接続されている必要があります。

サンプル

下で表示されているコードリスティングは データ要素 **F2380** と**F2379** が"DTM"で開始するライン内で表示されている個所の UN/EDIFACT ファイルのフラグメントを表示しています。具体的には**F2380** の値は`200404301742PDT` で**F2379** の値は`303` です。

```
UNB+UNOB:1+003897733:01:MFGB+PARTNER ID:ZZ:ROUTING ADDR+970101:1230+00000000000001+
+ORDERS++++1'
UNH+0001+ORDERS:D:19B:UN'
BGM+221+ABC123456XYZ+9'
DTM+4:200404301742PDT:303'
FTX+PUR+3++Pizza purchase order'
RFF+CT:123-456'
RFF+CR:1122'
NAD+SE+999::92++24h Pizza+Long Way+San-Francisco+CA+34424+US'
CTA+SR+:Ted Little'
```

下で説明されているマッピングは`to-datetime` 関数のヘルプを使用して**F2379** を使用して**F2380** の値を抽出します。このマッピングは**EDI-Order.mfd** と呼ばれ＜マイドキュメント＞¥Altova¥MapForce2021¥MapForceExamples¥フォルダー内で見つけることができます。

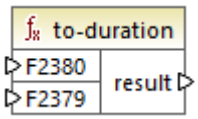


マッピング出力の関連した部分は以下に表示されています。

```
<?xml version="1.0" encoding="UTF-8"?>
<Order>
   <Header>
      <Number>ABC123456XYZ</Number>
      <Received>2004-04-30T17:42:00-09:00</Received>
   </Header>
   <!-- ...-->
</Order>
```

7.10.12.4 to-duration

データ要素 **F2379** ([日付または時刻または期間フォーマットコード]) を使用してデータ要素 **F2380** ([日付、時刻、期間の値]) から抽出された `duration` 値を返します。



言語

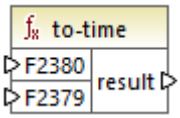
Built-in、C++、C#、Java。

パラメーター

名前	説明
F2380	この入力はUN/EDIFACT コンポーネントの**F2380** アイテムに接続されている必要があります。
F2379	この入力はUN/EDIFACT コンポーネントの**F2379** アイテムに接続されている必要があります。

7.10.12.5 to-time

データ要素 **F2379** ([日付または時刻または期間フォーマットコード]) を使用してデータ要素 **F2380** ([日付、時刻、期間の値]) から抽出された `time` 値を返します。



言語

Built-in、C++、C#、Java。

(C) 2015-2021 Altova GmbH

パラメーター

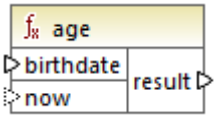
名前	説明
F2380	この入力はUN/EDIFACT コンポーネントの**F2380** アイテムに接続されている必要があります。
F2379	この入力はUN/EDIFACT コンポーネントの**F2379** アイテムに接続されている必要があります。

7.10.13 lang | datetime functions（日付時刻関数）

lang ライブラリからの日付と時刻関数は日付、時刻、期間を操作するために使用することができます。**core** ライブラリからの日付と時刻関数とは異なり、これらの関数はビルトイン、Java、C#、またはC++ 言語を選択すると使用することができます。

7.10.13.1 age

引数として与えられた生年月日から今日まで経過した年数を数値で返します。



言語

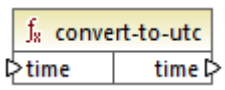
Built-in、C++、C#、Java。

パラメーター

名前	型	説明
birthdate	`xs:date`	必須。`xs:date` 値として生年月日を提供します。
now	`xs:date`	任意のパラメーターです。デフォルトは現在のシステムの日付です。値がnow 引数にマップされると、関数は生年月日と現在の日付の差分を年数で返します。

7.10.13.2 convert-to-utc

UTC（世界協定時刻）に引数として提供されている時刻の値を変換します。関数は（例えば、[+5:00] などの）タイムゾーンコンポーネントを考慮します。



言語

Built-in、C++、C#、Java。

パラメーター

名前	型	説明
time	`xs:dateTime`	変換する `xs:dateTime` 値を与えます。

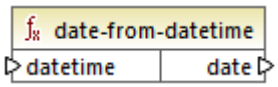
サンプル

入力値が `2001-12-17T09:30:02+05:00` の場合、関数の結果は `2001-12-17T04:30:02` です。

入力値が `2001-12-17T09:30:02Z` の場合、関数の結果は `2001-12-17T09:30:02` です。この場合、行末 [Z] が既にこの時刻を [UTC] と同じゼロ（または「ズールー」）時刻に設定しているため変換は発生しません。

7.10.13.3 date-from-datetime

引数として与えられた `xs:dateTime` 値から *date* の部分を返します。*time* 部分はゼロに設定されています。タイムゾーンは変更されていません。



言語

Built-in、C++、C#、Java。

パラメーター

名前	型	説明
datetime	`xs:dateTime`	処理する `xs:dateTime` 値を与えます。

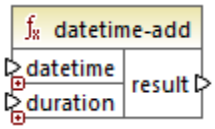
サンプル

入力値が `2001-12-17T09:30:02+05:00` の場合、関数の結果は `2001-12-17T00:00:00+05:00` です。

(C) 2015-2021 Altova GmbH

7.10.13.4 datetime-add

期間（2番目の引数）をdatetime（最初の引数）に足して得られた`xs:dateTime`値を返します。



言語

Built-in、C++、C#、Java。

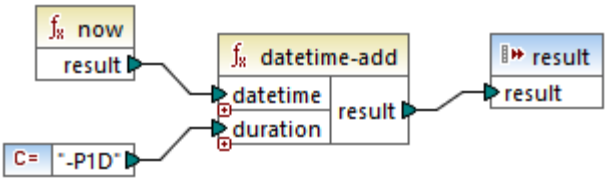
パラメーター

名前	型	説明
datetime	`xs:dateTime`	入力として使用される`xs:dateTime`値を与えます。
duration	`xs:duration`	`xs:duration`値を提供します。 サンプルの期間は`P1Y2M3DT04H05M59S`： • [P]が期間の指定子で、必須です; • 残りの文字は以下の順序で表示されます: 1年、2カ月、3日、T（時間指定子）、04時間、05分、59秒。 マイナス文字が[P]指定子の前に表示されると、これはマイナスの期間を示しています。例: `-P1D`.

サンプル

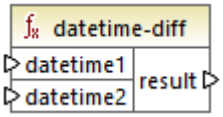
入力**datetime**値が`2001-12-17T09:30:02+05:00`と想定します。**duration**が`P10D`（10日）の場合関数の結果は`2001-12-27T00:00:00+05:00`になります。

昨日の日付を取得するには`now`関数を**datetime**入力に接続します。下のマッピングでは期間`-P1D`は[マイナス1日]を意味し、マッピングは昨日の日付を返します。



7.10.13.5 datetime-diff

datetime2（2番目の引数）を**datetime1**（1番目の引数）から引き算し得られた期間を返します。結果は文字列または期間データ型にマップすることができます。



言語

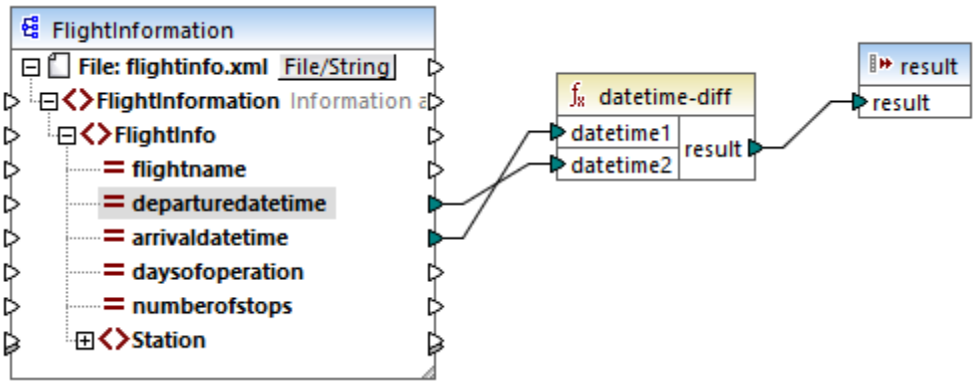
Built-in、C++、C#、Java。

パラメーター

名前	型	説明
datetime1	`xs:dateTime`	最初の`xs:dateTime` 値を提供します。
datetime2	`xs:duration`	2番目の`xs:dateTime` 値を提供します。

サンプル

下に表示されるマッピングでは`datetime-diff` 関数は飛行機の離陸日時 datetime `2001-12-17T09:30:02+05:00` を到着日時 `2001-12-17T19:30:02+05:00` から引き算します。到着日時は離陸時間より先大きな値であり、関数の最初の入力に接続されています。



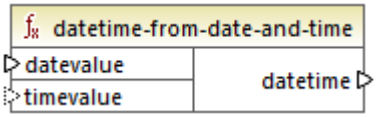
マッピングの出力は2つの値の差分です（10時間の期間）:

```
PT10H
```


(C) 2015-2021 Altova GmbH

7.10.13.6 datetime-from-date-and-time

xs:date 値（最初の引数）と xs:time 値（2番目の引数）からビルドされた xs:dateTime 値を返します。結果は文字列型または xs:dateTime データ型へマッピングすることができます。



言語

Built-in、C++、C#、Java。

パラメーター

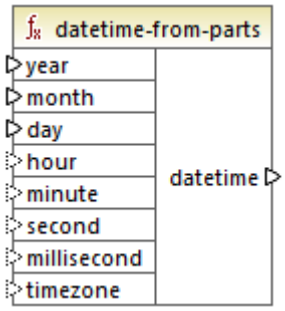
名前	型	説明
datevalue	**xs:date**	型 xs:date の値を提供します。
timevalue	**xs:time**	型 xs:time の値を提供します。

サンプル

最初の引数が2012-06-29 で2番目の引数が11:59:55 の場合、関数は2012-06-29T11:59:55 を返します。

7.10.13.7 datetime-from-parts

以下のパーツを組み合わせて引数としてビルドされた型 xs:dateTime の値: 年、月、時間、分、秒、ミリ秒、タイムゾーン。この関数は自動的に提供されたパラメーターを正規化します。例えば、1月32日は自動的に2月1日に変換されます。



言語

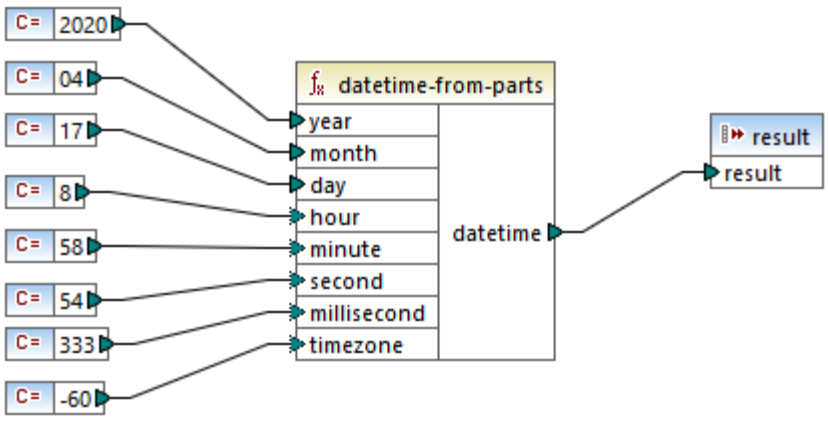
Built-in、C++、C#、Java。

パラメーター

名前	型	説明
year	`xs:int`	年の値を与えます。
month	`xs:int`	月の値を与えます。
day	`xs:int`	月の日付を与えます。
hour	`xs:int`	任意。時間の値を与えます。
minute	`xs:int`	任意。分の値を与えます。
second	`xs:int`	任意。秒の値を与えます。
millisecond	`xs:decimal`	任意。ミリ秒の値を与えます。
timezone	`xs:int`	任意。タイムゾーンを分数で与えます。この値は負の値であることができます。

サンプル

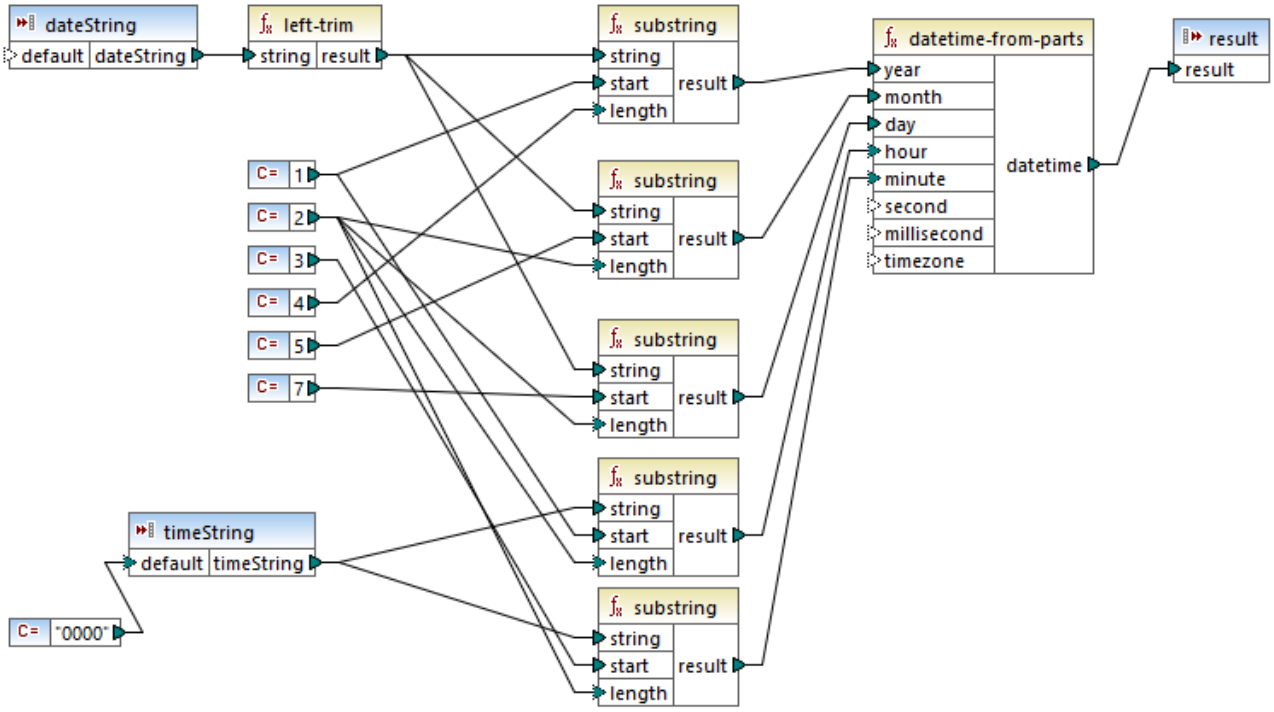
以下のマッピングは `xs:dateTime` 値を定数により提供される部分から作成します。



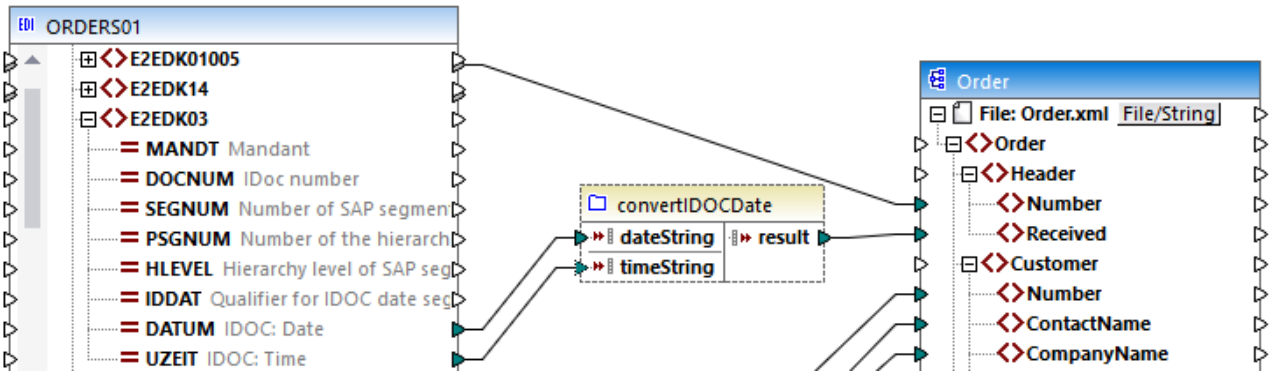
マッピングの出力は `2020-04-17T08:58:54.333-01:00` です。

更に詳細なサンプルに関しては<**マイドキュメント**>¥Altova¥MapForce2021¥MapForceExamples¥フォルダー内の **IDoc_Order.mfd** マッピングを参照してください。このマッピングでは SAP Idoc インスタンスファイルからのデータの読み取り方法について説明されています。**left-trim** と **substring** 関数を使用して `xs:dateTime` 値を2つの入力文字列から作成するユーザー定義関数 [convertIDOCDate] が含まれています。

(C) 2015-2021 Altova GmbH



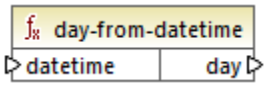
dateString と**timeString** コンポーネントはユーザー定義関数のパラメーターです。これらはSAP Idoc インスタンスファイルのメインマッピングから提供されています:



このマッピングでは文字列としての日付の値は`19990621` です。時刻の値は`0930` です。結果 `datetime-from-parts` 関数は値 `1999-06-21T09:30:00` を返します。この値はユーザー定義関数の結果で、ターゲット XML ドキュメントの**Received** アイテムに書き込まれます。

7.10.13.8 day-from-datetime

引数として与えられた`xs:dateTime` 値から整数の値として日数を返します。



言語

Built-in、C++、C#、Java。

パラメーター

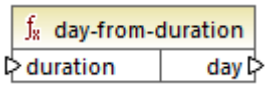
名前	型	説明
datetime	**xs:dateTime**	型 `xs:dateTime` の入力値を提供します。

サンプル

datetime が`2001-12-17T10:30:03+01:00` の場合、関数は`17` を返します。

7.10.13.9 day-from-duration

引数として与えられた`xs:duration` 値から整数の値として日数を返します。



言語

Built-in、C++、C#、Java。

パラメーター

名前	型	説明
duration	**xs:duration**	型 `xs:duration` の入力値を提供します。

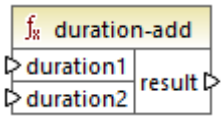
サンプル

duration が`P1Y2M3DT10H30M` の場合 `day-from-datetime` 関数は`3` を返します。

(C) 2015-2021 Altova GmbH

7.10.13.10 duration-add

2つの期間を足し合わせることで得られる期間を返します。



言語

Built-in、C++、C#、Java。

パラメーター

名前	型	説明
duration1	**xs:duration**	型 xs:duration の入力値を提供します。
duration2	**xs:duration**	型 xs:duration の2番目の入力値を提供します。

サンプル

最初の期間は`P0Y0M3DT03H0M`(3日と3時間) で2番目の期間は`P0Y0M3DT01H0M`(3日と1時間) の場合、関数は`P6DT4H` (6日と4時間)を返します。

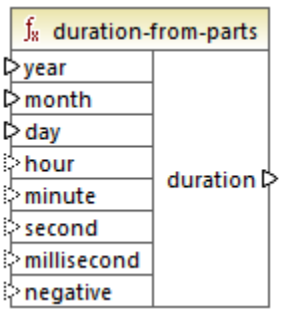
7.10.13.11 duration-from-parts

引数として提供された以下の部分を組み合わせて計算された型 xs:duration の値を返します: year、month、day、hour、minute、second、millisecond、negative。

サンプルの期間は`P1Y2M3DT04H05M59S`:

- [P] が期間の指定子で、必須です;
- 残りの文字は以下の順序で表示されます: 1 年、2 カ月、3 日、T(時間指定子)、04 時間、05 分、59 秒。

マイナス文字が[P] 指定子の前に表示されると、これはマイナスの期間を示しています。例: `-P1D`.



言語

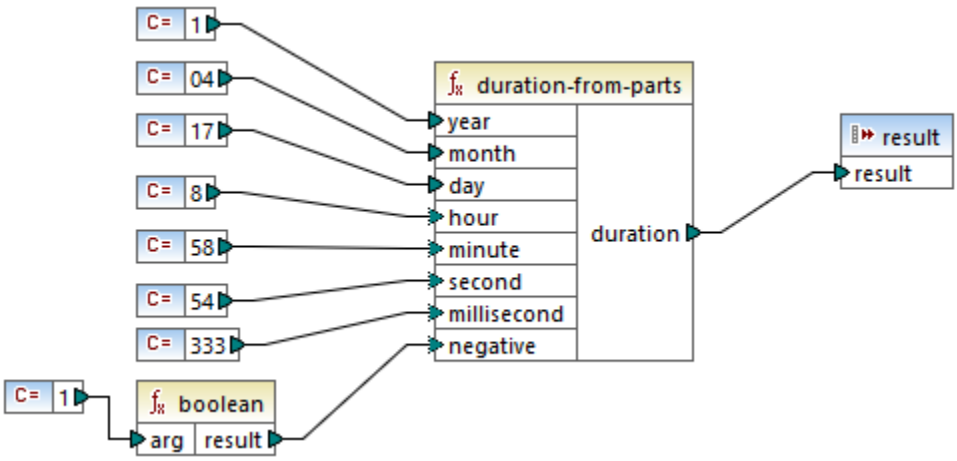
Built-in、C++、C#、Java。

パラメーター

名前	型	説明
year	`xs:int`	年の値を与えます。
month	`xs:int`	月の値を与えます。
day	`xs:int`	月の日付を与えます。
hour	`xs:int`	任意。時間の値を与えます。
minute	`xs:int`	任意。分の値を与えます。
second	`xs:int`	任意。秒の値を与えます。
millisecond	`xs:decimal`	任意。ミリ秒の値を与えます。
negative	`xs:boolean`	任意。負の期間の場合 **true** である必要があり、それ以外の場合は **false** です。

サンプル

以下のマッピングは1 年、4 カ月、17 日、8 時間、58 分、54.333 秒の負の期間を生成します。



マッピング出力は `-P1Y4M17DT8H58M54.333S` です。

7.10.13.12 duration-subtract

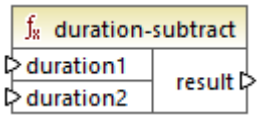
duration1 から **duration2** を減算し取得された `xs:duration` 値を返します。

サンプルの期間は `P1Y2M3DT04H05M59S` :

(C) 2015-2021 Altova GmbH

- [P] が期間の指定子で、必須です;
- 残りの文字は以下の順序で表示されます: 1 年、2 カ月、3 日、T (時間指定子)、04 時間、05 分、59 秒。

マイナス文字が[P] 指定子の前に表示されると、これはマイナスの期間を示しています。例: `-P1D`.



言語

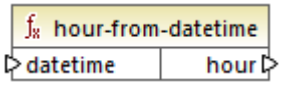
Built-in、C++、C#、Java。

サンプル

duration1 が`P0Y0M0DT05H07M` (5 時間 7 分) で**duration2** が`PT1H` (1 時間) の場合、関数は`PT4H7M` (4 時間 7 分)を返します。

7.10.13.13 hour-from-datetime

引数として与えられた`xs:dateTime` 値から整数の値として時間数を返します。



言語

Built-in、C++、C#、Java。

パラメーター

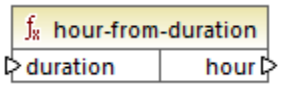
名前	型	説明
datetime	`xs:dateTime`	型 `xs:dateTime` の入力値を提供します。

サンプル

datetime が`2001-12-17T09:30:02+05:00` の場合、関数は`9` を返します。

7.10.13.14 hour-from-duration

引数として与えられた `xs:duration` 値から整数の値として時間数を返します。



言語

Built-in、C++、C#、Java。

パラメーター

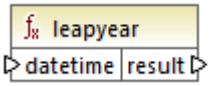
名前	型	説明
duration	**xs:duration**	型 `xs:duration` の入力値を提供します。

サンプル

duration が `P0Y0M0DT05H07M` の場合、関数は `5` を返します。

7.10.13.15 leapyear

現在の `xs:dateTime` クラスがうるう年の場合、**true** を返します。それ以外の場合 **false** を返します。



言語

Built-in、C++、C#、Java。

パラメーター

名前	型	説明
datetime	**xs:dateTime**	型 `xs:dateTime` の入力値を提供します。

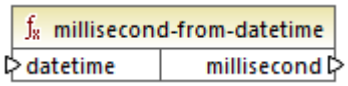
サンプル

datetime が `2020-04-17T09:30:02+02:00` の場合、2020 はうるう年のため関数は **true** を返します。

(C) 2015-2021 Altova GmbH

7.10.13.16 millisecond-from-datetime

引数として与えられた`xs:dateTime` 値から整数の値としてミリ秒数を`xs:decimal` として返します。



言語

Built-in、C++、C#、Java。

パラメーター

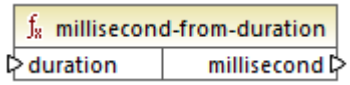
名前	型	説明
datetime	**xs:dateTime**	型 `xs:dateTime` の入力値を提供します。

サンプル

datetime が`2001-12-17T09:30:02.544+05:00` の場合、関数は`544` を返します。

7.10.13.17 millisecond-from-duration

引数として与えられた`xs:duration` 値から整数の値としてミリ秒数を`xs:decimal` として返します。



言語

Built-in、C++、C#、Java。

パラメーター

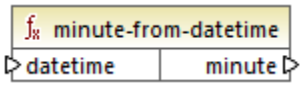
名前	型	説明
duration	**xs:duration**	型 `xs:duration` の入力値を提供します。

サンプル

duration が`P0Y0M0DT05H07M02.227S` の場合、関数は`227` を返します。

7.10.13.18 minute-from-datetime

引数として与えられている `xs:dateTime` 値から整数の値として分数を返します。



言語

Built-in、C++、C#、Java。

パラメーター

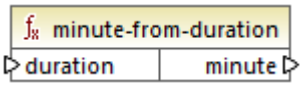
名前	型	説明
datetime	**xs:dateTime**	型 `xs:dateTime` の入力値を提供します。

サンプル

datetime が `2001-12-17T09:30:02.544+05:00` の場合、関数は `30` を返します。

7.10.13.19 minute-from-duration

引数として与えられた `xs:duration` 値から整数の値として分数を返します。



言語

Built-in、C++、C#、Java。

パラメーター

名前	型	説明
duration	**xs:duration**	型 `xs:duration` の入力値を提供します。

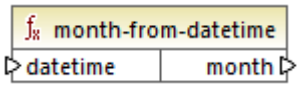
サンプル

duration が `P0Y0M0DT05H07M02.227S` の場合、関数は `7` を返します。

(C) 2015-2021 Altova GmbH

7.10.13.20 month-from-datetime

引数として与えられた`xs:dateTime`値から整数の値として月数を返します。



言語

Built-in、C++、C#、Java。

パラメーター

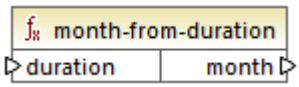
名前	型	説明
datetime	**xs:dateTime**	型`xs:dateTime`の入力値を提供します。

サンプル

datetimeが`2001-12-17T09:30:02.544+05:00`の場合、関数は`12`を返します。

7.10.13.21 month-from-duration

引数として与えられた`xs:duration`値から整数の値として月数を返します。



言語

Built-in、C++、C#、Java。

パラメーター

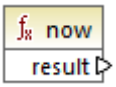
名前	型	説明
duration	**xs:duration**	型`xs:duration`の入力値を提供します。

サンプル

durationが`P0Y04M0DT05H07M02.227S`の場合、関数は`4`を返します。

7.10.13.22 now

(タイムゾーンを含む)現在の日時を `xs:dateTime` 値として返します。

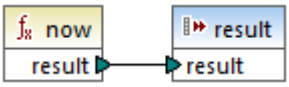


言語

Built-in、C++、C#、Java。

サンプル

以下のマッピングは現在の日時を出力します: 出力はマッピングを実行する都度変更されます。

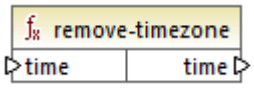


サンプルの出力は `2020-04-17T11:42:34.684+02:00` です。

1日前の値を得る方法については、core | lang | datetime-add 関数を参照ください。

7.10.13.23 remove-timezone

(型 `xs:dateTime` の) **time**入力パラメーターからのタイムゾーンコンポーネントを削除します。



言語

Built-in、C++、C#、Java。

パラメーター

名前	型	説明
time	**xs:dateTime**	型 `xs:dateTime` の入力値を提供します。

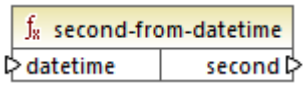
サンプル

time が `2001-12-17T09:30:02+05:00` の場合、関数は `2001-12-17T09:30:02` を返します。

(C) 2015-2021 Altova GmbH

7.10.13.24 second-from-datetime

引数として与えられた`xs:dateTime` 値から整数の値として秒数を返します。



言語

Built-in、C++、C#、Java。

パラメーター

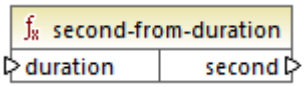
名前	型	説明
datetime	**xs:dateTime**	型 `xs:dateTime` の入力値を提供します。

サンプル

datetime が`2001-12-17T09:30:02.544+05:00` の場合、関数は`2` を返します。

7.10.13.25 second-from-duration

引数として与えられた`xs:duration` 値から整数の値として秒数を返します。



言語

Built-in、C++、C#、Java。

パラメーター

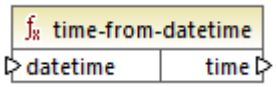
名前	型	説明
duration	**xs:duration**	型 `xs:duration` の入力値を提供します。

サンプル

duration が`P0Y04M0DT05H07M02.227S` の場合、関数は`2` を返します。

7.10.13.26 time-from-datetime

引数として与えられた `xs:dateTime` 値から `xs:time` 値として時間コンポーネントを返します。



言語

Built-in、C++、C#、Java。

パラメーター

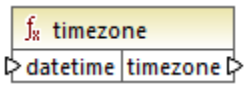
名前	型	説明
datetime	**xs:dateTime**	型 `xs:dateTime` の入力値を提供します。

サンプル

datetime が `2001-12-17T09:30:02.544+05:00` の場合、関数は `09:31:02+05:00` を返します。

7.10.13.27 timezone

引数として与えられた `xs:dateTime` 値からタイムゾーンオフセットを分数で返します。UTC のために 0 を返します。



言語

Built-in、C++、C#、Java。

パラメーター

名前	型	説明
datetime	**xs:dateTime**	型 `xs:dateTime` の入力値を提供します。

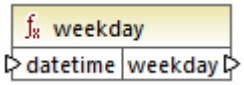
サンプル

datetime が `2001-12-17T09:30:02.544+05:00` の場合、関数は `300` を返します。

(C) 2015-2021 Altova GmbH

7.10.13.28 weekday

引数として与えられた `xs:dateTime` 値から整数の値として曜日を返します。関数は値 **1** を月曜のために、値 **2** を火曜のために返します。



言語

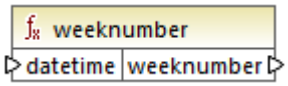
Built-in、C++、C#、Java。

パラメーター

名前	型	説明
datetime	**xs:dateTime**	型 `xs:dateTime` の入力値を提供します。

7.10.13.29 weeknumber

引数として与えられた `xs:dateTime` 値から週番号を返します。関数は値 **1** を年度の最初の週のために、値 **2** を2番目の週のために返します。



言語

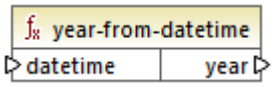
Built-in、C++、C#、Java。

パラメーター

名前	型	説明
datetime	**xs:dateTime**	型 `xs:dateTime` の入力値を提供します。

7.10.13.30 year-from-datetime

引数として与えられた`xs:dateTime` 値から整数の値として年数を返します。



言語

Built-in、C++、C#、Java。

パラメーター

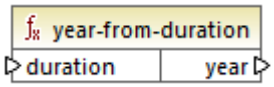
名前	型	説明
datetime	**xs:dateTime**	型 `xs:dateTime` の入力値を提供します。

サンプル

datetime が`2001-12-17T09:30:02.544+05:00` の場合、関数は`2001` を返します。

7.10.13.31 year-from-duration

引数として与えられた`xs:duration` 値から整数の値として年数を返します。



言語

Built-in、C++、C#、Java。

パラメーター

名前	型	説明
duration	**xs:duration**	型 `xs:duration` の入力値を提供します。

サンプル

duration が`P01Y04M0DT05H07M02.227S` の場合、関数は`1` を返します。

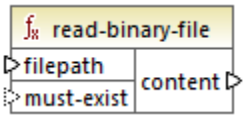
(C) 2015-2021 Altova GmbH

7.10.14 lang | file functions (ファイル関数)

バイナリコンテンツをファイルから読み取る、またはファイルへバイナリコンテンツを書き込むために**file** 関数を**lang** ライブラリから使用します。

7.10.14.1 read-binary-file

この関数は型 `xs:base64Binary` のBLOB (バイナリラージオブジェクト) として指定されたファイルのコンテンツを返します。データ型が [base64Binary] と呼ばれる場合でも内部の表記はBLOB です。関数の結果を型 `xs:base64Binary` のXML ノードにマップする場合のみ、ベース64 エンコードになります。関数の結果を `xs:hexBinary`、データベースブロブ、またはプロトコールバッファー構造内のバイナリフィールドにマップすることもできます。



バイナリファイルをマッピングに読み込むには**filepath** 引数に対しての入力としてパスを提供します。**filepath** が相対的な場合 MapForce はマッピングと同じディレクトリ内でファイルを探します。**must-exist** 引数が任意です。ファイルを開くことはできず、このパラメーターは**true**, になり、マッピングはエラーをスローします。ファイルを開くことができなく、このパラメーターが**false** の場合、からのバイナリが返されます。

言語

Built-in。

パラメーター

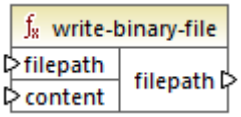
名前	型	説明
filepath	`xs:string`	ファイルパス。
must-exist	`xs:boolean`	任意のパラメーターです。ファイルを開くことができなく、このパラメーターが**true** の場合、マッピングはエラーをスローします。ファイルを開くことができなく、このパラメーターが**false** の場合、からのバイナリが返されます。 デフォルトの値は、**true** です。

サンプル

例: バイナリファイルの読み取りを参照してください。

7.10.14.2 write-binary-file

この関数はバイナリコンテンツを指定されたファイルパスに書き込み、書き込まれたファイルのパスを返します。バイナリファイルが唯一の希望する出力の場合 単純型出力 コンポーネントに関数の結果を接続します。出力がマッピング内で使用される都度この関数はファイルに書き込むため、間に他の処理を使用することなく、関数の結果を直接ターゲットコンポーネントに接続することが奨励されます。



バイナリファイルを書き込むには**filepath** 引数に対しての入力としてパスを提供します。**filepath** が相対的な場合 MapForce はマッピングと同じディレクトリ内でファイルを生成します。The **content** 引数は実際のバイナリコンテンツ（例えば、データベース内のBLOB フィールド）に接続されている必要があります。

MapForce 内でマッピングをプレビューすると、ディスクにファイルを直接書き込む代わりに関数は一時的なファイルをデフォルトで生成します。一時的なファイルをディスクに保存するには、最初に**出力** タブをクリックし適用可能な場合 **生成された出力を保存** または **全ての生成済みの出力** ツールバーボタンをクリックします。

MapForce が出力を一時的なファイルではなくファイナルファイルに直接書き込むように構成するには[**ツール | オプション** メニューコマンド]を選択し [**全般**]をクリックして [**最終出力ファイルに直接書き込む**] オプションを選択します。このオプションは同じ名前を持つ既存のファイルを上書きすることに注意してください。

この関数はファイルがディスクに保存されていない場合（マッピングをプレビューし [**最終出力ファイルに直接書き込む**] オプションが無効化されている場合）でも常に最終（一時的ではない）ファイル名を返します。

> マッピングが自身の出力ファイルを読み取ることはサポートされていないことに注意してください。

言語

Built-in。

パラメーター

名前	型	説明
filepath	`xs:string`	入力ファイルパス。
content	`xs:base64Binary`	型 `xs:base64Binary` のバイナリコンテンツ

サンプル

例: バイナリファイルの書き込みを参照してください。

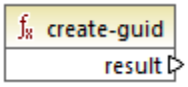
7.10.15 lang | generator functions（ジェネレーター関数）

lang ライブラリからのジェネレーター関数は値を生成する関数です（現在 `create-guid` はこの種類の唯一の関数です）。

7.10.15.1 create-guid

グローバル一意識別子（GUID）を16進数にエンコードされた文字列として作成します。この関数はデータベースフィールド、または他のエンポーネント型のためにマッピングから直接一意の値を生成するために使用されます。関数 format-guid-string を参照してください。

(C) 2015-2021 Altova GmbH



言語

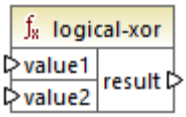
Built-in、C++、C#、Java。

7.10.16 lang | logical functions (論理関数)

lang ライブラリからの関数には**ブール値理論**を使用する多種の値を評価する関数が含まれています。

7.10.16.1 logical-xor

value1 が**value2** と異なる場合 **true** を返し、それ以外の場合は**false** を返します。



言語

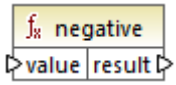
Built-in、C++、C#、Java。

パラメーター

名前	型	説明
value1	`xs:boolean`	最初の入力値。
value2	`xs:boolean`	2番目の入力値。

7.10.16.2 negative

入力値が負の値 (ゼロより小さい値) の場合 **true** を返し、それ以外の場合は**false** を返します。



言語

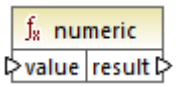
Built-in、C++、C#、Java。

パラメーター

名前	型	説明
value	`xs:decimal`	入力値。

7.10.16.3 numeric

入力値が数値、または文字として解析することができる文字列である場合 **true** を返し、それ以外の場合は **false** を返します。



言語

Built-in、C++、C#、Java。

パラメーター

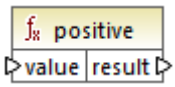
名前	型	説明
value	`xs:decimal`	入力値。

サンプル

入力値が文字列 `"4.33"` の場合、関数は **true** を返します。入力値が文字列 `"4.33 USD"` の場合、関数は **false** を返します。

7.10.16.4 positive

入力値が正の値（ゼロより大きい値）の場合 **true** を返し、それ以外の場合は **false** を返します。



言語

Built-in、C++、C#、Java。

パラメーター

名前	型	説明
value	`xs:decimal`	入力値。

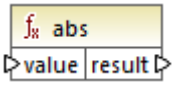
(C) 2015-2021 Altova GmbH

7.10.17 lang | math functions (数学関数)

lang ライブラリからの数学関数をマッピング内で異なる数学オペレーションを実行するために使用することができます。

7.10.17.1 abs

引数として提供された数値の絶対値を返します。引数が負の値ではない場合、引数が返されます。引数が負の値の場合、引数のネゲーションが返されます。



言語

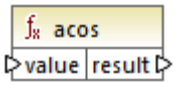
Built-in、C++、C#、Java。

パラメーター

名前	型	説明
value	`xs:decimal`	入力値。

7.10.17.2 acos

`-pi/2` から `pi/2` までの範囲で **value** のアークコサインを返します。



言語

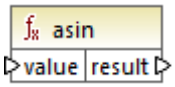
Built-in、C++、C#、Java。

パラメーター

名前	型	説明
value	`xs:double`	入力値。

7.10.17.3 asin

`-pi/2` から`pi/2` までの範囲で **value** のアークサインを返します。



言語

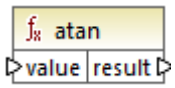
Built-in、C++、C#、Java。

パラメーター

名前	型	説明
value	`xs:double`	入力値。

7.10.17.4 atan

`-pi/2` から`pi/2` までの範囲で **value** のアークタンジェントを返します。



言語

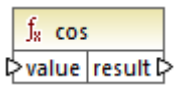
Built-in、C++、C#、Java。

パラメーター

名前	型	説明
value	`xs:double`	入力値。

7.10.17.5 cos

value により与えられた角度の三角コサインのを返します。値のユニットはラジアンです。



(C) 2015-2021 Altova GmbH

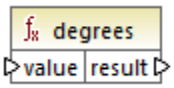
言語
Built-in、C++、C#、Java。

パラメーター

名前	型	説明
value	`xs:double`	入力値。

7.10.17.6 degrees

ラジアンで測定された角度を度数で測定されたほぼ等価の角度に変換します。



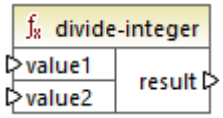
言語
Built-in、C++、C#、Java。

パラメーター

名前	型	説明
value	`xs:double`	入力値。

7.10.17.7 divide-integer

value1 を **value2** で割った整数結果を返します。



言語
Built-in、C++、C#、Java。

パラメーター

名前	型	説明
value1	`xs:decimal`	最初の入力値。

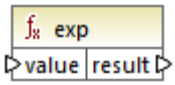
名前	型	説明
value2	`xs:decimal`	2番目の入力値。

サンプル

最初の引数が**15** で2番目の値が**2** の場合、関数は**7** を返します。

7.10.17.8 exp

value に累乗したオイラー数 **e** が返されます。



言語

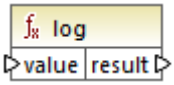
Built-in、C++、C#、Java。

パラメーター

名前	型	説明
value	`xs:double`	入力値。

7.10.17.9 log

value の(底がe の) 自然対数を返します。



言語

Built-in、C++、C#、Java。

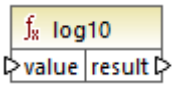
パラメーター

名前	型	説明
value	`xs:double`	入力値。

(C) 2015-2021 Altova GmbH

7.10.17.10 log10

value の(底が10 の) 10 進数対数を返します。



言語

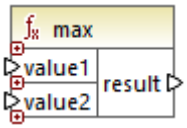
Built-in、C++、C#、Java。

パラメーター

名前	型	説明
value	`xs:double`	入力値。

7.10.17.11 max

引数として提供された数値の最大値の数値を返します。デフォルトでは、この関数には2 つの パラメータのみ存在しますが、更に パラメーターを追加することができます。**パラメーターの追加** (⊕) または **パラメーターの削除** (⊗) をクリックして パラメーターを追加または削除します。関数の引数の追加、または削除も参照してください。



言語

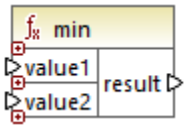
Built-in、C++、C#、Java。

パラメーター

名前	型	説明
value1	`xs:decimal`	最初の入力値。
value2	`xs:decimal`	2番目の入力値。
valueN	`xs:decimal`	*N* 番目の入力値。

7.10.17.12 min

引数として提供された数値の最小値の数値を返します。デフォルトでは、この関数には2つのパラメータのみ存在しますが、更にパラメーターを追加することができます。**パラメーターの追加**（ ）または**パラメーターの削除**（ ）をクリックしてパラメーターを追加または削除します。関数の引数の追加、または削除も参照してください。



言語

Built-in、C++、C#、Java。

パラメーター

名前	型	説明
value1	`xs:decimal`	最初の入力値。
value2	`xs:decimal`	2番目の入力値。
valueN	`xs:decimal`	*N* 番目の入力値。

7.10.17.13 pi

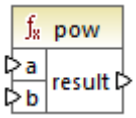
数理定数 *pi* の値を返します。

言語

Built-in、C++、C#、Java。

7.10.17.14 pow

b に累乗された **a** の値を返します。



(C) 2015-2021 Altova GmbH

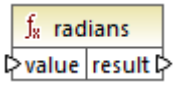
言語
Built-in、C++、C#、Java。

パラメーター

名前	型	説明
a	`xs:double`	値 **a**（底）を返します。
b	`xs:double`	値 **b**（累乗）を返します。

7.10.17.15 radians

度数で測定された角度をラジアンで測定されたほぼ等価の角度に変換します。



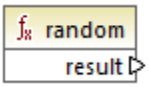
言語
Built-in、C++、C#、Java。

パラメーター

名前	型	説明
value	`xs:double`	入力値。

7.10.17.16 random

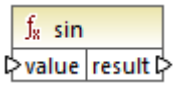
0.0 と等しいまたは大きい、そして 1.0 よりも小さい値を正号と共に返します。戻り値は範囲内からほぼ一様な分布を使用して疑似的に選択されます。



言語
Built-in、C++、C#、Java。

7.10.17.17 sin

value のサインを返します。



言語

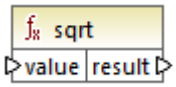
Built-in、C++、C#、Java。

パラメーター

名前	型	説明
value	`xs:double`	入力値。

7.10.17.18 sqrt

value の正確に丸め処理された平方根を返します。



言語

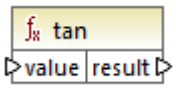
Built-in、C++、C#、Java。

パラメーター

名前	型	説明
value	`xs:double`	入力値。

7.10.17.19 tan

value のタンジェントを返します。



(C) 2015-2021 Altova GmbH

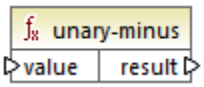
言語
Built-in、C++、C#、Java。

パラメーター

名前	型	説明
value	`xs:double`	入力値。

7.10.17.20 unary-minus

サインされた入力値の負の値を返します。



言語
Built-in、C++、C#、Java。

パラメーター

名前	型	説明
value	`xs:decimal`	入力値。

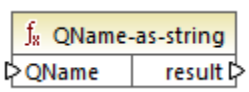
サンプル
入力文字列が**3** の場合、関数は**-3** を返します。入力文字列が**-3** の場合、関数は**3** を返します。

7.10.18 lang | QName functions (QName 関数)

lang ライブラリからのQname 関数は修飾名 (QName) 値を文字列にします。または文字列を修飾名値に変換します。**core** ライブラリからの関数とは異なり、これらの関数はビルトイン、Java、C#、またはC++ 言語でのみ使用することができます。

7.10.18.1 QName-as-string

引数として提供されたQname 値の文字列表記を返します。



言語

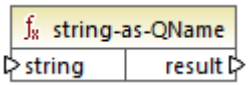
Built-in、C++、C#、Java。

パラメーター

名前	説明
QName	入力 `xs:QName` 値。

7.10.18.2 string-as-QName

QName の文字列表記を QName に変換します。



言語

Built-in、C++、C#、Java。

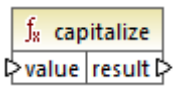
名前	説明
string	入力 `string` 値。

7.10.19 lang | string functions (文字列関数)

lang ライブラリからの文字列関数により文字列を処理することが可能になります(例えば、トリミング、パディング、置換、文字列を大文字または小文字に置き換えるなどの操作)。

7.10.19.1 capitalize

各単語の頭文字が大文字に変換されている入力文字列 **value** でを返します。



言語

Built-in、C++、C#、Java。

(C) 2015-2021 Altova GmbH

パラメーター

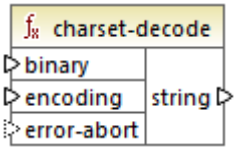
名前	型	説明
value	`xs:string`	入力値。

サンプル

入力文字列が `The quick  brown fox` の場合、関数は`The Quick Brown Fox` を返します。

7.10.19.2 charset-decode

`charset-decode` 関数は、入力文字列データを取り、Base64 テキストとしてデコードします。指定された文字セット（例えば、"utf-8"）に従いデータをデコードし、結果する文字列の値を設定します。Base64 テキストとして以前にエンコードされているバイナリデータをデコードするには、charset-encode 関数を使用します。



言語

Built-in。

パラメーター

名前	型	説明
binary-data	`xs:base64Binary`	Base64 テキストとしてのバイナリデータ。
encoding	`xs:string`	エンコードのために使用される文字セット（例、"utf-8"）。
error-abort	`xs:boolean`	エラーが発生した場合処理がどのように継続されるかを指定する任意の引数。有効な値: • **true** – 無効な文字の例外により処理を終了します。 • **false** – 処理を続行し、無効な文字を置換文字 ? と置き換えます。 デフォルトの値は、**true** です。

サンプル

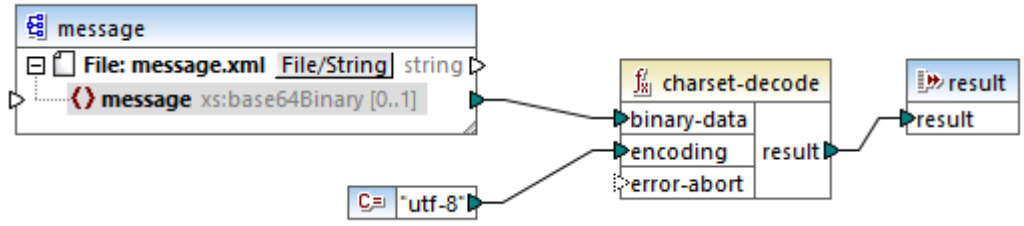
ソースXML ファイルからバイナリデータをデコードすると仮定します。**Message** 要素はBase64 テキストとしてエンコードされているバイナリデータを含むことに注意してください。

```
<?xml version="1.0" encoding="UTF-8"?>
<message xsi:noNamespaceSchemaLocation="message.xsd"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">TG9yZW0gaXBzdW0=</message>
```

スキーマで示されるとおり、**message** 要素のデータ型は`xs:base64Binary` です:

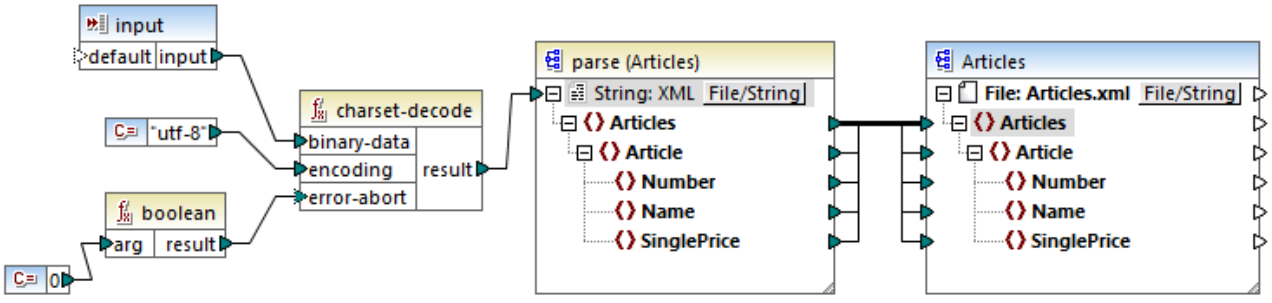
```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
   <xs:element name="message" type="xs:base64Binary"/>
</xs:schema>
```

上記のメッセージをデコードするマッピングは以下の通りです



このサンプルのマッピングはテキスト "Lorem ipsum" を出力します。

MapForce シリアル化コンポーネントを使用して、マッピングはBase64 データとしてエンコードされているテキストまたはXML ファイルを処理することもできます。例えば、下に示されるマッピングには、Base64 テキストデータを期待する **input** パラメーターが存在します。[charset-encode](#) サンプルで示されるようにXML ファイルからBase64 データが作成されていると仮定し、基のXML ファイルを下のマッピングで示されるように再作成することができます:

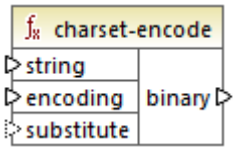


このマッピング内では、**error-abort** 引数は、`boolean`n ビルドイン関数を使用して作成された**false** の値を取得します。これにより、無効な文字が発生しても、処理が継続されることが保証されます。関数の文字列の結果は、XML ファイルに変換されるXML 解析コンポーネントにパスされます。XML 解析を可能にするために、XSD スキーマファイルが必要になります。詳細に関しては、[文字列の解析とシリアル化](#)を参照してください。

(C) 2015-2021 Altova GmbH

7.10.19.3 charset-encode

charset-encode 関数は、入力文字列データを取り、Base64 テキストとしてエンコードします。指定された文字列セット（例えば、"utf-8"）内でエンコードされたデータは、`xs:base64Binary` 型として返されます。Base64 テキストとして以前にエンコードされているバイナリデータをデコードするには、charset-decode 関数を使用します。



言語

Built-in。

パラメーター

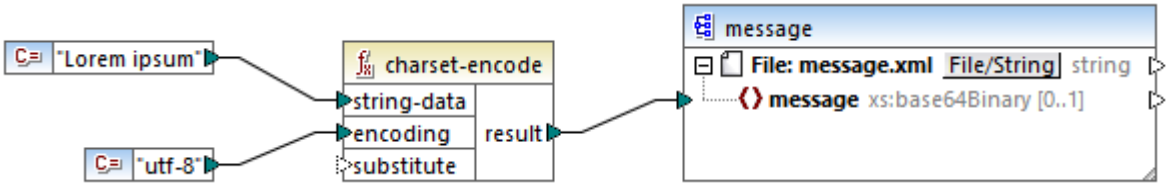
名前	型	説明
string-data	**xs:string**	エンコードされる文字列データ
encoding	**xs:string**	エンコードのために使用される文字セット（例、"utf-8"）。
substitute	**xs:string**	無効な文字が発生した場合、置換の文字を指定する任意の引数。非-Unicode エンコードを使用する場合、この引数を適用できます。Unicode エンコードのための、置換文字は です。

サンプル

UTF-8 文字セットを使用して、テキスト "Lorem ipsum" を Base64 データをエンコードしターゲット XML ファイルに書き込むと仮定します。ターゲット XML ファイルには、`xs:base64Binary` 型の**message** 要素がスキーマにより示されているように存在します:

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
   <xs:element name="message" type="xs:base64Binary"/>
</xs:schema>
```

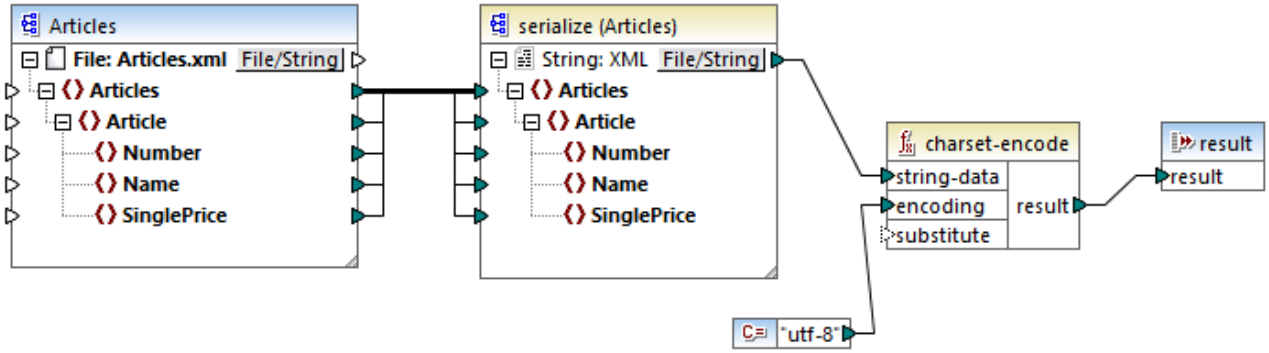
Base64 エンコードを実行するマッピングは以下の通りです:



このマッピングは下のような XML 出力を生成します（スキーマ参照とXML 宣言はスキップされています）:

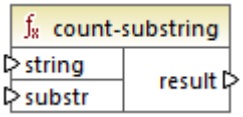
```
<message>TG9yZW0gaXBzdW0=</message>
```

MapForce シリアル化コンポーネントを使用して、serializationテキストまたはXML ファイルを Base64 としてエンコードすることができます。例、下で示されるマッピングは、ソースXML ファイルを文字列にシリアル化します。結果する文字列は`charset-encode` 関数の引数として提供されます。最後に、関数の結果は、単純型出力コンポーネントを使用してマッピングの出力として返されます。マッピングからの文字列の値を返すを参照してください。シリアル化の情報に関しては、文字列の解析とシリアル化を参照してください。



7.10.19.4 count-substring

string 内で発生する**substr** の回数を表した整数値が返されます。



言語

Built-in、C++、C#、Java。

パラメーター

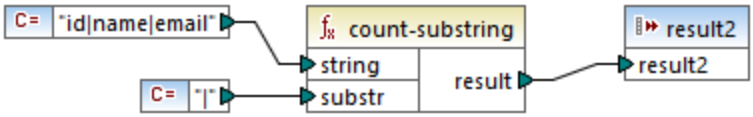
名前	型	説明
string	`xs:string`	入力文字列。

(C) 2015-2021 Altova GmbH

名前	型	説明
substr	`xs:string`	テストするサブ文字列。

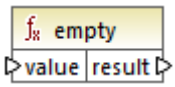
サンプル

次のマッピングは**2** を返します。これは入力文字列 `id|name|email` 内で発生するパイプセパレーターの発生する回数です。



7.10.19.5 empty

入力文字列が空の場合 **true** を返します。それ以外の場合は**false** を返します。



言語

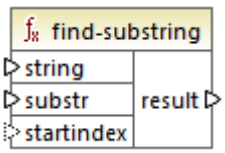
Built-in、C++、C#、Java。

パラメーター

名前	型	説明
value	`xs:string`	入力値。

7.10.19.6 find-substring

string 内の**substr** の最初の発生の位置を返します。デフォルトではポジション（インデックス）1 を持ち、最初の文字から検索を開始する関数は最初の文字列からの検索を無視しますが、開始インデックスを任意で指定することができます。**substr** が見つからない場合関数は**0** を返します。



言語

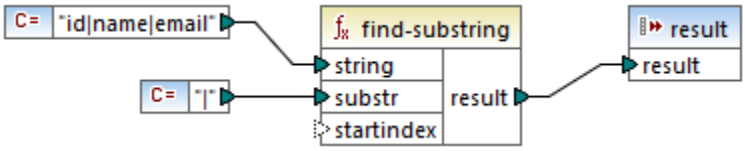
Built-in、C++、C#、Java。

パラメーター

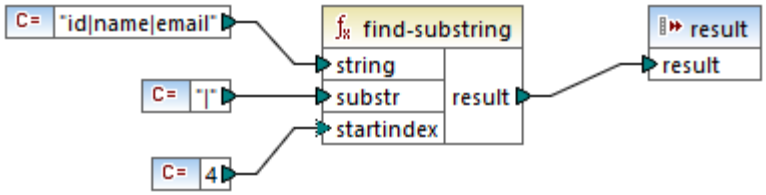
名前	型	説明
string	`xs:string`	入力文字列。
substr	`xs:string`	検索するサブ文字列。
startindex	`xs:int`	任意。検索の開始ポジション（インデックス）を指定します。パラメーターが指定されていない場合、検索はポジション1で開始されます。

サンプル

次のマッピングは入力文字列 `id|name|email` 内のパイプ文字の最初の発生のポジションである **3** を生成します。

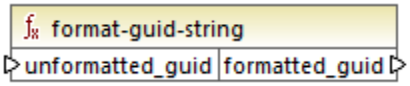


4 を開始インデックスと指定する場合、関数の検索は4番目の文字から検索します。この結果、下のマッピングは4番目の文字以降からの検索したパイプ文字の最初の発生である **8** を出力します。



7.10.19.7 format-guid-string

通常データベースのフィールドで使用するための、正しくフォーマットされたグローバル一意識別子（GUID）がformatted_guid から返されます。create-guid 関数の項目も参照ください。



言語

Built-in、C++、C#、Java。

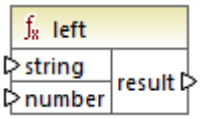
(C) 2015-2021 Altova GmbH

パラメーター

名前	型	説明
unformatted_guid	`xs:string`	入力16進エンコード文字列のみがフォーマットされます。

7.10.19.8 left

入力文字列の最初の**number** 文字を含む文字列を返します。



言語

Built-in、C++、C#、Java。

パラメーター

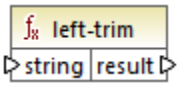
名前	型	説明
string	`xs:string`	入力文字列。
number	`xs:int`	文字列先頭から開始して返される文字列の数値を指定します。

サンプル

入力文字列が`This is a sentence` で、数値が**4** の場合、関数は`This` を返します。

7.10.19.9 left-trim

全ての先頭空白文字が削除されている入力文字列を返します。



言語

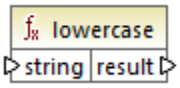
Built-in、C++、C#、Java。

パラメーター

名前	型	説明
string	`xs:string`	入力文字列。

7.10.19.10 lowercase

入力 **string** を小文字に変換します。Unicode の文字の場合、(Unicode コンソーシアムにより定義された) 対応する小文字が使用されます。



言語

Built-in、C++、C#、Java。

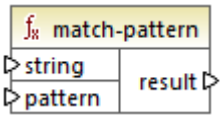
パラメーター

名前	型	説明
string	`xs:string`	入力文字列。

7.10.19.11 match-pattern

pattern により定義された正規表現が入力文字列に一致する場合、ブール値 **true** を返し、それ以外の場合には **false** を返します。正規表現も参照してください。

メモ C++、C#、または Java コードを生成する際には、正規表現構文の高度な機能が異なる可能性があります。詳細に関しては各言語の正規表現ドキュメントを確認してください。



言語

Built-in、C++、C#、Java。

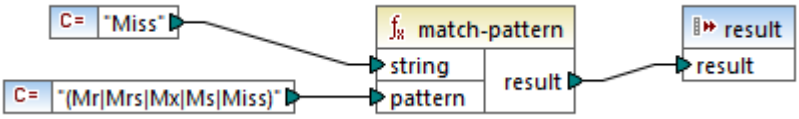
(C) 2015-2021 Altova GmbH

パラメーター

名前	型	説明
string	xs:string	入力文字列。
pattern	xs:string	一致する正規表現。

サンプル

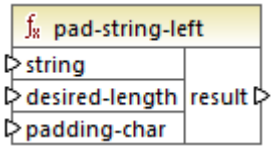
以下のマッピングは多種の個人のタイトルを検証します。具体的にはマッピングは以下のタイトルに対して **true** を出力する必要があります: Mr、Mrs、Mx、Ms、Miss。



入力文字列が上記以外のタイトルの場合、マッピングは **false** を返します。

7.10.19.12 pad-string-left

単一の特定文字により左にパドされた文字列を返します。希望する文字列の長さとパディング文字は引数として与えられます。



言語

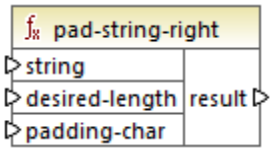
Built-in、C++、C#、Java。

パラメーター

名前	型	説明
string	xs:string	入力文字列を指定する。
desired-length	xs:int	パディングの後の文字列の希望する長さを定義する。
padding-char	xs:string	パディング文字として使用する文字を定義する。

7.10.19.13 pad-string-right

単一の特定文字により右にパドされた文字列を返します。希望する文字列の長さとパディング文字は引数として与えられます。



言語

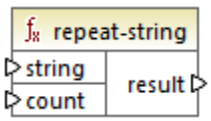
Built-in、C++、C#、Java。

パラメーター

名前	型	説明
string	`xs:string`	入力文字列を指定する。
desired-length	`xs:int`	パディングの後の文字列の希望する長さを定義する。
padding-char	`xs:string`	パディング文字として使用する文字を定義する。

7.10.19.14 repeat-string

引数 *n* 回与えられた文字列を繰り返します。**count** 引数は、文字列の繰り返しの回数を指定します。



言語

Built-in、C++、C#、Java。

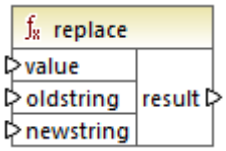
パラメーター

名前	型	説明
string	`xs:string`	入力文字列。
count	`xs:int`	文字列を繰り返す回数。

(C) 2015-2021 Altova GmbH

7.10.19.15 replace

value により得られた入力文字列のうち、**oldstring** の文字列に一致する各文字列が**newstring** の文字列に置き換えられます。



言語

Built-in、C++、C#、Java。

パラメーター

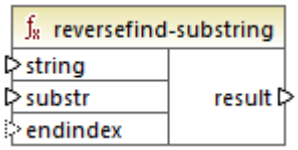
名前	型	説明
value	`xs:string`	入力値。
oldstring	`xs:string`	置き換えられる古い文字列。
newstring	`xs:string`	置換としての役割を果たす新規の文字列。

サンプル

特殊文字の置換を参照してください。

7.10.19.16 reversefind-substring

string 内の**substr** の最後の発生の位置を返します。デフォルトではポジション（インデックス）1 を持ち、最後の文字で検索を終了する関数は最初の文字列からの検索を無視しますが、終了インデックスを任意で指定することができます。**substr** が見つからない場合関数は **0** を返します。



言語

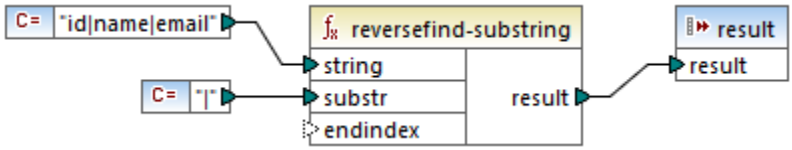
Built-in、C++、C#、Java。

パラメーター

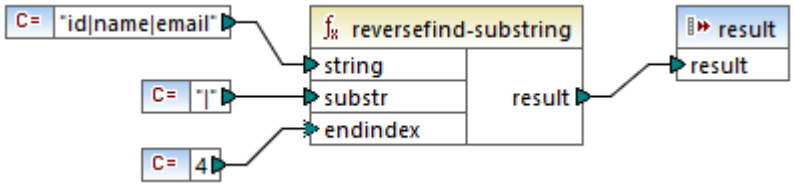
名前	型	説明
string	`xs:string`	入力文字列。
substr	`xs:string`	検索するサブ文字列。
endindex	`xs:int`	任意。検索の終了ポジション（インデックス）を指定します。パラメーターが指定されていない場合、検索は**string** 内の最後の文字で終了します。

サンプル

次のマッピングは入力文字列 `id|name|email` 内のパイプ文字の最後の発生のポジションである **8** を生成します。

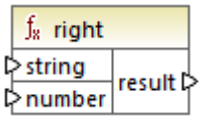


4 を終了インデックスと指定する場合、関数の検索は4番目の文字まで検索します。この結果、下のマッピングは **3** を出力します。



7.10.19.17 right

入力文字列の最後の**number** 文字を含む文字列を返します。



言語

Built-in、C++、C#、Java。

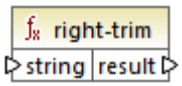
(C) 2015-2021 Altova GmbH

サンプル

入力文字列が`The brown red fox` で、数値が**3** の場合、関数は`fox`を返します。

7.10.19.18 right-trim

全ての行末空白文字が削除されている入力文字列を返します。



言語

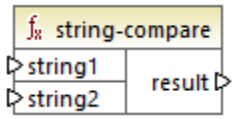
Built-in、C++、C#、Java。

パラメーター

名前	型	説明
string	`xs:string`	入力文字列。

7.10.19.19 string-compare

大文字と小文字を考慮して**string1** と**string2** の文字列の比較の結果を返します。両方の文字列が等価の場合 **0** が返されます。**string1** が**string2** より小さい場合 **-1** が返されます。**string1** が**string2** より大きい場合 **1** が返されます。



言語

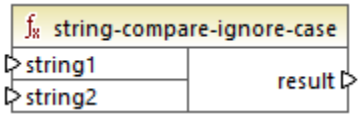
Built-in、C++、C#、Java。

パラメーター

名前	型	説明
string1	`xs:string`	最初の入力文字列。
string2	`xs:string`	2番目の入力文字列。

7.10.19.20 string-compare-ignore-case

大文字と小文字を無視した**string1** と**string2** の比較文字列の結果が返されます。両方の文字列が等価の場合 **0** が返されます。**string1** が**string2** より小さい場合 **-1** が返されます。**string1** が**string2** より大きい場合 **1** が返されます。



言語

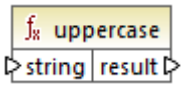
Built-in、C++、C#、Java。

パラメーター

名前	型	説明
string1	`xs:string`	最初の入力文字列。
string2	`xs:string`	2番目の入力文字列。

7.10.19.21 uppercase

入力 **string** を大文字に変換します。Unicode の文字の場合、(Unicode コンソーシアムにより定義された) 対応する大文字が使用されます。



言語

Built-in、C++、C#、Java。

パラメーター

名前	型	説明
string	`xs:string`	入力文字列。

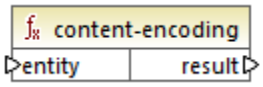
7.10.20 mime

mime ライブラリには、MapForce にMIME (Multipurpose Internet Mail Extensions) コンテンツとインタラクトする関数を含んでいます。(バイナリファイルなどの) カスタムMIME コンテンツを送信するW eb サービスの呼び出しを作成するためには通常 **mime** 関数が必要になります。**mime** 関数を使用してW eb サービスコールから返されたレスポンスMIME エントリを処理することができます。

(C) 2015-2021 Altova GmbH

7.10.20.1 content-encoding

`content-encoding` 関数は、引数として与えれているMIME エンティティの文字エンコードを返します。HTTP メッセージの場合、これは**Content-Type** ヘッダーからの**charset** パラメーターの値です。



言語

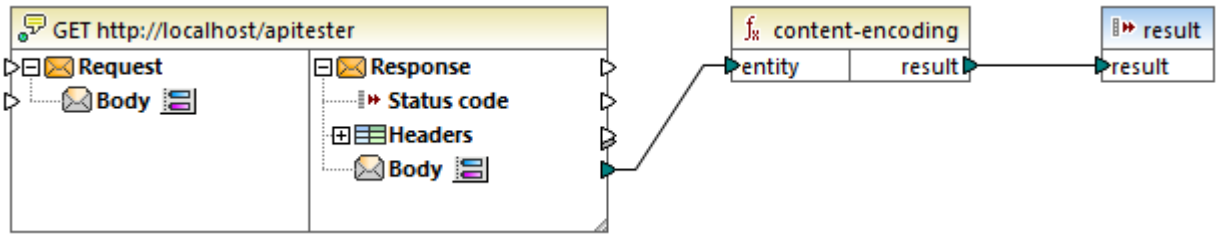
Built-in。

パラメーター

名前	型	説明
entity	`mf:MimeEntity`	MIME エンティティコンテンツ

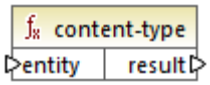
サンプル

マッピングが**Content-Type** ヘッダーを `text/json; charset=utf-8` として返すW eb サービスを呼び出す場合、関数出力も `utf-8` です。



7.10.20.2 content-type

`content-type` 関数は引数として与えらえているMIME エンティティのメディア型（**Content-Type**）を返します。



言語

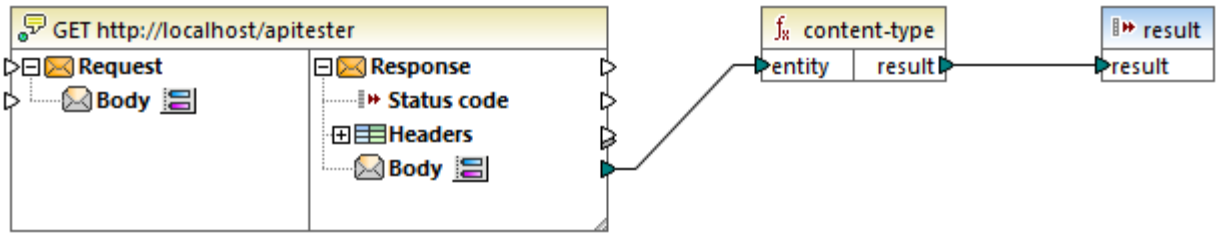
Built-in。

パラメーター

名前	型	説明
entity	`mf:MimeEntity`	MIME エンティティコンテンツ

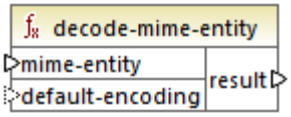
サンプル

マッピングが**Content-Type** ヘッダーを `text/json; charset=utf-8` として返す W eb サービスを呼び出す場合、関数出力も `text/json; charset=utf-8` です。



7.10.20.3 decode-mime-entity

`decode-mime-entity` 関数は**Content-Type** ヘッダー内に含まれている**charset** パラメーターを使用してデコードされたMIME エンティティを示す文字列を返します。**charset** が指定されていない場合、希望する文字列の値を**default-encoding** パラメータにつなぎ、デフォルトのエンコードを提供することができます。



言語

Built-in。

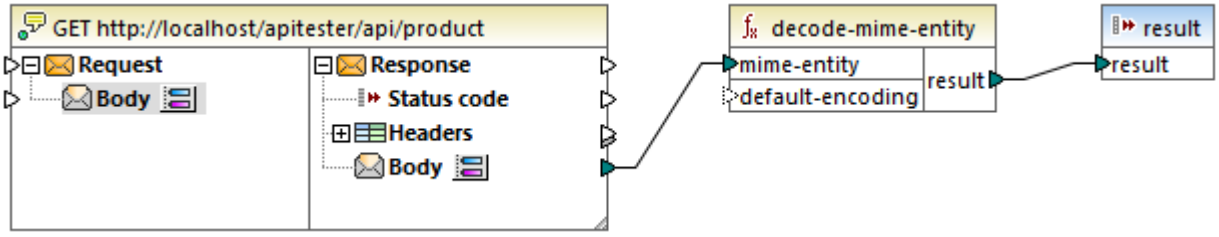
パラメーター

名前	型	説明
mime-entity	`mf:MimeEntity`	デコードするMIME エンティティ
default-encoding	`xs:string`	**charset** パラメーター内でエンコードが指定されていない場合デフォルトのエンコードが使用されます。

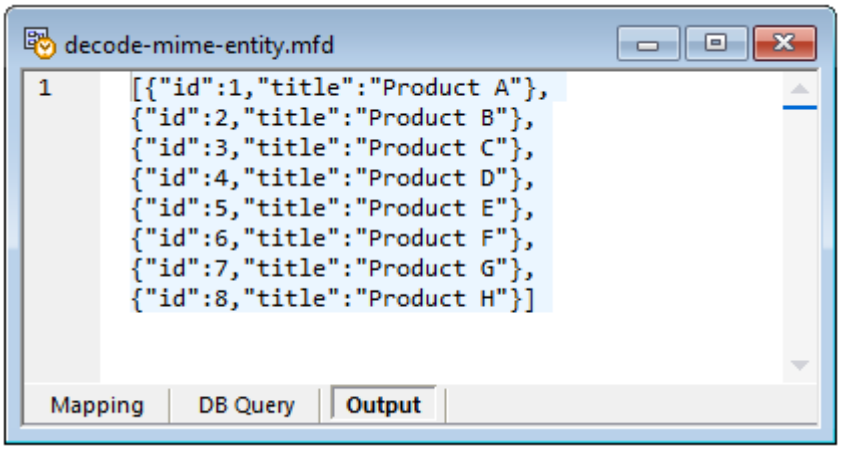
(C) 2015-2021 Altova GmbH

サンプル

次のマッピングはJSON コンテンツを返すW eb サービスを呼び出します。

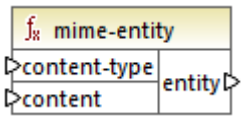


このマッピングが作動すると、W eb サービスの呼び出しの結果は`decode-mime-entity` 関数の入力として提供されます。**Content-Type** ヘッダーの**charset** パラメーターに従って呼び出しにより返された生ボディを関数はデコードします。結果は以下のとおりです: 結果は以下のようになります:



7.10.20.4 mime-entity

`mime-entity` 関数はMIME エンティティを返します。MIME エンティティは**content** 引数に与えられたコンテンツから作成され、**type** 引数に与えられたメディア型を持っています。



言語

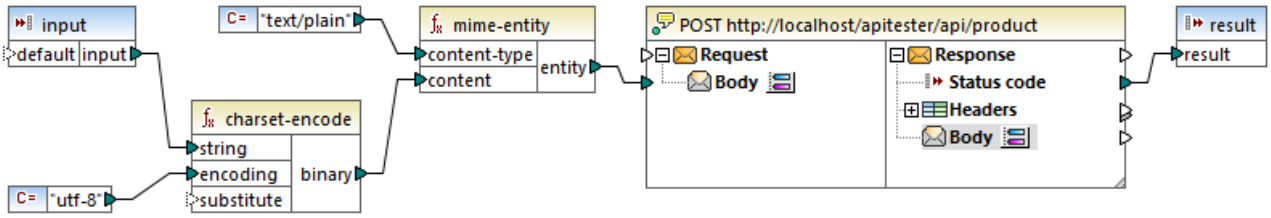
Built-in。

パラメーター

名前	型	説明
content-type	`mf:MediaType`	このMIME エンティティを説明するメディア型。例: `image/png`。
content	`xs:base64Binary`	Base64 エンコードスキームを使用してエンコードされた実際のデータ。charset-encode 関数を使用してこのようなデータをエンコードすることができます。

サンプル

次のマッピングはPOST リクエストにより提出されている構造化されていないテキストを期待します。W eb サービスを呼び出します。

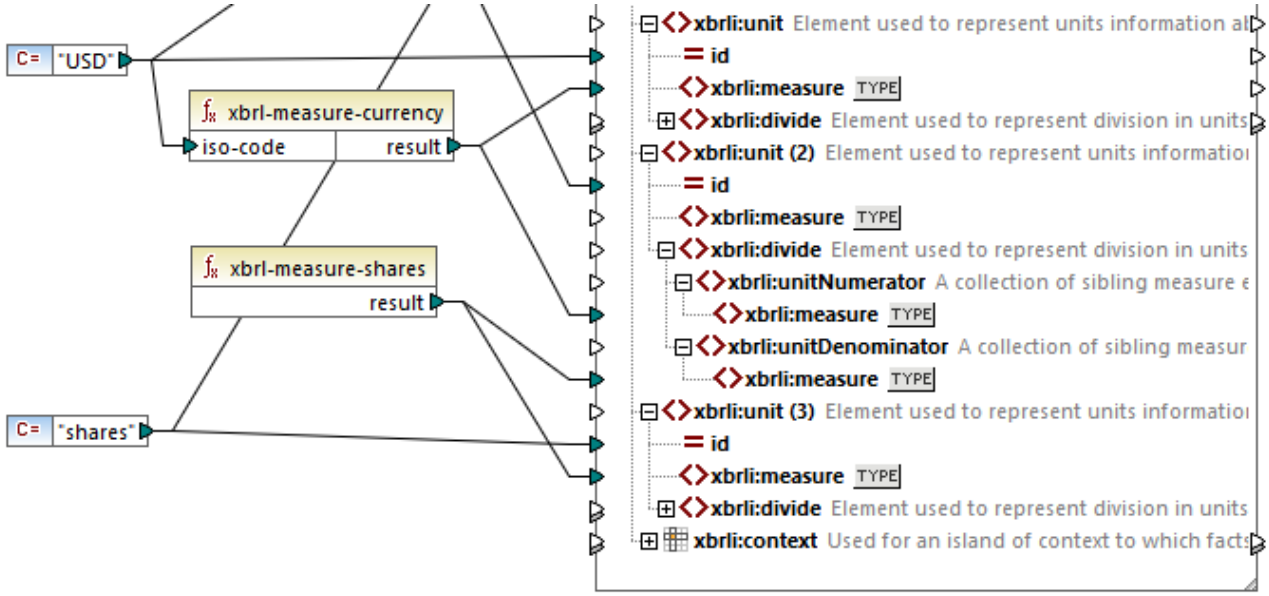


上記のマッピングでは、入力コンポーネントを使用して、入力テキストがマッピングへのパラメーターとして与えられています。マッピングにパラメーターを提供するを参照してください。charset-encode 関数を使用して **utf-8** エンコードを使用し、入力として受け取った文字列はバイナリ書式に変換されます。`mime-entity` 関数はバイナリコンテンツを引数としてとり、サービスの呼び出しの **Body** 入力にマップされるリクエスト（ MIME エンティティ）のボディを返します。

7.10.21 xbrl

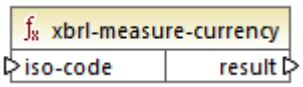
XBRL ライブラリには、XBRL インスタンス、またはタクソノミファイルに必要な QName フォーマットへ、データを変換するための関数が用意されています。

xbrl:unit 要素の**xbrl:ID** と**xbrl:measure** 子要素はXBRL インスタンスファイルで必須の要素となっており、妥当なファイルを作成するためにはマッピングを行う必要があります。



7.10.21.1 xbrl-measure-currency

ISO 4217 通貨コードから`monetaryItemType` のアイテムに対して QName を返します。



言語

Built-in、C++、C#、Java。

パラメーター

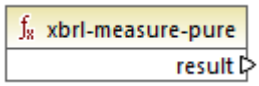
名前	型	説明
iso-code	`xs:string`	ISO 4217 通貨コード。

サンプル

Microsoft Excel から XBRL へマッピング を参照してください。

7.10.21.2 xbrl-measure-pure

レート、パーセンテージ、または比率に対して QName が返されます。

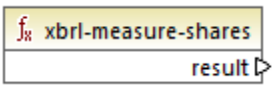


言語

Built-in、C++、C#、Java。

7.10.21.3 xbrl-measure-shares

`sharesItemType` のアイテムに対して QName が返されます。



言語

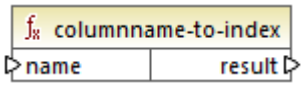
Built-in、C++、C#、Java。

7.10.22 xlsx

xlsx ライブラリには Excel の日付/時間フォーマットとのデータ変換を行うための関数が用意されています。**date-to-xlsx**、**datetime-to-xlsx**、および **time-to-xlsx** 関数はターゲット Excel シートに日付と時刻の値を書き込むために通常使用されます。**xlsx-to-date**、**xlsx-to-datetime**、および **xslx-to-datetime** 関数はソース Excel シートから日付と時刻の値を 型 `xs:date`、`xs:time`、および `xs:dateTime` の値に変換する際に通常役に立ちます。

7.10.22.1 columnname-to-index

与えられた名前に対するカラムのインデックスが返されます。カラム [A] のインデックスは 1 です。



言語

Built-in、C#、Java、XSLT 2.0、XSLT 3.0。

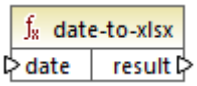
(C) 2015-2021 Altova GmbH

パラメーター

名前	型	説明
name	`xs:string`	カラム名は[A]、[B]、[C] などから開始されExcel で表示されるとおりです。

7.10.22.2 date-to-xlsx

ソースから抽出された *date* 値のExcel 表現を日付を表すシリアル番号として返します。デフォルトではのためのMicrosoft Excel では1900ねん1 月1 日のシリアル番号は**1** です。



言語

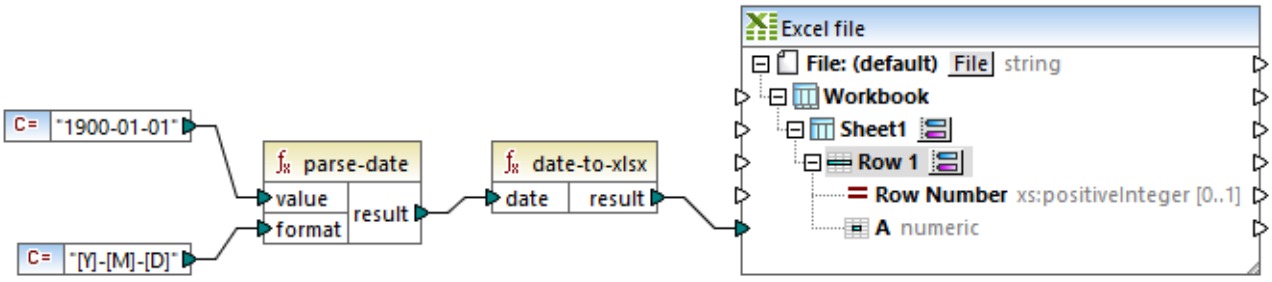
Built-in、C#、Java、XSLT 2.0、XSLT 3.0。

パラメーター

名前	型	説明
date	`xs:date`	入力日付の値。

サンプル

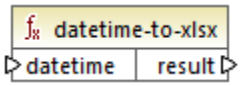
以下のマッピングは`parse-date` を使用して文字列 "1900-01-01" を日付に変換します。取得された`xs:date` 値（1900年1 月1 日）は数値表現を返しExcel セルに書き込む`date-to-xlsx` 関数に引数として提供されます。結果するセルの値は期待通り1 です。



ターゲットExcel セルの書式は[数値]であることご注意して下さい。Excel 内のセルを右クリックし、セルのフォーマット コマンドを使用して日付に書式設定すると、セルの値は`1/1/1900` になります。

7.10.22.3 datetime-to-xslx

ソースから抽出された*date and time* 値のExcel 表示を返します。



言語

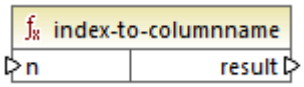
Built-in、C#、Java、XSLT 2.0、XSLT 3.0。

パラメーター

名前	型	説明
datetime	**xs:dateTime**	型 xs:dateTime の入力値。

7.10.22.4 index-to-columnname

(例えば、[A]、[B]、[C])カラムインデックスがある(1、2、3 などの) Excel 列の名前を返します 。カラムの名前は[A] です。



言語

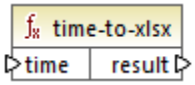
Built-in、C#、Java、XSLT 2.0、XSLT 3.0。

パラメーター

名前	型	説明
n	**xs:int**	カラムの1-ベースのインデックス。

7.10.22.5 time-to-xlsx

ソースから抽出された*time* 値のExcel 表示を返します。



(C) 2015-2021 Altova GmbH

言語

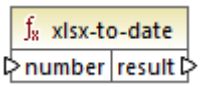
Built-in、C#、Java、XSLT 2.0、XSLT 3.0。

パラメーター

名前	型	説明
time	**xs:time**	型 xs:time の入力値。

7.10.22.6 xlsx-to-date

Excel 内のデータの数値表現された xs:date 値を返します。



言語

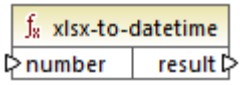
Built-in、C#、Java、XSLT 2.0、XSLT 3.0。

パラメーター

名前	型	説明
number	**xs:double**	Excel 内の日付のシリアル番号です

7.10.22.7 xlsx-to-datetime

Excel 内の日時の値の数値表現された xs:dateTime 値を返します。



言語

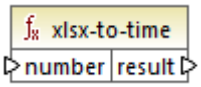
Built-in、C#、Java、XSLT 2.0、XSLT 3.0。

パラメーター

名前	型	説明
number	**xs:double**	Excel 内の日時表現です

7.10.22.8 xlsx-to-time

Excel 内の時刻の値の数値表現された `xs:time` 値を返します。



言語

Built-in、C#、Java、XSLT 2.0、XSLT 3.0。

パラメーター

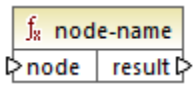
名前	型	説明
number	`xs:double`	Excel 内の時刻表現です

7.10.23 xpath2 | accessors

xpath2 | accessors サブライブラリからの関数は XML ノードまたはアイテムに関する情報を抽出します。これらは関数は XSLT2 または XQuery 言語が選択されている状態で利用することができます。

7.10.23.1 base-uri

`base-uri` 関数はノードを入力として取り、ノードを含む XML リソースの URI を返します。出力は型 `xs:string` 型になります。



言語

XQuery、XSLT 2.0、XSLT 3.0。

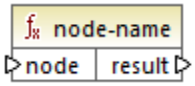
パラメーター

名前	型	説明
node	`mf:node`	入力ノード。

(C) 2015-2021 Altova GmbH

7.10.23.2 node-name

node-name 関数は入力パラメーターとしてノードを受け取り、そのQName を返します。QName が文字列として表示され、ノードにプレフィックスがある場合は `prefix:localname` 形式の入力を受け取り、ノードにプレフィックスが無い場合は `localname` 形式の入力が受け取られます。ノードの名前空間 URI を取得するには、(qnameに関連したfunctions 以下にある) **namespace-uri-from-QName** 関数を使用してください。



言語

XQuery、XSLT 2.0、XSLT 3.0。

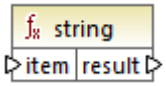
パラメーター

名前	型	説明
node	**mf:node**	入力ノード。

7.10.23.3 string

string 関数は、`xs:string` のように作動し、入力された値を `xs:string` 型に変換します。

入力が(例えば `xs:decimal` の)原子型である場合、原子型の値が `xs:string` 型の値に変換されます。入力引数がノードの場合、ノードの文字列値が抽出されます。(ノードの文字列値とはノードの子孫ノードから得られた値を連結したものになります)。



言語

XQuery、XSLT 2.0、XSLT 3.0。

パラメーター

名前	型	説明
item	**mf:item**	入力値。

7.10.24 xpath2 | anyURI functions (anyURI 関数)

xpath2 | anyURI サブライブラリは`resolve-uri` 関数を含んでいます。この関数はXSLT2 またはXQuery 言語が選択されている状態で利用することができます。

7.10.24.1 resolve-uri

`resolve-uri` 関数は最初のパラメーターとして相対的なURI を取り、2 番目の引数内のベースURI に対して解決します。結果は型`xs:string` になります。関数の実装は両方の入力を文字列として扱います。これらのURI により識別されるリソースが実在するかチェックされません。

言語

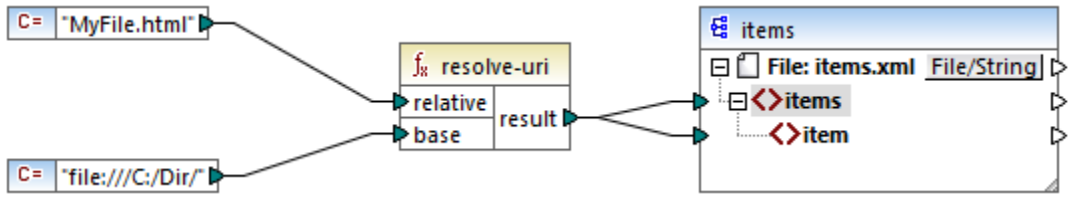
XQuery、XSLT 2.0、XSLT 3.0。

パラメーター

名前	型	説明
relative	`xs:string`	ベースに対して解決される相対的なURI。
base	`xs:string`	ベースURI。

サンプル

下で表示されているマッピング内では、最初の引数は相対的なURI `MyFile.html` を提供しており、2番目の引数はベースURI `file:///C:/Dir/` を提供しています。解決されたURI は2つを連結したもので`file:///C:/Dir/MyFile.html` になります。



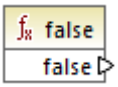
7.10.25 xpath2 | boolean functions (ブール値関数)

boolean 関数 `true` ならびに`false` 関数は引数を受け取ること無く、boolean の定数値である**true** と**false** をそれぞれ返します。定数のboolean 値が必要な場所で使用することができます。

(C) 2015-2021 Altova GmbH

7.10.25.1 false

boolean 値の**false** を挿入します。

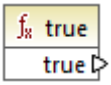


言語

XQuery、XSLT 2.0、XSLT 3.0。

7.10.25.2 true

boolean 値の**true** を挿入します。



言語

XQuery、XSLT 2.0、XSLT 3.0。

7.10.26 xpath2 | constructors

XPath 2.0 ライブラリの[constructors] サブライブラリ内の関数は入力テキストから特定のデータ型をコンストラクトします。以下のテーブルは使用可能なすべてのコンストラクタ関数をリストしています。

xs:ENTITY	xs:double	xs:nonPositiveInteger
xs:ID	xs:duration	xs:normalizedString
xs:IDREF	xs:float	xs:positiveInteger
xs:NCName	xs:gDay	xs:short
xs:NMTOKEN	xs:gMonth	xs:string
xs:Name	xs:gMonthDay	xs:time
xs:QName	xs:gYear	xs:token
xs:anyURI	xs:gYearMonth	xs:unsignedByte
xs:base64Binary	xs:hexBinary	xs:unsignedInt
xs:boolean	xs:int	xs:unsignedLong

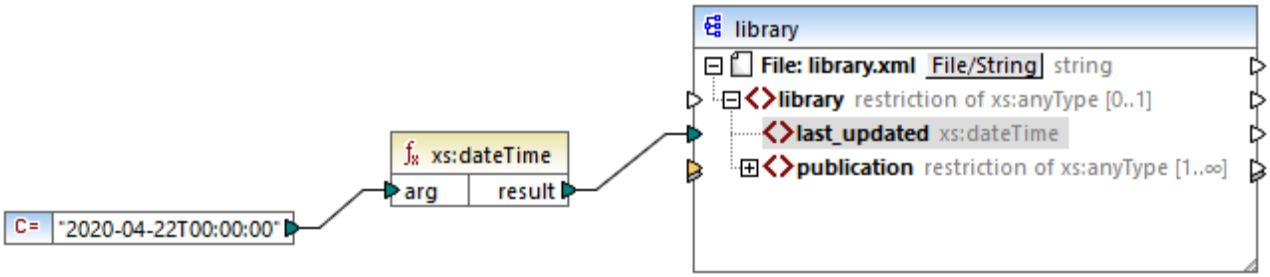
xs:byte	xs:integer	xs:unsignedShort
xs:date	xs:language	xs:untypedAtomic
xs:dateTime	xs:long	xs:yearMonthDuration
xs:dayTimeDuration	xs:negativeInteger	
xs:decimal	xs:nonNegativeInteger	

言語

XQuery、XSLT 2.0、XSLT 3.0。

サンプル

通常、入力されたテキストはコンストラクトされるデータ型に対応した書式で記述されている必要があります。そうでない場合は変換に失敗します。例えば、 xs:dateTime 値を **xs:dateTime** コンストラクタ関数を使用してコンストラクトする場合、入力テキストは xs:dateTime データ型の書式である **YYYY-MM-DDTHH:mm:ss** の形式で記述されている必要があります。



上で説明されているマッピング内では（**"2020-04-28T00:00:00"**）は関数の入力引数を提供するために使用されています。入力はソースドキュメント内のアイテムから取得することもできます。 **xs:dateTime** 関数は型 xs:dateTime の値 **2020-04-28T00:00:00** を返します。

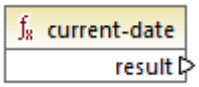
（関数引数のデータ型を含む）マッピングアイテムの期待されるデータ型を確認するには マウスカーソルを対応する入力または出力コネクタにポイントします。

7.10.27 xpath2 | context functions（コンテキスト関数）

xpath2 ライブラリからのコンテキスト関数は現在の日時のその他の情報、以前に使用されたデフォルトの照合順序、現在のシーケンスのサイズ、現在のノードのポジションを提供します。

7.10.27.1 current-date

システムクロックから得られた現在の日付（`xs:date`）を返します。

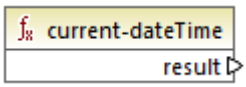


言語

XQuery、XSLT 2.0、XSLT 3.0。

7.10.27.2 current-dateTime

システムクロックから得られた現在の日時（`xs:dateTime`）を返します。

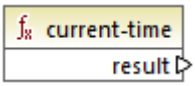


言語

XQuery、XSLT 2.0、XSLT 3.0。

7.10.27.3 current-time

システムクロックから得られた現在の時刻（`xs:time`）を返します。



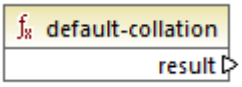
言語

XQuery、XSLT 2.0、XSLT 3.0。

7.10.27.4 default-collation

`default-collation` 関数は入力パラメーターを取らず、デフォルトの照合、つまり指定することができる関数に対して特定の照合が指定されなかった場合に使用される照合を返します。

`max-string` と**`min-string`** 関数を含む比較はデフォルトの照会順序をベースにしています。

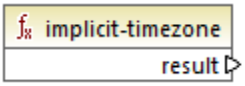


言語

XQuery、XSLT 2.0、XSLT 3.0。

7.10.27.5 implicit-timezone

評価コンテキストからの[implicit timezone] プロパティの値を返します。

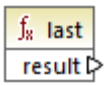


言語

XQuery、XSLT 2.0、XSLT 3.0。

7.10.27.6 last

現在処理されているアイテムのシーケンス内のアイテムの数量を返します。下のサンプルで説明される通り、重要な点はアイテムのシーケンスは現在の[マッピングコンテキスト]() により決定されます。



言語

XQuery、XSLT 2.0、XSLT 3.0。

サンプル

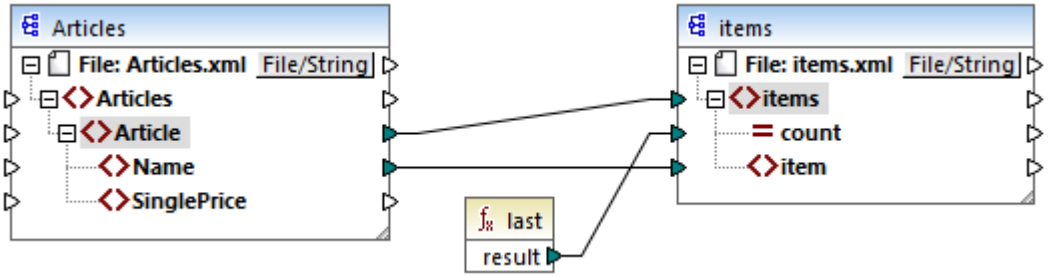
次のソースXML ファイルが存在すると仮定します:

```
<Articles>
   <Article>
      <Name>T-Shirt</Name>
      <SinglePrice>25</SinglePrice>
   </Article>
   <Article>
      <Name>Socks</Name>
      <SinglePrice>2.30</SinglePrice>
   </Article>
```


(C) 2015-2021 Altova GmbH

```
   <Article>
      <Name>Jacket</Name>
      <SinglePrice>57.50</SinglePrice>
   </Article>
</Articles>
```

目的はデータを異なるスキーマを持つXML ファイルにコピーすることです。また、全てのアイテムのカウントはターゲット XML ファイルに保存される必要があります。これは以下のようなマッピング内で達成することができます:



上のサンプルでは`last` 関数は現在の親コンテキスト内と最後のノードの位置を返し、値 **3** を持つ**count** 属性を生成します。

```
<items count="3">
   <item>T-Shirt</item>
   <item>Pants</item>
   <item>Jacket</item>
</items>
```

値 **3** は**Article** と**items** の接続により作成されたマッピングコンテキスト内の最後のアイテム(すなわちすべてのアイテムのカウント)のポジションです。この接続が存在しない場合でもアイテムはターゲットにコピーされますが反復する 親コンテキスト が存在しないため`last` 関数は値 **1** を間違って返します。(具体的には、両方のコンポーネントのルートアイテム間で作成され、期待される通り3個ではなく1個のアイテムのシーケンスを作成する)デフォルトの明示的なコンテキストを使用します。

一般的には`last` 関数の代わりに**core** からの`count` 関数を使用することが奨励されます。これは前者がマッピングコンテキストを明示的に変更することのできる**parent-context** 引数を持っているからです。

7.10.28 xpath2 | durations, date and time functions (期間、日付、および時刻関数)

xpath2 ライブラリからの期間、日付、および時刻関数により日付と時刻の値内でタイムゾーンを調整することができ、特定のコンポーネントを日付、時刻、および期間値から抽出し、日付時刻の値を減算します。

タイムゾーンの調整

日付および時刻の値内でタイムゾーンを調節するには、以下の関数を使用することができます:

- `adjust-date-to-timezone`
- `adjust-date-to-timezone` (タイムゾーン引数を持つ)
- `adjust-dateTime-to-timezone`
- `adjust-dateTime-to-timezone` (タイムゾーン引数を持つ)
- `adjust-time-to-timezone`
- `adjust-time-to-timezone` (タイムゾーン引数を持つ)

これら関数はそれぞれ xs:date、xs:time、または xs:dateTime 値を最初の引数として取り、(存在する場合)2 番目のパラメーターから得られた値により、タイムゾーンの削除や編集を行います。

次のシチュエーションは最初の引数にタイムゾーンが含まれない場合のみ可能です(例えば、日付 2020-01 または時刻 14:00:00)。

- **timezone** 引数が存在する場合、結果には2番目の引数により指定されたタイムゾーンが含まれています。2番目の引数内のタイムゾーンが追加されます。
- **timezone** 引数が不在の場合、結果にはシステムのタイムゾーンである明示的なタイムゾーンが含まれます。システムのタイムゾーンが追加されます。
- **timezone** 引数が不在の場合、結果にはタイムゾーンは含まれません。

次のシチュエーションは最初の引数にタイムゾーンが含まれない場合のみ可能です(例えば、日付 2020-01-01+01:00 または時刻 14:00:00+01:00)。

- **timezone** 引数が存在する場合、結果には2番目の引数により指定されたタイムゾーンが含まれています。オリジナルのタイムゾーンが2番目の引数のタイムゾーンに置き換えられます。
- **timezone** 引数が不在の場合、結果にはシステムのタイムゾーンである明示的なタイムゾーンが含まれます。オリジナルのタイムゾーンがシステムのタイムゾーンに置き換えられます。
- **timezone** 引数が不在の場合、結果にはタイムゾーンは含まれません。

日付と時刻のコンポーネントの抽出

時刻、分数、日数、月数などの数値を日付と時刻の値から抽出するには、以下の関数を使用することができます:

- **day-from-date**
- **day-from-dateTime**
- **hours-from-dateTime**
- **hours-from-time**
- **minutes-from-dateTime**
- **minutes-from-time**
- **month-from-date**
- **month-from-dateTime**
- **seconds-from-dateTime**
- **seconds-from-time**
- **timezone-from-date**
- **timezone-from-dateTime**
- **timezone-from-time**
- **year-from-date**
- **year-from-dateTime**

これらの関数はそれぞれ xs:date、xs:time、xs:dateTime、および xs:duration 値から特定のコンポーネントを抽出します。結果は xs:integer または xs:decimal です。

期間からコンポーネントを抽出する

時刻コンポーネントを期間から抽出するには、以下の関数を使用することができます:

- **days-from-duration**
- **hours-from-duration**
- **minutes-from-duration**
- **months-from-duration**
- **seconds-from-duration**

(C) 2015-2021 Altova GmbH

- **years-from-duration**

期間は(年数と月数を抽出するための) `xs:yearMonthDuration` または(日数、時間数、分数、秒数を抽出するための) `xs:dayTimeDuration` として指定される必要があります。`xs:decimal` を返す **seconds-from-duration** 関数の例外と共に型 `xs:integer` の結果を返します。

日付と時刻の値の減算

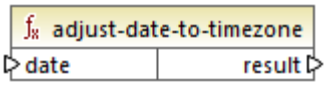
日付と時刻の値を減算するには、以下の関数を使用することができます:

- **subtract-dateTimes**
- **subtract-dates**
- **subtract-times**

それぞれの減算関数により、時間に関する値を他の値から差し引き、期間の値を得ることができます。

7.10.28.1 adjust-date-to-timezone

`xs:date` 値を評価コンテキスト(システムのタイムゾーン)内で明示的なタイムゾーンに調節します。



言語

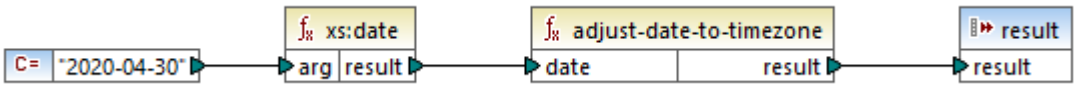
XQuery、XSLT 2.0、XSLT 3.0。

パラメーター

名前	型	説明
date	**xs:date**	型 `xs:date` の入力値。

サンプル

次のマッピングは `xs:date` から文字列をコンストラクトし **adjust-date-to-timezone** 関数の引数として提供します。

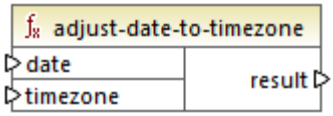


XSLT 2.0 マッピング

システムのタイムゾーンが+02:00 のコンピューターでマッピングが作動する場合、関数は日付の値がシステムのタイムゾーンを含むように調整します。この結果、マッピングの出力は `2020-04-30+02:00` になります。

7.10.28.2 adjust-date-to-timezone

`xs:date` 値を特定のタイムゾーンまたはタイムゾーン無しに調整します。**timezone** 引数が空のシーケンスの場合、関数はタイムゾーン無しで `xs:dateTime` を返します。それ以外の場合、タイムゾーン付きで `xs:date` を返します。



言語

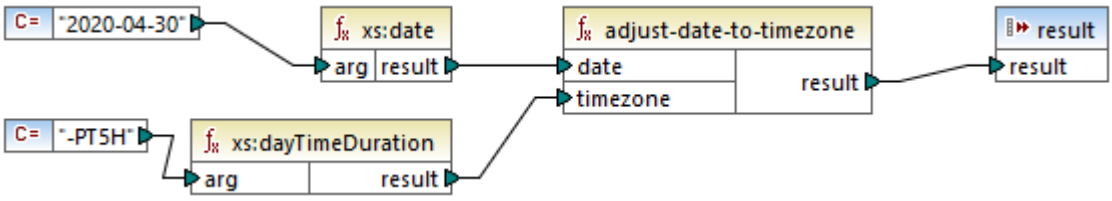
XQuery、XSLT 2.0、XSLT 3.0。

パラメーター

名前	型	説明
date	`xs:date`	型 `xs:date` の入力値。
timezone	`xs:dayTimeDuration`	タイムゾーンは `xs:dayTimeDuration` 値として表示されます。この値は負の値であることができます。例えば -5 時間のタイムゾーンの値は `-PT5H` と表記することができます。

サンプル

XPath 2 constructor 関数を使用して次のマッピングは `adjust-date-to-timezone` 関数への両方のパラメーターを文字列からコンストラクトします。このマッピングの目的はタイムゾーンを -5 時間に調整することです。このタイムゾーンは `-PT5H` と表記することができます。



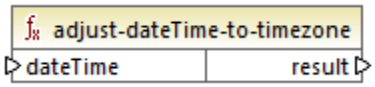
XSLT 2.0 マッピング

この関数は引数として与えられたタイムゾーンに日付の値を調整します。この結果、マッピングの出力は `2020-04-30-05:00` になります。

(C) 2015-2021 Altova GmbH

7.10.28.3 adjust-dateTime-to-timezone

xs:dateTime 値を評価コンテキスト(システムのタイムゾーン)内で明示的なタイムゾーンに調節します。



言語

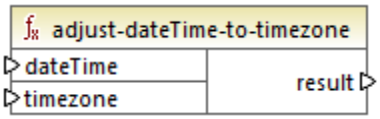
XQuery、XSLT 2.0、XSLT 3.0。

パラメーター

名前	型	説明
dateTime	**xs:dateTime**	型 xs:dateTime の入力値。

7.10.28.4 adjust-dateTime-to-timezone

xs:dateTime 値を特定のタイムゾーンまたはタイムゾーン無しに調整します。**timezone** 引数が空のシーケンスの場合、関数はタイムゾーン無しで xs:dateTime を返します。それ以外の場合、タイムゾーン付きで xs:dateTime を返します。



言語

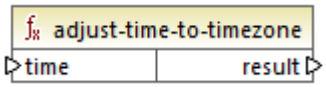
XQuery、XSLT 2.0、XSLT 3.0。

パラメーター

名前	型	説明
dateTime	**xs:dateTime**	型 xs:dateTime の入力値。
timezone	**xs:dayTimeDuration**	タイムゾーンは xs:dayTimeDuration 値として表示されます。この値は負の値であることができます。例えば -5 時間のタイムゾーンの値は **-PT5H** と表記することができます。

7.10.28.5 adjust-time-to-timezone

xs:time 値を評価コンテキスト(システムのタイムゾーン)内で明示的なタイムゾーンに調節します。



言語

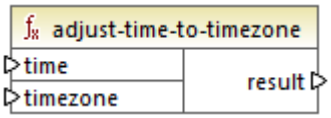
XQuery、XSLT 2.0、XSLT 3.0。

パラメーター

名前	型	説明
time	**xs:time**	型 xs:time の入力値。

7.10.28.6 adjust-time-to-timezone

xs:time 値を特定のタイムゾーンまたはタイムゾーン無しに調整します。**timezone** 引数が空のシーケンスの場合、関数はタイムゾーン無しで xs:time を返します。それ以外の場合、タイムゾーン付きで xs:time を返します。



言語

XQuery、XSLT 2.0、XSLT 3.0。

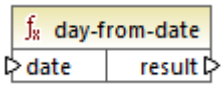
パラメーター

名前	型	説明
time	**xs:time**	型 xs:time の入力値。
timezone	**xs:dayTimeDuration**	タイムゾーンは xs:dayTimeDuration 値として表示されます。この値は負の値であることができます。例えば -5 時間のタイムゾーンの値は **-PT5H** と表記することができます。

(C) 2015-2021 Altova GmbH

7.10.28.7 day-from-date

`xs:date` 値の日付の部分を表す `xs:integer` を返します。



言語

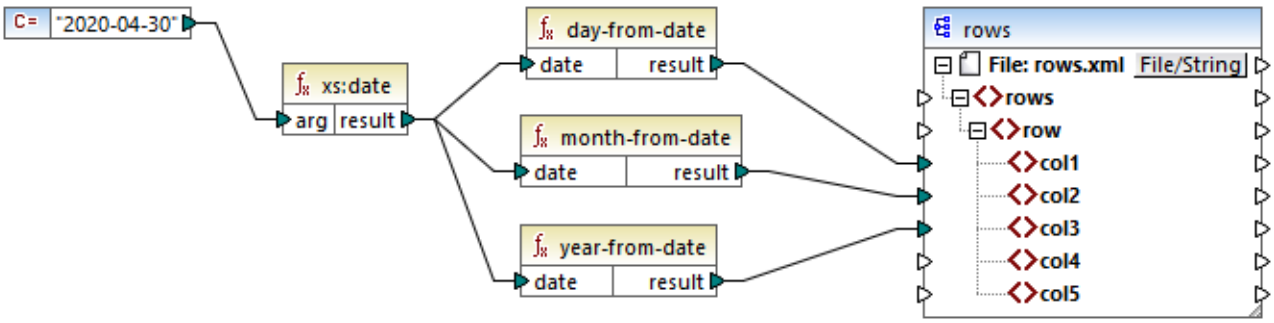
XQuery、XSLT 2.0、XSLT 3.0。

パラメーター

名前	型	説明
date	**xs:date**	型 `xs:date` の入力値。

サンプル

xs:date コンストラクタ関数を使用して次のマッピングは文字列を `xs:date` に変換します。**day-from-date**、**month-from-date**、および **year-from-date** 関数は日付の関連する部分を抽出しターゲット XML ファイル内の個別のアイテムに書き込みます。



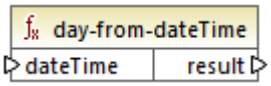
XQuery 1.0 マッピング

マッピングの出力は以下のようになります:

```
<rows>
   <row>
      <col1>30</col1>
      <col2>4</col2>
      <col3>2020</col3>
   </row>
</rows>
```

7.10.28.8 day-from-dateTime

`xs:dateTime` 値の日付の部分を表す `xs:integer` を返します。



言語

XQuery、XSLT 2.0、XSLT 3.0。

パラメーター

名前	型	説明
dateTime	**xs:dateTime**	型 `xs:dateTime` の入力値。

7.10.28.9 days-from-duration

引数として与えられた期間の値の正規表記の日付コンポーネントの部分を表す `xs:integer` を返します。

言語

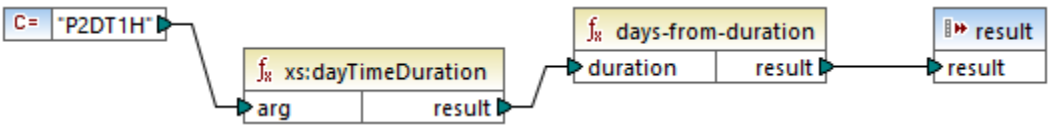
XQuery、XSLT 2.0、XSLT 3.0。

パラメーター

名前	型	説明
duration	**xs:duration**	型 `xs:duration` の入力値。

サンプル

下で表示されるマッピングは `P2DT1H`（2 日と1 時間）の `xs:dayTimeDuration` をコンストラクトし、入力として **days-from-duration** 関数に提供します。結果は **2** です。



XSLT 2.0 マッピング

メモ 期間が `P1DT24H`（1 日と24 時間）の場合、関数は **1** ではなく **2** を返します。これは `P1DT24H` の正規表現が実際 `P2D`（2 日）であるからです。

(C) 2015-2021 Altova GmbH

7.10.28.10 hours-from-dateTime

xs:dateTime 値の時間の部分を表す xs:integer を返します。

言語

XQuery、XSLT 2.0、XSLT 3.0。

パラメーター

名前	型	説明
dateTime	**xs:dateTime**	型 xs:dateTime の入力値。

7.10.28.11 hours-from-duration

引数として与えられた期間の値の正規表記の時間コンポーネントの部分を表す xs:integer を返します。

言語

XQuery、XSLT 2.0、XSLT 3.0。

パラメーター

名前	型	説明
duration	**xs:duration**	型 xs:duration の入力値。

サンプル

期間が**PT1H60M**（1 時間と60 分）の場合、関数は**1**ではなく**2**を返します。これは**PT1H60M** の正規表現が実際には**PT2H**（2 時間）であるからです。

7.10.28.12 hours-from-time

xs:time 値の時間の部分を表す xs:integer を返します。

言語

XQuery、XSLT 2.0、XSLT 3.0。

パラメーター

名前	型	説明
time	**xs:time**	型 xs:time の入力値。

7.10.28.13 minutes-from-dateTime

xs:dateTime 値の分数の部分を表す xs:integer を返します。

言語

XQuery、XSLT 2.0、XSLT 3.0。

パラメーター

名前	型	説明
dateTime	**xs:dateTime**	型 xs:dateTime の入力値。

7.10.28.14 minutes-from-duration

引数として与えられた期間の値の正規表記の分数コンポーネントの部分を表す xs:integer を返します。

言語

XQuery、XSLT 2.0、XSLT 3.0。

パラメーター

名前	型	説明
duration	**xs:duration**	型 xs:duration の入力値。

サンプル

期間が **PT1M60S** (1 分 60 秒) の場合 **1** ではなく **2** を返します。これは **PT1M60S** の正規表現は **PT2M** (2 分)だからです。

7.10.28.15 minutes-from-time

xs:time 値の分数の部分を表す xs:integer を返します。

言語

XQuery、XSLT 2.0、XSLT 3.0。

パラメーター

名前	型	説明
time	**xs:time**	型 xs:time の入力値。

(C) 2015-2021 Altova GmbH

7.10.28.16 month-from-date

xs:date 値の月数の部分を表す xs:integer を返します。

言語

XQuery、XSLT 2.0、XSLT 3.0。

パラメーター

名前	型	説明
date	**xs:date**	型 xs:date の入力値。

7.10.28.17 month-from-dateTime

xs:dateTime 値の月数の部分を表す xs:integer を返します。

言語

XQuery、XSLT 2.0、XSLT 3.0。

パラメーター

名前	型	説明
dateTime	**xs:dateTime**	型 xs:dateTime の入力値。

7.10.28.18 months-from-duration

引数として与えられた期間の値の正規表記の月数コンポーネントの部分を表す xs:integer を返します。

言語

XQuery、XSLT 2.0、XSLT 3.0。

パラメーター

名前	型	説明
duration	**xs:duration**	型 xs:duration の入力値。

7.10.28.19 seconds-from-dateTime

dateTime のローカライズされた値内の秒コンポーネントを表す xs:integer を返します。

言語

XQuery、XSLT 2.0、XSLT 3.0。

パラメーター

名前	型	説明
dateTime	**xs:dateTime**	

7.10.28.20 seconds-from-duration

引数として与えられた期間の値の正規表記の秒数コンポーネントの部分を表す `xs:integer` を返します。

言語

XQuery、XSLT 2.0、XSLT 3.0。

パラメーター

名前	型	説明
duration	**xs:duration**	型 `xs:duration` の入力値。

7.10.28.21 seconds-from-time

引数として提供された `xs:time` 値の分数の部分を表す `xs:integer` を返します。

言語

XQuery、XSLT 2.0、XSLT 3.0。

パラメーター

名前	型	説明
time	**xs:time**	型 `xs:time` の入力値。

(C) 2015-2021 Altova GmbH

7.10.28.22 subtract-dateTimes

dateTime1 の正規化された値と**dateTime2** の正規化された値間の差分に対応する `xs:dayTimeDuration` を返します。

言語

XQuery、XSLT 2.0、XSLT 3.0。

パラメーター

名前	型	説明
dateTime1	`xs:dateTime`	最初の入力値。
dateTime2	`xs:dateTime`	2番目の入力値。

7.10.28.23 subtract-dates

date1 の正規化された値と**date2** の正規化された値間の差分に対応する `xs:dayTimeDuration` を返します。

言語

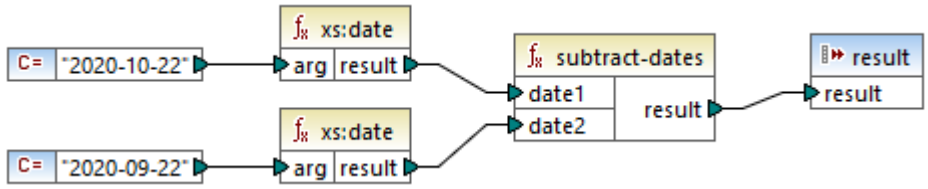
XQuery、XSLT 2.0、XSLT 3.0。

パラメーター

名前	型	説明
date1	`xs:date`	最初の入力値。
date2	`xs:date`	2番目の入力値。

サンプル

下に示されているマッピングは2つの日付の減算を表しています（2020-10-22 から 2020-09-22 を差し引く減算）。結果は30日を表す型 `xs:dayTimeDuration` の値 `P30D` です。



7.10.28.24 subtract-times

time1 の正規化された値と**time2** の正規化された値間の差分に対応する `xs:dayTimeDuration` を返します。

言語

XQuery、XSLT 2.0、XSLT 3.0。

パラメーター

名前	型	説明
time1	**xs:time**	最初の入力値。
time2	**xs:time**	2番目の入力値。

7.10.28.25 timezone-from-date

引数として提供された日付のタイムゾーンコンポーネントを返します。結果はUTC からの偏差を示す `xs:dayTimeDuration` 。この値は+14:00 から-14:00 時間の両方の値が含まれる範囲です。

言語

XQuery、XSLT 2.0、XSLT 3.0。

パラメーター

名前	型	説明
date	**xs:date**	型 `xs:date` の入力値。

7.10.28.26 timezone-from-dateTime

引数として提供された `xs:dateTime` 値のタイムゾーンコンポーネントを返します。結果はUTC からの偏差を示す `xs:dayTimeDuration` 。この値は+14:00 から-14:00 時間の両方の値が含まれる範囲です。

言語

XQuery、XSLT 2.0、XSLT 3.0。

パラメーター

名前	型	説明
dateTime	**xs:dateTime**	型 `xs:dateTime` の入力値。

(C) 2015-2021 Altova GmbH

7.10.28.27 timezone-from-time

引数として提供された `xs:time` 値のタイムゾーンコンポーネントを返します。結果はUTC からの偏差を示す `xs:dayTimeDuration` 。この値は+14:00 から-14:00 時間の両方の値が含まれる範囲です。

言語

XQuery、XSLT 2.0、XSLT 3.0。

パラメーター

名前	型	説明
time	**xs:time**	型 `xs:time` の入力値。

7.10.28.28 year-from-date

`xs:date` 値の年数の部分を表す `xs:integer` を返します。

言語

XQuery、XSLT 2.0、XSLT 3.0。

パラメーター

名前	型	説明
date	**xs:date**	型 `xs:date` の入力値。

7.10.28.29 year-from-dateTime

`xs:dateTime` 値の年数の部分を表す `xs:integer` を返します。

言語

XQuery、XSLT 2.0、XSLT 3.0。

パラメーター

名前	型	説明
dateTime	**xs:dateTime**	型 `xs:dateTime` の入力値。

7.10.28.30 years-from-duration

引数として与えられた期間の値の正規構文表記の年数コンポーネントの部分を表す `xs:integer` を返します。

言語

XQuery、XSLT 2.0、XSLT 3.0。

パラメーター

名前	型	説明
duration	**xs:duration**	型 `xs:duration` の入力値。

7.10.29 xpath2 | node functions (ノード関数)

xpath2 ライブラリからのノード関数はマッピングコンポーネント上のノード (アイテム) の情報を与えます。

lang 関数は ([en] などの) 言語コードを識別する文字列の引数を取ります。関数はコンテキストノードに関数の引数に一致する値を持つ `xml:lang` 属性があるかに従い **true** または **false** を返します。

local-name、**name**、および **namespace-uri** 関数はそれぞれローカル名、名前、入力ノードの名前空間 URI を返します。例えば、ノード **altova:Products** のローカル名は **Products**、名前は **altova:Products**、名前空間 URI は altova: prefix がバウンドされる名前空間の URI です (**local-name** 関数のためのサンプルを参照してください)。これら3個の関数には2つのバリアントが存在します:

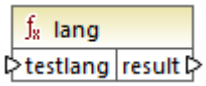
- 引数無し: 関数はコンテキストノードに適用されます (コンテキストノードのサンプルのためには **lang** 関数のために与えられたサンプルを参照してください)。
- ノードである必要がある引数: 関数は接続済みのノードに適用されます。

number 関数はノードを入力として取り、ノードを自動化し (すなわち、コンテンツを抽出し)、および値を小数に変換し、変換された値を返します。**number** 関数の2つのバリアントが存在します:

- 引数無し: 関数はコンテキストノードに適用されます (コンテキストノードのサンプルのためには **lang** 関数のために与えられたサンプルを参照してください)。
- ノードである必要がある引数: 関数は接続済みのノードに適用されます。

7.10.29.1 lang

コンテキストノードが **testlang** 引数に完全一致する、または サブセット である値を持つ `xml:lang` 属性を持つ場合 **true** を返します。それ以外の場合関数は **false** を返します。



言語

XQuery、XSLT 2.0、XSLT 3.0。

パラメーター

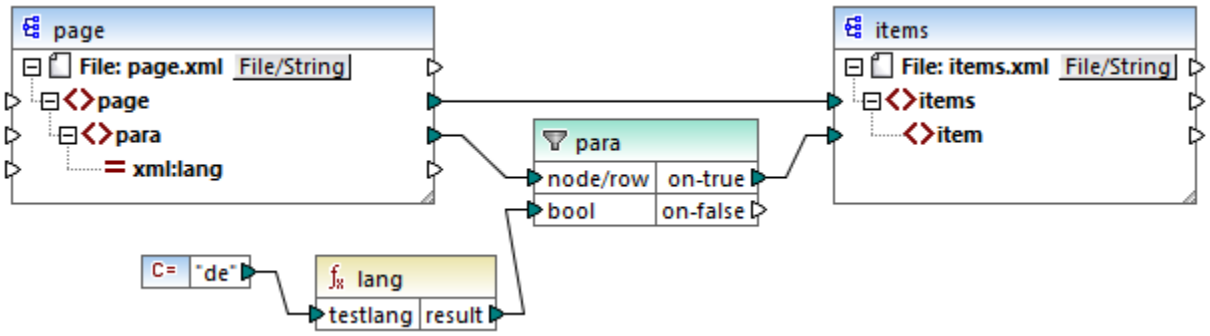
名前	型	説明
testlang	`xs:string`	チェックする言語コードは、例えば[en] です。

サンプル

以下のXML は`xml:lang` 属性のために異なる値を持つ**para** 要素を含んでいます。

```
<page>
   <para xml:lang="en">Good day!</para>
   <para xml:lang="fr">Bonjour!</para>
   <para xml:lang="de-AT">Grüss Gott!</para>
   <para xml:lang="de-DE">Guten Tag!</para>
   <para xml:lang="de-CH">Grüezi!</para>
</page>
```

下に示されてるマッピングは`lang` 関数を使用して国名のバリアントに関わらずドイツ語の段落のみをフィルターします。



XSLT 2.0 マッピング

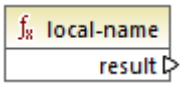
上のマッピング内では、ソース内の各 **para** 内で**item** はターゲット内に条件付けられて作成されます。`lang` 関数が**true** を返す箇所のノードのみをターゲットにパスするフィルターにより条件は提供されています。すなわち`xml:lang` 属性を[de]（[de]のサブセット）に設定しているこれらのノードはフィルターの条件を満たします。この結果、マッピングの出力は以下の通りです:

```
<items>
   <item>Grüss Gott!</item>
   <item>Guten Tag!</item>
   <item>Grüezi!</item>
</items>
```

para と**item** の間の親接続のため`lang` 関数は各 **para** のコンテキスト内で操作されます。マッピングコンテキストを参照してください。

7.10.29.2 local-name

引数として提供されたコンテキストノードの名前のローカルの部分を `xs:string` として返します。これは **local-name** 関数のパラメーター無しのバリアントです。ノードを明示的に指定するには入力ノードをパラメーターとして取る **local-name** 関数を使用してください。

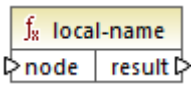


言語

XQuery、XSLT 2.0、XSLT 3.0。

7.10.29.3 local-name

node の名前のローカルの部分を `xs:string` として返します。



言語

XQuery、XSLT 2.0、XSLT 3.0。

パラメーター

名前	型	説明
node	**node()**	入力ノード。

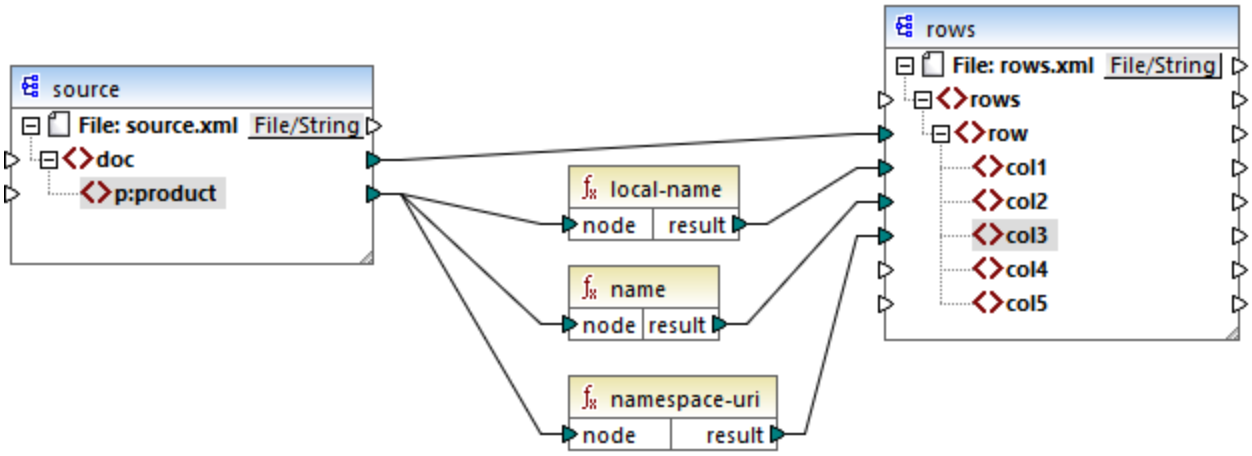
サンプル

次のXML ファイル内では `p:product` 要素の名前はプレフィックスされた装飾名（QName）です。プレフィックス [p] は名前空間 http://mycompany.com にマップされています。

```
<?xml version="1.0" encoding="UTF-8"?>
<doc xmlns:p="http://mycompany.com" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:noNamespaceSchemaLocation="source.xsd">
   <p:product/>
</doc>
```

次のマッピングはローカル名、名前、およびノードの名前空間 URI を抽出し、これらの値をターゲットファイルに書き込みます:

(C) 2015-2021 Altova GmbH



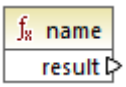
XSLT 2.0 マッピング

マッピングの出力は以下のように表示します: 各 **col** アイテムは`local-name`、`name`、および`namespace-uri` 関数の結果をリストしています。

```
<rows>
   <row>
      <col1>product</col1>
      <col2>p:product</col2>
      <col3>http://mycompany.com</col3>
   </row>
</rows>
```

7.10.29.4 name

コンテキストノードの名前を返します。これは`name` 関数のパラメーター無しのバリアントです。ノードを明示的に指定するには入力ノードを引数としてとる`name` 関数を使用してください。

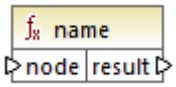


言語

XQuery、XSLT 2.0、XSLT 3.0。

7.10.29.5 name

ノードの名前を返します。



言語

XQuery、XSLT 2.0。

パラメーター

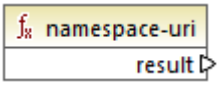
名前	型	説明
node	`node()`	入力ノード。

サンプル

`local-name` 関数のために与えられたサンプルを参照してください。

7.10.29.6 namespace-uri

コンテキストノードのQName の名前空間 URI を`xs:string` として返します。コンテキストノードがマッピング内の接続により決定される個所の`namespace-uri` 関数のパラメーター無しのバリアントです。ノードを明示的に指定するには入力ノードをパラメーターとして取る `namespace-uri` 関数を使用してください。



言語

XQuery、XSLT 2.0、XSLT 3.0。

7.10.29.7 namespace-uri

`xs:string` としての**node** のQName 名前空間 URI を返します。

(C) 2015-2021 Altova GmbH

言語

XQuery、XSLT 2.0。

パラメーター

名前	型	説明
node	`node()`	入力ノード。

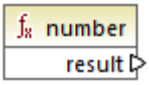
サンプル

`local-name` 関数のために与えられたサンプルを参照してください。

7.10.29.8 number

`xs:double` に変換されたコンテキストノードの値を返します。これは`number` 関数のパラメーター無しのバリアントです。ノードを明示的に指定するには入力ノードを引数として取る`number` 関数を使用してください。

数値に変換可能な方はBoolean、数値文字列、他の数値型 のみです。(非数値文字列などの)非数値入力値 は(数字ではない) NaN 内に結果します。

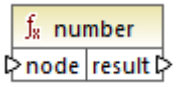


言語

XQuery、XSLT 2.0、XSLT 3.0。

7.10.29.9 number

`xs:double` に変換された**node** の値を返します。数値に変換可能な方はBoolean、数値文字列、他の数値型 のみです。(非数値文字列などの)非数値入力値 は(数字ではない) NaN 内に結果します。



言語

XQuery、XSLT 2.0、XSLT 3.0。

パラメーター

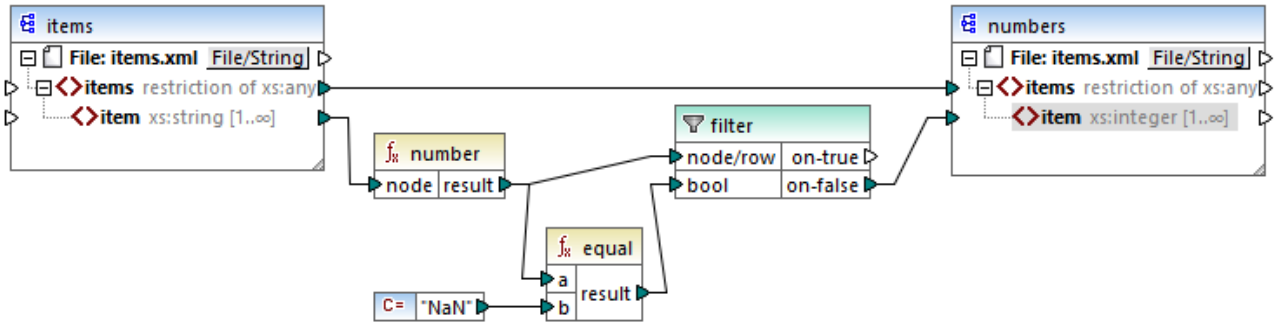
名前	型	説明
node	`mf:atomic`	入力ノード。

サンプル

次のXML は型 `string` のアイテムを含んでいます:

```
<items>
   <item>1</item>
   <item>2</item>
   <item>Jingle Bells</item>
</items>
```

下で説明されているマッピングはこれらすべての文字列を数値に変換しターゲット XML ファイルに書き込もうとします。ターゲット XML コンポーネント内の**item** のデータ型は`xs:integer` ですが、ソース**item** は`xs:string` データソースです。変換に成功しない場合、アイテムはスキップされなければならず、ターゲットファイルにコピーされません。



XSLT 2.0 マッピング

マッピングの目的を達成するためにフィルターが使用されています。`equal` 関数は変換の結果が[NaN] であるかをチェックします。これが false の場合、これは変換の成功を示し、アイテムはターゲットにコピーされます。マッピングの出力は以下のようになります:

```
<items>
   <item>1</item>
   <item>2</item>
</items>
```

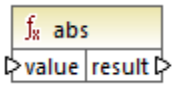
7.10.30 xpath2 | numeric functions (数値関数)

xpath2 ライブラリの数値関数には`abs` と`round-half-to-even` 関数が含まれます。

(C) 2015-2021 Altova GmbH

7.10.30.1 abs

引数として提供された絶対値を返します。例えば入力引数が**-2** または**2** の場合、関数は**2** を返します。



言語

XQuery、XSLT 2.0、XSLT 3.0。

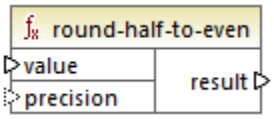
パラメーター

名前	型	説明
value	`xs:decimal`	入力値。

7.10.30.2 round-half-to-even

`round-half-to-even` 関数は任意の2番目の引数内で提供された数値（最初の引数）を精度（小数の数）の度数に提供します。例えば、最初の引数が**2.141567** で2番目の引数が**3** の場合、最初の引数（数値）は3番目の小数で端数処理され、結果は**2.141** になります。精度（2 番目パラメーター）が指定されていない場合、番号は小数が無いように端数処理され与えられた値は整数になります。

与えられている数値内の数字が2つの値の中間にある場合関数の名前内の[偶数]は数値を偶数に端数処理することを指しています。例えば、`round-half-to-even(3.475, 2)` は**3.48** という値を返します。



言語

XQuery、XSLT 2.0、XSLT 3.0。

パラメーター

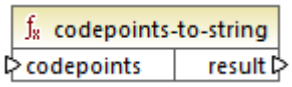
名前	型	説明
value	`xs:decimal`	端数処理される入力値を提供する必須の引数。
precision	`xs:integer`	端数処理する小数点の数値を指定する任意の引数。デフォルト の値は**0** です。

7.10.31 xpath2 | string functions (文字列関数)

xpath2 ライブラリの文字列関数により文字列を処理することができます (これには文字列の比較、文字列を大文字または小文字への変換、文字列またはその他からサブ文字列の抽出などが含まれます)。

7.10.31.1 codepoints-to-string

Unicode コードポイントのシーケンスから文字列を作成します。この関数は `string-to-codepoints` 関数の反対です。



言語

XQuery、XSLT 2.0、XSLT 3.0。

パラメーター

名前	型	説明
codepoints	`ZeroOrMore xs:integer`	この入力は各整数がUnicode コードポイントを指定する箇所で整数型のアイテムのシーケンスに接続されている必要があります。

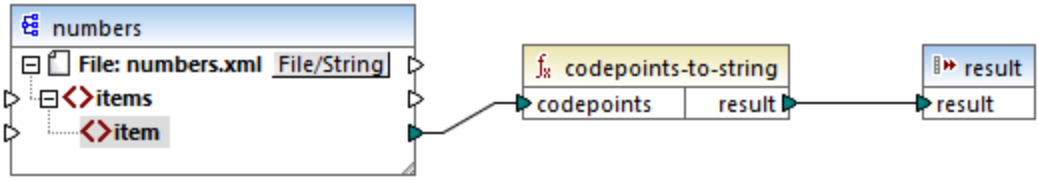
サンプル

次のXML はUnicode コードポイント値を保管する複数の **item** 要素を含んでいます。

```
<items>
   <item>77</item>
   <item>97</item>
   <item>112</item>
   <item>70</item>
   <item>111</item>
   <item>114</item>
   <item>99</item>
   <item>101</item>
</items>
```

下で示されているマッピングは `codepoint-to-string` 関数への引数としてアイテムのシーケンスを提供します。

(C) 2015-2021 Altova GmbH



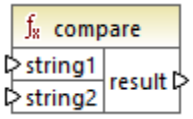
XSLT 2.0 マッピング

マッピング出力は`MapForce` です。

7.10.31.2 compare

`Compare` 関数は2つの文字列を引数としてとり、等価とアルファベット順に比較します。**string1** が**string2** よりもアルファベット順に比較して小さい場合(例えば、2つの文字列が[A] と[B] の場合) 関数は**-1** を返します。2つの文字列が等価の場合(例えば[A] と[A]) 関数は**0** を返します。**string1** が**string2** よりも大きい場合(例えば[B] と[A]) 関数は**1** を返します。

関数のこのバリアントはUnicode であるデフォルトの照会順序を使用しています。この関数の他のvariant は照会順序を引数として提供可能な箇所で存在しています。



言語

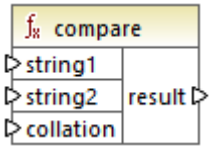
XQuery、XSLT 2.0、XSLT 3.0。

パラメーター

名前	型	説明
string1	`xs:string`	最初の入力文字列。
string2	`xs:string`	2番目の入力文字列。

7.10.31.3 compare

引数として提供された照会順序を使用して `compare` 関数は2つの文字列を引数としてとり、等価とアルファベット順に比較します。**string1** が**string2** よりもアルファベット順に比較して小さい場合(例えば、2つの文字列が[A] と[B] の場合) 関数は**-1** を返します。2つの文字列が等価の場合(例えば[A] と[A])関数は**0** を返します。**string1** が**string2** よりも大きい場合(例えば[B] と[A]) 関数は**1** を返します。



言語

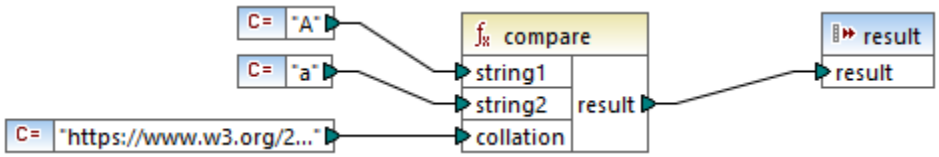
XQuery、XSLT 2.0、XSLT 3.0。

パラメーター

名前	型	説明
string1	`xs:string`	最初の入力文字列。
string2	`xs:string`	2番目の入力文字列。
collation	`xs:string`	文字列比較のために使用する照会順序を指定します。この入力は`default-collation` 関数の出力を元にする可能性があります、または `http://www.w3.org/2005/xpath-functions/collation/html-ascii-case-insensitive` などの照会順序の場合があります。

サンプル

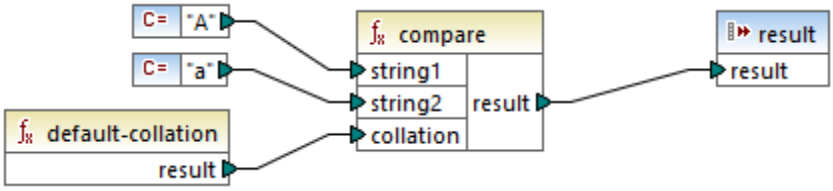
以下のマッピングは文字列 [A] と[a] を定数により提供される大文字と小文字を区別した照会順序 `http://www.w3.org/2005/xpath-functions/collation/html-ascii-case-insensitive` を使用して比較しています。



XSLT 2.0 マッピング

上のマッピングの結果は **0** です。これは両方の文字列が同様に扱われることを意味します。`default-collation` 関数により与えられている照会順序を置き換えると照会順序はデフォルトのUnicode コードポイント照会順序に変更され、マッピングの結果は **-1** になります。([A] は[a] よりもアルファベット順では小さいからです)。

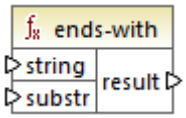
(C) 2015-2021 Altova GmbH



7.10.31.4 ends-with

string が**substr** で終わる場合は**true** を返し、それ以外の場合は**false** を返します。戻り値は`xs:boolean` です。

関数のこのバリアントはUnicode であるデフォルトの照会順序を使用しています。この関数の他のvariant は照会順序を引数として提供可能な箇所で存在しています。



言語

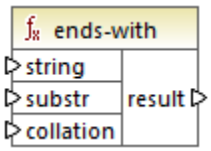
XQuery、XSLT 2.0、XSLT 3.0。

パラメーター

名前	型	説明
string	`xs:string`	入力文字列（すなわち[haystack]）。
substr	`xs:string`	サブ文字列（すなわち[needle]）。

7.10.31.5 ends-with

string が**substr** で終わる場合は**true** を返し、それ以外の場合は**false** を返します。戻り値は`xs:boolean` です。



言語

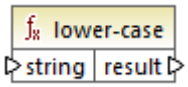
XQuery、XSLT 2.0、XSLT 3.0。

パラメーター

名前	型	説明
string	`xs:string`	入力文字列（すなわち [haystack]）。
substr	`xs:string`	サブ文字列（すなわち [needle]）。
collation	`xs:string`	文字列比較のために使用する照会順序を指定します。この入力は `default-collation` 関数の出力を元にする可能性があります、または `http://www.w3.org/2005/xpath-functions/collation/html-ascii-case-insensitive` などの照会順序の場合があります。

7.10.31.6 lower-case

全ての文字を対応する小文字に変換後 **string** の値を返します。



言語

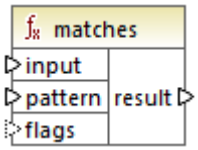
XQuery、XSLT 2.0。

パラメーター

名前	型	説明
string	`xs:string`	入力値。

7.10.31.7 matches

`matches` 関数により、(input パラメーターにより) 与えられた文字列が、(pattern パラメーターから得られる) 正規表現にマッチするかがチェックされます。正規表現の構文は、XML スキーマの `pattern` ファセットにて定義する必要があります。文字列が正規表現にマッチすれば **true** が返され、それ以外の場合は **false** が返されます。



(C) 2015-2021 Altova GmbH

言語

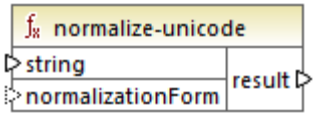
XQuery、XSLT 2.0、XSLT 3.0。

パラメーター

名前	型	説明
input	`xs:string`	入力文字列。
pattern	`xs:string`	一致する正規表現に関しては正規表現を参照してください。
flags	`xs:string`	一致に影響を与える任意の引数。この引数は以下のフラグの任意の組み合わせを与える場合があります: `i`、`m`、`s`、`x`。例えば `imx` のように、複数のフラグを使用することもできます。フラグが使用されていない場合、4 つのフラグのデフォルト値が使用されます。4 種類あるフラグは以下の通りです: `i` 大文字と小文字を区別するモードを使用します。デフォルトでは大文字と小文字を区別します。 `m` 複数行モードを使用して、改行文字 (`x0a`) により複数行に分けられている入力文字列を、1 つの文字列として扱います。メタ文字の `^` と `$` が、それぞれ行頭と行末を表します。デフォルトは文字列モードになっており、個々の文字列が `^` から始まり `$` で終わります。 `s` dot-all モードを使用します。デフォルトではメタ文字は新規のライン文字 (`x0a`) 以外のすべての文字に一致する not-dot-all モードです。dot-all mode ではドットはすべての新規ライン文字に一致します。 `x` 空白文字を無視する。デフォルトでは空白文字は無視されません。

7.10.31.8 normalize-unicode

指定されている正規化書式のルールに従い正規化されている **string** の値 (2番目の引数) を返します。Unicode 正規化に関する詳細は §2.2 of https://www.w3.org/TR/charmod-norm/ を参照してください。



言語

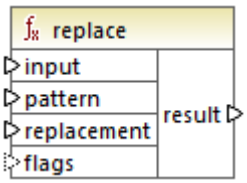
XQuery、XSLT 2.0、XSLT 3.0。

パラメーター

名前	型	説明
string	xs:string	正規化される文字列の値
normalizationForm	xs:string	正規化フォームを提供するオプションの引数。デフォルトはUnicode 正規化形式 C（NFC）です。 フォームNFC、NFD、NFKC、および NFKD の正規化は、、サポートされています。

7.10.31.9 replace

この関数は入力文字列、正規表現、置換文字列を引数としてとります。置換文字列を使用して入力文字列内の正規表現のすべての一致を置き換えます。入力文字列内で正規表現が2つのオーバーラップする文字列に一致する場合、最初の一致のみが置き換えられます。



言語

XQuery、XSLT 2.0、XSLT 3.0。

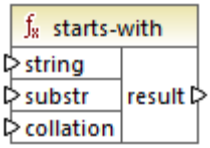
パラメーター

名前	型	説明
input	xs:string	入力文字列。
pattern	xs:string	一致する正規表現に関しては正規表現を参照してください。
replacement	xs:string	置換文字列。
flags	xs:string	一致に影響を与える任意の引数。この引数は`matches` 関数内の**flags** 引数と同じ方法で使用されます。

(C) 2015-2021 Altova GmbH

7.10.31.10 starts-with

string が**substr** で始まる場合は**true** を返し、それ以外の場合は**false** を返します。戻り値は`xs:boolean` です。文字列の比較は指定されている照会順序に従い行われます。



言語

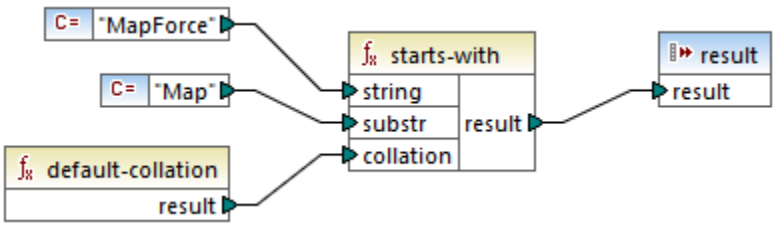
XQuery、XSLT 2.0、XSLT 3.0。

パラメーター

名前	型	説明
string	`xs:string`	入力文字列（すなわち[haystack]）。
substr	`xs:string`	サブ文字列（すなわち[needle]）。
collation	`xs:string`	文字列比較のために使用する照会順序を指定します。この入力は`default-collation` 関数の出力を元にする可能性があります、または`http://www.w3.org/2005/xpath-functions/collation/html-ascii-case-insensitive` などの照会順序の場合があります。

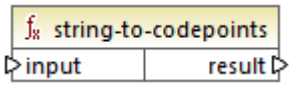
サンプル

デフォルトのUnicode 照会順序が使用されるため入力文字列 [MapForce] はサブ文字列 [Map] から開始するため次のマッピングは値 `true` を返します。



7.10.31.11 string-to-codepoints

引数として提供された文字列を構成するUnicode コードポイント(整数値) のシーケンスを返します。この関数は **codepoints-to-string** 関数の反対です。



言語

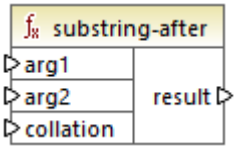
XQuery、XSLT 2.0、XSLT 3.0。

パラメーター

名前	型	説明
input	**xs:string**	入力文字列。

7.10.31.12 substring-after

文字列 **arg2** の後に発生する文字列 **arg1** の部分を返します。



言語

XQuery、XSLT 2.0、XSLT 3.0。

パラメーター

名前	型	説明
arg1	**xs:string**	入力文字列 (すなわち [haystack])。
arg2	**xs:string**	サブ文字列 (すなわち [needle])。
collation	**xs:string**	文字列比較のために使用する照合順序を指定します。この入力は **default-collation** 関数の出力を元にする可能性があります、または http://www.w3.org/2005/xpath-functions/collation/html-

(C) 2015-2021 Altova GmbH

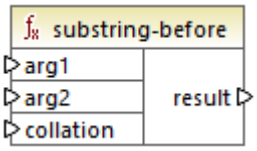
名前	型	説明
		ascii-case-insensitive などの照会順序の場合があります。

サンプル

arg1 が[MapForce]、**arg2** が[Map]の場合、**collation** は`default-collation`で、関数は[Force]を返します。

7.10.31.13 substring-before

文字列 **arg2** の前に発生する文字列 **arg1** の部分を返します。



言語

XQuery、XSLT 2.0、XSLT 3.0。

パラメーター

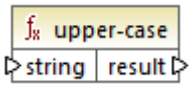
名前	型	説明
arg1	`xs:string`	入力文字列（すなわち[haystack]）。
arg2	`xs:string`	サブ文字列（すなわち[needle]）。
collation	`xs:string`	文字列比較のために使用する照会順序を指定します。この入力は`default-collation`関数の出力を元にする可能性があります、または http://www.w3.org/2005/xpath-functions/collation/html-ascii-case-insensitive などの照会順序の場合があります。

サンプル

arg1 が[MapForce]、**arg2** が[Force]の場合 **collation** は`default-collation`で、関数は[Map]を返します。

7.10.31.14 upper-case

全ての文字を対応する大文字に変換後 **string** の値を返します。



言語

XQuery、XSLT 2.0。

パラメーター

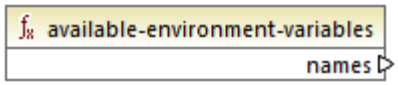
名前	型	説明
string	`xs:string`	入力文字列。

7.10.32 xpath3 | external information functions

xpath3 ライブラリの外部情報関数はXSLT 実行環境に関する情報の取得、または外部リソースからのデータの抽出を可能にします。

7.10.32.1 available-environment-variables

`environment-variable` 関数に適切な環境変数名のリストを文字列の(空である可能性のある) シーケンスで返します。

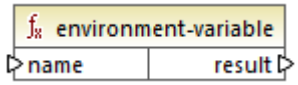


言語

XSLT 3.0。

7.10.32.2 environment-variable

存在する場合システム環境関数の値を返します。戻り型は `xs:string` です。



(C) 2015-2021 Altova GmbH

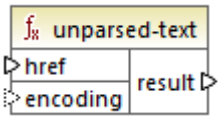
言語
XSLT 3.0。

パラメーター

名前	型	説明
name	`xs:string`	環境変数の名前

7.10.32.3 unparsed-text

外部リソース（例えばファイル）を読み取り、リソースの文字列表記を返します。



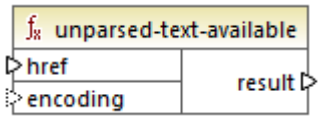
言語
XSLT 3.0。

パラメーター

名前	型	説明
href	`xs:string`	URI レファレンスのフォーム内の文字列
encoding	`xs:string`	オプションの引数。エンコードの名前を指定します。例えば"UTF-8"、""UTF-16"など。エンコードが自動的に決定されない場合 UTF-8 が想定されます。

7.10.32.4 unparsed-text-available

特定の引数を持つ`unparsed-text` への呼び出しが成功するかを決定します。戻り型は`xs:boolean` です。



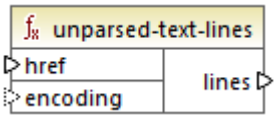
言語
XSLT 3.0。

パラメーター

名前	型	説明
href	xs:string	URI レファレンスのフォーム内の文字列
encoding	xs:string	オプションの引数。エンコードの名前を指定します。例えば "UTF-8"、""UTF-16" など。エンコードが自動的に決定されない場合 UTF-8 が想定されます。

7.10.32.5 unparsed-text-lines

外部リソース（例えば、ファイル）を読み取り、そのコンテンツを文字列のシーケンスとして返します。リソースの文字列表記がテキストのそれぞれのラインにつき1個表示されます。



言語

XSLT 3.0。

パラメーター

名前	型	説明
href	xs:string	URI レファレンスのフォーム内の文字列
encoding	xs:string	オプションの引数。エンコードの名前を指定します。例えば "UTF-8"、""UTF-16" など。エンコードが自動的に決定されない場合 UTF-8 が想定されます。

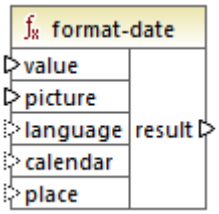
7.10.33 xpath3 | formatting functions

xpath3 ライブラリを使用可能な書式設定関数は日付、時刻、および整数の値を書式設定するために使用されます。

(C) 2015-2021 Altova GmbH

7.10.33.1 format-date

表示のために`xs:date`値を含む文字列を返します。



言語

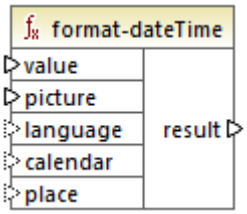
XSLT 3.0。

パラメーター

名前	型	説明
value	`xs:date`	書式設定する入力`xs:date`値です。必須のパラメーターです。
picture	`xs:string`	必須のパラメーターです。 "XPath とXquery 関数と演算子 3.1" W 3C 推奨のセクション 9.8.4.1 を参照してください (https://www.w3.org/TR/xpath-functions-31)。
language	`xs:string`	任意のパラメーターです。 "XPath とXquery 関数と演算子 3.1" W 3C 推奨のセクション 9.8.4.8 を参照してください (https://www.w3.org/TR/xpath-functions-31)。
カレンダー	`xs:string`	上記と同様です。
place	`xs:string`	上記と同様です。

7.10.33.2 format-dateTime

表示のために`xs:dateTime` 値を含む文字列を返します。



言語

XSLT 3.0。

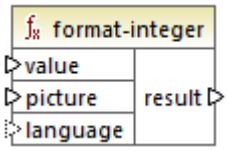
パラメーター

名前	型	説明
value	**xs:dateTime**	書式設定する入力 `xs:dateTime` 値です。
picture	**xs:string**	必須のパラメーターです。 "XPath とXquery 関数と演算子 3.1" W 3C 推奨のセクション 9.8.4.1 を参照してください (https://www.w3.org/TR/xpath-functions-31)。
language	**xs:string**	任意のパラメーターです。 "XPath とXquery 関数と演算子 3.1" W 3C 推奨のセクション 9.8.4.8 を参照してください (https://www.w3.org/TR/xpath-functions-31)。
カレンダー	**xs:string**	上記と同様です。
place	**xs:string**	上記と同様です。

(C) 2015-2021 Altova GmbH

7.10.33.3 format-integer

指定されている場合与えられた自然言語の変換を使用して与えられた文字列に従い整数を書式設定します。



言語

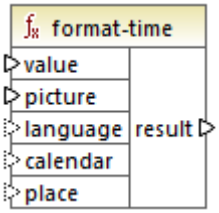
XSLT 3.0。

パラメーター

名前	型	説明
value	`xs:integer`	書式設定する入力整数値
picture	`xs:string`	必須のパラメーターです。 "XPath とXquery 関数と演算子 3.1" W 3C 推奨のセクション 4.6.1 を参照してください (https://www.w3.org/TR/xpath-functions-31)。
language	`xs:string`	任意のパラメーターです。 値が書式設定するために従う自然言語を設定します。指定される場合、この値は空の文字列または "拡張マークアップ言語 (XML) 1.0 W 3C 奨励 (https://www.w3.org/TR/xml) に従った`xml:lang` 属性のために許可される値になります。according to the.

7.10.33.4 format-time

表示のために`xs:time`値を含む文字列を返します。



言語

XSLT 3.0。

パラメーター

名前	型	説明
value	`xs:time`	書式設定する入力`xs:time`値です。
picture	`xs:string`	必須のパラメーターです。 "XPath とXquery 関数と演算子 3.1" W 3C 推奨のセクション 9.8.4.1 を参照してください (https://www.w3.org/TR/xpath-functions-31)。
language	`xs:string`	任意のパラメーターです。 "XPath とXquery 関数と演算子 3.1" W 3C 推奨のセクション 9.8.4.8 を参照してください (https://www.w3.org/TR/xpath-functions-31)。
カレンダー	`xs:string`	上記と同様です。
place	`xs:string`	上記と同様です。

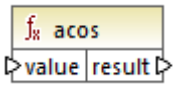
7.10.34 xpath3 | math functions

xpath3 ライブラリの数学関数は三角および他の数学計算を行うために使用されます。

(C) 2015-2021 Altova GmbH

7.10.34.1 acos

0 から **pi** までの範囲で角度のアークコサインを返します。



言語

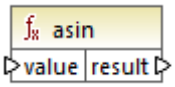
XSLT 3.0。

パラメーター

名前	型	説明
value	`xs:double`	入力値。

7.10.34.2 asin

-pi/2 から **pi/2** までの範囲で角度のアークサインを返します。



言語

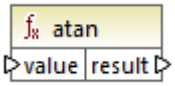
XSLT 3.0。

パラメーター

名前	型	説明
value	`xs:double`	入力値。

7.10.34.3 atan

-pi/2 から **pi/2** までの範囲で角度のアークタンジェントを返します。



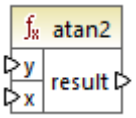
言語
XSLT 3.0。

パラメーター

名前	型	説明
value	`xs:double`	入力値。

7.10.34.4 atan2

座標 (x, y) と正 x-軸を持つ平面乗の原点に対しての角度をラジアンで返します。



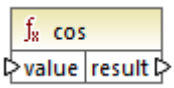
言語
XSLT 3.0。

パラメーター

名前	型	説明
y	`xs:double`	x 座標。
x	`xs:double`	y 座標。

7.10.34.5 cos

value により与えられた角度の三角コサインを返します。値のユニットはラジアンです。



言語
XSLT 3.0。

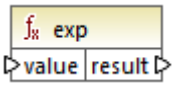
(C) 2015-2021 Altova GmbH

パラメーター

名前	型	説明
value	`xs:double`	入力値。

7.10.34.6 exp

value に累乗したオイラー数 e が返されます。



言語

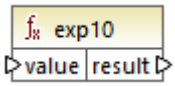
XSLT 3.0。

パラメーター

名前	型	説明
value	`xs:double`	入力値。

7.10.34.7 exp10

10の値乗を返します。



言語

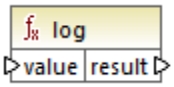
XSLT 3.0。

パラメーター

名前	型	説明
value	`xs:double`	入力値。

7.10.34.8 log

value の(底がe の) 自然対数を返します。



言語

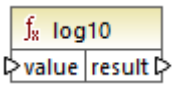
XSLT 3.0。

パラメーター

名前	型	説明
value	`xs:double`	入力値。

7.10.34.9 log10

value の(底が10 の) 10 十進対数を返します。



言語

XSLT 3.0。

パラメーター

名前	型	説明
value	`xs:double`	入力値。

7.10.34.10 pi

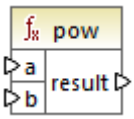
数理定数 **pi** の近似値を返します。

(C) 2015-2021 Altova GmbH

言語
XSLT 3.0。

7.10.34.11 pow

b に累乗された **a** の値を返します。



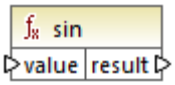
言語
XSLT 3.0。

パラメーター

名前	型	説明
a	`xs:double`	入力値 **a**。
b	`xs:double`	入力値 **b**。

7.10.34.12 sin

value により与えられた角度の三角サインを返します。値のユニットはラジアンです。



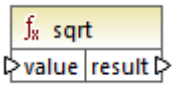
言語
XSLT 3.0。

パラメーター

名前	型	説明
value	`xs:double`	入力値。

7.10.34.13 sqrt

引数の負以外平方根を返します。



言語

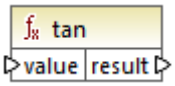
XSLT 3.0。

パラメーター

名前	型	説明
value	`xs:double`	入力値。

7.10.34.14 tan

value により与えられた角度の三角タンジェントを返します。値のユニットはラジアンです。



言語

XSLT 3.0。

パラメーター

名前	型	説明
value	`xs:double`	入力値。

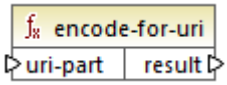
7.10.35 xpath3 | URI functions

xpath3 ライブラリ内のURI 関数は、URI 内での使用を目的としたエンコード、エスケープおよび変換を行います。

(C) 2015-2021 Altova GmbH

7.10.35.1 encode-for-uri

URI のパスセグメント内で使用される予定の文字列内の予約された文字をエンコードします。この関数の詳細については、"XPath と Xquery 関数と演算子 3.1" W 3C 推奨のセクション 6.2 を参照してください(https://www.w3.org/TR/xpath-functions-31)。



言語

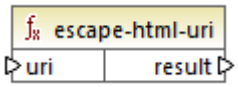
XSLT 3.0。

パラメーター

名前	型	説明
uri-part	`xs:string`	エンコードする入力 URI 値。

7.10.35.2 escape-html-uri

HTML ユーザーエージェントが URI を含むことを期待される属性値を処理する方法と同じ方法で URI をエスケープします。この関数の詳細については、"XPath と Xquery 関数と演算子 3.1" W 3C 推奨のセクション 6.4 を参照してください (https://www.w3.org/TR/xpath-functions-31)。



言語

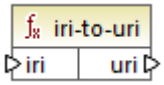
XSLT 3.0。

パラメーター

名前	型	説明
uri	`xs:string`	エスケープする入力 URI 値。

7.10.35.3 iri-to-uri

IRI (国際リソース識別子) を含む文字列を URI (統一資源識別子)に変換します。この関数の詳細については、"XPath と Xquery 関数と演算子 3.1" W 3C 推奨のセクション 6.3 を参照してください(https://www.w3.org/TR/xpath-functions-31)。



言語

XSLT 3.0。

パラメーター

名前	型	説明
iri	`xs:string`	入力 IRI 値。

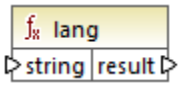
7.10.36 xslt | xpath functions (xpath 関数)

このサブグループ内の関数はマッピングアイテム(またはノード)に関する情報を抽出する XPath 1.0 関数です。関数の多くはノードを引数として取り、そのノードに関する情報を返します。The `last` と`position` 関数はマッピング上の接続により決定される現在のマッピングコンテキスト 内で作動します。

メモ その他の XPath 1.0 関数を **core** 関数 ライブラリで確認することもできます。

7.10.36.1 lang

コンテキストノードが**string** 引数に完全一致する、またはサブセットである値を持つ `xml:lang` 属性を持つ場合 **true** を返します。それ以外の場合関数は**false** を返します。



言語

XSLT 1.0。

パラメーター

名前	型	説明
string	`xs:string`	チェックする言語コードは、例えば[en] です。

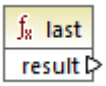
サンプル

xpath2 ライブラリの`lang` 関数のために与えられたサンプルを参照してください。

(C) 2015-2021 Altova GmbH

7.10.36.2 last

処理されたノードリスト内の最後のノードのポジション番号を返します。



言語

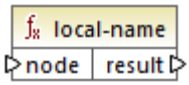
XSLT 1.0。

サンプル

xpath2 ライブラリの`last` 関数のために与えられたサンプルを参照してください。

7.10.36.3 local-name

引数として提供されたノードの名前のローカルの部分を返します。



言語

XSLT 1.0、XSLT 2.0。

パラメーター

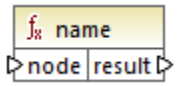
名前	型	説明
node	`node()`	入力ノード。

サンプル

xpath2 ライブラリの`local-name` 関数のために与えられたサンプルを参照してください。

7.10.36.4 name

引数として提供されたノードの名前を返します。



言語
XSLT 1.0、XSLT 2.0。

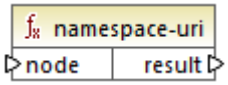
パラメーター

名前	型	説明
node	node()	入力ノード。

サンプル
xpath2 ライブラリの local-name 関数のために与えられたサンプルを参照してください。

7.10.36.5 namespace-uri

引数として提供されたノードの名前空間 URI を返します。



言語
XSLT 1.0、XSLT 2.0。

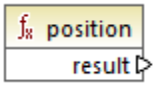
パラメーター

名前	型	説明
node	node()	入力ノード。

サンプル
xpath2 ライブラリの local-name 関数のために与えられたサンプルを参照してください。

7.10.36.6 position

現在処理されているノードセット内の現在のノードのポジションを返します。



(C) 2015-2021 Altova GmbH

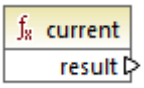
言語
XSLT 1.0。

7.10.37 xslt | xslt functions (xslt 関数)

このグループ内の関数はその他のXSLT 1.0 関数です。

7.10.37.1 current

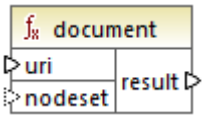
`current` 関数は引数を取らず、現在のノードを返します。



言語
XSLT 1.0。

7.10.37.2 document

外部 XML ドキュメントからノードにアクセスします。関数の結果は、出力ドキュメント内にあるノードへ返されます。



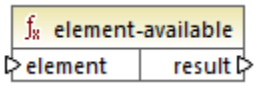
言語
XSLT 1.0。

パラメーター

名前	型	説明
uri	`xs:string`	必須。XML ドキュメントへのパスを指定します。XML ドキュメントは有効で解析可能である必要があります。
nodeset	`node()`	任意。ノード、および相対的である場合最初の引数として与えられているURI を解決するため使用されたベースURI を指定します。

7.10.37.3 element-available

element-available 関数は関数の文字列引数として入力されている要素がXSLT プロセッサーによりサポートされているかをテストします。引数文字列はQName として評価されます。これらはマッピングのために生成される基になるXSLT 内のこれらの名前空間のために宣言されているプレフィックスのため、XSLT 要素は`xsl:` プレフィックスを持つ必要があり、XML スキーマ要素は`xs:` プレフィックスを持つ必要があります。関数はブール値を返します。



言語

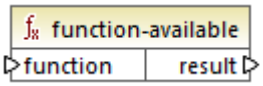
XSLT 1.0。

パラメーター

名前	型	説明
element	**xs:string**	要素名。

7.10.37.4 function-available

function-available 関数は**element-available** 関数に類似しており、関数の引数として提供されている関数名がXSLT プロセッサーによりサポートされているかテストします。入力文字列はQName として評価されます。関数はブール値を返します。



言語

XSLT 1.0。

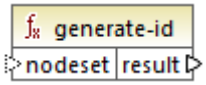
パラメーター

名前	型	説明
function	**xs:string**	関数名。

7.10.37.5 generate-id

generate-id 関数は任意の入力引数により識別されるノードセット内の最初のノードを識別する一意の文字列を生成します。入力パラメーターが与えられない場合、コンテキストノード上のID が生成されます。出力ドキュメントにある任意のノードに対して出力結果を接続することができます。

(C) 2015-2021 Altova GmbH



言語

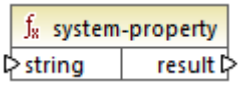
XSLT 1.0、XSLT 2.0、XSLT 3.0。

パラメーター

名前	型	説明
nodeset	`node()`	入力ノードを提供するオプションの引数。

7.10.37.6 system-property

`system-property` 関数はXSLT プロセッサー(システム)のプロパティを返します。入力された文字列はQName として評価されるため、XSLT スタイルシート内でXSLT 名前空間に関連付けられているxsl:prefix を使用する必要があります。これらは`xsl:version`、`xsl:vendor`、と`xsl:vendor-url` です。基になるXSLT スタイルシート内のXSLT 名前空間とプレフィックスが関連付けられるためQname として入力文字列は評価され`xsl:` プレフィックスを持つ必要があります。



言語

XSLT 1.0、XSLT 2.0、XSLT 3.0。

パラメーター

名前	型	説明
string	`xs:string`	以下のいずれかであるプロパティ名を指定します: `xsl:version`、`xsl:vendor`、`xsl:vendor-url`。

7.10.37.7 unparsed-entity-uri

関数への入力文字列はDTD 内部にて宣言された(パースされていない)エンティティーにマッチする必要があります。パースされていないエンティティ(例えばイメージ)はパースされていないエンティティをロケートするURI を持ちます。関数の入力文字列はDTD 内で宣言されているパースされていないエンティティの名前と一致する必要があります。関数は出力ドキュメント内のノードに移動されるパースされていないエンティティのURI を返します。例えば、**href** ノードなど。

言語

XSLT 1.0。

パラメーター

名前	型	説明
string	`xs:string`	URI が抽出される解析されていないエンティティの名前。

(C) 2015-2021 Altova GmbH

8 SOAP Web サービスの実装

MapForce Enterprise エディションを使用して、既存のW eb サービス記述言語（W SDL）ファイルをベースにしたSOAP W eb サービスを実装するJava またはC# プログラムコードを生成することができます。以下のようにデータをW SDL オペレーションにマップすることができます:

- フラットファイル、XML、XBRL、EDI、Microsoft Excel、とデータベースを含むMapForce によりサポートされている、全てのデータベースへのオW SDL ペレーションからの入力。
- MapForce によりサポートされる データソースからW SDL オペレーションへの出力。

MapForce はW SDL 1.1 とW SDL 2.0 をサポートします（追加情報と制限に関しては、SOAP/W SDL サポートメモとJava SOAP W eb サービスの指定を参照してください）。

メモ SOAP W eb サービスを既存のW SDL ファイルから開発する代わりに、MapForce マッピングをAPI またはコマンドライン呼び出しを介して実行することのできるMapForce Server にデプロイすることができます。更に、FlowForce Server を使用すると、FlowForce Server が作動するマシンでマッピングをジェネリックな（SOAP ではない）W eb サービスとして更に公開することができます。BUILT-IN 変換言語のためにデザインされたマッピングは、MapForce Server またはFlowForce Server へのデプロイの必要条件を満たします。FlowForce Server はMapForce マッピングをW eb サービスに変換するだけでなく、計画的に、またはオンデマンドでジョブ実行することも可能な専用のサーバーソリューションです。詳細に関しては、FlowForce Server にマッピングをデプロイするを参照してください。

必要条件

MapForce でW eb サービスを作成するには、W eb サービスのW SDL ファイルが必要です。W SDL ファイルをデザインし、XMLSpy でSOAP リクエストをテストすることも可能です。更に、 W eb サービスサーバーを実行するためにプラットフォーム特定のソフトウェアが必要になる場合があることにご注意してください。例:

Java

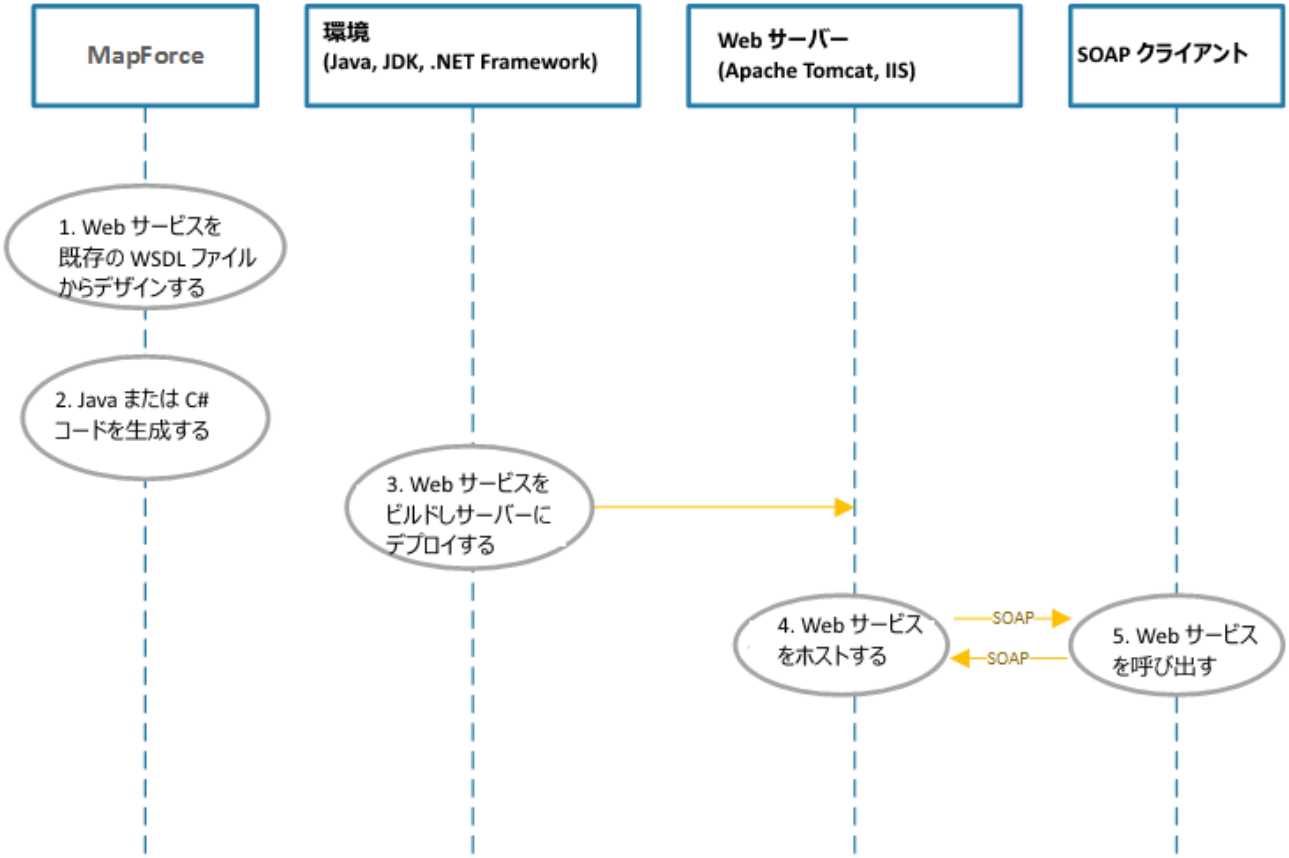
- （OpenJDK を含む）Java SE JDK 7、8、または11
- Apache Tomcat: http://tomcat.apache.org
- Apache Axis2: http://ws.apache.org/axis2/、Tomcat 内のSOAP フレームワーク
- Apache Ant: http://ant.apache.org/

C#

- Microsoft Visual Studio 2005 または以降
- Microsoft Internet Information Services（IIS）verson 5.0 または以降。

作動のしくみ

次のダイアグラムは、W eb サービスの実装のプロセス全体を、既存のW SDL ファイルからSOAP クライアントアプリケーションの呼び出しまでの段階を表示しています。最初の2 つのステップのみがMapForce 内で行われます。ステップ3 と4 は機関により使用される第三者ツールと環境により異なります。



上記のプロセスは、以下の用に作動します:

ステップ #	説明
1. 既存のW SDL ファイルからW eb サービスをデザインする	W SDL ファイルの準備ができると、新規 MapForce W eb サービスプロジェクトを開始します。MapForce W eb サービスプロジェクトには特定のW SDL オペレーション、またはW eb サービスにアクセスする定義済みの構造があります。MapForce W eb サービスプロジェクトを開始すると、任意でW SDL ファイル内で検索された全てのW SDL オペレーションために自動的にマッピングファイルを生成するようにMapForce に命令することができます。代わりに、W SDL サービスとW SDL Endpoints をMapForce W eb サービスプロジェクトに含まれるように個別に選択することもできます。y MapForce の視覚的なユーザーインターフェイスでは、W SDL オペレーションはマッピングに相対し、オペレーションの出力と入力パラメーターはマッピング内にコンポーネントとして表示されます。 W SDL オペレーションの入力または出力を含むマッピングと作業中に、他のマッピングのために行うように実行の結果をプレビューすることができます。これを行うには、W eb サービスのW SDL 文法に準するサンプルSOAP リクエストファイル(XML 書式) が必要になります(XMLSpy を使用してこのようなインスタンスファイルを生成することができます)。
2. Java またはC# コードの生成	MapForce 内でW eb サービスプロジェクトの準備が整うと、(テストのために)個々のマッピング、またはプロジェクト全体のためにC# またはJava コードを生成することができます。後者の場合、MapForce は選択される言語(C# またはJava)でW eb サービスサーバーを作成します。

(C) 2015-2021 Altova GmbH

ステップ #	説明
3. ビルドし W eb サービスをサーバーにデプロイする	このステップでは、生成された C# または Java コード（MapForce 外）でコンパイルし、カスタム W eb サーバーにデプロイします。Your W eb サービスは SOAP サービスをホストし SOAP クライアントの呼び出しに応答可能である必要があります。上記の必要条件のセクションを参照してください。
4. W eb サービスをホストする	コンパイルされた W eb サービスは選択するサーバーによりホストすることができます。例: • Apache Tomcat • Internet Information Services (IIS)
5. W eb サービスを呼び出す	W eb サービスのクライアントは SOAP 呼び出しを生成し、レスポンスを処理することのできる XMLSpy または MapForce を含むアプリケーションが含まれます。

ステップ 1 に関する詳細は、SOAP W eb サービスプロジェクトの作成を参照してください。他のステップに関しては、例: SOAP W eb サービスの生成（C#）と例: SOAP W eb サービスの生成（Java）を参照してください。

8.1 SOAP/WSDL サポート情報

MapForce 内でのW SDL へのサポートの詳細をまとめたテーブルです

W SDL サポート	バージョン1.1 http://www.w3.org/TR/wsdl のW 3C ノート バージョン2.0 http://www.w3.org/TR/wsdl20/ のW 3C 勧告
W SDL 型システム	XML スキーマ2001
SOAP サポート	バージョン1.1: http://www.w3.org/TR/2000/NOTE-SOAP-20000508/ バージョン1.2: http://www.w3.org/TR/soap12-part0/
プロトコル	SOAP オーバーHTTP（HTTP POST、HTTP GET プロトコルはサポートされていません）。
C#	SOAPAction は、C# 内の各オペレーションに対して異なるものである必要があります。
バインディング	同名の複数オペレーションは、現在サポートされていません(W SDL 1.1 仕様のW SDL 2.5）。
スタイル/使用	• document/literal: サポート • RPC/literal: C# にてサポート • RPC/encoded: 限定されたサポート • 1 つのW eb サービス（Java）またはオペレーション（C#）に対して1 つのstyle/use が現在サポートされています。
SOAP ヘッダー	使用しているプラットフォームに依存
SOAP encodingStyle	**"http://schemas.xmlsoap.org/soap/encoding/"** がsoap:Body 全体に対して指定され、それ以外のエンコーディングスタイルはサポートされません。 encodingStyle 属性はメッセージ内で無視されます（ SOAP 1.1 仕様のSOAP 4.1.1）。
参照	• 外部リソースへの参照は現在サポートされていません（SOAP 1.1 仕様のセクション5.4.1 ）。 • 参照は独立した要素に対してのみサポートされます。
SOAP-ENC:Array	リニアアクセスがサポートされます。部分アレイと疎な配列は現在サポートされていません。
カスタム SOAP 強化:	未サポート
スキーマ内にあるデフォルトまたな固定値:	未サポート
SOAP メッセージ以外の検証	検証無しで、フレームワークへ受け渡し。
名前空間	非名前空間のエントリーはW SDL にて不正となるため、サポートされていません(W SDL ならびにXML 1.0)

WSDL 1.1

ポートタイプ

<portType> 要素によりW eb サービスのインターフェースが定義されます:

(C) 2015-2021 Altova GmbH

- 処理するオペレーション
- 各オペレーションに入力または出力として含まれるメッセージ

タイプ

<types> 要素により、W eb サービスにより使用されるデータ型が定義されます。MapForce ではW SDL ファイルにて最も広く使用されている種類の、W SDL 内における XML スキーマがサポートされています。MapForce ではこれらの要素(データ型)を(メッセージ)コンポーネント内のアイテムとして表示して、その他のアイテム/制約へ直接マッピングを行います。

メッセージ

<message> 要素により、各メッセージのパートとオペレーションの入力ならびに出力パラメーターのデータ要素が定義されます。これらはクライアントとサーバー間でやり取りされるメッセージとなります。メッセージには、input、output、fault の3種類あります。
MapForce では、各メッセージは(例えばgetPersonSoapIn という名前の)コンポーネントとして表示されており、他のアイテムへのマッピングを行うことができます。メッセージは1 つ以上のメッセージ パーツからなります。

MapForce にてドキュメントとリテラルを混合して使う場合、**message** / part 要素は**type** ではなく 名前空間を参照する必要があります。"element" 属性により、スキーマ内で定義されたグローバル要素を参照します(ns2:Vendor):

```
<message name="processRequest">
      <part name="inputData" element="ns2:Vendor"/>
</message>
```

以下の場合は、スキーマ内部のtype を参照します:

```
<message name="processRequest">
      <part name="inputData" type="ns2:VendorType"/>
</message>
```

オペレーション

オペレーションは、入力ならびに出力パラメーターとしてメッセージを使用します。オペレーションには以下の要素を含めることができます:

- 1 つのInput メッセージ
- 0 、または1 つのOutput メッセージ
- 0 、または1 つのFault メッセージ

Input メッセージはソースコンポーネントとしてのみ使用することができます。出力ならびにFault メッセージはターゲットコンポーネントとしてのみ使用することができます。

WSDL 2.0

W SDL 2.0 は根底部分でW SDL 1.1 と異なっており、主な違いを以下に記します:

- PortTypes がInterfaces に改名されました
- メッセージとパーツがtypes 要素のXML スキーマ型システムにより定義されます
- Ports はエンドポイント に改名されました
- W SDL 2.0 Operation 入力ならびに出力がXML スキーマにより定義されます

W SDL コンポーネントの(MapForce 内における)コンポーネント設定ダイアログボックスにより、W SDL 1.1 Ports とW SDL 2.0 エンドポイント の両方が表示されます。

8.2 WSDL ファイルから SOAP Webサービスプロジェクトを作成

この例は、MapForce 内でのSOAP W eb サービスの作成方法について説明しています。この例内で作成されているマッピングプロジェクトは次のパスでみつけることができます: <マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥Tutorial¥Query Person database.mfp。この例のW eb サービスは次のゴールがあります:

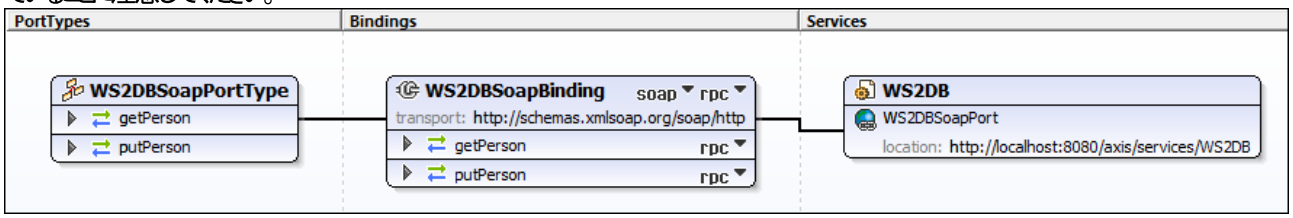
- SOAP リクエストを使用してサーバー上のMicrosoft Access データベースから検索条件を満たす個人のレコードのセットを抽出します(クエリはクライアントにランタイムに入力され、サーバーに送信されます。サーバーは検索結果とともにXML レスポンスを返します)。
- Microsoft Access データベースにSOAP リクエストを使用して新規のレコードを挿入します(SOAP リクエストにより個人の詳細は提供されます。成功すると、サーバーはデータベースへの個人の詳細の書き込みの成功の通知を返します)。

次に注意してください:

- MapForce 内でのW eb サービスの作成はターゲットプログラム言語に依存しません。Java またはC# W eb サービスを生成する場合でも同じです。W eb サービスをコンパイルしW eb サーバーにデプロイする際に違いが発生します。この点については後に説明されています。
- マッピング出力のプレビューに必要なサンプルSOAP リクエストを生成するには、この例ではXMLSpy Enterprise Edition が使用されています。このMapForce プロジェクトのソースW SDL ファイルはXMLSpy を使用して作成されています(検証することもできます)。MapForce 内でXMLSpy を使用することなくW eb サービスをデザインすることができます。W SDL ファイルからSOAP リクエストを作成することのできるSOAP クライアントが必要になります。

W eb サービスプロジェクトの作成

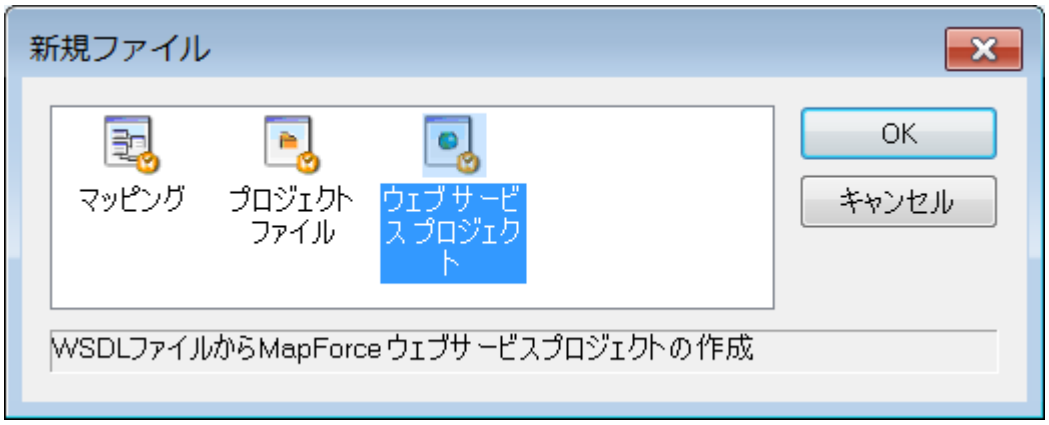
上記のように、MapForce を使用してSOAP W eb サービスを作成するためにW SDL ファイルが必要になります。この例では、XMLSpy Enterprise エディションを使用して作成されたW SDL ファイルが使用されます。下のイメージではXMLSpy のW SDL ビューで表示されるW SDL 構造が示されています。上記の目的に対応する2 つのオペレーション(`getPerson` と`putPerson`) が定義されていることに注意してください。



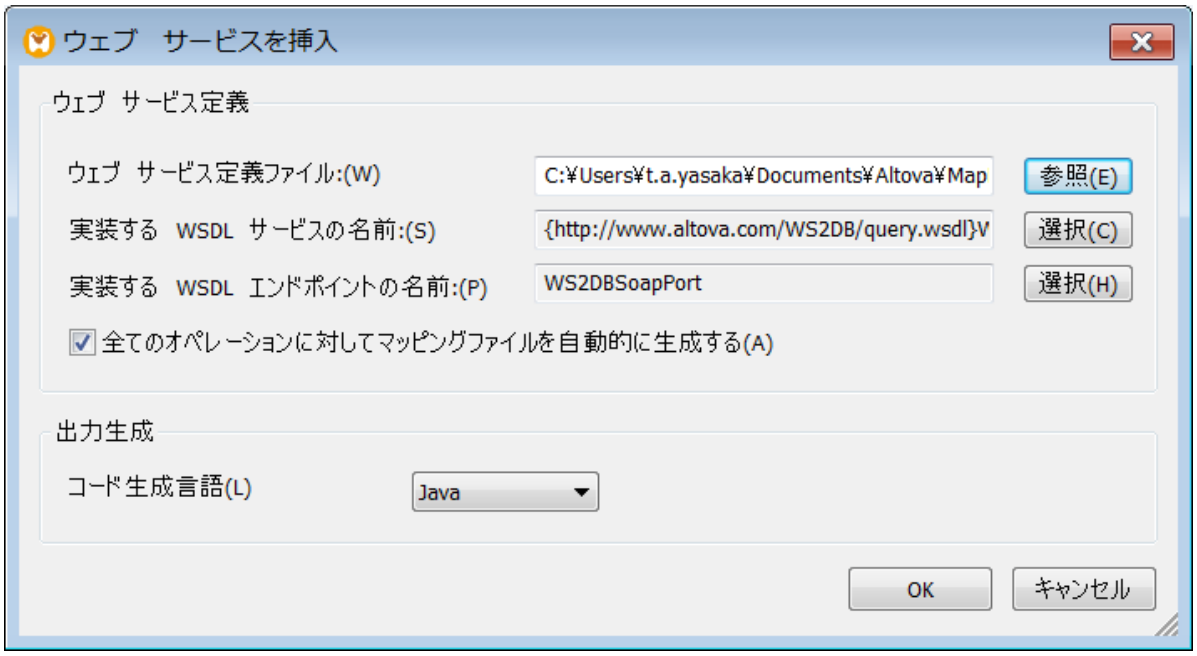
メモ W SDL ファイルはMapForce W eb サービスプロジェクトが作成される前に有効である必要があります。

W eb サービスからW eb サービスプロジェクトを作成するには、下記のステップに従います:

1. 「ファイル」メニューから「新規作成」をクリックし、「W eb サービスプロジェクト」を選択します。

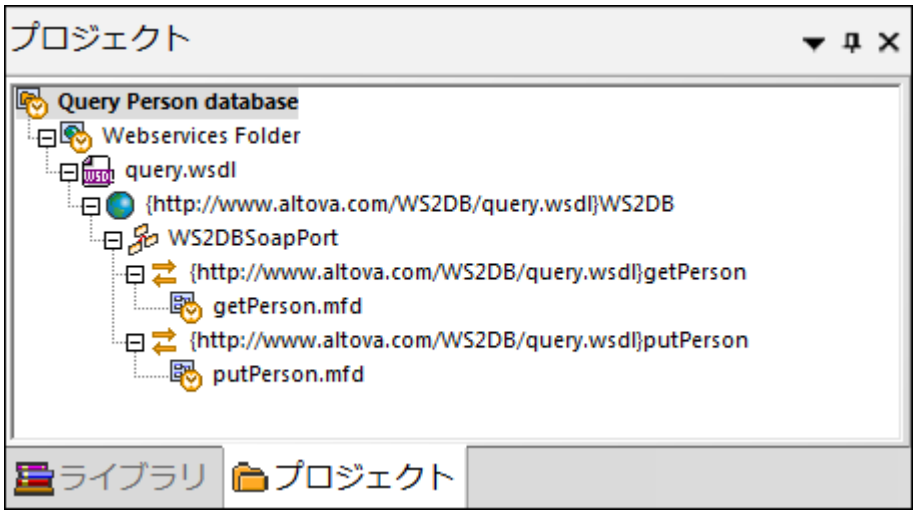


2. フォルダー<マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥Tutorial¥内で使用することのできる **query**.W SDL ファイルを参照します。W SDL ファイルを選択すると、MapForce は自動的に残りのフィールドを入力します。



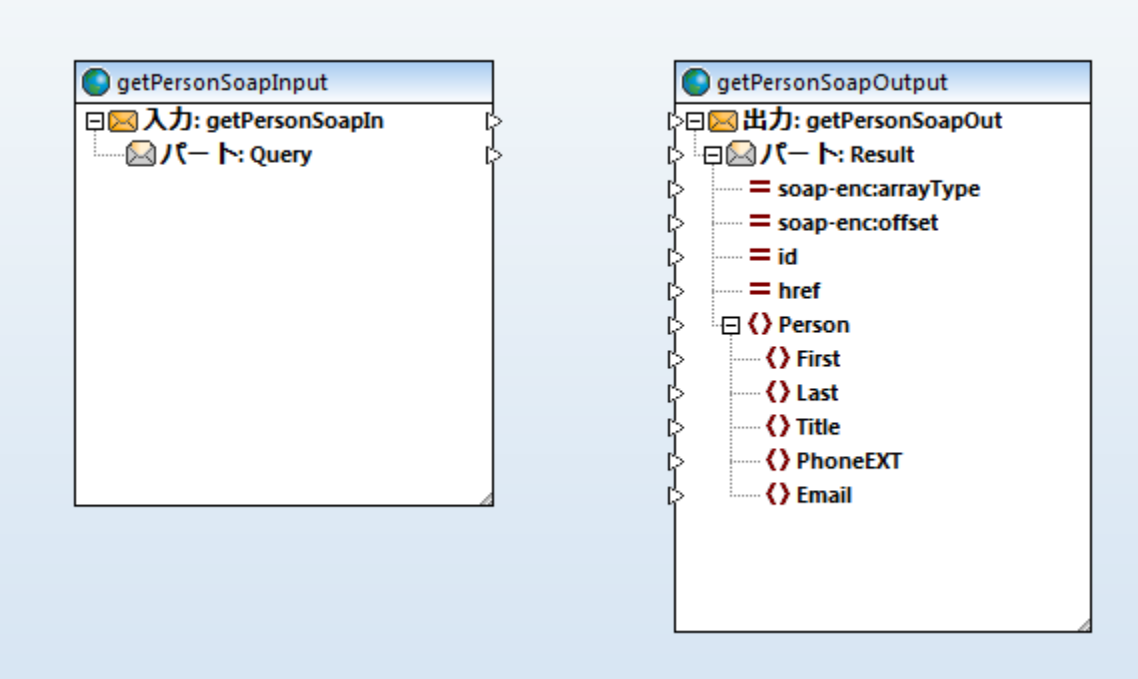
3. 「OK」をクリックします。
4. プロンプトされると、新規のW SDL プロジェクトの名前を入力し、「保存」をクリックします。

プロジェクトウィンドウはプロジェクトとW SDL の名前、およびW SDL ファイル内で定義されているオペレーションを表示しています。2つのオペレーションは**getPerson** と**putPerson** です。



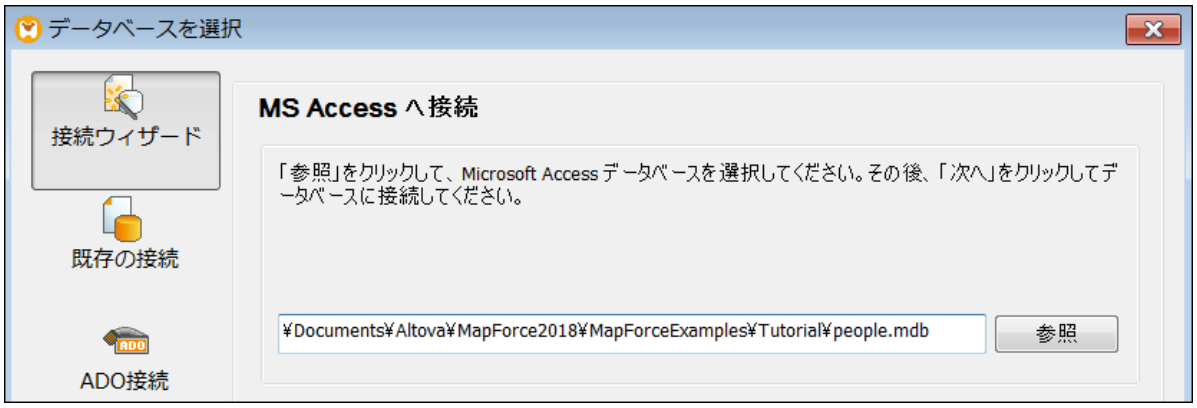
「getPerson」オペレーションのデザイン

プロジェクトウィンドウ内の**getPerson.mfd** ファイルをダブルクリックし、メインペイン内にコンテンツをロードします。**getPersonSoapIn** コンポーネントはW eb サービスのデータベースをクエリするために使用される クエリ（アイテム）を含んでいます。**getPersonSoapOut** コンポーネントは、W SDL ファイル内で定義されている `Person` 構造を含んでいます。

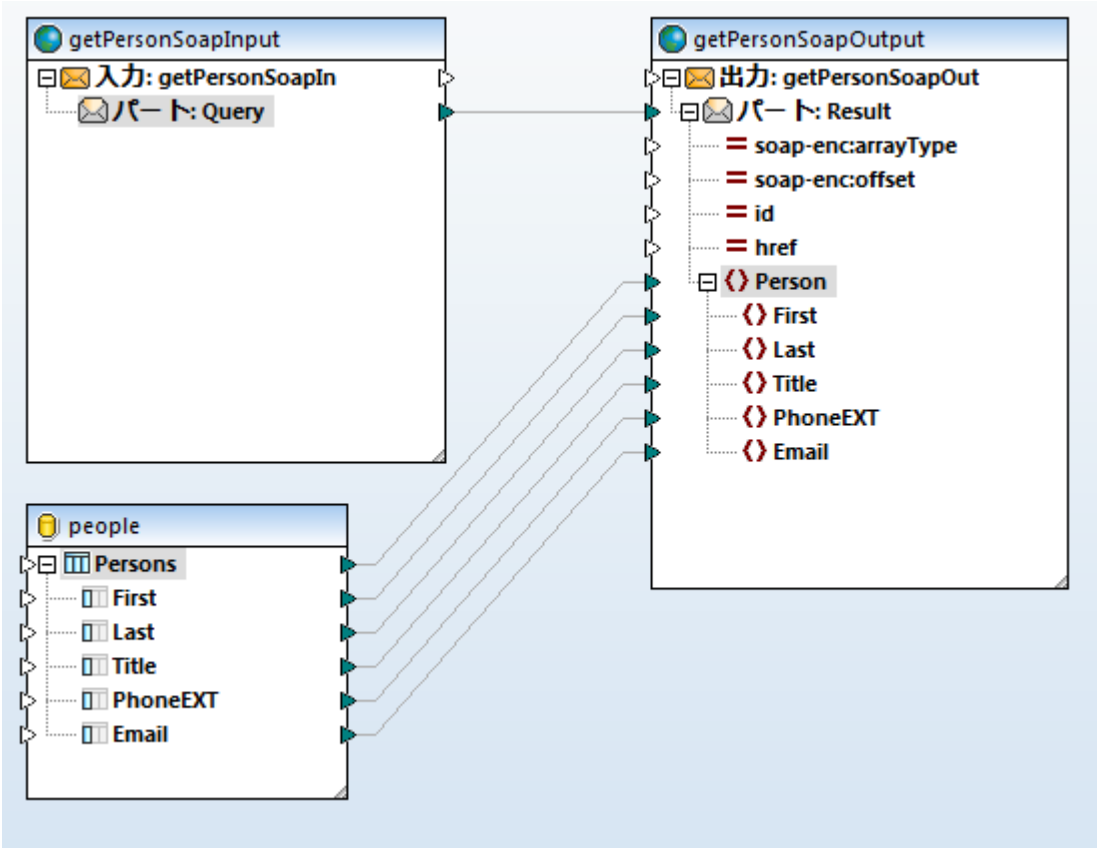


「挿入」メニューから「データベース」をクリックします。次のAccess データベースをマッピングエリアに追加します: **<マイドキュメント>\Altova\MapForce2021\MapForceExamples\Tutorial\people.mdb**. データベースには `getPersonSoapOut` コンポーネント内のフィールドに一致するフィールドを持つ1つのテーブル `Persons` のみが含まれており、データベースをマッピングコンポーネントとして追加することに関する詳細は、データベースとMapForce を参照してください。

(C) 2015-2021 Altova GmbH



下に示されるようにコンポーネント間に接続を描きます:



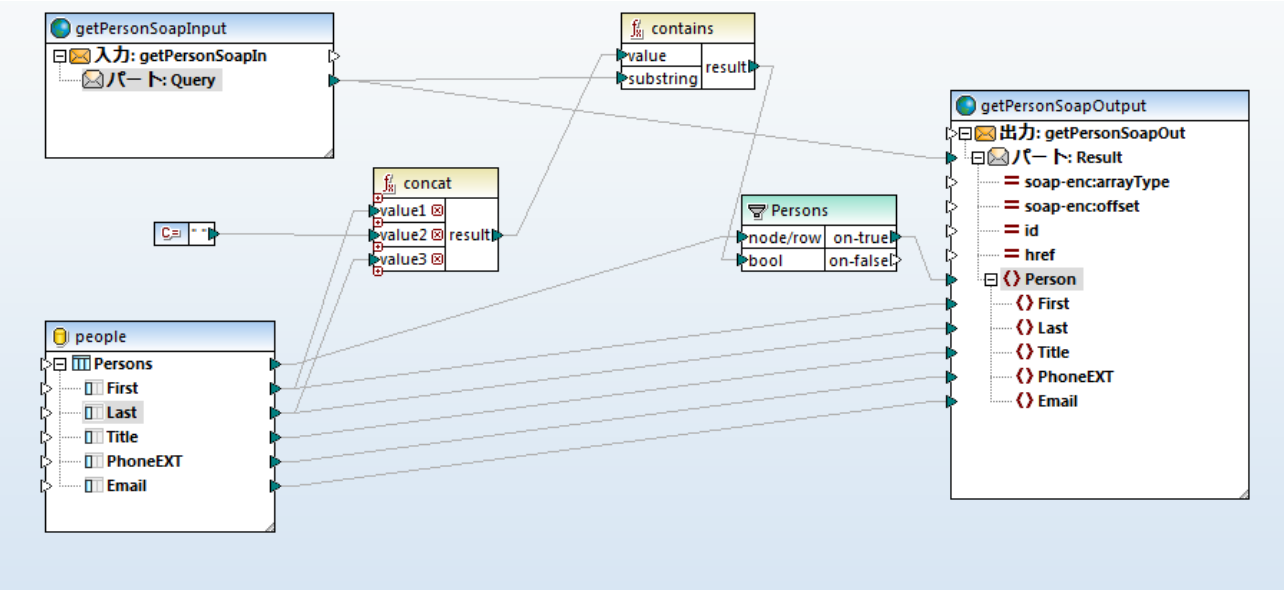
上のマッピングデザインはまだ完成していませんが、ランタイムに何が起こっているかが示されています。**part:Query** と **part:Result** の間の接続はW eb サービスの呼び出しを表しています。`getPersonSoapIn` コンポーネントの**part:Query** アイテムは、クエリプレースホルダ(クエリ文字列がSOAP クライアント内に入力される箇所)です。データベーステーブルと**getPersonSoapOutput** コンポーネントの **Person** アイテム間の接続は、対応するデータベースフィールドからの値を抽出し対応するターゲットアイテムにパスします。次のように出力をプレビューすることができます:

1. **getPersonSoapInput** コンポーネントを右クリックします。
2. 「参照」をクリックして、次のサンプルリクエストを選択します: **<マイドキュメント>\Altova\MapForce2021\MapForceExamples\Tutorial\getPersonRequest.xml**.
3. 「OK」をクリックします。

4. **出力** タブをクリックします。

この時点では、マッピングはデータベーステーブルから全てのレコードを抽出します。これはとても役にたちます。このため、フィルターを追加し、リクエスト文字列に一致するアイテムのみが抽出される必要があります。

1. ライブラリウィンドウからドラッグして、`concat` 関数を追加します。この関数は、各個人の`First` と`Last` 名を連結するために使用されます。
2. 「挿入」メニューから「定数」をクリックし1つの空白文字を含む定数を追加します。定数はデータベースから抽出された氏名の間に空白文字を与えます。
3. ライブラリウィンドウからドラッグして `contains` 関数を追加します。
4. ソース**Persons** テーブルと**Person** ターゲットノード間の接続を右クリックします。コンテキストメニューから**Insert Filter: Nodes/Rows** を選択します。
5. 下に示されるようにコンポーネント間に接続を描きます。



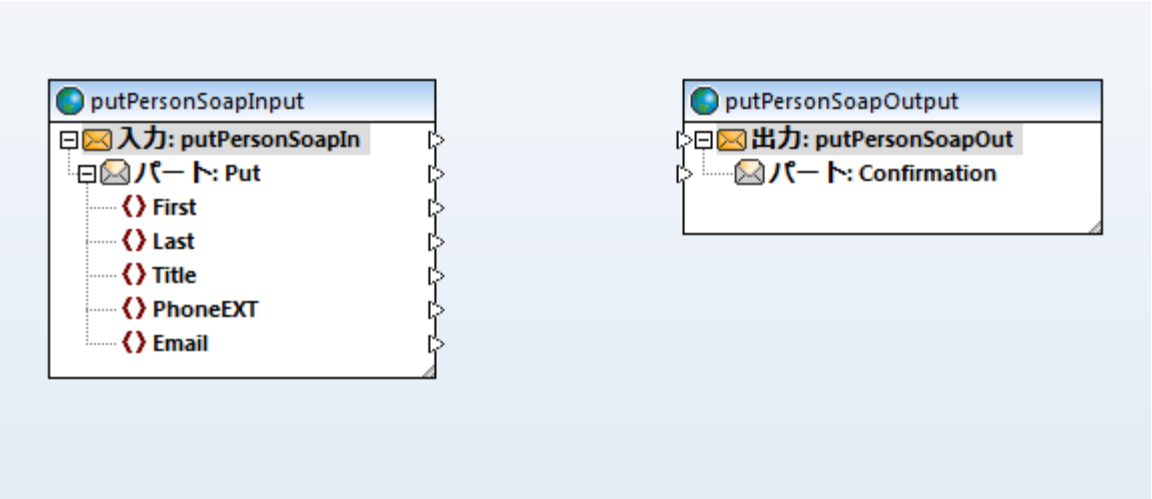
"getPerson" マッピング

上記のように、クエリ文字列が `concat` 関数により名前が与えられ低増す。名前は名の後にスペース文字と氏がリストされているように構成されています)。データベース内で名前の全部または一部が一致する場合 `contains` 関数は**true** を返します。一致が存在する場合、一致する個人の情報がレスポンスメッセージ内に含まれます。ですから、この例の場合、検索の値 "Ro" は次のレコードを返します: "Martin **Ro**pe"、"**Ro**nald Superstring"、"**Ro**bert Darkmatter"、および"**Ro**ger Gravity"。

「putPerson」オペレーションのデザイン

メインペイン内にコンテンツをロードするためにプロジェクトウィンドウ内の**putPerson.mfd** ファイルをダブルクリックします。**putPersonSoapInput** コンポーネントはリクエスト構造を提供します(Web サービスを介してアイテムはデータベースに挿入されます)。**putPersonSoapOutput** コンポーネントはWeb サービス呼び出しの結果を示しています。Web サービス呼び出し元への呼び出しの結果を表示する確認の部分が含まれています。

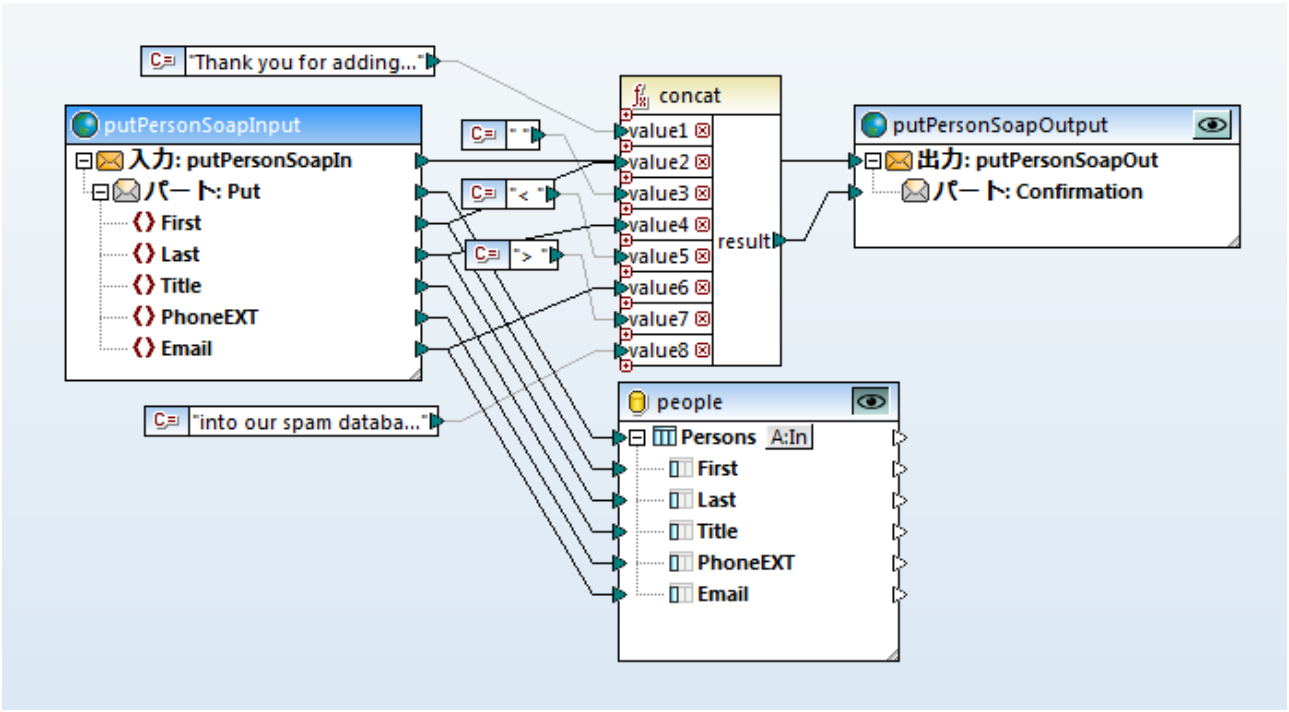
(C) 2015-2021 Altova GmbH



以下をマッピングに追加するために上記の説明と同様のオペレーションをおこないます:

- **people.mdb** Access データベース（W eb サービス呼び出し元により提供されるデータが挿入されます）。
- `concat` 関数と5つの定数（W eb サービスの呼び出し元を表示するための情報文字列をビルドするために使用されます）。

下に示されるように接続を描きます:



"putPerson" マッピング

上のマッピングでは、**part: Put** とデータベース間の接続によりランタイムにデータベースが更新されます。ターゲット側で **part: Confirmation** レスポンスはリクエスト、および、定数により与えられたテキストから多種のフィールドを連結することにより取得されます。データベースを更新するマッピングに関する詳細は、データをデータベースにマッピングするを参照してください。

マッピングの出力のプレビューの方法:

1. **getPersonSoapInput** コンポーネントを右クリックします。
2. 「参照」をクリックして次のサンプルリクエストを選択します: <**マイドキュメント >\Altova\MapForce2021\MapForceExamples\Tutorial\putPersonRequest.xml**。
3. 「出力」タブをクリックします。

マッピングをプレビューする際、MapForce はデータベースを更新し、出力ウィンドウに表示するために必要な INSERT クエリを生成しますが実際の更新は行いません。データベースに対してクエリを実行するには、「**出力 | SQL-Script の実行**」を選択します。

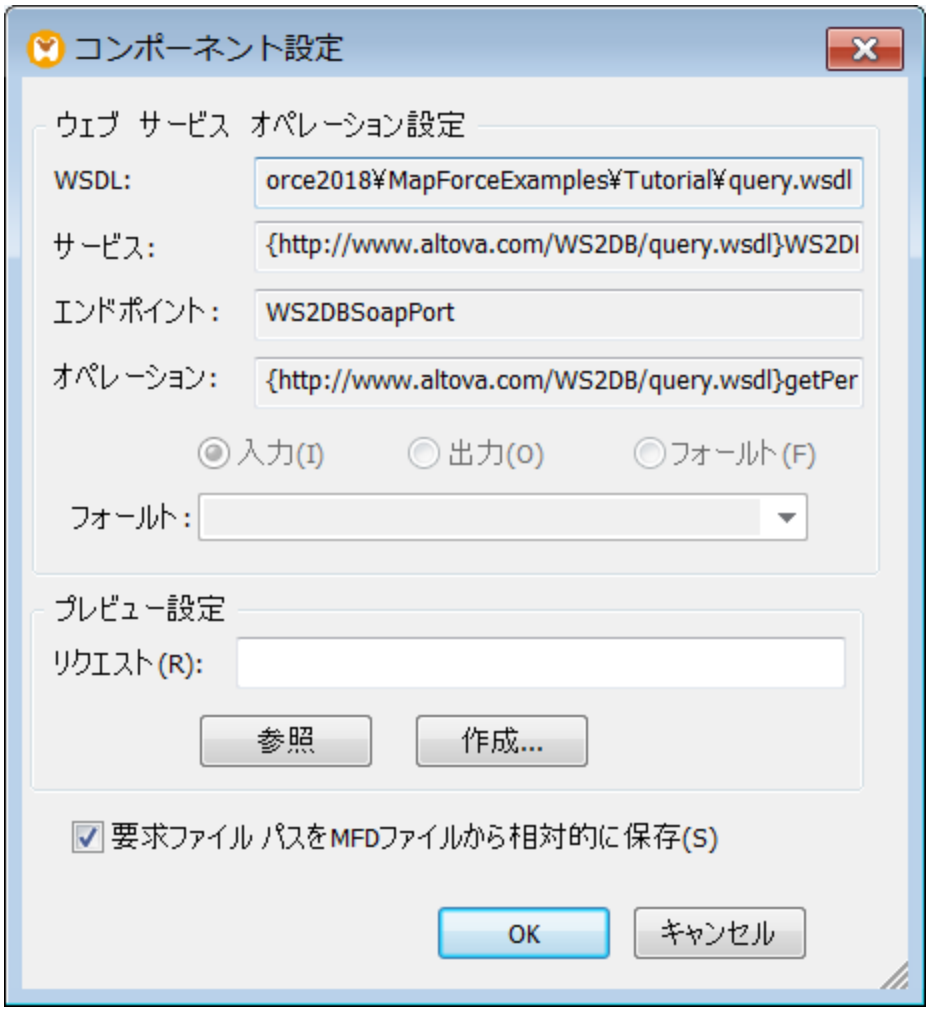
```
1   /*
2   The following SQL statements are only for preview and may not be executed in another SQL query tool!
3   To execute these statements use function "Run SQL-script" from menu "Output".
4   Connect to database using the following connection-string:
5   Data Source=C:\Users\altova\Documents\Altova\MapForce2018\MapForceExamples\Tutorial\people.mdb;Provider=Microsoft.Jet.OLEDB.4.0
6   */
7
8   INSERT INTO [Persons] ([First], [Last], [Title], [PhoneEXT], [Email]) VALUES ('John', 'Doe', 'Mr', '123', 'john.doe@example.org')
9
```

他方、コンパイルされた Web サービスがサーバーにデプロイされクライアントにより呼び出されると、データベースはすぐに更新され、効果が反映されます。次も参照してください: データベースを変更するマッピングの実行.

マッピング出力のプレビュー

WSDL ファイルから作成されるマッピングは実際の Web サーバーにデプロイされる前に MapForce 内でプレビューすることができます。このようなマッピングをプレビューするには、WSDL ファイルに準じたサンプル SOAP リクエスト（XML ファイル）が必要になります。Web サービスプロジェクト内の各オペレーション（マッピング）のために XMLSpy または SOAP クライアントを使用してサンプルリクエストを作成することができます。XMLSpy を使用する場合、メニューコマンド「**SOAP | 新規 SOAP リクエストを作成する**」を実行してリクエストを生成します。XMLSpy が MapForce と共にインストールされている場合、MapForce から直接サンプルリクエストを生成することができます。例:

1. マッピング上の **getPersonSoapInput** コンポーネントをクリックし、コンテキストメニューから **Properties** を選択します。



2. 「作成」をクリックします。
3. プロンプトされると、ディスク上の場所にXML ファイルを保存します。

リクエストファイルは入力パラメーターの代わりにダミーの入力値(例えば、「文字列) を使用して生成されます。リクエストファイルを使用する前に、ダミーの値を使用する実際のリクエストの値と置き換えてください。このプロジェクトのために、2つのサンプルリクエストを使用することができます:

1. <マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥Tutorial¥getPersonRequest.xml. **getPerson** マッピングの出力をプレビューするためにこのファイルを使用します。デフォルトでは、このリクエストファイルは、W eb サービスの入力として「Ro」の値を提供します。テキスト"Ro" が氏または名に含まれる全ての個人のレコードがデータベースから抽出されます。
2. <マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥Tutorial¥putPersonRequest.xml. **putPerson** マッピングの出力をプレビューするためにこのファイルを使用します。デフォルトでは、このリクエストファイルは、W eb サービスへの入力として架空の個人の詳細を提供します。

2つのマッピングをプレビュー中に異なる出力を取得するには、上のファイルを編集し、テストする入力の値が含まれるようにします。

次のステップ

W eb サービス(Java または C#) を実装するコードを生成する準備ができると、次のサンプル内で説明されているとおりサーバーにデプロイします:

- 例: SOAP Web サービスの生成 (C#)
- 例: SOAP Web サービスの生成 (Java)

(C) 2015-2021 Altova GmbH

8.3 SOAP Web サービス フォールトの定義

WSDL ファイルはフォールト メッセージを含むオペレーションとメッセージ属性のために `fault` 要素を含むことができます。同様にSOAP Web サービスを MapForce 内でデザインする際、「フォールト」コンポーネントをマッピングに追加することができます。Web サービスが特定の条件を満たすと例外がスローされます。

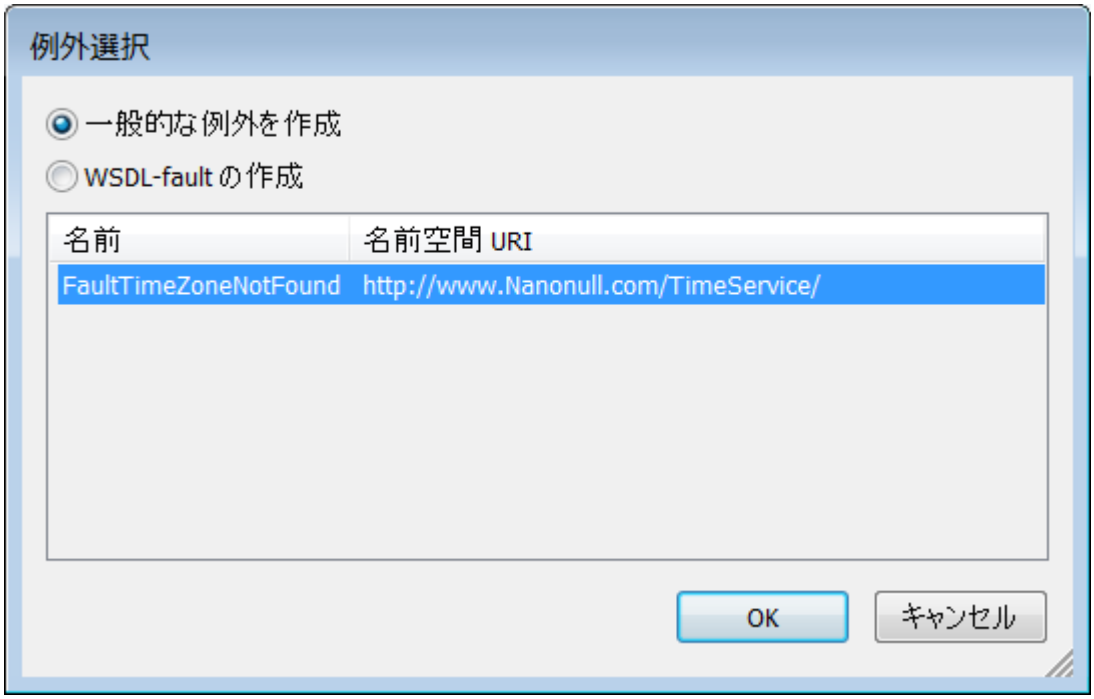
> マッピングにフォールトコンポーネントを挿入するには、`fault` 要素がWSDL ファイル内に存在する必要があります。

マッピング内の例外の条件を定義することができ、エラーメッセージのテキストを設定することができます。条件が満たされると、Web サービスは以下のようにエラーを表示します:

- 実際の環境では(Web サービスがサーバーにデプロイされた後の実行時に) Web サービスは失敗し、カスタム定義 メッセージを呼び出し元に返します。
- (デザイン時に) MapForce 内で、マッピングの実行が中止され、「メッセージ」ウィンドウ内でメッセージを確認することができます。

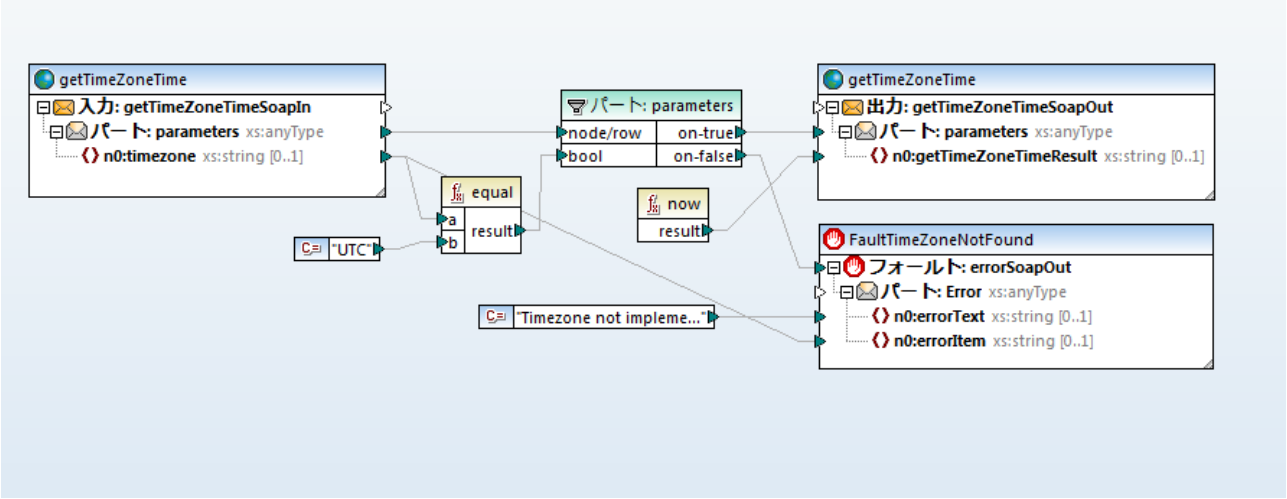
MapForce マッピングにフォールトコンポーネントを追加する方法:

1. 「挿入」メニューから「例外」をクリックします。(または、「例外」ツールバーボタンをクリックします)。
2. 「WSDL-fault の作成」を選択します



メモ 「WSDL-fault の作成」オプションが無効化されている場合、「オペレーションにエラーはありません」が表示されます。これはWSDL ファイル内にフォールトが存在せず、結果 MapForce に追加できないことを意味します。

Web サービスフォールトを含むマッピングの例に関しては、次のサンプル MapForce プロジェクトを開いてください: **<マイドキュメント>\Altova\MapForce2021\MapForceExamples\TimeService\TimeService.mfp**。マッピング(オペレーション) **getTimeZoneTime.mfd** をダブルクリックしてください。



上のマッピングでは、**n0:timezone** がUTC に等しくない場合、例外がトリガーされます。

- **equal** 関数はフィルターにパスされたブールの値の結果を使用してタイムゾーンがUTC と等しいかチェックします。
- 条件がfalse の場合、例えば、UTC とは異なる値の場合、フィルターの**on-false** パラメーターがフォールト **:errorSoapOut** 例外 を有効化し、マッピングプロセスは中断されます（例外をon-true パラメーターに接続することもできます）。
- SoapFault メッセージにより2 つのセットのエラーテキストが提供されます。

サンプル内では、フィルターの配置が重要です:

- フィルターコンポーネントの両方のパラメーター**on-true** と**on-false** がマップされる必要があります。パラメーターの1 つがフォールトコンポーネントにマップされる必要があり、フィルターされたソースデータを受け取るターゲットコンポーネントにもう1 つが接続される必要があります。それ以外の場合、フォールトコンポーネントはトリガーされません。
- 例外とターゲットコンポーネントはフィルターコンポーネントに直接接続される必要があります。関数または他のコンポーネントはフィルターと例外、またはコンポーネントの間に配置されない場合があります。

(C) 2015-2021 Altova GmbH

8.4 Java SOAP Web サービスの指定

MapForce ではW eb サービスを作成するために必要な全てのコードならびにスクリプトが生成されます(コード ジェネレータを参照)。通常のコード生成処理と唯一異なる点は、生成されたコードをAxis2 (Tomcat) サーバーへデプロイする必要があるということです。

Java コードを生成する:

1. <マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥Tutorial¥フォルダーから前のセクションで使用した**Query Person database.mfp** プロジェクトを開きます。上記のようなプロジェクトの作成に関しては、W SDL ファイルからW eb サービスプロジェクトを作成するを参照してください。
2. **プロジェクト** メニューから「プロジェクト | コードの生成 | Java」を選択します。コード生成が完了すると、Altova ジェネリッククラスとW eb サービス プロジェクトの実際のクラスなど **com** ディレクトリを含む、複数のフォルダーとファイルがターゲット ディレクトリ内に作成されます。
3. 生成されたJava コードをビルドします(MapForce により生成された**build.xml** ファイルをApache Ant に提供します)。この結果、Axis2 のデプロイに対応したAxis Archive (*.aar) ファイルが作成されます。

生成されたプロジェクトをEclipse で開くと、"org.apache のインポートが解決されません"のようなエラーが生じる場合があります。この場合、Axis2 ライブラリがJava ビルド パスに追加されていることを確認してください。Axis2 ライブラリをEclipse 4.4.2 Java ビルド パスに追加するには、以下を行います:

1. パッケージエクスプローラー内のプロジェクトを右クリックして、「**プロパティ**」を選択します。
2. 「**Java ビルド パス**」をクリックします。
3. 「**ライブラリ**」タブで「**外部 JAR を追加**」をクリックし、Axis2 ライブラリを `<AXIS2_HOME>\lib` フォルダーから追加します。

W eb サービスのデプロイ

W eb サービスをデプロイするには以下を行います:

- Axis2 の"サービスをアップロード" W eb 管理者ページを開き、前のステップで作成された.aar ファイルをアップロードします。
- 手動でアップロードします。例: Tomcat サーバーがフォルダー `<TOMCAT_HOME>` にインストールされている場合、手動で .aar ファイルを `<TOMCAT_HOME>\webapps\axis2\WEB-INF\services` にコピーすることができます。

デプロイを解除する

`<TOMCAT_HOME>\webapps\axis2\WEB-INF\services` フォルダーから*.aar ファイルを削除します。

Axis2 の制約

AXIS2 ではrpc/encoded に対するサポートが制限されています。MapForce ではrpc/encoded W eb サービス(SOAP 1.1 ならびにSOAP 1.2)を生成することができます。これにより、オリジナルのW SDL がウェブサーバーから取得されないという制約があります。

これはつまり、http://127.0.0.1/axis2/services/W S2DB?wsdl により、利用可能なW SDL ファイルが取得されないということになります。

上にある**ドキュメント/リテラル**W eb サービスのURL により、正しいW SDL ファイルが与えられますが、それはオリジナルのファイルではありません: コメントは取り除かれ、名前空間が変更されます。それでもオリジナルのW SDL ファイルとは機能的に等価なものとなります。

AXIS2 では**RPC/encoded** はサポートされていませんが、デプロイされたJava コード(コンパイルされたコード)からW SDL を生成することができ、MapForce により生成されたコードにより**RPC/encoded** メッセージが処理されます。AXIS2 は移送のためだけに使用されます。

既知の問題: SOAP レスポンスメッセージ内の名前空間

MapForce により生成されたコードは、`javax.xml.transform.TransformerFactory` クラスをインスタンス化します。クラス実装がロードされる場合、`javax.xml.transform.TransformerFactory` システムプロパティから読み込まれる可能性があります。(詳細に関しては、このクラスの Java ドキュメント を参照してください)。

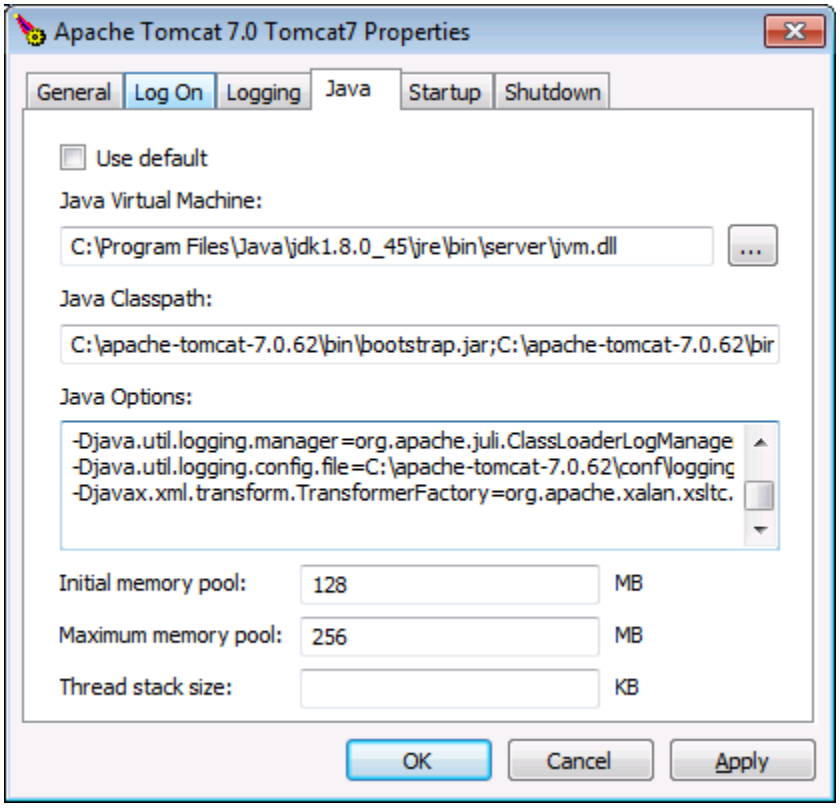
Java 環境内のシステムプロパティ `javax.xml.transform.TransformerFactory` が実装 `org.apache.xalan.transformer.TransformerIdentityImpl` を定義する場合 SOAP レスポンスメッセージが正確ではない名前空間を用いて生成される可能性があります。

この問題を回避するため、`org.apache.xalan.xsltc.trax.TransformerFactoryImpl` 実装を使用してください。これを以下のパラメーターを Java Virtual Machine (JVM仮想マシン)に追加することで行うことができます:

```
-
Djavax.xml.transform.TransformerFactory=org.apache.xalan.xsltc.trax.TransformerFactoryI
mpl
```

パラメーターを JVM に追加する命令は、使用中のオペレーティングシステムとサーバーにより異なります。以下の命令は Windows 7 上のサービスアプリケーションとして実行される Apache Tomcat 7.0 構成に適用することができます:

1. **Tomcat7w.exe** を実行します(このファイルは Tomcat 配布ルートフォルダー `CATALINA_HOME` の **\bin** サブフォルダー内にあります)。
2. **Java** タブをクリックします。
3. "Java オプション" ボックスの最後に、以下を入力します: `-Djavax.xml.transform.TransformerFactory=org.apache.xalan.xsltc.trax.TransformerFactoryImpl`



(C) 2015-2021 Altova GmbH

8.5 例: SOAP Web サービスの生成 (C#)

このサンプルはMapForce を使用してSOAP W eb サービスを生成する方法について説明しています。プログラムコード 次のパスで見つけることのできるサンプルMapForce プロジェクトからC# プログラムコードを生成します: <**マイドキュメント >\Altova\MapForce2021\MapForceExamples\Tutorial\\Query Person database.mfp**。このプロジェクトは2 つのオペレーションを公開するSOAP W eb サービス実装します。: `getPerson` と`putPerson`. このW eb サービスは個人の詳細を抽出、またはお追加するために上記のオペレーションをおこない、背景でMicrosoft Access データベースと通信します。MapForce 内でこのようなプロジェクトを作成するサンプルに関しては、次を参照してください: SOAP W eb サービスプロジェクトの作成.

生成されたC# コードをコンパイルするために、このサンプルはVisual Studio 2015 を使用します。ソリューションはASP.NET が有効化された状態でInternet Information Services (IIS) 10 にデプロイされます。

簡素化のためこのサンプルは基本のW eb サーバー構成を使用していることにご注意してください。生産環境では、W eb サーバー、または所属機関内のセキュリティポリシー、その他の要素が指定するとおりの異なるアプローチを使用しなければならない場合があります。

SOAP W eb サービスを呼び出すために、このサンプルはXMLSpy Enterprise Edition を使用しています。XMLSpy がない場合、W eb サービスをテストするために、SOAP クライアントまたはW eb サービスを呼び出すことのできるプログラムコードが必要になります。

メモ IIS 7.x を使用する場合、W indows 機能「IIS Metabase and IIS 6 configuration compatibility」を最初にインストールする必要がある場合があります。

ステップ 1: query.wsdl ファイルの準備

このサンプルW eb サービスは次のW SDL ファイルが基礎になっています: <**マイドキュメント >\Altova\MapForce2021\MapForceExamples\Tutorial\query.wsdl**. デフォルトでは、**query.wsdl** ファイルはC# とJava のためのサービスの詳細を1 つの箇所がコメントアウトされている2 つの異なる箇所で定義しています。ですから、C# コード生成前に、**query.wsdl** ファイル内の次のセクションをロケートし、コメントアウトされていることを確認してください:

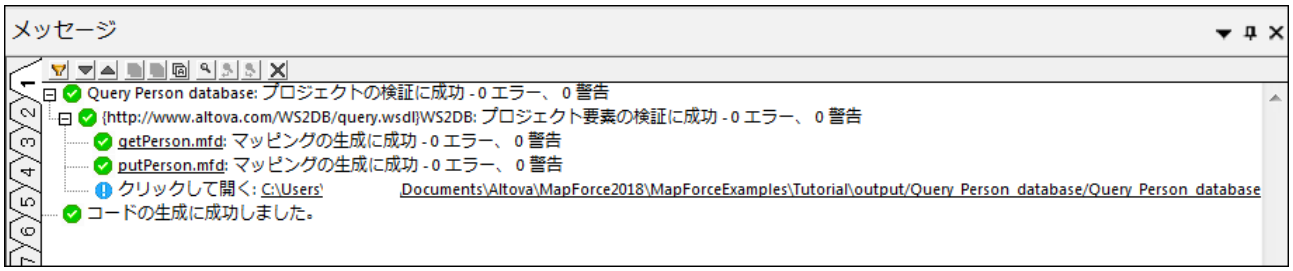
```
<service name="WS2DB">
   <port name="WS2DBSoapPort" binding="tns:WS2DBSoapBinding">
      <soap:address location="http://localhost/WS2DB/WS2DB.asmx"/>
   </port>
</service>
```

(Java にのみ適用することのできる) 次のセクション がコメントアウトされていることを確認してください:

```
<!--<service name="WS2DB">
   <port name="WS2DBSoapPort" binding="tns:WS2DBSoapBinding">
      <soap:address location="http://localhost:8080/axis/services/WS2DB"/>
   </port>
</service>-->
```

ステップ 2: C# コードの生成とビルド

MapForce 内で <**マイドキュメント**>¥Altova¥MapForce2021¥MapForceExamples¥Tutorial¥ フォルダーから「Query Person database.mfp」プロジェクトを開きます。プロジェクトウィンドウ内のプロジェクトを右クリックし、「**コードの生成 | C#**」を選択します。MapForce はプログラムコードを生成し、メッセージウィンドウ内に結果を表示します:



デフォルトでは、コードはプロジェクトのディレクトリに対応する「出力」と呼ばれるサブディレクトリに生成されます。このチュートリアルのスコープのため、デフォルトの設定がそのままにされています。出力ディレクトリを変更するには、次を参照してください:コード生成設定の設定。

Visual Studio を実行して生成されたソリューションを開きます。ソリューション内の全てのプロジェクトがロードされるまで待ってください。

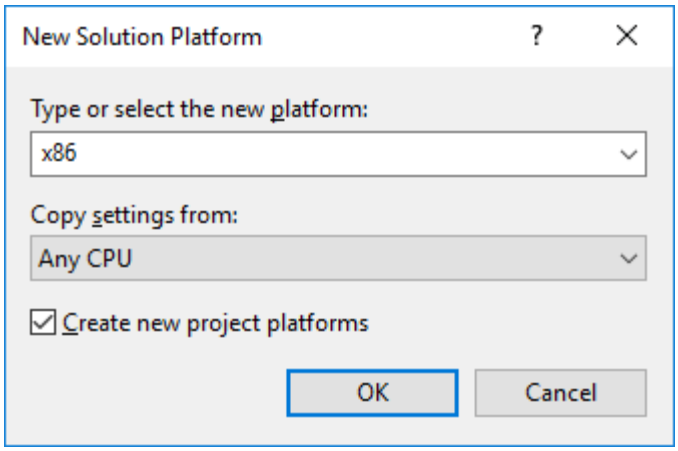
> Visual Studio を管理者として作動する必要があるかもしれません。また、このサンプルソリューションは下記のとおり 32-ビットアプリケーション としてコンパイルされる必要があります。

ソリューションを次のようにビルドすることができます:

- **Build** メニューから**Build Solution (Ctrl+Shift+B)**をクリックします。

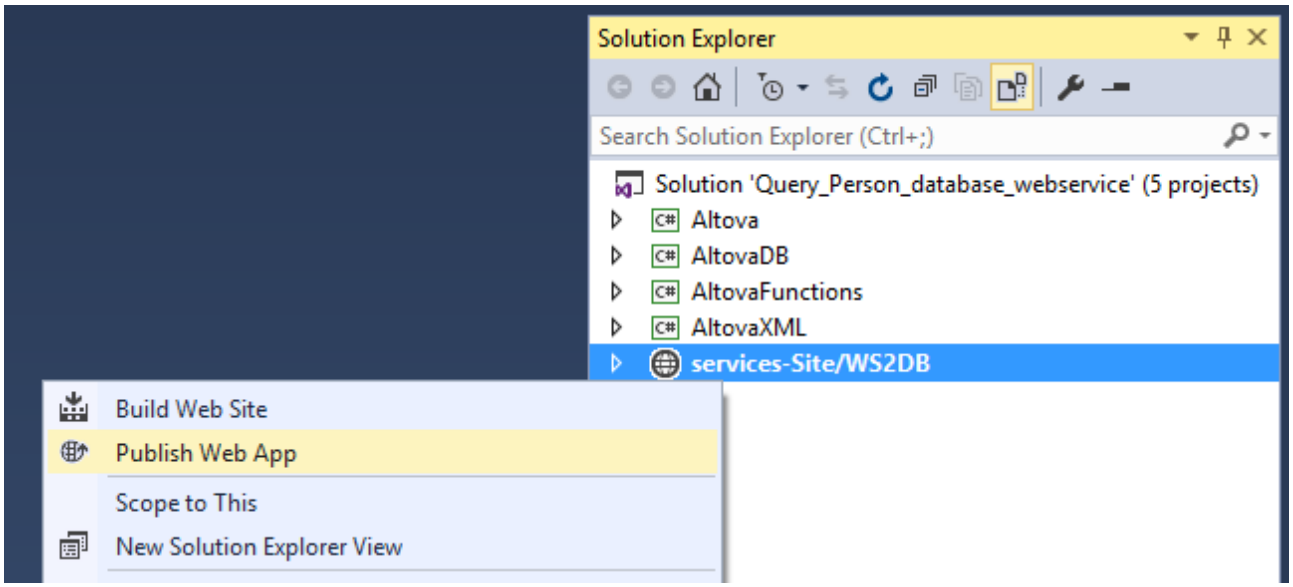
このサンプルは32-ビットアプリケーションのみで作動する Access のための ADO プロバイダーを使用して Microsoft Access データベースをクエリすることに注意してください。このため、32-ビットアプリケーションとしてソリューションを以下のようにコンパイルする必要があります:

- 「Build」メニューから「Configuration Manager」をクリックします。x86 のために新規のソリューションプラットフォームを作成し、再度ビルドします。

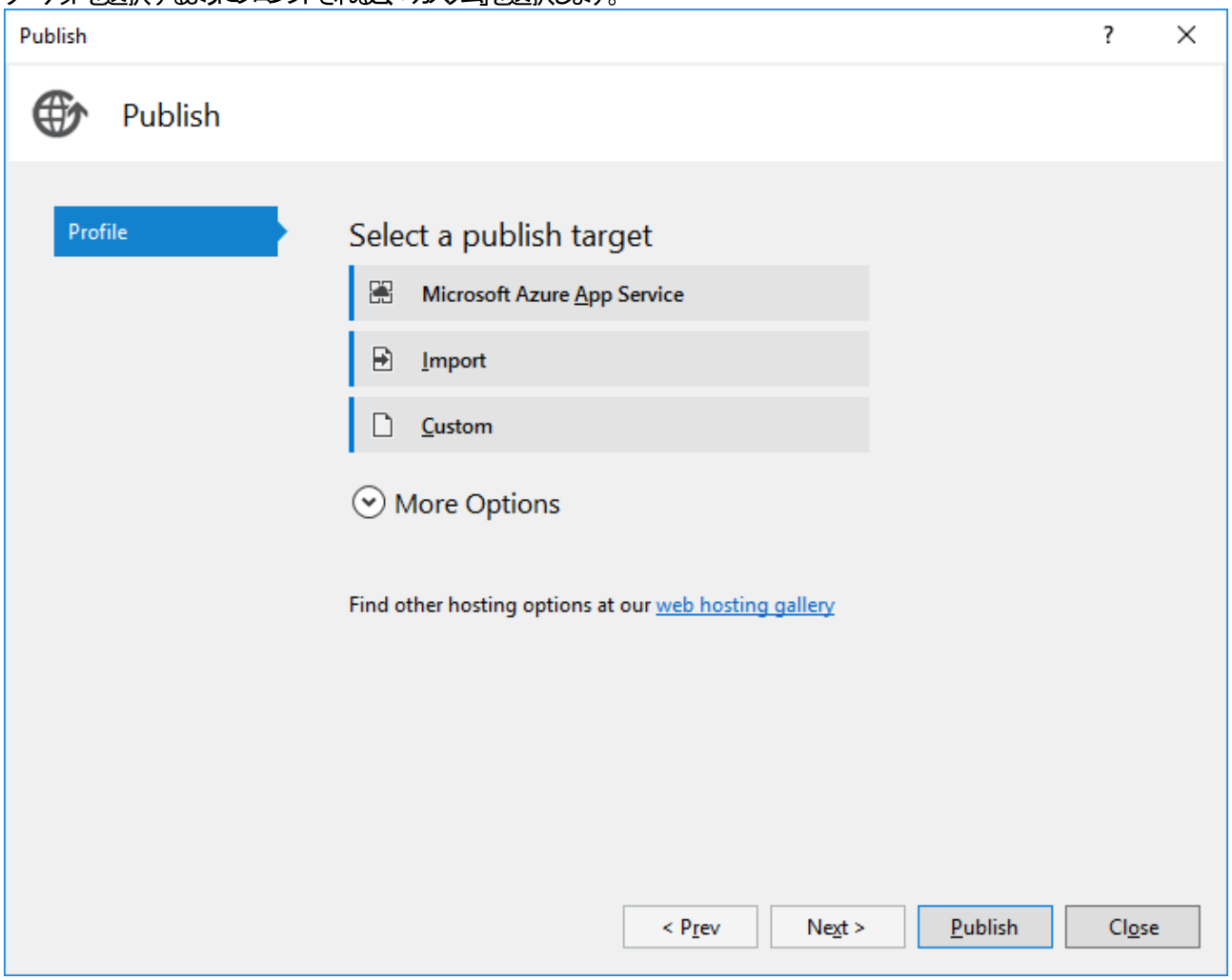


ステップ 3: Web サービスを IIS にデプロイする

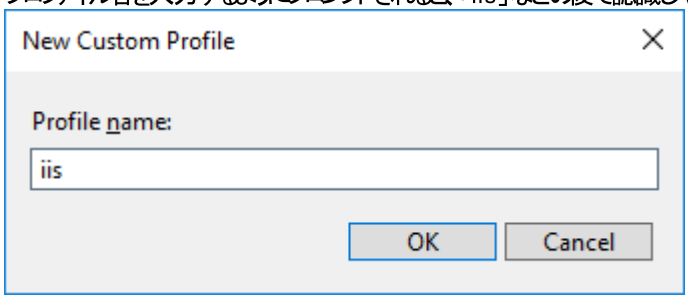
Visual Studio の Solution Explorer ウィンドウ内で、プロジェクト **services-Site/WS2DB** を右クリックし、「Publish Web App」をコンテキストメニューから選択します。



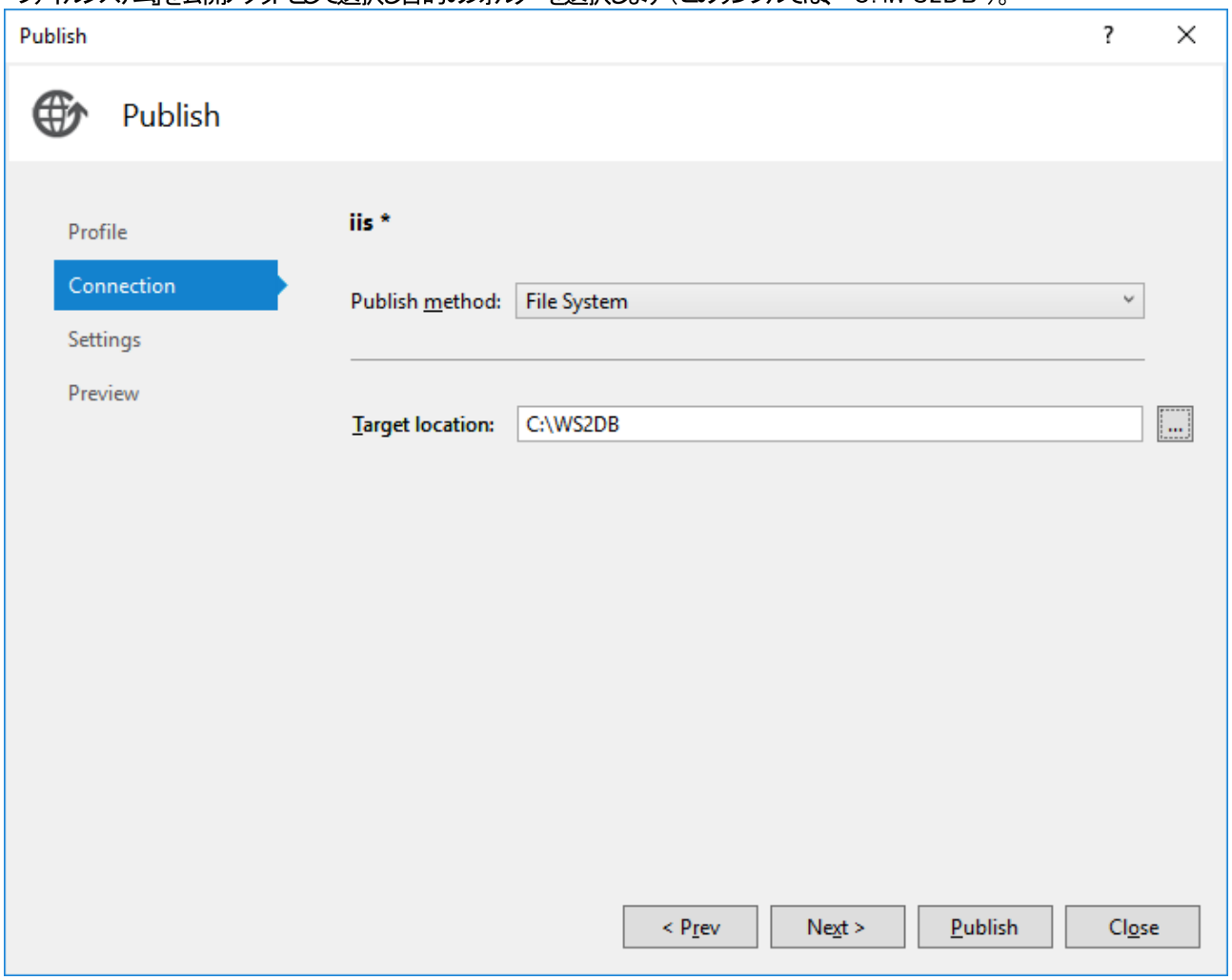
ターゲットを選択するようにプロンプトされると、「カスタム」を選択します。



プロファイル名を入力するようにプロンプトされると、「iis」などの後で認識しやすい値を入力します。

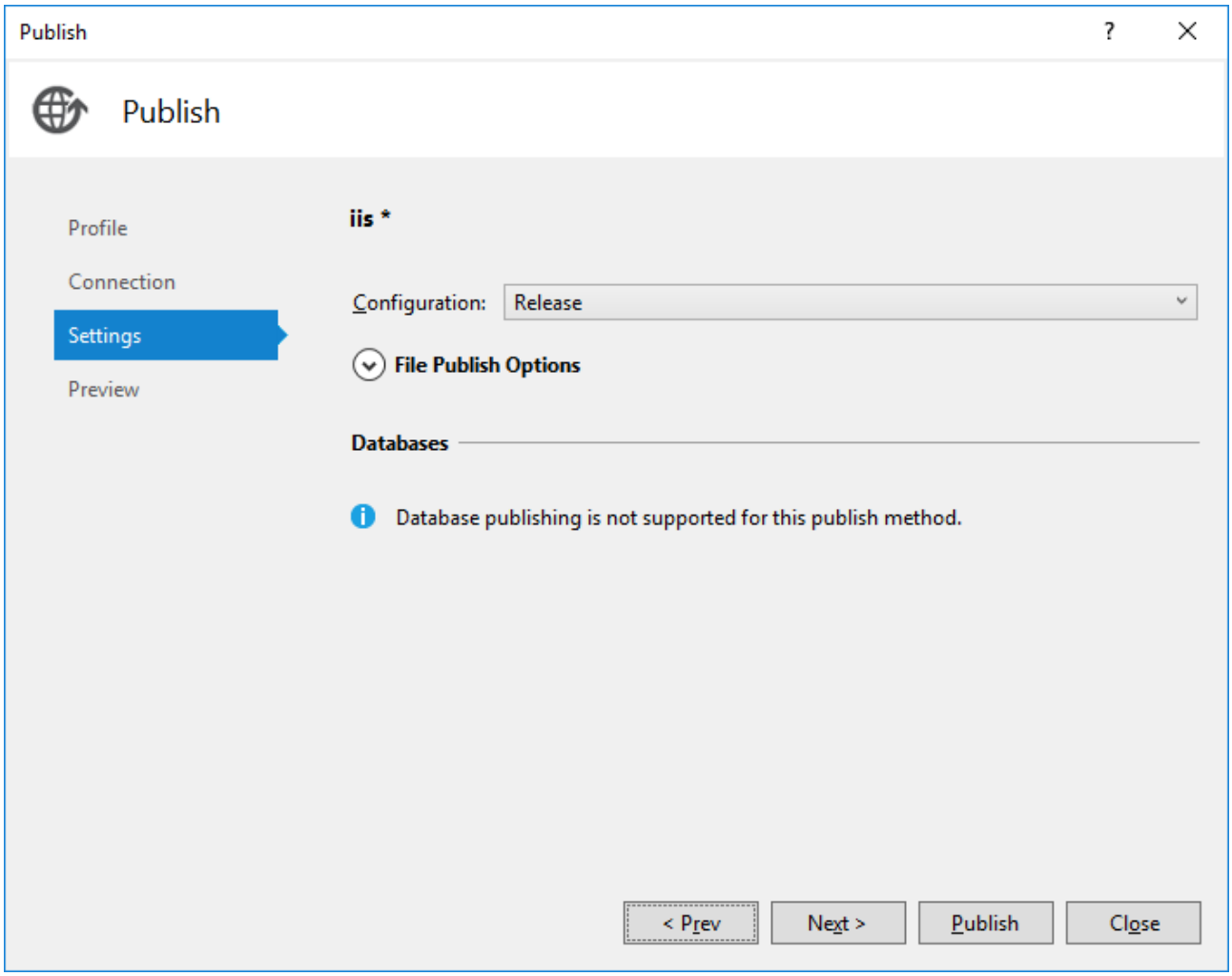


「ファイルシステム」を公開メソッドとして選択し目的のフォルダーを選択します(このサンプルでは、"C:¥W S2DB")。

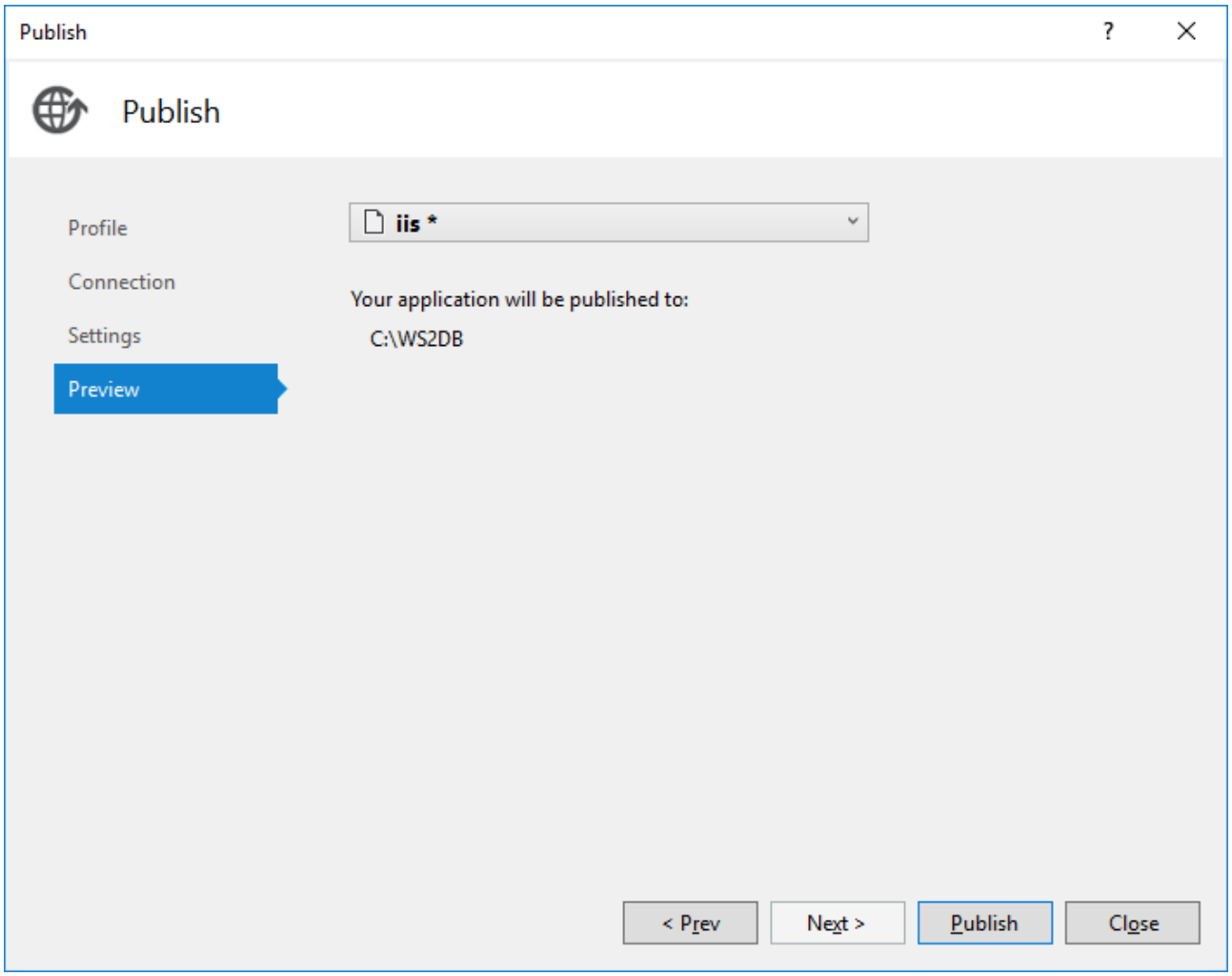


「Next」をクリックします。デフォルトの設定はそのままにします。

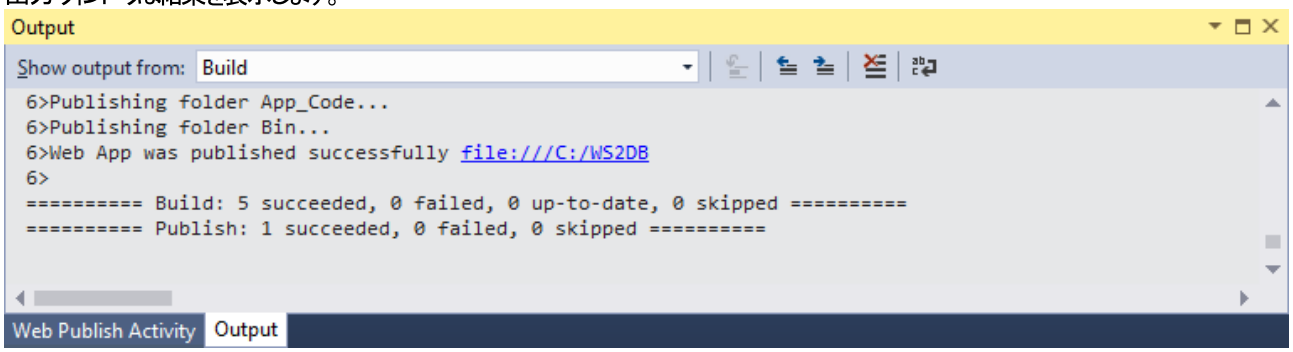
(C) 2015-2021 Altova GmbH



「Next」をクリックして、「Publish」をクリックします。



出力ウィンドウは結果を表示します。

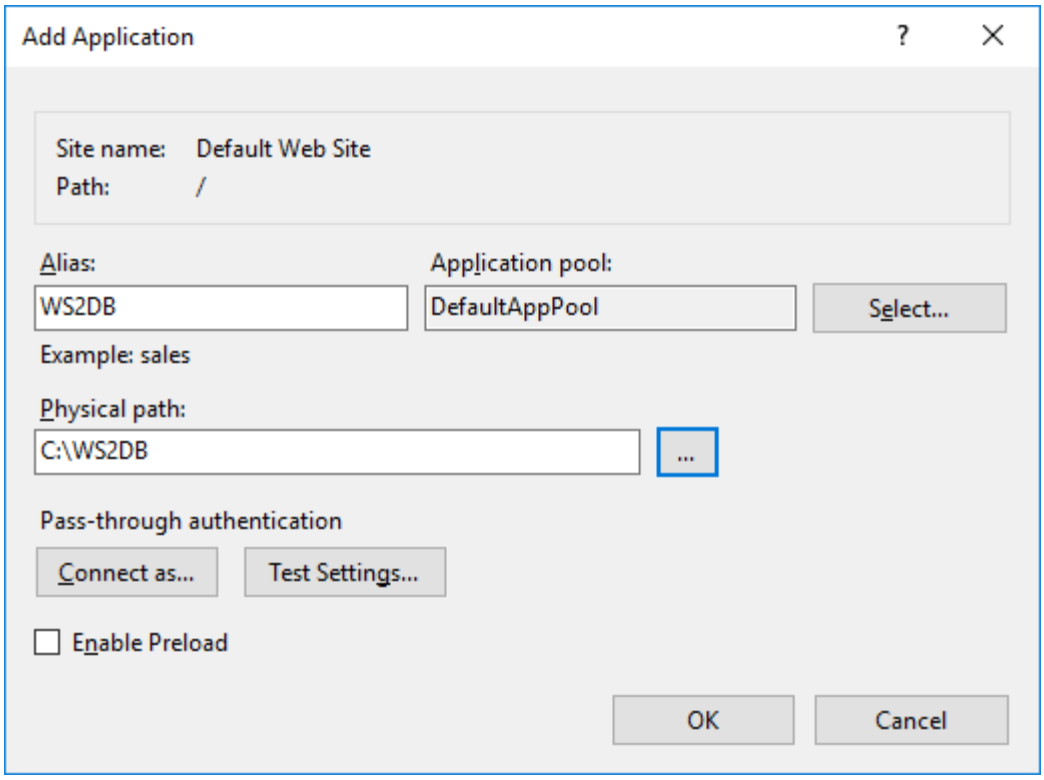


ステップ 4: IIS マネージャー内のアプリケーションの構成

新規アプリケーションを IIS に次のように追加することができます:

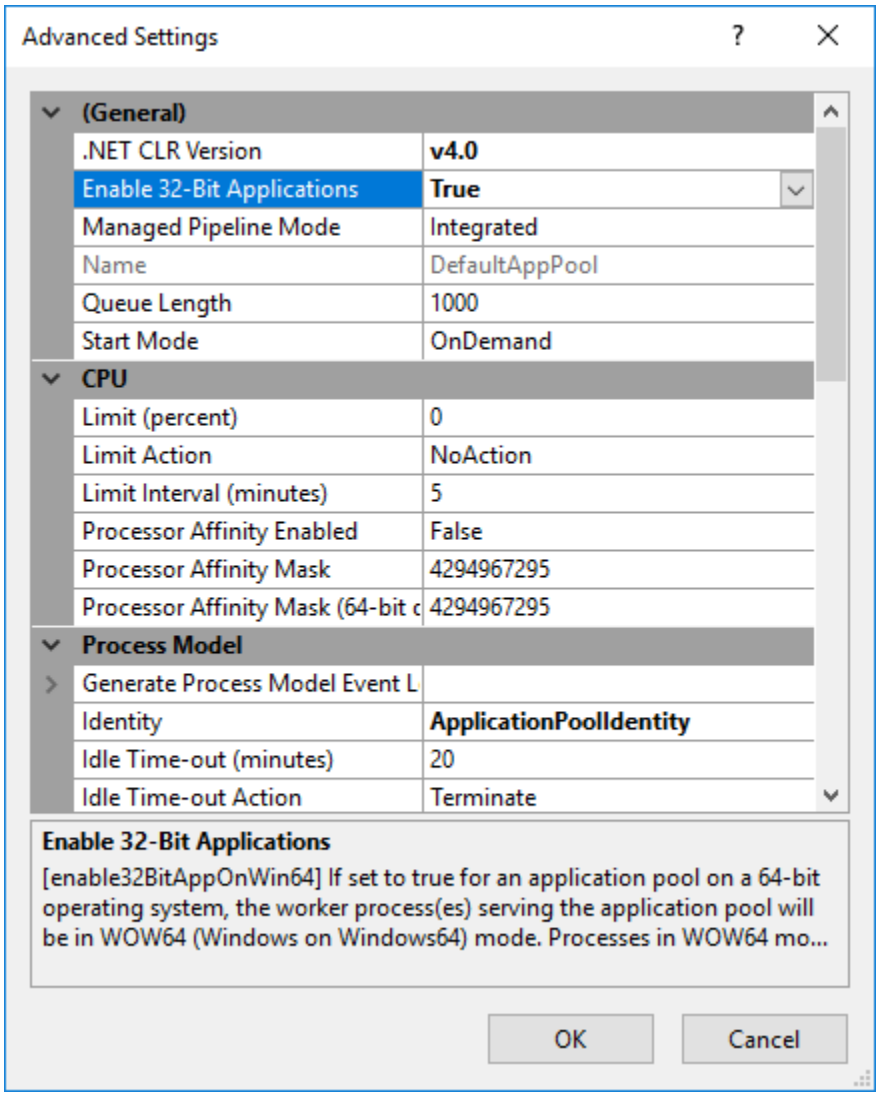
1. IIS 内では、「Sites | Default W eb Site」を右クリックし、コンテキストメニューから「Add Application」を選択します。
2. 「Alias name」の下に、W eb アプリケーションを認識する名前を入力します(このサンプルの場合「サービス」)。
3. 「Physical path」の下に、アプリケーションが以前公開されているパスを入力します(このサンプルの場合「C:¥W S2DB」)。

(C) 2015-2021 Altova GmbH



更に、この特定のサンプルに関しては、アプリが作動するプールのために32-ビットアプリケーションが有効化されている必要があります。

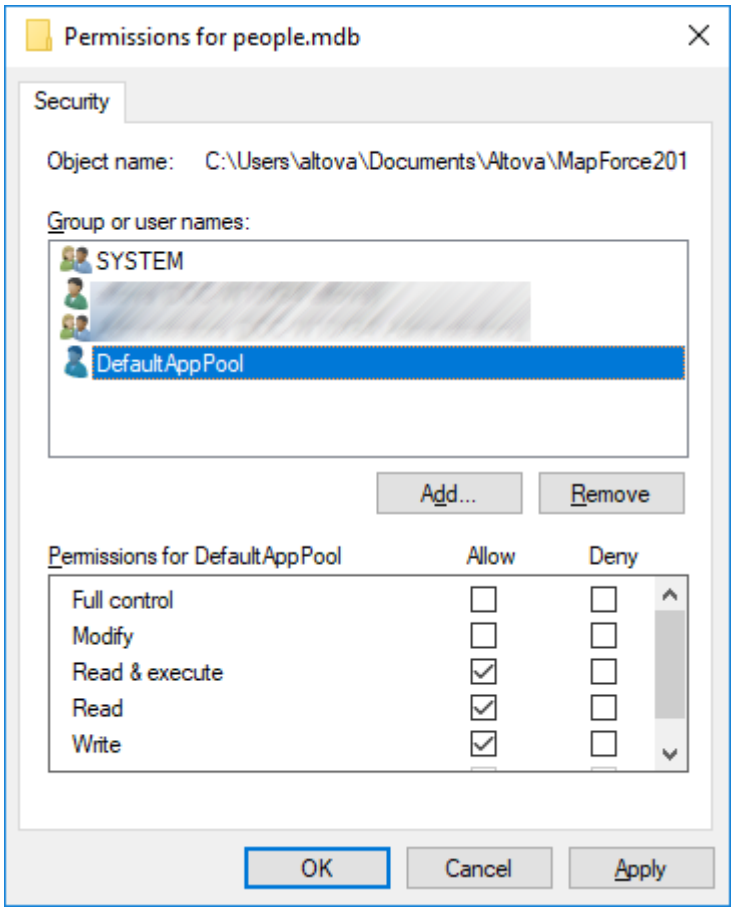
1. IIS 内で、アプリがデプロイされているアプリケーションプールを右クリックします。コンテキストメニューから「高度な設定」を選択します。
2. 「32-ビットアプリケーションを有効化する」からオプション「True」を選択します。



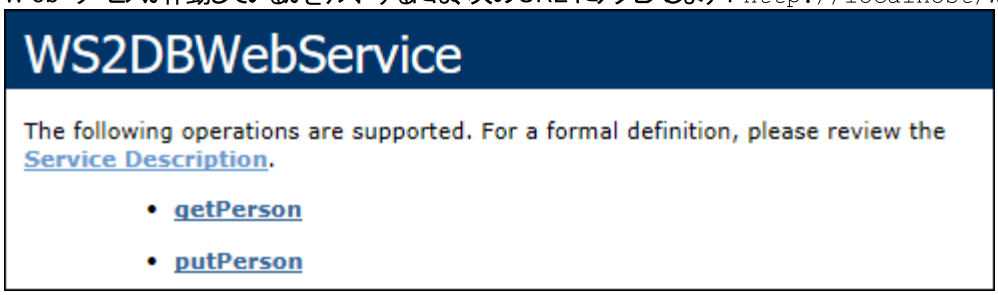
Microsoft Access データベースからデータが読み取られるため、IIS アプリケーションツールはデータベースファイルにアクセスするパーミッションが必要になります。MapForce プロジェクトを変更していない限り、データベースを次のパスから使用することができます: **<マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥Tutorial¥people.mdb**。

1. Windows Explorer 内で、 **people.mdb** ファイルを右クリックし、「プロパティ」を選択します。
2. 「Security」 タブから「Edit」をクリックします。
3. 「Add」 をクリックし、オブジェクト名を次のように入力します: **IIS AppPool\DefaultAppPool**。デフォルト以外のプールにアプリをデプロイした場合、「DefaultAppPool」 を異なる値に変更します。
4. 必要とされるパーミッションを与え、「Apply」をクリックします (このサンプルでは、データベースファイルへの書き込みとデータベースファイルの読み取りのパーミッションが必要になります)。

(C) 2015-2021 Altova GmbH



W eb サービスが作動しているかをテストするには、次のURL にアクセスします: `http://localhost/WS2DB`.



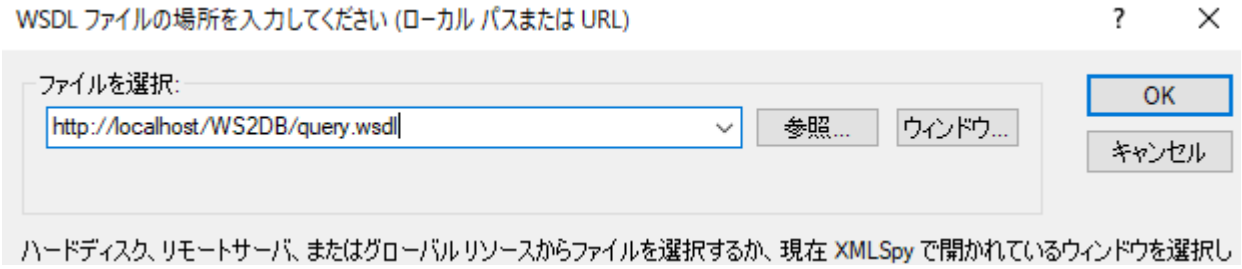
W eb サービスの詳細がブラウザー内でロードされる場合、下記のとおりクライアントアプリケーションからW eb サービスの呼び出しに進むことができます。

W eb サービスの呼び出し

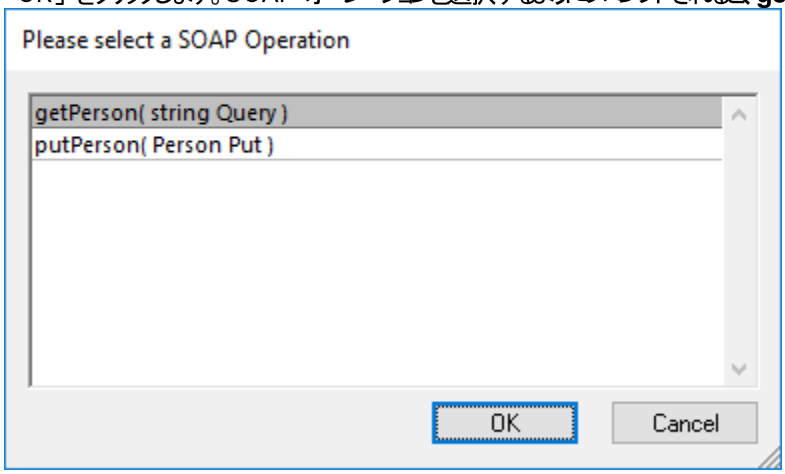
文字通り上記のステップをおこなうと、W eb サービスのW SDL は以下で使用することができます: `http://localhost/WS2DB/query.wsdl`. デプロイしたW eb サービスを呼び出すには、他のSOAP クライアントを使用することもできますが、XMLSpy を使用します。

メモ "?wsdl" クエリメソッドを使用する代わりに`http://localhost/WS2DB/query.wsdl` で正規のW SDL を使用することが奨励されます。前者は機能の一部が不足している、または作動しないW SDL ファイルを返す可能性があります。

XMLSpy Enterprise Edition を起動します。「SOAP」メニューから「新規の SOAP リクエストを作成する」をクリックします。プロンプトされると、W eb サービスの W SDL を入力します(このサンプルでは、`http://localhost/WS2DB/query.wsdl`)。



「OK」をクリックします。SOAP オペレーションを選択するようにプロンプトされると、**getPerson** を選択します。



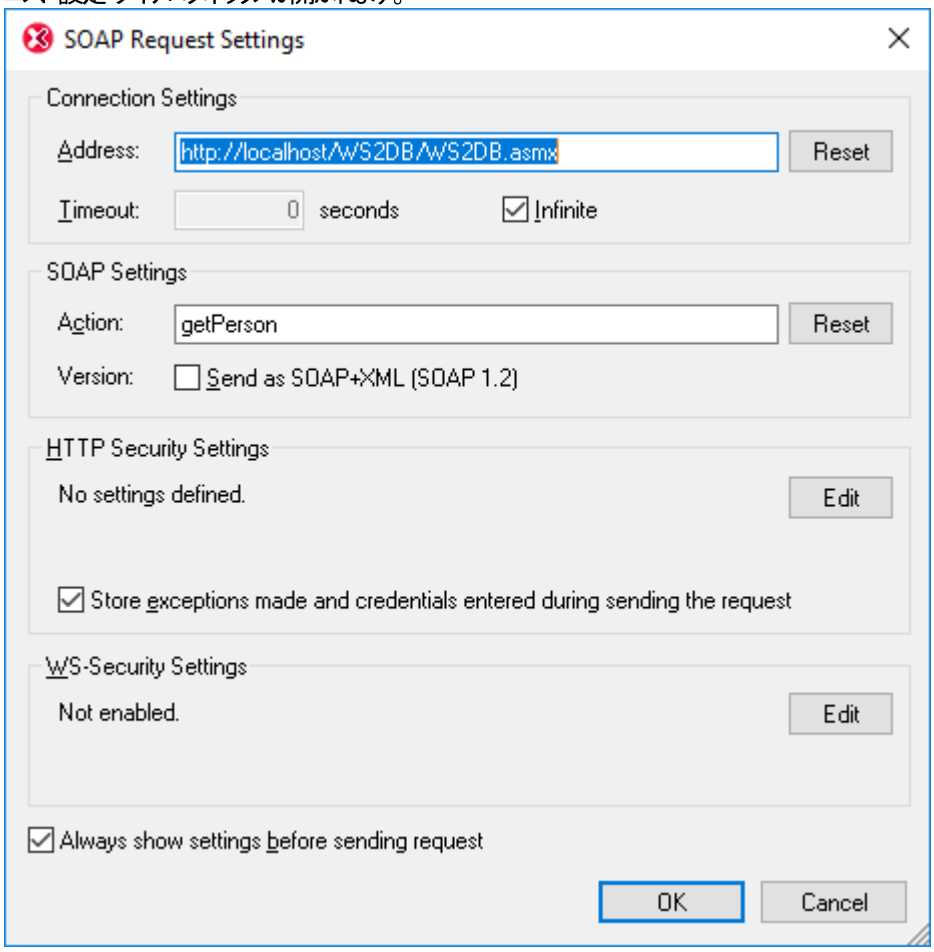
XMLSpy は以下のようなサンプルリクエストを生成します。

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
   <SOAP-ENV:Body>
      <getPerson SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
         <Query xsi:type="xsd:string">String</Query>
      </getPerson>
   </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

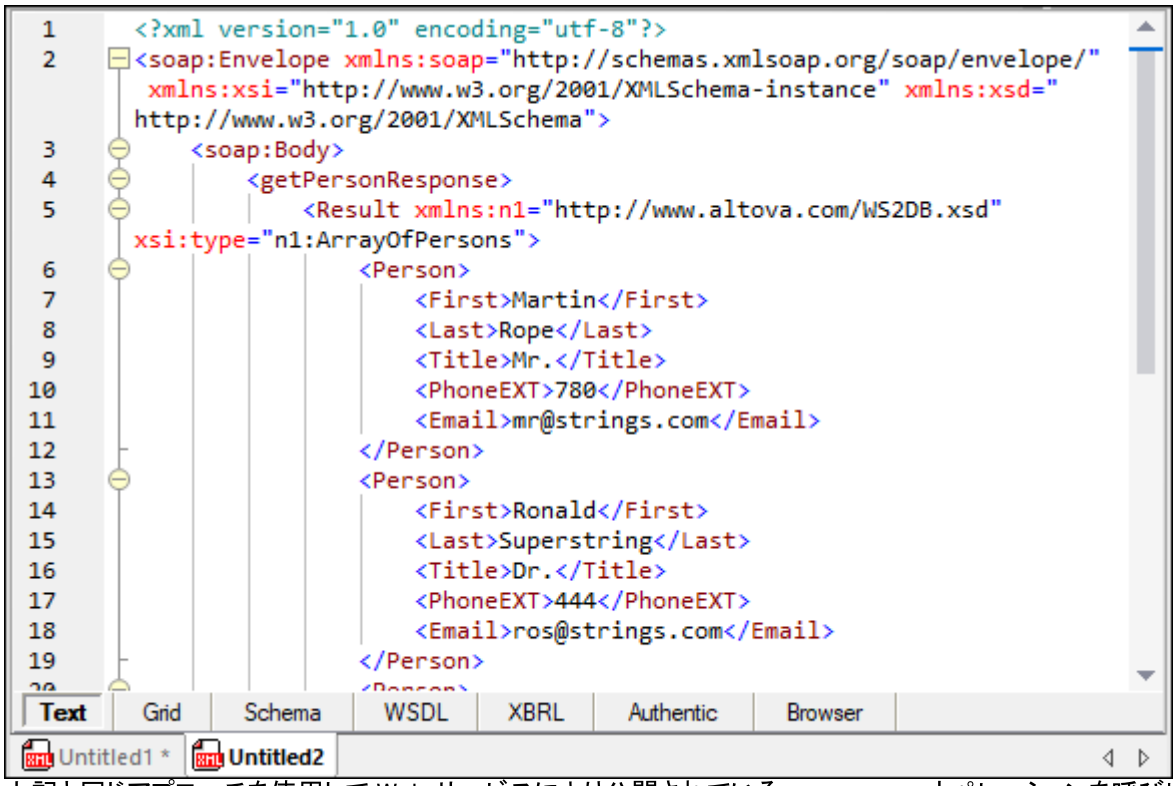
リクエストの<Query> 要素を検索し、W eb サービスに入力として提供する値を「文字列」と置き換えます。上記のとおり、このW eb サービスは Access データベースから個人の情報を抽出します。このサンプルのスコープのため、実際にデータの一部を抽出するために、「文字列」を「Ro」と置き換える必要があります(すなわち、氏名に「Ro」が含まれている個人のみが抽出されます)。新しいリクエストは、以下のとおりです:

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
   <SOAP-ENV:Body>
      <getPerson SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
         <Query xsi:type="xsd:string">Ro</Query>
      </getPerson>
   </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

サーバーに上記のリクエストを送信することができます。**SOAP** メニューから「サーバーにリクエストを送信する」をクリックします。SOAP リクエスト設定 ダイアログボックスが開かれます。



「OK」をクリックします。XMLSpy は呼び出しを開始し、エディター内にレスポンスを返します。下のイメージは成功したレスポンスを表示しています。



上記と同じアプローチを使用して Web サービスにより公開されている `putPerson` オペレーションを呼び出すことができます。詳細な説明に関しては、Java チュートリアル内の「Web サービスの呼び出し」セクションを参照してください。Java チュートリアル内では、WSDL URL のみが異なりますが他の Web サービスの呼び出しのための命令は同様です。

トラブルシューティング

次のテーブルは Web サービスと他のソリューションをビルドする際に発生する一般的な問題をリストしています。

問題	ソリューション
Web サービスの呼び出しは次のエラーを返します: `The 'Microsoft.Jet.OLEDB.4.0' provider is not registered on the local machine."`	1. 32-ビットアプリケーションとして Visual Studio 内で C# ソリューションをビルドします。 2. IIS にデプロイします。 3. アプリケーションがデプロイされている IIS アプリケーションツールを右クリックし「高度な設定」を選択します。 4. 32-ビットアプリケーションを有効化から「True」を選択します。
Web サービスの呼び出しは次のエラーを返します: `The Microsoft Jet database engine cannot open the file '<path>\people.mdb'. It is already opened exclusively by another user, or you need permission to view its data.`	• Access データベースファイルを読み取るために、IIS アプリケーションにプール パーミッションを与えます。「ステップ 4: IIS マネージャー内でアプリケーションを構成する」を参照してください。
Web サービスの呼び出しは次のエラーを返します:	• Access データベースファイに書き込むために、IIS アプリケーションにプール パーミッションを与えます。「ステップ

(C) 2015-2021 Altova GmbH

問題	ソリューション
`System.Data.OleDb.OleDbException: Operation must use an updateable query.`	4: IIS マネージャー内でアプリケーションを構成する」を参照してください。

8.6 例: SOAP Web サービスの生成 (Java)

このサンプルはMapForce を使用してSOAP W eb サービスを生成する方法について説明しています。プログラムコード 次のパスで見つけることのできるサンプルMapForce プロジェクトからJava プログラムコードを生成します: <**マイドキュメント >\Altova\MapForce2021\MapForceExamples\Tutorial\Query Person database.mfp**。このプロジェクトは2 つのオペレーションを公開するSOAP W eb サービス実装します。: `getPerson` と`putPerson`. このW eb サービスは個人の詳細を抽出、または追加するために上記のオペレーションをおこない、背景でMicrosoft Access データベースと通信します。MapForce 内でこのようなプロジェクトを作成するサンプルに関しては、次を参照してください: SOAP W eb サービスプロジェクトの作成.

生成されたJava コードをコンパイルするために、このサンプルはJava 7 とApache Ant を必要とします。.aar ファイルは Axis2 がインストールされた状態で作動中の Apache Tomcat にデプロイされます。

簡素化のためこのサンプルは基本のW eb サーバー構成を使用していることにご注意してください。生産環境では、W eb サーバー、または所属機関内のセキュリティポリシー、その他の要素が指定するとおりの異なるアプローチを使用しなければならない場合があります。

SOAP W eb サービスを呼び出すために、このサンプルはXMLSpy Enterprise Edition を使用しています。XMLSpy がない場合、W eb サービスをテストするために、SOAP クライアントまたはW eb サービスを呼び出すことのできるプログラムコードが必要になります。

メモ 上記のとおり、このW eb サービスのサンプルはMicrosoft Access データベースにアクセスします。Java 7 を使用すると、Java 内に含まれるODBC-JDBC ブリッジを使用してAccess データベースに接続できるようになります。しかしながら、ODBC-JDBC ブリッジはJava 8 内から削除されるため、Oracle はベンダーから提供されるJDBC ドライバーの使用を奨励しています。次を参照してください: http://docs.oracle.com/javase/7/docs/technotes/guides/jdbc/bridge.html。このサンプルをJava 8 または更に新しい環境で作動するには、Microsoft Access データベースに接続することのできる第三者パーティのJDBC ドライバーが必要になります。

ステップ 1: query.wsdl ファイルの準備

このサンプルW eb サービスは次のW SDL ファイルが基礎になっています: <**マイドキュメント >\Altova\MapForce2021\MapForceExamples\Tutorial\query.wsdl**。デフォルトでは、**query.wsdl** ファイルはC# とJava のためのサービスの詳細を1 つの箇所がコメントアウトされている2 つの異なる箇所で定義しています。ですからJava コードを生成前に、**query.wsdl** ファイル内の次のセクションをロケートし、コメントアウトされていることを確認してください:

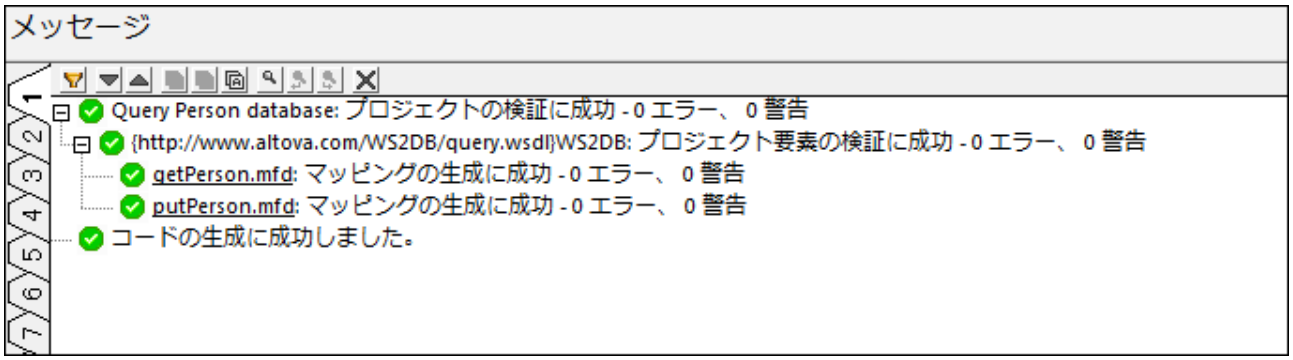
```
<!--service name="WS2DB">
   <port name="WS2DBSoapPort" binding="tns:WS2DBSoapBinding">
      <soap:address location="http://localhost/WS2DB/WS2DB.asmx"/>
   </port>
</service-->
```

(Java にのみ適用可能な) 次のセクションがコメントアウトされていないことを確認してください:

```
<service name="WS2DB">
   <port name="WS2DBSoapPort" binding="tns:WS2DBSoapBinding">
      <soap:address location="http://localhost:8080/axis/services/WS2DB"/>
   </port>
</service>
```

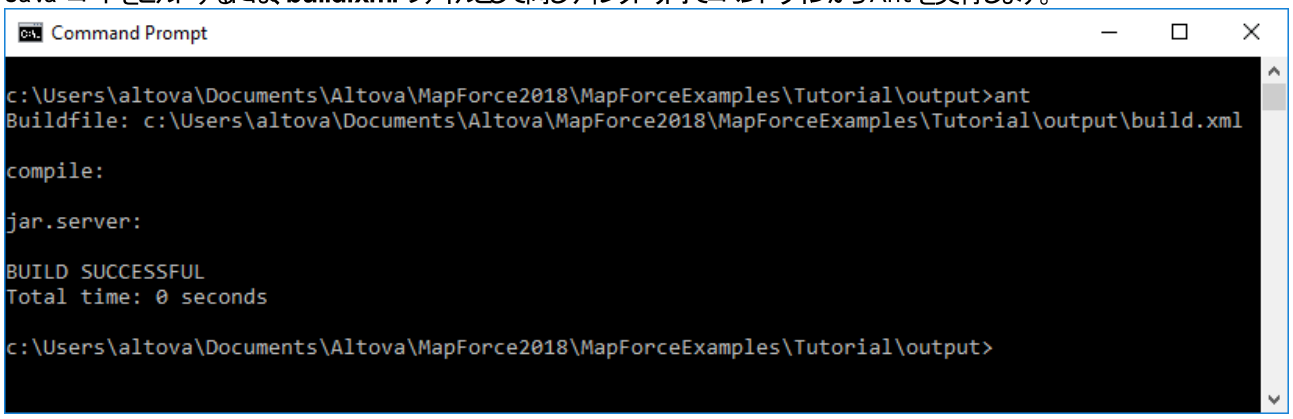
ステップ 2: Java コードの生成とビルド

MapForce 実行し、 <マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥Tutorial¥フォルダーから「Query Person database.mfp」プロジェクトを開きます。プロジェクトウィンドウ内のプロジェクトを右クリックし、「**コードの生成 | Java**」を選択します。MapForce はプログラムコードを生成し、メッセージウィンドウ内に結果を表示します:



デフォルトでは、コードはプロジェクトのディレクトリに対応する「出力」と呼ばれるサブディレクトリに生成されます。このチュートリアルのスコープのため、デフォルトの設定がそのままにされています。出力ディレクトリを変更するには、次を参照してください: コード生成設定の設定.

Java コードをビルドするには、**build.xml** ファイルとして同じディレクトリ内でコマンドラインから Ant を実行します。



この結果、**Query_Person_database.aar** と呼ばれる Axis Archive (.aar) ファイルが出力ディレクトリ内で生成されます。

上記のとおりディレクトリから Ant を呼び出す前に、インストールと構成を正しくおこなう必要があります。簡単な概要は、以下のとおりです (詳細に関しては Ant のマニュアルを参照してください: http://ant.apache.org):

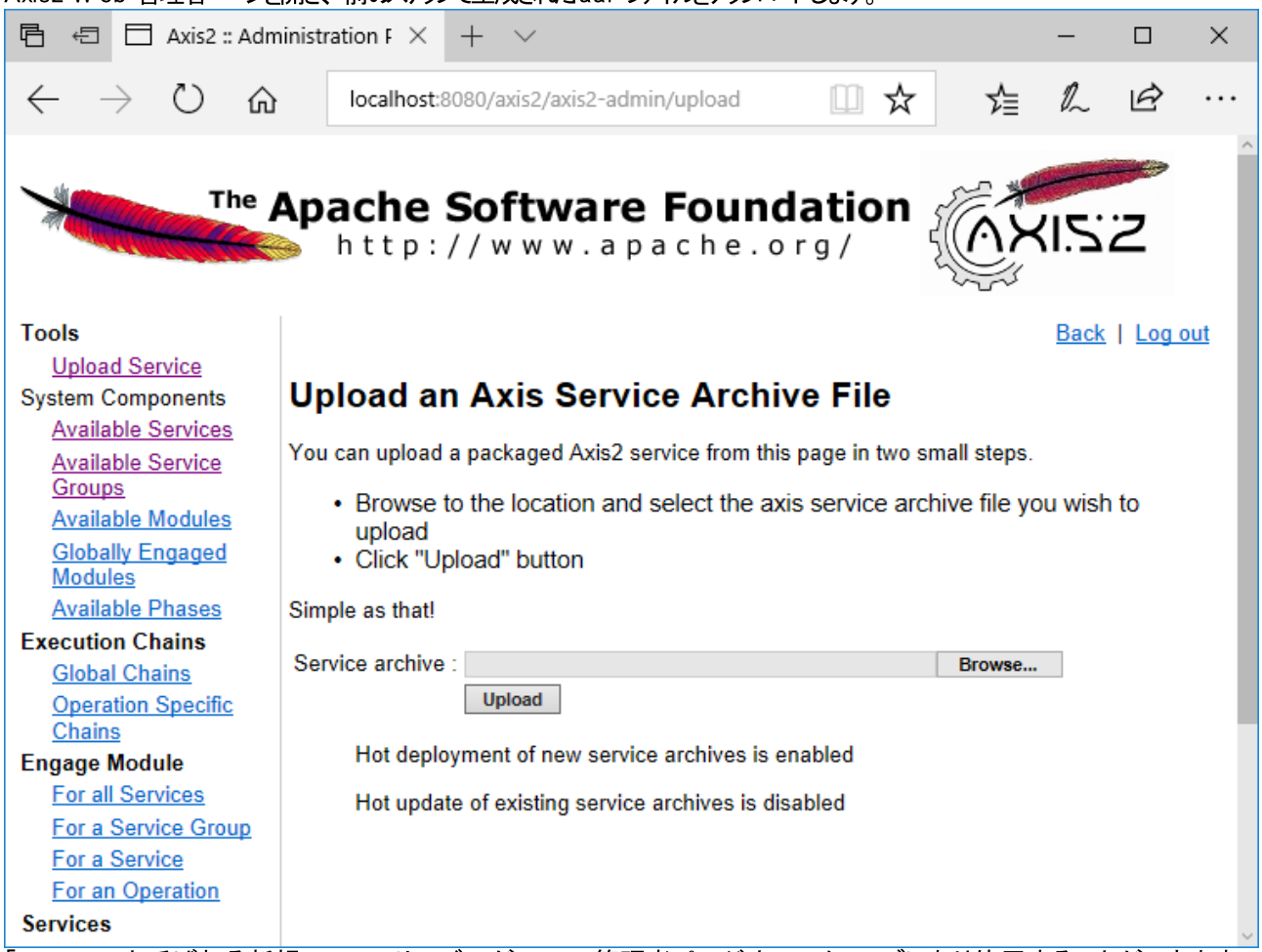
- PATH 環境関数 は Ant ルートディレクトリの "bin" サブディレクトリへのパスを含む必要があります。
- ANT_HOME 環境関数は Ant がインストールされるディレクトリを設定する必要があります。
- AXIS2_HOME 環境変数は Axis2 の標準の配布がインストールされているディレクトリに設定されている必要があります。
- JAVA_HOME 環境関数は Java JDK がインストールされるディレクトリを設定する必要があります。
- コマンドラインに「ant」を入力してが作動しているかを確認してください。メッセージ「Buildfile: build.xml は存在しません!」は Ant が構成されており、ビルドファイルが検索されていることを意味します。

ステップ 3: Web サービスを Axis 2 にデプロイする

このサンプルでは、Web サービスは Axis2 管理ページから Tomcat にアップロードされます。後者はインストール後、サーバーに以下のように作動中の Tomcat サーバーにデプロイされることができます (詳細に関しては、Axis2 ドキュメントを参照してください):

1. Axis2 WAR (Web Archive) 配布を http://axis.apache.org/axis2/java/core/download.html からダウンロードしてください。
2. ローカルディレクトリでダウンロードされたパッケージをアンパックしてください。
3. **axis2.war** ファイルを Tomcat のディレクトリ **%CATALINA_HOME%\webapps** にコピーしてください。
4. http://localhost:8080/axis2 を開き、Axis2 が作動しているか確認してください。

Axis2 Web 管理者ページを開き、前のステップで生成された aar ファイルをアップロードします。



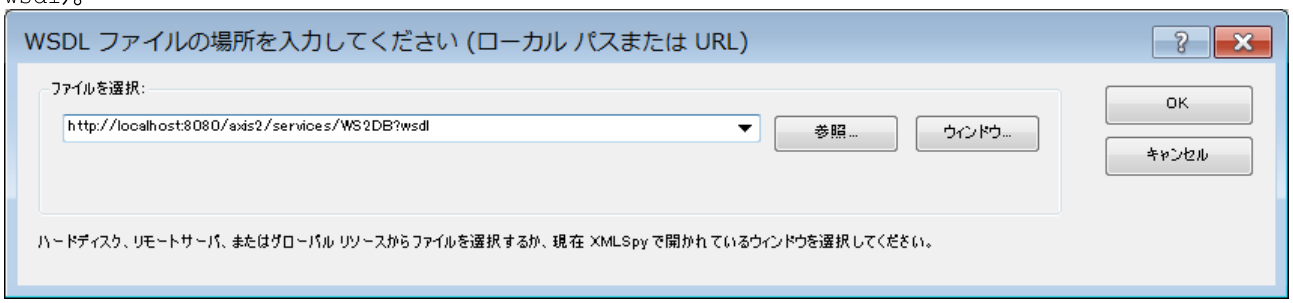
「WS2DB」と呼ばれる新規の Web サービスが Axis2 管理者ページ内でアクティブになり使用することができます。

(C) 2015-2021 Altova GmbH

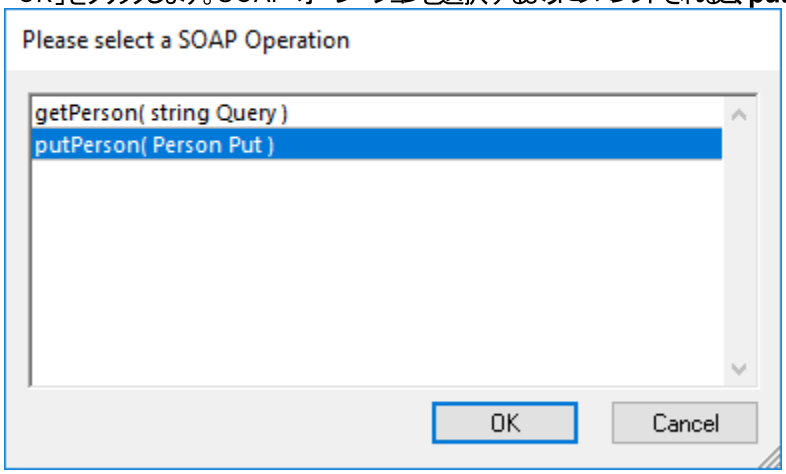
Web サービスの呼び出し

文字通り上記のステップをおこなうと、Web サービスのWSDL は以下で使用することができます: `http://localhost:8080/axis2/services/WS2DB?wsdl`。デプロイしたWeb サービスを呼び出すには、他のSOAP クライアントを使用することもできますが、XMLSpy を使用します。

XMLSpy (Enterprise Edition) を起動します。**SOAP** メニューから「**新規のSOAP リクエストを作成**」をクリックします。プロンプトされると、Web サービスのWSDL を入力します (このサンプルでは `http://localhost:8080/axis2/services/WS2DB?wsdl`)。



「OK」をクリックします。SOAP オペレーションを選択するようにプロンプトされると、**putPerson** を選択します。



XMLSpy は次のようなサンプルリクエストを生成します。

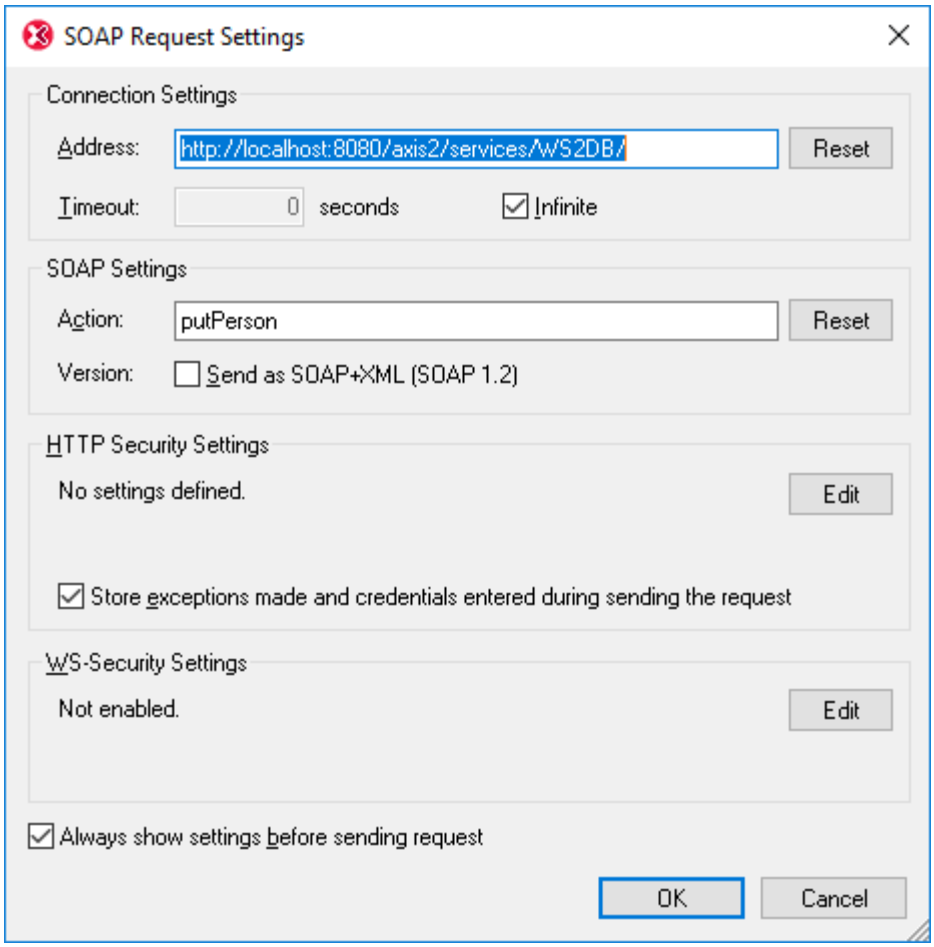
```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:m0="http://www.altova.com/WS2DB.xsd">
   <SOAP-ENV:Body>
      <putPerson SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
         <Put xsi:type="m0:Person">
            <First xsi:type="xsd:string">String</First>
            <Last xsi:type="xsd:string">String</Last>
            <Title xsi:type="xsd:string">String</Title>
            <PhoneEXT xsi:type="xsd:string">String</PhoneEXT>
            <Email xsi:type="xsd:string">String</Email>
         </Put>
      </putPerson>
   </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

「文字列」を W eb サービスに入力として提供する個人の詳細と置き換えます。このサンプルのスコープのために下のようなリクエストを使用することができます:

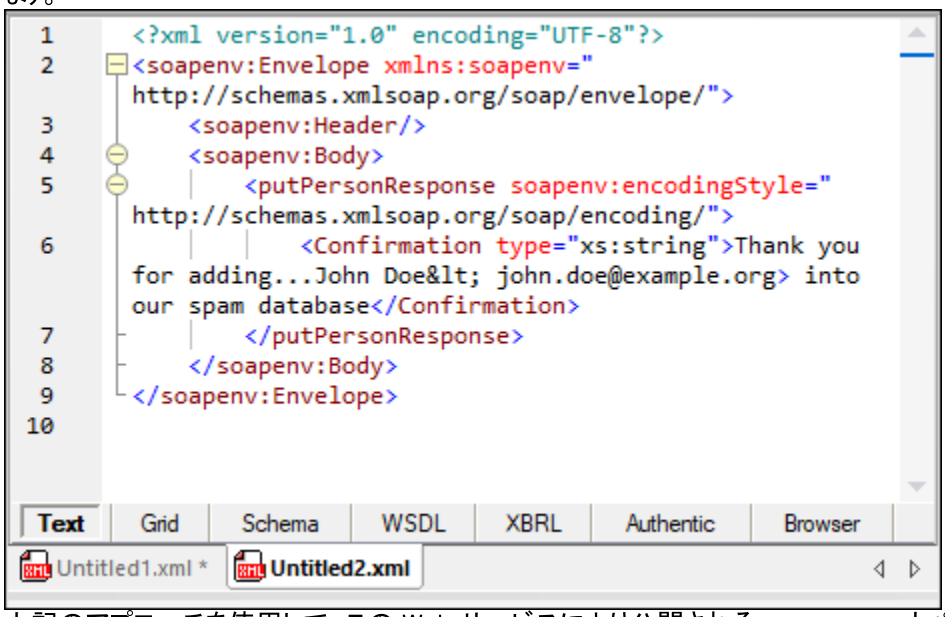
```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:m0="http://www.altova.com/WS2DB.xsd">
   <SOAP-ENV:Body>
      <putPerson SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
         <Put xsi:type="m0:Person">
            <First xsi:type="xsd:string">John</First>
            <Last xsi:type="xsd:string">Doe</Last>
            <Title xsi:type="xsd:string">Mr</Title>
            <PhoneEXT xsi:type="xsd:string">123</PhoneEXT>
            <Email xsi:type="xsd:string">john.doe@example.org</Email>
         </Put>
      </putPerson>
   </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

サーバー上でリクエストを送信することができます。**SOAP** メニューから「サーバーにリクエストを送信する」をクリックします。SOAP リクエスト設定ダイアログボックスが開かれます。

(C) 2015-2021 Altova GmbH



「OK」をクリックします。XMLSpy は呼び出しを開始し、エディター内にレスポンスを返します。下のイメージは正しいレスポンスを表示しています。



上記のアプローチを使用して、この Web サービスにより公開される `getPerson` オペレーションを呼び出すこともできます。詳細な手順 に関しては、C# チュートリアル内の Web サービスの呼び出しセクションを参照してください。

WSDL URL のみが C# チュートリアル内で異なりますが、その他の Web サービスの呼び出しのための命令は同じです。

(C) 2015-2021 Altova GmbH

9 Web サービスの呼び出し

Web サービスを実行の一部として呼び出すようにマッピングをデザインすることができます。具体的には、Web サービスの呼び出しを関数に類似したマッピングに入力、入力と出力コンポーネントを接続し、必要とされる通り結果を使用することができます（例えば、MapForce 出力ウィンドウでプレビューし、他のコンポーネントにパス、またはファイルに書き込むことができます）。これは、視覚的なユーザーインターフェイスから簡単に構成することのでき、効果的にMapForce をパワフルな Web サービスクライアントします。MapForce を使用したWeb サービスの呼び出しはMapForce 自信で、または異なるマシン、またはMapForce Server を使用して他のプラットフォームで実行することができます。後者は、API 呼出、またはFlowForce Server ジョブとして、マッピングをコマンドラインインターフェイスで実行することができます。MapForce Server を使用して自動化するを参照してください。

MapForce は、WSDL-スタイルとREST-スタイルWeb サービスの両方の呼び出しをサポートします。このため、Web サービスをマッピングに追加すると、次のアプローチを使用することができます:

- WSDL-スタイルWeb サービスのために、呼び出されるWeb サービスのWSDL（Web サービス定義言語）をMapForce に与えます。MapForce は.wsdl ファイルをサーバーと通信するために使用します。.wsdl ファイルが複数のサービス、エンドポイント、オペレーションを実装する場合、MapForce から直接複数のサービス、エンドポイント、オペレーションを選択や更新することができます。
- REST-スタイルWeb サービスに関しては、Web サービスの詳細をMapForce に手動で入力します。これは、URL、リクエストメソッド（例えば、GET、POST、PUT）、リクエストとレスポンス構造（XML、JSON、またはカスタムMIME 型などの）、およびパラメーターなどを含みます。オプションで、Web サービスのWADL（Web Application Definition Language）ファイルが呼び出される場合、WADL ファイルから定義をインポートすることができます。また、Web サービスのサンプルURL がある場合、手動で定義する必要がなく、MapForce に、URL を解析し、自動的にクエリ、テンプレートまたはマトリックスパラメーターを抽出するように命令することができます。

次のテーブルにはMapForce 内のREST-スタイルとWSDL-スタイルWeb サービスのサポートに関する詳細がまとめられています。

機能	REST-スタイル	WSDL-スタイル
マッピング言語	BUILT-IN	BUILT-IN、C#、Java
MapForce Server を使用した自動化	はい	言語が"BUILT-IN" の場合、はい
プロトコル	HTTP (GET、POST、PUT、DELETE、カスタム動詞)	• HTTP からのSOAP 1.1、1.2 • HTTP からの非-SOAP サービス
リクエスト/レスポンス構造	• XML • JSON • プロトコルバッファー • カスタムMIME 型	SOAP メッセージ
セキュリティ	• HTTP/HTTPS • サーバーの証明書 • クライアントの証明書 • HTTP 認証 • プリエンプティブ認証 • OAuth 認証	• HTTP/HTTPS • サーバーの証明書 • クライアント証明書* • HTTP 認証 • プリエンプティブ認証* • WS-Security* • OAuth 認証 *C# とJava ではサポートされていません

機能	REST-スタイル	WSDL-スタイル
サービス定義のインポート	任意で、WADL ファイルから	WSDL ファイルから（必須）
URL からパラメーターをインポートする	はい	適用不可
動的な Web サービス URL	Web サービスのフル URL をパラメーターとしてマッピングに提供、またはマッピング内で定義します。ベース URL がマッピング内で定義されている一方、特定の URL パートをマッピングへのパラメーターとして提供することができます。	Web サービスのフル URL をパラメーターとしてマッピングに提供、または（マッピングと保存され）フィックスすることができます。
タイムアウト[1]	はい	はい
動的な認証[2]	はい	はい

脚注:

1. 「タイムアウト」とは MapForce 内で、Web サーバーが応答しない場合、呼び出しのタイムアウトの間隔を構成することができます。
2. ダイアログボックス内に入力しマッピングと保存する代わりに、「動的な認証」とは認証の詳細（ユーザー名、パスワード）をマッピング（または、マッピングのパラメーター）から提供することができることを意味します。詳細に関しては動的な認証を参照してください。

WSDL-スタイルの Web サービスの場合、次のプロトコルを使用することができます:

HTTP からの SOAP 1.1、1.2	*RPC/エンコード* と *ドキュメント/リテラル* スタイルがサポートされています。Web サーバーが WSDL フォールトを返すと、マッピングの実行は停止されます。このような場合、エラーを処理するために任意でマッピングエリアに例外コンポーネントを挿入することができます。Web サーバーが非-WSDL エラーを返す場合、マッピングの実行は停止し、エラーメッセージが返されます（または、MapForce 内でマッピングをプレビューする場合、スクリーン上に表示されます）。
HTTP からの非-SOAP	一般的ではない非-SOAP HTTP サービスを指しています。HTTP GET を使用する場合、*url-エンコード* スタイルがサポートされています。HTTP POST を使用する場合、*url-エンコード* と *text/xml* スタイルがサポートされています。

(C) 2015-2021 Altova GmbH

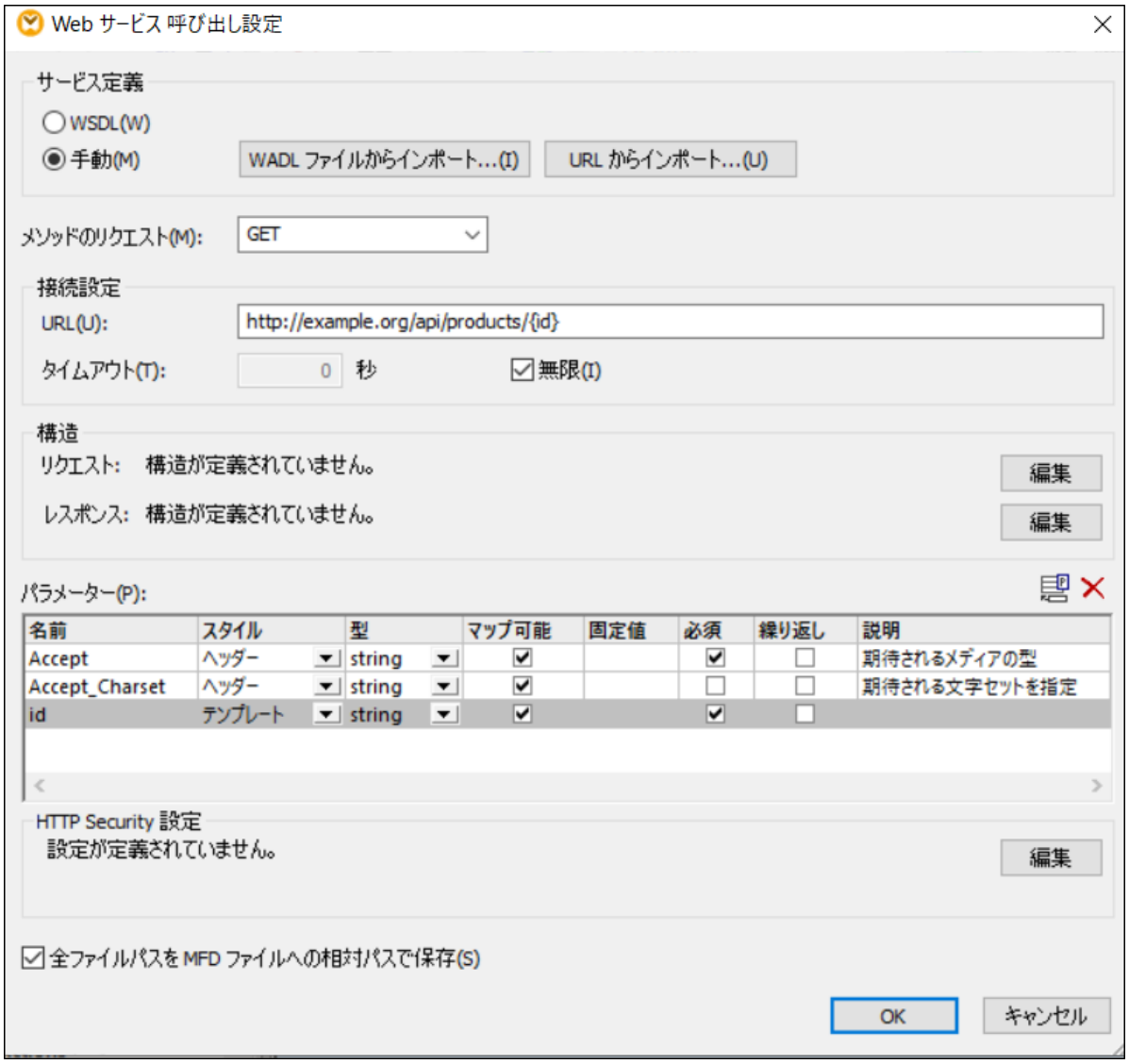
9.1 Web サービス呼び出しの追加 (REST-スタイル)

ジェネリックな (WSDL-スタイルではない) Web サービスには "REST" (また、通常 "RESTful" または REST-スタイル Web サービス) と称されるアーキテクチャのスタイルに従う Web サービスの大きなカテゴリが含まれます。ジェネリックな HTTP Web サービスでは、メッセージの本文は、通常カスタムリクエストまたはレスポンスです。MapForce は以下の種類のリクエスト、またはレスポンスボディをサポートします: JSON、XML、プロトコルバッファー、カスタム MIME 型を持つ構成されていないボディ。ですから、MapForce パースペクティブからは、Web サービスを呼び出す必要条件はリクエストの型により異なります:

- XML または JSON リクエスト、またはレスポンスの場合、MapForce は JSON、XML、または DTD スキーマとしてのリクエスト、またはレスポンスを必要とします。MapForce は有効なスキーマレファレンスを持つ XML ファイルを構造として受け入れます。Web サービスの構造は、XML または JSON スキーマとしてのサービスプロバイダ、または WADL などの正式な言語、または人間が読解することのできる仕様により公開することができます。リクエスト/レスポンスの XML または JSON スキーマが既に存在する場合、Web サービスの呼び出しを簡単に行うことができます。WADL ファイルとしての Web サービスが存在する場合、WADL ファイルからインポートし、手動で調整することができます。WADL は JSON 構造を定義する標準的な方法を提供しません。XML 構造を定義する方法のみを提供します。最後に、スキーマファイルが存在せず、XML または JSON インスタンスファイルのみが存在する場合、XMLSpy (http://www.altova.com/ja/xmlspy.html) を使用してスキーマを作成、または生成することができます。必要な場合、XMLSpy は XML からのインスタンスファイルを JSON に変換、またはその逆に変換することができます。
- プロトコルバッファーリクエスト、またはレスポンスの場合、プロトコルバッファーバイナリファイルを説明する .proto ファイルが必要とされます。このシナリオでは、Web サービスのボディをプロトコルバッファーコンポーネントからマップすることができます。詳細に関しては、サンプル: プロトコルバッファーからデータを読み取るを参照してください。サンプル: プロトコルバッファーにデータを書き込むを参照してください。
- レスポンス構造が柔軟で、特定のスキーマに制限されていない個所で Web サービスを呼び出す、またはリクエストすることができます。このような場合、MapForce ビルトイン mime 関数を使用して Web サービスに送信される生メッセージボディを作成、または Web サービス (MIME エンティティ) により返される MIME エンティティのマッピングを処理することができます。

ジェネリックな Web サービスに呼び出しを追加する方法

1. 「**挿入**」メニューから「**Web サービス関数**」をクリックします。(または、「**Web サービス関数の挿入**」 ツールバーボタンをクリックします)。
2. 「**サービス定義**」から「**手動**」をクリックします。



3. 任意で、サービスを説明するWADL ファイルが存在する場合 「WADL ファイルからインポート」をクリックし、ファイルを選択します。WADL ファイルからWeb サービス情報をインポートするを参照してください。
4. MapForce がWeb サービスを呼び出すために使用するHTTP リクエストメソッドを選択します。既存のリストから値を選択するか、またはリクエストメソッドの名前を入力します。HTTP メソッドは大文字と小文字を区別します。
5. 以下の一つを行います:

   a. **URL** テキストボックス内にWeb サービスのURL を入力します。このアプローチではURL の特定の部分をオプションでパラメーターに変更し、マッピングから提供することができます。この方法で定義されたURL に関しては以下に注意してください:
      i. Web サービスを"テンプレート" または「マトリックス」スタイルのパラメーターで呼び出す場合、パラメーターを中かっこで囲みます。例えば `http://example.org/api/products/{id}`。「パラメーター」テーブル内で各パラメーターの実際の設定を定義します。ランタイムで、MapForce は、中かっこ内のパラメーター名を処理し、実際の値を含む最終 URL を生成します。
      ii. [クエリ] URL パラメーターでWeb サービスを呼び出す場合、(例えば、`http://example.org/api/products?sort=asc&category=1&page=1`)など。URL テキストボックス内にクエリの部分を入力せず、「パラメーター」テーブル内でのみパラメーターを定義します。
      iii. サンプルに関してはWeb サービスパラメーターの定義を参照してください。

(C) 2015-2021 Altova GmbH

b. メインマッピングからWeb サービスの フルURL をマッピングから(または マッピングの パラメーターとして) 提供する場合、(マッピングにより提供される) 動的な URL チェックボックスを選択します。これは **URL** テキストボックスを無効化し、この場合はメインマッピングから(URL パラメーターを含む) Web サービスのフルURL を作成しサービスコンポーネント上に表示される対応する入力に接続される必要があります。このアプローチは[ヘッダー]スタイル パラメーターを除き[パラメーター]テーブル内でパラメーターを指定する有効なアプローチではもうありません。これを行う場合、ダイアログボックスが警告を表示します。

Dynamic URLs
上記の2 つのアプローチ(フルまたは部分的な URL) では必要に応じて柔軟的にURL を調整することができます。例えば、マッピングを変更することなしに開発中にマッピングを特定のURL で実行することができ、生産過程で異なるURL を使用することができます。ホスト名のみが生産とテスト URL の間の唯一の差分であると想定し、これをマッピングへの パラメーターとして提供することにより `https://{host}/some/path/to/service` のような URL はこれを可能にします。指定されたURL 部分がマップ可能な部分または ランタイム値と置き換えられるため部分的な URL 動的なは更に厳しい検証を強制します。

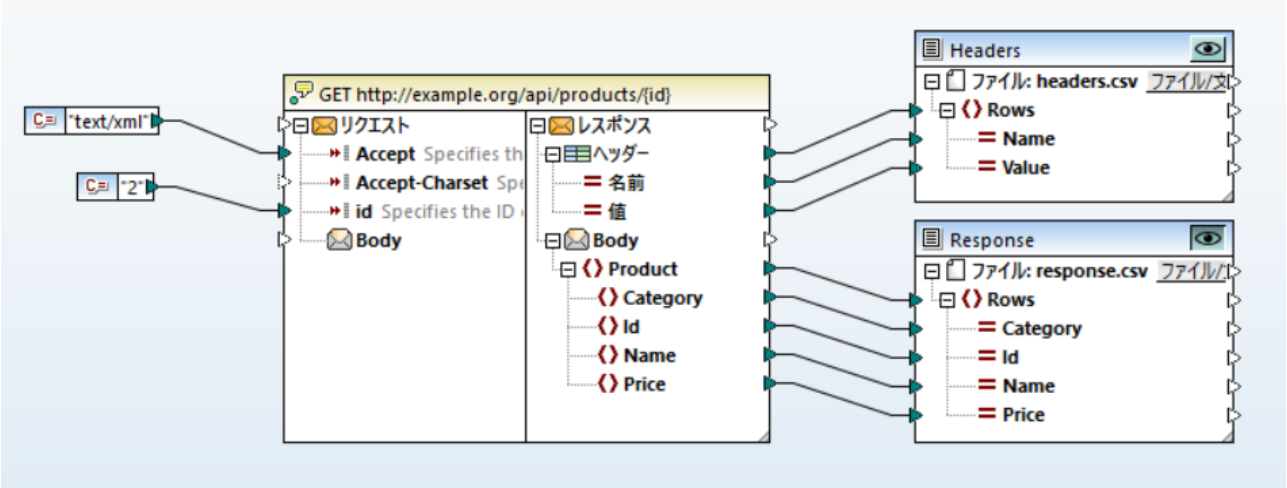
完全に動的な URL では、URL 全体がマップ可能で管理を完了する必要があります。唯一の必要条件はURL が **http://** または **https://** で開始し、有効な URL であることです。動的な URL はWeb サービス呼び出しから同じ(または他の) Web サービスを呼び出す可能性のある他のコンポーネントの入力として返されたURL をサポートすることを可能にします。

6. 任意で、**タイムアウト** の項目に、サーバーが応答しない場合、接続がタイムアウトする秒数を入力します。
7. HTTP メソッドが必要な場合、または XML、JSON および他などの ボディの部分を返す場合、「**構造**」の下の「**編集**」ボタンをクリックし、ボディの部分の スキーマを参照します詳細に関しては リクエストの構造の定義 と レスポンスの構造の定義を参照してください。
8. **パラメーター**でWeb サービスの パラメーターを定義します。任意で、「**URL からインポート**」をクリックして、Web サービスのサンプルURL から パラメーターをインポートし、「パラメーター」テーブルを自動的に作成します。URL ファイルからWeb サービス パラメーターをインポートするを参照してください。パラメーターをURL からインポートした後、「パラメーター」テーブルのエンテンツを必要があれば変更することができます。

メモ　カスタムリクエストヘッダーを指定するには、スタイル「ヘッダー」を持つ パラメーターを、パラメーター名がヘッダー名に対応し、また、パラメーター値がヘッダーの値に対応する箇所に追加します。更に、マッピング自身からヘッダーのリクエストの値を提供する場合、パラメーター型を"マップ可能に設定します。詳細に関しては、Web サービス パラメーターの定義を参照してください。

9. Web サービスが基本のHTTP 認証、OAuth 2.0 認証、または 証明書ベースのセキュリティを必要とする場合、「**HTTP セキュリティ設定**」の下の「**編集**」ボタンをクリックし、必要とされるフィールドに入力します(HTTP セキュリティの設定を参照)。

「**OK**」をクリックした後、新規のWeb サービス コンポーネントがマッピングエリアに追加されます。例えば、下のマッピングは、GET リクエストを使用して、ID により製品を取得するためにWeb サービスを呼び出します。この特別な例では、HTTP リクエスト内で与えられたID は定数の値「2」を持ちます。しかしながら、マッピングの パラメーターにもなることができます。または、MapForce によりサポートされる全てのコンポーネントにより与えられます。`id` パラメーターに加え、リクエストは、`Accept: text/json` を含みます。Web サービス パラメーターの定義で説明される通りヘッダー パラメーターを使用してリクエストヘッダーを設定することができます。



W eb サービスコンポーネントは2つの部分から構成されています: **リクエスト** と**レスポンス**。**リクエスト** パートにより、W eb サービスへマッピングからデータを提供することができ、**レスポンス** 部分により、W eb サービスにより返されたデータにアクセスすることができ、他のフォーマットにマップすることができます。リクエストとレスポンスの構造は、パラメータ、および、 ボタンをクリックして定義されたリクエストとレスポンス構造により異なります。上記のW eb サービスの呼び出しでは、リクエストボディの構造は設定されていませんが、レスポンスボディの構造は、JSON に設定されています。レスポンスボディの構造は、実際のJSON ファイルにW eb サービスの結果をマップできるようにします。

(適用できる場合)リクエストパラメーターを持つW eb サービスを呼び出す場合、(例えば、XML または JSON ファイルなどの) MapForce によりサポートされるコンポーネントと「**リクエスト**」パート間にマッピング接続を描きます。同様にW eb サービスにより返されたデータを他の書式にマップするには、MapForce によりサポートされている**レスポンス**パートと他のコンポーネント型にマッピング接続を描きます。MapForce の初心者の場合は、マッピング接続の描き方の説明を確認してください。接続との作業を参照してください。

(「コンテンツ」で開始しない) 追加ヘッダーの場合 W eb サービスにより返されるレスポンスヘッダーもマップ可能です。ヘッダーの値は2つの子アイテムを含む「ヘッダー」( )と呼ばれるアイテムを介してヘッダー値はW eb サービスコンポーネント上で使用することができます: 「名前」と「値」。この構造はシーケンスとしての役割を果たし、W eb サービスにより返されたヘッダーの数からのデータをマップすることができます。レスポンスヘッダーからのデータを MapForce によりサポートされる他の書式にマップするには、「ヘッダー」構造ノードをマッピング内のターゲットシーケンスに接続します。例えば、「**ヘッダー**」シーケンス(とその子) をCSV コンポーネントの「**行**」シーケンス(とその子) に接続する場合、ヘッダーはCSV ファイル内の行に対応します。

ボディ は、HTTP メッセージのエンティティボディを表示しています。このレベルでのデータはエンコードされたバイナリですので、直接のインタラクションはMapForce mime 関数を必要とします。通常 W eb サービスがXML または JSON などの構造化されたデータを期待または返す場合、**ボディ** とインタラクトする必要はありません。**ボディ** への直接のマッピングデータ、または**ボディ** からのマッピングデータは、構造化されていないコンテンツを期待する、または返すW eb サービスを呼び出す場合のみ必要となります。

> デフォルトでは、**ボディ** アイテムはHTTP ステータスコードが200 から299 への範囲の場合、結果を解析するように構成されています。これは、マッピングが299 より多きステータスコードのためにエラーを返すことを意味します。また、レスポンスを解析できない場合、またはW eb サービスが接続エラー、またはDNS 精度の問題により呼び出されない場合にエラーが返されます。

一部の場合では、HTTP ステータスコードが299 より大きい場合でも、マッピングがエラーをスローしないことがあります。これを行うには、レスポンス**ボディ** の横の ボタンをクリックし、HTTP ステータスコードの範囲を変更します。または、複数のレスポンス**ボディ** アイテムを作成することができます。複数の**ボディ** アイテムの作成は、W eb サービスにより返されるHTTP ステータスコードに従いマッピングを条件づけて処理する場合に役に立ちます。これを行うには、レスポンスパートの**ボディ** を右クリックし、「**前/後にボディノードを追加する**」をコンテキストメニューから選択します。例えば、2つの**ボディ** アイテムを作成することができます:

1. 全てのステータスコード **ボディ** は成功の範囲(200 から299) です。状態コードの範囲を ボタンをクリックして構成することができます。このシナリオのために、**ボディ** 構造を通常、JSON およびXML ファイルなどの成功する出力にマップします。

(C) 2015-2021 Altova GmbH

2. 他のステータスコードのためのボディ は300 と599 の範囲内です。W eb サービスの一部は、ボディ内のエラーステータスコードについての追加の情報を提供する可能性があります。エラーの理由を確認するには、エラーのあるボディを(例えば、文字列などの)成功する出力以外にマップすることもできます。

複数のボディアイテムが存在する場合、MapForce はアイテムを上から下に評価します。優先順位を変更するには、ボディアイテムを右クリックし、「上に移動」または「下に移動」をコンテキストメニューから選択します。HTTP レスポンスを条件付きで処理するための詳細に関しては、HTTP レスポンスを条件付きで処理するを参照してください。

ジェネリックな HTTP W eb サービスの呼び出しを含むマッピングは、他のマッピング同様に実行することができます。具体的には以下の通りです:

- 手動では、MapForce で**出力** タブをクリックします。この場合、マッピングの呼び出し結果はすぐに「**出力**」ペインで使用することができるようになります。上記のサンプル内のようにマッピングに複数の出力コンポーネントが存在する場合、マッピングに戻る前に希望するコンポーネント上の「**プレビュー**」 ボタンをクリックします。
- コマンドライン、または MapForce Server (https://www.altova.com/ja/mapforce-server) と共に、コマンドラインまたは API 呼び出しを使用します。これは、マッピングをマッピング実行可能ファイルに最初にコンパイルすることを必要とします (MapForce Server 実行可能ファイルにマッピングをコンパイルするを参照してください)。
- FlowForce Server コントロール下で作動する MapForce Server を使用した再帰的なジョブとして (https://www.altova.com/flowforceserver)。マッピングを FlowForce Server にデプロイするも参照してください。

ジェネリックな W eb サービスの呼び出しの作成方法のステップバイステップのサンプルに関しては、以下を参照してください:

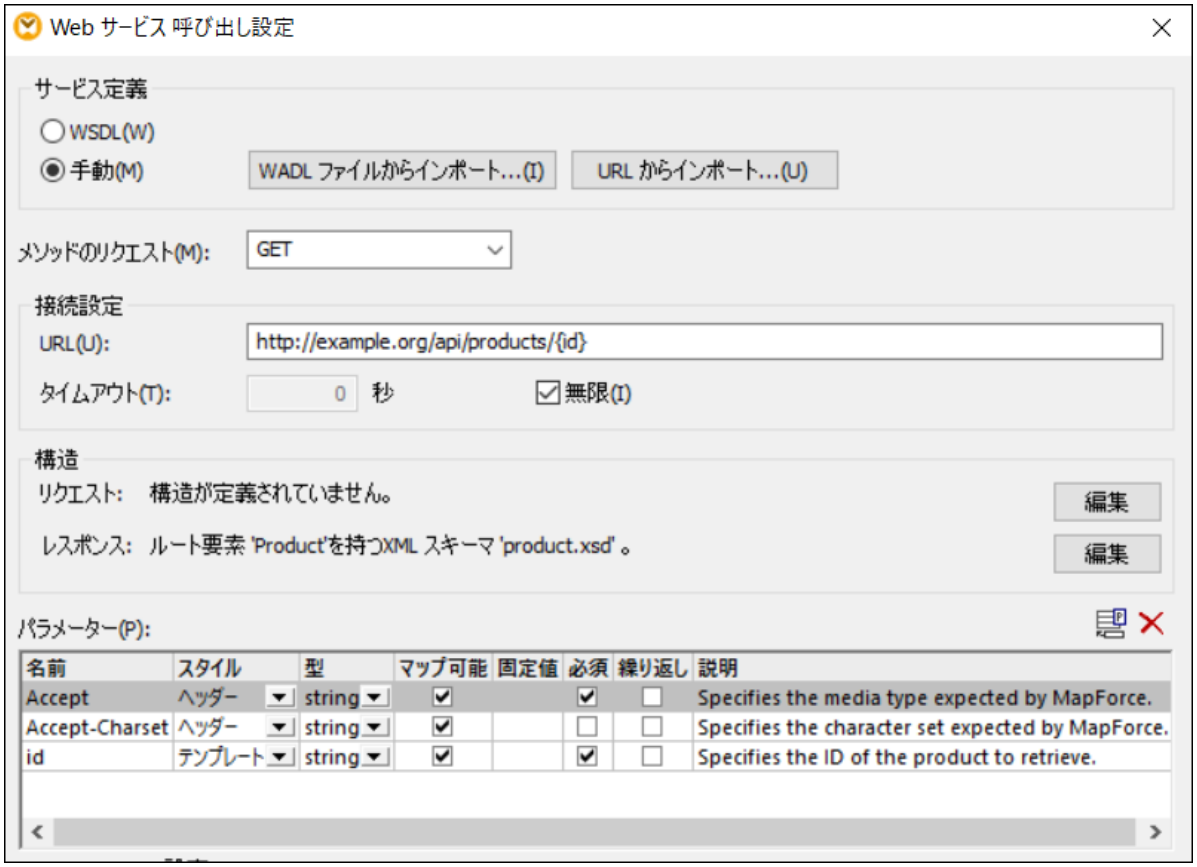
- 例: REST-スタイル W eb サービスの呼び出し
- 例: RSS フィードからのデータのマッピング

9.1.1 Web サービスパラメーターの定義

URL パラメーターを持つ W eb サービスを呼び出す必要がある場合、W eb サービス呼び出し設定 ダイアログボックス内で パラメーターが定義される必要があります。このダイアログボックスにより、カスタムヘッダーを持つ W eb サービスリクエストを構成することができます。以下で詳細に説明されている通り、パラメーターの値を *マップ可能* (マッピングにより提供される値) または *固定値* (ダイアログボックスで定義されている値)に構成することができます。

W eb サービスパラメーターを定義する方法:

1. マッピングの W eb サービスコンポーネントのタイトルバーをダブルクリックします。または、右クリックして「**プロパティ**」を選択します。W eb サービス呼び出し設定 ダイアログボックスが開かれます。



2. 「**サービス定義**」から「**手動**」を選択します。上記の通り、手動で**パラメーター**を入力、または代替として、既存のURL 、またはW ADL ファイルからインポートすることができます。必要に応じて更に変更することのできる、「**パラメーター**」テーブル内でインポートされたテーブルを使用できるようになります。
3. 「**パラメーターの追加**」( ) と「**パラメーターの削除**」( ) ボタンをそれぞれ使用します。

「**パラメーター**」テーブルには以下を意味します:

名前	URL パラメーターの名前を指定します。パラメーター名は一意で文字、数字、ピリオド (.) 、ハイフン (-) 、アンダースコア (_) から構成されている必要があります。パラメーター名内で空白文字は許可されていません。
スタイル	URL パラメーターの構文 (スタイル) を指定します。 W eb サービスを呼び出す際に「ヘッダー」スタイルを使用して、HTTP ヘッダーにパラメーターを追加してください。例えば、値 `text/xml` を持つパラメーター `Accept` を追加することは、W eb サーバーにMapForce が XML 書式のレスポンスを期待することを通知する `Accept: text/xml` ヘッダーを指定することと同じです。HTTP ヘッダーに関する詳細は、https://www.iana.org/assignments/message-headers/message-headers.xhtml を参照してください。 `?key=value&key=value` 書式を使用してキー値ペアを定義するURL パラメーターのための「クエリ」スタイルを使用します: 例: `http://example.org/api/products?sort=asc&category=1&page=1`

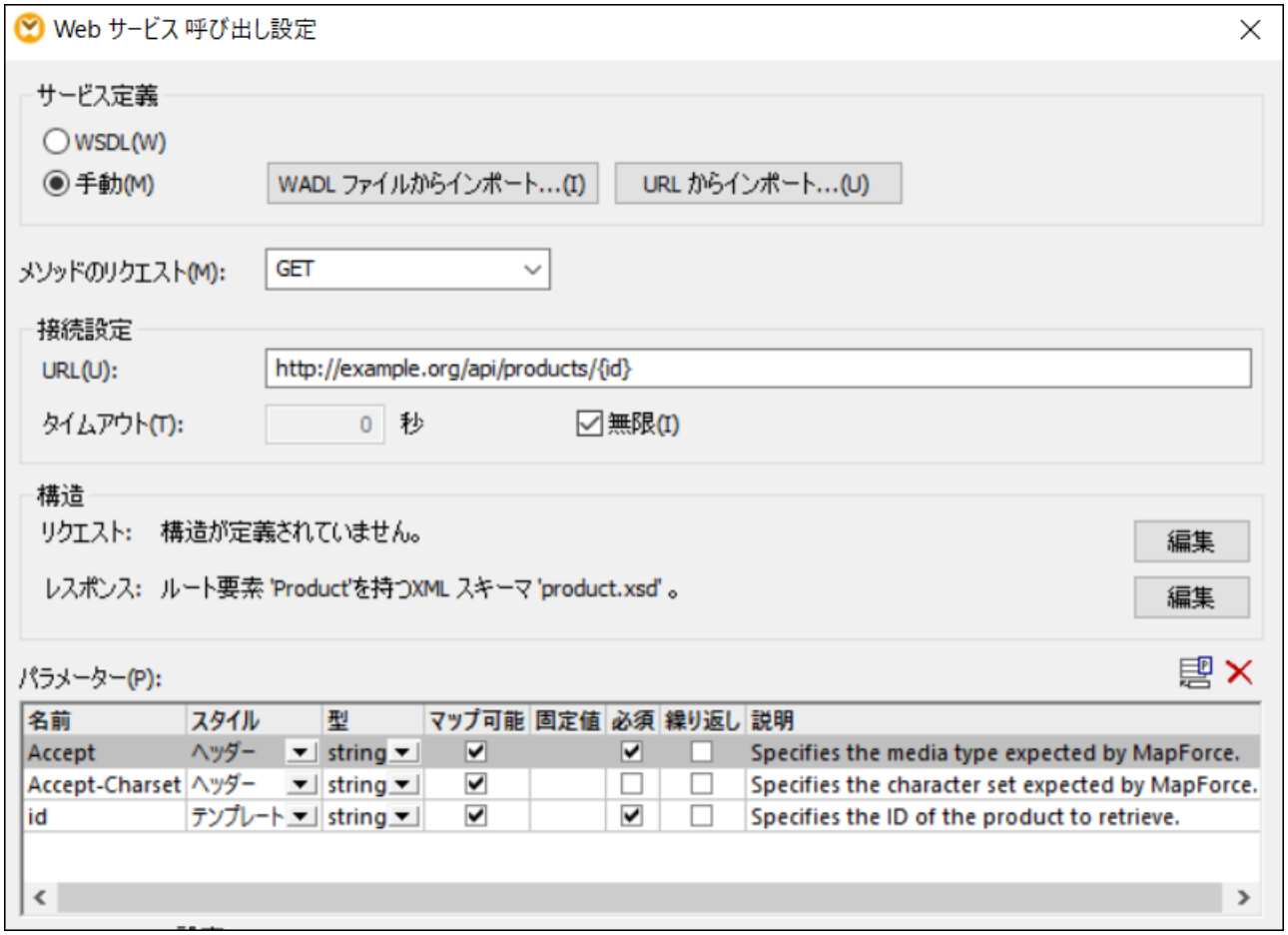
(C) 2015-2021 Altova GmbH

	中かっこ内に囲まれたURL パラメーターのために「テンプレート」スタイルを使用します。例: `http://example.org/api/products/{id}` 。このようなパラメーターのために、MapForce はRFC 6570 ルールに基づき値をエスケープします (https://tools.ietf.org/html/rfc6570)。 フォーマット `;key=value;key=value;` 内のキー値ペアを定義するURL パラメーターのために「マトリックス」スタイルを使用します。 例: `http://example.org/api/products;sort=asc;category=1;page=1;` Boolean マトリックス パラメーターを使用するには、スタイルを「マトリックス」に設定して、「boolean」を入力します (次のオプションを参照してください)。
型	パラメーター (文字列、整数、日付など) のデータの型を指定します。XML スキーマ型であることができます。W eb サービスの呼び出しが発生すると、文字列ではない値は文字列に変換されることに注意してください。しかし、間違った値と共にW eb サービスを呼び出そうと試みた場合 MapForce が変換エラーメッセージを表示するために、型を設定することは意味があります。
マップ可能	マッピングから値をこのパラメーターにパスする場合、このチェックボックスを選択します。このオプションは「固定値」オプションと相互に排他的です。
固定値	パラメーターの値を指定します。パラメーターに定数地が存在する場合のみ、適用することができます。パラメータがマップ可能な場合は適用することができません (前のオプションを参照)。
必須	W eb サービスによりパラメーターが必要とされている場合、このチェックボックスを選択してください。必要とされ、マップ可能なパラメーターのために、MapForce は検証チェックを強制します (すなわち、パラメータが価を持たない場合、エラーメッセージが表示されます)。
繰り返し	パラメーターが単一の値、または複数の値を持つかを指定します。この設定は、マップ可能なパラメーターにのみ適用可能です。単一のパラメーターを使用して 同じ W eb サービスの呼び出し内の複数の値にパスすることができます。 チェックボックスを選択して場合、値のシーケンスを単一の値の代わりにマッピング上のパラメーター構造ノードに接続することができます。MapForce は値のシーケンスをパラメーターのスタイルにより以下のように処理します: • 「テンプレート」パラメーターに関しては、値はW eb サービスにコンマで区切られて提供されます。例: `http://example.org/api/products/1,2,3` • 「クエリ」パラメーターに関しては、パラメーター名は各値のために繰り返されます。例: `http://example.org/api/products?color=red&color=green&color=blue` • スタイルが「マトリックス」の場合、複数の値はコンマで区切られて表示されます。例: `http://example.org/api/products;color=red;color=green;color=blue;size=big;size=small;` • スタイルが「ヘッダー」の場合、HTTP ヘッダーは各値のために繰り返されます。
説明	パラメーターの任意の説明を指定します。パラメーターがマップ可能な場合、ここに入力されている説明がマップコンポーネントとしてマッピングアイテムの横に説明として表示されます。

サンプル 1: ヘッダーとテンプレート パラメーター

下で説明されるW eb サービスは、製品をHTTP GET メソッドを使用して識別子 (`id`) により抽出します。W eb サービスのURL は、`id` パラメーターを中かっこ内で指定します。`id` パラメーターはパラメーターテーブルに存在し、すらいる「テンプレート」を持っています。また「マップ可能」でもあります: これにより、パラメーターがマッピング上で (例えば、データベース、ファイル、または定数から抽出された) 実際の値の `id` に接続することのできる構造ノードとして表示されます。マッピング実行ランタイムでは、このパラメーターは実際の値と置き換えられます。ですから、値が「1」の場合、URL は `http://example.org/api/products/1` になります。

マッピングから値の代わりに定数 `id` 値を提供するには、チェックボックス「マップ可能」をクリアして、「固定値」列に値を入力します。



パラメーター`Accept` と`Accept-Charset` は「ヘッダー」スタイルが選択されています。これらのパラメーターは、カスタムリクエストヘッダーを持つWeb サービスを呼び出すために使用されます。 ヘッダー値を提供するために2つの方法があります:

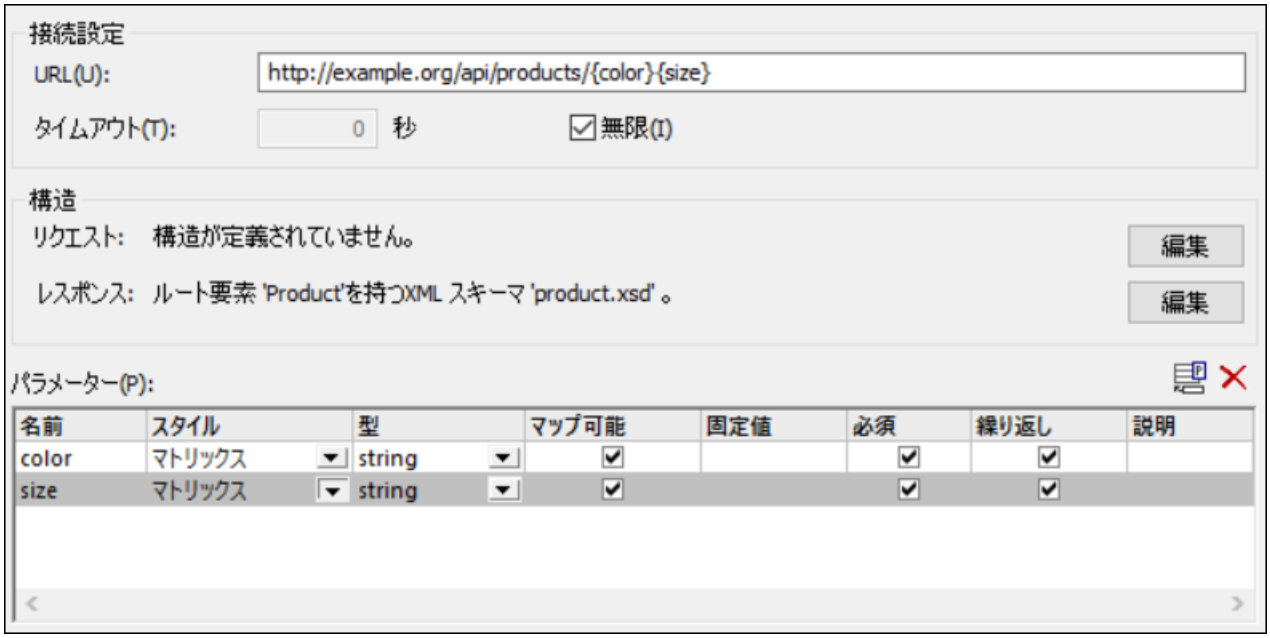
- オプション「**マップ可能**」をチェックしたままで、マッピングからカスタムのヘッダー値を提供します。または、
- オプション「**マップ可能**」をクリアし、「固定値」列内に直接値を入力します。

サンプル2: マトリックスパラメーター

下で説明されているWeb サービスは、引数として与えられている色とサイズに一致する製品のリストを抽出します。パラメーターのスタイルは「マトリックス」であるため、URL の内部、およびマッピングテーブル内のプレースホルダとして定義されています。パラメーターはマップ可能で、「繰り返し」オプションがチェックされていることに注意してください。これは例えば、テキストファイル、XML ノード、またはデータベース列の内部の行のリストなど)値がマッピング上の値のシーケンスから読み取られ、ランタイムでWeb サービスに提供されることを意味します。ですから、下記のようなURL は以下のようになります:

`http://example.org/api/products/;color=red;color=blue;size=big;size=small` :下記のような例は、`red` と`blue` を色として、`big` と`small` をサイズとしてマッピングに提供しています。

(C) 2015-2021 Altova GmbH



URL が`http://example.org/api/products;color=red,blue;size=big,small` である必要がある場合、以下を行います:

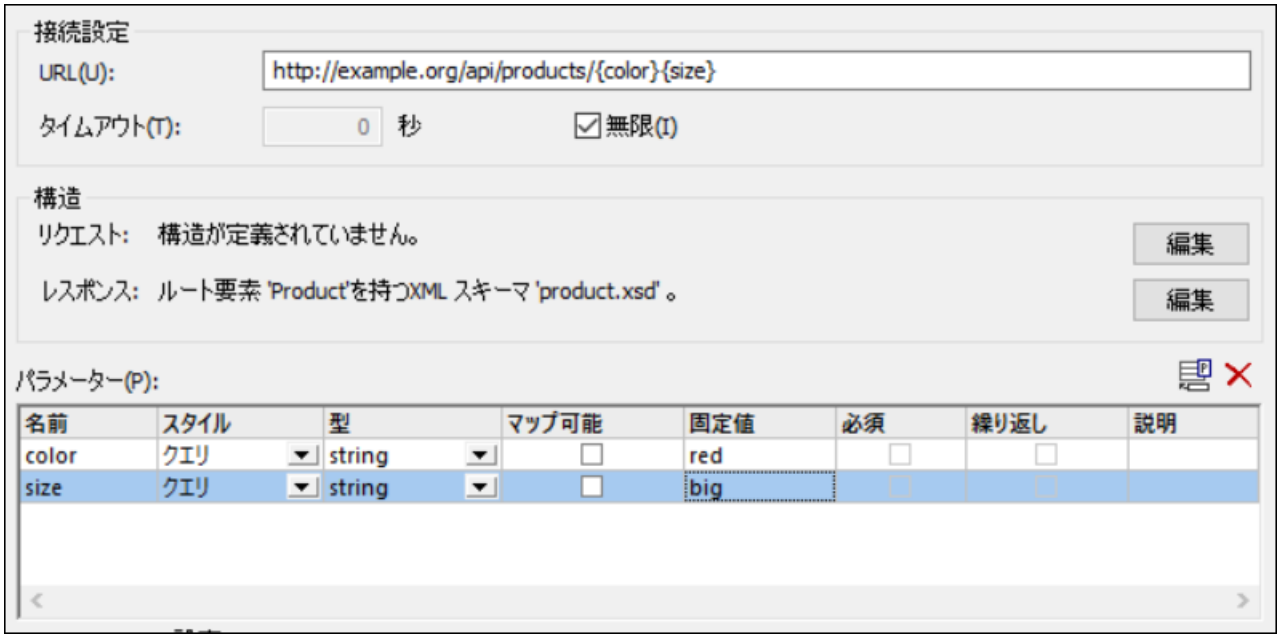
1. 以下のとおりURL を入力します: `http://example.org/api/products;color={color};size={size}`
2. 「パラメーター」テーブル内で、パラメーターのスタイルを「マトリックス」に変更します。

URL が`http://example.org/api/products?color=red,blue&size=big,small` である必要がある場合、以下を行います:

1. 以下のとおりURL を入力します: `http://example.org/api/products?color={color}&size={size}`
2. 「パラメーター」テーブル内で、パラメーターのスタイルを「マトリックス」に変更します。

サンプル3: クエリパラメーター

下で説明されるW eb サービスは、引数として提供される色とサイズに一致する製品のリストを抽出します。今回は、スタイル「クエリ」が使用されています。このスタイルに関しては、URL 内のプレースホルダとしてパラメーターを定義する必要はありません。ですからこれらは「パラメーター」テーブル内でのみ定義されています。このサンプルでは、パラメータの値は固定されており、マッピングから提供される必要はないものとします。これを達成するために、パラメーターの値は、「固定値」に入力され、「マップ可能」オプションのチェックは解除されています。このためマッピングランタイムでは、下のURL は`http://example.org/api/products?color=red&size=big` になります。



9.1.2 Web サービスパラメーターを URL からインポートする

Web サービスへのURL がパラメーターを使用する場合、MapForce にURL を解析するように命令し、パラメーターを自動的にインポートすることができます。このオプションを使用する場合、テンプレート、マトリックス、またはクエリパラメーターから抽出されるURL は、必要に応じて更に操作することのできるWeb サービス呼び出し設定 ダイアログボックス内の「パラメーター」テーブル内で使用できるようになります。

異なるURL スタイルの例は以下の通りです:

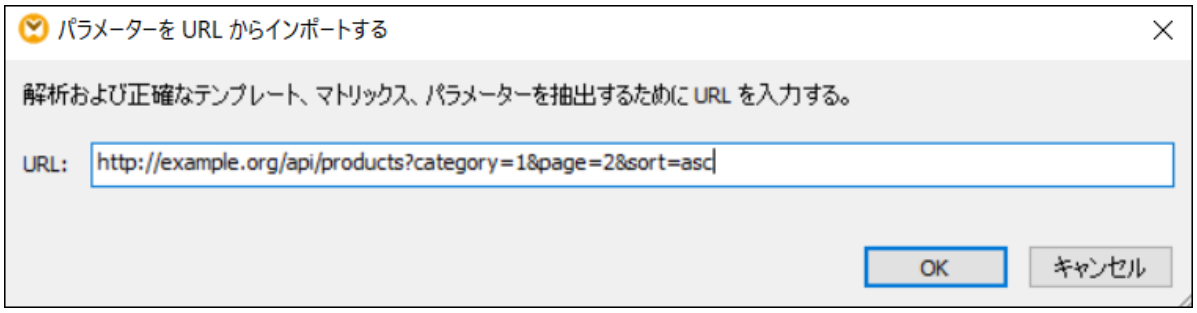
テンプレートパラメーターを持つURL のサンプル	`http://example.org/api/products/{id}`
マトリックスパラメーターを持つURL のサンプル	`http://example.org/api/products;sort=asc;category=1;page=1`
クエリパラメーターを持つURL のサンプル	`http://example.org/api/products?category=1&page=2&sort=asc`

メモ URL を解析可能にするために"http://" または"https://" で開始する必要があります。

URL からWeb サービスパラメーターをインポートする方法:

1. (Web サービス呼び出し設定を参照してください)。
2. 「サービス定義」から「手動」を選択します。
3. 「URL のからインポート」をクリックします。

(C) 2015-2021 Altova GmbH



4. Web サービスへのURL をテキストボックスに入力、または張り付け、「OK」をクリックします。

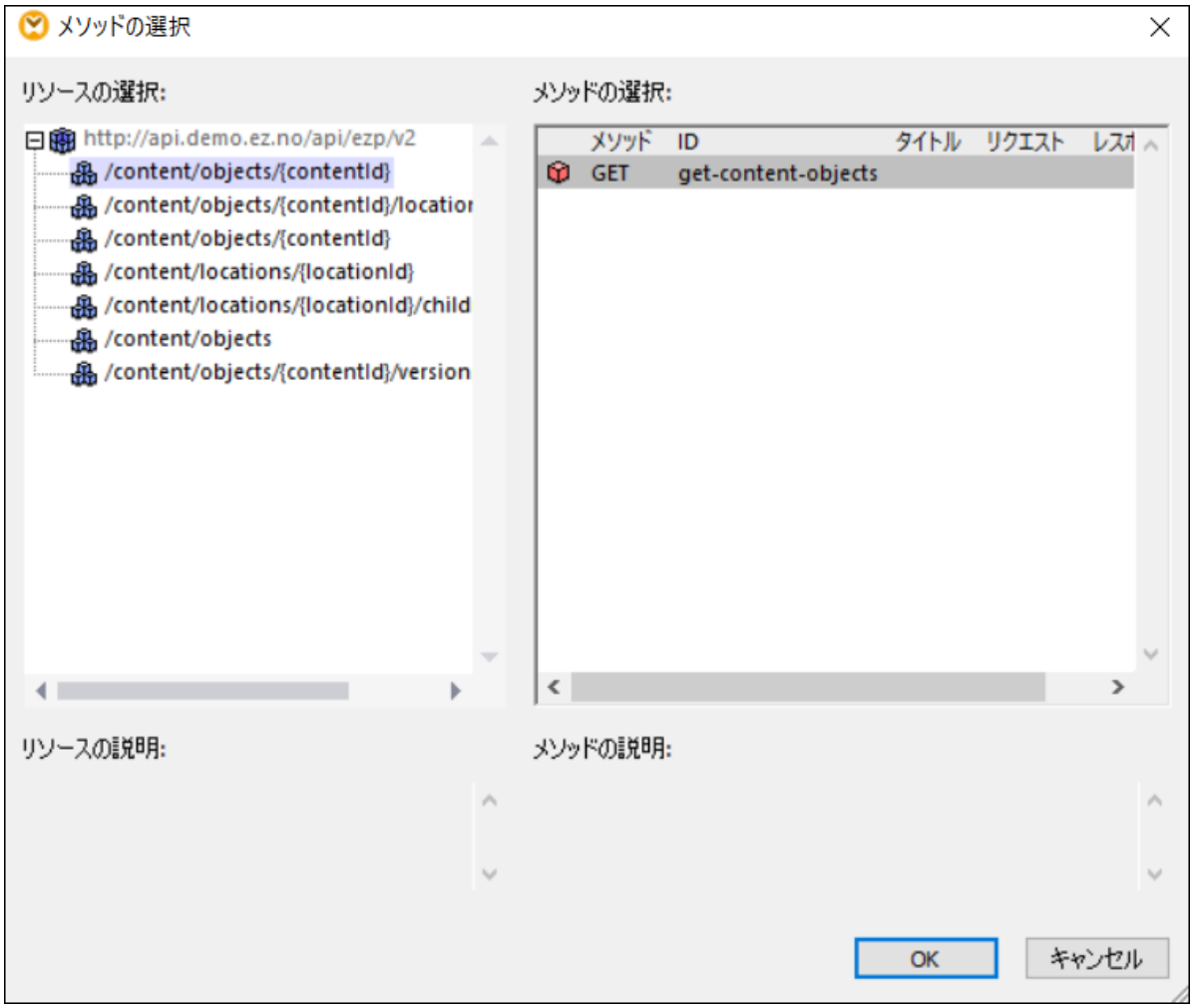
9.1.3 Web サービス情報を WADL からインポートする

Web Application Description Language (WADL) はRESTful Web サービスを含むアプリケーション内で使用されるレスポンスとリクエストのコンテンツを表示する方法です。MapForce はWADL バージョン 2009 (https://www.w3.org/Submission/2009/SUBM-wadl-20090831/) からのWeb サービス情報のインポートをサポートしています。

Web サービスがWADL ファイルを持つ場合、手動で入力する代わりにWADL ファイルからWeb サービス定義をインポートすることができます。WADL ファイルから情報をインポートする際、パラメーターとXML メッセージボディ構造はWeb サービス呼び出し設定 ダイアログボックス上で自動的に作成されます。インポートの後、パラメーターとレスポンス/リクエスト構造を手動で必要に応じて変更することができます。これは基になるWADL ファイルに影響を及ぼしません。

WADL ファイルからWeb サービス情報をインポートする方法:

1. Web サービス呼び出し設定 ダイアログボックスを開きます (Web サービス呼び出し設定を参照してください)。
2. サービス定義から、「手動」を選択します。
3. 「WADL ファイルからインポート」をクリックして、WADL ファイルを参照します。



4. ダイアログボックスから、リソースとメソッドを選択し、「OK」をクリックします。

9.1.4 リクエスト構造の定義

Web サービスが、リクエストメッセージのボディパート内にコンテンツを期待する場合、リクエスト構造を以下の1つに設定することができます:

- 構造無し
- XML
- JSON
- プロトコルバッファーファイル。プロトコルバッファーを参照してください。

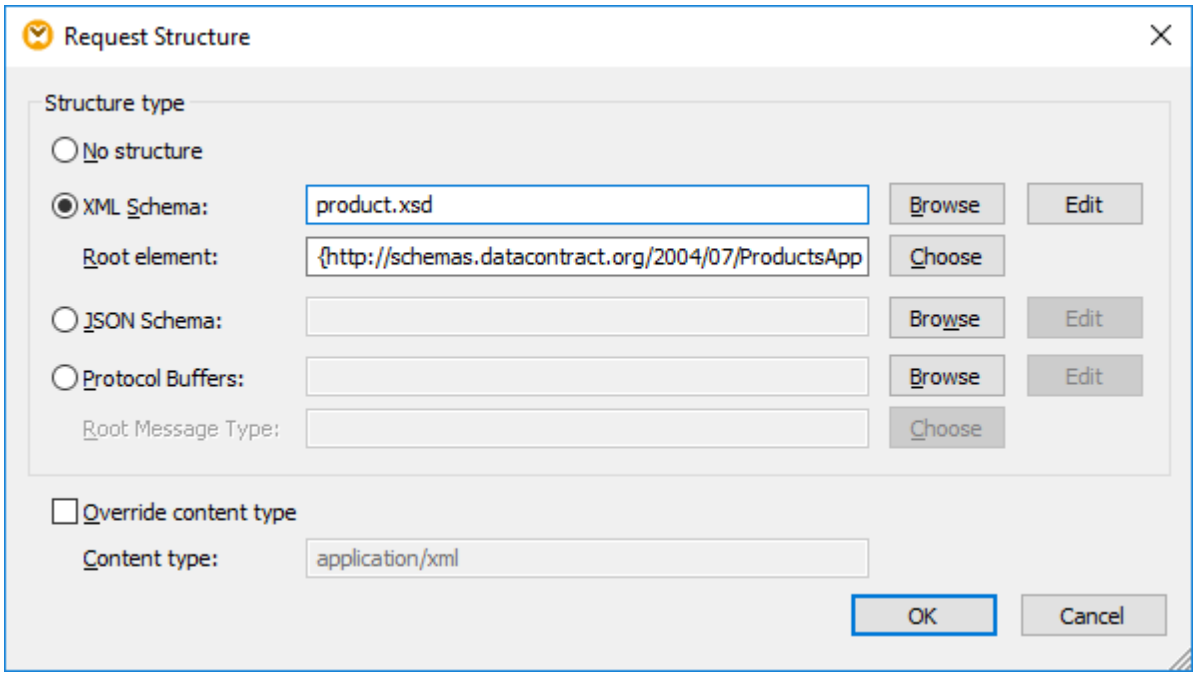
メモ MapForce を使用して Web サービスの呼び出しが作成された場合、標準の JSON をリクエストまたはレスポンスの構造として使用することができます。JSON ライン、または JSON5 構造はサポートされていません。

ここで選択される構造は **Body** アイテムの子としてマッピング上で表示されるマップ可能なアイテムを決定します。

(C) 2015-2021 Altova GmbH

リクエスト構造を定義する方法:

1. Web サービス呼び出し設定 ダイアログボックスを開きます (Web サービス呼び出し設定を参照してください)。
2. サービス定義から、「手動」を選択します。
3. 「構造」から「リクエスト」の横の「編集」をクリックします。



ヒント: このダイアログボックスをショートカットとして表示するには、マッピング上の Web サービスコンポーネントのリクエストパート内の ボタンをクリックします。

4. 次のオプションから1つ選択してください:

   a) Web サービスがボディパート内にコンテンツを期待しない場合、または生ボディ (MIME エンティティ) を直接マッピングから期待しない場合、「構造無し」を選択してください。

   b) XML のために、Web サービスのプロバイダーからのリクエストから、XML または DTD スキーマを取得します。XML インスタンスファイルを選択した場合、有効なスキーマ参照が必要になります。XML スキーマの一部は、グローバル宣言 (親が `schema` 要素である要素) を持つ要素を定義します。このようなスキーマの場合、スキーマ内でどの要素が MapForce 内のマッピング構造のルート要素であるかを選択することができます。これを行うには、「選択」をクリックして、表示されるダイアログボックス内で希望されるルート要素を選択します。

   c) JSON のために、Web サービスのプロバイダーからのリクエストの JSON スキーマを取得します。または、XMLSpy などのツールを使用してサンプル JSON ファイルから生成します。

ヒント: サンプル XML または JSON リクエストファイルが存在するが、スキーマファイルが存在しない場合、XMLSpy を使用して、スキーマファイルを生成することができます。必要な場合、XMLSpy は XML からのインスタンスファイルを JSON に変換、またはその逆に変換することができます。

   b) プロトコルバッファーファイルのために、バイナリファイルを説明する `.proto` ファイルを Web サービスプロバイダーから取得します。「参照」をクリックしてファイルを選択します。ルートメッセージ型 フィールドはプロトコルバッファーファイルに複数のメッセージ型が含まれる場合のみ有効です。「選択」をクリックしてエオズのルートにあるメッセージ型を選択します。

5. 任意で、MapForce により明示的に設定される **Content-Type** ヘッダーをオーバーライドするには、「**コンテンツ型チェックボックスをオーバーライド**」を選択します。例: `Content-Type: text/html; charset=utf-8`.

マッピングを保存する場合、コンポーネント設定内でチェックボックス「**mfd へ対して全てのファイルパスを相対的に保存する**」が選択されている場合、このダイアログボックス上の絶対パスは、マッピングデザインファイル(.mfd)に対して相対的になります。Web サービスの呼び出しの設定を参照してください。

9.1.5 レスポンス構造の定義

Web サービスが、リクエストメッセージのボディパート内にコンテンツを返す場合、レスポンス構造を以下の1つに設定することができます:

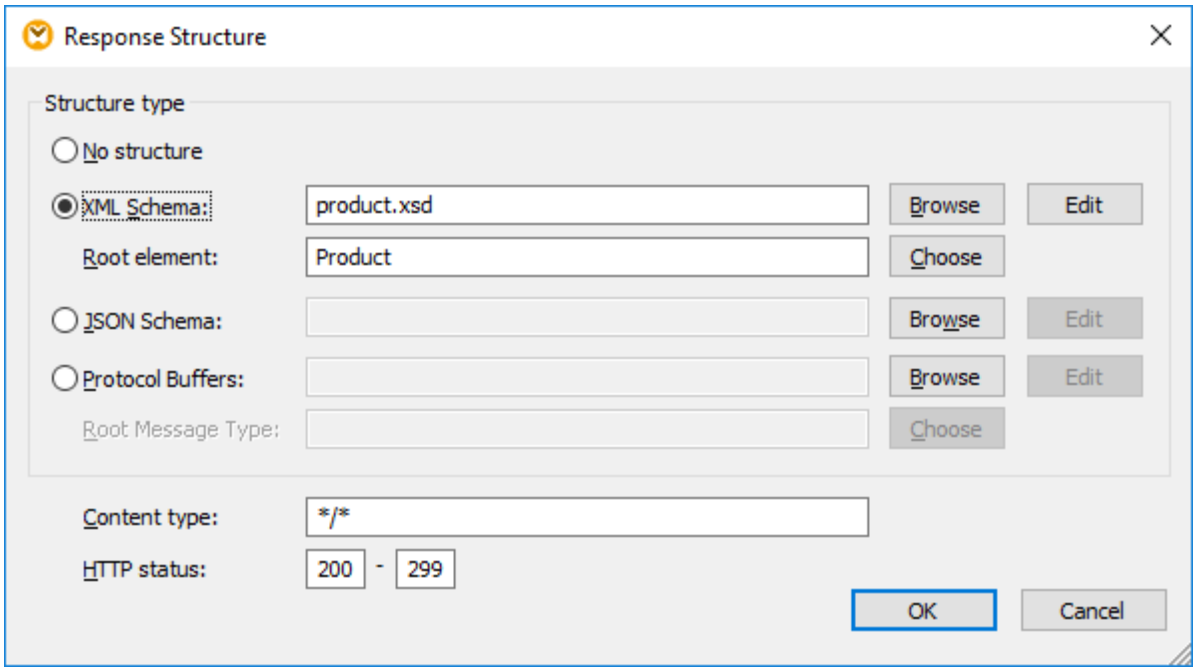
- 構造無し
- XML
- JSON
- プロトコルバッファーファイル、プロトコルバッファーを参照してください。

メモ MapForce を使用して Web サービスの呼び出しが作成された場合、標準の JSON をリクエストまたはレスポンスの構造として使用することができます。JSON ライン、または JSON5 構造はサポートされていません。

ここで選択される構造は **Body** アイテムの子としてマッピング上で表示されるマップ可能なアイテムを決定します。

レスポンス構造を定義する方法:

1. (Web サービス呼び出し設定を参照してください)。
2. サービス定義から、「**手動**」を選択します。
3. 「**構造**」から「**レスポンス**」の横の「**編集**」をクリックします。



ヒント: このダイアログボックスをショートカットとして表示するには、マッピング上のWeb サービスコンポーネントのリクエストパート内の ボタンをクリックします。

4. 次のオプションから1つ選択してください:

   Web サービスがボディパート内にコンテンツを返さない場合、Web サービスがボディパート内にコンテンツを期待しない場合、または生ボディ(MIME エンティティ) を処理する場合、「**構造無し**」を選択してください。

   b) XML のために、Web サービスのプロバイダーからのリクエストから、XML またはDTD スキーマを取得します。XML インスタンスファイルを選択した場合、有効なスキーマ参照が必要になります。XML スキーマの一部は、グローバル宣言(親が `schema` 要素である要素).を持つ要素を定義します。このようなスキーマの場合、スキーマ内でどの要素がMapForce 内のマッピング構造のルート要素であるかを選択することができます。これを行うには、「**選択**」をクリックして、表示されるダイアログボックス内で希望されるルート要素を選択します。

   c) JSON のために、Web サービスのプロバイダーからのリクエストのJSON スキーマを取得します。または、XMLSpy などのツールを使用してサンプルJSON ファイルから生成します。

ヒント: サンプルXML またはJSON リクエストファイルが存在するが、スキーマファイルが存在しない場合、XMLSpy を使用して、スキーマファイルを生成することができます。必要な場合、XMLSpy はXML からのインスタンスファイルをJSON に変換、またはその逆に変換することができます。

   b) プロトコルバッファーファイルのために、バイナリファイルを説明する `.proto` ファイルをWeb サービスプロバイダーから取得します。「**参照**」をクリックしてファイルを選択します。**ルートメッセージ型** フィールドはプロトコルバッファーファイルに複数のメッセージ型が含まれる場合のみ有効です。「**選択**」をクリックしてエオズのルートにあるメッセージ型を選択します。

レスポンスMIME エンティティをメディア(コンテンツ) 型とHTTP ステータスコードによりフィルターする必要がある場合、オプションで**コンテンツ型** とHTTP **ステータス** テキストボックスをフィルします。これは、Web サービスの呼び出しにより返されるHTTP ステータスコード上の条件を処理する場合に役立ちます。サンプルに関してはHTTP レスポンスを条件付きで処理するを参照してください。

デフォルトで、**コンテンツ型** フィルターが `*/*` に設定されている場合、メディア型に一致するものに設定してください。デフォルトのHTTP ステータスコードは**200** から**299** です。Web サービスの呼び出しが、明示的に処理されない他のステータスコードの場合、マッピングの実行は失敗、またはエラーを返します。

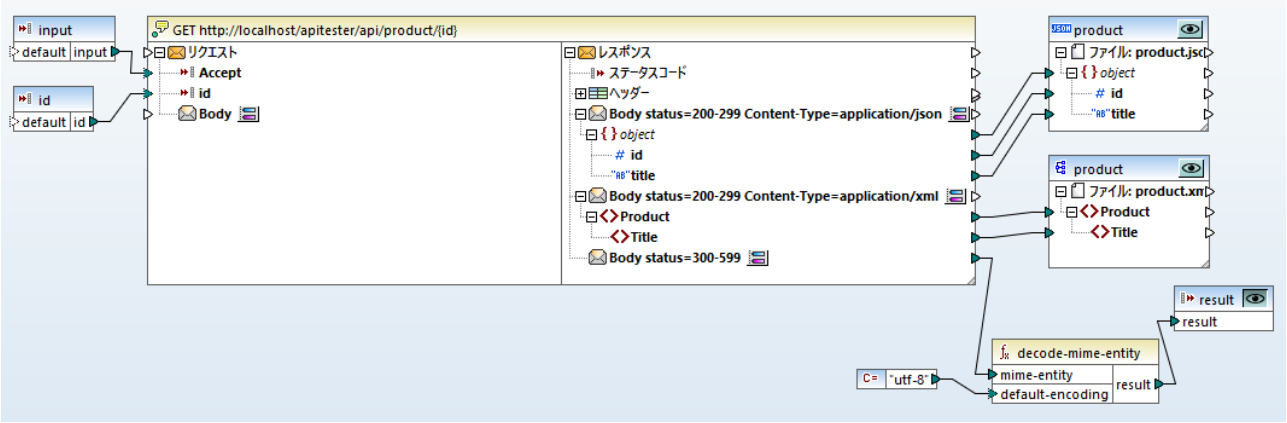
> マッピングを保存する場合、コンポーネント設定内でチェックボックス「**mfd へ対して全てのファイルパスを相対的に保存する**」が選択されている場合、このダイアログボックス上の絶対パスは、マッピングデザインファイル(.mfd) に対して相対的になります。Web サービスの呼び出しの設定を参照してください。

9.1.6 HTTP レスポンスを条件付きで処理する方法

Accept HTTP リクエストヘッダーの値に従いWeb サービスがレスポンスボディにJSON またはXML コンテンツ型を返すと仮定します。更に、エラーの場合、(ステータスコード > 299) Web サービスは、定義されていないコンテンツ型を返す場合があります。これはHTTP ステータスコードとコンテンツ型をベースに条件付きでレスポンスを扱う場合に特に役に立ちます。

JSON とXML レスポンスを同じマッピング内のWeb サービスから処理するには、複数のボディアイテムを作成し、コンポーネント上のボディを右クリックし、コンテキストメニューから「**ボディノードを後に追加する**」を選択します。

マッピングでは、上記のシナリオのために構成されている複数のボディアイテムを持つWeb サービスコンポーネントは以下のようになります:



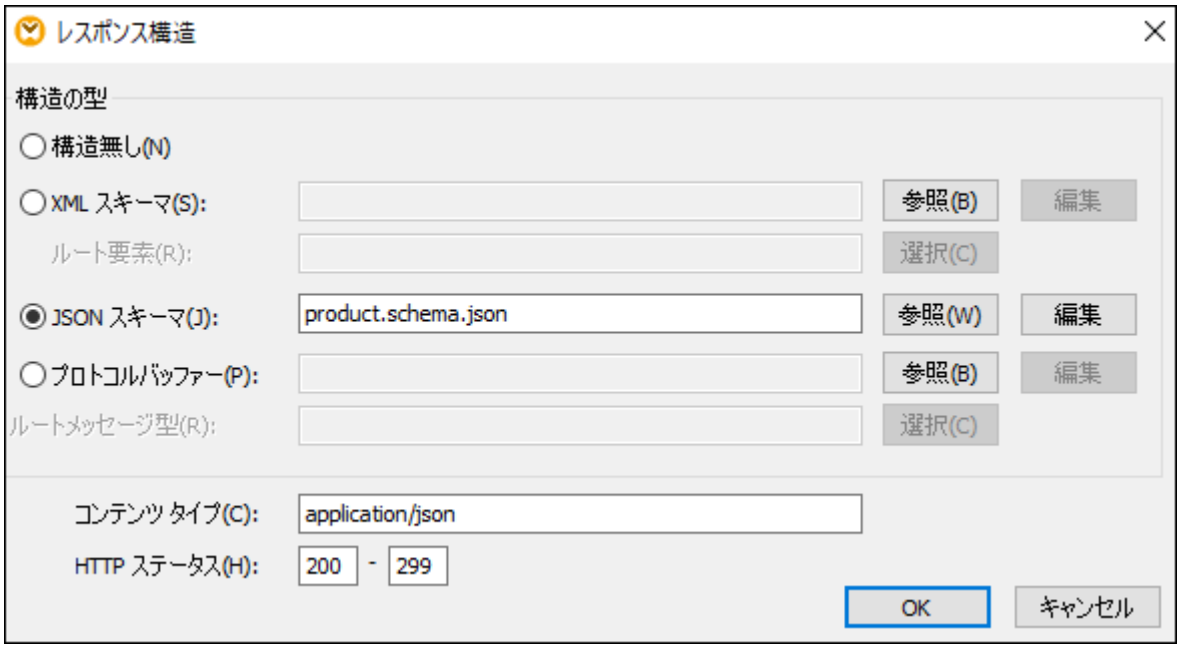
上で説明されるマッピングは、2 つのパラメーターを取ります。最初のパラメーターは期待されるコンテンツ型を提供し、値は `application/json` または `application/xml` であることができます。2 番目のパラメータは取得される製品の数値 ID を提供します。

値 `application/json` を入力パラメーターとして提供すると、JSON ターゲットコンポーネントのプレビュー ボタンをクリックします。それ以外の場合、入力パラメータの値は `application/xml` になり、XML ターゲットコンポーネントのプレビュー ボタンをクリックします。Web サービスの呼び出しが成功すると、対応する出力ファイル(JSON または XML) が生成されます。

Web サービスの呼び出しが 300 (エラー) より大きいステータスコードを返す場合、HTTP レスポンスは、3 番目のボディアイテムに達成します。コンテンツの正確な型が既知ではないため、decode-mime-entity 関数を使用してデコードします。Web サービスはレスポンスヘッダー内の文字列セット値を適用しません。このため、定数の値 **utf-8** がデコード関数に接続されています。

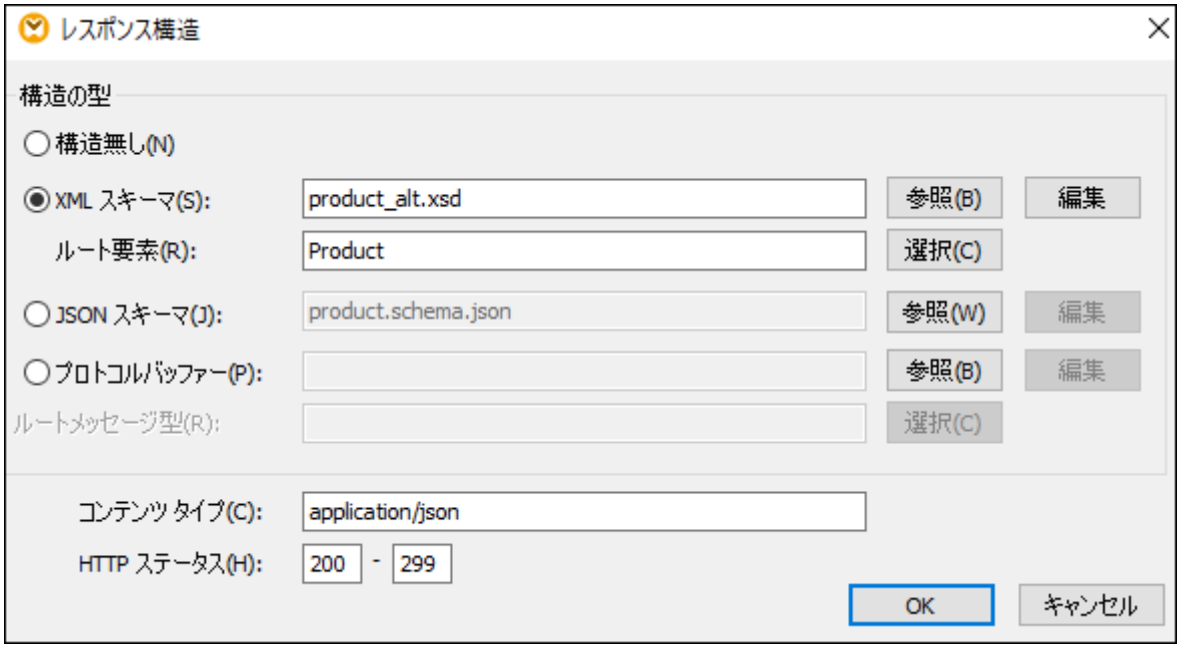
各ボディの横にある ボタンをクリックしてコンポーネント上の 3 つのレスポンスボディアイテムを構成することができます:

1. ステータスコードが 200 から 299 の場合、最初のボディは返されたコンテンツを `application/json` として処理します。期待される JSON レスポンス構造が設定されます。

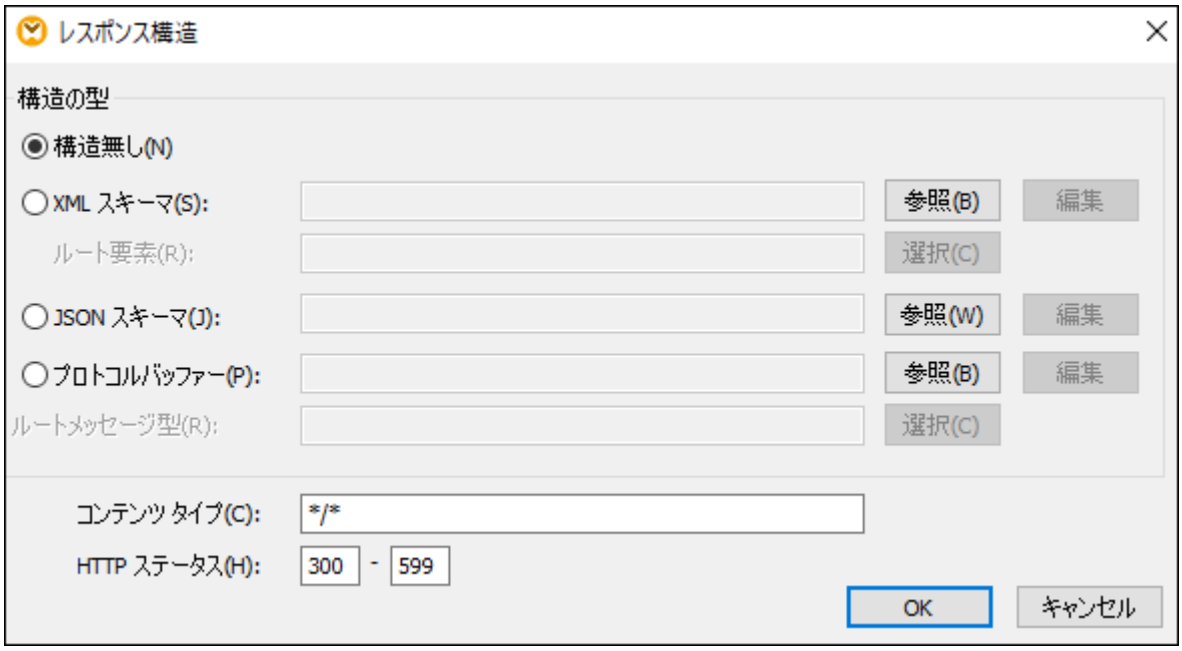


(C) 2015-2021 Altova GmbH

2. ステータスコードが200 から299 の場合、2番目のボディは現在のコンテンツを `application/xml` として処理します。また、期待されるXML レスポンス構造のスキーマが設定されます。



3. Web サービスがエラーのステータスコード (300 と599 の間)を返す場合、3番目のボディは `*/*` として返されたコンテンツを処理します。



9.2 Web サービス呼び出しの追加 (SOAP)

このトピックでは、WSDL-スタイルWeb サービスへ呼び出しをマッピングする方法について説明します。ジェネリックなWeb サービスをマッピングに追加する方法に関しては、Web サービスの呼び出しの追加 (REST-Style) を参照してください。

マッピングエリアに WSDL-スタイルのWeb サービスの呼び出しを追加する前に、接続するWeb サービスを説明する、Web サービス定義言語 (.wsdl) が搭載されていることを確認してください。

WSDL スタイルWeb サービスに呼び出しを追加する方法:

1. 「**挿入**」メニューから「**Web サービス関数**」をクリックします。(または、「**Web サービス関数の挿入**」 ツールバーボタンをクリックします)。
2. サービスの定義から、このオプションが既に選択されていない場合、**WSDL** をクリックします。
3. WSDL 設定から、「**参照**」をクリックし、Web サービスの定義言語 (.wsdl) ファイルを選択します。WSDL (1.0 と2.0) のバージョンそれぞれ1つずつの2つのサンプル.wsdl ファイルが、MapForce Example フォルダー内にあります。対応するパスは、以下のとおりです: **<マイドキュメント>\Altova\MapForce2021\MapForceExamples\TimeService\TimeService.wsdl** と**<マイドキュメント>\Altova\MapForce2021\MapForceExamples\TimeServiceWsdl2\TimeService20.wsdl**。
4. .wsdl ファイルが複数のサービス、オペレーションとエンドポイントを定義する場合、「**選択**」をクリックして、必要なオプションを選択します。
5. 必要であれば、他の設定を定義します(Web サービス呼び出し設定を参照してください)。
6. 「**OK**」をクリックします。

例: SOAP Web サービスの呼び出しも参照してください。

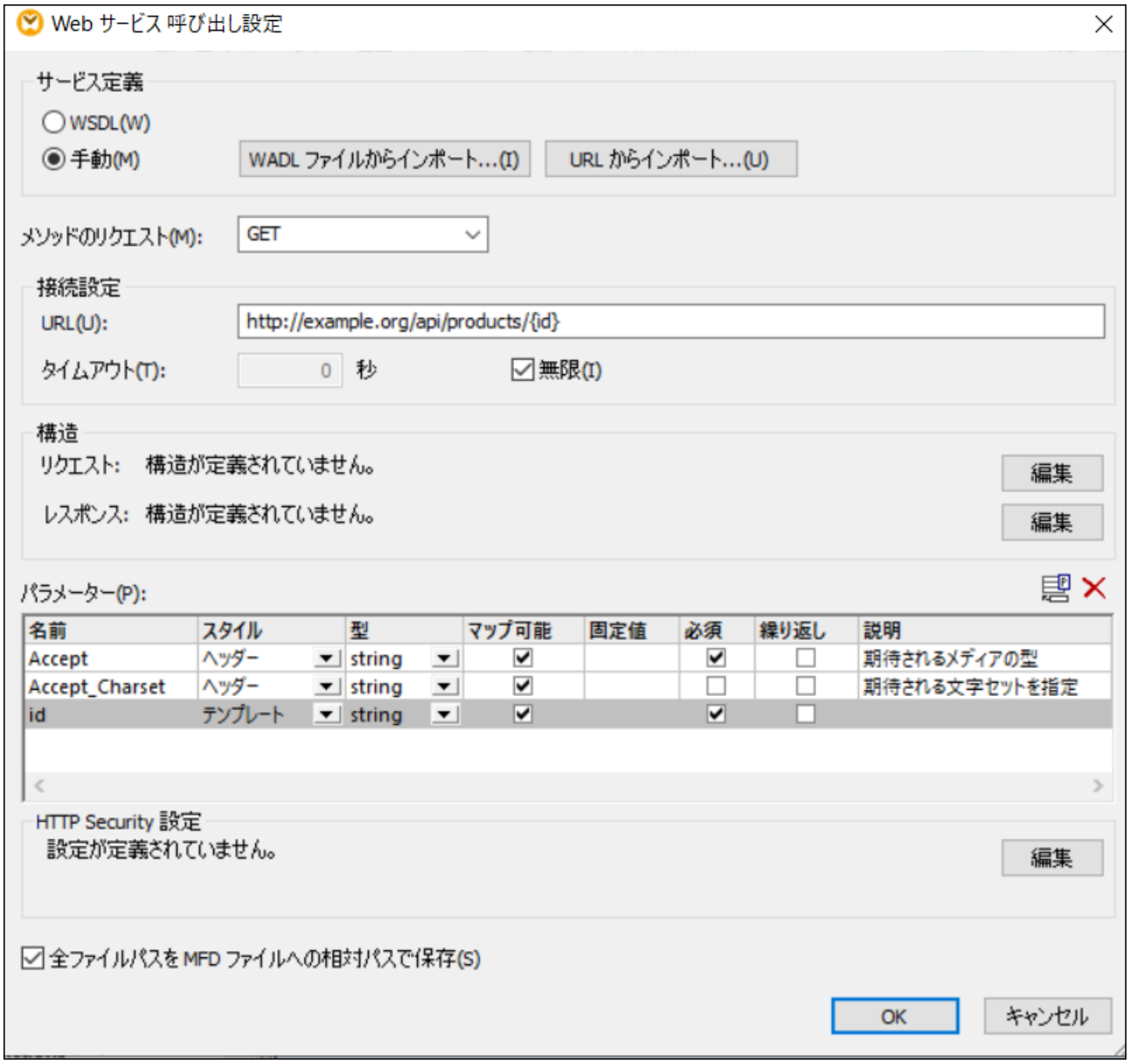
(C) 2015-2021 Altova GmbH

9.3 Web サービス呼び出し設定

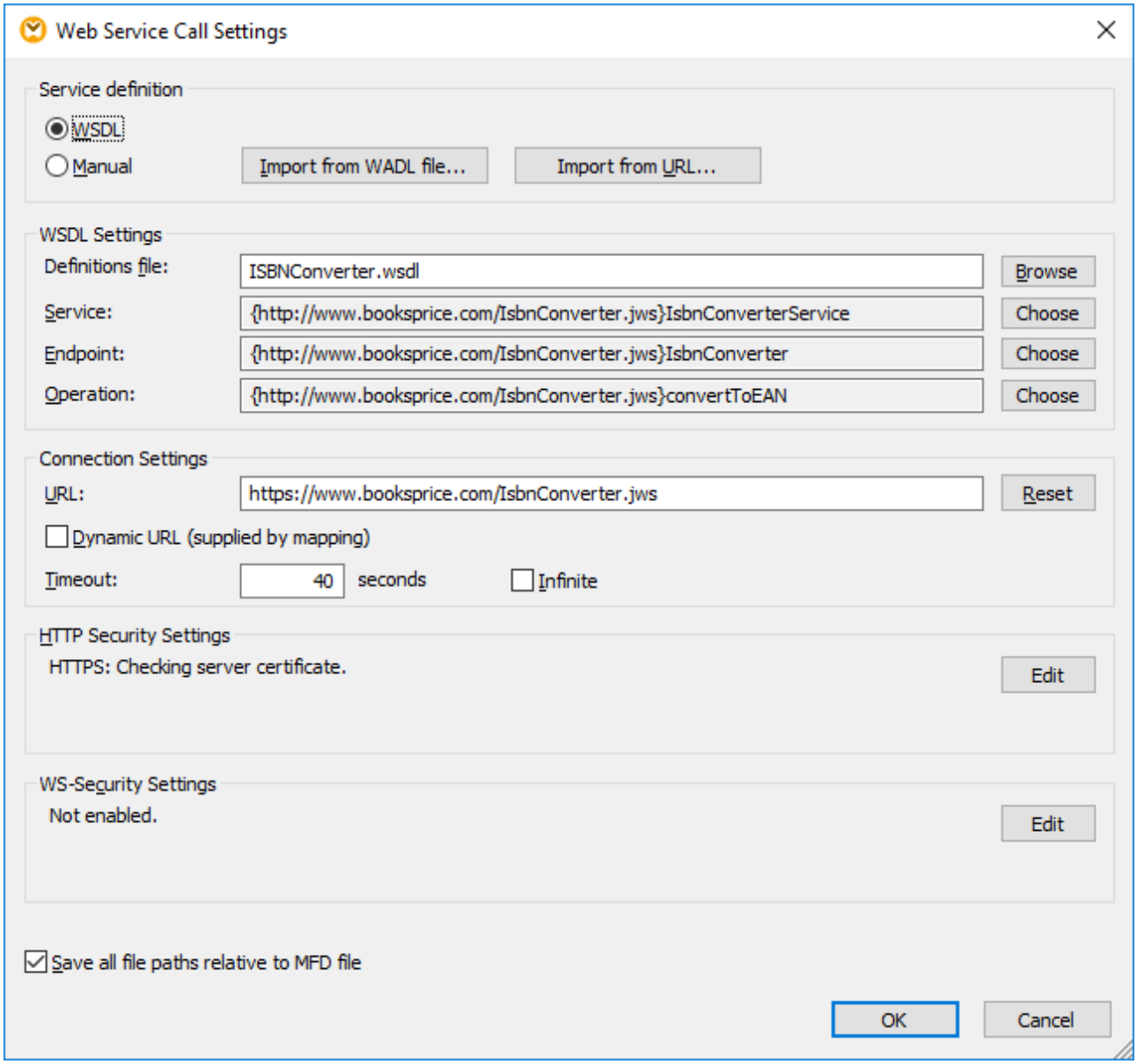
Web サービス呼び出し設定 ダイアログボックスから Web サービスに適用することのできる設定を変更することができます。このダイアログボックスを開くには、以下を行います:

- マッピングの Web サービス コンポーネントをクリックし、 **コンポーネント** メニューから「**プロパティ**」をクリックします。
- コンポーネント ヘッダーをダブルクリックします。
- コンポーネント ヘッダーを右クリックして、「**プロパティ**」をクリックします。

ダイアログボックスの上の **サービス定義** オプションが **WSDL** または **手動** に設定されているかにより使用することのできる設定のリストは異なります。次のイメージはジェネリックな HTTP Web サービスに適用することのできる設定を表しています:



下のイメージはWSDL-スタイルWeb サービスに適用することのできる設定を示しています:



使用することのできる設定は以下で説明されるとおりです。

サービスの定義

定義する設定がWSDL-スタイルWeb サービスに適用される場合、**WSDL** を選択し、それ以外の場合手動を選択します。

手動が選択されている場合、Web サービスの設定をダイアログボックスに手動で入力する必要があります。オプションで、Web サービスのWADL ファイルが存在する場合、Web サービスの設定をWADL **からインポート** をクリックすることでインポートできます(WADL から Web サービス情報をインポートするを参照) 。また、URL からWeb サービスパラメーターを抽出する場合は、URL **からインポート** ボタンをクリックします(URL ファイルからWeb サービスパラメーターをインポートするを参照)。

WSDL 設定

WSDL-スタイルWeb サービスのみに適用することのできる設定のグループ

WSDL 定義	呼び出されるWeb サービスのWeb サービス定義言語 (WSDL) ファイルを指定します。
サービス	呼び出されるWeb サービスの名前を指定します。WSDL ファイルが複数のWeb サービス複数のWeb サービスを定義する場合、「**選択**」をクリックして、必要なWeb サービスを選択します。
エンドポイント	呼び出されるWeb サービスのエンドポイント (またはポート) を指定します。選択されたWeb サービスの名前が複数のWeb サービスエンドポイントを定義する場合、「**選択**」をクリックして、必要な項目を選択してください。
オペレーション	呼び出されるWeb サービスのオペレーションを指定します。選択されたエンドポイントが複数のWeb サービスオペレーションを定義する場合、「**選択**」をクリックして、必要な項目を選択してください。

リクエストメソッド

この設定はジェネリックな (WSDL ではない) Web サービスのみに適用することのできる設定です。これはサービス (例えば、GET, POST, PUT など) により使用されるHTTP メソッド (動詞) を定義します。

接続設定

URL テキストボックスはWeb サービスのアドレス (URL) を指定します。**タイムアウト** は、サーバーからのレスポンスが内場合、Web サービスの呼び出しがタイムアウトされる間隔を定義します。呼び出しがレスポンスを無制限の時間待つ場合は、**無限** を選択してください。

Web サービスのURL をマッピングから提供する場合、(マッピングにより提供される) **動的な URL** チェックボックスを選択します。これにより、マッピング上のWeb サービスコンポーネント上にWeb サービスのURL を適用する追加入力コネクタが作成されます。

メモ (**マッピングにより提供されている**) **動的な URL** チェックボックスが選択されている場合、**URL** テキストボックスが無効化 (灰色表示) され、メインマッピングからURL が提供される必要があります。For REST-スタイルWeb サービスのためには、これは [Parameters] テーブル内でヘッダースタイルパラメーターのみが定義可能なことを意味します。

パラメーター

ジェネリックな (WSDL ではない) Web サービスのみに対してこのグループの設定を適用することができます。**パラメーター** テーブルは、MapForce がWeb サービスを呼び出すために使用するURL パラメーターを指定します (Web サービスパラメーターの定義を参照)。

構造

ジェネリックな (WSDL ではない) Web サービスのみに対してこのグループの設定を適用することができます。詳細に関してはリクエスト構造の定義 とレスポンス構造の定義を参照してください。

HTTP セキュリティ設定

Web サービスが (HTTP またはHTTPS) 認証 を必要とする場合、必要な認証設定を指定するために「**編集**」をクリックします (HTTP セキュリティの設定を参照してください)。

WS-Security 設定

WSDL-スタイルWeb サービスのために、これらの設定を適用することができます。Web サービスがカスタムSOAP セキュリティを使用する場合、「編集」をクリックして設定の詳細を指定してください（WS-Security の設定を参照してください）。

MFD ファイルに相対的なすべてのファイルパスを保存する

このオプションが有効化されていると、MapForce は、コンポーネント設定ダイアログボックスに表示されているMapForce Design (.mfd) ファイルの場所に対して相対的なファイルパスを保存します。相対と絶対パスの使用も参照してください。

(C) 2015-2021 Altova GmbH

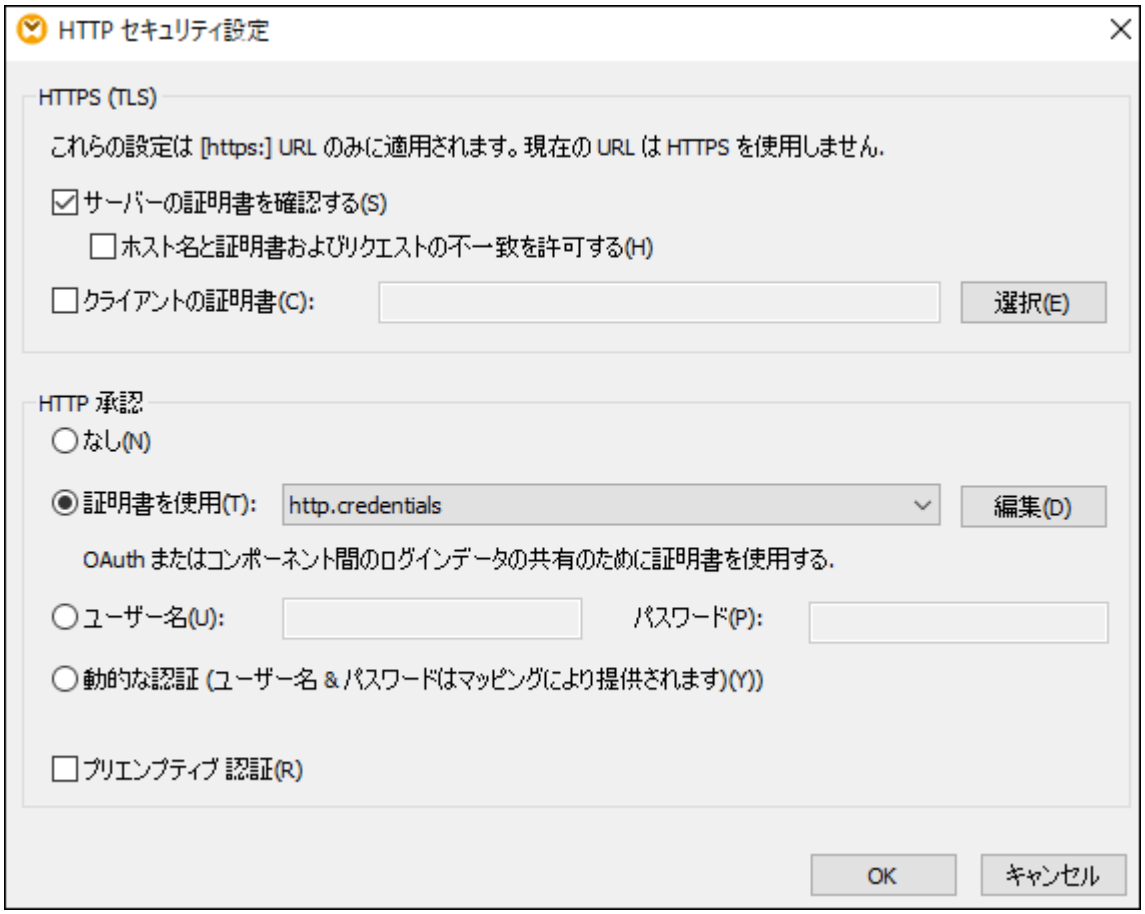
9.4 HTTP セキュリティの設定

HTTP セキュリティに関連する多種のシナリオを考慮したWeb サービスコンポーネントを構成することができます。例:

- Web サービスがHTTPS を介して呼び出され、クライアントの証明書が必要な場合。
- サービスが正確ではない証明書を使用し、証明書とリクエスト間の名前の不一致を許可する場合。
- Web サービスが基本的なHTTP 認証またはOAuth 2.0 認証を必要とする場合。

HTTP セキュリティの設定:

1. 「Web サービス呼び出し設定」ダイアログボックスを開きます。(Web サービス呼び出し設定を参照してください)。
2. HTTP セキュリティ設定の横の「編集」ボタンをクリックします。



HTTP セキュリティ設定 ダイアログボックス

オプションの"HTTPS (TLS)" グループはHTTPS を介して呼び出されるWeb サービスに適用されます。

サーバー証明書のチェック	デフォルトでこのチェックボックスが選択されており、リクエストを処理する前に、MapForce は、サーバーの資格情報をチェックするように構成されています。このオプションが有効化されると、 Web サービスリクエスト (とマッピング) サーバーが信

	頼できない場合、または使用中のオペレーティングシステムがW eb サーバーを信頼するように構成されていない場合は失敗します。 特別な理由がない限り、このオプションをスイッチオフすることは奨励されません。 以下も参照: デジタル署名の管理。
証明書とリクエストのホスト名の不一致を許可する	特定のホスト名のために発行されたサーバー証明書（例: *www.example.com*）が異なったホスト名（例: *example.com*）インストールされている場合に使用することができます。 証明書のホスト名がW eb サービスのホスト名に一致しない場合でも、チェックボックスを選択して、認証 を続行してください。
クライアント証明書	**現在のユーザー¥パーソナル** 証明書 ストアからクライアント証明書を選択し、ストアをクリックします。これは、クライアント証明書が **現在のユーザー¥パーソナル** 証明書ストアに既に存在していると仮定しています。それ以外の場合、 証明書のスナップインを使用してインポートしてください（ W indows 証明書ストアにアクセスするを参照してください）。 マッピングが他のオペレーティングシステムへの実行のためにデプロイされる場合は、同じ証明書がターゲットオペレーティングシステムにもインストールされている必要があります。更に詳しい情報に関しては、 デジタル署名の管理 を参照してください。

W eb サービスがHTTP 認証を必要とする場合、オプションの "HTTP 認証" グループが適用されます。これらのオプションの詳細に関しては、認証を使用してマッピングを実行する方法を参照してください。

無し	W eb サービスが認証を必要としない場合このオプションを選択してください。
資格情報の使用	HTTP またはOAuth 2.0 資格情報を構成するためにこのオプションを選択してください。詳細に関しては 資格情報 を参照してください。
ユーザー名、パスワード	マッピングファイル内でユーザー名とパスワードを暗号化された書式で保管するにはこのオプションを選択してください。このオプションは下位互換性のためにのみサポートされています。 保存されたユーザー名とパスワードを参照してください。
動的な認証	このダイアログボックスに入力する代わりにマッピングからユーザー名とパスワードを提供する場合、このチェックボックスを選択します。 動的な認証 を参照してください。
プリエンプティブ認証	W eb サービスが、承認データを最初の呼び出しで期待する場合、このチェックボックスを選択します。 それ以外の場合、MapForce は、ユーザー名とパスワード無しにアクセスを試み、サーバーが認証を必要とすると、ユーザー名とパスワードを使用します（HTTP ステータス401）。

9.5 WS-Security の設定

WS-Security 設定は、Web サービスが WS-Security により保護されていて、ユーザー名 トークン セキュリティトークンの情報を必要とする場合構成される必要があります。

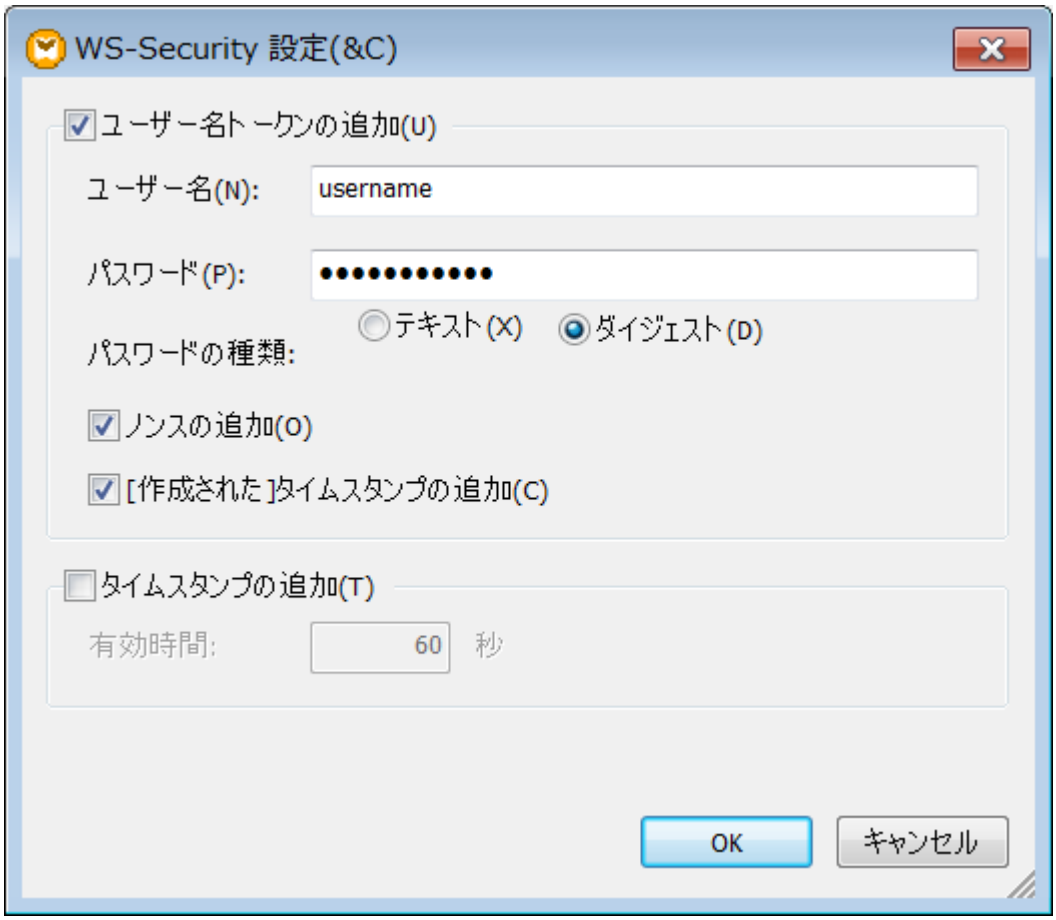
規約

Web サービスへ適用することのできる名前空間の省略名称がこのトピックで使用されています:

プレフィックス	名前空間
wsse	http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd
wsu	http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd

WS-Security の設定方法:

1. 「Web サービス呼び出し設定」ダイアログボックスを開きます。(Web サービスの呼び出しの設定を参照してください)。
2. WS-Security 設定の横の「編集」ボタンをクリックします。



WS-Security 設定ダイアログボックス

ユーザー名トークンの追加	ユーザー名 トークンは、SOAP メッセージのヘッダーに存在する WS-security の任意の要素です。ユーザー名 トークンは、Web サービスの発信者を認証するために Web サービスにより使用されます。 **wsse** 仕様内で(規約を参照) このフィールドは以下に対応します: `/wsse:UsernameToken`
ユーザー名	ユーザー名 トークン内に含まれるユーザー名を入力します。 **wsse** 仕様内で(規約を参照) このフィールドは以下に対応します: `/wsse:UsernameToken/wsse:Username`
パスワード	ユーザー名 トークン内に含まれるパスワードのテキストを入力します。 **wsse** 仕様内で(規約を参照) このフィールドは以下に対応します: `/wsse:UsernameToken/wsse:Password`
パスワードの種類	ユーザー名 トークン内に含まれるパスワードの種類を選択します。Web サーバーがこのモード内でパスワードを期待する場合は、**ダイジェスト** を選択します。それ以外

	の場合は、**テキスト** を選択します。 **wsse** 仕様内で（規約を参照） このフィールドは以下に対応します: `/wsse:UsernameToken/wsse:Password/@Type`
動的な認証	このダイアログボックスに入力する代わりにマッピングからユーザー名とパスワードを提供する場合、このチェックボックスを選択します。
ノンスの追加	ユーザー名トークンにノンスを追加する場合、このチェックボックスをチェックします。ノンスは、追加セキュリティを追加するために各ユーザー名トークンを一意に識別するランダムな値です。このオプションを有効化する場合、**"作成された" タイムスタンプを追加** オプションも有効化することが奨励されます。 **wsse** 仕様内で（規約を参照） このフィールドは以下に対応します: `/wsse:UsernameToken/wsse:Nonce`
"作成された" タイムスタンプの追加	各ナンスのためにタイムスタンプを追加するためにこのオプションを選択します。 **wsse** 仕様内で（規約を参照） このフィールドは以下に対応します: `/wsse:UsernameToken/wsu:Created`
タイムスタンプの追加	SOAP メッセージのための 有効時間（TTL） 値を有効化するにはこのチェックボックスをチェックします（次のオプション参照）。 **wsse** 仕様内で（規約を参照） このフィールドは以下に対応します: `/wsu:Timestamp`
有効時間	メッセージの取得または返信の可能性を削減するために、SOAP メッセージのための 有効時間（TTL） を入力します。

9.6 例: REST-スタイル Web サービスの呼び出し

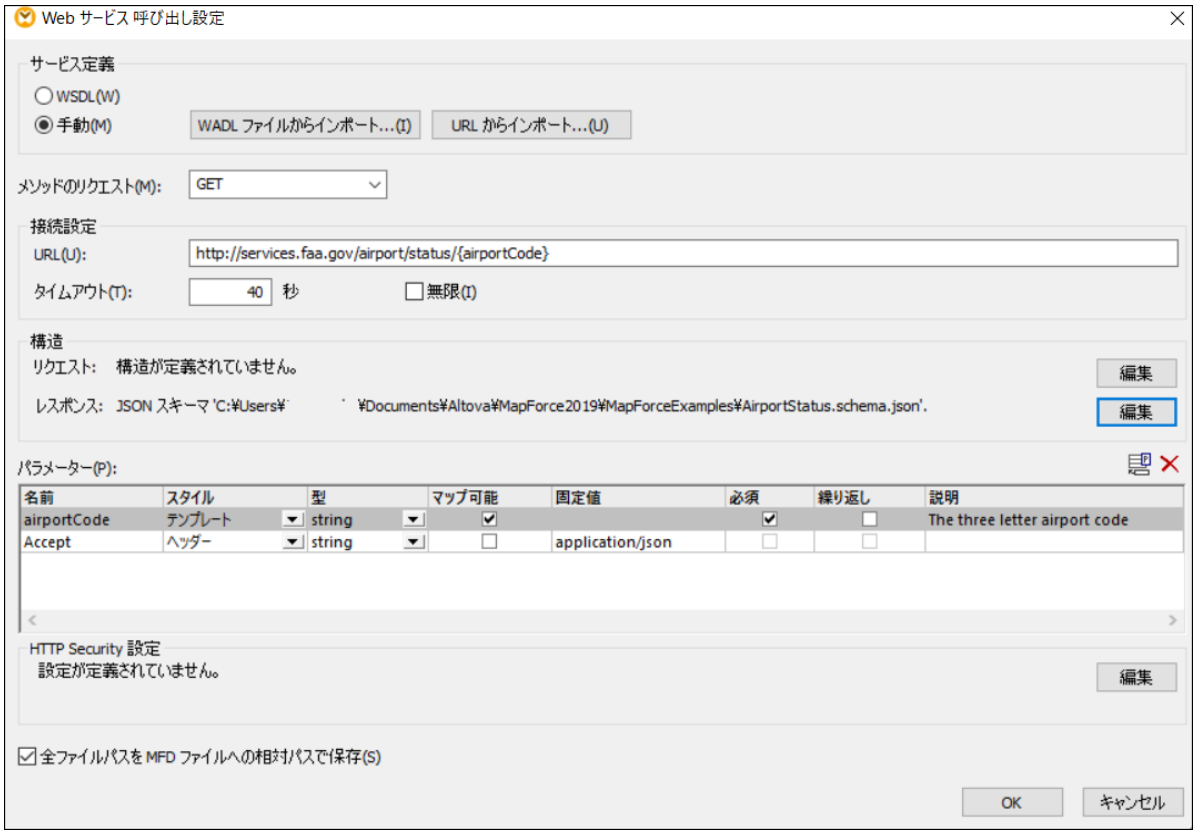
この例は、MapForce から、ジェネリックな HTTP W eb サービスを呼び出す方法について説明しています。この例で呼び出される W eb サービスは以下で見つけることができます: https://app.swaggerhub.com/apis/FAA/ASWS/1.1.0。この W eb サービスは、(例えば、"SFO"、"IAD"、"ABE"、"DFW" などの) 3つの英数文字からなる空港名コードを引数として受け入れ、米国の主要な空港の現状を XML または JSON 構造として返します。この例は、次のパスで見つけることのできるマッピングデザインファイルが伴われます: **<マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥AirportStatus.mfd** .

この例では、W eb サービスのレスポンスは、JSON フォーマットで、JSON ファイルにマップされます。ですから、W eb サービスのレスポンス構造の JSON スキーマが必要になります。JSON スキーマは W eb サービスプロバイダーにより提供されたサンプルレスポンスより XMLSpy を使用して生成されています。このスキーマをマイドキュメント フォルダーに相対的な次のパスで見つけることができます: **<マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥AirportStatus.schema.json**.

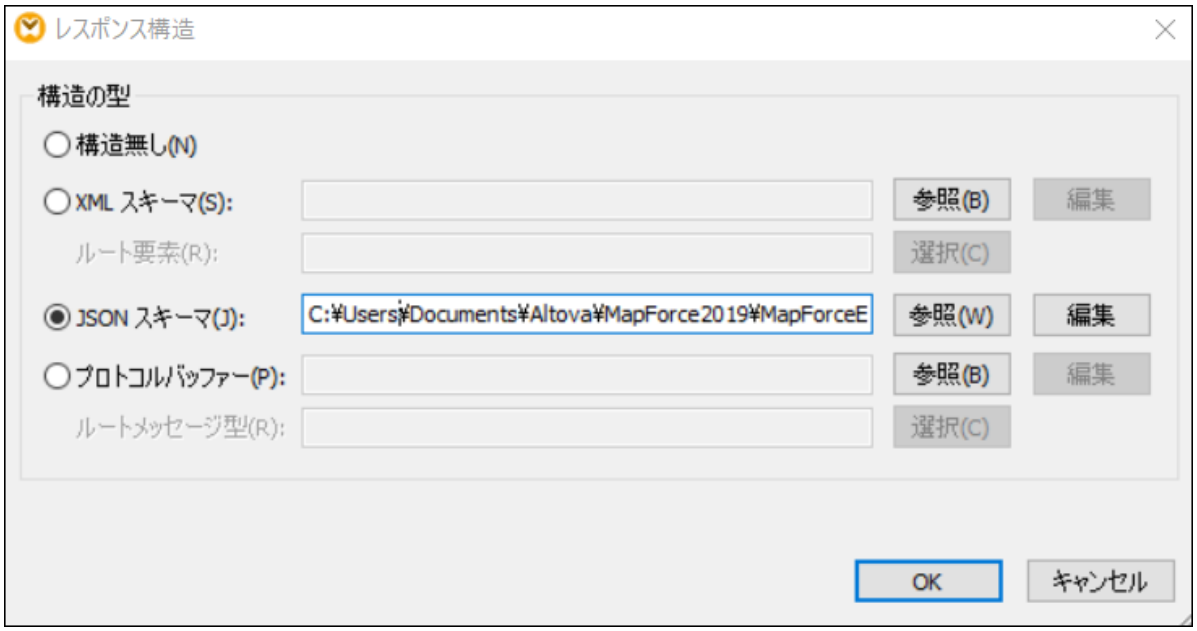
リクエスト パートに関しては、リクエストに応じて "SFO" の値を提供する W eb サービスを呼び出します。レスポンス パートに関しては、レスポンスデータを JSON ファイルにマップします。次の3つの主要なコンポーネントから構成されます: W eb サービス呼び出し、入力パラメーター、JSON 出力ファイル。

ステップ 1: W eb サービスコンポーネントを追加する

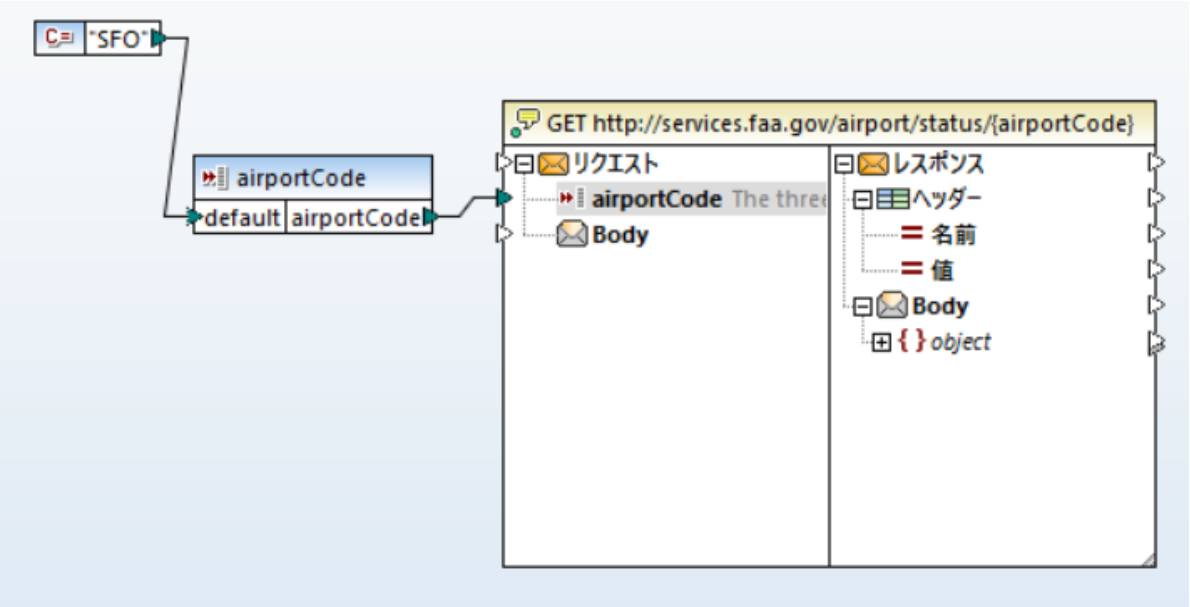
1. マッピングの変換言語が BUILT-IN に設定されていることを確認してください (変換言語の選択 を参照する)。
2. **「挿入」** メニューから **「W eb サービス関数」** をクリックします。
3. **「サービス定義」** から **「手動」** をクリックします。
4. リクエスト メソッドを GET に設定し、URL を `http://services.faa.gov/airport/status/{airportCode}` に設定します。中かっこ内の値は、ランタイム時に実際の空港名コードを置き換えられるテンプレート パラメーターです (W eb サービス呼び出しの追加 (REST-スタイル) を参照)。
5. **「パラメーターの追加」** ( ) ボタンをクリックして、新規のパラメーターを「パラメーター」テーブルに追加します。パラメーターの名前は、パラメーターの名前は、URL 内の中かっこ内で指定されているパラメーターと同じである必要があることにご注意してください。スタイルを "テンプレート" に設定し、"文字列" を選択し、"マップ可能" と "必須" オプションをチェックします。"説明" フィールドは任意です。
6. W eb サーバーにクライアント (つまり、MapForce) がレスポンスに JSON を期待することを伝えるために、ヘッダーを追加します。これを行うには、**「パラメーターの追加」** ( ) ボタンを再度クリックし、パラメーターを "Accept" と名づけ、スタイルを「ヘッダー」とし、**application/json** を固定値として入力します。



7. 「レスポンス」から「編集」ボタンをクリックして、Web サービスレスポンスのためのスキーマを参照します。スキーマは以下のパスで検索することができます: **<マイドキュメント>\Altova\MapForce2021\MapForceExamples\AirportStatus.schema.json**.

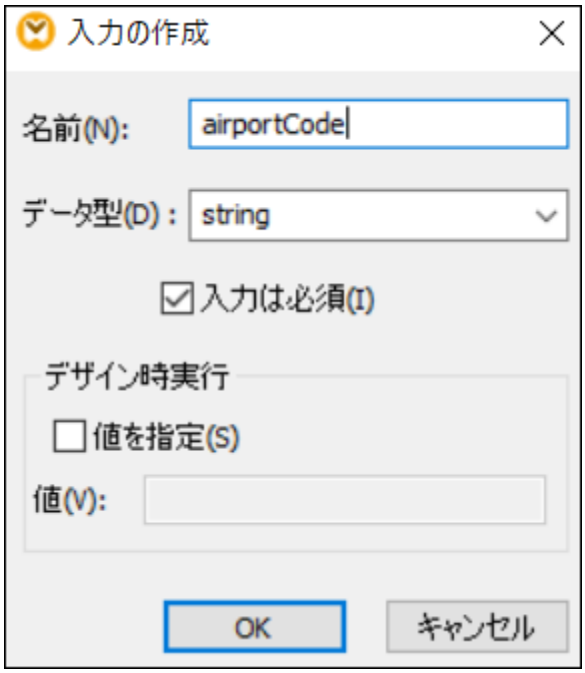


8. 「OK」をクリックします。Web サービスをマッピングエリアで使用することができます。



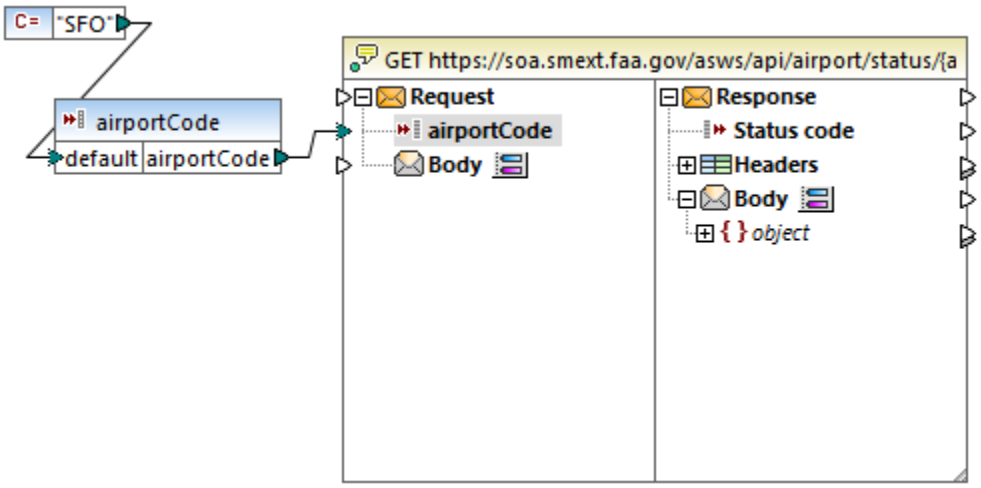
ステップ 2: 入力パラメーターの追加

1. 「**挿入**」メニューから「**入力の挿入**」をクリックします。
2. "airportCode" をパラメーター名として入力し、「**入力が必要です**」オプションをクリックして、チェックを解除します（上に示されるように、入力は定数から与えられます）。



3. （メニューコマンド「**挿入 | 定数**」を使用して）"SFO" の値を持つ定数を追加し、パラメーター入力と接続します。
4. aiportCode パラメーターの出力を上に表示されるようにリクエストの入力に接続します。

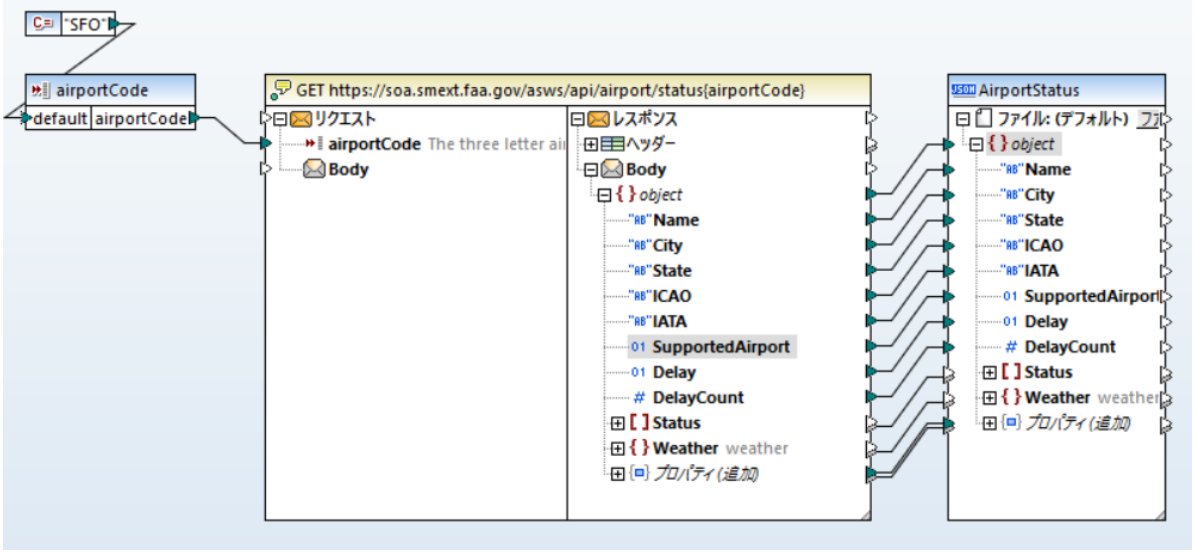
(C) 2015-2021 Altova GmbH



この段階では、Web サービスのリクエストパートは、完成しています。マッピングランタイムにおいて、Web サービスは定数により与えられた値（この場合、"SFO"）により呼び出されます。

ステップ 3: JSON 出力を追加する

1. 「挿入」メニューから「JSON スキーマ/ファイル」をクリックします。
2. <マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥**AirportStatus.schema.json** を参照して、「開く」をクリックします。
3. サンプル JSON ファイルを与えるようにプロンプトされると、「スキップ」をクリックします（サンプル JSON ファイルは生成されるため、必要ありません）。
4. 「接続」メニューで、「一致する子を自動的に接続」メニューアイテムが選択されていることを確認してください。このオプションを有効化することにより、次のステップで時間を節約することができます。このオプションを有効化することにより、次のステップで時間を節約することができます。
5. Web サービスのレスポンス本文の `object` ノードを、JSON コンポーネントの `object` ノードに下に示されるように接続します。「一致する子を自動的に接続」が前のステップで有効化されているため、全ての子孫ノードが自動的に接続されるため、各接続を描く必要がありません。



ステップ 4: マッピングの実行

W eb サービスを呼び出す準備が整うと、「出力」ボタンをクリックして、マッピングを実行し、一般的な出力をプレビューします。W eb サービス呼び出しに成功すると、「出力」タブは、返されたJSON 構造を表示します。呼び出しに失敗すると、MapForce は、エラーを状況に応じて返します。W eb サービスで説明されているとおり、エラーレスポンスとしてHTTP コード 500 がW eb サービスにより返される場合があります。情報が存在しない場合 HTTP コード 404 が返される場合があります。

MapForce Server が存在する場合、(https://www.altova.com/mapforce-server) マッピングをマッピング実行可能ファイル (*.mfx) にコンパイルし、コマンドライン、またはMapForce Server が実行されるサーバーマシン上のMapForce Server API から実行することができます。(マッピングを MapForce Server 実行可能ファイルへコンパイルするを参照)。

GET メソッドを使用してリアルタイムに飛行場の状態を取得するジェネリックな HTTP W eb サービスの呼び出しの作成が完了しました。W eb サービスとの作業に関する一般的な情報に関しては、次を参照してください W eb サービスの呼び出しの追加 (RE ST-スタイル)を参照してください。

(C) 2015-2021 Altova GmbH

9.7 例: RSS フィードからのデータのマッピング

この例は、RSS (Rich Site Summary) フィードからのデータを Microsoft Excel にマップするために、MapForce からジェネリックな HTTP Web サービスを呼び出す方法について説明しています。この例は、次のパスで見つけることのできるマッピングデザインファイルが伴われます: <マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥RssReader.mfd 。

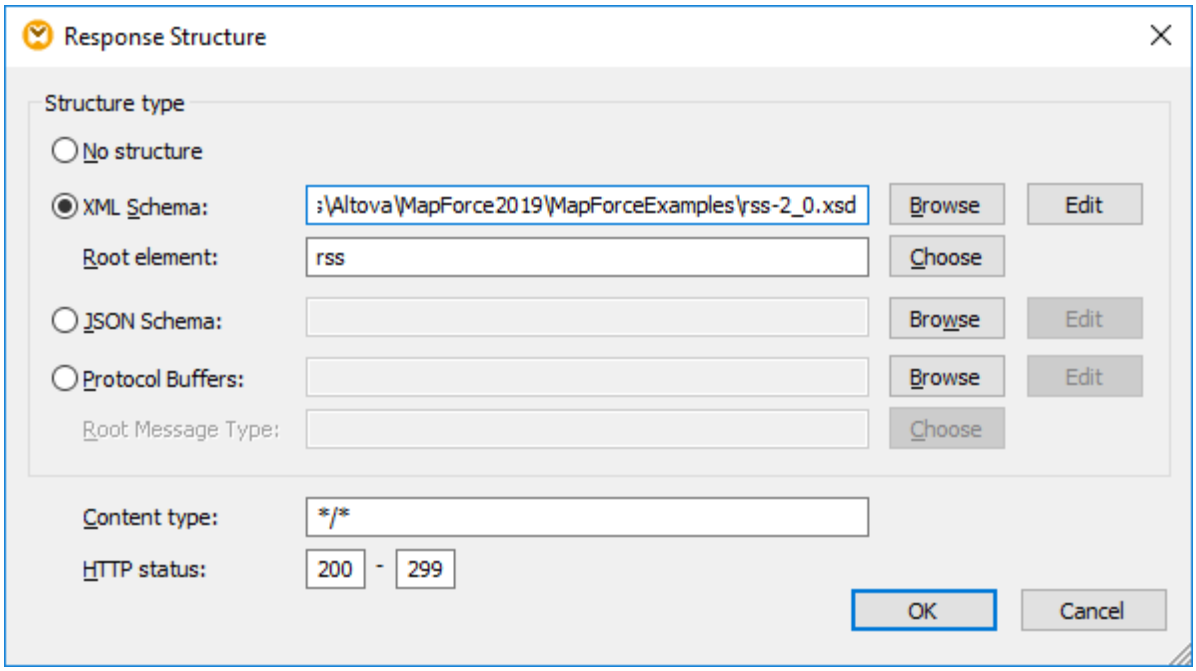
この例では、Web サービスのレスポンスが Altova ブログの RSS フィードからマップされます (http://blog.altova.com/feed)。Web サービスレスポンス構造のスキーマが必要になり、MapForce は、フィードより返されたデータの構造を作成することができます。必要なスキーマをマイドキュメント フォルダーに相対的な次のパスで見つけることができます: **<マイドキュメント >\Altova\MapForce2021\MapForceExamples\rss-2-0.xsd**。

RSS フィードの呼び出しは、リクエストパラメーターを必要としません。ですから、 Web サービスの呼び出しのリクエストパートは空です。レスポンスは、Microsoft Excel ファイルにマップされます。また、各 RSS エントリの公開日を読みやすくするために、Excel ファイル内で YYYY-MM-DD にフォーマットされます。

ですから、マッピングは、次の3つの主要なコンポーネントから構成されます: Web サービスの呼び出し、ターゲット Excel コンポーネント、データ処理関数。

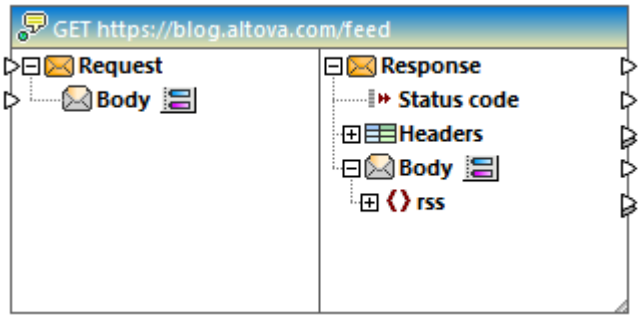
ステップ 1: Web サービスコンポーネントを追加する

1. マッピングの変換言語が BUILT-IN に設定されていることを確認してください (変換言語の選択 を参照する)。
2. 「挿入」メニューから、「Web サービス関数」をクリックします。
3. 「サービスの定義」から「手動」をクリックします。
4. リクエストメソッドを GET に設定し、URL を http://blog.altova.com/feed に設定します。
5. 「レスポンス」から「編集」ボタンをクリックして、Web サービスレスポンスのためのスキーマを参照します。スキーマは以下のパスで検索することができます: <マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥rss-2-0.xsd.



6. 「OK」をクリックします。

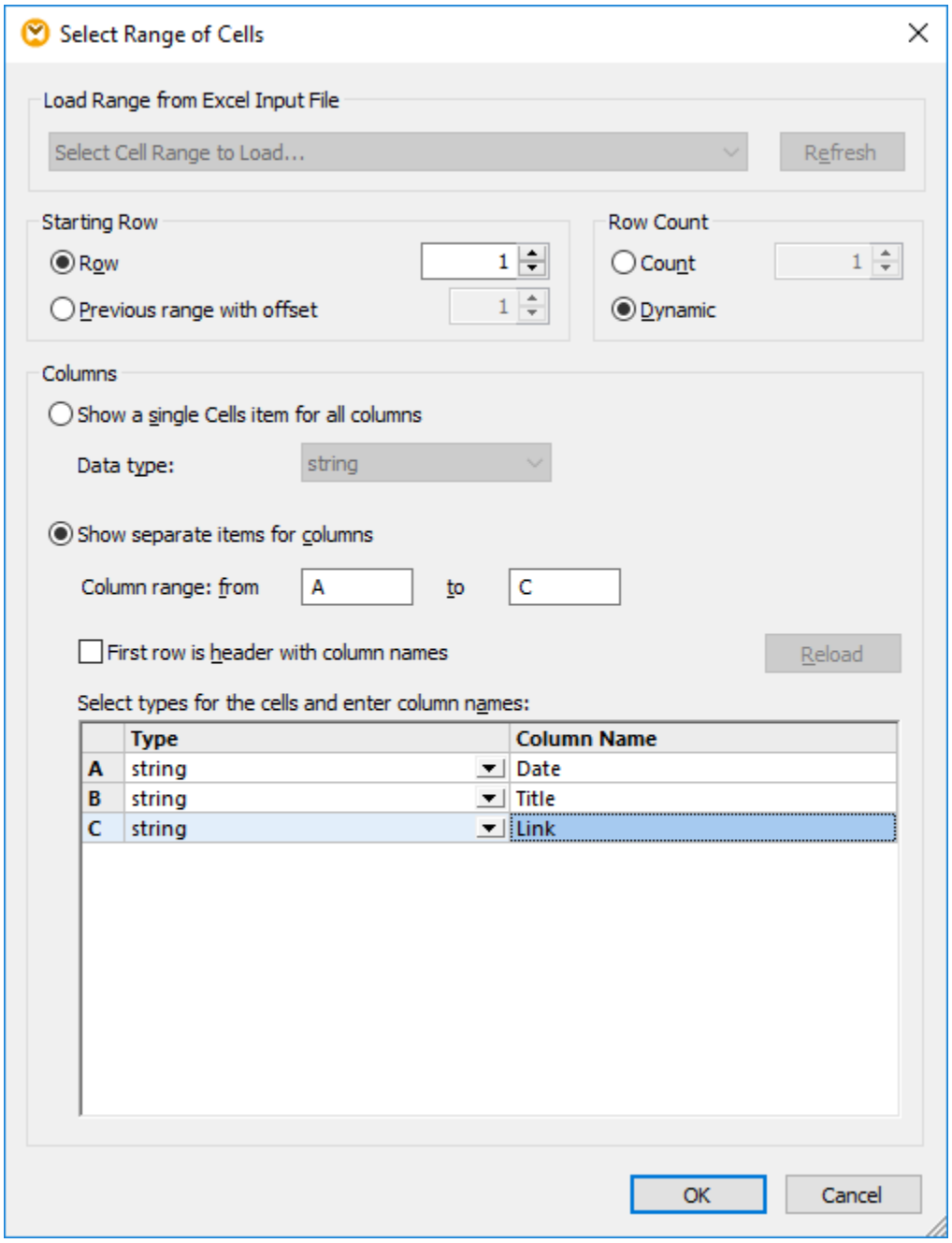
この段階では、Web サービスは、マッピングエリアで使用することができ、レスポンスパートからデータをマップすることができます。



ステップ 2: ターゲット Excel コンポーネントを追加する

1. 「挿入」メニューから、「Excel 2007+ ファイル」をクリックします。
2. サンプルファイルを与えるようにプロンプトされると、「スキップ」をクリックします。Excel コンポーネントをマッピングエリアで使用することができます（サンプル JSON ファイルは生成されるため、必要ありません）。
3. シート 1 の横のボタン をクリックします。「名前順にワークシートを表示」オプションのチェックを解除するためにクリックします。
4. 行 1 の横のボタン をクリックします。下に示されるようにオプションを変更します。

(C) 2015-2021 Altova GmbH



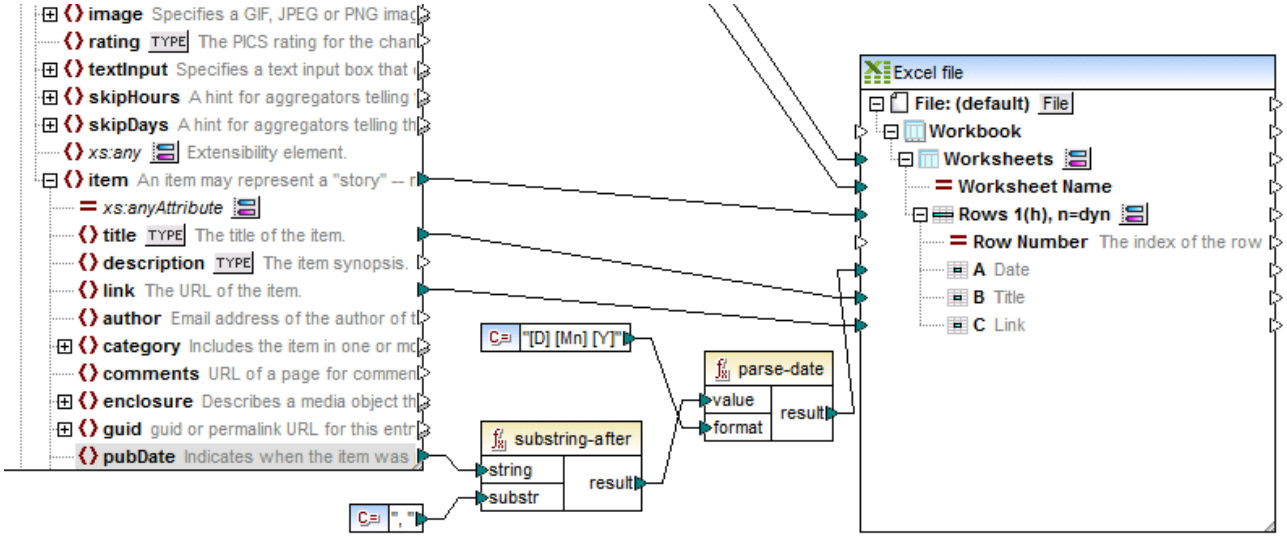
7. レスポンス本文の次のアイテムを Excel コンポーネントの次のアイテムに接続します:

Response body	Excel file
channel	Worksheets
title	Worksheet Name
item	Rows
item/title	Row B (Title)
item/link	Row C (Link)

この段階では、行 A (Date) の入力の値が不足しています。この値は次のステップで追加されます。

ステップ 3: 日付処理関数を追加する

RSS フィード内のレスポンス内の日付は、RFC-822 フォーマットで表示されます。例: `Wed, 20 Jan 2016 14:49:35 +0000`。目標は、しかしながら、`YYYY-MM-DD` として、Excel ファイルファイル内で日付をフォーマットすることです。これを達成するには、MapForce core ライブラリの **substring-after** と **parse-date** 関数マッピングに追加します (関数をマッピングに追加する方法に関しては、次を参照してください: マッピングに関数を追加する)。



上に表示されるとおり、**substring-after** 関数は、**pubDate** ノードから値を取り、`, ` の後のテキストを返します。結果の値はマスク `[D] [Mn] [Y]` を使用して解析し、値を `xs:date` として返す MapForce core 関数 **parse-date** にパスされます。フォーマットマスクとは、月初めから数えた日の数値、スペース、月名、4桁の年数の順番で表示することを意味します (詳細に関しては、次を参照してください: **format-dateTime** 関数)。

最後に、解析された値は、Excel コンポーネントの「日付」行に与えられます。

ステップ 4: マッピングの実行

Web サービスを呼び出す準備が整うと、「出力」ボタンをクリックして、マッピングを実行し、一般的な出力をプレビューします。Web サービス呼び出しに成功すると、「出力」タブは、返されたデータ構造を表示します。呼び出しに失敗すると、MapForce は、エラーを状況に応じて返します。

(C) 2015-2021 Altova GmbH

MapForce Server が存在する場合、(http://www.altova.com/ja/mapforce/mapforce-server.html)マッピングをマッピング実行可能ファイル(*.mfx) にコンパイルし、コマンドライン、またはMapForce Server が実行されるサーバーマシン上のMapForce Server API から実行することができます。(次を参照: マッピングを MapForce Server 実行可能ファイルへコンパイルする)。

GET メソッドを使用してリアルタイムにRSS フィード取得するジェネリックな HTTP W eb サービスの呼び出しの作成が完了しました。W eb サービスとの作業に関する一般的な情報に関しては、次を参照してください: W eb サービス呼び出しの追加 (REST-スタイル).

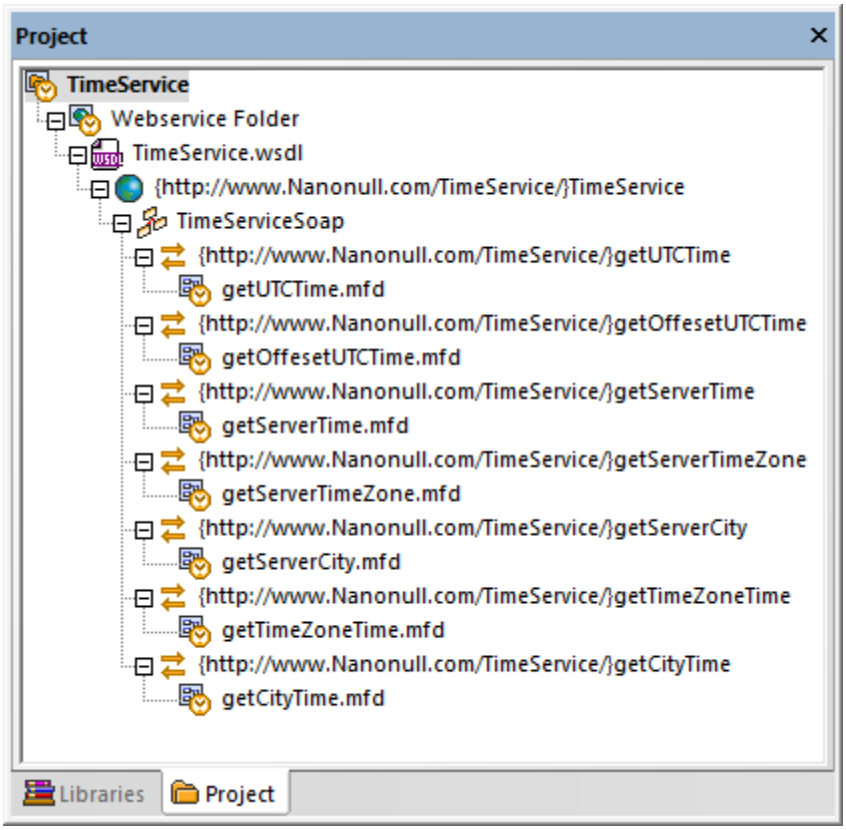
9.8 例: SOAP Web サービスの呼び出し

以下のサンプルでは、MapForce により実装された W eb サービスを呼び出します。このサンプルはデモを行うために作成されたもので、互換性のあるプロトコルを使用したその他テクノロジーにより実装された W eb サービスを呼び出すことも可能です。

<マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥TimeService フォルダーに含まれている **TimeService.mfp** マッピングプロジェクトの一部を以下に示します。**<マイドキュメント>\Altova\MapForce2021\MapForceExamples\TimeserviceWsdl2** フォルダーに含まれている、**TimeService2.mfp** プロジェクトファイルでは W SDL 2.0 もサポートされます。

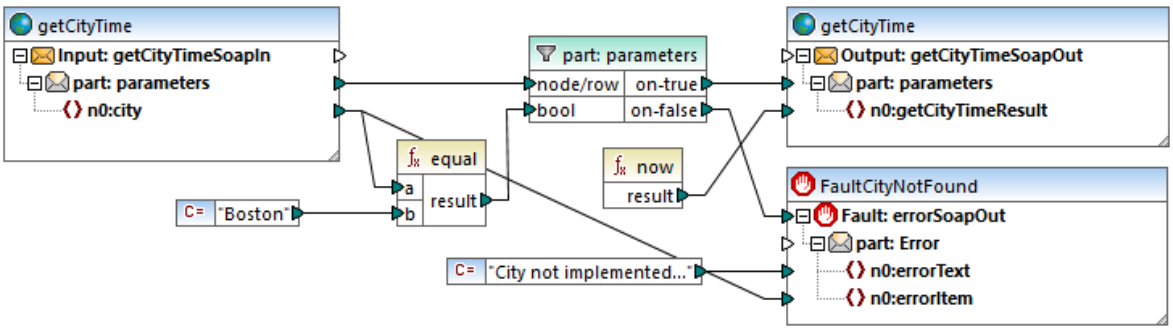
W eb サービスがどのように実装されているかをビューする:

1. **<マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥Timeservice** フォルダーに含まれている **TimeService.mfp** ファイルを開きます。プロジェクトに関連したファイルはプロジェクトウィンドウ内にロードされます。



2. プロジェクトウィンドウ内の **getCityTime.mfd** エントリをダブルクリックします。

(C) 2015-2021 Altova GmbH

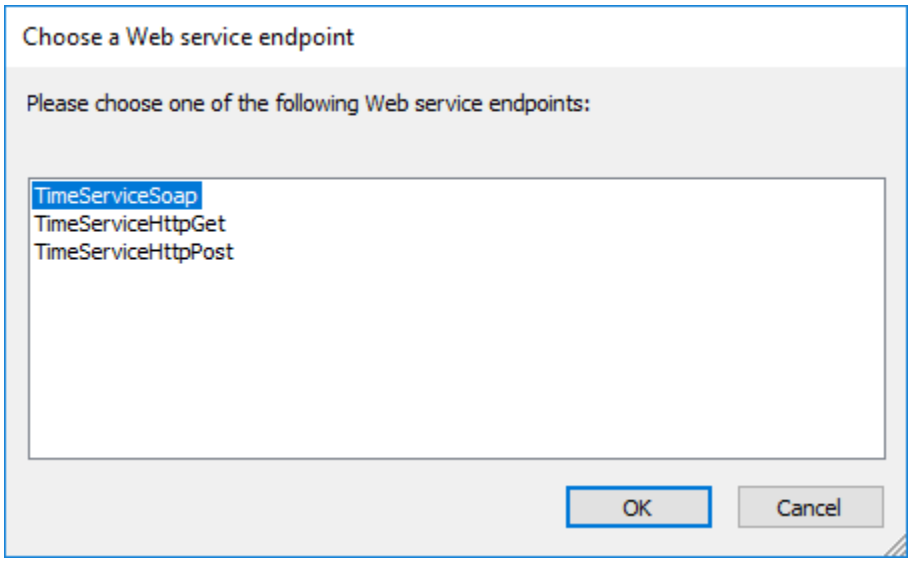


getCityTime.mfd マッピングは、都市の名前を入力として受け、現在の時刻を出力として返します。都市が"Boston" ではない場合、W SDL フォールトが返されます。<**マイドキュメント**>¥Altova¥MapForce2021¥MapForceExamples¥**TimeService** ディレクトリ内にある **getCityTimeRequest.xml** ファイルから入力データをマッピングは取得します。**getCityTimeSoapIn** コンポーネントをダブルクリックすると、このファイルがデータソースとして設定されていることを確認できます。

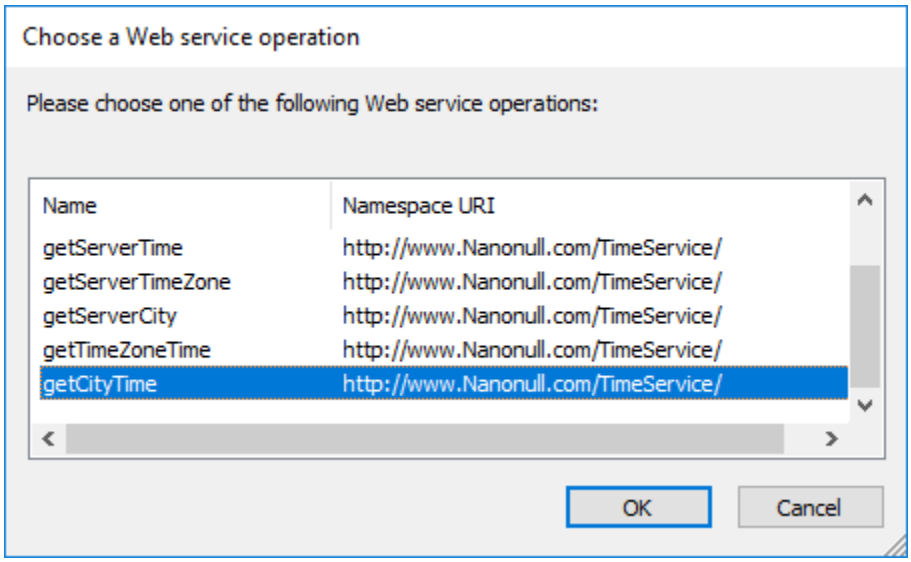
この例のスコープのために、この特定の W eb サービス プロジェクトは既にコンパイルされ、このサンプルで次のステップとして呼び出される実在の W eb サーバーにデプロイされていると仮定します。http://nanonull.com/TimeService/TimeService.asmx でこの目的のためのデモオンラインサービスを使用することができます。MapForce を使用してこのような W eb サービスを作成するには、例: SOAP W eb サービスの生成 (C#) と例: SOAP W eb サービスの生成 (Java) を参照してください。

W eb サービスをマッピングに追加する方法:

1. 「**ファイル | 新規作成**」 を選択して、「**マッピング**」 アイコン をクリックし、「OK」をクリックします。
2. メニューオプション「**挿入 | W eb サービス関数...**」を選択、または ツールバーをクリックします。
3. W SDL 定義ファイルを選択するために「**参照**」 をクリックします。TimeService ディレクトリから **TimeService.wsdl** を選択し、「**開く**」ボタンクリックします。
4. W eb サービスエンドポイントを選択するように問われると、**TimeServiceSoap** をクリックします。

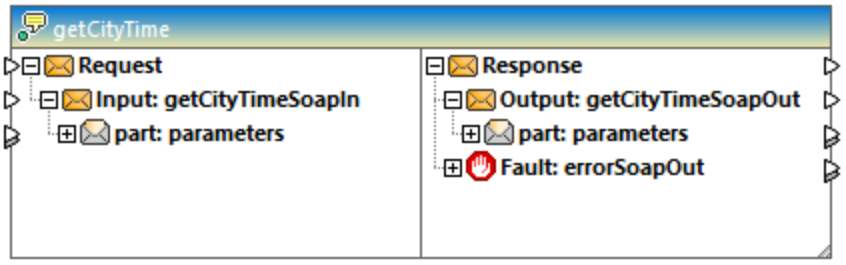


5. W eb サービス操作を選択するように問われると、**getCityTime** をクリックします。



6. Web サービス呼び出し設定の設定を変更せず、「OK」をクリックします。更に詳しい情報に関しては、Web サービス呼び出し設定を参照してください。

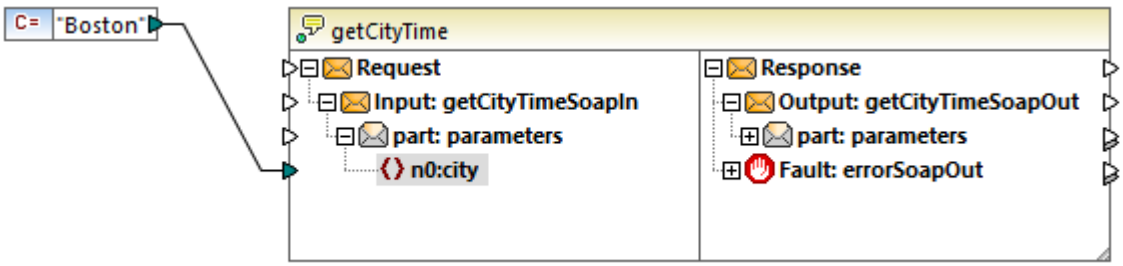
   getCityTime Web サービス関数は、単一コンポーネントとして挿入されます。WSDL プロジェクト内で保存されたとおりに **getCityTime.mfd** ファイルを構成する 8 つすべてのコンポーネントを示していることに注意してください。



   .wsdl ファイル内で、以下のうちの 1 つが定義されている場合、コンポーネントの左側でデータ入力 データ入力（SoapIn）が定義され、フォールトのセクションを含んでいる可能性がある右側でデータ出力（SoapOut）が定義されます。

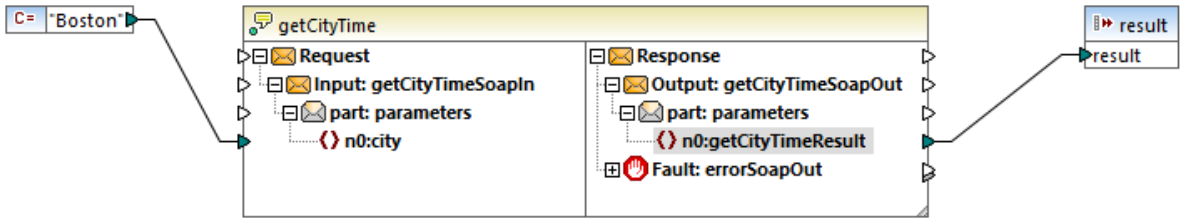
Web サービスを呼び出す方法:

1. たとえば、定数、テキスト、またはスキーマコンポーネント、入力データを提供するためのコンポーネントを挿入します。この例のスコープのために、定数コンポーネントを挿入して、"Boston" を入力文字として挿入します。

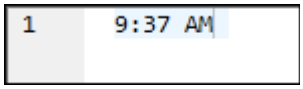


2. 定数を **n0:city** アイテムに接続します。

3. 単純型出力コンポーネントを挿入します（**関数** メニューから、「**出力の挿入**」をクリックします）。
4. **n0:getCityTimeResult** を出力コンポーネントに接続します。



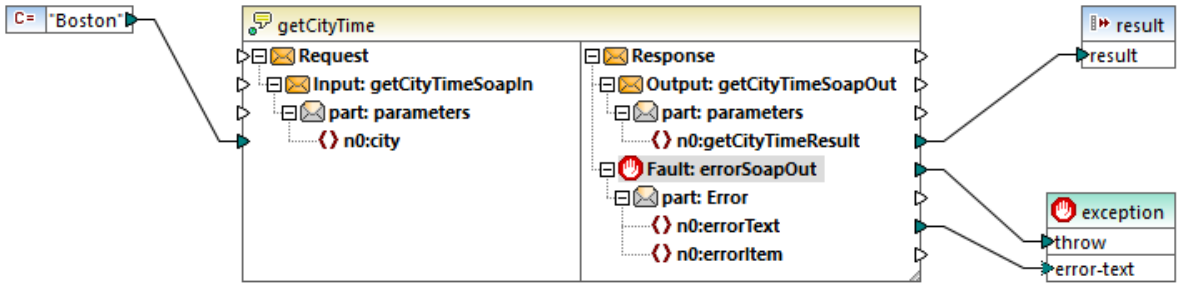
5. マッピング上の"getCityTime" コンポーネントを右クリックし、**プロパティ**を選択します。Web サービス呼び出し設定ダイアログが開かれます。`http://nanonull.com/TimeService/TimeService.asmx` を **URL** ボックスに入力します（これは MapForce により作成されるデモオンライン Web サービスです。SOAP Web サービスの実装を参照してください）。
6. **出力** タブをクリックします。下に示されるように、ボストンの現在の時間が**出力**ペインに表示されます。



メモ　Web サービス関数の入力値は、元のマッピングのデータソースを優先します。例えば、定数コンポーネントの値 "Boston" は、元のマッピング内の **getCityTimeRequest.xml** データソースファイルを優先します。

Web サービスのマップに失敗した場合:

1. **挿入 | 例外** を選択、または ツールバーボタン をクリックします。
2. **Fault:** アイテムを例外コンポーネントの **throw** アイテムにマップします。
3. n1:errorText アイテムを 例外コンポーネントの **error-text** アイテムにマップします。



9.9 例: OAuth 2.0 認証

このサンプルはOAuth 2.0 認証を必要とするREST-スタイルW eb サービスを呼び出す方法について説明しています。クライアントアプリケーションはGoogle カレンダーAPI (https://developers.google.com/calendar/)を使用してのイベントを抽出するMapForce マッピング です。簡素化するためにマッピング はカレンダーの情報をそのまま抽出し、他の処理無しにJSON 結果を生成します。

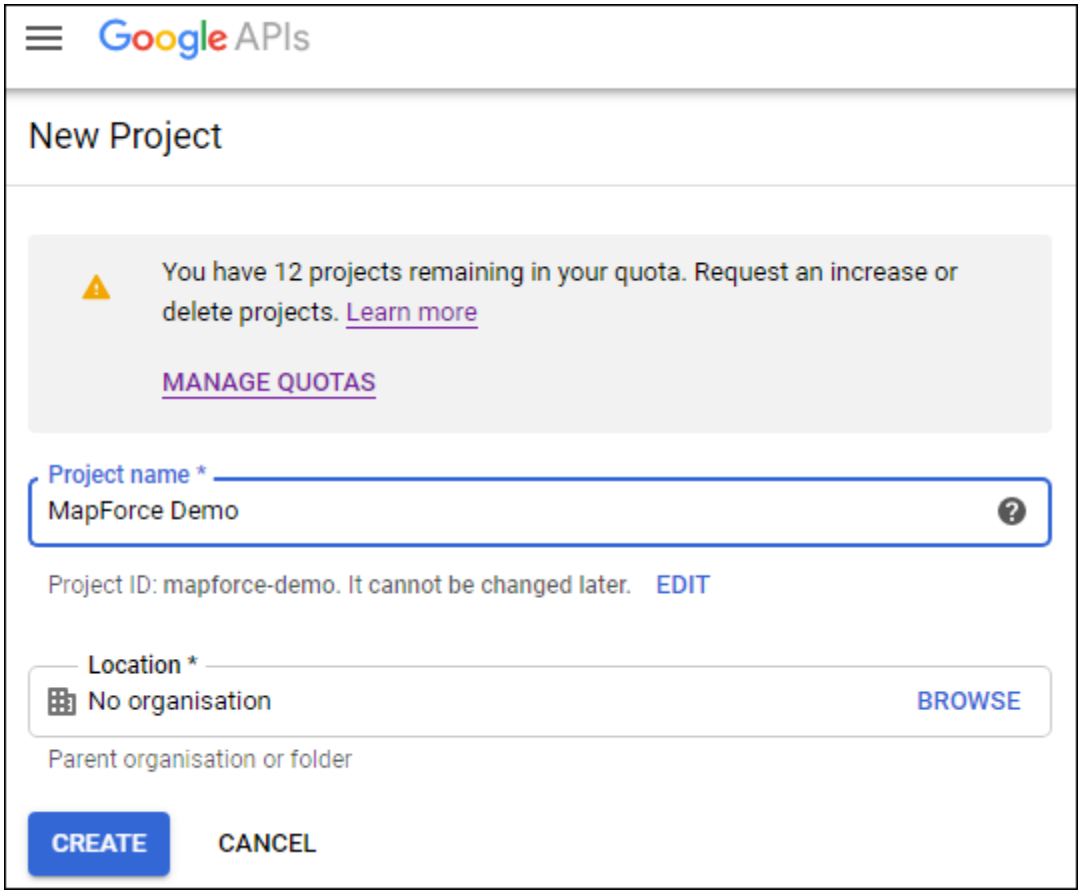
必要条件:

- MapForce Enterprise Edition
- このサンプルをステップバイステップでフォローするにはGoogle アカウント が必要です。他のW eb サービスを呼び出す必要がある場合 OAuth 2.0 資格情報 W eb サービスプロバイダーから取得し、以下のステップで使用してください。

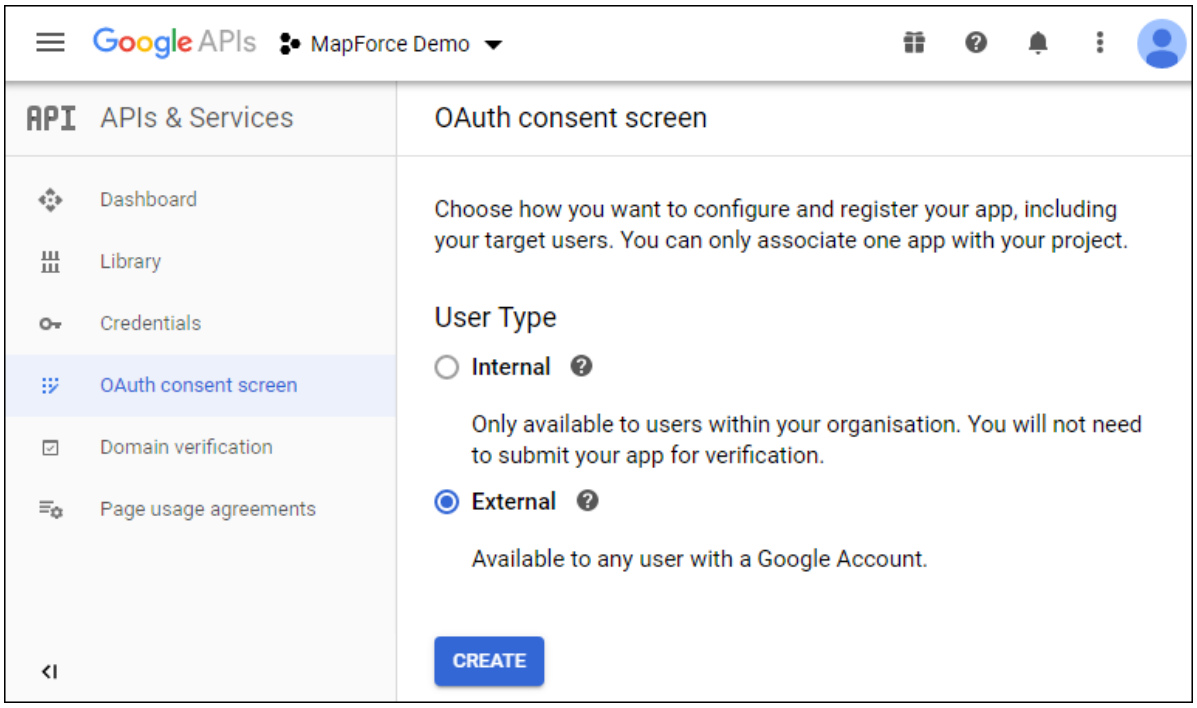
OAuth 2.0 認証情報を取得する

W eb サービスにアクセスするために必要な OAuth 2.0 認証情報が既存の場合このステップをスキップすることができます。それ以外の場合、取得方法はマッピングが呼び出すW eb サービスのプロバイダーにより異なります。このサンプルのような Google カレンダーAPI を呼び出す場合以下のステップに従ってください:

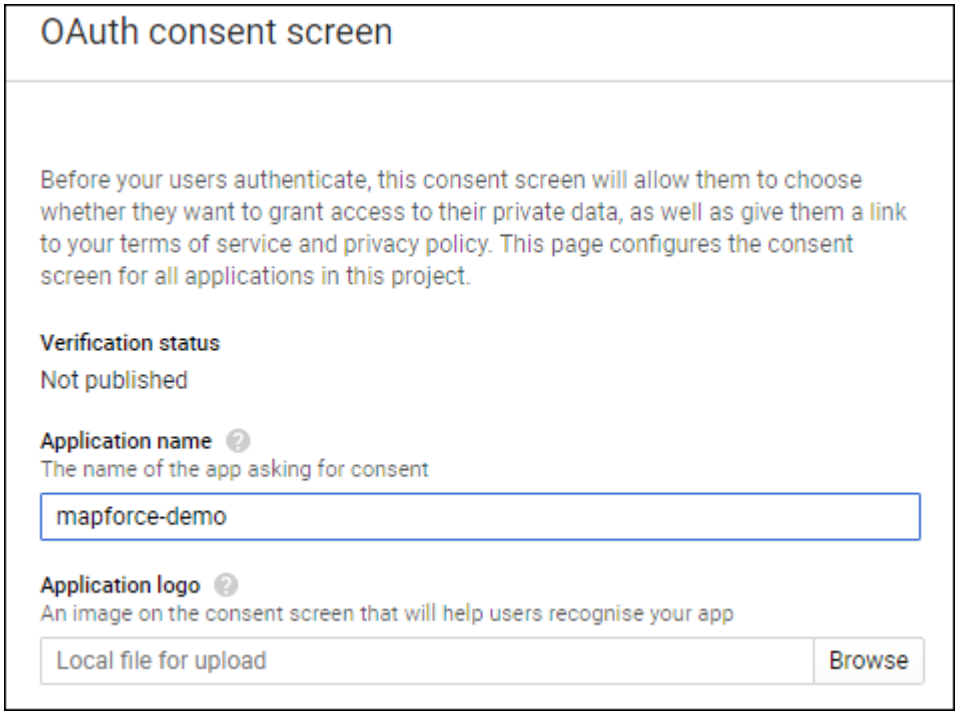
1. Google API コンソールにログインしてください (https://console.developers.google.com/)。
2. 新規のプロジェクトを作成します。



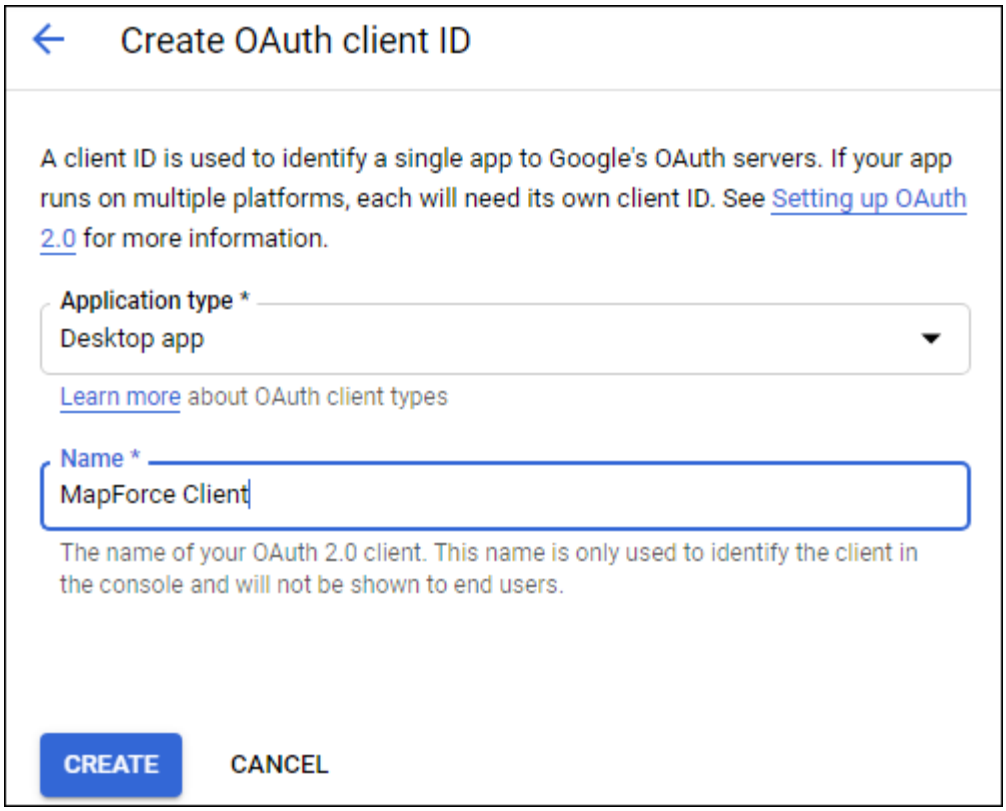
3. OAuth 同意画面をクリックしてください。
4. 所属機関内のユーザーのみにAPI アクセスを許可することを可能にするG Suite アカウント が既存の場合を除き 外部 をユーザー型として選択します。



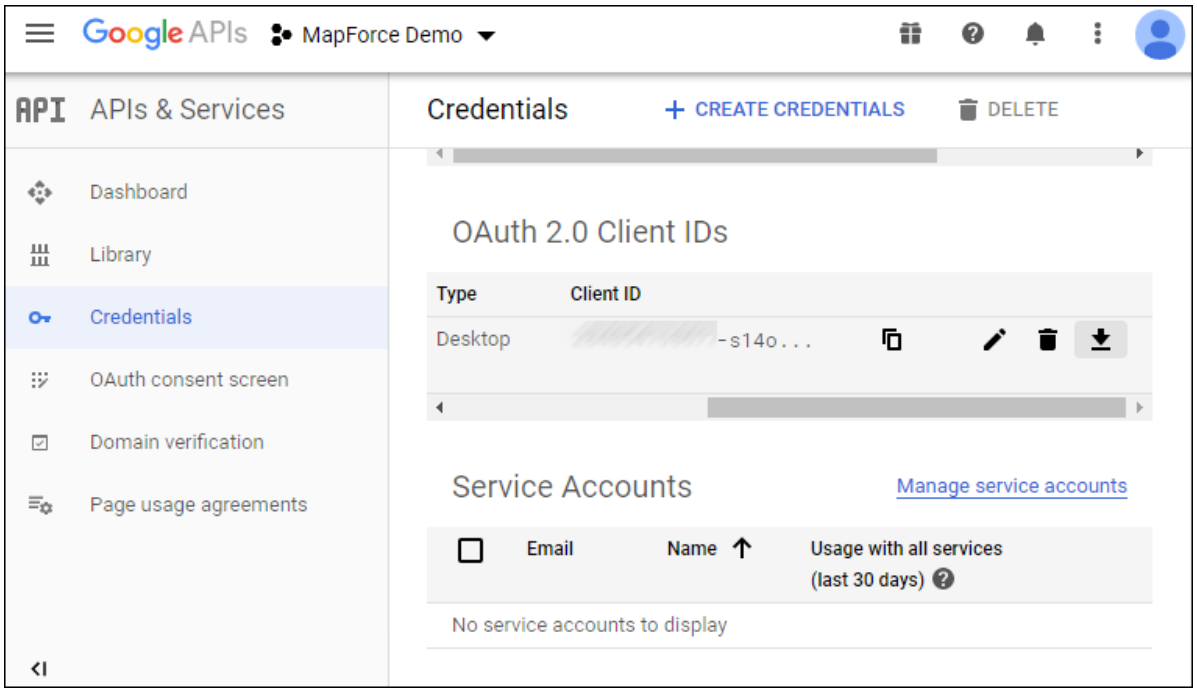
5. [mapforce-demo]をアプリケーション名として入力し、設定を保存します。



6. **認証情報の作成** をクリックし **OAuth Client ID** を選択します。
7. **デスクトップアプリ** と[MapForce Client] をクライアント名として入力します。



8. 「作成」をクリックします。ID が作成され 認証情報 ページで使用できるようになります。



9. をクリックして OAuth 2.0 認証情報を JSON ファイルとしてダウンロードします。

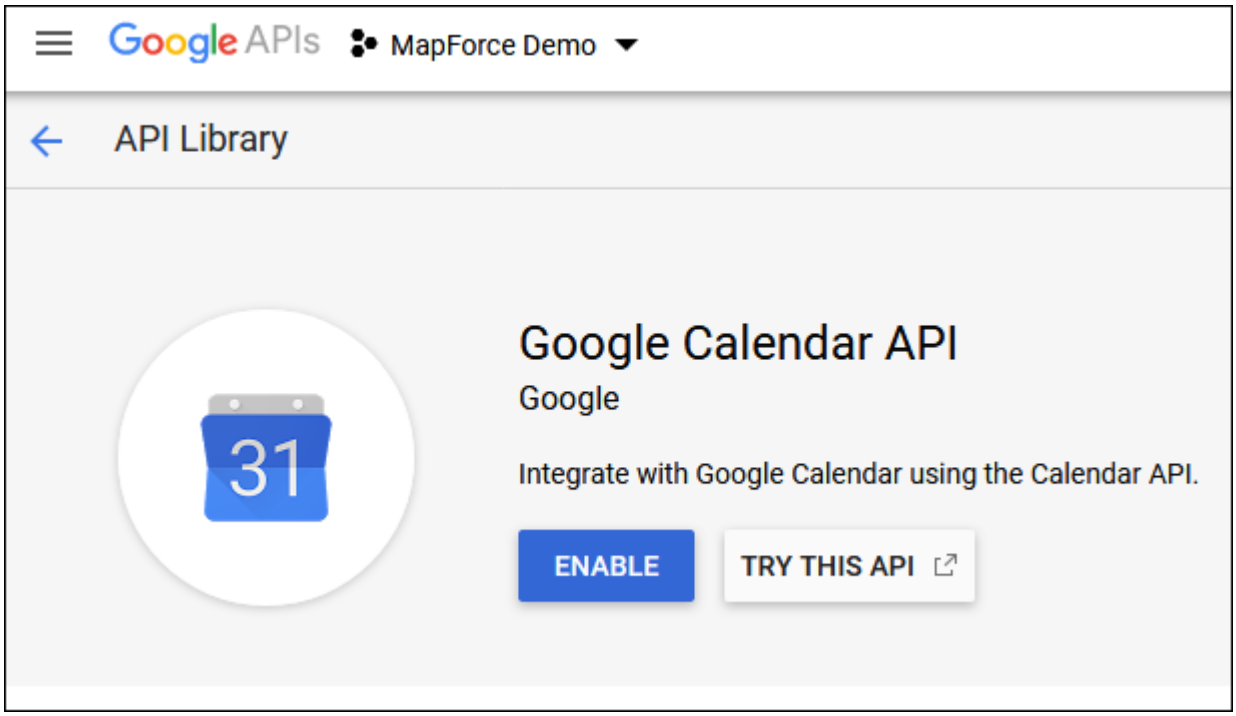
(C) 2015-2021 Altova GmbH

OAuth 2.0 認証情報がGoogle コンソールAPI から取得されました

1. 認証エンドポイント
2. トークンエンドポイント
3. クライアント ID
4. クライアントの秘密

Google カレンダーAPI の有効化

クライアントからの呼び出しを受け入れるには、このサンプルで使用されたGoogle カレンダーAPI が有効化されます。Google API コンソール内でライブラリをクリックし Google カレンダーAPI を検索し有効化します:



このサンプルではイベント エンティティの**list** メソッドが呼び出されます。
https://developers.google.com/calendar/v3/reference/events/list でこのAPI メソッドへの詳細な参照を見つけることができます。以下の重要点に注意してください:

1. このドキュメンテーションで指摘されているとおり `calendarId` がGoogle カレンダーの識別子である個所でGET リクエストを `https://www.googleapis.com/calendar/v3/calendars/calendarId/events` に送信することによりメソッドは呼び出される必要があります。**calendarId** リクエスト パラメーターは次のステップ内でMapForce から構成されます。
2. API メソッドの呼び出しは少なくとも以下のスコープの一つを必要とします:

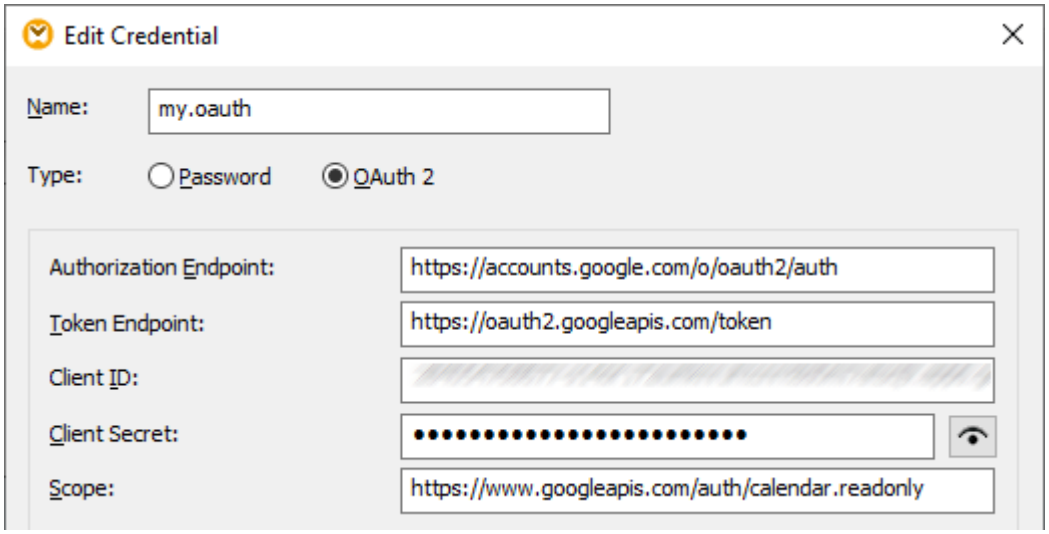
   - `https://www.googleapis.com/auth/calendar.readonly`
   - `https://www.googleapis.com/auth/calendar`
   - `https://www.googleapis.com/auth/calendar.events.readonly`
   - `https://www.googleapis.com/auth/calendar.events`

   OAuth 2 認証プロセス中で、マッピングは上記のスコープの一つを提供し、これは次のステップでも構成することができます。このサンプルの目的のためには[読み取り専用]スコープで十分です。

認証トークンのリクエスト

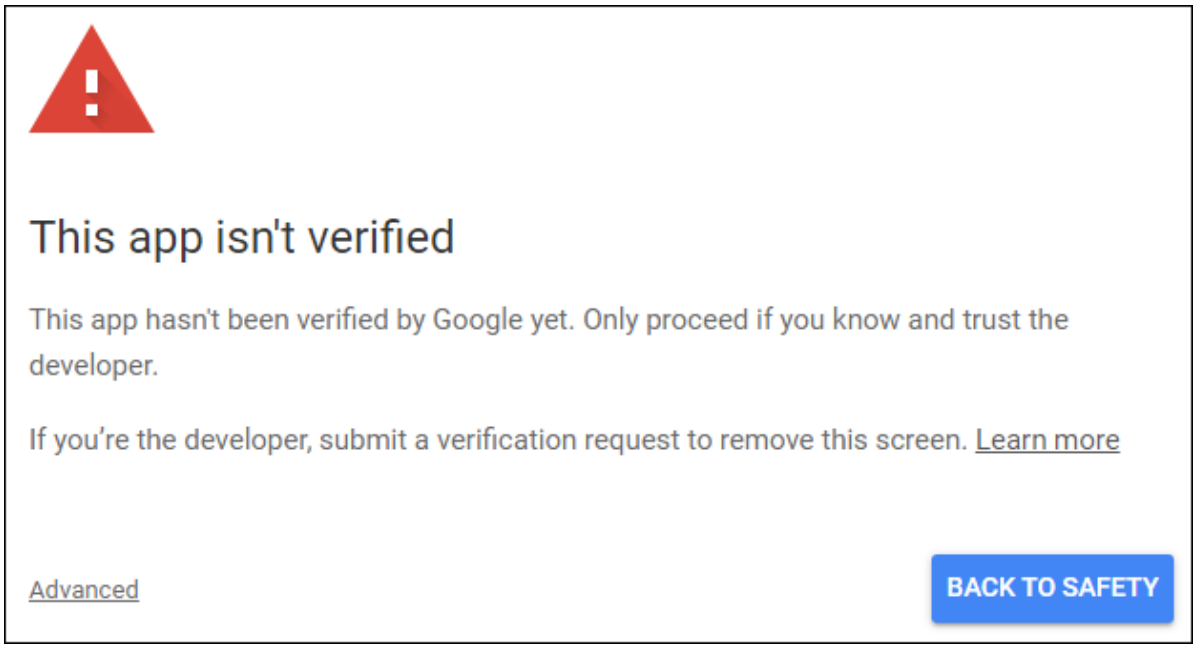
MapForce 内でマッピングをプレビューするには、下で説明される通りOAuth 2.0 認証詳細をマッピングに追加し、認証トークンをリクエストする必要があります。

1. MapForce 内でマッピング上の空白のエリアを右クリックし **資格情報マネジャーを開く**をコンテキストメニューから選択します。
2. **認証情報の追加**をクリックします。
3. 名前（このサンプルでは"my.oauth"）を入力して **OAuth 2** を型として選択します。
4. **認証エンドポイント**、**トークンエンドポイント**、**クライアント Id**、**クライアントの秘密**、**クライアントの秘密** テキストボックスを以前にダウンロードされているJSON ファイルからの対応する値で入力します。
5. `https://www.googleapis.com/auth/calendar.readonly` を **スコープ** テキストボックスに入力します。
6. 他の設定をそのままにします。

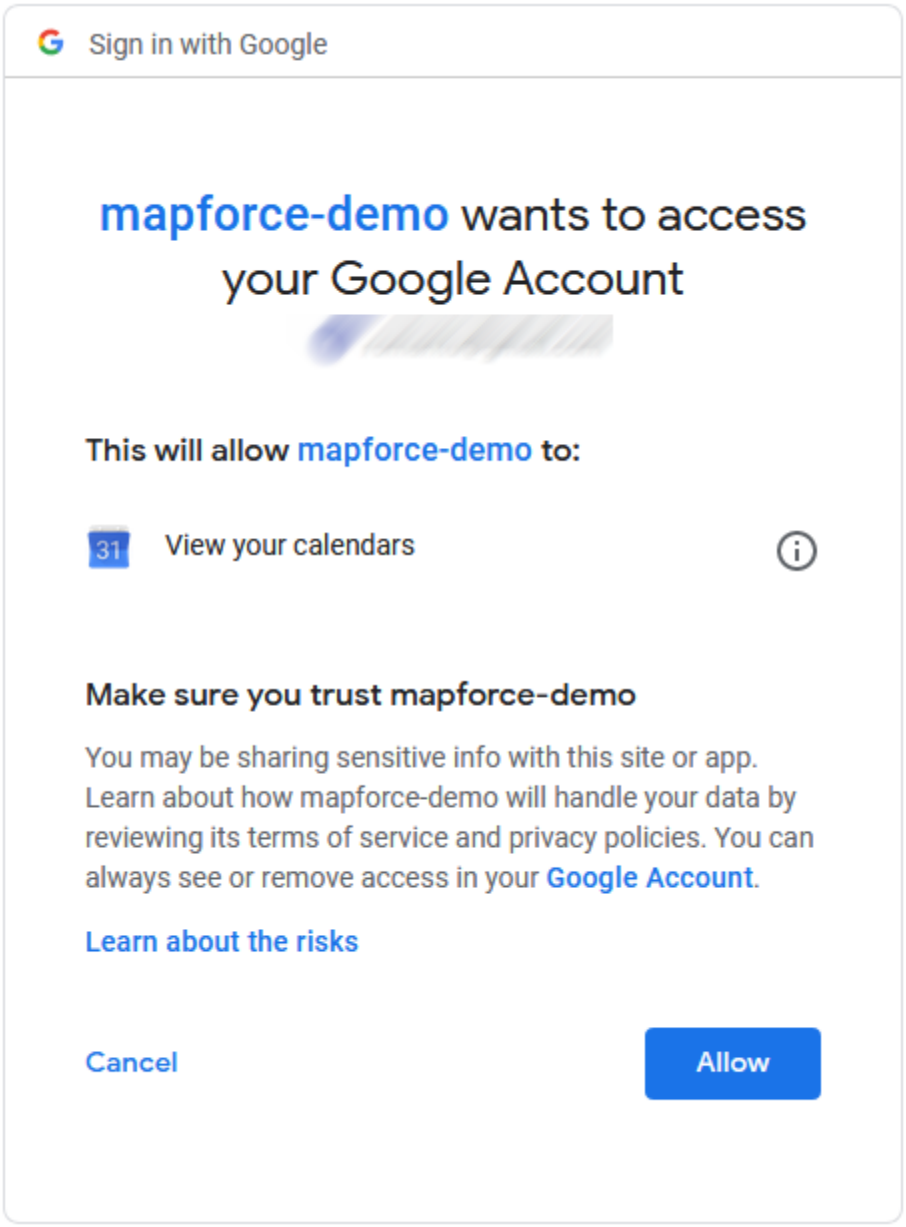


7. **アクセストークンのリクエスト** をクリックして認証サーバーからトークンを取得します（このサンプルではGoogle です）。Google アカウントに接続するように問うブラウザーウィンドウが開かれます。
8. Google アカウントにログインします。アプリの検証リクエストがGoogle に提出されていないため、以下のページが表示されます。

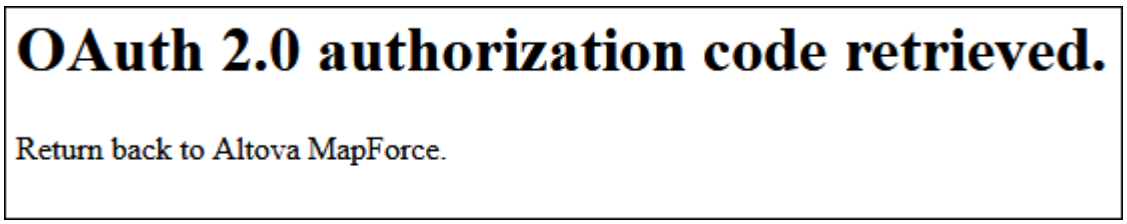
(C) 2015-2021 Altova GmbH



9. 高度な をクリックして mapforce-demo (安全ではない)に移動するをクリックします。

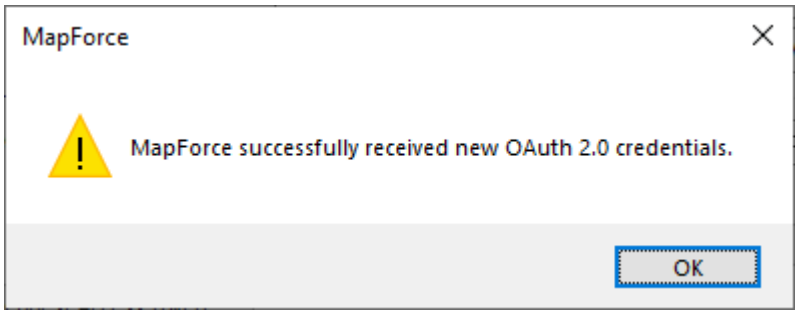


10. 許可をクリックします。ブラウザー内に確認が表示されます。

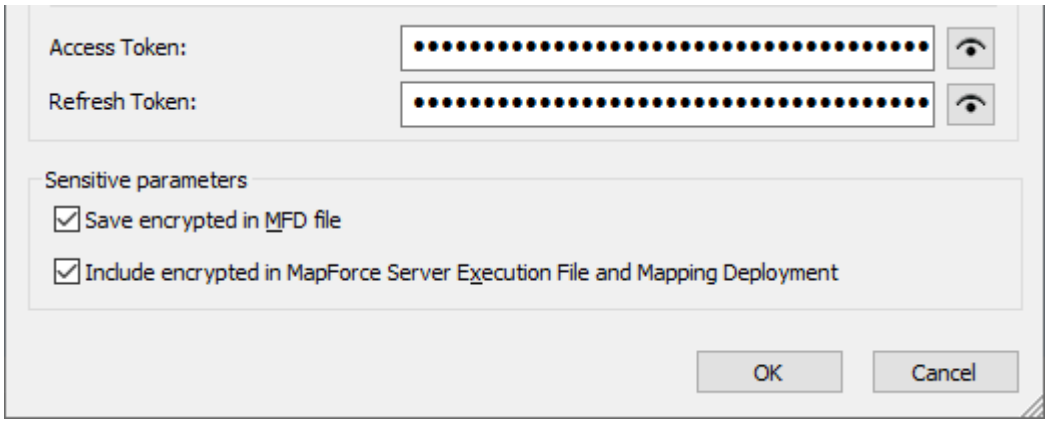


MapForce はOAuth 2.0 認証コードの抽出の成功も通知します。

(C) 2015-2021 Altova GmbH



11. 「OK」をクリックします。**アクセストークン** と **更新トークン** フィールドにデータが入力されていることにご注意して下さい。



12. マッピングを **GetCalendarEvents.mfd** asとして保存します。

> このチュートリアルでは **MFD ファイルに暗号化を保存する** チェックボックスが資格情報の編集ダイアログボックスで選択されています。このため、機敏なフィールド **クライアントの秘密**、**認証トークン**、**更新トークン** はマッピングの保存時にマッピングデザインファイル（.mfd）内で暗号化された書式で保存されます。

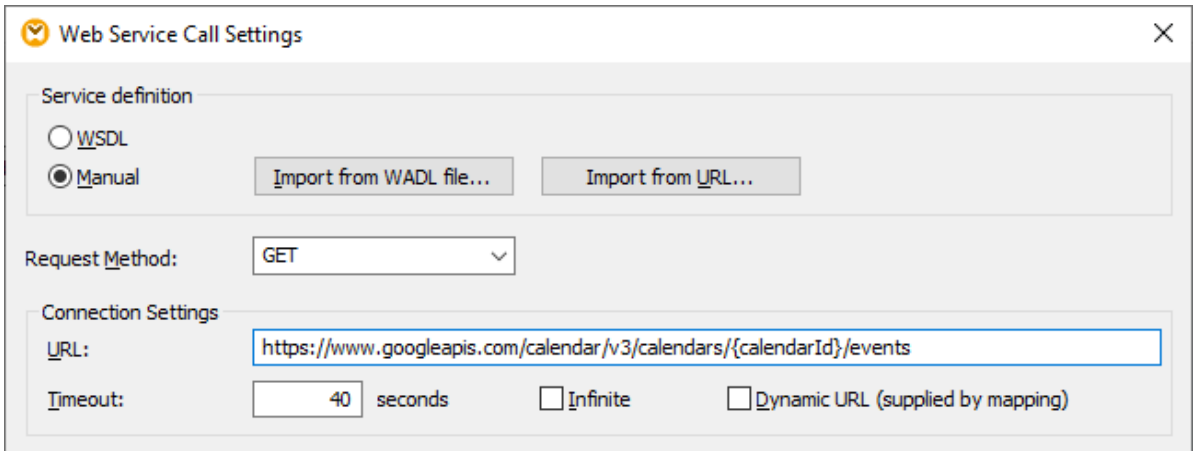
認証トークンは一定期間が過ぎると期限が切れることにご注意してください。これが発生すると、マッピングを実行することができなくなります（この段階では、マッピングはデザインされず、これは次のステップで発生します）。認証コードを手動で取得する必要がある場合 **アクセストークンのリクエスト** をクリックして上記のステップに従ってください。

Web サービスの呼び出しのデザイン

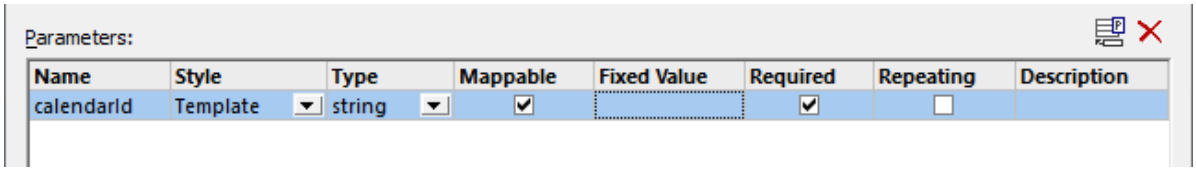
作成されたマッピング **GetCalendarEvents.mfd** はまだ何も行いません。含まれている唯一のアイテムはGoogle カレンダーAPI へのアクセスを有効化するOAuth 2.0 資格情報のみです。

Web サービスの呼び出しをMapForce 内でデザインしましょう:

1. **GetCalendarEvents.mfd** マッピングを開きます。
2. 「**挿入**」メニューから「**Web サービス関数**」をクリックします。Web サービス呼び出し設定ダイアログボックスが表示されます。
3. **手動**をクリックします。
4. **GET** をリクエストメソッドとして選択しURL を前のステップで入力されたWeb サービスに入力します: `https://www.googleapis.com/calendar/v3/calendars/calendarId/events`。
5. **calendarId** はパラメーターとして提供される必要のあるプレースホルダのため、以下に示される通り中かっこで囲みます。



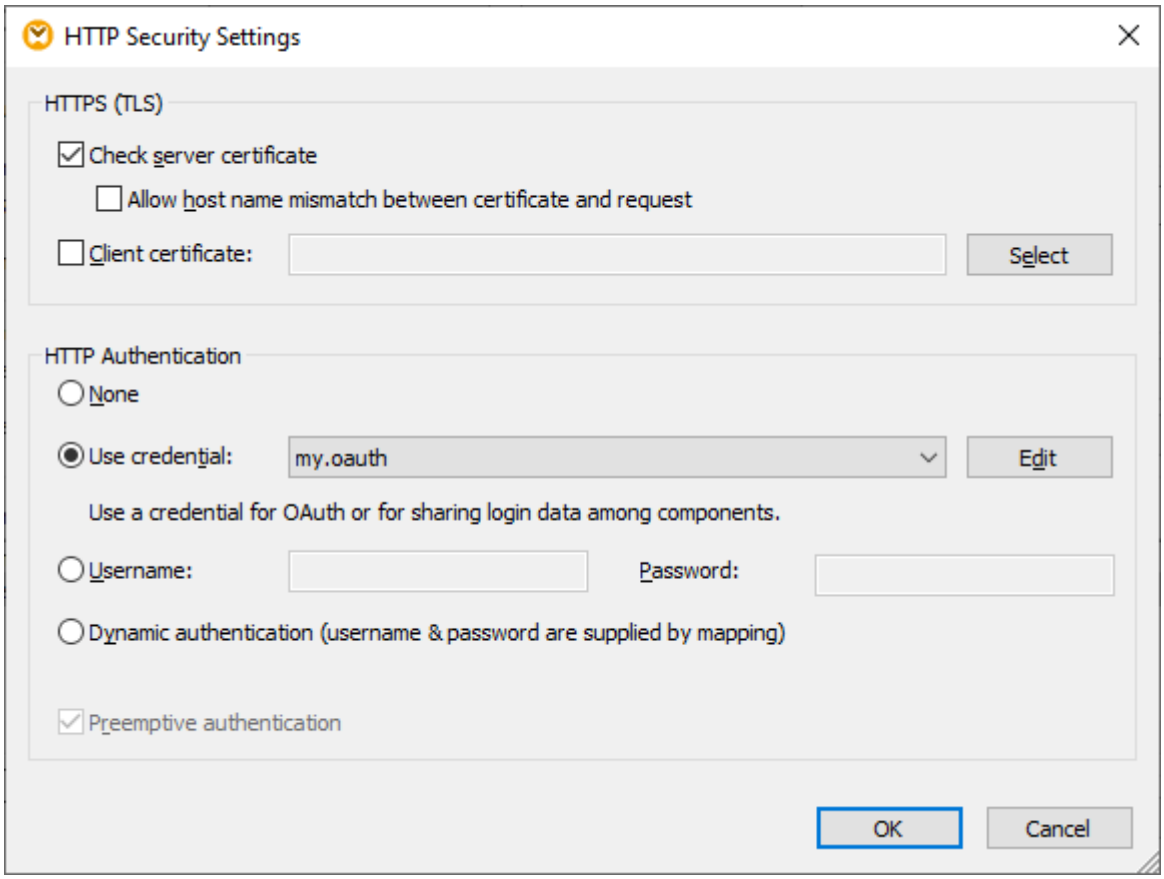
6. **パラメーターの追加** ボタンをクリックし以下のように パラメーターの詳細を定義します:



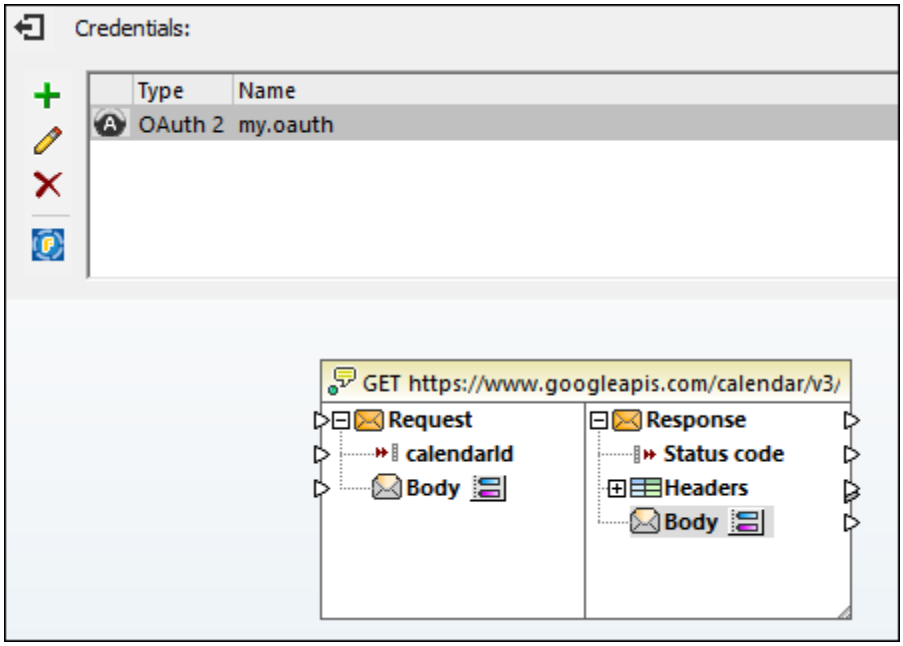
上記の構成では[テンプレート]スタイルにより中かっこで囲まれたURL 部分をランタイムの パラメーター値と置き換えることができます。[マッピング可能]はマッピングから値を提供できることを意味します(例えば、定数から、または入力パラメーターから値を提供すること)。最後に、パラメーターは必須とマークされます。これは呼び出しはこの パラメーター無しでは発生しないからです。 詳細に関しては、Web サービス パラメーターの定義を参照してください。

7. HTTP セキュリティ設定の横の「編集」ボタンをクリックします。
8. [HTTP セキュリティ設定] ダイアログボックスで、**資格情報を使用する** を選択し、以前に構成されている[my.oauth] 資格情報レコードを選択します。

(C) 2015-2021 Altova GmbH

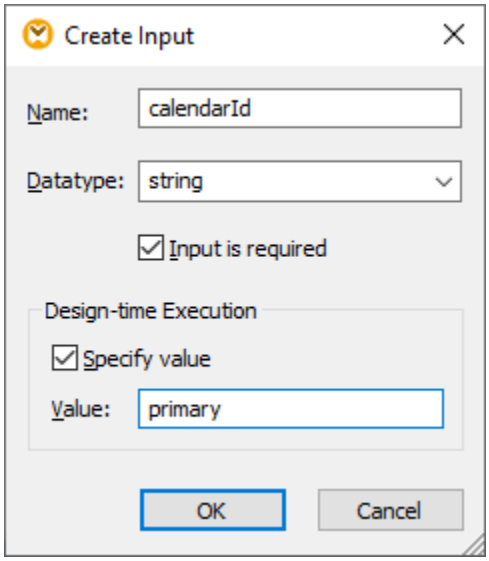


ここまでに構成済みのW eb サービスには以下のマッピングの外観を持っています:



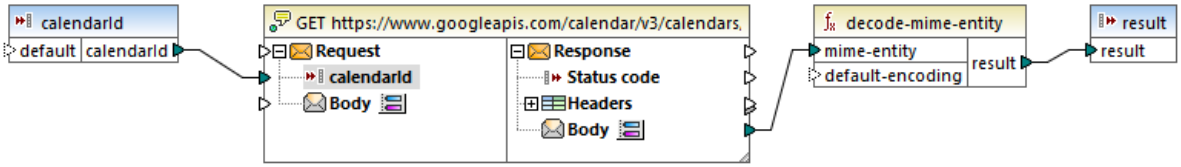
以下のステップに従いデザインを完了することができます:

1. **挿入** メニューで **入力の挿入**をクリックして、以下のようにコンポーネントを構成します:



上で表示されている通り、入力コンポーネントはデザインタイム値[プライマリ]を持っています。API のドキュメントによると、値[プライマリ]は API サーバーに現在ログインしているユーザーのプライマリ Google カレンダーにアクセスするように命令します。この値はデザイン時の値でないでマッピングをプレビューする際にのみ適用することが可能です。マッピングがサーバー環境で作動する場合、ランタイムで希望される値を提供する必要があります。

2. decode-mime-entity 関数をライブラリウィンドウからマッピングエリアにドラッグします。この関数はサーバーから取得した MIME ボディを文字列に変換します。 関数をマッピングに追加するを参照してください。
3. **挿入** メニューで **出力の入力** をクリックし、プレーンな文字列として結果を出力するロールを持つ単純型出力コンポーネントを追加します。
4. コンポーネント間の接続が以下に表示されているようになっていることを確認してください。



ここで MapForce 内でのデザインパートは終わりです。

マッピングの実行のテスト

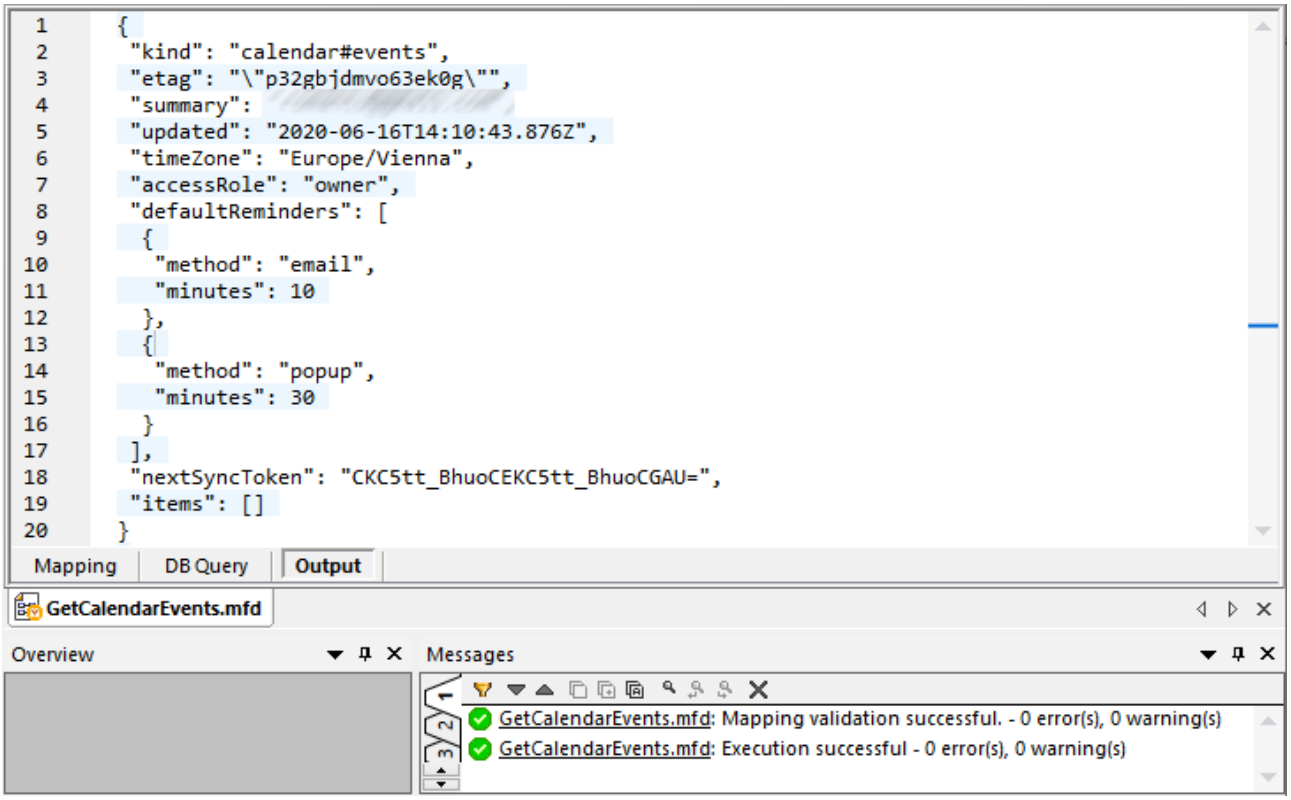
MapForce 内でマッピングの実行をテストする場合、click the **出力** タブをクリックし、メッセージウィンドウ内で表示される結果を確認してください。

[承認されていません(401)] などの認証エラーが表示されると、以下のトラブルシューティングのヒントを考慮してください:

1. Google カレンダー API が有効化されていることを確認してください。Google カレンダー API の有効化を参照してください。
2. 以前に取得されたアクセストークンの期限が切れている場合、新規の認証トークンのリクエスト してください。
3. 全ての OAuth 2.0 詳細が MapForce 内に正確に入力されているか再度確認してください。

実行と MapForce からの OAuth 2.0 認証に成功すると、マッピング出力は以下に表示されるようなマッピングになります:

(C) 2015-2021 Altova GmbH



このサンプル内に存在するようなイベントが存在しない Google アカウントを使用すると [アイテム] 配列はレスポンス内では空白です。しかしながら、Google カレンダーにイベントを追加してマッピングを再度実行すると、出力はこれを反映します。サイドメモとしては、デフォルトのカレンダー以外の公開されているカレンダーからイベントを抽出することもできます。例えば、[Holidays in United States] などの公開されているカレンダーからデータを抽出することができます。これを行うには **calendarId** パラメーターを `primary` の代わりに `en.usa#holiday@group.v.calendar.google.com` に変更してください。

API 呼び出しに追加可能な他のパラメーターの詳細に関しては https://developers.google.com/calendar/v3/reference/events/list で API メソッドのドキュメンテーションを参照してください。

9.10 デジタル証明書の管理

デジタル証明書の管理は、クライアントコンピューターとWeb サーバーの間のデータのやり取りの重要な一部です。マッピングはMapForceによりWindows 上でのみだけでなく、Linux またはmacOS サーバー上でMapForce Server（スタンドアロン、または、FlowForce Server 実行）により実行することができるため、このセクションは多種のプラットフォーム上でのHTTPS 証明書の管理について説明しています。

安全なHyperText Transport Protocol（HTTPS）のコンテキストでは、サーバーとクライアント間の証明書を区別することが重要です。

サーバーの証明書

サーバーの証明書は、サーバーをMapForce などのクライアントアプリケーションの信頼されているエンティティとして識別します。サーバーの証明書は、商業証明機関により電子的に署名され手いる可能性がありますが、所属期間により自身で署名される場合もあります。この場合、MapForce 内でマッピングをデザイン中に、次の設定を指定することができます:

- サーバーの証明書がチェックされるか否か。
- 証明書の名前とホストの名前の間に不一致が検出された場合、リクエストが継続されるか否か。

MapForce のHTTP セキュリティ設定 ダイアログボックスで以下の設定を使用することができます（次を参照してください:HTTP セキュリティの設定）。サーバーの証明書 のチェックを選択する場合、以下に注意してください:

- 信頼される証明機関により証明書が署名されるWeb サーバーを呼び出す場合、使用中のオペレーティングシステムはサーバーの証明書を信頼するように構成されている場合が多く、追加の構成は必要ありません。
- 自身により署名される証明書を与えるWeb サーバーを呼び出す場合、（例えば、所属機関内のローカルネットワークサーバー）オペレーティングシステムを構成し、証明書を信頼するように構成する必要があります。

多くの場合、ブラウザーのアドレスバー内にWeb サービスのURL を入力し、オペレーティングシステムとWeb サーバー間の信頼のレベルをチェックすることができます。サーバーが信頼されていない場合、または、オペレーティングシステムがサーバーを信頼するように構成されていない場合、ブラウザーは、「この接続は信頼されていません」「このWeb サイトのセキュリティ証明書には問題があります」などのメッセージを表示します。ブラウザーがオペレーティングシステム以外の証明書 データベースを使用する場合、（例えば、Ubuntu やFirefox を使用など）ブラウザーを使用してWeb サーバーの信頼レベルをチェックすることはできません。

Windows 上では、サーバーへの信頼をブラウザーの命令に従い、または、システムの信頼された証明機関ストアにインストールすることにより信頼を構築することができます（次を参照してください:Windows 上でサーバーの証明書を信頼する）。macOS 上では、Safari 内で同様の操作を行うことができます（次を参照してください:macOS 上のサーバーの証明書を信頼する）。Linux に適用可能な命令に関しては、次を参照してください:Linux 上のサーバーの証明書を信頼する。

クライアントの証明書

サーバーの証明書はサーバーを信頼されるエンティティとして識別するために使用されますが、クライアントの証明書は主にWeb サーバーに対しして呼び出しを認証するために使用されます。クライアントの証明書を必要とするWeb サーバーを呼び出す場合、クライアント 構成命令のためのWeb サーバーの管理者に連絡をする必要があります。IIS（Internet Information Services）を例にすると、Web サーバーはHTTPS とクライアントの証明書を処理するために以下の方法により構成されます:

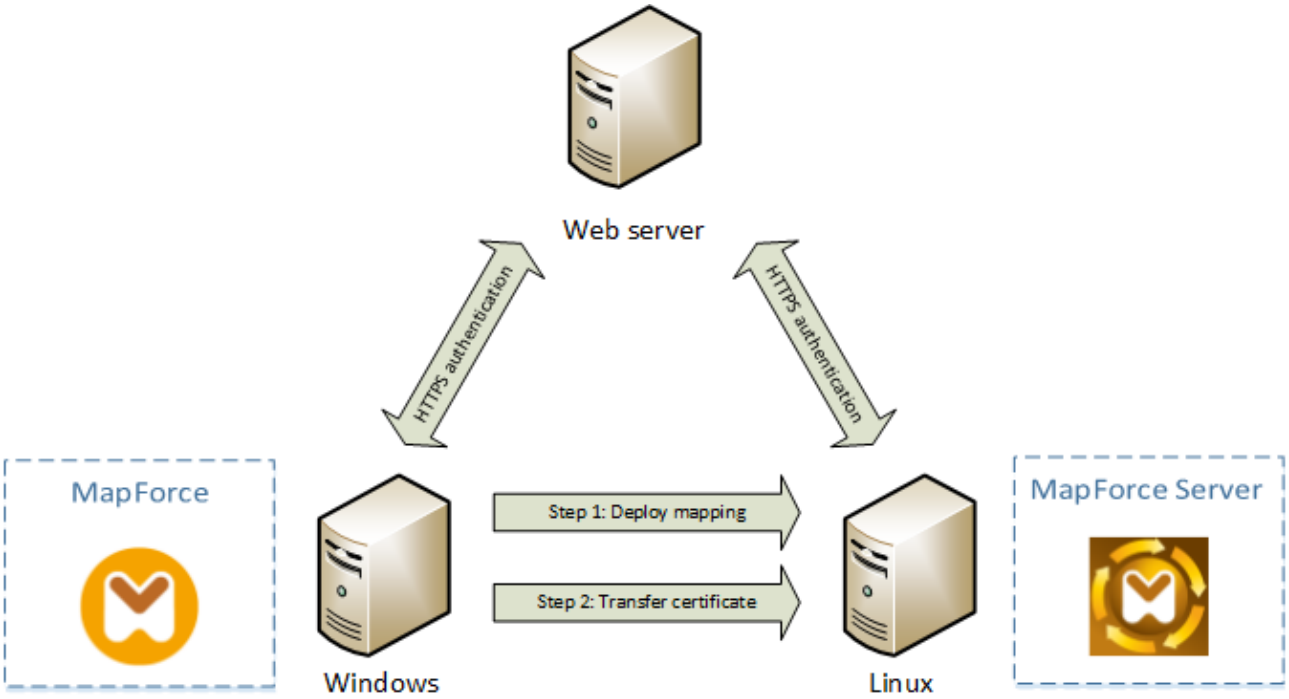
- HTTPS を必要とし、クライアントの証明書を無視します。
- HTTPS を必要とし、クライアントの証明書を受け入れます。
- HTTPS を必要とし、クライアントの証明書を必要とします。

Web サービスリクエストの成功、または失敗は、Web サーバーとクライアントとアプリケーションの構成により異なります。例えば、Web サーバーがクライアントの証明書を必要とするように構成されている場合、呼び出しに成功するには、呼び出しアプリケーションは有効なクライアントの証明書を表示する必要があります。

MapForce のパースペクティブからは、HTTPS を使用するWeb サービス呼び出しを含むマッピングにも同様が適用されます。特に、この

ようなマッピングの実行に成功するには、Web サーバーはクライアントの証明書を受け入れる、または、を必要とするように構成され、マッピングが実行されるオペレーティングシステムは、Web サーバーへの正確なクライアントの証明書を提供します。

下のダイアグラムは、MapForce 内で仕様されているクライアントの証明書が、MapForce Server を作動している Linux サーバーに転送されるシナリオを表しています。ターゲット オペレーティングシステムに証明書が転送されると、MapForce Server は証明書を使用して自身を Web サーバーに対して認証し、マッピングを実行することができます。



クライアントの証明書を持つマッピングを他のコンピューターにデプロイする方法

Web サービス呼び出し内の HTTPS 認証に関しては、MapForce は Secure Sockets Layer (SSL) プロトコールの後続である Transport Layer Security (TLS) を HTTP に加え、使用することができます。クライアント 実装、または、サーバーが TLS をサポートしない場合、SSL のフィードバックが起こる可能性があります。

複数のプラットフォーム上で、クライアントの証明書 認証を使用した Web 呼び出しをサポートするには、MapForce (と MapForce Server) は各プラットフォームの証明書の管理 実装に依存し、証明書の管理がオペレーティングシステムの基と成るスコープにより証明書が管理されることを保証します。各オペレーティングシステムは、証明書の管理のために異なるサポートを下のテーブルに示されるようにサポートします。

プラットフォーム	証明書の管理と実装
Windows	Windows 上では、証明書 スナップインを使用して証明書を管理することができます (次を参照してください: Windows 上で証明書ストアにアクセスする)。 TLS サポートは (*SChannel* として既知の) *安全なチャンネル* ライブラリ を介して使用することができます。
Linux	Linux 上では、OpenSSL (`openssl`) コマンド ライン ツールとライブラリを使用して、証明書を管理することができます。MapForce Server がインストールされている Linux マシン上で OpenSSL サポートが使用できない場合、証明書を管理する前に、ダウンロードしてインストールする必要があります。

プラットフォーム	証明書の管理と実装
	OpenSSL ライブラリを介して TLS サポートを使用することができます (https://www.openssl.org/)。
macOS	macOS 上では、「検索 > アプリケーション > ユーティリティ」にあるキーチェーンアクセスマネジャーを使用して証明書を管理することができます。 TLS サポートは、オペレーティングシステムにネイティブな安全なトランスポートライブラリにより提供されます。

MapForce からの呼び出しを想定する既に成功している同じ Web サービスを使用する場合、Windows オペレーティングシステム上でマッピングを実行すると、更なる証明書の構成は通常必要ありません(Windows 上でマッピングの実行に成功するための条件に関しては、次を参照してください:Windows 上のクライアント証明書)。しかしながら、Windows コンピューター上で MapForce を使用して、マッピングをデザインし、(異なるオペレーティングシステムで作動する)他のコンピューターにデプロイするには、クライアントの証明書を保管しない、または、デプロイされたパッケージにコピーします。Web サービスの呼び出し(とマッピング) の実行の成功に関しては、クライアントの証明書がターゲット オペレーティングシステム上に存在する必要があります。

Windows システムから証明書を他の Windows ベースのコンピューターに転送する場合、必要とする証明書を(秘密キーを使用して)ソース システムからエクスポートします。(次を参照してください:Windows 上で証明書をエクスポートする) 。ターゲット オペレーティング システム上の「現在のユーザー-個人用」ストアに同じ 証明書をインポートします(次を参照してください:Windows 上のクライアント証明書).

Linux とmacOS プラットフォームにクライアントの証明書を転送する方法に関しては、次を参照してください:Linux 上のクライアント証明書 と macOS 上のクライアント証明書。

9.10.1 Linux 上でサーバー証明書を信頼する方法

Linux 上では信頼される証明書をシステムの証明書ストアに以下のようにインポートすることができます。

サーバーの証明書の信頼性が確実の場合、次のステップをおこなってください。

Debian とUbuntu では、以下のステップに従います:

1. Web サーバーの証明書ファイルを次のディレクトリにコピーします。

```
sudo cp /home/downloads/server_cert.crt /usr/local/share/ca-certificates/
```

2. 証明書を以下のように更新します

```
sudo update-ca-certificates
```

CentOS では、以下のステップに従います:

1. `ca-certificates` パッケージをインストールします:

```
yum install ca-certificates
```

2. 動的な証明機関構成機能を有効化します:

(C) 2015-2021 Altova GmbH

```
update-ca-trust enable
```

3. サーバーの証明書を次のディレクトリにコピーします:

```
cp server_cert.crt /etc/pki/ca-trust/source/anchors/
```

4. コマンドを使用します:

```
update-ca-trust extract
```

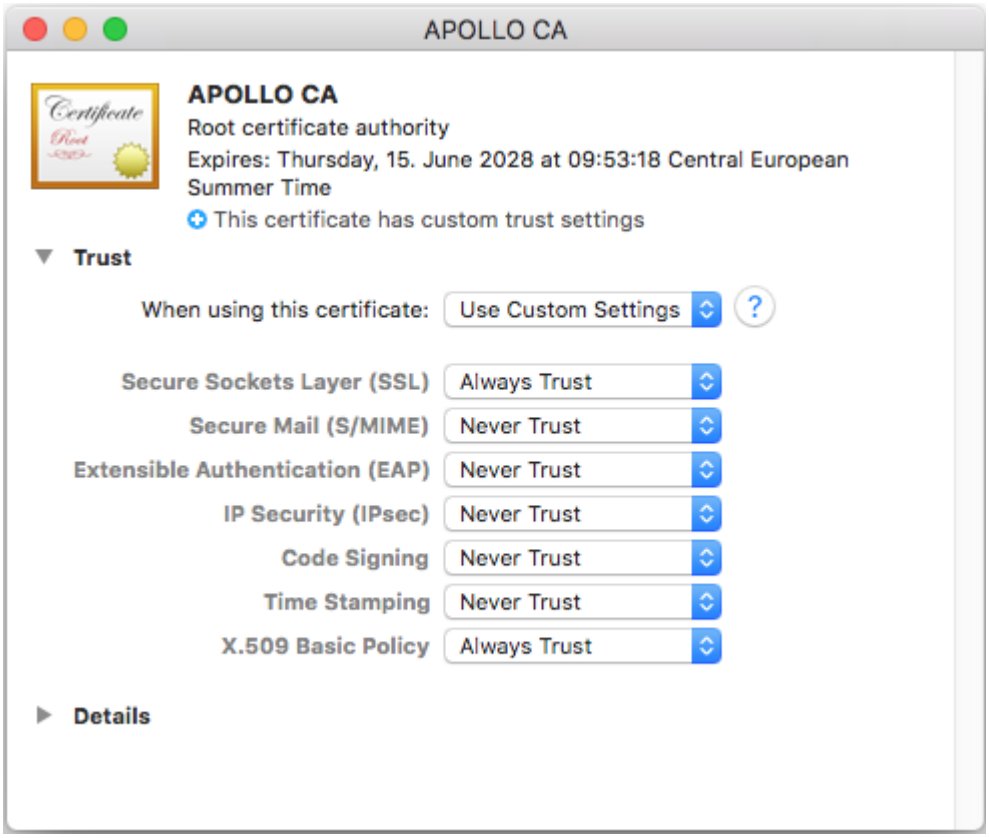
ブラウザーからのみサーバーにアクセスする必要がある場合、各ブラウザーストアに証明書をインポートするのみで十分です。各ブラウザーにより正確な手続きは異なります。例えば、Firefox 59.0.2 では、以下を行うことができま

1. 「**オプション | プライバシーとセキュリティ**」から、「**証明書を表示**」をクリックします。
2. 「**認証局証明書**」タブで、「**インポート**」をクリックし、以前に作成されているルート証明書ファイルを参照します。
3. プロンプトされると「**この証明書をウェブサイトの識別に使用する**」を選択します。

9.10.2 macOS 上でサーバー証明書を信頼する方法

macOS では、信頼される証明書を Keychain Access に以下のようにインポートすることができます。

1. Keychain Access を実行します。
2. 「**システム**」をクリックして、「**証明書**」をクリックします。
3. 「**ファイル**」メニューから「**アイテムのインポート**」をクリックします。
4. 信用される証明書を参照し、「**開く**」をクリックします。
5. Keychain Access パスワードをプロンプトされると入力し、「**キーチェーンの変更**」をクリックします。
6. 証明書をダブルクリックし、「**信頼**」セクションを拡張し、「**常に信頼する**」を選択します。



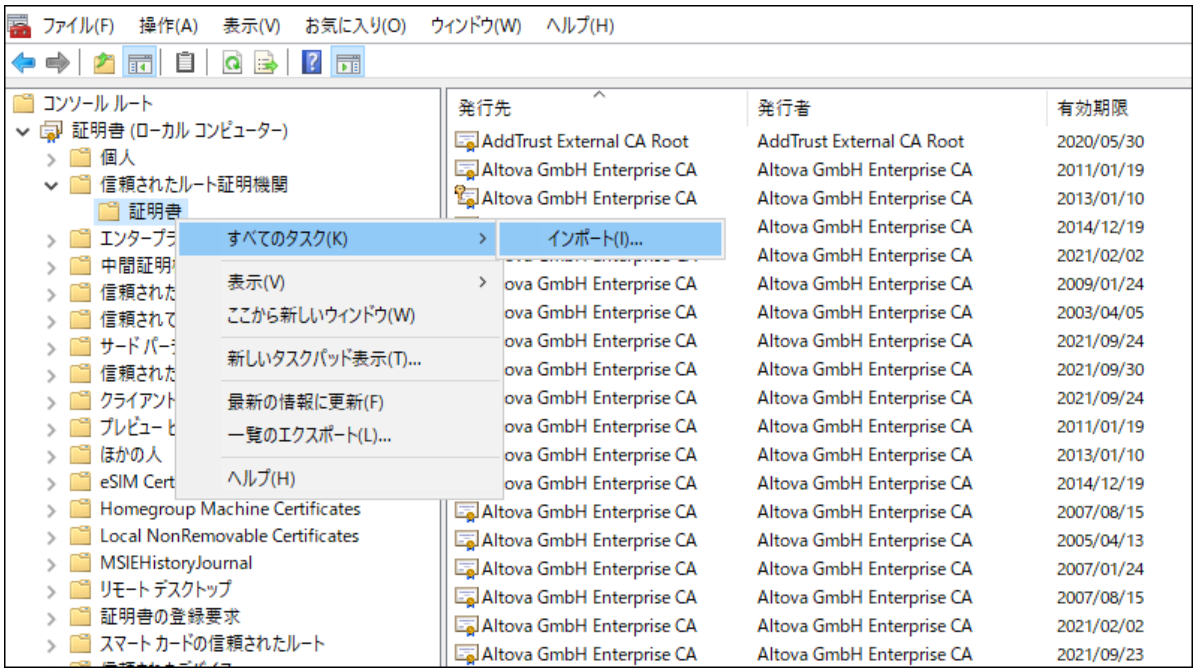
9.10.3 Windows 上でサーバー証明書を信頼する方法

Windows では、信頼された証明書をシステム証明書ストアに以下のようにインポートすることができます:

1. Windows 証明書ストアをコンピューターアカウントのために開きます、Windows 証明書ストアへのアクセスを参照してください。

   > Web サーバー証明書の認証を確認するだけの場合、次のステップを実行します。

2. 「信頼されたルート証明書機関」で「**証明書**」を右クリックし、「**すべてのタスク | インポート**」を選択し、証明書のインポートウィザードに従ってください。



詳細に関しては https://technet.microsoft.com/en-us/library/cc754489(v=ws.11).aspx を参照してください。

9.10.4 Windows 上で証明書ストアにアクセスする方法

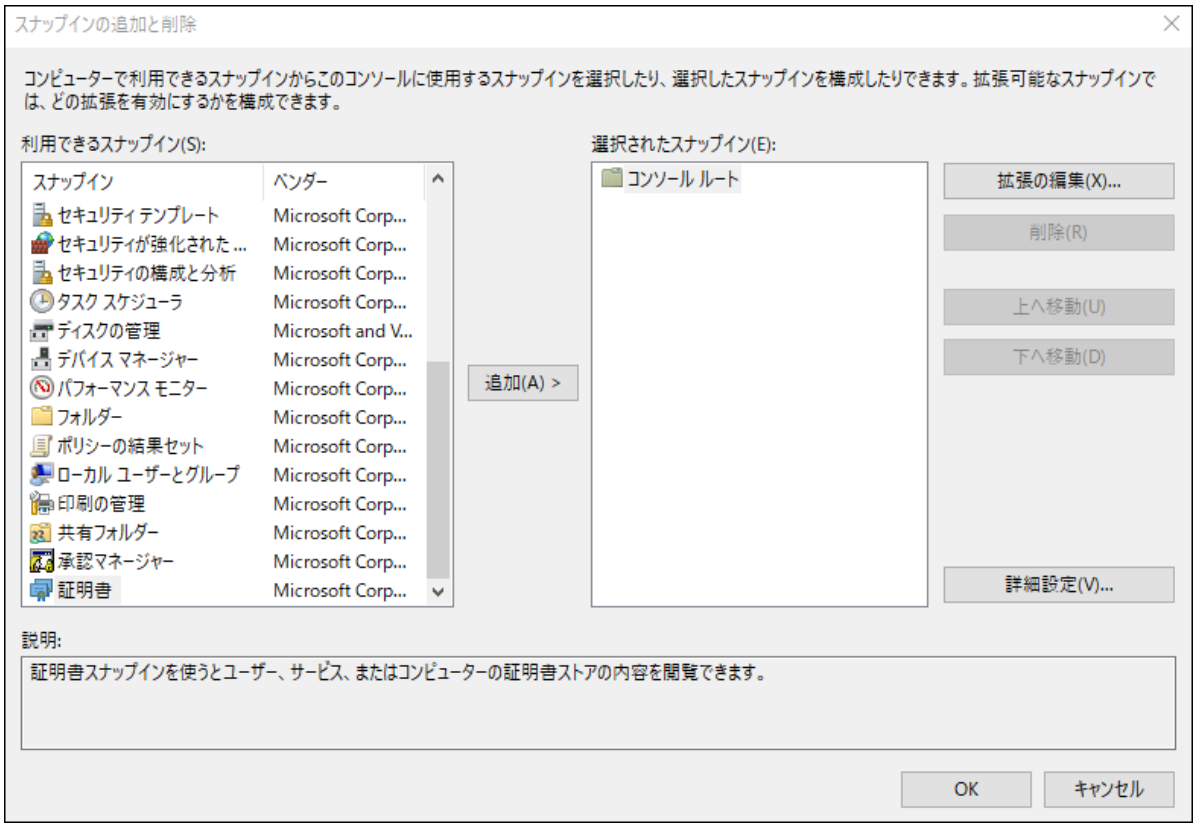
Windows 上では、Microsoft 管理コンソール(MMC) スナップイン、または、Internet Explorer から証明書を管理することができます。

(現在 Windows を使用するユーザーのための)証明書 スナップインを開く方法 :

- `certmgr.msc` をコマンドラインで実行します。

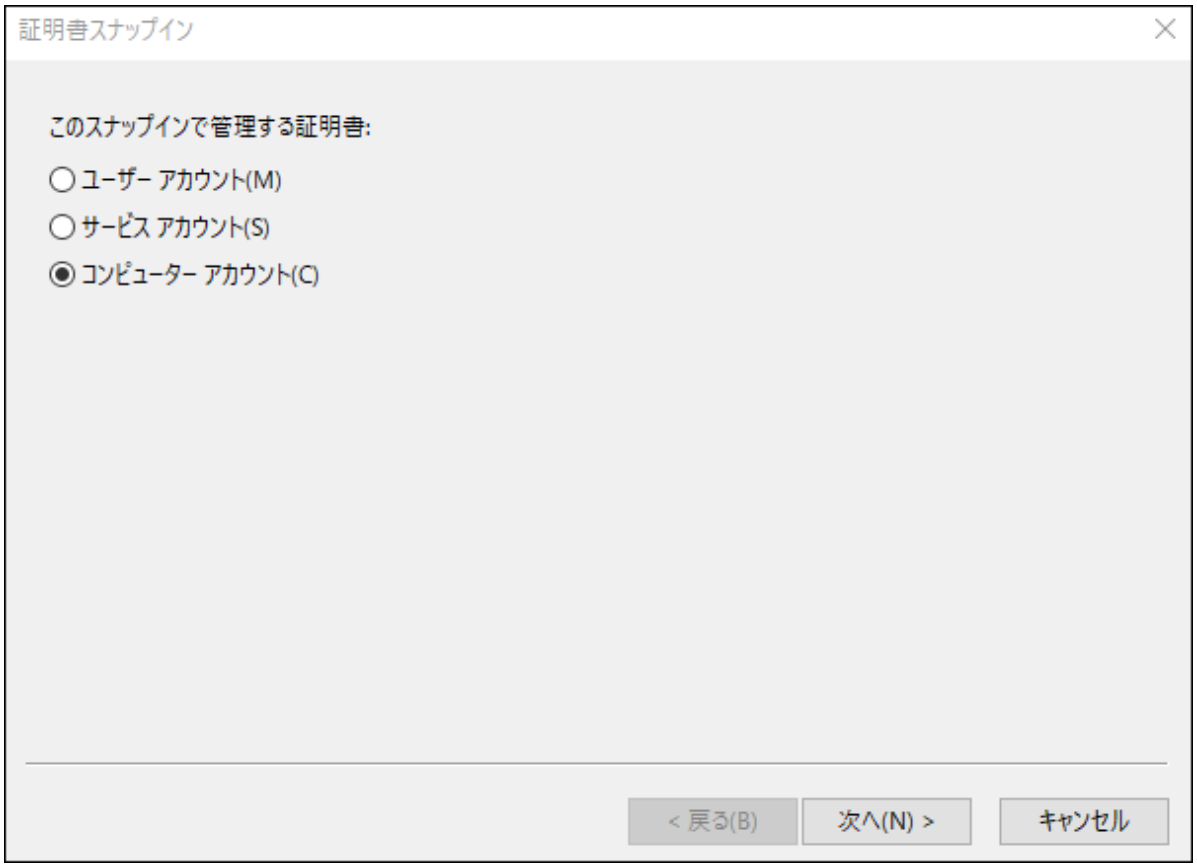
(コンピューターのアカウントのために)証明書スナップインを開く方法 :

1. `mmc` をコマンドラインで実行します。
2. 「ファイル」メニューから、「スナップインの追加/削除」をクリックします。
3. 「証明書」をクリックして、「追加」をクリックします。

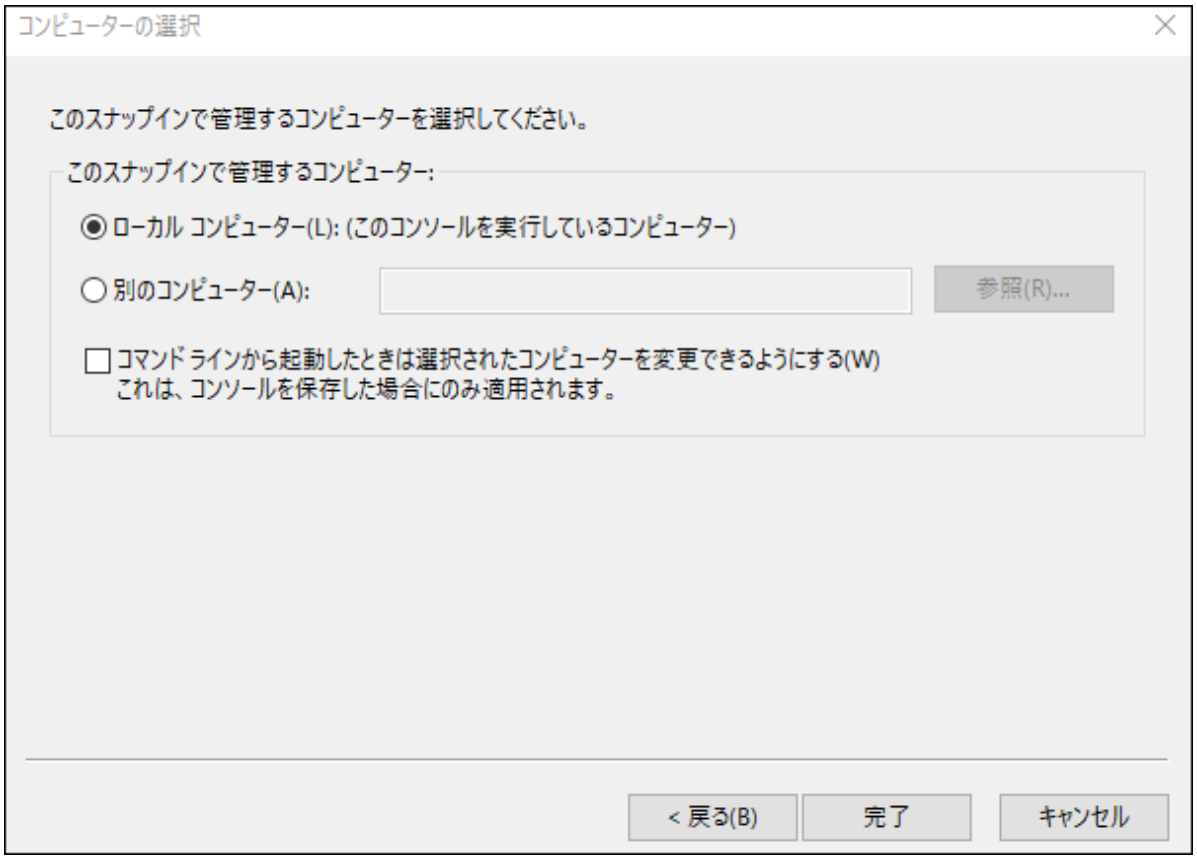


4. 「**コンピュータのアカウント**」をクリックして、「**次へ**」をクリックします。

(C) 2015-2021 Altova GmbH



5. 「ローカルコンピューター」をクリックして、「完了」をクリックします。



9.10.5 Windows から証明書をエクスポートする方法

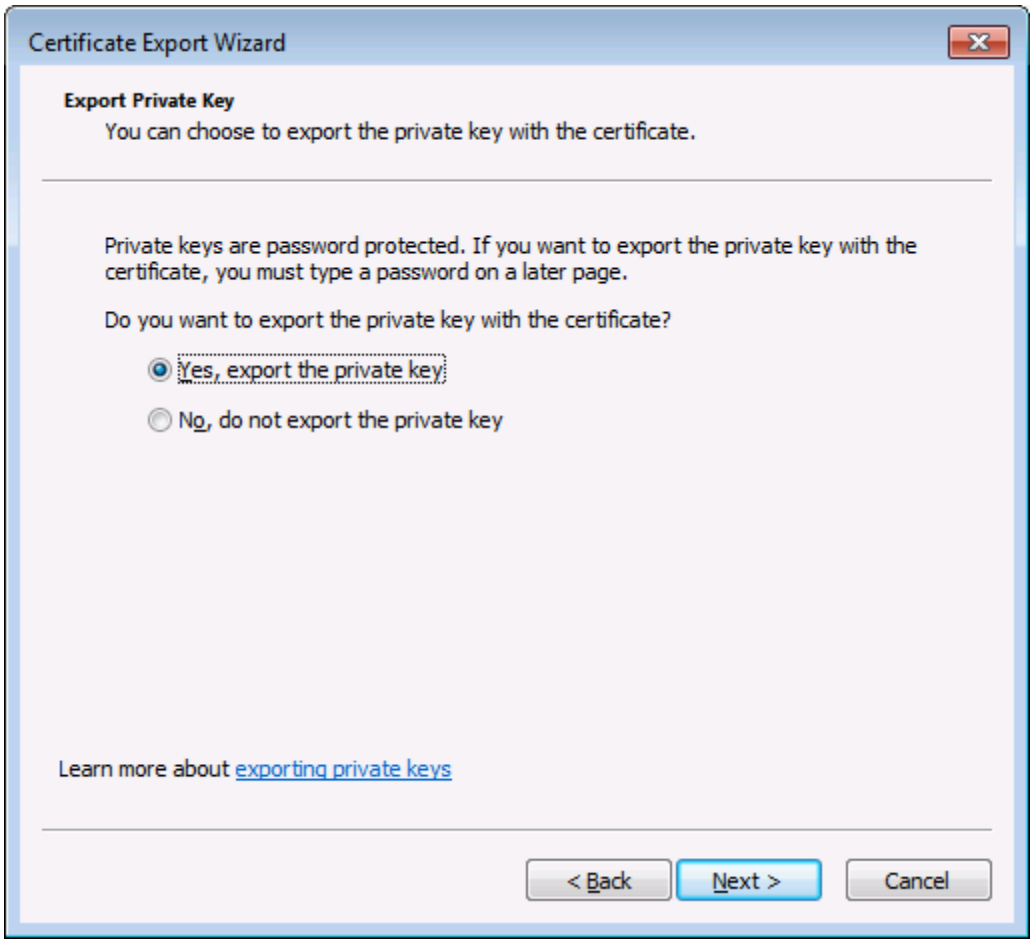
HTTPS を使用して Web サービスを呼び出し、MapForce Server または FlowForce Server を実行する macOS または Linux サーバーにデプロイされるマッピングは、Windows 上でマッピングをデザインしテストする、同じクライアントの証明書が非 Windows オペレーティングシステムで使用可能である必要があります。このようなマッピングを MapForce Server を実行する非 Windows オペレーティングシステムで実行するには、Windows から必要とする秘密キーを持つ証明書をインポートし、ターゲット オペレーティングシステムにエクスポートします。

Windows から秘密キーを持つ証明書をエクスポートする方法:

1. Windows 上で、証明書 スナップインを開きます(次を参照してください: Windows 上で証明書ストアにアクセスする)。
2. エクスポートする証明書を右クリックし、「すべてのタスク」をポイントし、「エクスポート」をクリックします。
3. 「次へ」をクリックします。

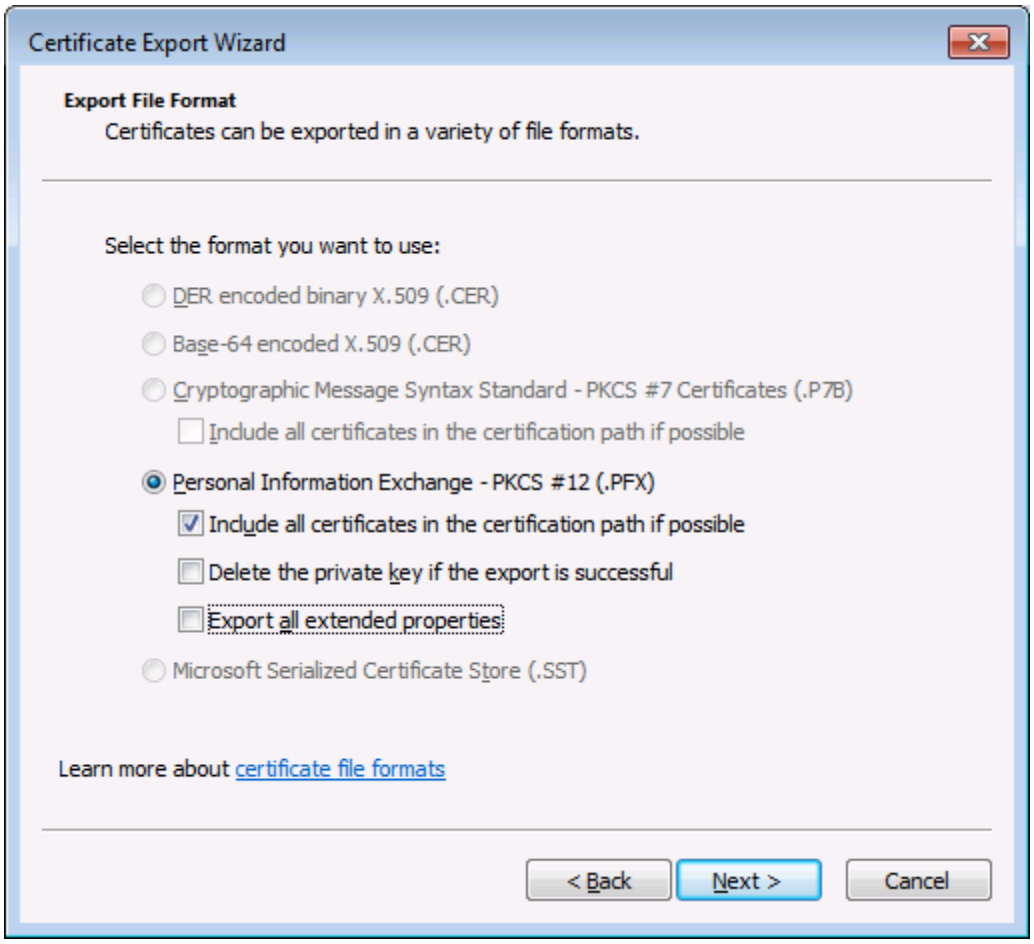
(C) 2015-2021 Altova GmbH

4. Windows から秘密キーと共にエクスポートするオプションを選択して、「次へ」をクリックします。



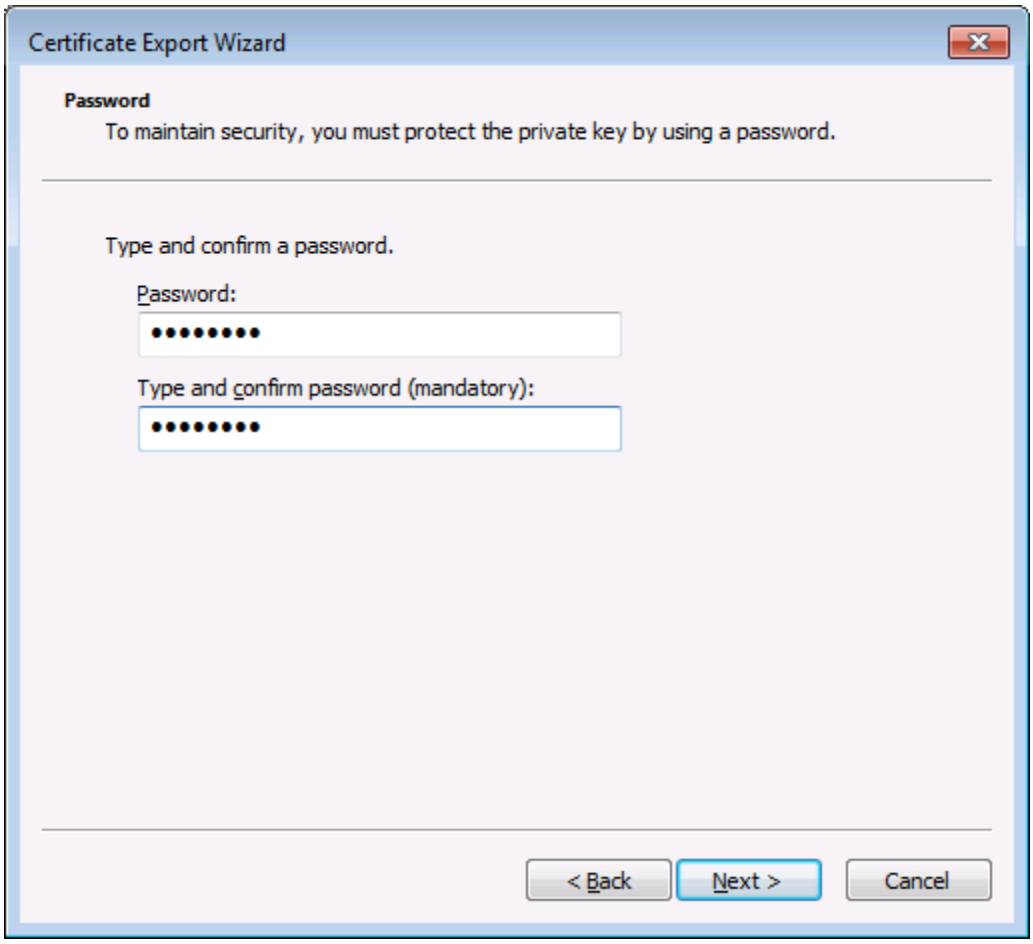
5. 個人情報のエクスチェンジ - *PKCS #12 (.pfx)* ファイル書式を選択し、「次へ」をクリックします。

(C) 2015-2021 Altova GmbH



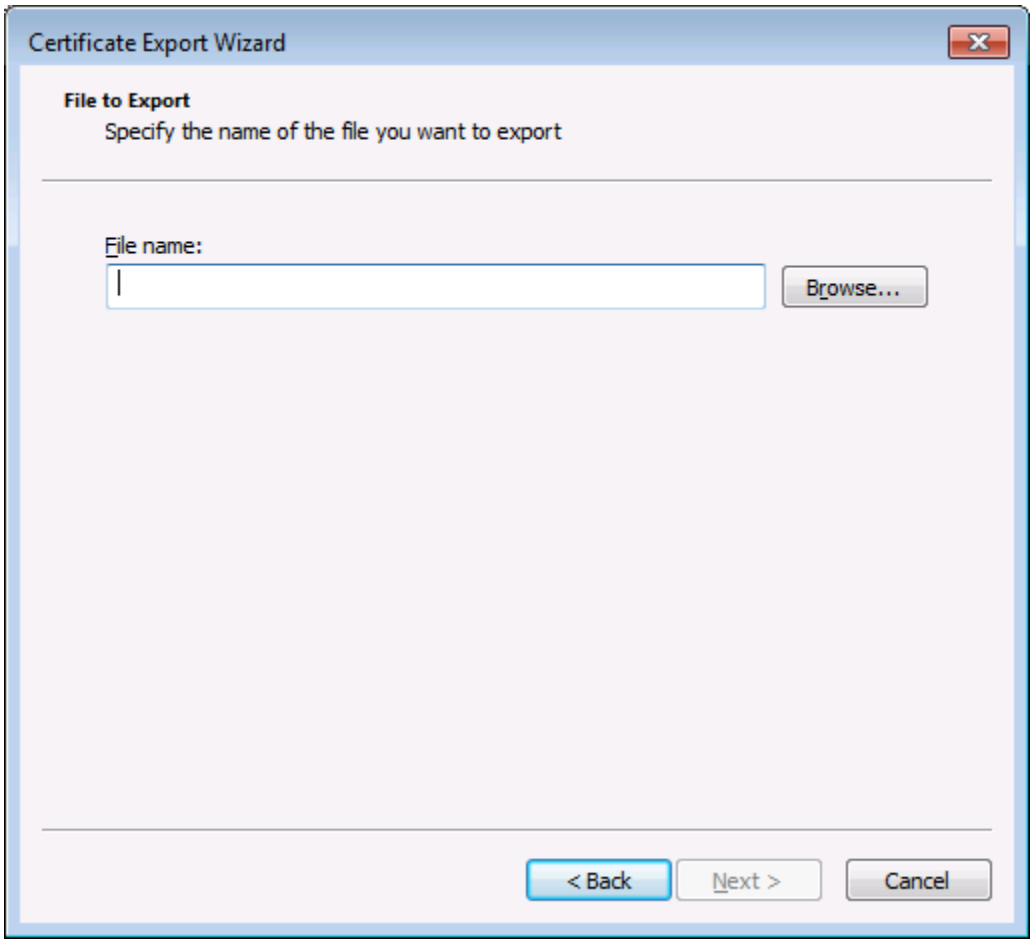
メモ エクスポート後証明書を使用できなくなるため、「**エクスポートに成功すると秘密キーを削除する**」が選択されていないことを確認してください。

6. パスワードを入力し、「**次へ**」をクリックします。このパスワードは、証明書がターゲット オペレーティングシステムにコピーされると必要になります。

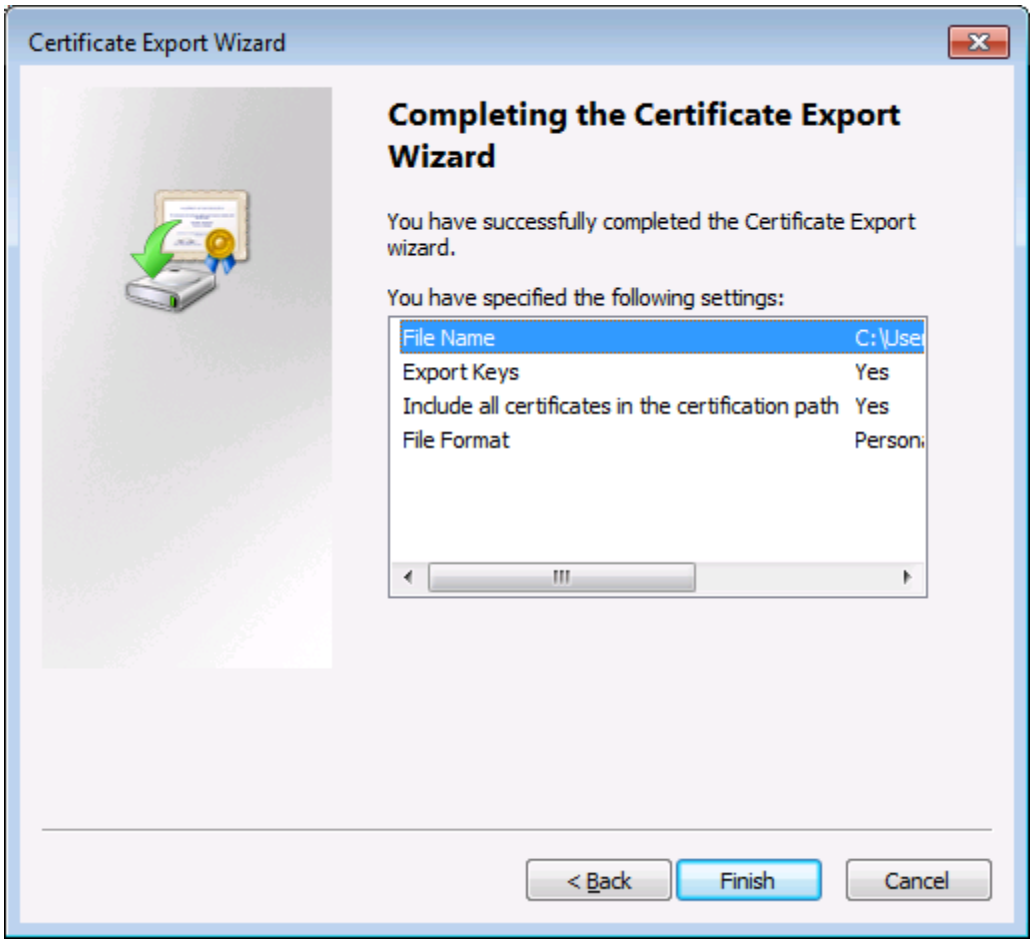


7. ファイルをエクスポートする場所を参照し、「次へ」をクリックします。

(C) 2015-2021 Altova GmbH



8. 「完了」をクリックします。



9.10.6 Linux 上のクライアントの証明書

クライアントの証明書を介して HTTPS を使用してマッピングが Web サービス認証を含む場合、MapForce Server を作動する Linux マシンにマッピングを以下のステップでデプロイします:

1. Web サービスを呼び出すマッピングを開きます。
2. Web サービスのコンポーネントのヘッダーをダブルクリックします。コンポーネントの設定 ダイアログボックスが表示されます。
3. HTTP セキュリティ設定の横の「編集」をクリックします。
4. HTTP セキュリティ設定 ダイアログボックス内で、「**クライアントの証明書**」をクリックして、Windows 上の「**現在のユーザー**」や「**個人用**」ストアから必要とする証明書を選択します(次を参照してください: HTTP セキュリティの設定)。
5. マッピングを保存し、マッピング実行ファイルにコンパイル、または、FlowForce Server にデプロイします(次を参照してください: マッピングをサーバー実行ファイルにコンパイルする と マッピングを FlowForce Server にデプロイする)。
6. Web サービス呼び出しにより必要とされるクライアントの証明書をターゲット オペレーティングシステムに移動します。証明書が秘密キーを確認し、「**拡張キー使用法**」プロパティには「クライアントの認証」が目的として含まれます

Linux にクライアントの証明書を転送する方法:

1. Windows から *Personal Information Exchange - PKCS #12 (.pfx)* ファイル書式内の秘密キーを持つクライアントの証明書をエクスポートします(次を参照してください: Windows 上で証明書をエクスポートする)。

(C) 2015-2021 Altova GmbH

2. Linux マシンに証明書ファイルをコピーします。
3. コマンドを使用して、.pfx ファイルを .pem に変換します:

```
openssl pkcs12 -in cert.pfx -out "John Doe.pem" -nodes
```

このコマンドは、秘密キーを暗号化することなく .pfx ファイルを解析し、.pem ファイルを出力します。暗号化された秘密キーはパスワードをプロンプトし、サーバー実行ではサポートされていません。

マッピングの実行

クライアントの証明書として .pem ファイルを使用するために MapForce Server に命令するには、マッピングを実行する際に `--certificatespath` パラメーターを設定します。`--certificatespath` パラメーターはディレクトリのパスを定義し現在のマッピングにより必要とされる証明書に保管されます。 例えば、証明書ファイルパスが `/home/John/John Doe.pem` の場合、`--certificatespath` は `/home/John` に設定される必要があります。

デフォルトでは、`--certificatespath` パラメーターが与えられないと、MapForce Server は現在のユーザーの ディレクトリ `$HOME/.config/altova/certificates` 内の証明書を検索します。

> マッピングの実行に成功すると、証明書ファイルが .pem 拡張子を持つことが期待され、ファイル名は証明書のスペースを含む共通名（CN）（例えば、**John Doe.pem**）に一致する必要があります。CN にスラッシュ（ / ）が含まれる場合、アンダースコア（ _ ）文字で置き換えられる必要があります。

FlowForce Server ジョブとしてマッピングを実行する場合、証明書ファイルを `$HOME/.config/altova/certificates` ディレクトリにコピーします。ジョブを実行する際に、FlowForce Server はマッピングにより必要とされる証明書ファイルを検索するためにこのディレクトリを使用します。

> セキュリティ上の理由から、機密な情報が含まれているため、証明書ファイルを他のユーザーから読み取れないことを確認してください。

9.10.7 macOS 上のクライアントの証明書

HTTPS クライアントの証明書を使用する Web サービス認証が含まれる場合、MapForce Server を作動する macOS にこのようなマッピングをデプロイするために次のステップを行ってください:

1. Web サービスを呼び出すマッピングを開きます。
2. Web サービスのコンポーネントのヘッダーをダブルクリックします。コンポーネントの設定 ダイアログボックスが表示されます。
3. HTTP セキュリティ設定の横の「編集」をクリックします。
4. HTTP セキュリティ設定 ダイアログボックス内で、**クライアントの証明書** をクリックして、必要とする証明書を選択します（次を参照してください: HTTP セキュリティの設定）.
5. 証明書名がサーバーのホスト名に一致しない場合、「**証明書とリクエスト間の一致しない名前を許可する**」を選択します。保存して、ターゲット オペレーティングシステムにマッピングをデプロイします。
6. （次を参照してください: MapForce マッピングのデプロイ）.
7. Web サービス呼び出しにより必要とされるクライアントの証明書をターゲット オペレーティングシステムに移動します。証明書が秘密キーを持っていることを確認すると、証明書の「**拡張キー使用法**」プロパティには「クライアントの認証」が目的として含まれます

macOS にクライアントの証明書を移動させる方法:

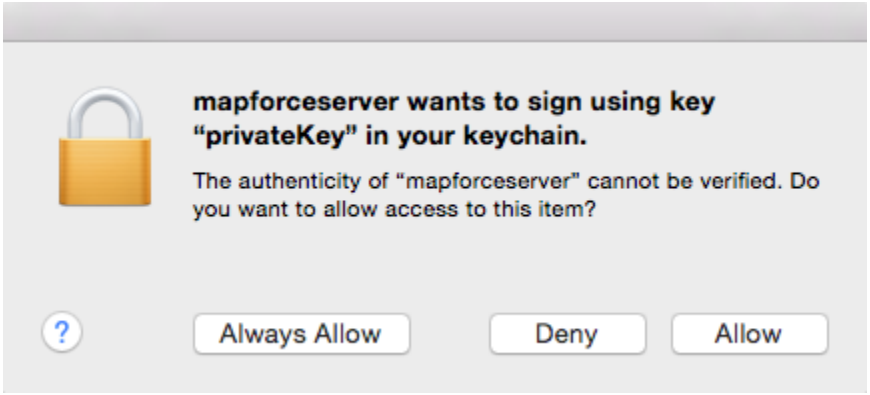
1. Windows から秘密キーを持つクライアントの証明書を *Personal Information Exchange - PKCS #12 (.pfx)* ファイル書式でエクスポートします（次を参照してください: Windows 上で証明書をエクスポートする）.pfx ファイルを macOS にコピーします。

2. まだ行われていない場合、オペレーティングシステムがサーバーの証明書を信頼していることを確認してください（次を参照してください Mac OS 上のサーバーの証明書を信頼する）.
3. 「検索 > アプリケーション > ユーティリティ」からキーチェーンを実行します。
4. 「ファイル」メニューから「アイテムのインポート」をクリックします。
5. ステップ1で Windows からエクスポートされたクライアントの証明書を参照し、目的キーチェーンを選択します。
6. 「開く」をクリックして、暗号化された証明書とパスワードを入力します。

マッピングの実行

MapForce Server `run` コマンドを使用してマッピングを実行する準備が整いました。以下の点に注意してください：

- SSH を使用してマッピングを実行した場合、`security unlock-keychain` コマンドを使用して、キーチェーンを最初に案ロックします。
- macOS のグラフィカルなユーザーインターフェイスを使用してマッピングを実行した場合、MapForce Server サーバーにキーチェーンへのアクセスを許可するようにプロンプトされると、「許可」をクリックします。

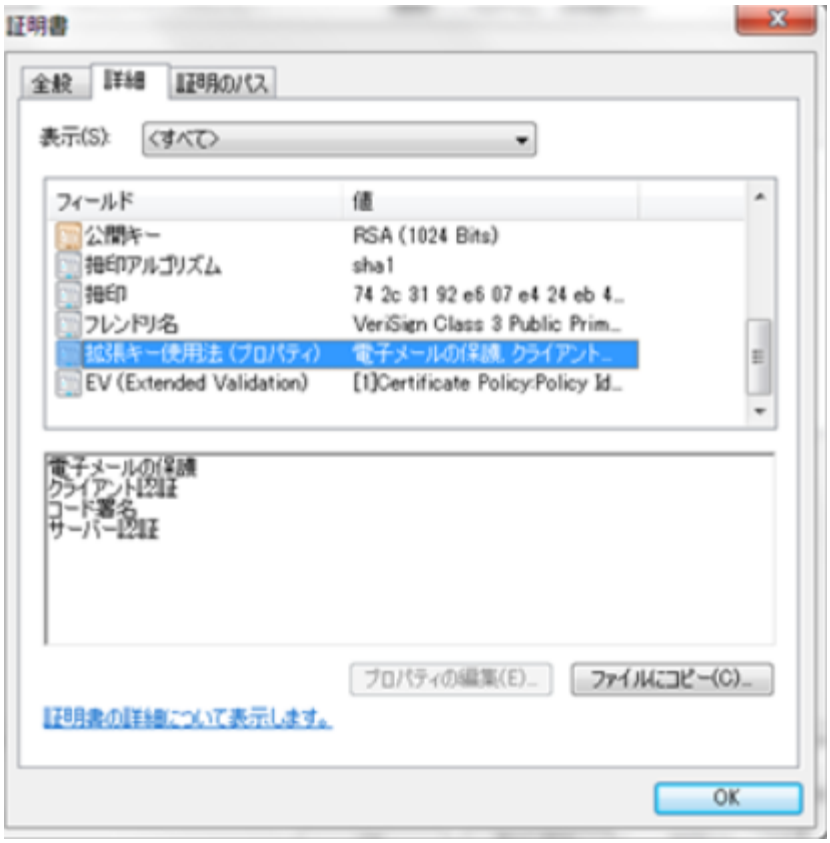


9.10.8 Windows 上のクライアント証明書

クライアントの証明書を必要とするマッピングを Windows 上で実行する場合、マッピングの実行に成功するための条件は以下の通りです:

- クライアントの証明書が「現在のユーザー¥個人用」証明書ストア内に存在する必要があります（マイ ストアとして称されることもあります）。ストアに証明書が存在するには、証明書 インポート ウィザードを使用してインポートされる必要があります。 命令に関しては、次を参照してください： https://technet.microsoft.com/en-us/library/cc754489(v=ws.11).aspx。
- 証明書は秘密キーを持つ必要があります。
- 証明書の「拡張キー使用法」プロパティは「クライアント認証」を目的として含む必要があります。

(C) 2015-2021 Altova GmbH



MapForce の現在のバージョンでは、MapForce により使用することのできるライブラリの制限により、Windows はマッピングが実行されると、必要とする証明書を自動的に証明書ストアから選択します。「**現在のユーザー¥個人用**」 証明書ストアをフィルターすると、マッピングの実行は成功し、サーバーは適切な証明書を検索します。HTTPS 認証（および、証明書の選択操作）は、Windows により管理され、MapForce または MapForce Server によりコントロールされていないことにご注意してください。一部の場合、複数の証明書が「**現在のユーザー¥個人用**」ストア内に存在し、適切ではない証明書がオペレーティングシステムにより選択される場合があり、結果、マッピングの実行に失敗する場合があります。「**現在のユーザー¥個人用**」ストア内で使用することのできる証明書の数を制限することにより、このような状況を回避することができます。

10　マッピングの自動化と MapForce

MapForce を使用してデザインされたマッピングは（Linux とmacOS サーバーを含む）サーバー環境内でサーバーレベルのパフォーマンスを使用して次のAltova 変換エンジンにて実行することができます（これらのエンジンは個別にライセンスが供与される必要があります）:

- *RaptorXML Server*：マッピングの変換言語がXSLT 1.0、XSLT 2.0、XSLT 3.0 またはXQuery の場合、マッピングをこのエンジンを使用して実行することが最適です。RaptorXML Server を使用して自動化するを参照してください。
- *MapForce Server（またはMapForce Server Advanced Edition）*。変換言語がBUILT-IN* の場合このエンジンの使用が最適です。BUILT-IN 言語はMapForce 内のマッピング機能の多数をサポートしますが、MapForce Server（そして特に、MapForce Server Advanced Editionin ）はマッピングを実行するために最高のパフォーマンスを提供します。MapForce Server を使用して自動化するを参照してください

＊BUILT-IN 変換言語は、MapForce Professional またはEnterprise Edition を必要とします。

更に、MapForce には、XSLT、XQuery、C#、C++、およびJava コードをコマンドラインインターフェイスから自動的に生成する機能が搭載されています。これにはMapForce Server 実行を想定したサーバー実行ファイル（.mfx）をコンパイルする機能も含まれています。詳細に関しては、MapForce コマンドラインインターフェイスを参照してください。

(C) 2015-2021 Altova GmbH

10.1 RaptorXML Server を使用して自動化する

RaptorXML Server (今後は、略して RaptorXML) は、Altova 第3世代の超高速な XML と XBRL のためのプロセッサーです。最新の標準とパラレルコンピューティング環境のために最適化されています。クロスプラットフォームに対応可能で、エンジンは今日のマルチコアコンピューティングを有効に利用し、XML と XBRL データの高速処理を提供します。

RaptorXML には、Altova ダウンロードページ (https://www.altova.com/ja/download-trial-server.html) からダウンロードしてインストールすることのできる複数のエディションがあります:

- RaptorXML Server は、XML、XML スキーマ、XSLT、XPath、XQuery、などへのサポートを搭載した高速な XML 処理エンジンです。このエディションは、FlowForce Server インストールパッケージの一部です。
- RaptorXML+XBRL Server は、XBRL の一連の標準を処理および検証する追加機能を使用して、RaptorXML Server の全ての機能をサポートします。

制約事項:

- XML 署名はサポートされていません
- COM インターフェイスを介してグローバルリソースはサポートされていません
- ODBC と ADO データベース接続は W indows のみでサポートされています。他のオペレーティングシステムでは、JDBC を使用する必要があります。

XSLT または in XQuery、でコード生成するには、MapForce は、生成の際に選択する出力フォルダーに保存される **DoTransform.bat** と呼ばれるバッチファイルを作成します。バッチファイルの実行は、RaptorXML Server を呼び出し、サーバー上で XSLT (または XQuery) 変換を行います、

XSLT/XQuery 変換を実行します。

サーバー上の他の出力のために、MapForce マッピングを実行または自動化するには、Altova MapForce Server を使用して自動化する を参照してください。

メモ 内蔵のエンジンを使用して、XSLT と XQuery コードをプレビューすることができます。

10.2 MapForce Server を使用して自動化する

MapForce Server は、Windows、Linux とmacOS オペレーティングシステムのためのエンタープライズサーバーソフトウェアソリューションです。MapForce Server の役割は、(非-Windows プラットフォームを含む) サーバー環境で、サーバーレベルのマッピングを実行することです。MapForce マッピング内でBUILT-IN がターゲット実行言語である箇所でサーバー実行をおこなうことができます (次も参照 変換言語の選択)。MapForce Server は、(コマンドライン、またはAPI から呼び出される) スタンドアロンで、またはFlowForce Server の管理下で作動します。

MapForce Server がスタンドアロン製品として使用されている場合、MapForce マッピングは、コンパイルされMapForce Server が実行されているマシンにコピーされる必要があります。MapForce Server コマンドランコマンド `run` を使用してマッピングは実行されます。MapForce Server API の `run` メソッドを呼び出してマッピングを呼び出すこともできます。詳しい情報に関しては、以下を参照してください: MapForce Server 実行ファイルへマッピングをコンパイルする。

MapForce Server がFlowForce Server 管理下で実行されている場合、マッピングをHTTP (またはSSL/HTTPS) 接続を使用してMapForce から、直接ターゲットマシンにデプロイすることができます。サーバー上では、マッピングは、トリガーされた、または予定されたジョブとして実行されます。または、FlowForce Server 管理インターフェイスから定義されたサービス呼び出しを介して実行されることができます。詳しい情報に関しては、以下を参照してください: FlowForce Server にマッピングをデプロイする。

MapForce Server には以下の2 つのエディションがあります:

- MapForce Server
- MapForce Server Advanced Edition (このエディションはFlowForce Server インストールパッケージの一部です)

MapForce Server Advanced Edition にはMapForce Server と同じ機能が搭載されており、追加して、最適化を行うことのできるマッピングに対して、マッピングを最適化する機能が含まれています。これは、多量のデータをジョイン、またはフィルターするマッピングの場合、または多量のデータをフィルターする場合、実行スピードを向上するためにジョインの最適化が適用されます。MapForce Server とは異なり、MapForce Server Advanced Edition はノード関数が存在する箇所でマッピングを実行することができます。次を参照してください: デフォルトとノード関数。

制約事項:

- XML 署名はサポートされていません
- ADO、ADO.NET、およびODBC データベース接続はWindows でのみサポートされており (その他のオペレーティングシステムに関してはサーバー実行のためにマッピングを準備を参照してください)。

MapForce Server に関する詳細は、付随するドキュメントを参照してください (https://www.altova.com/ja/documentation).

(C) 2015-2021 Altova GmbH

10.3 サーバーでの実行のためにマッピングを準備する

MapForce を使用してデザインされ、プレビューされるマッピングは、現在使用中のマシンと（データベースなどの）オペレーティングシステムの外部にあるリソースを参照することができます。更に、MapForce 内では、全てのマッピングのパスがデフォルトで Windows スタイルの規則に従っています。また、MapForce Server が作動するサーバーでは、マッピングがデザインされたマシンと同じデータベースをサポートしない場合はあります。このため、サーバー環境でマッピングを実行するには、また、ターゲットマシンがソースマシンと異なる場合は特に、若干の準備が必要とされます。

メモ　ソースマシンという用語は、MapForce がインストールされているマシンを指し、ターゲットマシンは、MapForce Server 、または、FlowForce Server がインストールされているマシンを指します。最もシンプルなシナリオは、ソースマシンとターゲットマシンが同じコンピューターの場合を指します。最も複雑なシナリオは、MapForce が Windows マシンで作動し、MapForce Server 、または、FlowForce Server が Linux 、または、macOS マシンで作動する場合を指します。

FlowForce Server にデプロイする前に MapForce 内でマッピングの検証を常に行うことが奨励されます。または、MapForce Server 実行ファイルにコンパイルすることが奨励されます（マッピングの検証を参照してください）。

MapForce Server がスタンドアロンで（FlowForce Server 無しで）作動する場合、必要とされるライセンスは、以下のとおりです:

- ソースマシン上で、MapForce Enterprise 、または、Professional Edition がマッピングをデザインし、サーバー実行ファイル（.mfx）にコンパイルするために必要とされます。 MapForce Server 実行可能ファイルにマッピングをコンパイルするを参照してください。
- ターゲットマシン上で、MapForce Server 、または、MapForce Server Advanced Edition がマッピングを実行するために必要とされます。

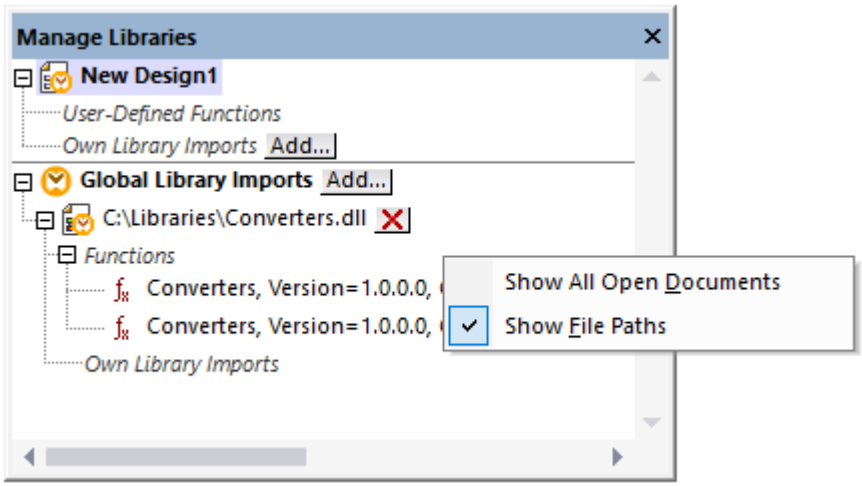
MapForce Server が FlowForce Server 管理下で作動する場合、次の条件が適用されます:

- ソースマシン上で、MapForce Enterprise 、または、Professional Edition がマッピングをデザインして、ターゲットマシンにデプロイするために必要です。, FlowForce Server にマッピングをデプロイするを参照してください。
- MapForce Server と FlowForce Server がターゲットマシンにインストールされている必要があります。MapForce Server の役割は、マッピングの実行であり、FlowForce の役割はマッピングを、スケジュールまたは、オンデマンドの実行、としての実行、エラーの処理、条件付きの処理、電子メールの通知などの機能を活用することのできるジョブとして使用できるようにすることです。
- FlowForce Server は構成されているネットワークアドレスとポートで作動している必要があります。すなわち、「FlowForce Web Server」サービスが開始され、構成されており、HTTP（または、構成されている場合は HTTPS ）クライアントからの接続をファイアウォールにブロックされることなく受け入れる必要があります。「FlowForce Server」サービスも開始され、指定されたアドレスとポートで作動されている必要があります。
- コンテナーの 1 つへのパーミッションが与えられている FlowForce Server ユーザーアカウントを所有している必要があります（デフォルトでは、**/public** コンテナーは認証済みのユーザーであればアクセスすることができます）。

全般的な注意点

- スタンドアロン MapForce Server で作動するターゲットマシン上のマッピングを作動するには、マッピングにより参照される全ての入力ファイルをターゲットマシンにもコピーする必要があります。MapForce Server が FlowForce Server 管理化で作動する場合、ファイルを手動でコピーする必要はありません。この場合、インスタンスとスキーマファイルはターゲットマシンにデプロイされたパッケージ内に含まれます。, see マッピングを FlowForce Server にデプロイする。
- マッピングに特定のデータベースドライバーを必要とするデータベースコンポーネントが含まれる場合、このようなドライバーはターゲットマシンにもインストールされている必要があります。例えば、マッピングが Microsoft Access データベースからデータを読み取る場合、Microsoft Access 、または、Microsoft Access Runtime がターゲットマシンにもインストールされている必要があります（https://www.microsoft.com/en-us/download/details.aspx?id=50040）。
- マッピングを非 Windows プラットフォームにデプロイする場合、ADO、ADO.NET と ODBC データベース接続は自動的に JDBC に変更されます。ネイティブ SQLite とネイティブ PostgreSQL 接続は保存され、追加の構成を必要としません。「データベース接続」も参照してください。
- （例えば、 .dll 、または、.class ファイルなどの）カスタム関数呼び出しをマッピングが含む場合、このような依存は、ランタイム前に

既知ではないため、マッピングと共にデプロイされません。この場合、これらを手動でターゲットマシンにコピーすることができます。サーバー上の.dll または.class ファイルのパスはMapForce 内の「ライブラリの管理」ウインドウと同じである必要があります。例:



- マッピングの一部は、ワイルドカードパスを使用して複数の入力ファイルを読み取ります（複数の入力または出力ファイルを動的に処理するを参照してください）。この場合、入力ファイル名はランタイム前に既知ではないため、デプロイされません。マッピングの実行に成功するには、入力ファイルがターゲットマシン上に存在する必要があります。
- マッピングの出力パスにディレクトリが含まれている場合、これらのディレクトリはターゲットマシン上に存在している必要があります。それ以外の場合、マッピングを実行するとエラーが生成されます。この振る舞いは、「一時的なファイルに出力を生成する」オプションが有効化されている場合既存ではないディレクトリが自動的に生成されるMapForce とは異なります（MapForce オプションの変更を参照してください）。
- マッピングがクライアント証明書を使用するHTTPS 認証を必要とするW eb サービスを呼び出す場合、証明書をターゲットマシンに手動で移動する必要があります。次も参照してください:デジタル署名の管理。
- Microsoft Access とSQLite などのファイルをベースにしたデータベースにマッピングが接続される場合、データベースファイルは手動でターゲットマシンに移動、または、参照することのできる共有されるディレクトリに保存される必要があります。次を参照:「ファイルをベースにしたデータベース」。

パスをポータブルにする方法

サーバー上でマッピングを実行する場合、マッピングが適用することのできる変換パスを使用していること、および、サポートされるデータベース接続を使用していることを確認してください。

パスを非W indows オペレーティングシステムへ移動可能にするためには、MapForce 内でマッピングをデザインする際に絶対パスの代わりに相対パスを使用してください。

1. 希望するマッピングデザインファイル(.mfd) をW indows 上のMapForce で開きます。
2. 「ファイル」メニューで**マッピングの設定** を選択し、チェックされている場合「**生成されたコード内でパスを絶対パスにする**」チェックボックスをクリアします。
3. 各マッピングコンポーネントのために**プロパティ** ダイアログボックスを開き、（例えば、コンポーネントのタイトルバーをダブルクリックして）全てのファイルパスを絶対パスから相対パスに変更します。「**MFD ファイルに対して全てのパスを相対的に保存する**」チェックボックスを選択します。便宜上、マッピングと同様全ての入力ファイルとスキーマを同じファイルにコピーし、ファイル名で参照することができます。

マッピングのデザイン時に相対および絶対パスを処理するには、相対、および、絶対パスの使用を参照してください。

更に、MapForce Server とFlowForce Server は、すべての相対パスが解決される、いわゆる「作業ディレクトリ」をサポートします。多種の実行環境内のパスを参照してください。作業ディレクトリは以下のとおりマッピングのランタイムに指定することができます:

- FlowForce Server では、ジョブの「作業ディレクトリ」パラメーターを編集します。

(C) 2015-2021 Altova GmbH

- MapForce Server API では、COM と.NET API の`WorkingDirectory` プロパティにより、または、Java API の `setWorkingDirectory` メソッドにより行うことができます。
- MapForce Server コマンドラインでは、作業ディレクトリはコマンドシェルの現在のディレクトリです。

データベース接続

ADO、ADO.NET、およびODBC 接続はLinux とmacOS マシンではサポートされていないことにご注意して下さい。このため、ターゲットマシンがLinux またはmacOS の場合、このような接続は、マッピングがFlowForce にデプロイされると、または、MapForce Server 実行ファイルにマッピングをコンパイルすると、JDBC に変換されます。この場合、マッピングをデプロイ、または、サーバー実行ファイルにマッピングをコンパイルする前に以下のオプションを使用することができます:

- MapForce 内では、データベースへのJDBC 接続を作成します（JDBC 接続のセットアップを参照してください）
- MapForce 内では、データベースコンポーネントの「JDBC 固有の設定」セクション内のJDBC データベース接続の詳細に入力してください。（データベースコンポーネント設定を参照してください）。

マッピングがPostgreSQL またはSQLite データベースへのネイティブな接続を使用する場合、ネイティブな接続が保存され、JDBC 変換は発生しません。多数の実行環境内のデータベースマッピング を参照してください。マッピングがMicrosoft Access とSQLite などのファイルベースのデータベースに接続される場合、追加構成が必要とされます。下記の「ファイルベースのデータベース」を参照してください。

JDBC 接続を使用してマッピングを作動することは、Java Runtime Environment またはJava Development Kit がサーバーマシンにインストールされていることを必要とします。Oracle JDK またはOracle OpenJDK などのオープンソースビルドであることができます。

- `JAVA_HOME` 環境変数はJDK インストールディレクトリを指している必要があります。
- Windows 上では、Windows レジストリ内で検出されたJava Virtual Machine パスは`JAVA_HOME` 変数よりも高い優先順位を有します。
- JDK プラットフォーム（64-ビット、32-ビット）はMapForce Server を同じである必要があります。それ以外の場合は、理由があるエラーが挙げられる可能性があります:「JVM へのアクセスが不可能です。」

Linux またはmacOS 上でのJDBC 接続のセットアップ:

1. データベースベンダーにより提供されるJDBC ドライバーをダウンロードして、オペレーティングシステムにインストールします。オペレーティングシステムが32-ビットで作動する場合は32-ビット版バージョンを、オペレーティングシステムが64-ビットで作動する場合は64-ビット版バージョンを選択してください。
2. 環境変数をJDBC ドライバーがインストールされている場所に設定してください。通常、CLASSPATH 変数と、必要があればその他複数を設定します。どの特定の環境変数が構成されるかを検出するには、JDBC ドライバーに付随するドキュメントを確認してください。

メモ　macOS では、システムはJDBC ライブラリが**/Library/Java/Extensions** ディレクトリにあることを必要とします。ですから、JDBC ドライバーをこの場所でアンパックすることが奨励されます。それ以外の場合、システムを構成し、JDBC ライブラリがJDBC ドライバーをインストールしたパスを検索するようにする必要があります。

macOS 上でのOracle Instant Client 接続

macOS 上で、**Oracle Database Instant Client** を介してOracle データベースに接続する場合これらの命令を適用することができます。必要条件:

- Java 8.0 または、以降がインストールされている必要があります。Mac マシンがJava 8 以前のJava バージョンで作動作動している場合、「JDBC Thin for All Platforms」ライブラリを介して接続し、以下の命令を無視することも可能です。
- Oracle Instant Client がインストールされている必要があります。Oracle Instant Client をOracle の公式ダウンロードページからダウンロードすることができます。Oracle ダウンロードページでは複数のInstant Client パッケージがあることにご注意してください。Oracle Call Interface（OCI）サポートのあるパッケージが選択されていることを確認してください（例えばInstant

Client Basic)。また、オペレーティングシステムが32-ビットで作動する場合は32-ビット版バージョンを、オペレーティングシステムが64-ビットで作動する場合は64-ビット版バージョンを選択してください。

ダウンロードした後、Oracle Instant Client をアンパックし、インストーラと共にパックされているプロパティリストファイル(.plist) を編集すると、以下の環境変数が対応するドライバーパスをポイントします。例えば

変数	サンプル値
CLASSPATH	/opt/oracle/instantclient_11_2/ojdbc6.jar:/opt/oracle/instantclient_11_2/ojdbc5.jar
TNS_ADMIN	/opt/oracle/NETWORK_ADMIN
ORACLE_HOME	/opt/oracle/instantclient_11_2
DYLD_LIBRARY_PATH	/opt/oracle/instantclient_11_2
パス	$PATH:/opt/oracle/instantclient_11_2

メモ 使用中のオペレーティングシステムで Oracle Instant Client ファイルがインストールされている場所で上記のサンプル値を編集してください。

ファイルをベースにしたデータベース

Microsoft Access とSQLite などのファイルをベースにしたデータベースは、FlowForce Server にデプロイされるパッケージ内に含まれません、または、コンパイルされた MapForce Server 実行ファイルにも含まれません。このため、ソースとターゲットマシンが同じでは内場合、次のステップに従ってください:

1. MapForce 内では、マッピングを右クリックして、「生成されたコード内でパスを絶対パスにする」のチェックボックスのチェックを解除します (マッピング設定の変更を参照してください)。
2. マッピング上のデータベースコンポーネントを右クリックし、相対パスを使用してデータベースファイルへの接続を追加します。ファイルをベースにしたデータベースへのパスを設定を参照してください。マッピングデザイン(.mfd ファイル) をデータベースファイルと同じディレクトリに保存し、ファイル名別(相対パスを使用して)にマッピングから参照することによりパスに関連した問題を回避することができます。
3. データベースファイルをターゲットマシン内のディレクトリにコピーします ("作業ディレクトリ"と呼ぶことにします)。以下に表示されるようにマッピングを実行する際必要とされるため、このディレクトを忘れないでください。

このようなマッピングをサーバー上で実行するには以下を行います:

- FlowForce Server コントロール下で作動する MapForce Server によりマッピングが実行される場合、FlowForce Server ジョブが前に作成された作業ディレクトリをポイントするように構成します。データベースファイルは作業ディレクトリ内に存在する必要があります。例えば、「W eb サービスとしてジョブを公開する」を参照してください:(http://manual.altova.com/FlowForceServer/FlowForceServerAdvanced/fs_example_web_service.htm)。
- コマンドラインでスタンドアロン MapForce Server によりマッピングが実行される場合、MapForce Server 上で `run` コマンドを呼び出す前に、現在のディレクトリを作業ディレクトリに変更します(例えば、`cd path\to\working\directory`) 。
- MapForce Server API によりマッピングが実行される場合、作業ディレクトリをプログラミング的にマッピングを実行する前に設定します。これを効果的に行うために、COM と.NET API 内の MapForce Server オブジェクトのためにプロパティ `WorkingDirectory` を使用することができます。Java API 内では、メソッド `setWorkingDirectory` を使用することができます。

ローカルネットワーク上のW indows マシン上にソースとターゲットマシンが存在する場合、共有されるディレクトリからデータベースファイルを読み取るためにマッピングを以下のように構成することができます:

1. ソースとターゲットマシンによりアクセスすることのできる共有されるディレクトリにデータベースファイルを保管します。
2. マッピング上のデータベースコンポーネントを右クリックし、絶対パスをしようしてデータベースファイルに接続を追加します (ファイルをベ

ースにしたデータベースへのパスを設定）。

グローバルリソース

マッピングが直接のパスまたはデータベース接続の代わりにグローバルリソースへの参照を含む場合、サーバー側でもグローバルリソースを使用することができます。MapForce Server 実行可能ファイル（.mfx）にマッピングをコンパイルする場合、グローバルリソースへの参照は変更されないので、マッピングランタイムにサーバー側に提供することができます。FlowForce Server にマッピングをデプロイする場合、リソースをサーバー上で使用するかを任意で選択することができます。

マッピング（または FlowForce Server の場合マッピング関数）の実行に成功するために、グローバルリソースとして提供する実際のファイル、フォルダー、またはデータベース接続の詳細はサーバー環境で互換性を持つ必要があります。例えば、マッピングが Linux サーバーで作動する場合、パスのために Linux 変換を使用する必要があります。同様に、データベース接続として定義されているグローバルリソースはサーバーマシンで利用が可能である必要があります。

詳細に関しては MapForce Server 内のグローバルリソース と FlowForce Server 内のグローバルリソース。を参照してください。

XBRL タクソノミパッケージ

マッピングが FlowForce Server への XBRL タクソノミパッケージを参照する場合 MapForce はマッピングから全ての外部レファレンスを集め、現在の構成と現在インストールされているタクソノミパッケージを使用して解決します。タクソノミパッケージを指す解決された外部レファレンスが存在する場合、タクソノミパッケージはマッピングと共にデプロイされます。FlowForce Server は、そのパッケージを、デプロイ時と同様、マッピングを実行するために使用します。FlowForce Server により使用されるタクソノミパッケージを更新するには、MapForce 内で変更し、マッピングをもう一度デプロイします。

MapForce Server のルートカタログは、ターゲットマシン内でタクソノミの解決方法に影響を与えます。ルートカタログは、MapForce Server インストールディレクトリに対して相対的に以下のパスで見つけることができます：**etc/RootCatalog.xml**.

MapForce Server のルートカタログがこのようなカタログを含んでいない場合、または、同 URL じプレフィックスのために定義されているパッケージを含まない場合、マッピングと共にデプロイされるタクソノミパッケージが使用されます。MapForce Server のルートカタログはデプロイされたタクソノミより先優先順位が高いです。

MapForce Server がスタンドアロンで（FlowForce Server 無しで）作動する場合、次に用にマッピングにより使用されるルートカタログを指定することができます:

- コマンドラインでは、オプション `-catalog` を `run` コマンドに追加することが可能です。
- MapForce Server API で、メソッド `SetOption` を呼び出し、文字列「`catalog`」を最初の引数として、ルートカタログを2番目の引数として与えます。

マッピングがテーブルリンクベースを使用する XBRL コンポーネントである場合、タクソノミパッケージ、または、タクソノミパッケージ構成ファイルがランタイムでマッピングに提供される必要があります:

- MapForce Server コマンドラインにて、オプション `--taxonomy-package`、または、`--taxonomy-packages-config-file` を `run` コマンドに追加します。
- MapForce Server API 内で、メソッド `SetOption` を呼び出します。最初の引数は、「`taxonomy-package`」、または、「`taxonomy-packages-config-file`」である必要があります。2番目の引数は、タクソノミパッケージ（、または、タクソノミパッケージ構成）ファイルへの実際のパスである必要がありますファイル。

10.4 MapForce Server 実行可能ファイルにマッピングをコンパイルする

MapForce 内で作成されるマッピングのターゲット言語がBUILT-IN に設定されている場合 MapForce のみでなく、MapForce Server もがマッピングを実行することができます(次を参照してください: MapForce Server について)。MapForce Server を使用して2通りの実行を行うことができます:

- MapForce Server がスタンドアロン モードで作動している場合、(つまり、FlowForce Server がインストールされていない状態で)、マッピングは、下に表示されるとおり、サーバー実行ファイル(.mfx) にコンパイルされる必要があります。その後、.mfx ファイルをコマンドラインでコマンド `run` を使用して実行します。`run` メソッドを MapForce Server API から呼び出して、マッピングを実行することもできます。詳細に関しては、次を参照してください: MapForce Server ドキュメント ((https://www.altova.com/ja/documentation)。
- または、MapForce Server がFlowForce Server 管理下で作動している場合、マッピングは、MapForce Server と FlowForce Server の双方が作動しているマシンにデプロイすることができます。このシナリオの詳細に関しては、次を参照してください: マッピングを FlowForce Server にデプロイする。

必要条件

サーバーでの実行のためにマッピングを準備するを参照してください。

MapForce Server Execution (.mfx) ファイルにマッピングをコンパイルする

1. MapForce 内でマッピングを開きます(例えば、**myMapping.mfd**)。
2. 「ファイル」メニューから、「MapForce Server 実行ファイルにコンパイル」をクリックします。
3. .mfx ファイルを保存するフォルダーを選択し、必要であれば名前を変更します。
4. 「保存」をクリックします。MapForce Server 実行ファイル**myMapping.mfx** は、選択されたフォルダー内に生成されます。

MapForce Server 実行可能ファイル(.mfx) ファイルにマッピングをコマンドラインを使用してコンパイルする

- コマンドラインで MapForce を実行し、マッピングファイルと `/COMPILE` コマンドラインオプションを指定します。

例えば、次のコマンドは、マッピング<マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥SimpleTotal.mfd を MapForce Server 実行ファイルにコンパイルし、ターゲット出力ディレクトリ**C:\Users\altova\Desktop** に作成されます。

```
"C:\Program Files (x86)\Altova\MapForce2021\MapForce.exe" "C:
\Users\altova\Documents\Altova\MapForce2021\MapForceExamples" /COMPILE "C:
\Users\altova\Desktop"
```

以下も参照してください: MapForce コマンドラインインターフェイス。

.mfx ファイルに含まれている内容

.mfx ファイルには次のデータが含まれています:

- 他のマッピングからインポートされたユーザー定義関数(UDF) を全て含むマッピングアルゴリズム。
- コンポーネントから参照されている入力と出力ファイル名。マッピング設定に従いパスは絶対的または相対的です。多種の実行環境内のパスを参照してください。
- マッピングがXML コンポーネントを含む場合、マッピングの実行に必要な XML スキーマに関する情報はマッピングアルゴリズムにエンコードされる必要があります。
- マッピングがデータベース接続を含む場合にはデータベース接続の詳細。パスワードは暗号化されています。

(C) 2015-2021 Altova GmbH

- 各マッピングへの適用に従い以下から派生した情報[1]：
  - EDI とFlexText 構成
  - JSON スキーマ
  - .proto ファイル
- 「MapForce Server 実行ファイルとマッピングデプロイに含む」チェックボックスを選択した場合、マッピングに資格情報が含まれると、機密なフィールドが暗号化された書式で含まれます。それ以外の場合、機密なフィールドはスキップされます。詳細に関しては、資格情報を参照してください。

脚注:

1. EDI またはFlexText 構成、JSON スキーマ、または.proto ファイルから派生した情報はコンパイルされ処理された書式で .mfx ファイル内に埋め込まれます。コンパイルされた.mfx ファイルから元のソースファイルを取得することは不可能です。コンパイル時の状態でデータはコンパイルされます。以降変更されると.mfx ファイルは再コンパイルされる必要があります。

マッピングにより使用された入力インスタンスファイル（XML、CSV、テキスト、EDI、Excel、JSON、XBRL）コンパイルされた.mfx ファイルには含まれません。Access またはSQLite などのファイルベースのデータベースに関しても同様に適用されます。マッピングがXBRL テーブルサポートに関連するXBRL タクソノミを参照する場合、タクソノミ情報は.mfx ファイルにコンパイルされません。詳細に関しては サーバー実行のためにマッピングを準備するを参照してください。

特定のMapForce Server バージョンのためにマッピングをコンパイルする

MapForce Server がMapForce より古いバージョンの場合、前者は、新機能が搭載されていない可能性があるため、新しいバージョンのMapForce で作成された.mfx ファイルを実行できない可能性があります。このような場合には、.mfx ファイルを特定のMapForce Server バージョンのために、以下の方法でコンパイルすることができます:

1. 「ツール」メニューから、「オプション」をクリックして、「生成」をクリックします。
2. 「サーバー実行ファイル」の下の「MapForce Server バージョンの生成」横から、必要とするMapForce Server バージョンをドロップダウンリストから選択します。

新しいMapForce Server バージョンを取得すると、このオプションを必要に応じて変更してください。MapForce Server の特定のバージョンのためにコンパイルする特定の理由が存在しない場合は、"最新" オプションを選択します（これはデフォルトのオプションです）。このオプションが選択されていると、.mfx ファイルは、MapForce Server の最新のバージョンのためにコンパイルされるため、前のバージョンでは使用することのできない最新の機能を活用することができます。

ターゲット MapForce Server バージョンをコマンドラインで指定するには、/MFXVERSION スイッチを利用して /COMPILE コマンドを実行します。例:

```
"C:\Program Files (x86)\Altova\MapForce2021\MapForce.exe" /COMPILE /MFXVERSION:2021
```

以下も参照してください: MapForce コマンドラインインターフェイス。

他のオプション

MapForce Server 実行可能ファイルのコンパイルは、次のオプションによっても影響されます:

すべてのADO とODBC データベースの接続を JDBC に変換する	オプションが有効化されていると、ADO とODBC データベースの接続は、JDBC ドライバーとデータベース コンポーネント 設定ダイアログボックス内で定義されたデータベースURLを使用して、JDBC に変換されます（次を参照してください： データベースコンポーネントの設定）。 JDBC 接続はターゲット マシンがLinux または macOS サーバーである場合明示的に使用されます。
（MapForce Server によりサポートされていない）デジタル署名を無視する	このオプションはMapForce Enterprise にのみ適用することができます。また、デフォルトで有効化されています。マッピングがXML デジタル署名を使用する場合、MapForce Server が XML デジタル署名をサポートしないため、デジタル署名に関する情報をスキップします。

これらのオプションをビュー、または変更することができます:

- 「ツール」 メニューから、「オプション」をクリックして、「生成」をクリックします。

これらのオプションはコマンドラインインターフェイスからも使用することができます。以下も参照してください： MapForce コマンドラインインターフェイス。

(C) 2015-2021 Altova GmbH

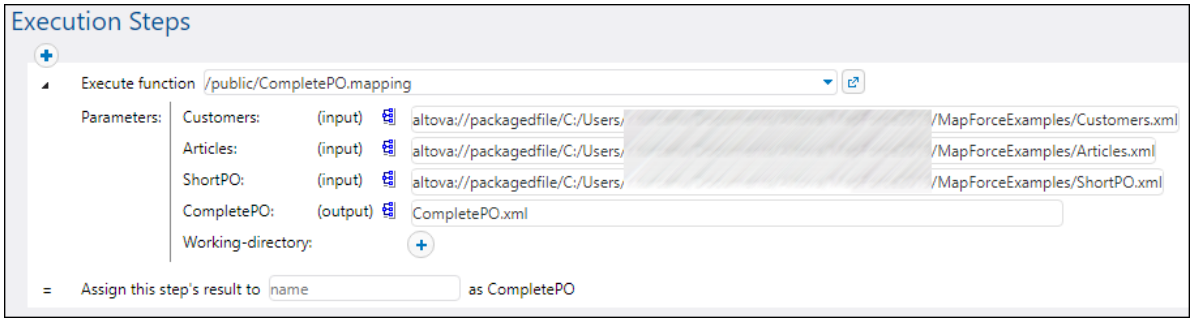
10.5 マッピングを FlowForce Server にデプロイする

FlowForce Server へのマッピングのデプロイは MapForce が特定のマッピングにより使用される*ソース*を整理し HTTP（または構成されている場合 HTTPS ）を介して FlowForce Server が作動するマシンにパスすることを意味します。メモ「*ソースマシン*」という用語は、MapForce がインストールされているコンピューターを指し、「*ターゲットマシン*」という用語は、FlowForce Server がインストールされているコンピューターを指します。マッピングがデプロイされると、完全機能を搭載した FlowForce Server ジョブを作成することができ、（例えば、ジョブのために*カスタムトリガー*を定義する、ジョブをサービスとして公開するなど）ジョブ固有の機能全てを活用することができます。

メモ　「*ソースマシン*」という用語は、MapForce がインストールされているコンピューターを指しており、「*ターゲットマシン*」は FlowForce Server がインストールされているマシンを指しています。最も*シンプル*なシナリオは、*ソースマシン*と*ターゲットマシン*が同じコンピューターの場合を指します。最も複雑なシナリオは、MapForce が W indows *マシン*で作動し、FlowForce Server が Linux 、または、macOS *マシン*で作動する場合を指します。

FlowForce にデプロイされる*パッケージ*は以下の通りです:

- マッピング自身コンポーネントは入力引数となり、ターゲットコンポーネントはこの関数の出力引数となります。全ての*ソース*コンポーネントは入力引数になり、全てのターゲットコンポーネントはこの関数の出力関数になります。



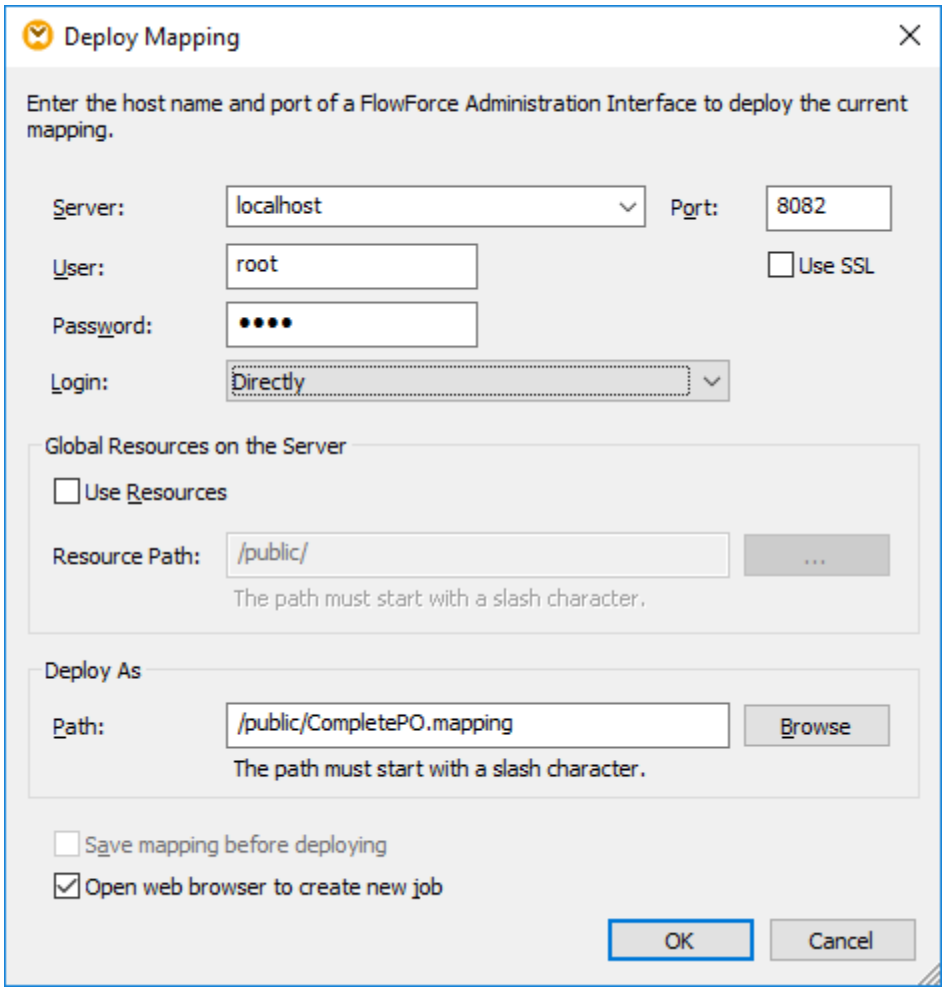
- マッピングにより使用される入力*インスタンスファイル*（XML、CSV、テキスト、EDI、Excel、JSON、XBRL）。
- デプロイ後に EDI または FlexText 構成が変更されると、マッピングを FlowForce Server に再度デプロイする必要があります。マッピングが XBRL *タクソノミ*、または XBRL *タクソノミパッケージ*を使用する場合、これらは、デプロイ時に*ソースマシン*上で存在する状態でデプロイされます。
- マッピングが XBRL *タクソノミ*または XBRL *タクソノミパッケージ*を使用する場合、デプロイの際*ソースマシン*上で存在する状態でこれらがデプロイされます。

必要条件

サーバー実行のためにマッピングを準備するを参照してください。

FlowForce Server にマッピングをデプロイする

1. MapForce を作動し（**Built-in** ツールバーボタンをクリック、または **出力 | Built-in 実行エンジン** メニューコマンドを選択して）変換言語が BUILT-IN に設定されていることを確認してください。
2. 「**ファイル**」メニューから「**FlowForce Server にデプロイ**」を選択します。マッピングのデプロイダイアログボックスが開かれます。



3. （下で説明される通り）デプロイ設定を入力し [OK]を押します。「**新しいジョブ作成のためにウェブブラウザを開く**」チェックボックスが選択されている場合、FlowForce Server 管理インターフェイスが開かれ、すぐにFlowForce Server ジョブの作成を開始することができます。

以下のテーブルは、マッピングのデプロイダイアログボックスで使用することのできるマッピングデプロイ設定をリストしています。

設定	説明
サーバー、ポート、SSL の使用	FlowForce Server のサーバーホスト名（またはI.P. アドレス）とポートを入力します。これはFlowForce Server がデフォルトのポートと同じマシンで実行されている場合、**localhost** と**8082** である可能性があります。疑問がある場合は、FlowForce Server Web 管理インターフェイスにログインして、Web ブラウザーのアドレスバーに表示されるI.P. アドレスとポートを確認してください。 接続エラーが生じた場合、FlowForce Server が実行されているマシンが入力接続を許可するように構成されていることを確認してください。 SSL 暗号化済み接続を介してマッピングをデプロイする場合 **SSL の使用** チェックボックスを選択しますこれは、FlowForce Server が既にSSL 接続を受け入れるように構成されているものの想定しています。詳細に関しては、FlowForce Server ドキュメント (https://www.altova.com/ja/documentation) を参照してください。

(C) 2015-2021 Altova GmbH

設定	説明
ユーザーとパスワード	ユーザー名とパスワードは、ログインドロップダウンリストの値（次のオプション参照）により入力されます。ログインドロップダウンリストが **<Default>** または **Directly** に設定されている場合、FlowForce Server ユーザー名とパスワードを入力します。それ以外の場合ドメインユーザー名とパスワードを入力し、ドメイン名をログインドロップダウンリストから選択します。
ログイン	FlowForce Server 内で Directory Service が有効化されていると このドロップダウンリストからドメイン名を選択し、ユーザーとパスワードフィールドにドメイン資格情報を入力します（前のオプションを参照してください）。
リソースの使用、リソースパス	マッピング関数がサーバーにデプロイ後にリソースを使用する場合**リソースの使用** チェックボックスを選択しますチェックボックスを選択すると、「**リソースパス**」テキストボックス上の対応するリソースのパスを入力する必要があります。リソースを選択し、「**省略記号**」ボタンをクリックします。 サーバー上に選択するリソースが存在しない場合、「**グローバルリソースのデプロイ**」をクリックして、必要とされるグローバルリソースをサーバーにデプロイします。詳細に関しては、FlowForce Server にリソースをデプロイする を参照してください。 「**リソースの使用**」チェックボックスを選択しない場合、グローバルリソースは現在選択されている構成をベースに解決されます。サーバー上では、マッピング関数はグローバルリソースを必要とせず、代わりに解決された値を使用します。
パス	「**参照**」をクリックして、FlowForce Server コンテナーの階層内でマッピング関数が保存されるパスを選択します。デフォルトでパスは FlowForce Server の **/public** コンテナーに設定されています。 必要な FlowForce Server パーミッションと特権が与えられている場合、既存のコンテナーとマッピングを削除できることもできます。
デプロイ前にマッピングを保存する	保存されていないマッピングをデプロイする時にこのオプションを使用することができます。デプロイ前にマッピングを保存する場合このチェックボックスを選択します。
新しいジョブを作成するために ブラウザを開く	このチェックボックスを選択した場合、FlowForce Server W eb 管理インターフェイスがデプロイ後開かれ、FlowForce Server ジョブの作成の開始をすることができます。

トラブルシューティング

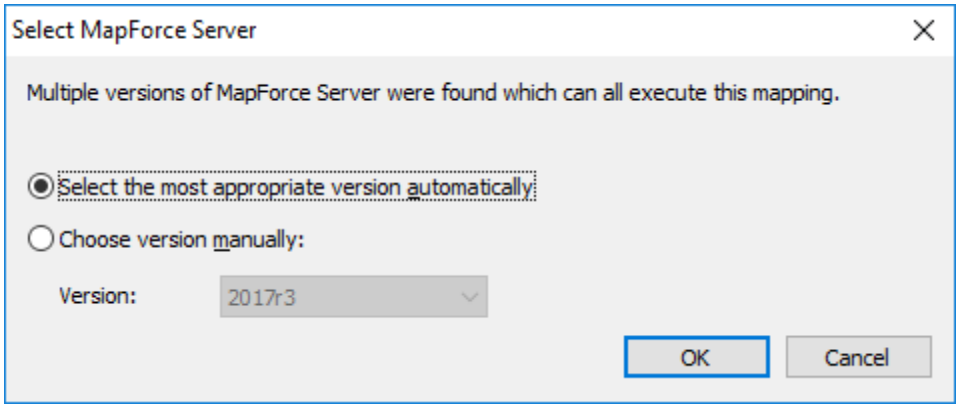
次のテーブルはマッピングをデプロイする際に発生する一般的な問題をリストしています。

問題	ソリューション
マッピングのデプロイは以下のエラーを返します: ファイル上の I／O オペレーションに失敗しました。 I／O エラー 28: 〈server〉 ポート 8082 への接続に失敗しました。タイムアウト システムエラー 10060: 接続されたパーティが一定の時間内に正確にレスポンドしなかった、または、接続されたホストがレスポンドに失敗し確立された接続が失敗したため接続の試みは失敗しました。	ターゲットマシン上で *FlowForce Web Server* サービスが作動しており、指定されているポート（デフォルトで **8082** ）で接続のためにリッスンするように構成されていることを確認してください。またファイアウォールがこのポートを介して受信される接続をブロックしないようにしてください。 デプロイを可能にするために *FlowForce Server* サービスも作動している必要があります。
マッピングのデプロイは以下のエラーを返します:	デプロイされるマッピングの入力ファイルが FlowForce Server により

問題	ソリューション
ファイル上の I/O オペレーションに失敗しました。 I/O エラー 413: ペイロードが大きすぎます	許可されている HTTP リクエストの最大量(大体 100 MB)を超えるとエラーが発生する場合があります。リミットを `max_request_body_size` オプションを(in bytes) **flowforceweb.ini** と **flowforce.ini** ファイル内で設定して増量することができます。詳細に関しては FlowForce Server ドキュメントを参照してください。

サーバーバージョンの選択方法 (Windows のみ)

マッピングをデプロイするサーバーで FlowForce Server 管理下で、複数のバージョンの MapForce Server を作動している場合、(Windows サーバーのみに適用することができます) このマッピングを実行する MapForce Server のバージョンを指定するようにプロンプトされます。



メモ FlowForce Server インストールディレクトリに .tool ファイルが含まれている場合、FlowForce Server の管理下で実行される各 MapForce Server のバージョンのために、ダイアログボックスが表示されます。デフォルトでは、MapForce Server を FlowForce Server の一部としてインストールすると、 MapForce Server .tool ファイルは、このディレクトリに自動的に追加されます。FlowForce 内でファイルが保管されるパスは以下のとおりです: **C:\Program Files\Altova\FlowForceServer2021\tools**。FlowForce Server 管理下での他の MapForce Server バージョンを作動する場合、.tool ファイルを手動で上記のディレクトリにコピーする必要があります。MapForce Server の .tool ファイルは以下に保管されています: **C:\Program Files\Altova\MapForceServer2021**。

(C) 2015-2021 Altova GmbH

10.6 AS2 統合

AS2 (Applicability Statement 2) はインターネット上でファイルを安全に交換することを可能にする仕様です。AS2 はHTTP HTTPS を使用してEDIINT (インターネット上のEDI) とXML ファイルを伝送することができます。

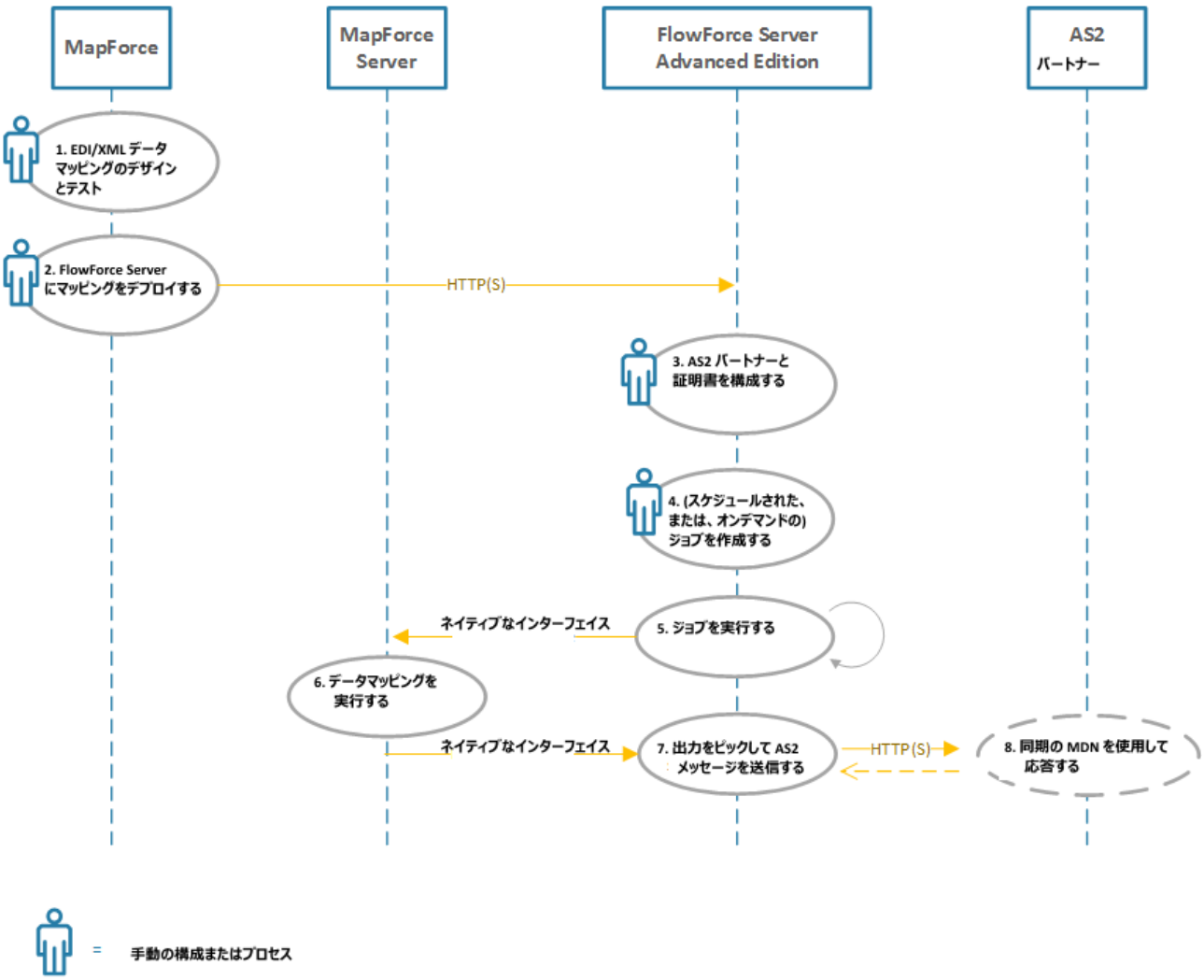
FlowForce Server Advanced Edition は、取引パートナーにAS2 メッセージを送信、または、取引パートナーからAS2 を受信するために必要な機能を提供します。更に、FlowForce Server はビルドインの関数を使用してAS2 データを処理し、ローカルで保管することができます。更に高度な必要性に関しては、既存のソース(例えば、データベース)からAS2 データを準備する場合、または、他の書式に変換する場合、または、W eb サービスに送信する場合、MapForce とMapForce Server をAS2 プロセスに含むことができます。

MapForce とMapForce Server が必要なシナリオが2つ存在します:

1. AS2 パートナーに送信する前に、MapForce によりサポートされる書式(XML、XBRL、Excel、データベース、W eb サービス)でデータをマップまたは生成する場合。
2. AS2 パートナーから受信下データを多種の方法で変換する場合 (例えば、Excel に変換する。異なるXML スキーマに変換する場合。データベースに保管する、W eb サービスに保管するなど)。

AS2 データの生成と作成

パートナーに送信する前に、MapForce を使用してAS2 データを準備、または、生成するシナリオでは、高度なプロセスは以下のようになります:



ハイレベルの AS2 プロセスのダイアグラム

上のダイアグラムでは、MapForce Server とFlowForce Server は同じマシンにインストールされている必要があります（W indows、Linux、または、macOS オペレーティングシステムを使用することができます。を参照してください（W indows マシンであることが前提で）MapForce はMapForce Server とFlowForce Server と同じマシン、または、異なるマシン上でHTTP またはHTTPS を介してFlowForce Server に接続することができます。AS2 パートナーは、HTTP(S) を使用してFlowForce Server が通信する遠隔のサーバーです。

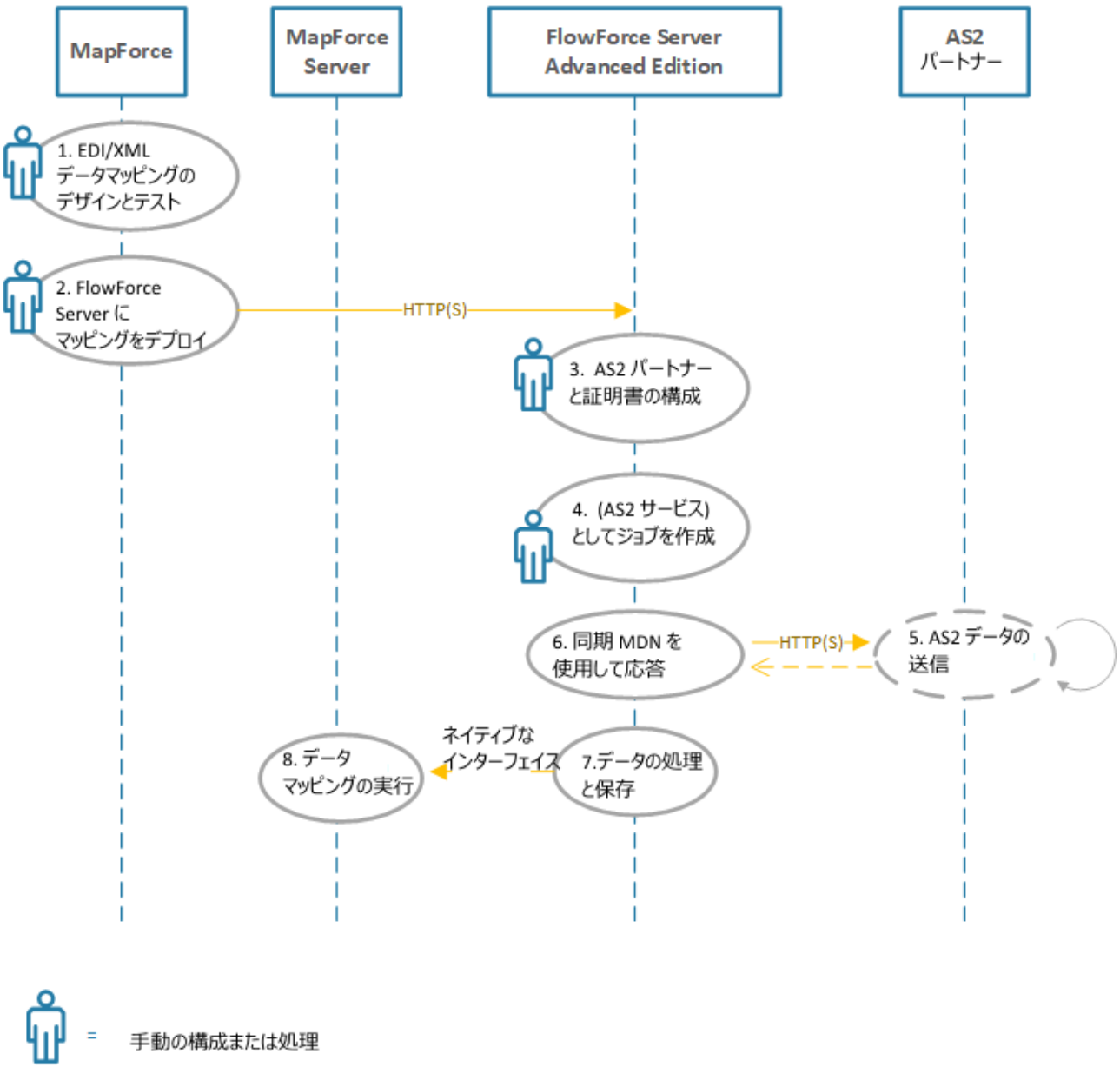
上記のAS2 プロセスは以下のように作動します:

ステップ #	説明
1. EDI/XML データマッピングのデザインとテスト	MapForce を使用すると、（テキスト、CSV、JSON、XML、多種のEDI フレーバー、データベース、W eb サービス）異なるフォーマットのデータを入力として取るデータマッピングの変換をデザインし、（例えば、UN/EDIFACT）目的のフォーマット内に1つまたは複数のファイルとして出力することができます。データマッピング EDI 目的のためにデータマッピングをデザインすることは、他のデータマッピングと違いはありません。MapForce ドキュメントの中に多種のサンプルが含まれています。次を参照してください:EDI チャプター。マッピングのデザイン中、出力タブをクリックして、MapForce 上で直接マッピングを

ステップ #	説明
	検証しプレビューすることができます。マッピングがサーバー環境内での実行に適するように、BUILT-IN 変換言語をデザインしてストする必要があります。変換言語の選択を参照してください。
2.2. マッピングを FlowForce Server にデプロイする	FlowForce Server は W eb インターフェイス内で定義されているオンデマンド、または、計画されたジョブにより多種のジョブを自動化します。MapForce Server が FlowForce Server の管理下で作動することを前提に、FlowForce Server は MapForce でデザインされたマッピングの実行も自動化することができます。MapForce マッピングが必要な出力を生成すると、FlowForce Server にデプロイすることにより実行を自動化することができます。命令に関しては、次を参照してください: マッピングを FlowForce Server にデプロイする。
3.AS2 パートナーと証明書の構成	AS2 パートナーとの通信をセットアップするには、(URI とAS2 名など) AS2 接続の詳細と交換証明書を取得する必要があります。証明書 (とパートナーの詳細) は FlowForce Server にインポートされる必要があります
4.ジョブの作成	FlowForce ジョブはマッピングを実行し、(a) 必要な出力を生成(b) AS2 メッセージを送信するために作成されます (ステップ 7 を参照してください)。これらの2 つのアクションは、同じジョブの実行ステップ、または、2 つの異なるジョブの実行ステップです。
5.ジョブの実行	前のステップで作成された FlowForce Server ジョブは必要に応じて多種の方法で実行されるように構成することができます。例えば、W eb サービスの呼び出しとして実行することができます。または、ファイルシステム上のファイルに変更が加えられると、特定の日時に実行されるようにスケジュールすることができます
6.データマッピングの実行	このステップは MapForce Server により自動的に実行することもできます。(オンデマンド、または、計画されている) データマッピングを実行するようにジョブが構成されている場合、MapForce Server への内部呼び出しが発生します。この結果、MapForce Server はマッピングを実行し、出力を FlowForce Server に返します。
7.出力をピックし、AS2 メッセージを送信する	AS2 メッセージを送信するには、ジョブ (または、ジョブ内の実行ステップ) は FlowForce Server ビルトイン 関数 `/system/as2/send` を呼び出す必要があります。この関数は、ステップ3 で定義されているパートナーオブジェクト、パートナー URI 、および送信するメッセージコンテンツを含む、AS2 メッセージを送信するために必要なパラメーターをとります。ジョブは多種の FlowForce Server AS2 式関数をマッピングの出力を必要とされる書式に変換するために呼び出す必要があります (例えば、ファイルからストリームに変換する場合など)。
8.同期 MDN を使用してのパートナーの応答	AS2 パートナーオブジェクトをステップ 3 で作成すると、オプションでパートナーに、FlowForce Serverにより送信される AS2 メッセージの応答として Message Disposition Notification (MDN) をリクエストすることができます

AS2 データの受信と処理

取引パートナーからの AS2 データを受信する場合、ワークフローを受信するデータを追加して構成することができます。このシナリオの場合、AS2 データを受信し保管するだけではなく、他のフォーマットに変換し、データベースに保存し、他の W eb サービスに送信することができます。例えば、EDI または XML 書式のファイルを AS2 取引パートナーから受信し、FlowForce Server ジョブに再帰して実行されるマッピングに入力として提供することができます。このセクションでは、サンプルの AS2 プロセスが示されています:



AS2 データの受信と処理

上記のAS2 処理は以下のように作動します:

ステップ #	説明
1, 2, 3	これらは前のテーブル内のステップと同じです。唯一の違いは、マッピングが入力として AS2 取引パートナーから受信を期待するファイル(例えば、EDI または XML ファイル)を期待することです。
4.(AS2 サービスとしての)ジョブの作成	これは1 度きりのステップです。このステップでは、AS2 サービスを公開する FlowForce Server ジョブを作成します。AS2 サービスは構成された HTTP(S) アドレスとポートで、AS2 パートナーからのリクエストをリッスンします。

(C) 2015-2021 Altova GmbH

ステップ #	説明
5. AS2 データの送信	このステップでは、取引パートナーは AS2 サービスを AS2 メッセージに送信します。コミュニケーションに成功するには、パートナーの AS2 名と証明書が FlowForce Server 内で既に定義されている必要があります。
6. 同期 MDN を使用しての回答	FlowForce Server は操作の出力(成功、または、エラー)を示す同期 MDN を使用して AS2 パートナーに応答します。
7. データの処理と保存	メッセージが受信されると、FlowForce Server ジョブは、受信されたデータを文字列、または、ファイルに変換し、ディレクトリに保存、または、引数として他のジョブにパスします。正確な処理ロジックは、FlowForce Server ビルトインと式関数を使用して構成することができます。
8. データマッピングの実行	AS2 データを受信する FlowForce Server ジョブは最初のステップで作成されたデータマッピングジョブを任意で呼び出すことができます。マッピングジョブは入力としてパートナーから受信した AS2 データを取り MapForce によりサポートされる方法で処理します: 例えば、他の書式への変換、他の W eb サービスに送信するなど。

FlowForce Server に固有の AS2 構成の詳細に関しては、FlowForce Server Advanced Edition ドキュメントを参照してください(次を参照してください: https://www.altova.com/ja/documentation)。

10.7 MapForce コマンドラインインターフェイス

コマンドラインでのMapForce コマンドの一般的な構文:

```
MapForce.exe <filename> [/{target} [[<outputdir>] [/options]]]
```

レジェンド

コマンドライン 構文を示すために次の表記が使用されます:

表記	説明
かっこ無しのテキスト	アイテムは表示されるとおり入力されなければなりません。
<山かっこ内のテキスト>	値を提供する必要のあるプレースホルダです。
[角かっこ内のテキスト]	任意のアイテムです。
{かっこ内のテキスト}	必要とされるアイテムを設定します。1 つ選択します。
垂直線 (|)	相互的に排除するアイテムのためのセパレーターです。1 つ選択します。
省略記号 (…)	繰り返すことのできるアイテム。

<filename>

コードが生成されるマッピングデザイン (.mfd) 、またはマッピングプロジェクト (.mfp) ファイルです。プロジェクト全体のためにコードを生成する場合、ターゲット /GENERATE を設定して、<filename> としてプロジェクト パスを入力します (下の説明を参照してください)。例えば、**MapForceExamples.mfp**。

/{target}

コードが生成されるターゲット言語または環境を指定します。次のコード生成ターゲットがサポートされます。

ターゲット	説明
/COMPILE[:compileoptions]	MapForce Server 実行可能ファイル (.mfx) にマッピングをコンパイルします。任意で、次のコンマで区切られたオプションが与えられます: JDBC — JDBC ドライバーとデータベースコンポーネント設定ダイアログボックス内で定義されたデータベースURL を使用するJDBC への全てのデータベース接続を変換します。データベースコンポーネント設定も参照してください。 NOXMLSIGNATURES — MapForce Server 実行ファイル内の電子署名の生成を抑制します (MapForce Server は自動署名をサポートしないことご注意してください)。
/GENERATE	プロジェクトファイル内に全てのマッピングのためのプロジェクトコードを現在のフォルダー設定を使用して生成します。プロジェクトフォルダーの管理を参照してください。このターゲッ

(C) 2015-2021 Altova GmbH

ターゲット	説明
	トを選択すると、MapForce プロジェクト (.mfp ファイル) プロジェクトを <filename> として提供してください。
/XSLT	XSLT 1.0 コードを生成します。
/XSLT2	XSLT 2.0 コードを生成します。
/XSLT3	XSLT 3.0 コードを生成します。
/XQuery	XQuery コードを生成します。
/JAVA	Java コードを生成します。
/CS	C# コードを生成します。このコマンドは任意で特定のコード生成オプションの設定を許可します、具体的には /CS[:{VS2010|VS2013|VS2015|VS2017|VS2019}] VS2010 Visual Studio 2010 VS2013 Visual Studio 2013 VS2015 Visual Studio 2015 VS2017 Visual Studio 2017 VS2019 Visual Studio 2019 Visual Studio バージョンが指定されていないと、コード生成オプション内で定義された Visual Studio バージョンを使用してコードが生成されます。コードジェネレーターオプションを参照してください。
/CPP	C++ コードを生成します。このコマンドはオプションで特定のコード生成オプションの設定を許可します。具体的には /CPP[:{VS2010|VS2013|VS2015|VS2017},{MSXML|XERCES3},{LIB|DLL},{MFC|NoMFC}] 最初のオプショングループはターゲット Visual Studio バージョンを設定します。有効な値: VS2010 Visual Studio 2010 VS2013 Visual Studio 2013 VS2015 Visual Studio 2015 VS2017 Visual Studio 2017 VS2019 Visual Studio 2019 2番目のオプショングループは生成されたコードによりターゲットとされる XML ライブラリを指定します。有効な値:

ターゲット	説明
	MSXML — MSXML 6.0 のためにコードを生成します XERCES3 — Xerces 3 のためにコードを生成します 3番目のオプショングループは、動的なライブラリとは反対に静的なライブラリが生成されるかを指定します。有効な値: LIB — 静的な LIB ライブラリを生成します DLL — DLL ライブラリを生成します 4番目のオプショングループはMFC サポート有り、または無しでコードが生成されるかを指定します。有効な値: MFC — MFC サポートを有効化します NoMFC — MFC サポートを無効化します 上のオプションが指定されていない場合、コードはコード生成オプション内で指定されているVisual Studio バージョンを使用して生成されます。コードジェネレーターオプションを参照してください。

<outputdir>

出力ディレクトリを指定する任意のパラメーターです。出力パスが与えられていない場合、現在の作業ディレクトリが使用されます。相対的なファイルパスは、現在の作業ディレクトリに対して相対的であることに注意してください。

/GENERATE がターゲットで、<outputdir> パラメーターが設定されていない場合、コード生成言語と各マッピングの出力パスは、プロジェクト内の各フォルダーのために定義されている設定により提供されます。プロジェクトフォルダーの管理を参照してください。

/GENERATE がターゲットで、<outputdir> パラメーターが設定されている場合、<outputdir> 値はコマンドラインで与えられ、ルートプロジェクトレベルで定義されている出力ディレクトリより先高い優先順位が与えられます。しかしながら、プロジェクト内の各フォルダーで定義されているコード生成設定よりは高い優先順位が与えられません。

/options

/options は相互排他的ではありません。1つ、または複数のオプションを設定することができます。

オプション	説明
[/MFXVERSION[:<version>]	ターゲットが /COMPILE の場合このオプションを適用することができます。特定の MapForce Server のバージョンのために MapForce Server Execution (.mfx) ファイルをコンパイルします。**2013r2** から現在の MapForce バージョンまでの MapForce Server のバージョンを値として与えることができます。次も参照してください :MapForce Server 実行可能ファイルにマッピングをコンパイルする。
/GLOBALRESOURCEFILE <filename>	マッピングが入力ファイル、出力ファイル、フォルダーパス、またはデータベースを解決するためにグローバルリソースを使用する場合、このオプションを適用することができます。詳細に関

(C) 2015-2021 Altova GmbH

オプション	説明
	しては、Altova グローバルリソースを参照してください。 オプション /GLOBALRESOURCEFILE はグローバルリソース.xml ファイルへのパスを指定します。/GLOBALRESOURCEFILE が設定されている場合、/GLOBALRESOURCECONFIG が設定されなければなりません。
/GLOBALRESOURCECONFIG <config>	このオプションはグローバルリソース構成の名前を指定します（前のオプションも参照してください）。/GLOBALRESOURCEFILE が設定されている場合、/GLOBALRESOURCECONFIG が設定されなければなりません。
/LIBRARY <libname> (...)	コード生成ターゲット言語と共に使用して追加関数ライブラリを指定します。このオプションを、複数のライブラリをロードするために複数回ロードすることができます。関数ライブラリの管理も参照してください。
/LOG <logfilename>	指定されたパスでログファイルを生成します。<logfilename> はフルパスとして使用することができます。例えば、ディレクトリとファイル名の両方を含むことができます。しかしながら、フルパスが与えられると、ログファイルを生成するためにログファイルが存在する必要があります。ファイル名のみが指定されると、ファイルは Windows コマンドプロンプトの現在のディレクトリ内に保存されます。

リマーク

- 相対パスは MapForce を呼び出すアプリケーションの現在のディレクトリである作業ディレクトリに対して相対的です。これは、.mfd ファイル名、.mfp ファイル名、出力ディレクトリ、ログファイル名、およびグローバルリソースファイル名のパスに対して適用されます。
- コマンドラインではバックスラッシュと終わり引用符を最後に使用しないでください（例えば、"C:¥My directory¥"）。これら2つの文字はコマンドラインパーサーにより二重引用符マークとして解釈されます。引用符が必要な場合、（"c:¥My Directory¥¥"）、二重の円記号 ¥¥ を使用します。または、スペースの使用を回避してください。

サンプル

1) MapForce を開始して、マッピング <filename>.mfd を開くには、次を使用します:

```
MapForce.exe <filename>.mfd
```

2) XSLT 2.0 コードを生成し、<logfilename> という名前のログファイルを作成するには、次を使用します:

```
MapForce.exe <filename>.mfd /XSLT2 <outputdir> /LOG <logfilename>
```

3) グローバルリソースファイル<grfilename> からのグローバルリソース構成<grconfigname> を考慮して XSLT 2.0 コードを生成するには、次を使用します:

```
Mapforce.exe <filename>.mfd /XSLT2 <outputdir> /GLOBALRESOURCEFILE
<grfilename> /GLOBALRESOURCECONFIG <grconfigname>
```

4) Visual Studio 2015 のために C# アプリケーションを生成し、ログファイルを出力する場合、次を使用します:

```
MapForce.exe <filename>.mfd /CS:VS2015 <outputdir> /LOG <logfilename>
```

5) 「ツール | オプション」内で定義されているコード生成設定を使用し C++ アプリケーションを生成し、ログファイルを作成するには、次を使用します:

```
MapForce.exe <filename>.mfd /CPP <outputdir> /LOG <logfilename>
```

6) Visual Studio 2015、MSXML、のための C++ アプリケーションを静的なライブラリ、MFC サポート、を使用してログファイル無しで生成するには、次を使用します:

```
MapForce.exe <filename>.mfd /CPP:VS2015,MSXML,LIB,MFC
```

7) Visual Studio 2015、Xerces、のための C++ アプリケーションを動的なライブラリを使用して MFC サポート無しでログファイルと共に生成するには、次を使用します:

```
MapForce.exe <filename>.mfd /CPP:VS2015,XERCES,DLL,NoMFC <outputdir> /LOG <logfilename>
```

8) Java アプリケーションを生成し、ログファイルを出力する場合、次を使用します:

```
MapForce.exe <filename>.mfd /JAVA <outputdir> /LOG <logfilename>
```

9) (プロジェクト内部の各フォルダーの)フォルダー設定内で定義されている言語と出力ディレクトリを使用してプロジェクト内に全てのマッピングのためのコードを生成するには、次を使用します:

```
MapForce.exe <filename>.mfp /GENERATE /LOG <logfilename>
```

10) プロジェクトファイル内の全てのマッピングのために Java コードを生成するには、次を使用します:

```
MapForce.exe <filename>.mfp /JAVA /LOG <logfilename>
```

フォルダー設定内で定義されているコード生成言語は無視され、全てのマッピングのために Java が使用されることにご注意してください。

11) 以前にコンパイルされた Java マッピングのためにコマンドラインで入力と出力ファイルを提供するには、次を使用します:

```
java -jar <mappingfile>.jar /InputFileName <inputfilename> /OutputFileName
<outputfilename>
```

`/InputFileName` と `/OutputFileName` パラメーターは、コマンドラインの実行内でパラメーターの使用を許可する MapForce マッピング内の特別な入力コンポーネントの名前です(マッピングにパラメーターを与えるを参照してください)。

12) MapForce Server バージョン 2021 のために MapForce Server 実行ファイルにマッピングをコンパイルし、XML 署名を抑制するには、次を使用します:

```
MapForce.exe <filename>.mfd /COMPILE:NOXMLSIGNATURES <outputdir> /MFXVERSION:2021 /LOG
<logfilename>
```


(C) 2015-2021 Altova GmbH

11 Altova グローバルリソース

Altova グローバルリソースはファイル、フォルダー、またはデータベースへのポータブルなレファレンスです。グローバルリソースとして保管されると、パスとデータベース接続の詳細は再利用できるようになりAltova アプリケーション全体で使用可能になります。例えば、複数のAltova デスクトップアプリケーション内で同じファイルを頻繁に開く必要がある場合、グローバルリソースとして定義すると便利な場合があります。このようにすると、「ファイルを開く」ダイアログボックスから対応するグローバルリソースを開くことができるため、ファイルパスを覚えている必要がありません。この方法はファイルパスが変更されると、パスの一部を変更するのみという利点があります。

グローバルリソースの一般的な使用法は、データベース接続を一度定義し、グローバルリソースをサポートする全てのAltova アプリケーションで再利用することです。例えば、MapForce マッピングがデザインされるマシンでデータベース接続を作成し、MapForce Server がマッピングを実行するマシン上で同じ接続を再利用することができます（これは、一部の場合、両方のマシンに同じデータベースクライアントソフトウェアがインストールされている場合があります）。

任意で、（「構成」として既知の）同じグローバルリソースの複数のバリエーションを作成することができます。これによりファイル、またはフォルダーパス（またはデータベース）を必要に応じて変更することができます。例えば、「開発」と「生産」の構成を持つデータベースリソースを作成することができます。MapForce を例に挙げると、マッピングをプレビューする前に、希望する構成をリストから選択して、開発、または生産データベースからデータを簡単に抽出することができます。

次のAltova デスクトップアプリケーションからグローバルリソースを作成することができます: Altova Authentic、DatabaseSpy、MobileTogether Designer、MapForce、StyleVision、およびXMLSpy。サーバー製品では、次のAltova サーバーアプリケーションでグローバルリソースを作成することができます: FlowForce Server、MapForce Server、RaptorXML Server、RaptorXML+XBRL Server。

グローバルリソース は、多種のシナリオのためにMapForce 内で使用することができます:In MapForce 内では、グローバルリソース（ファイル、フォルダー、または、データベース参照など）を多種のシナリオのために使用することができます。例:

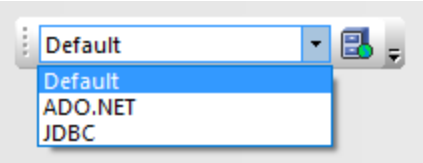
- マッピングの入力として構成することのできるファイルパスを宛てる場合。次を参照してください: 例: 変数入力ファイルを持つマッピングを実行する。
- マッピングの出力を構成することのできるパスにリダイレクトする詳細に関しては、次を参照してください: 例: 出力を変数フォルダに出力する。
- マッピングで使用される場合、構成することのできるパスをStyleVision Power Stylesheet（.sps）ファイルに与える場合。例えば、StyleVision .sps ファイルをMapForce コンポーネント設定から参照する代わりに、グローバルリソースとして定義されている、2つの構成を持つことのできる.sps ファイル（例えば、「W ebsite.sps」と「Print.sps」）を使用することができます。次も参照してください: StyleVision を使用してマッピング出力のスタイルを指定する。
- データベース接続を再利用する場合。（Altova アプリケーション内で）データベースが既にグローバルリソースとして定義されている場合、セットアップのステップをもう一度行う必要なく、接続を構築することができます。次を参照してください: グローバルリソースから接続を使用する。
- マッピングがデータを読み取るデータベースを切り替える、またはマッピングデータを書き込むデータベースを簡単に切り替えることができます（これは、2つ以上のデータベースが同じ構成を持ち、異なるデータを含むことが前提とされます。次を参照してください: 例: データベースの切り替え）。

- MapForce Basic Edition は、グローバルリソースとして定義されるデータベース接続をサポートしません。

11.1 グローバルリソースの作成

グローバルリソースはファイル、またはフォルダーパス、またはデータベース接続を表す再利用可能なレファレンスです。グローバルリソースは一度定義されると、複数のAltova アプリケーション全般でサポートするコンテキスト内で必要な回数再利用することができます。データベースを例に挙げると、1つ以上のAltova アプリケーションで特定のデータベースと共に頻繁に作業する場合、グローバルリソースとしてデータベース接続を追加することはよいアイデアです。このようにすると、他のAltova アプリケーションから同じデータベースに接続する必要がある都度データベース接続ウィザードのステップを実行する必要がありません。

各グローバルリソースはいわゆる「構成」を持つことができます。構成によりAltova アプリケーションにより使用、または作成されるファイル、フォルダー、およびデータベース間を簡単に切り替えることができます。これはテストのシナリオで特に役に立ちます。例えば、異なるドライバーの型を持つ同じデータベースに接続する以下の3つの個別の接続から構成されるデータベースリソースを作成することができます: (a) デフォルトの接続型であるODBC。(b) JDBC、と(c) ADO.NET。このようにして、特定のドライバーを持つデータベースに接続するには、グローバルリソースドロップダウンリストから対応する構成を選択するだけです。



グローバルリソースドロップダウンリスト

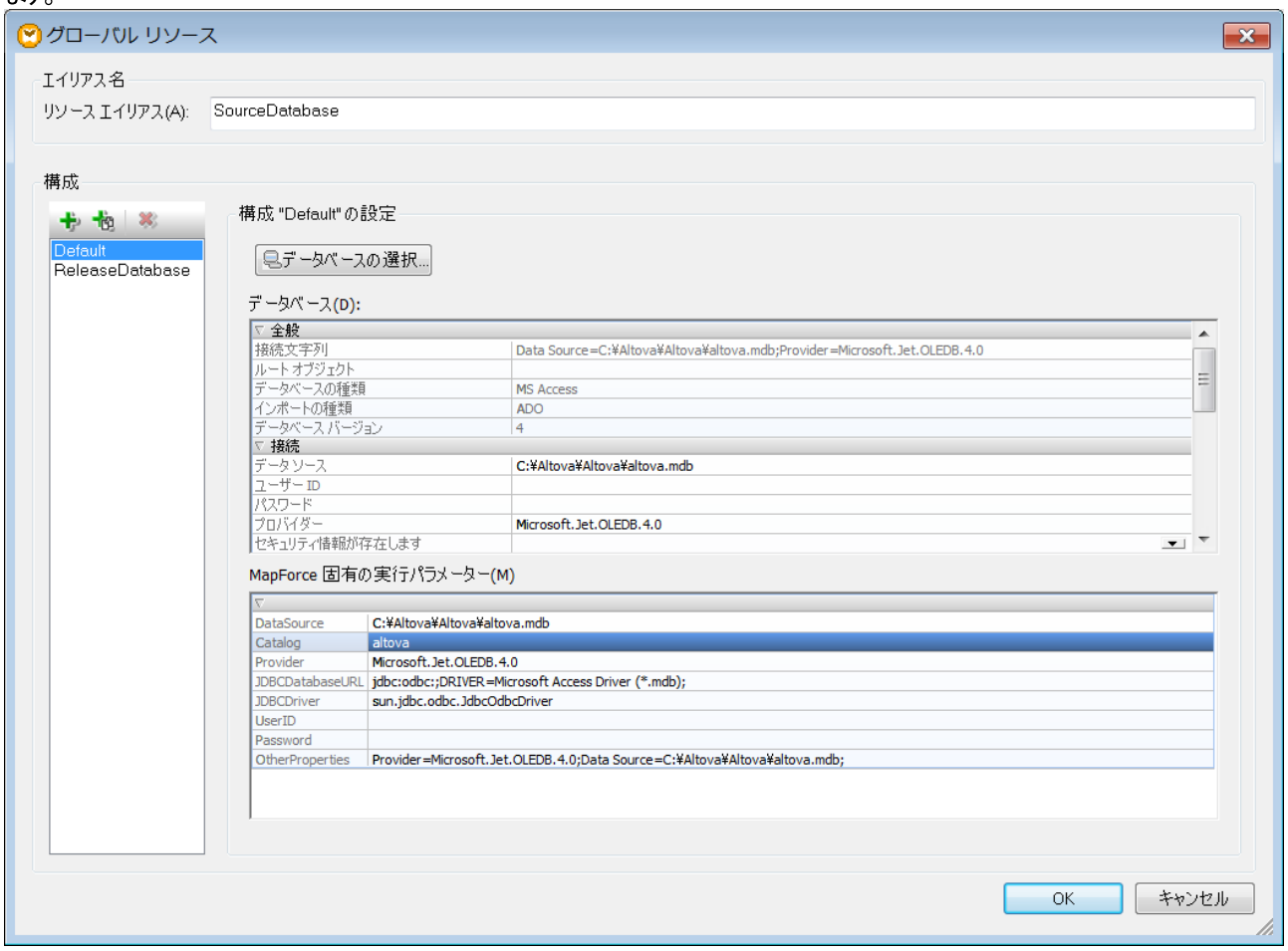
ボタンクリックのみで、構成は変数フォルダーにマッピング出力を生成することができます。例えば、フォルダーリソースを2つの構成から作成することができます: (a) ディレクトリ **C:\Testing** を指す「テスト」(b) ディレクトリ **C:\Production** を指す「生産」。グローバルリソースドロップダウンリストから必要とする構成をマッピングの実行の前に選択し **C:\Testing** または **C:\Production** フォルダーに出力を生成するマッピングを構成することができます。サンプル: 変数フォルダーに出力を生成する内で詳細について説明されています。

グローバルリソースを作成する方法:

1. 「ツール」メニューから「グローバルリソース」をクリックします。(または、「**グローバルリソースの管理**」 ツールバーボタンをクリックします。)
2. 「**追加**」をクリックして、(ファイル、フォルダー、データベースなどの) 作成するリソースの型を選択します。
3. 「**リソースエイリアス**」テキストボックス内に詳細名を入力します (例えば、"InputFile"、"OutputFolder"、"DatabaseConnection").
4. 「Default」構成をセットアップします:
   a) ファイル、またはフォルダーの場合、デフォルトでポイントされるこのリソースへのファイル、またはフォルダーを参照します。
   b) データベース接続の場合、「**データベースの選択**」をクリックして、データベース接続ウィザードに従い、データベースに接続します (データベースへ接続ください)。(グローバルリソースドロップダウンリストから他の構成が明示的に選択されている場合、または、サーバー実行内でコマンドラインパラメーターとして提供されていない場合以外) このデータベース接続はマッピングの実行の際にデフォルトで使用されます。
5. オプションとして、リソースに追加構成が存在する場合 (例えば、データベースの場合のドライバーの種類、または、ファイルとフォルダーの場合代替パスなどの) 「**構成の追加**」 ボタンをクリックして、(例えば、"ProductionFolder" or "JDBC_Alternative" など) 詳細名を入力し、次にようにセットアップします:
   a) ファイル、またはフォルダーの場合、前のステップ内で定義済みのデフォルトの構成の代替としてこのリソースによりポイントされるファイル、またはフォルダーを参照します。
   b) データベース接続の場合、データベースに接続するためにデータベース接続ウィザードに従います。このデータベース接続はデフォルトの代替として使用されます。
   一部の場合、デフォルトの構成のコピーとして構成を作成し、編集するほうが便利な場合があります。この場合、「**現在選択されている構成のコピーとして構成を追加する**」 ボタンをクリックします。
6. 必要とされる各追加構成のために前のステップを繰り返します。

11.2 グローバルリソースとしてのデータベース

データベース接続をグローバルリソースとして追加する場合、接続ウィザードは接続をセットアップするステップを導きます。データベース接続ウィザードの開始 を参照してください。ウィザードが完了すると、データベース接続パラメーターはグローバルリソースダイアログボックス上に表示されます。



グローバルリソース ダイアログボックス

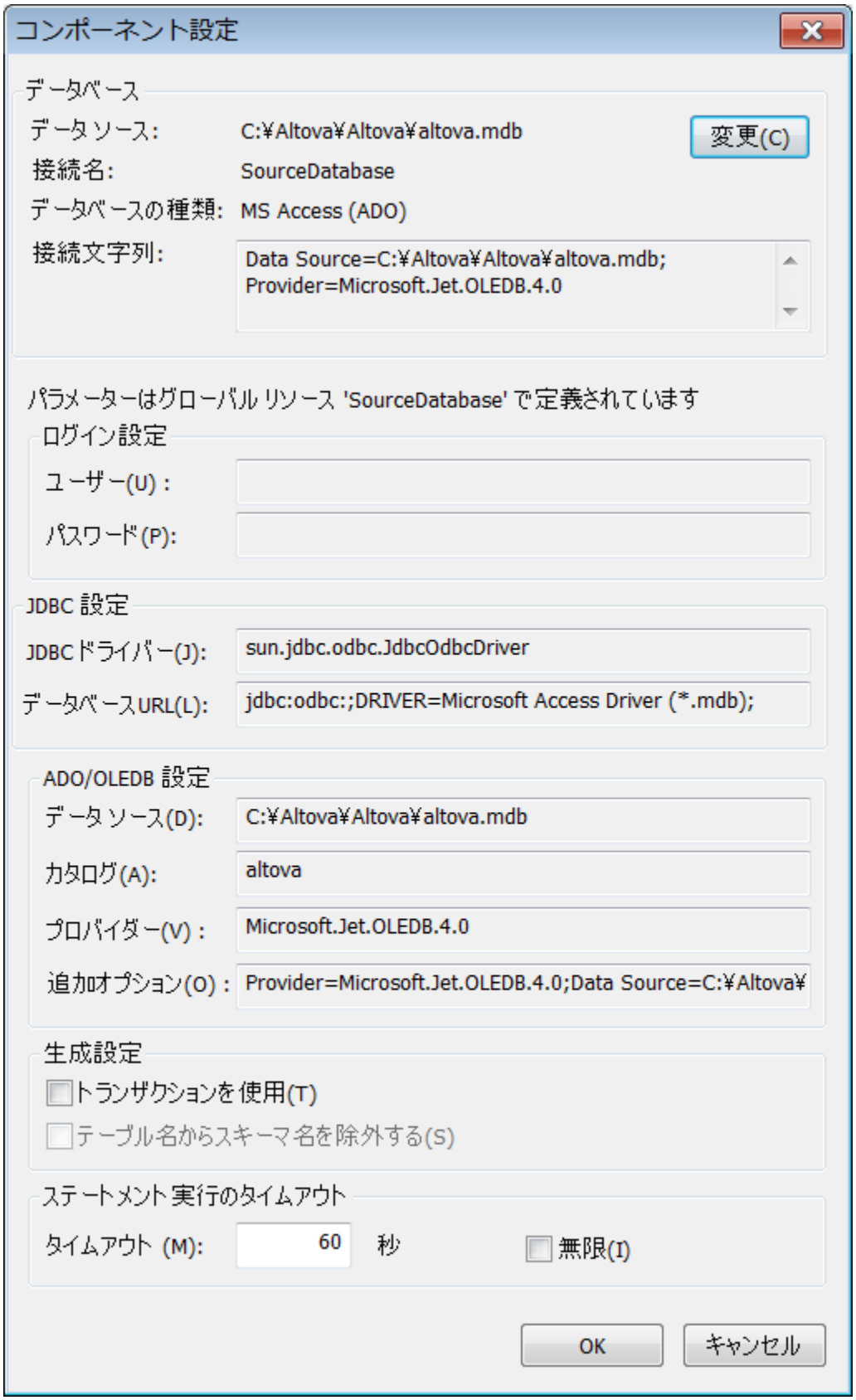
グローバルリソース ダイアログボックスでは、データベース 接続パラメーターの一部を編集することができます。上に表示されるように、パラメーターは2つのカテゴリにグループ化されています:

データベース	これらのパラメーターは、Altova アプリケーション間で共有されます。MapForce 内では、デザイン時に使用され、マッピングがロードされると、またはマッピングをプレビューするために MapForce 内で「出力」タブをクリックすると共有することができます。
MapForce 固有の実行 パラメーター	これらのパラメーターは、プログラムコードを生成した場合、または、マッピングを MapForce Server 実行ファイル(.mfx) にコンパイルした場合に適用されます。マッピングのランタイムで以下のように使用されます: • 生成された C++, C#、または Java プログラムコード内。

	• マッピングを MapForce Server 実行ファイルにコンパイルした場合、そして自動の JDBC 変換が実行された場合。自動の JDBC 変換に関する詳細に関しては、次を参照してください: 多種の実行環境内のデータベースマッピング。

マッピングがグローバルリソースを使用して、データベースに接続する場合、グローバルリソース ダイアログボックス内で表示されるデータベース接続の詳細がマッピング上で定義されている詳細よりも優先順位が高くなります。下で示される、コンポーネント設定 ダイアログボックス内では、データベース接続設定が灰色設定になっていることに注意してください。ダイアログボックスは、接続パラメーターがグローバルリソースとして設定されていることを通知します。

(C) 2015-2021 Altova GmbH


コンポーネント設定
データベース
データソース:
C:¥Altova¥Altova¥altova.mdb
変更(C)
接続名:
SourceDatabase
データベースの種類:
MS Access (ADO)
接続文字列:
Data Source=C:¥Altova¥Altova¥altova.mdb;
Provider=Microsoft.Jet.OLEDB.4.0
パラメーターはグローバル リソース 'SourceDatabase' で定義されています
ログイン設定
ユーザー(U):
パスワード(P):
JDBC 設定
JDBCドライバー(J):
sun.jdbc.odbc.JdbcOdbcDriver
データベースURL(L):
jdbc:odbc:;DRIVER=Microsoft Access Driver (*.mdb);
ADO/OLEDB 設定
データソース(D):
C:¥Altova¥Altova¥altova.mdb
カタログ(A):
altova
プロバイダー(V):
Microsoft.Jet.OLEDB.4.0
追加オプション(O):
Provider=Microsoft.Jet.OLEDB.4.0;Data Source=C:¥Altova¥
生成設定
トランザクションを使用(T)
テーブル名からスキーマ名を除外する(S)
ステートメント実行のタイムアウト
タイムアウト (M):
60
秒
無限(I)
OK
キャンセル

コンポーネント設定 ダイアログボックス

データベースに直接接続するために（グローバルリソースを使用せずに）データベースのコンポーネントを変更するには、**「変更」** をクリックして、データベースに接続するためにウィザードのステップを継続してください。

(C) 2015-2021 Altova GmbH

11.3 グローバルリソースとしての MapForce と StyleVision 変換結果

静的なファイルをポイントする代わりに、MapForce マッピング または StyleVision 変換より作成された特別なファイルを読み取るグローバルリソースを作成することができます この場合、グローバルリソースを使用する Altova アプリケーションは、最初に MapForce、または StyleVision を呼び出し、対応するマッピングまたは変換を実行し結果ファイルを取得します。これにより、Altova アプリケーション間のデータワークフローを定義することができます。(例えば、MapForce マッピング または StyleVision 変換の結果を入力として、他のマッピングまたは変換にパスする場合など)。例えば、グローバルリソースを使用して、XMLSpy が MapForce マッピングの結果 (出力) を使用する方法に関しては、次を参照してください: 例: アプリケーションワークフローを作成する。

マッピングの結果 (出力) をグローバルリソースとして使用できるようにするためには、 マッピングの変換言語を BUILT-IN に設定するか、または マッピングが BUILT-IN 言語によりサポートされるコンポーネントを含む必要があります。

> 上記のワークフローは同じコンピューターにインストール済みの Altova デスクトップアプリケーション間で意味があります。MapForce と StyleVision 変換の結果をグローバルリソースとして Altova サーバー製品と MapForce Basic Edition 内で使用することはできません。

Altova サーバーアプリケーションがどのようにグローバルリソースを使用方法の詳細は、MapForce Server 内のグローバルリソース と FlowForce Server 内のグローバルリソース を参照してください。

11.4 グローバルリソース XML ファイル

デフォルトでは、すべてのグローバルリソースは Altova アプリケーションの作成場所に関わらず、以下のパスに保存されます: **C:\Users\<username>\Documents\Altova\GlobalResources.xml**。これにより簡単にバックアップすることができ、Altova 製品がインストールされている他のワークステーションに対してポータブルになります。**GlobalResources.xml** ファイルの名前を変更、または、複製することができ、複数のグローバルリソースファイルを作成することができます。しかしながら、Altova アプリケーション内では 1 つのグローバルリソースファイルのみしかアクティブ化することはできません。

> 他の Altova アプリケーションとは異なり、FlowForce Server はグローバルリソースファイルを必要としません。代わりにリソースは他の FlowForce 構成データと同様に管理されます（アクセスパーミッションと共に再利用することが可能です）。詳細に関しては、FlowForce Server 内のグローバルリソースを参照してください。

アクティブなグローバルリソース ファイルをセットアップする方法:

1. 「**ツール**」メニューから、「**グローバルリソース**」をクリックします。（または、**グローバルリソース** ツールバーボタンをクリックします。）
2. 「**参照**」をクリックして、必要とされるグローバルリソース XML ファイルを選択します。

複数のグローバルリソースファイルを使用している場合、現在アクティブなファイルに現在のコンテキストにより必要とされているすべてのグローバルリソースファイルが含まれていることを確認してください。例えば、グローバルリソースを使用するパスからデータを読み取るようにマッピングが構成されている場合、現在アクティブなグローバルリソースファイルにはその特定のグローバルリソースが含まれている必要があります。それ以外の場合、「グローバルリソースを解決する際にエラーが発生しました」などのエラーメッセージが「**メッセージ**」ウィンドウに表示されます。

(C) 2015-2021 Altova GmbH

11.5 多種の実行環境内でのグローバルリソース

11.5.1 SLT、XSLT2、XQuery 内のグローバルリソース

XSLT、XSLT2、XQuery 内のグローバルリソース

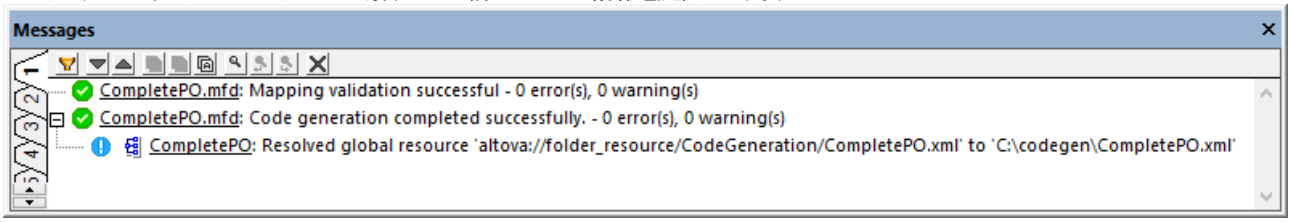
XSLT または XSLT2 コードを生成し、とマッピングがグローバルリソースを使用する場合、生成された XSLT スタイルシートに影響はありません。グローバルリソースの有無に関わらず、入力と出力ファイルは永続的に割り当てられず、XSLT プロセッサー内で XSLT スタイルシートを開始すると柔軟に指定することができます。同様が生成された XQuery コードに適用することができます。

RaptorXML 実行により生成された **DoTransform.bat** ファイルのみが例外です。マッピングにより使用されたグローバルリソースは、グローバルリソースドロップダウンリストから現在選択されている値（構成）を考慮して、**DoTransform.bat** 内の実際のパスに対して解決されます。

RaptorXML へグローバルリソースを適用するための情報は、RaptorXML ドキュメントを参照してください (https://www.altova.com/ja/documentation.html)。

11.5.2 C++、C#、Java 内のグローバルリソース

C#、C++、または Java プログラムコードを生成する場合、マッピングにより使用されるグローバルリソースは解決されます。例えば、グローバルリソースとして定義されているファイルまたは、フォルダーエイリアスは実際のファイル、またはフォルダーパスに変換されます。特定のグローバルリソース構成がグローバルリソースドロップダウンリストから選択された場合、選択された構成のためにコードが生成されます。「**メッセージ**」ウィンドウは、どのようにグローバルリソースが解決された課についての情報を提供します。例えば



特定のグローバルリソース構成のためのコードを生成するには、コードを生成する前に、グローバルリソースドロップダウンリストから選択します。または、コマンドラインからコードを生成する場合、コマンドラインで **GLOBALRESOURCEFILE** と **GLOBALRESOURCECONFIG** パラメーターを与えます。（次も参照してください: MapForce コマンドライン インターフェイス）。

グローバルリソースを生成されたコードから切り替え、または参照することはできません（代わりに、入力、または出力 ファイル パスを変更するためにコードを変更することができます）。

メモ C# または Java 内では、入力、または出力のパスだけでなくデータ型も変更することができます。次を参照してください: マッピング入力/出力（C#、Java）のデータ型の変更。

11.5.3 MapForce Server 内のグローバルリソース

MapForce Server 実行可能ファイル（.mfx）にマッピングをコンパイルすると、使用されるグローバルリソースレファレンスは解決されず保管されます。これはマッピングの実行に成功するためにサーバー側にこれらのレファレンスを提供する必要があることを意味します。具体的には、MapForce Server 内では、グローバルリソースを使用する .mfx ファイルを実行するために以下が必要とされています:

1. グローバルリソースが定義されているファイル。デフォルトでは、すべてのグローバルリソースは、MapForce がインストールされているマシン上の **Documents\Altova** フォルダー内の **GlobalResources.xml** というファイル内で定義されています。

MapForce Server が作動するマシンにこのファイルをコピーして、必要に応じて複数のこのようなファイルを作成することができます。グローバルリソース XML ファイルを参照してください。

2. グローバルリソース構成名。各グローバルリソースは、構成名により識別されます。デフォルト名は、「デフォルト」ですが、必要に応じて追加の構成を作成することができます。 グローバルリソースの作成 で説明されているとおり「デフォルト」以外の希望する構成名を指定する必要があります。

MapForce 内では、グローバルリソースファイルパスとグローバルリソース構成名は、グラフィカルなユーザーインターフェイスから変更、および、設定することができます。 MapForce Server 内では、これらはマッピングランタイムに設定されます。

- コマンド ラインインターフェイスを使用してマッピングを実行すると、 `run` コマンドの後、オプション **`--globalresourceconfig`** と **`--globalresourcefile`** を設定します。例:

```
C:\Program Files (x86)\Altova\MapForceServer2021\bin\MapForceServer.exe run
SomeMapping.mfx --globalresourcefile="C:
\Users\me\Documents\Altova\GlobalResources.xml" --globalresourceconfig="Default"
```

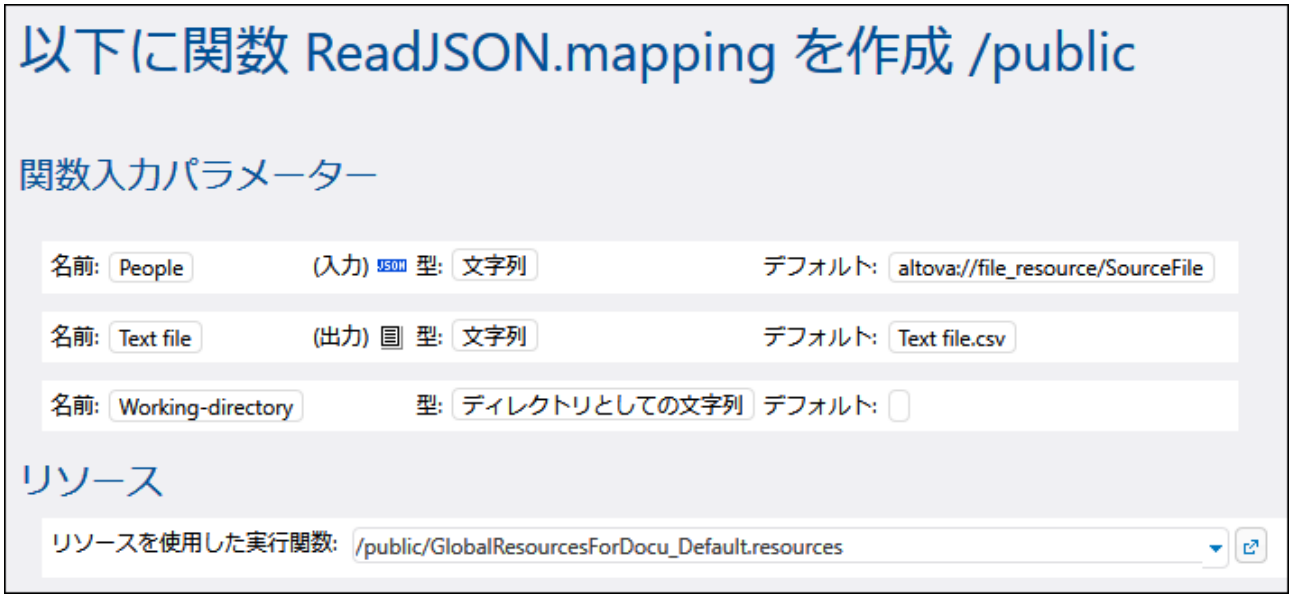
- MapForce Server API を使用してマッピングを実行する場合、**`Run`** メソッドを呼び出す前に、メソッド **`SetOptions`** を2回呼び出してください。最初の呼び出しは、オプションとしてグローバルリソース XML ファイルパスを提供するために必要です。2番目の呼び出しはグローバルリソース構成名を提供するために必要です。

詳細に関しては MapForce Server ドキュメントを参照してください(https://www.altova.com/ja/documentation.html)。

11.5.4 FlowForce Server 内のグローバルリソース

FlowForce Server の観点からは、デスクトップアプリケーション同様に1つの XML ファイルに保管されないため、グローバルリソースはある程度グローバルではなくなります。FlowForce 内では、各リソースファイルまたはフォルダーパス、またはデータベース接続詳細を含む可能性のあるは再利用可能なオブジェクトです。リソースをコピー、エクスポート、またはインポートすることができ、他の FlowForce Server オブジェクト同様に同じユーザーアクセスの対象となることができます。つまり、FlowForce ユーザーはマッピング関数内のすべてのリソースを使用することができます。

MapForce 内でグローバルリソースを使用するマッピングを作成すると、FlowForce Server にデプロイすることができます。デプロイ時にデプロイメントダイアログボックス上の「**リソースの使用**」チェックボックスを選択することにより、マッピングがグローバルリソースを使用するかを選択することができます。チェックボックスを選択しない場合、マッピングにより使用されるグローバルリソースは現在選択されている構成をベースに解決されます。チェックボックスを選択すると、FlowForce Server 内でもマッピング関数はリソースを必要とします。次のマッピング関数のサンプルはリソースの実行を必要とする FlowForce にデプロイされます。(最初のパラメーターがリソースファイルからデフォルトのファイルパスを取得することに注意してください):



上で説明されているとおりFlowForce Server 内では、ジョブではなくマッピング関数がグローバルリソースを使用します。例えば、マッピングがパスから読み取り、または、パスへ書き込むように構成されている場合が挙げられます。上のサンプルでは、マッピング関数はリソースから最初の入力ファイルを *読み取ります* 。ジョブ構成ページからのパスをオーバーライドしない限り、これはこの関数を使用するすべてのジョブが同じパスを使用することを意味します。

FlowForce Server にグローバルリソースをスタンドアロンオブジェクトとしてデプロイすることができます。つまり、グローバルリソースをデプロイするためにマッピングを最初にデプロイする必要はありません。詳細に関しては[FlowForce Server にリソースをデプロイする](#) を参照してください。。

FlowForce Server 内でのリソースの使用に関する詳細はFlowForce Server ドキュメントを参照してください (https://www.altova.com/ja/documentation.html).

11.5.5 FlowForce Server にリソースをデプロイする

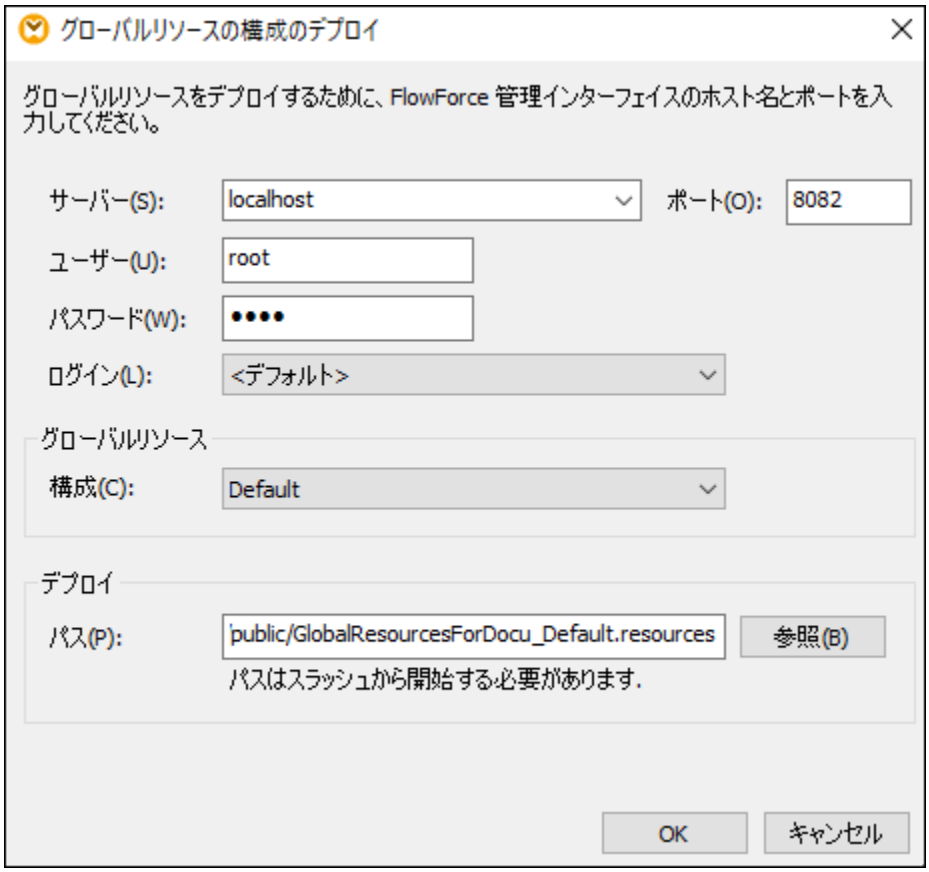
MapForce で作成されたグローバルリソースを FlowForce Server にデプロイすることができます。デプロイの際には、サーバーにデプロイされるリソースを持つ構成を選択する必要があります。サーバー上で同じグローバルリソースのすべての構成が必要な場合、複数回デプロイし、デプロイの都度必要な構成を選択することができます。サーバー上で各グローバルリソースの名前を変更し、サーバー上の目的コンテナーを選択することができます。

> Altova アプリケーションを作動するグローバルリソースはサーバー環境ではサポートされていません。詳細に関しては、[グローバルリソースとしてのMapForce とStyleVision 変換結果](#)を参照してください。

FlowForce Server にグローバルリソースをマッピングのデプロイ時、または、個別にデプロイすることができます。

FlowForce Server にグローバルリソースをデプロイする方法:

1. 「**ツール**」メニューから、「**グローバルリソース**」をクリックします。(または、「**グローバルリソースの管理**」 ツールバーボタンをクリックします。)
2. 「**サーバーにデプロイする**」をクリックします。ダイアログボックスを開きます。



3. FlowForce Server（サーバー、ポート、ユーザー、パスワード、ログインメソッド）への接続詳細を入力します。これらのパラメーターはマッピングをデプロイする際に必要とされるパラメーターと同じです。FlowForce Server にマッピングをデプロイするを参照してください。
4. 「構成」リストから構成を選択します。このリストにはグローバルリソースファイルからのすべての構成が含まれています。

グローバルリソース構成を一度にデプロイすることができます。上記の通り、全ての構成がサーバー上で必要な場合、異なる名前を持つ同じリソースを複数回デプロイすることができます。

5. サーバー上でリソースが保存されるターゲットパスを選択してください。「参照」をクリックして、ターゲット FlowForce コンテナーを選択、または必要な場合新規を作成することのできるダイアログボックスを表示します。
6. 「OK」をクリックします。

FlowForce Server へのグローバルリソースデプロイはメッセージボックスに報告されます。

(C) 2015-2021 Altova GmbH

11.6 例: 変数入力ファイルを持つマッピングを実行する

入力として XML ファイルを取るマッピングを頻繁に実行すると仮定します。入力 XML を変更する必要がある場合、ソース XML コンポーネントのプロパティを編集し、新規の入力ファイルを参照することができます。マッピングの入力 XML を変更する場合、ソース XML コンポーネントのプロパティを開き、新規の入力ファイルを参照します。次を参照してください: コンポーネント設定の変更。これは 1 度きりのタスクの場合、簡単に達成することができます。しかしながら、マッピングの入力 XML ファイルを一日に、または毎時に、複数回実行する場合、例えば、XML ファイルをマッピングの入力として使用し、レポートを生成するためにマッピングを実行する場合、他の XML ファイルから同じレポートを毎晩作成する場合などが挙げられます。

このような場合、グローバルリソースを使用することができます: マッピングを複数回編集する代わりに (または、複数のコピーを保管する代わりに) グローバルリソース (または「ファイルエイリアス」) として定義されているファイルから読み取りマッピングを構成することができます。このサンプルで挙げられる必要条件を満たすには、ファイルエイリアスが 2 つの構成を持つように構成することができます:

1. 「デフォルトの」 – この構成は、「morning」 XML ファイルをマッピングの入力として提供します。
2. 「EveningReports」 – この構成は、「evening」 XML ファイルをマッピングの入力として提供します。

構成を正しく設定することにより、これらの入力ファイルを使用してマッピングを実行することができます。ファイルエイリアスセットアップされると、下で示されるとおり、マッピングを実行する前に、希望する構成をドロップダウンリストから選択することができます。

ステップ 1: グローバルリソースを作成する

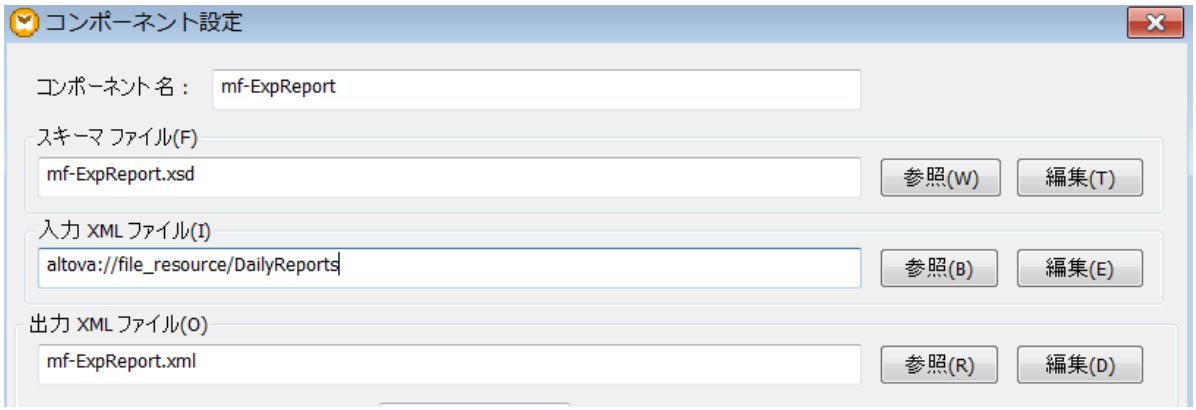
ファイルエイリアスを以下のとおり作成することができます:

1. 「ツール」 メニューから「**グローバルリソース**」 をクリックします。(または、**グローバルリソース** ツールバーボタンをクリックします)。
2. 「**追加 | ファイル**」をクリックします。
3. 「**リソースエイリアス**」 テキストボックスに名前を入力します (このサンプルでは、「DailyReports」 が適切な名前です)。
4. 「**参照**」 をクリックして、次のファイルを選択します: <**マイドキュメント>\Altova\MapForce2021\MapForceExamples\Tutorial\mf-ExpReport.xml**.
5. 「**ファイルの追加**」 をクリックして、「EveningReports」という名前を与えます。
6. 「**参照**」 をクリックして、今度は、次のファイルを選択します: <**マイドキュメント>\Altova\MapForce2021\MapForceExamples\Tutorial\mf-ExpReport2.xml**.

ステップ 2: マッピング内のグローバルリソースを使用する

必要とされる グローバルリソースが作成されましたが、マッピングを使用することはまだできません。前に定義されたファイルエイリアス (グローバルリソース) から読み込みマッピングを変更するには、以下を行います:

1. 次のマッピングを開きます <**マイドキュメント**>¥Altova¥MapForce2021¥MapForceExamples¥Tutorial¥**Tut-ExpReport.mfd**.
2. マッピング上のソースコンポーネントのヘッダーを右クリックして、コンテキストメニューから「**プロパティ**」 を選択します。
3. **入力** XML ファイルの横の「**参照**」をクリックします。
4. 「**グローバルリソースに切り替える**」 をクリックして、前に定義されたファイルエイリアス「DailyReports」 を選択します。
5. 「**開く**」をクリックします。入力 XML ファイルパスは、グローバルリソースを使用するパスを示す **altova://file_resource/DailyReports** になりました。



ステップ 3: 希望する構成を使用してマッピングを実行する

マッピングを実行するまえに、入力 XML ファイルを以下のように簡単に切り替えることができます:

- ファイル **mf-ExpReport.xml** を入力として使用するには、「**ツール**」メニューから「**アクティブな構成 | デフォルト**」をクリックします。
- ファイル **mf-ExpReport2.xml** を入力として使用するには、「**ツール**」メニューから「**アクティブな構成 | EveningReports**」をクリックします。,

または、**グローバルリソース**ドロップダウンリストから必要とされる構成を選択します。

マッピングの結果を構成を使用してプレビューするには、「**出力**」タブをクリックして、生成された出力の差分を確認してください。

(C) 2015-2021 Altova GmbH

11.7 例: 出力を変数フォルダに出力する

このサンプルはグローバルリソースを使用して、異なるフォルダーにマッピングの出力をリダイレクトする方法について説明しています。

マッピング出力をディレクトリ（例えば、**C:\Testing**）に生成する場合、または出力が他のディレクトリ（例えば、**C:\Production**）に生成する場合があるとします。グローバルリソースを使用すると、2つの構成を持つフォルダーエイリアスを作成することが可能になります:

1. 「デフォルトの」構成 – 出力を以下の場所に生成します: **C:\Testing**
2. 「Production」構成 – 出力を以下の場所に生成します: **C:\Production**.

この目的を達成するためにステップは以下のとおりです。

ステップ 1: グローバルリソースを作成する

フォルダーエイリアスを以下のとおり作成することができます:

1. 「**ツール**」メニューから「**グローバルリソース**」をクリックします。（または、**グローバルリソース** ツールバーボタンをクリックします）。
2. 「**追加 | フォルダー**」をクリックします。
3. 「**リソースエイリアス**」テキストボックスに名前を入力します（このサンプルでは、「OutputDirectory」は適切な名前です）。
4. 「**参照**」をクリックして、次のフォルダーを選択します: **C:\Testing**。（このフォルダが使用中のオペレーティングシステムで既存であることを確認してください）。
5. 「**ファイルの追加**」 をクリックして、新しい構成のために名前を入力します（このサンプルでは、「ProductionDirectory」）。
6. 「**参照**」をクリックして、次のフォルダーを選択します次のフォルダー: **C:\Production**（このフォルダが使用中のオペレーティングシステムで既存であることを確認してください）。

ステップ 2: マッピング内のグローバルリソースを使用する

必要とされるグローバルリソースが作成されました。しかしながら、マッピングをまだ使用することはできません。前に定義されたフォルダーエイリアス（グローバルリソース）から使用するようにマッピングを変更するには、以下を行います:

1. 次のマッピングを開きます <**マイドキュメント**>¥Altova¥MapForce2021¥MapForceExamples¥Tutorial¥**Tut-ExpReport.mfd**.
2. マッピング上のターゲットコンポーネントを右クリックし、コンテキストメニューから「**プロパティ**」を選択します。
3. **出力** XML ファイルの横の「**参照**」をクリックします。
4. 「**グローバルリソースに切り替える**」をクリックして、「**保存**」をクリックします。
5. 出力 XML ファイルを保存するようにプロンプトされると、**output.xml**（または他の出力ファイルに与える詳細なファイル名）を入力します。出力 XML ファイルパスは、グローバルリソースとして定義されるパスを示す **altova://folder_resource/OutputDirectory/output.xml** になります。

ステップ 3: 希望する構成を使用してマッピングを実行する

マッピングを実行する前に、希望するマッピング出力フォルダーを簡単に切り替えることが以下のように行うことができます:

- 「**ツール**」メニューから「**アクティブな構成 | デフォルト**」をクリックします。マッピングの結果をプレビューするために「**出力**」タブをクリックします。マッピング出力（下で説明されているとおり、一時、または永続ファイル）が **C:\Testing** ディレクトリに出力されます。
- 「**ツール**」メニューから**アクティブな構成 | ProductionDirectory** をクリックして、「**出力**」タブをクリックします。マッピング出力（下で説明されているとおり、一時、または永続ファイル）は、**C:\Production** ディレクトリ内に生成されます。

メモ MapForce を構成して、出力が永続ファイルに書き込まれるように設定されている以外は、マッピング出力は、デフォルトで、一時ファイルとして書き込まれます。

MapForce に一時ファイルの代わりに永続ファイルを生成するように構成するには、以下を行います:

1. 「**ツール**」メニューから「**オプション**」をクリックします。
2. 「**全般**」セクションで、オプション「**最終出力ファイルに直接書き込む**」を選択します。

(C) 2015-2021 Altova GmbH

11.8 例: データベースの切り替え

データベースからデータを読み取り、データベースにデータを書き込む場合、データベース接続をマッピングのランタイムの前に切り替えることができます。(例えば、リリースから生産データベースに、またはその逆を行うことができます)。このサンプルは、この目的をグローバルリソースを使用することにより達成します。データベースの切り替えは、同じ構造を持つ両方のデータベースが異なるデータを含んでいることを意味します。このサンプルの目的のために、次のMicrosoft Access データベースと作業します:

- **altova.mdb** は次のディレクトリ内にあります: <**マイドキュメント**>¥Altova¥MapForce2021¥MapForceExamples¥このデータベースはデフォルトの開発データベースの役割を果たします。
- **altova.mdb** は次のディレクトリ内にあります: <**マイドキュメント>\Altova\MapForce2021\MapForceExamples\Tutorial**. このデータベースはデフォルトの生産データベースの役割を果たします。

必要条件は、マッピングを編集することなく、ランタイムの前に、2つのデータベースのどちらかにマッピングを提供することです。この必要条件を達成するには、2つの構成を持つデータベースグローバルリソース(データベースエイリアス)を作成します:

1. *Default*. この構成はデフォルトの開発データベースを指します。
2. *Release*. この構成はデフォルトのリリースデータベースを指します。

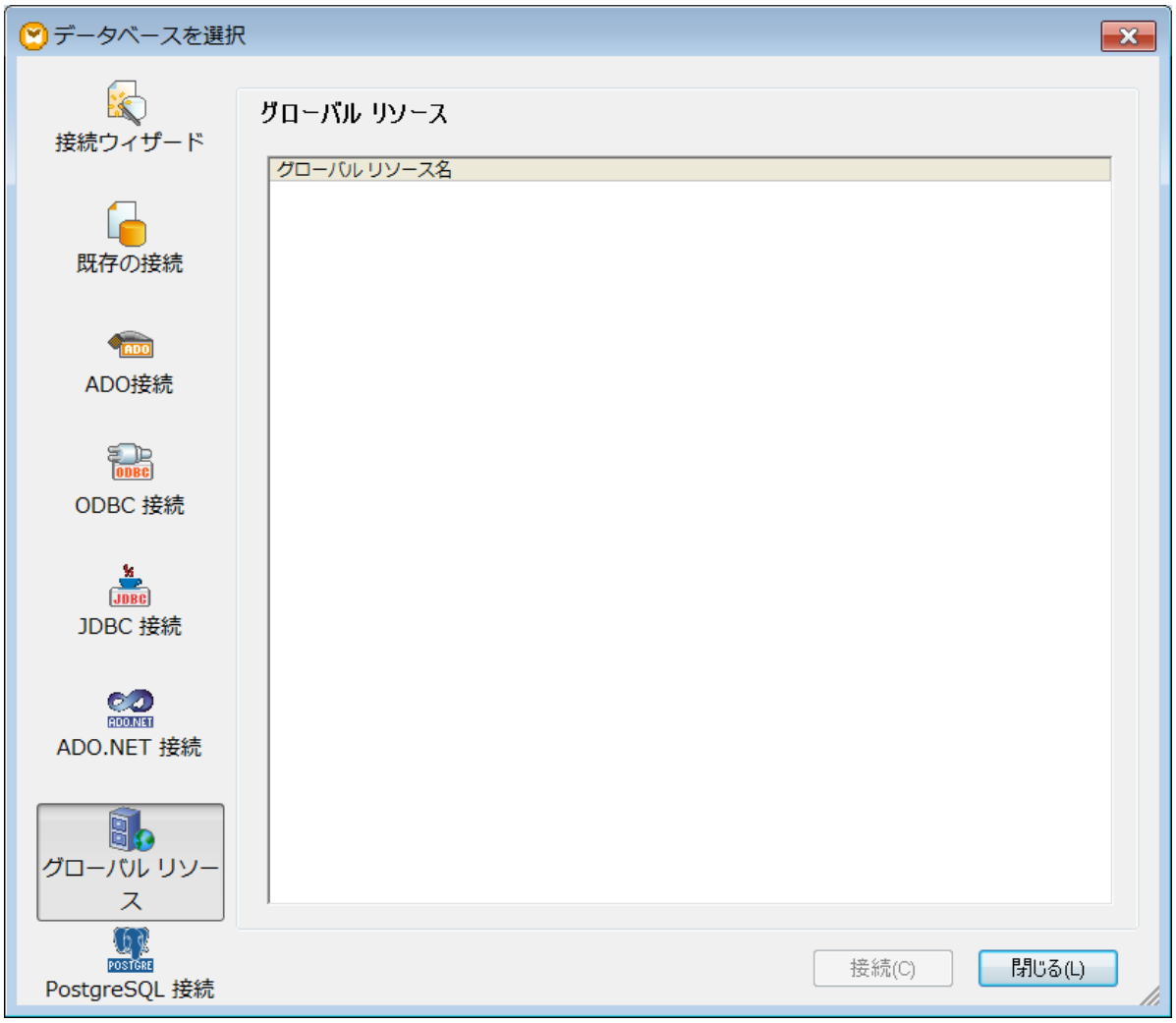
ステップ 1: グローバルリソース(データベースエイリアス)を作成する

1. 「**ツール**」メニューから「**グローバルリソース**」をクリックします。(または、**グローバルリソース** ツールバーボタンをクリックします)。
2. 「**追加 | データベース**」をクリックします。
3. 「**リソースエイリアス**」テキストボックスに名前を入力します(例えば、SourceDatabase)。
4. 「**データベースの選択**」を選択し、**Microsoft Access (ADO)** を選択し、開発データベースを参照します(<**マイドキュメント**>¥Altova¥MapForce2021¥MapForceExamples¥**altova.mdb**)。
5. 「**ファイルの追加**」 をクリックして、「ReleaseDatabase」という名前を与えます。
6. 「**データベースの選択**」をクリックして、**Microsoft Access (ADO)** を選択し、生産データベースを参照します(<**マイドキュメント**>¥Altova¥MapForce2021¥MapForceExamples¥Tutorial¥**altova.mdb**)。

ステップ 2: マッピング内のグローバルリソースを使用する

データベースエイリアスが再生されましたが、使用するためには、マッピングを変更する必要があります。

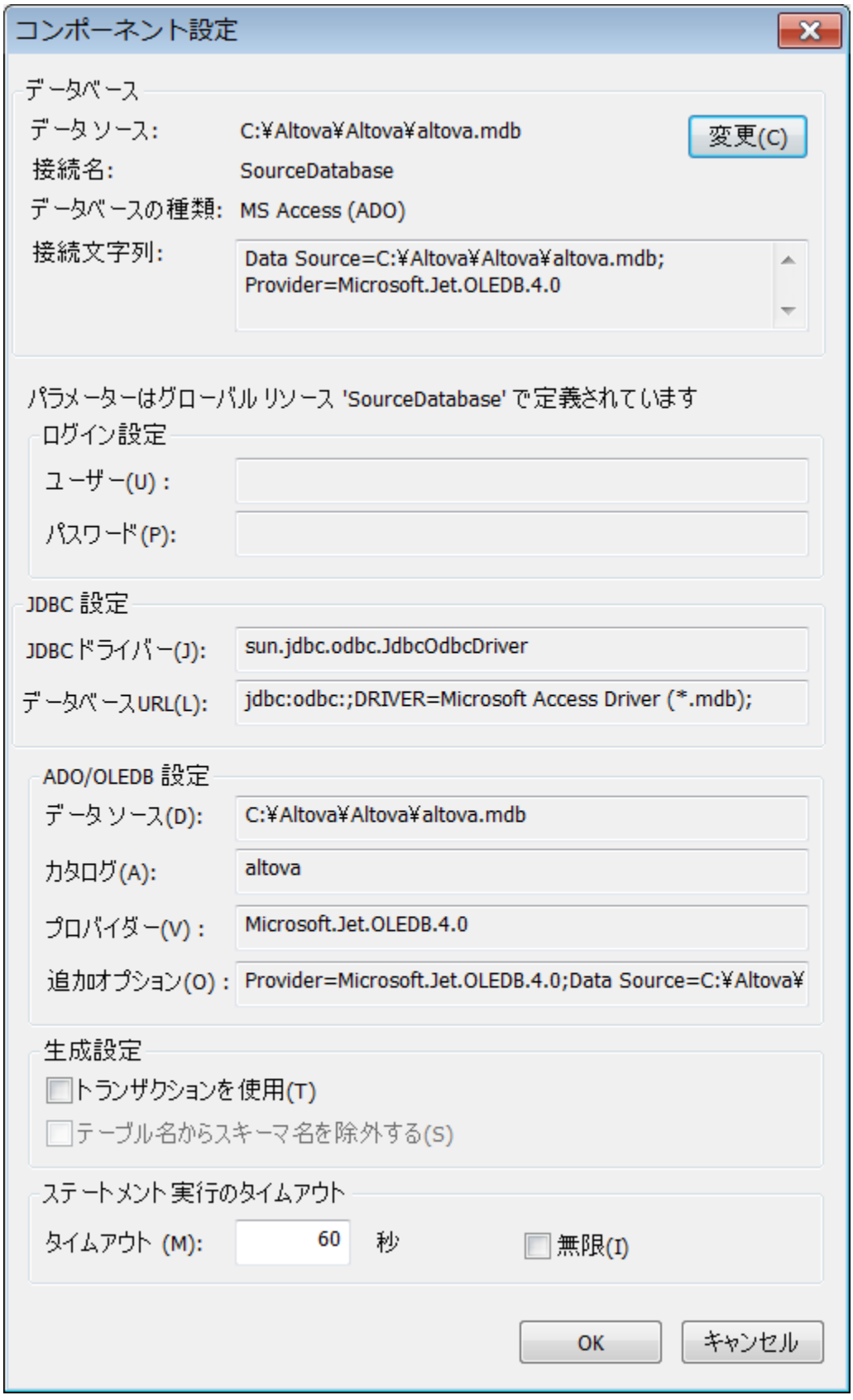
1. 次のマッピングを開きます: <**マイドキュメント>\Altova\MapForce2021\MapForceExamples\Tutorial\PersonDB.mfd**。
2. データベースコンポーネントを右クリックし、コンテキストメニューから「**プロパティ**」を選択します。
3. 「**変更**」をクリックして、作成されている「SourceDatabase」エイリアスを選択します。



4. 「**接続**」をクリックします。
5. データベースオブジェクトを選択するようにプロンプトされると、デフォルトのセクションをそのままにして、「OK」をクリックします。

メモ データベース接続がグローバルリソースとして設定されると、コンポーネント設定 ダイアログ上の設定は、下で示されるように灰色で表示されます。下で示される通り**グローバルリソース** ツールバーボタンをクリックして接続パラメーターを変更することができます。次も参照してください: グローバルリソースとしてのデータベース。

(C) 2015-2021 Altova GmbH


コンポーネント設定
データベース
データソース:
C:¥Altova¥Altova¥altova.mdb
変更(C)
接続名:
SourceDatabase
データベースの種類:
MS Access (ADO)
接続文字列:
Data Source=C:¥Altova¥Altova¥altova.mdb;
Provider=Microsoft.Jet.OLEDB.4.0
パラメーターはグローバル リソース 'SourceDatabase' で定義されています
ログイン設定
ユーザー(U) :
パスワード(P):
JDBC 設定
JDBCドライバー(J):
sun.jdbc.odbc.JdbcOdbcDriver
データベースURL(L):
jdbc:odbc:;DRIVER=Microsoft Access Driver (*.mdb);
ADO/OLEDB 設定
データソース(D):
C:¥Altova¥Altova¥altova.mdb
カタログ(A):
altova
プロバイダー(V) :
Microsoft.Jet.OLEDB.4.0
追加オプション(O) :
Provider=Microsoft.Jet.OLEDB.4.0;Data Source=C:¥Altova¥
生成設定
トランザクションを使用(T)
テーブル名からスキーマ名を除外する(S)
ステートメント実行のタイムアウト
タイムアウト (M):
60
秒
無限(I)
OK
キャンセル

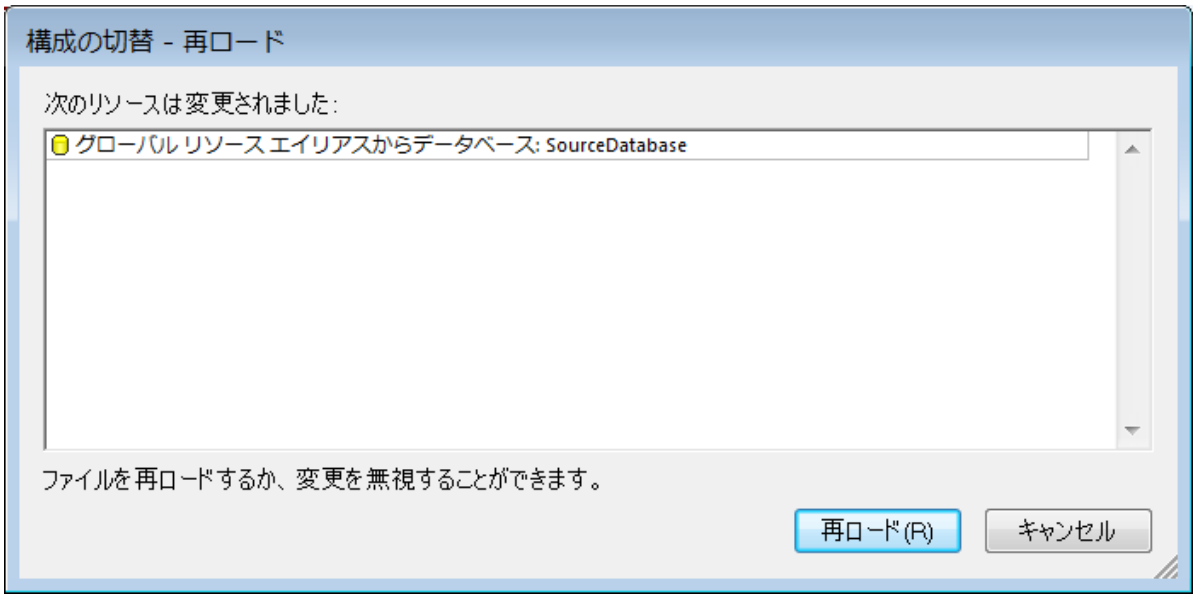
ステップ 3: 希望する構成を使用してマッピングを実行する

マッピングを実行する前に、希望するデータベースに簡単に切り替えることができます:

- 開発データベースからのデータを読み取るには、「**ツール**」メニューから「**アクティブな構成 | デフォルト**」をクリックします。
- 生産データベースからデータを読み取るには、メニューから「**アクティブな構成 | ReleaseDatabase**」をクリックします。

または、必要とされる構成を「**グローバルリソース**」ドロップダウンリストから選択します。

構成を切り替える場合、ダイアログボックスは、ソースデータベースがまもなく再ロードされることをプロンプトします:



メモ このサンプルで使用される両方のデータベースでは、類似しているデータが含まれており、マッピングを実行した後に生成される出力内には差異は存在しません。

(C) 2015-2021 Altova GmbH

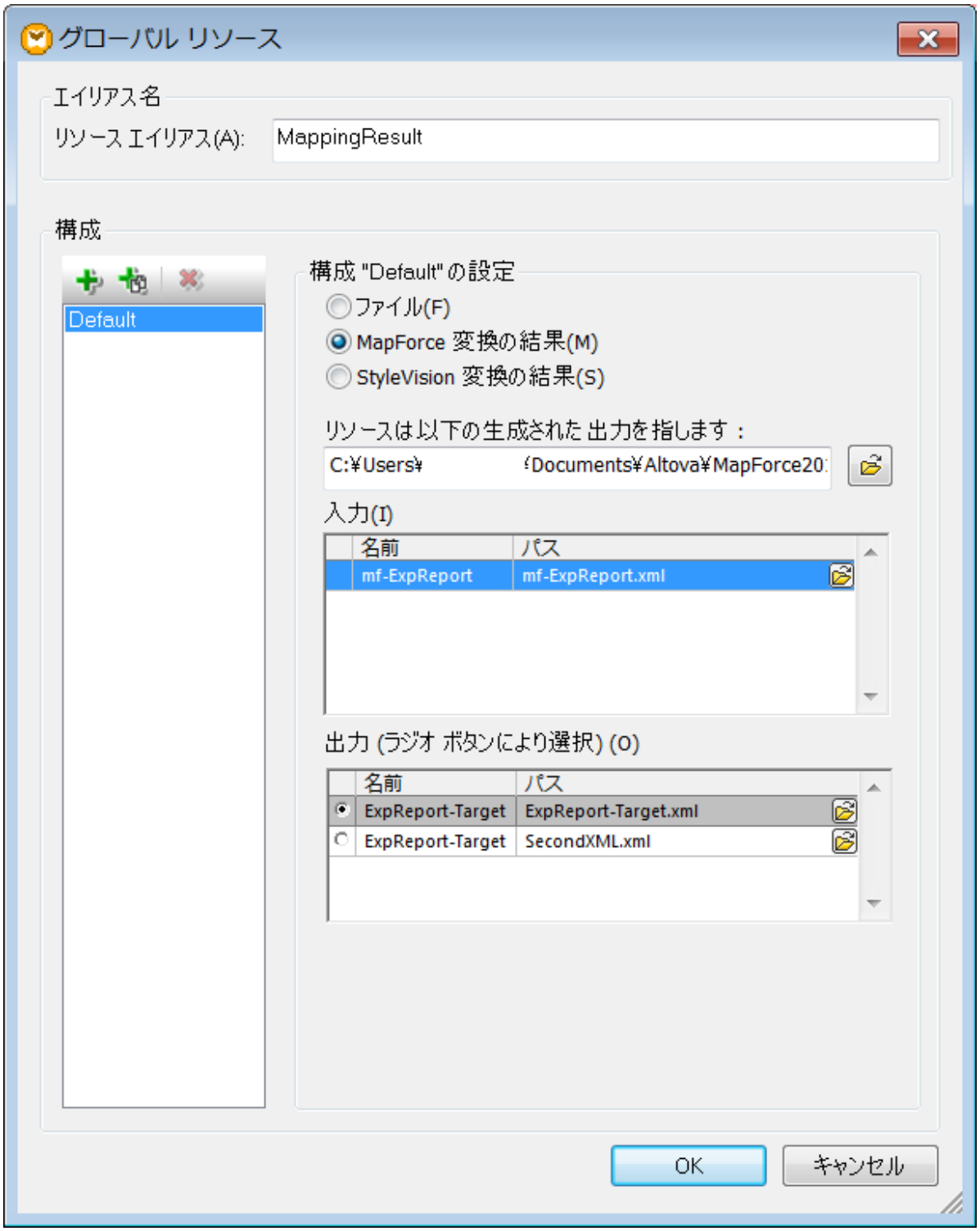
11.9 例: アプリケーションワークフローを作成する

このサンプルは、グローバルリソースを使用した、Altova MapForce とAltova XMLSpy 間のシンプルなワークフローの作成方法について説明しています。特に、MapForce マッピングの実行をXMLSpy から直接トリガーし、MapForce により生成されたマッピング出力をXMLSpy で開く方法について説明されています。これを可能にするには、下で説明されるように、型「変換の結果」のグローバルリソースを作成します。

ステップ 1: グローバルリソースを作成する

MapForce とXMLSpy の両方からこのステップを実行することができます。

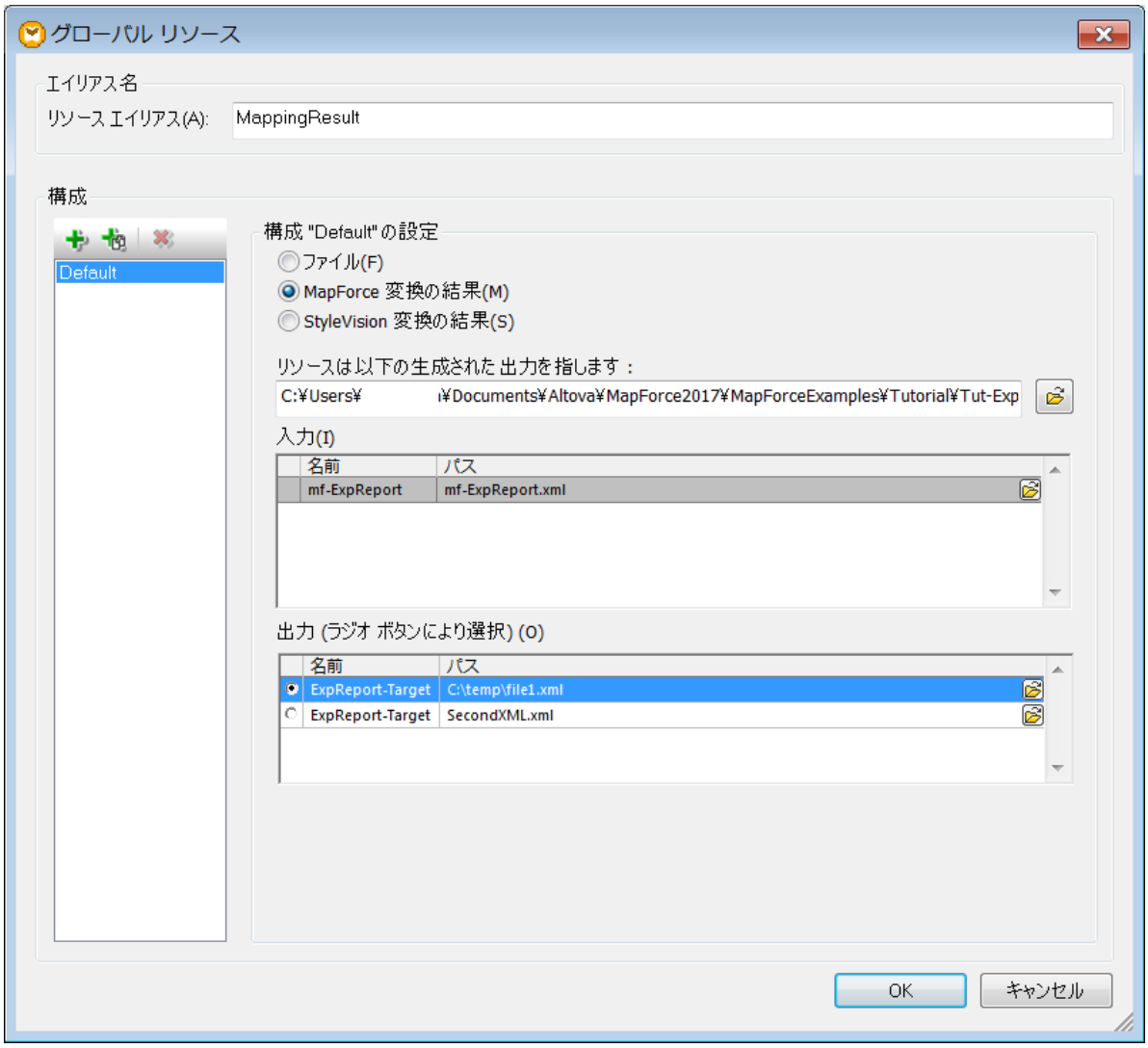
1. 「ツール」メニューから「**グローバルリソース**」をクリックします。(または、**グローバルリソース** ツールバーボタンをクリックします)。
2. 「**追加 | ファイル**」をクリックします。
3. 「**リソースエイリアス**」テキストボックスに名前を入力します(このサンプルでは、「MappingResult」)。
4. オプション「**MapForce 変換の結果**」を選択します。
5. 「**参照**」をクリックして、マッピング<**マイドキュメント**>**\Altova\MapForce2021\MapForceExamples\Tutorial\Tut-ExpReport-multi.mfd** を選択します。ダイアログボックス上で「入力」と「出力」セクションが作成されるまで待ってください。下で示されるとおり、このマッピングには、1つの入力と2つの出力が存在します。



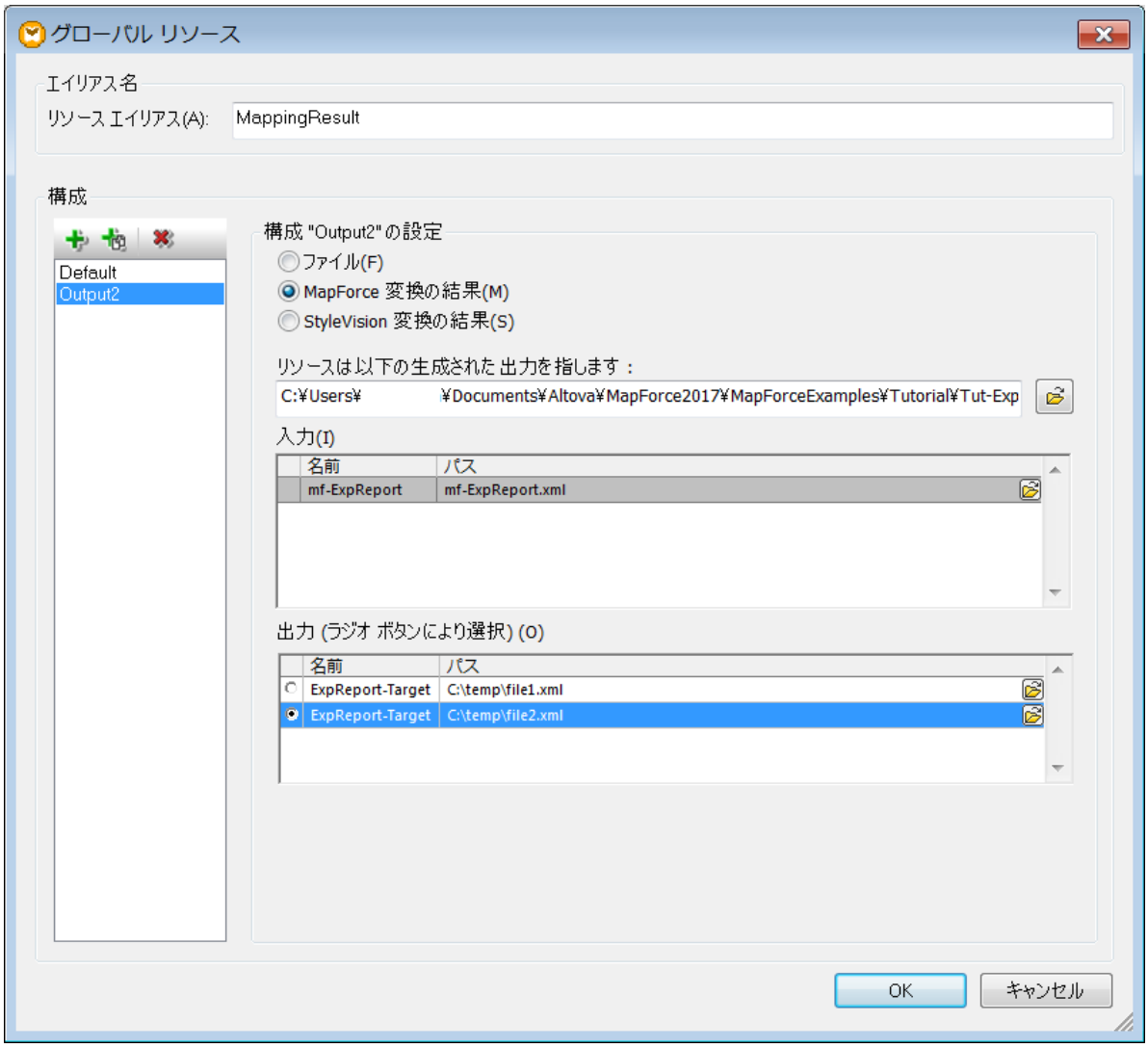
このサンプルのスコープのために、フォルダー **C:\temp** に2つのファイルをそれぞれ生成し、デフォルトのファイル名を変更します。この目的を達成するには、以下のとおり、希望される出力のために構成を作成します:

1. 「出力」の下から、2番目の出力の横の「参照」をクリックして、**C:\temp\file1.xml** 目的のファイル名として入力します。これは最初の出力ファイルがトリガーされると作成されるデフォルトの構成です。

(C) 2015-2021 Altova GmbH



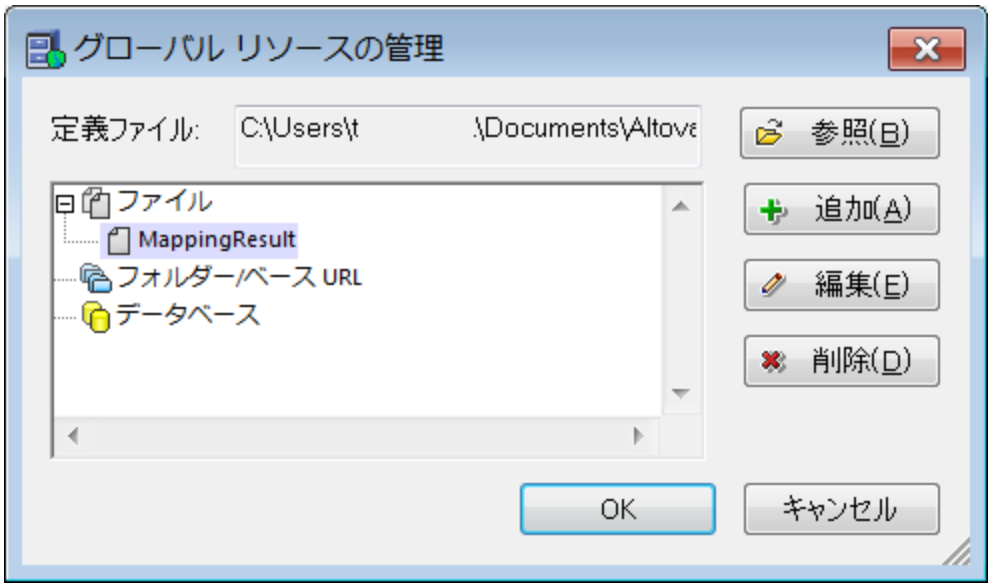
2. 「**構成をコピーとして追加する**」 をクリックし、 新しい構成のために名前を入力します（このサンプルでは、"Output2"）。
3. 「出力」の下から、2番目の出力の横の「**参照**」をクリックして、**C:\temp\file2.xml** 目的のファイル名として入力します。これは、2番目の出力ファイルを生成する代替の構成です。



ステップ 2: ワークフローをトリガーする

前のステップで作成されたグローバルリソースはXMLSpy で以下のように使用することができます。

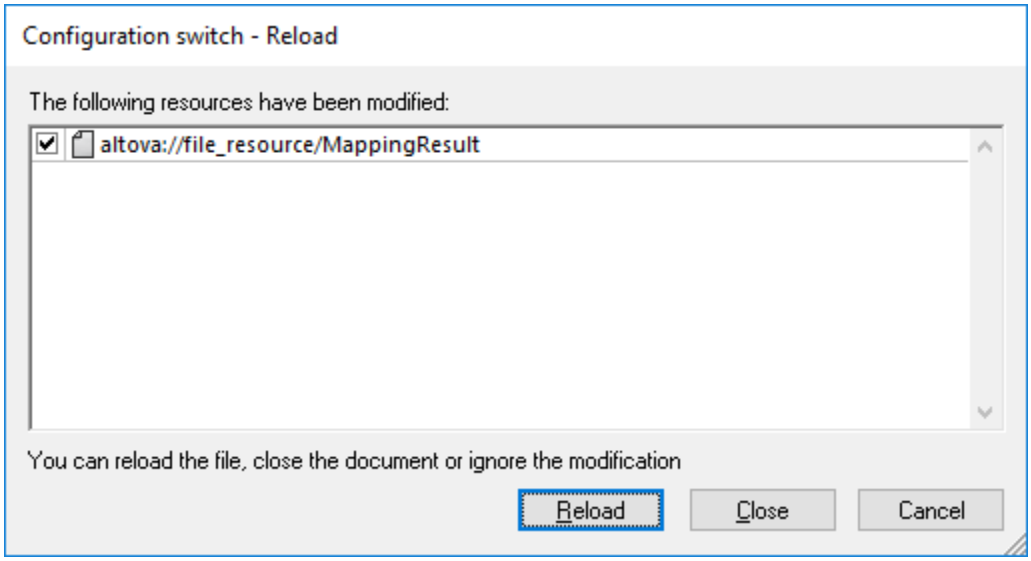
1. XMLSpy を作動します。
2. 「ツール」メニューから グローバルリソース をクリックします。
3. 「ファイル」グループ内で、作成されている「MappingResult」グローバルリソースを選択し、「表示」をクリックします。



マッピングを実行し、デフォルトのでの出力（**file1.xml**）を作成し、XMLSpy のメインペインにロードします。ファイルは、以下の名前で保存されます: **C:\temp\file1.xml**.

マッピングをトリガーするには、代替の構成を使用して以下のように実行します:

1. 「**ツール**」メニューから「**アクティブな構成 | Output2**」をクリックします。



この結果、代替の出力ファイルはXMLSpy のメインペインにロードされます。ファイルは以下の名前で保存されます: **C:\temp\file2.xml**。

12 MapForce のカスタマイズ

12.1 MapForce オプションの変更

MapForce 内の一般および他の優先順位を変更する方法は、以下の通りです:

- 「ツール」メニューから「オプション」をクリックします。

使用することのできるオプションは、以下に表示されているとおりです。

全般

このページで使用することのできるページ内の設定は、以下のとおりです:

**ロゴの表示	起動時に表示**	MapForce が起動されると、イメージ（スプラッシュスクリーン）を表示、または非表示にします。
背景をグラデーションで表示	マッピングペイン内の背景をグラデーションで表示する機能を有効化、または無効化します。
表示する注釈を N 列に制限する	このオプションは注釈をサポートするコンポーネントに対して適用されます（例えば、XML スキーマ、EDI）。注釈テキストに複数の行が含まれる場合、このオプションを有効化すると、コンポーネント上の最初の N 行のみが表示されます。N は指定することのできる値です。この設定は、コンポーネント内で表示される SELECT ステートメントにも適用されます。
名前のエンコード	新規コンポーネントのためにデフォルトの文字の円コードを設定します。この設定は、各コンポーネントのために個別に変更することができます。次を参照してください: コンポーネント設定の変更.
実行タイムアウトの使用	「出力」ペイン内でマッピングの結果をプレビューする際に実行のタイムアウトを設定します。
一時的なファイルの出力を生成する	このオプションが設定されていると、マッピングの結果をプレビューする際に生成される出力は、一時的なファイルに書き込まれます（これはデフォルトのオプションです）. 出力ファイルパスにまだ存在しないフォルダーが含まれる場合、MapForce がこれらのファイルを作成します。 警告: 実行のためにマッピングをサーバーにデプロイする場合、パス内のディレクトリはサーバー上で存在する必要があり、それ以外の場合、実行エラーが発生します。サーバー実行のためにマッピングを準備するも参照してください。
最後の出力ファイルに直接書き込む	このオプションが設定されていると、マッピングの結果をプレビューする際に生成される出力は、実際のファイルに書き込まれます 出力ファイルパスにまだ存在しないフォルダーが含まれる場合、マッピングエラーが発生します。 警告: このオプションは更なる確認をせず、既存の出力ファイルを上書きします。
N 文字のステップでテキストを表示する	大きな XML とテキストファイルを生成するマッピングをプレビューする際に、出力 ペイン内に表示されるテキストの最大のサイズを指定します。出力テキストがこの値を超えると、「更にロードする」ボタンをクリックして次のチャンクをロードする必要があります。詳細に関しては、次を参照してください: 出力のプレビュー。

編集

このページで使用することのできる設定は、以下のとおりです:

マウスのドラッグ時のコンポーネントの入れる	マウスでドラッグする際にコンポーネントまたは関数が他のコンポーネントと共に整列するかを指定します。コンポーネントの整列 を参照。
スマートコンポーネントの削除	有効化されると、**スマートコンポーネントの削除** オプションは、削除された接続を "記憶" します。コンポーネントの削除後、接続を保持する を参照。

メッセージ

「このメッセージを表示しない」オプションにより以前に無効化されたメッセージの通知を再有効化します。

生成

プログラムのコード生成中に適用できるIDE とコンパイラーの設定を定義することが有効化できます。コードジェネレーターのオプション および MapForce Server 実行ファイルにマッピングをコンパイルする をそれぞれ参照してください。

XBRL

一般 XBRL オプション を参照してください。

デバッガー

デバッガーの設定 を参照してください。

データベース

DB Query タブ内のデータベースをクエリするとき適用することのできる設定を定義することができます（次を参照してください： データベースの参照とクエリ）。

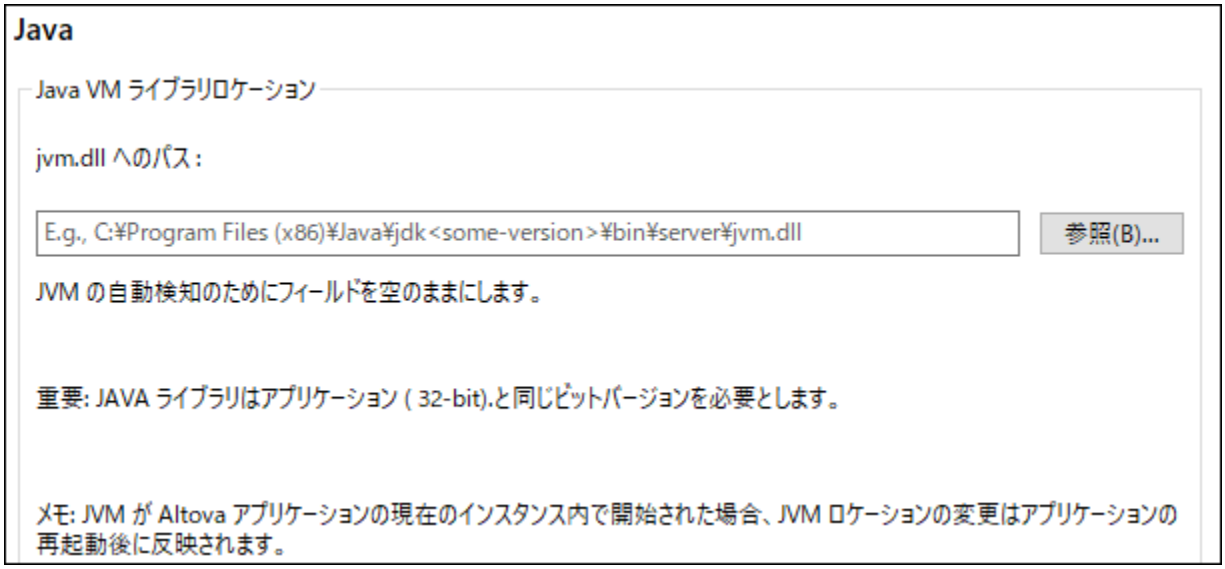
ネットワークプロキシ

ネットワークプロキシの設定を参照してください

12.1.1 Java 設定

Java タブ上では、ファイルシステム上で Java VM（仮想マシン）へのパスを任意で入力することができます。カスタム Java VM パスの追加は常に必要なわけではありません。デフォルトで、MapForce は、W indows レジストリと JAVA_HOME 環境変数を（この順序で）読み取ることにより、Java VM パスを自動的に影響しようとします。このダイアログボックスに追加されるカスタムパスは、自動的に検知される VM パスより先優先順位を与えられます。

カスタム Java VM パスを追加する必要がある場合、例えば、インストーラを持たず、（例えば、Oracle の OpenJDK などの）レジストリエントリを作成する必要のない Java 仮想マシンを使用している場合、カスタム Java VM パスを追加する必要がある場合があります。MapForce により自動的に検知された Java VM パスを何らかの理由でオーバーライドする場合、このパスを設定する必要があります。



以下の点に注意してください:

- Java VM パスは Altova デスクトップと(サーバーではない) アプリケーションにより共有されています。この結果、アプリケーション内で変更されると、他の Altova アプリケーションに自動的に適用されます。
- パスは JDK がインストールされているディレクトリに相対する **\bin\server** または **\bin\client** ディレクトリからの **jvm.dll** ファイルを指す必要があります。
- MapForce プラットフォーム(32-ビット、64-ビット) が JDK と同じである必要があります。
- Java VM パスの変更後、新規の設定を反映するために MapForce を再起動する必要があります。

Java VM パスの変更は以下のエリアに影響を与えます:

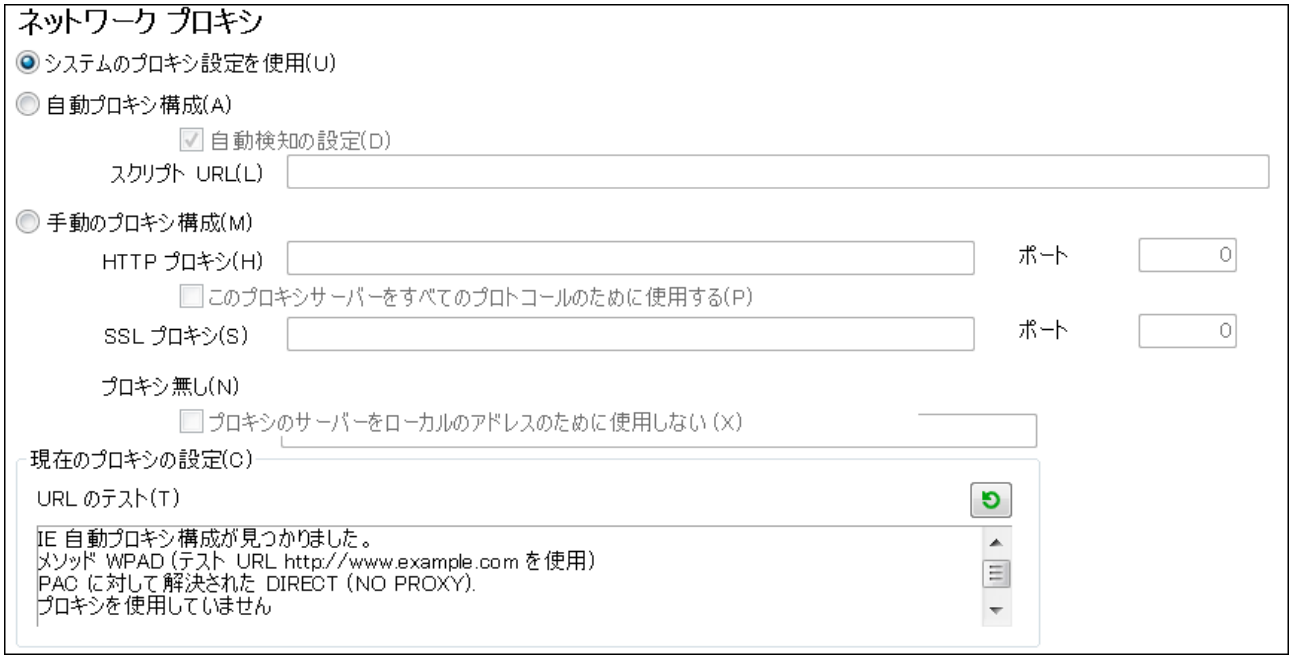
- JDBC 接続性
- XSLT/XPath のための Java 拡張関数

この設定は Java コード生成に影響を与えません。

12.1.2 ネットワークプロキシ設定

ネットワーク プロキシ セクションでは、カスタムのプロキシの設定を構成することができます。(XML 検証の目的のために) MapForce のインターネットへの接続方法に影響します。デフォルトでは、MapForce はシステムのプロキシの設定を使用します、ですから、多くの場合プロキシの設定を変更する必要はありません。必要な場合、代替のネットワークプロキシを下のオプションを使用して設定することができます。

メモ ネットワークプロキシ設定は、Altova MissionKit アプリケーション間で共有されています。結果、1 つのアプリケーション内で設定が変更されると、自動的に他の全てのアプリケーションに影響を与えます。



システムプロキシの設定の使用

*システムプロキシ*設定を介して構成可能な *インターネット エクスプローラー*(IE) 設定を使用します。**netsh.exe winhttp** を介して構成される設定が必要とされます。

自動プロキシの構成

以下のオプションを使用することができます:

- *自動検知の設定:* DHCP または DNS を使用して W PAD *スクリプト* (**http://wpad.LOCALDOMAIN/wpad.dat**) を検索し、*プロキシセットアップのために*このスクリプトを使用します。
- *スクリプト URL:* *プロキシセットアップのために*使用されるプロキシ自動構成 (**.pac**) *スクリプト*に対する HTTP URL を指定します。
- *再ロード:* 現在の自動プロキシ構成をリセットして再ロードします。*このアクション*には W indows 8 または以降が必要とされ、30 秒程の時間が必要です。

手動のプロキシの構成

ホスト名とポートを対応する製品の*プロキシのために*手動で指定します。サポートされるスキームはホスト名に含まれている場合があります (例: **http://hostname**)。プロキシがスキームをサポートする場合、対応するプロトコールと同じである必要はありません。

以下のオプションを使用することができます:

- *このプロキシサーバーをすべてのプロトコールのために使用する:* 全てのプロトコルのために HTTP プロキシのホスト名とポート使用します。
- *プロキシ無し:* セミコロン (**;**) により区別されているプロキシを使用しないホスト名、ドメイン名、または、ホストのための IP アドレスのリスト。IP アドレスは切り捨てられず、IPv6 アドレスは角かっこで囲まれる必要があります (例: **[2606:2800:220:1:248:1893:25c8:1946]**)。ドメイン名は、ドットと共に開始される必要があります (例: **.example.com**).

- *プロキシのサーバーをローカルのアドレスのために使用しない*：チェックされている場合、*プロキシ無しリストのために* `<local>` を追加します。このオプションが選択されている場合、次の場合、プロキシは使用されません：(i) `127.0.0.1`、(ii) [::1]、(iii) (`.`) ドット文字を含んでいないホスト名すべて。

メモ プロキシサーバーが設定されており、*マッピング* を Altova FlowForce Server にデプロイする場合、*ローカルアドレスのためにプロキシサーバーを使用しない* を選択する必要があります。

現在のプロキシの設定

プロキシの検知の詳細なログを提供します。*URL のテスト* フィールドの右の「**更新**」ボタンを使用して更新することができます（例、URL のテスト を変更する場合、または、プロキシの設定が変更された場合）。

- *URL のテスト*: A test URL のテストを使用して、どのプロキシが特定の URL ために使用されているかを確認することができます。URL を使用して、I/O はされません。プロキシの自動構成が選択されている場合、このフィールドは空にしておく必要があります（*システムプロキシの設定の使用*、または、*自動プロキシの構成* を使用して）。

(C) 2015-2021 Altova GmbH

12.2 キーボードのショートカット

デフォルトで MapForce は以下のキーボードのショートカットを提供しています:

F1	ヘルプメニュー
F2	次のブックマークへ(出力ウィンドウにて)
F3	次を検索
F10	メニューバーをアクティベート
Num +	現在のアイテムノードを展開
Num -	アイテムノードを縮退
Num *	現在のアイテムノード以下を全て展開
CTRL + TAB	開かれているマッピング間の切り替え
CTRL + F6	開かれているウィンドウをサイクル
CTRL + F4	アクティブなマッピングドキュメントを閉じる
Alt + F4	MapForce を閉じる
Alt + F, F, 1	最後に閉じられたファイルを開く
Alt + F, T, 1	最後に閉じられたプロジェクトを開く
CTRL + N	新規ファイル
CTRL + O	ファイルを開く
CTRL + S	ファイルの保存
CTRL + P	ファイルの印刷
CTRL + A	すべて選択
CTRL + X	切り取り
CTRL + C	コピー
CTRL + V	貼り付け
CTRL + Z	元に戻す
CTRL + Y	やり直し
Del	コンポーネントを削除(プロンプト付き)
Shift + Del	コンポーネントを削除(プロンプトなし)
CTRL + F	検索
F3	次を検索
Shift + F3	前を検索
矢印キー	
(上 / 下)	コンポーネント内にあるアイテムを選択
Esc	編集の破棄 / ダイアログボックスを閉じる
Return	選択の確定
出力ウィンドウホットキー	
CTRL + F2	ブックマークの挿入/削除
F2	次のブックマークへ
SHIFT + F2	前のブックマークへ
CTRL + SHIFT + F2	全てのブックマークを削除
ズーム ホットキーhotkeys	
CTRL + マウスホイール前	ズームイン

CTRL + マウスホイール後	ズームアウト
CTRL + 0 (ゼロ)	ズームのリセット

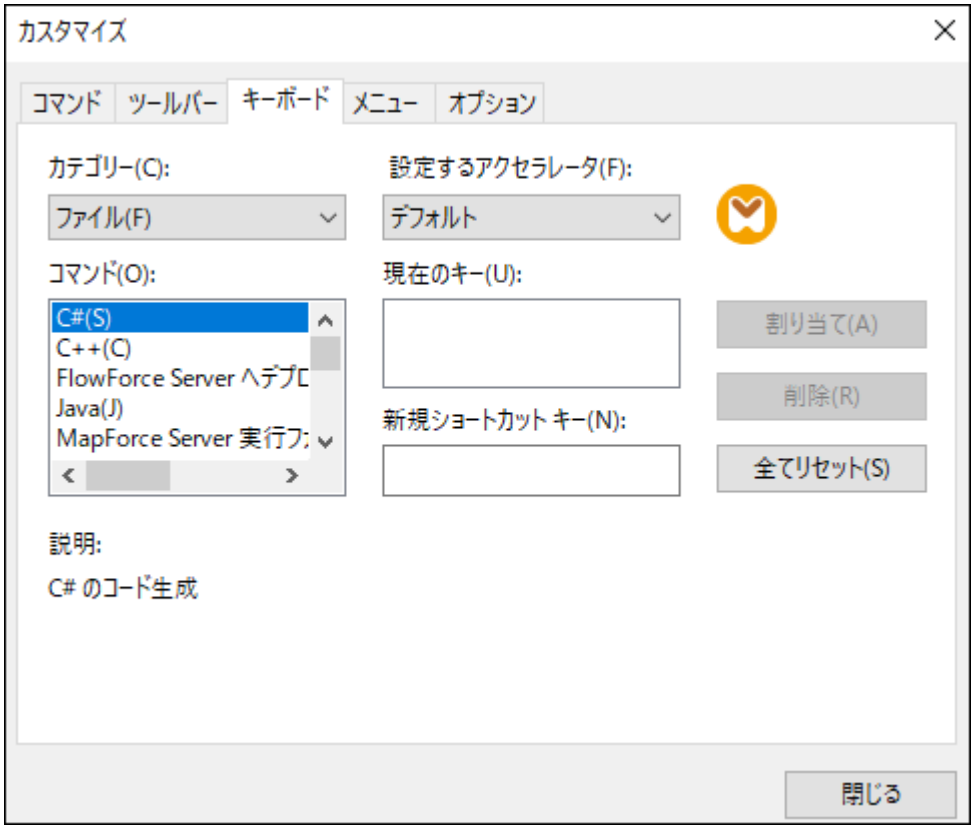
12.2.1 ショートカットのカスタム化

MapForce 内のキーボードのショートカットを以下のように定義または変更することができます:

1. 「**ツール**」メニューから「**カスタマイズ**」 をクリックします。
2. 「**キーボード**」タブをクリックします。

コマンドに新規ショートカットを割り当てる

1. 「**ツール | カスタマイ**」コマンドを選択して、キーボードタブをクリックします。
2. 「**カテゴリー**」コンボボックスをクリックして、メニュー名を選択します。
3. コマンドリストボックス内で新しいショートカットを割り当てる**コマンド**を選択します。
4. 「**新規ショートカットキー**」をクリックします: テキストボックスから、コマンドを有効化するショートカットキーを押します。



ショートカットはテキストボックス内にすぐに表示されます。ショートカットが以前に割り当てられている場合、その関数がテキストボックスの下に表示されます。

5. 「**割り当**」てボタンをクリックして、ショートカットを割り当てます。
現在のキーリストボックスにショートカットが表示されます。
(新規ショートカットキーテキストボックス内のエントリをクリアするには、コントロールキーを押します **CTRL**、**ALT** または **SHIFT**)。

(C) 2015-2021 Altova GmbH

ショートカットを元に戻す、または削除する

1. 現在のキーリストボックス内から削除するショートカットをクリックします。
2. 「**削除**」ボタンをクリックします。
3. 「**閉じる**」ボタンクリックして、確認します。

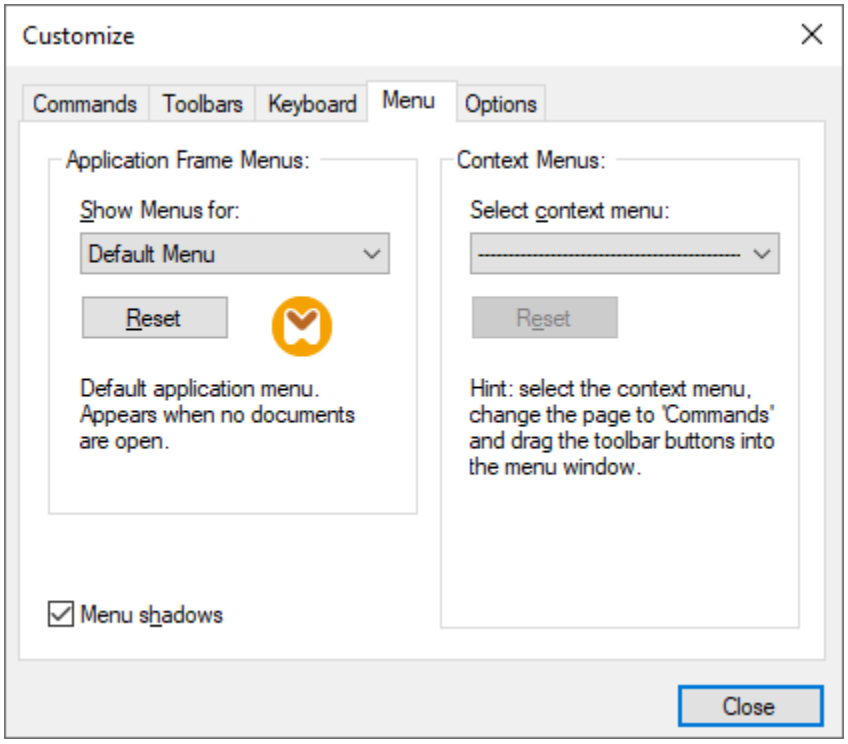
メモ 設定するアクセラレーターには現在関数が存在しません

12.3 メニューをカスタム化する方法

標準 MapForce メニューおよび、例えば、コマンドを追加、変更、および削除するために、コンテキストメニューをカスタム化することができます。メニューのカスタム化をデフォルトの状態にリセットすることができます。

メニューをカスタム化する方法

- 「ツール」メニューから、「オプション」をクリックして、「メニュー」をクリックします。



メニューをカスタム化する方法

[デフォルトメニュー] バーはメインウィンドウ内でドキュメントが開かれていない場合に表示されます。[MapForce デザイン] メニューバーは1つまたは複数のマッピングが開かれていると表示されるメニューバーです。各メニューバーを個別にカスタム化することができ、カスタム化の変更はお互いに影響を与えません。

メニューバーをカスタム化するには「メニューの表示」ドロップダウンリストから選択します。[コマンド] タブをクリックして [コマンド] リストからコマンドをメニューバーまたはメニューにドラッグします。

メニューからコマンドを削除しリセットする方法

メニュー内のメニューまたはコマンドを全て削除するには以下を行います:

1. **メニューの表示** ドロップダウンリストから以下の1つを選択します:
   - **デフォルトのメニュー**(これはドキュメントが開かれていない場合に使用可能なメニューを表示します)
   - **MapForce デザイン**(これはドキュメントが開かれていない場合に使用可能なメニューを表示します)

(C) 2015-2021 Altova GmbH

2. カスタム化ダイアログが開かれている状態で(i) アプリケーションのメニューバーから削除するメニュー、または(ii) これらのメニューのいずれかから削除するコマンドを選択します。
3. (i) メニューからメニューバーまたはメニューコマンドからドラッグし(ii) メニューまたはメニューコマンドを右クリックし[削除]を選択します。

[メニューの表示]ドロップダウンリストから選択し[リセット]ボタンをクリックしてメニューを元の状態にリセットすることができます。

アプリケーションのコンテキストメニューをカスタマイズする方法

コンテキストメニューはアプリケーションのインターフェイス内の特定のオブジェクトを右クリックすると表示されるメニューです。これらのコンテキストメニューのそれぞれは以下を行ってカスタム化することができます:

1. [コンテキストメニューの選択]ドロップダウンリストからコンテキストメニューを選択します。コンテキストメニューがポップアップされます。
2. [コマンド]タブをクリックします。
3. [コマンド]リストボックスからコマンドをコンテキスメニューにドラッグします。
4. コンテキストメニューからコマンドを削除するために、コンテキストメニュー内でそのコマンドを右クリックして[削除]を選択します。代わりに、コマンドをコンテキストメニューからドラッグします。

[コンテキストメニューの選択]ドロップダウンリスト 内で選択し[リセット]ボタンをクリックしてコンテキストメニューを初期のインストールの状態にします。

メニューとメニューの影

全てのメニューに影を与えるために[メニューの影]チェックボックスを選択します。

12.4 カタログファイル

MapForce はOASIS XML カタログメカニズムのサブセットをサポートします（https://www.oasis-open.org/committees/entity/spec-2001-08-06.html）。カタログメカニズムにより MapForce が共通に使用されるDTD と XML スキーマ（およびスタイルシートと他のファイル）をパブリックURI から解決する代わりにローカルフォルダからを抽出することができるようになります。これは全体的な処理スピードを増やしオフライン（すなわちネットワークに接続されていない状態）での作業を可能にし、ドキュメントのポータビリティを改善します。

カタログのしくみ

カタログは一般的にパブリックDTD または通常は、ローカルファイルパス）ローカルURI へのスキーマレファレンスをリダイレクトするために使用されます。これを達成するために、XML フォーマットのカタログファイルはパブリックスキーマURI とローカルURI の間のマッピングを定義します。MapForce がXML ドキュメントを解析する都度、カタログファイル内部のスキーマURI（または、適用可能の場合、DTD のパブリックまたはシステム識別子）が最初に検索されます。マッピングがカタログファイル内で見つかった場合、そのレファレンスが使用されスキーマはローカルファイルから読み取られます。カタログファイル内でマッピングが見つからない場合 XML ドキュメントのURI はそのまま解決されます。

例えば、以下のXML ファイルはMapForce により処理される必要があると仮定します。

```
<?xml version="1.0" encoding="UTF-8"?>
<Articles xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="Articles.xsd">
   <Article>
      <Number>1</Number>
      <Name>T-Shirt</Name>
      <SinglePrice>25</SinglePrice>
   </Article>
</Articles>
```

(MapForce が既知の通り) **catalog.xml** ファイルがローカルディレクトリに存在し、以下のラインが含まれると想定します。この点については以下で詳しく説明されています:

```
<catalog>
  <!--...-->
  <uri name="http://www.w3.org/2001/XMLSchema-instance.xsd" uri="files/XMLSchema-
instance.xsd"/>
  <!--...-->
</catalog>
```

XML ファイルを解析する場合、MapForce はカタログファイル内でスキーマレファレンス `http://www.w3.org/2001/XMLSchema-instance.xsd` のための一致を検知します。この結果スキーマはカタログファイルに対して相対的なローカルパス）`files/XMLSchema-instance.xsd` からロードされますカタログファイルファイル内でマッピングが見つからない場合、スキーマは`http://www.w3.org/2001/XMLSchema-instance` からロードされます。

ルートカタログ

MapForce が開始されると、**RootCatalog.xml** と呼ばれるファイルを"Program Files" ディレクトリからロードします。**RootCatalog.xml** はそれぞれが`nextCatalog` 要素内にあるカタログファイルのリストを含んでいます。これらのカタログファイルはルックアップされ内部にあるURI は指定されているマッピングに従いMapForce により解決されます。

(C) 2015-2021 Altova GmbH

```
<?xml version="1.0" encoding="UTF-8"?>
<catalog xmlns="urn:oasis:names:tc:entity:xmlns:xml:catalog"
xmlns:spy="http://www.altova.com/catalog_ext"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="urn:oasis:names:tc:entity:xmlns:xml:catalog
Catalog.xsd">
   <nextCatalog catalog="%PersonalFolder%/Altova/%AppAndVersionName
%/CustomCatalog.xml"/>
   <!-- Include all catalogs under common schemas folder on the first directory level
-->
   <nextCatalog spy:recurseFrom="%AltovaCommonFolder%/Schemas" catalog="catalog.xml"
spy:depth="1"/>
   <nextCatalog catalog="CoreCatalog.xml"/>
</catalog>
```

RootCatalog.xml

上記のリスティングで、以下のカタログがルックアップのためにリストされていることに注意してください

- **CustomCatalog.xml** は自身のマッピングを作成することができるファイルです。このファイルは以下のディレクトリ内にあります: **C:\users\<name>\Documents\Altova\MapForce2021**。Altova 構成済みのカタログファイルで既にカスタムスキーマのために対処されていない場合（次のポイントを参照してください）**CustomCatalog.xml** にマッピングを追加することができます。
- **%AltovaCommonFolder%/Schemas** ディレクトリからの複数の**catalog.xml** ファイル。各 **catalog.xml** ファイルは（SVG、DITA、DocBook、W SDL などの）特定のスキーマのディレクトリの内部にあり、それぞれは対応するスキーマのローカルで保存されたコピーを指す URI へのパブリックおよびまたはシステム識別子をマップします。
- **CoreCatalog.xml** にはスキーマをロケートするための Altova 固有のマッピングが含まれています。このファイルは MapForce [Program Files] 内にあります。

以下の点に注意してください:

- **CustomCatalog.xml** を変更する場合 サポートされる要素を使用してください。エラーを引き起こす可能性があるため既存のマッピングを複製しないでください。
- **%AltovaCommonFolder%\Schemas\schema** フォルダー内の **catalog.xml** ファイルは古い XML スキーマ仕様を実装する DTD へのレファレンスを含んできます。XML スキーマドキュメントをこれらのスキーマに対して検証してはなりません。古い XML スキーマ仕様はこのようなドキュメントと作業するための対応するスキーマ URI を効果的に解決する機能 をMapForce に提供するためにのみ含まれていました。

上記の一部のディレクトリパスは環境関数の助けを得て表示されています。レファレンステーブルに関しては 環境関数を参照してください。

サポートされる要素

CustomCatalog.xml 内でエントリを作成するには、以下にリストされる要素のみを使用して下さい。OASIS XML カタログ仕様の他の要素はサポートされていません。

要素	属性	例
public	• `systemId` はリソースのパブリック識別子を指定します。 • `uri` URI レファレンスを指定します（例えば、ローカルディレクトリに対する相対的なパス）	`<public publicId="-//W3C//DTD XMLSCHEMA 200102//EN" uri="files/XMLSchema.dtd"/>`
system	• `systemId` はリソースのシステム識別子を	`<system`

要素	属性	例
	指定します。 • uri URI レファレンスを指定します（例えば、ローカルディレクトリに対する相対的なパス）	systemId="http://www.w3.org/2009/XMLSchema/datatypes.dtd" uri="files/datatypes.dtd"/>
uri	• name はURI レファレンスを指定します • Uri は（例えば、ローカルファイルに対する相対的なパスなど）代替 URI レファレンスを指定します	<uri name="http://www.w3.org/2009/XMLSchema/XMLSchema.xsd" uri="files/XMLSchema.xsd"/>
rewriteURI	• uriStartString は再書き込みする URI の開始部分を指定します。 • Uri 置換文字列を指定します（例えば、ローカルディレクトリに対する相対的なパス）	<rewriteURI uriStartString='http://www.altova.com/schemas/svg/' rewritePrefix='files/'/>
rewriteSystem	• systemIdStartString は再書き込みするシステムの識別子の開始部分を指定します。 • rewritePrefix 置換文字列を指定します（例えば、ローカルディレクトリに対する相対的なパス）	<rewriteSystem systemIdStartString='http://www.altova.com/schemas/svg/' rewritePrefix='files/'/>

public、**system**、および**uri** 要素は相対的な URI が解決されるベース URI を指定する xml:base 属性を取ることができます。詳細に関してはXML カタログ仕様（http://www.oasis-open.org/committees/entity/spec-2001-08-06.html）を参照してください。

環境変数

下のテーブルはW indows 上での多種のシステムロケーションへのパスを指定する **nextCatalog** 要素内でサポートされるすべての環境変数をリストしています。

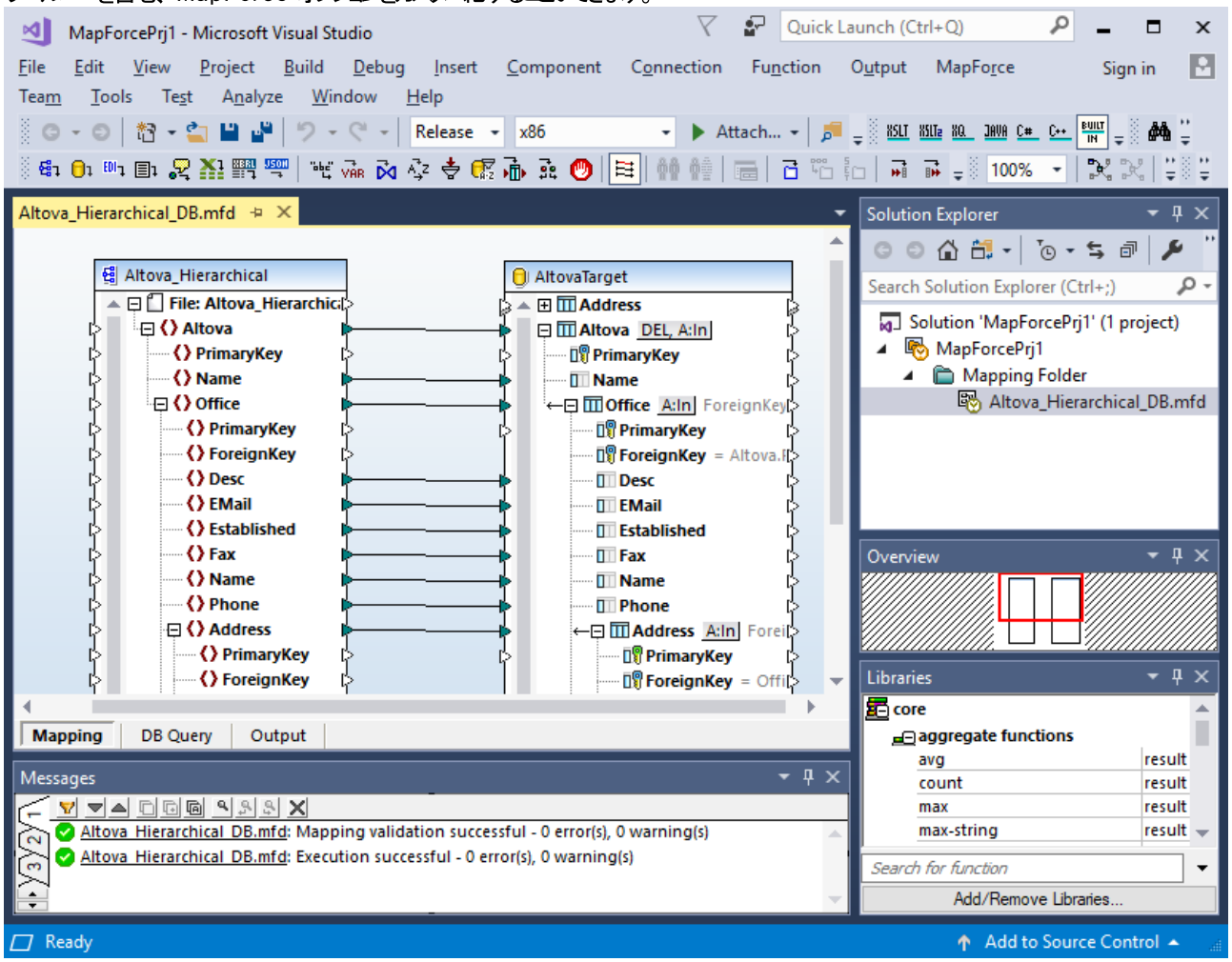
%AltovaCommonFolder%	すべてのAltova プログラムで共通のファイルを保管するために使用されているディレクトリへのフルパス。オペレーティングシステムとMapForce（32-ビットまたは64-ビット）のプラットフォームにより、パスはC:\Program Files\Altova\Common2021 またはC:\Program Files (x86)\Altova\Common2021 になります。
%DesktopFolder%	デスクトップ上にファイルオブジェクトを保管するために使用されるディレクトリへのフルパス。典型的なパスはC:\Users\Username\Desktop です。
%ProgramMenuFolder%	ユーザーのプログラムグループを含む自身がファイルシステムのディレクトリであるディレクトリへのフルパス。典型的なパスはC:\Users\Username\AppData\Roaming\Microsoft\Windows\Start Menu\Programs です。
%StartMenuFolder%	ユーザーの起動メニューアイテムを含むディレクトリへのフルパス。典型的なパスはC:\Users\Username\AppData\Roaming\Microsoft\Windows\Start Menu です。
%StartUpFolder%	ユーザーの起動プログラムグループに対応するディレクトリへのフルパス。典型的なパスはC:\Users\Username\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup です。

(C) 2015-2021 Altova GmbH

%TemplateFolder%	ドキュメントテンプレートのための共通レポジトリとしての役割を果たすディレクトリへのフルパス。典型的なパスは C:\Users\username\AppData\Roaming\Microsoft\Windows\Templates です。
%AdminToolsFolder%	現在のユーザーのための管理ツールを保管するファイルシステムディレクトリへのフルパス。典型的なパスは C:\Users\Username\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Administrative Tools です。
%AppDataFolder%	アプリケーション固有のデータのための共通レポジトリとしての役割を果たすファイルシステムディレクトリ。典型的なパスは C:\Users\username\AppData\Roaming です。
%FavoritesFolder%	現在のユーザーの[お気に入り]ディレクトリへフルパス。典型的なパスは C:\Users\Username\Favorites です。
%PersonalFolder%	現在のユーザーの[個人]ディレクトリへのフルパス。典型的なパスは C:\Users\Username\Documents です。
%SendToFolder%	宛先 のメニューアイテムを含むディレクトリへのフルパス。典型的なパスは C:\Users\username\AppData\Roaming\Microsoft\Windows\SendTo です。
%FontsFolder%	システムフォントディレクトリへのフルパス。典型的なパスは C:\Windows\Fonts です。
%ProgramFilesFolder%	プログラムファイルディレクトリへのフルパス。典型的なパスは C:\Program Files と C:\Program Files (x86) です。
%CommonFilesFolder%	共通ファイルディレクトリへのフルパス。典型的なパスは C:\Users\Public\Public Documents です。
%WindowsFolder%	(%WINDIR% 環境変数と同様)ウィンドウディレクトリへのフルパス。典型的なパスは C:\Windows です。
%SystemFolder%	システムフォルダーへのフルパス。典型的なパスは %WINDIR%\system32 です。
%CommonAppDataFolder%	アプリケーションデータを含むファイルディレクトリへのフルパス。典型的なパスは C:\ProgramData です。
%LocalAppDataFolder%	ローカル(非ローミング) アプリケーションのためのデータレポジトリとしての役割を果たすファイルシステムディレクトリへのフルパス。典型的なパスは C:\Users\username\AppData\Local です。
%MyPicturesFolder%	現在のユーザーの[ピクチャ]ディレクトリへのフルパス。典型的なパスは C:\Users\Username\Pictures です。

13 Visual Studio のための MapForce プラグイン

MapForce 2021 をMicrosoft Visual Studio バージョン2010/2012/2013/2015/2017/2019 に統合することができます。これはMapForce のマッピング機能とVisual Studio の開発環境と2つの世界をつなぎます。MapForce プラグインが有効化されると、マッピングプロジェクトとファイルを直接 Visual Studio から作成することができます。MapForce のスタンドアロンのバージョンと同様に、メニューとツールバーを含む、MapForce オプションをカスタム化することができます。



MapForce Enterprise Edition プラグイン(Visual Studio 2017)

(C) 2015-2021 Altova GmbH

13.1 プラグインの有効化

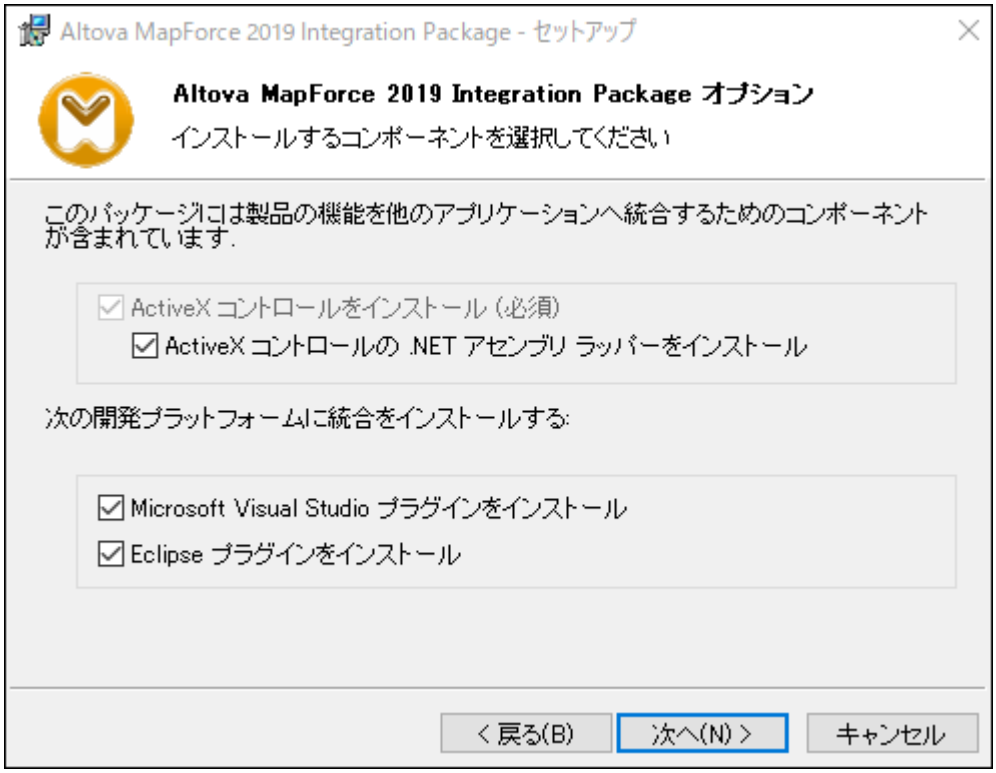
必要条件:

- Microsoft Visual Studio 2010/2012/2013/2015/2017/2019
- MapForce (Enterprise または Professional Edition)
- MapForce Integration Package を以下でダウンロードすることができます https://www.altova.com/mapforce/download。

メモ MapForce を Visual Studio プラグインとして使用するには、32-ビット のMapForce とMapForce Integration Package をインストールする必要があります。Visual Studio 内での64-ビット プラグイン はサポートされません。

Visual Studio のためのMapForce プラグインを有効化するには、(MapForce と同じバージョンの) MapForce Integration Package をダウンロードして、実行し、スクリーンのインストールの方法に従ってください。

インストール中、「Microsoft Visual Studio をインストール」プラグイン オプション が選択されていることを確認してください:



プロンプトされると、プラグインが有効化されるべき Visual Studio バージョンを選択します。例:

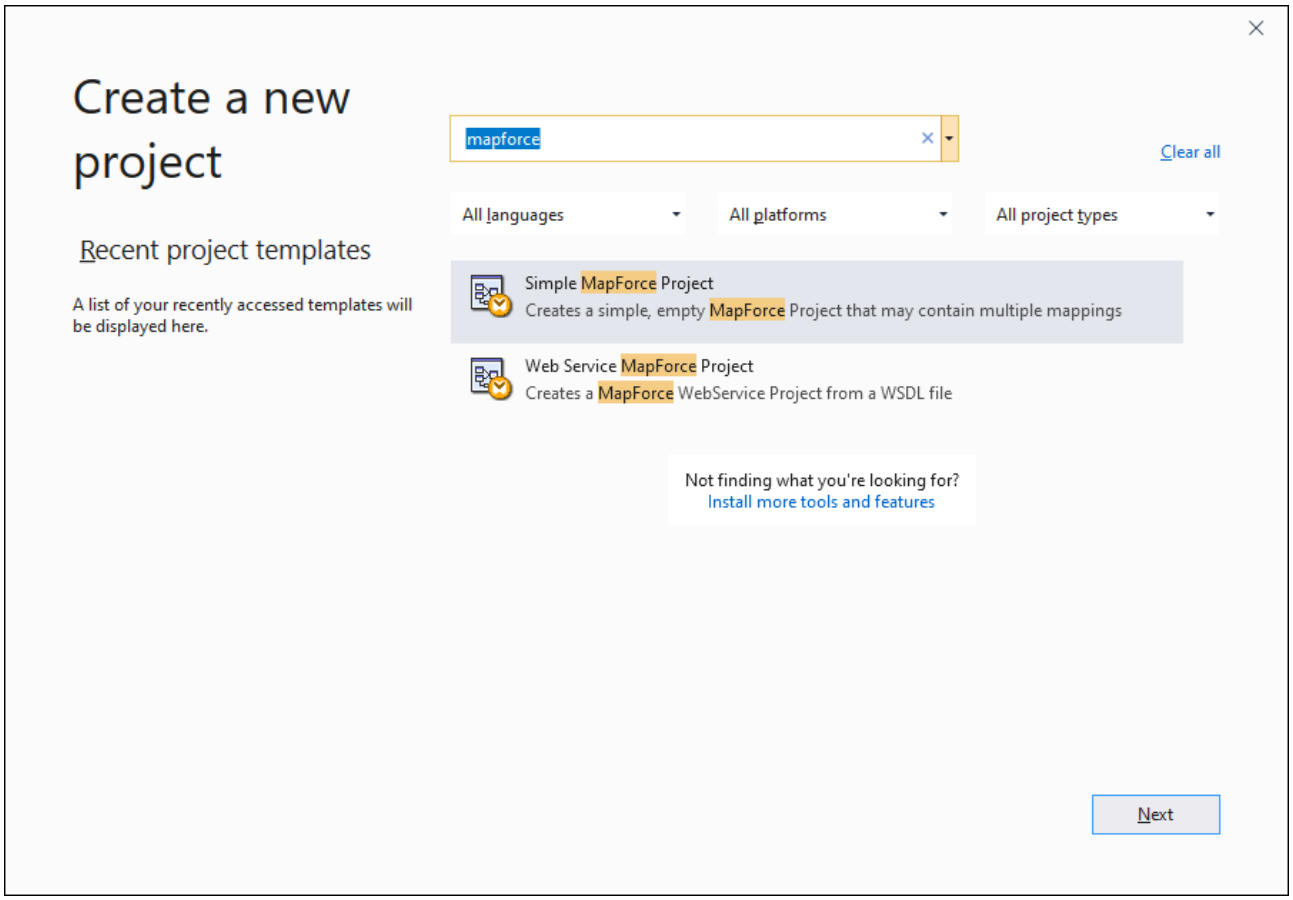
メモ 使用中のオペレーティングシステムにインストールされている Visual Studio のバージョンのみが選択のために表示されます。

Integration Package がインストールされると、MapForce 機能が Visual Studio 環境内で使用することができるようになります。

(C) 2015-2021 Altova GmbH

13.2 マッピングとプロジェクトと作業

Visual Studio のための MapForce プラグインが有効化されている場合、スタンドアロン MapForce グラフィカルユーザーインターフェイスとは異なり、Visual Studio 環境に適用できる方法でマッピングとマッピングプロジェクトを作成、または開くことができます。例:(「**ファイル | 新規作成**」メニューコマンドを使用して) Visual Studio 内に新規のファイルを作成すると、または (「**ファイル | 新規プロジェクト**」メニューコマンドを使用して) 新規アイテムをプロジェクトに追加すると、MapForce ファイル型として選択することができます。



新規ファイル ダイアログボックス (MapForce Enterprise エディションプラグインを使用した Visual Studio 2019)

既存の マッピングファイル または プロジェクト を開く場合、適用することのできる Visual Studio メニューを使用します (例:「**ファイル | 開く | ファイル**」または「**ファイル | 開く | プロジェクト /ソリューション**」)を使用して、 MapForce に関連したファイルの型を検索します。

13.3 共通メニューと関数へのアクセス

Visual Studio のための MapForce プラグインが有効化されている場合、下で表示されるように共通メニューと関数にアクセスすることができます。これはデフォルトのセットアップですが、Visual Studio の「**ツール | カスタマイズ**」メニューからメニューの場所を変更することができます。

グローバルリソース

MapForce グローバルリソースは、Visual Studio 内の「**MapForce | グローバルリソースの管理**」メニューで使用できます。

メモ Visual Studio 2019 内では対応するメニューは「**拡張子 | MapForce | グローバルリソースの管理**」です。

マッピングのデバッグ

「**MapForce | デバッグ**」メニューでマッピングのデバッグコマンドを使用することができ、「**デバッグ**」ツールバー内に表示されています。キーボードショートカットからデバッグを開始することもできます（デバッガーコマンドを参照してください）。

メモ Visual Studio 2019 内では対応するメニューは「**拡張子 | MapForce | デバッグ**」です。

MapForce オプション

MapForce オプションは、Visual Studio 内の「**ツール | MapForce オプション**」メニューで使用できます。

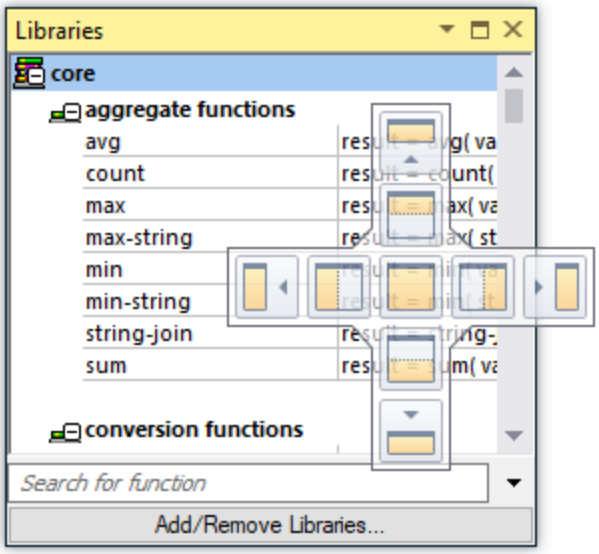
マッピングペインのカスタム化

Visual Studio のメインペインで MapForce マッピングが開かれると、「**表示 | MapForce**」メニューを使用することができます。MapForce のスタンドアロンバージョンと同じオプションが含まれます。

ライブラリウィンドウ

MapForce ライブラリウィンドウは、プラグインがインストールされた後、デフォルトでは Visual Studio 内で有効化されていません。このウィンドウで頻繁に作業する場合、「**表示 | MapForce | ライブラリウィンドウ**」メニューから有効化することができます（このメニューは、メインウィンドウ内でマッピングファイルが開かれると、Visual Studio 内で使用することができます）。ライブラリウィンドウが有効化されると、インターフェイスの特定の箇所に、他の Visual Studio のコンポーネントと同様ドッキングすることができます。

(C) 2015-2021 Altova GmbH



ライブラリウィンドウ(MapForce Enterprise エディション プラグインと Visual Studio 2017)

ツールバーとコマンドのカスタム化

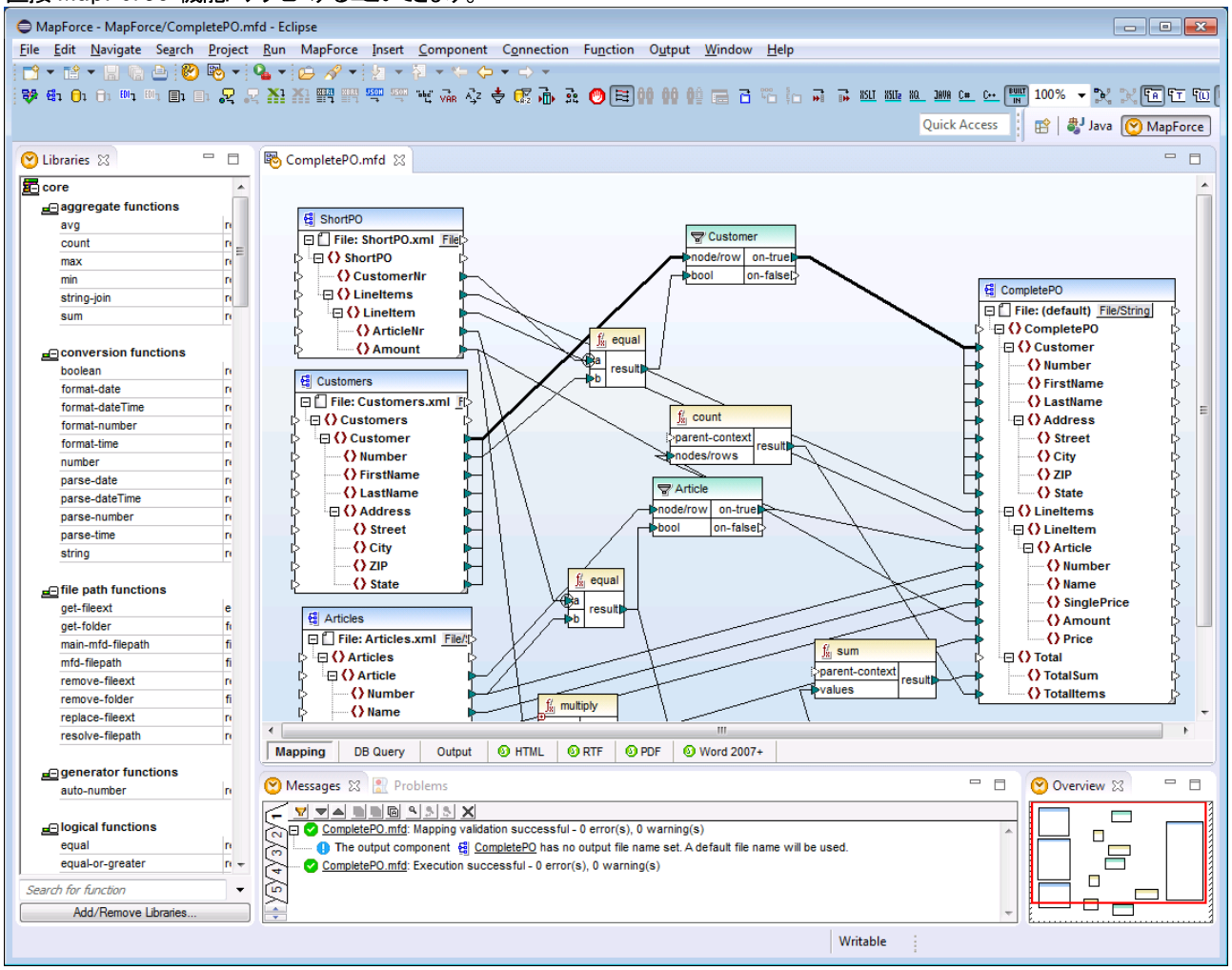
MapForce メニューとツールバーを Visual Studio の「**ツール | カスタマイズ**」メニューからカスタム化することができます。

ヘルプとサポート

Visual Studio の「**ヘルプ | MapForce ヘルプ**」メニューで MapForce ヘルプ、サポートセンター、最新情報のチェック、および MapForce についてのメニューを使用することができます。

14 Eclipse のための MapForce プラグイン

Eclipse は、プラグインのフォーム内で届けられる異なる種類のアプリケーションを統合するオープンソースのフレームワークです。MapForce Enterprise とProfessional Edition をEclipseバージョン 2020-12, 2020-09, 2020-06, 2020-03 に統合し、Eclipse から直接 MapForce 機能にアクセスすることができます。



MapForce Enterprise Edition プラグイン (Eclipse 4.4)

以下のトピックは、はのインストールおよび使用について説明しています。

- Eclipse のためのMapForce プラグインのインストール
- MapForce パースペクティブ
- 共通メニューと関数へのアクセス
- マッピングとプロジェクトとの作業
- Eclipse のためのMapForce プラグインの有効化

(C) 2015-2021 Altova GmbH

14.1 Eclipse のための MapForce プラグインのインストール

必要条件:

- Eclipse 2020-12, 2020-09, 2020-06, 2020-03 (http://www.eclipse.org)。
- the 64-ビットプラットフォームのための Java Runtime Environment (JRE) または Java Development Kit (JDK)
- MapForce Enterprise または Professional Edition 64-ビット
- MapForce Integration Package 64-ビット を以下でダウンロードすることができます https://www.altova.com/ja/components/download。下記の通り Eclipse 統合をインストールの一部として行うことができます。

上記の必要条件は 64-ビットプラットフォームを必要としています。古い Eclipse 32-ビットプラットフォームとの統合はサポートされていません。

Eclipse のための MapForce プラグインをインストールする方法

Eclipse のための MapForce プラグインの統合を以下の方法の一つで行うことができます:

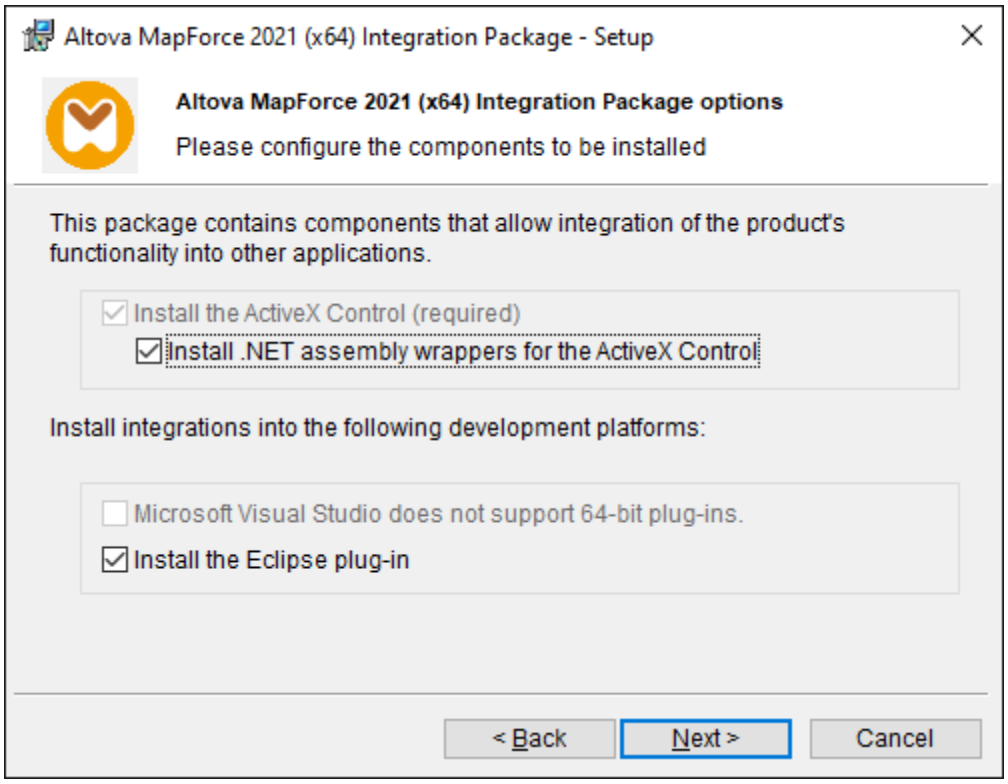
1. 自動的に MapForce Integration Package のインストール中に統合することができます (これは奨励されるオプションです)。
2. MapForce Integration Package を最初にインストールし、Eclipse からプラグインを手動で統合する。

Eclipse のための MapForce プラグインを統合する方法:

1. インストールウィザードを開始するために MapForce Integration Package を実行します。

   MapForce Integration Package をインストールまたはアンインストールする場合 Eclipse を閉じる必要があります。

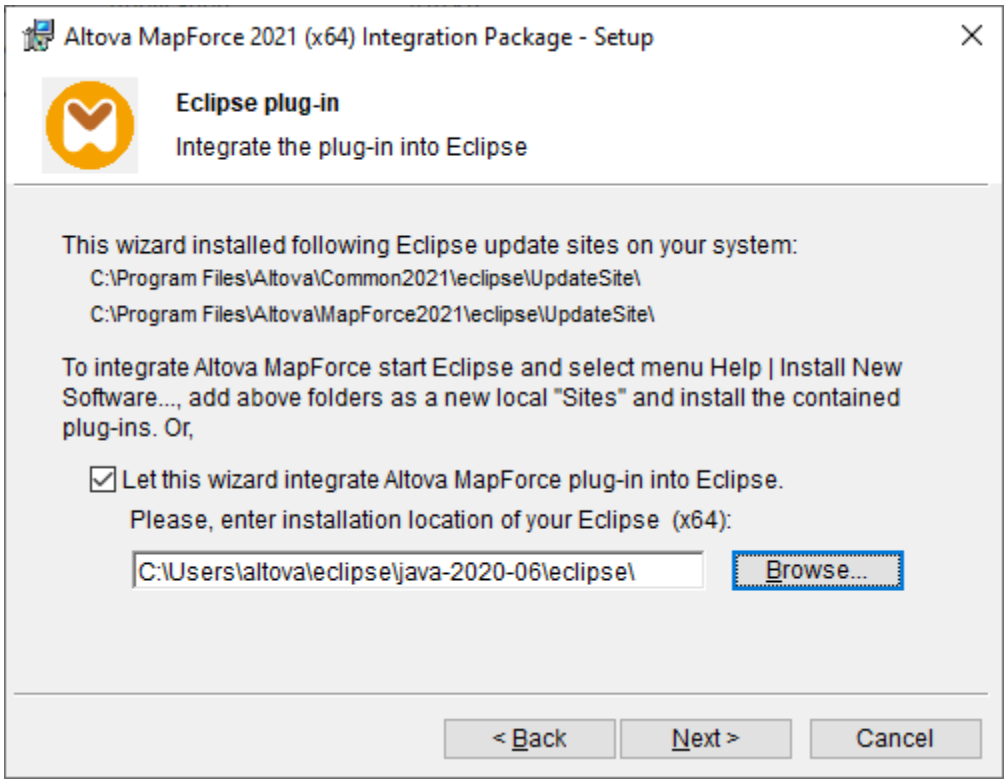
2. プロンプトされると Eclipse プラグインのインストール オプションを選択し [次へ] をクリックします。



3. プロンプトされるとMapForce プラグインのEclipse への統合方法を選択します。以下の1つを行ってください:

   - プラグインインストールを自動的に完了する場合 このウィザードに Altova **MapForce** プラグインを Eclipse に統合するを選択し、Eclipse 実行可能ファイル(`eclipse.exe`) がロードされている場所を参照します。
   - Eclipse 内で後にプラグインのインストールを完了する場合、clear the このウィザード... チェックボックスをクリアします。

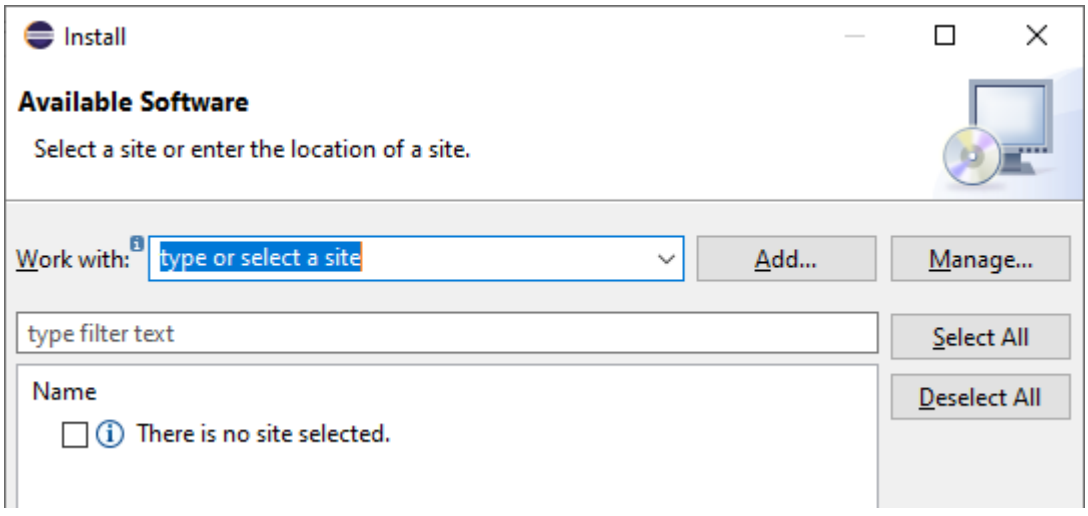
(C) 2015-2021 Altova GmbH



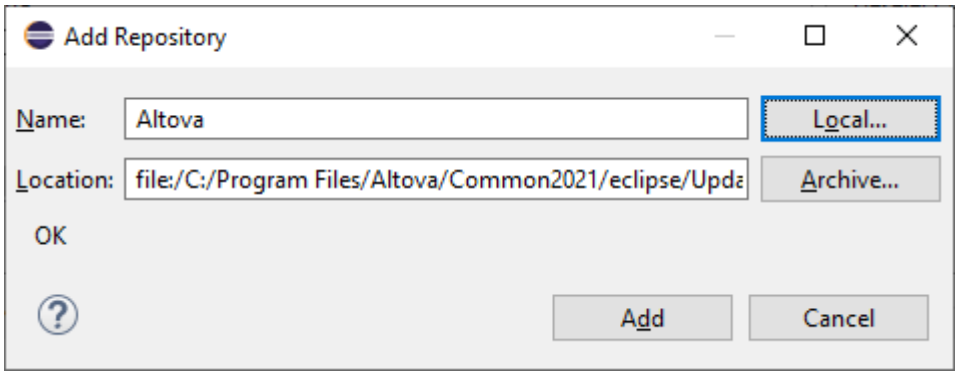
4. [次] をクリックすることでインストールを完了します。自動統合を選択する場合 MapForce パースペクティブとメニューがEclipse を次回起動する際にEclipse 内で使用可能になります。

Eclipse のための MapForce プラグインを手動で統合する方法

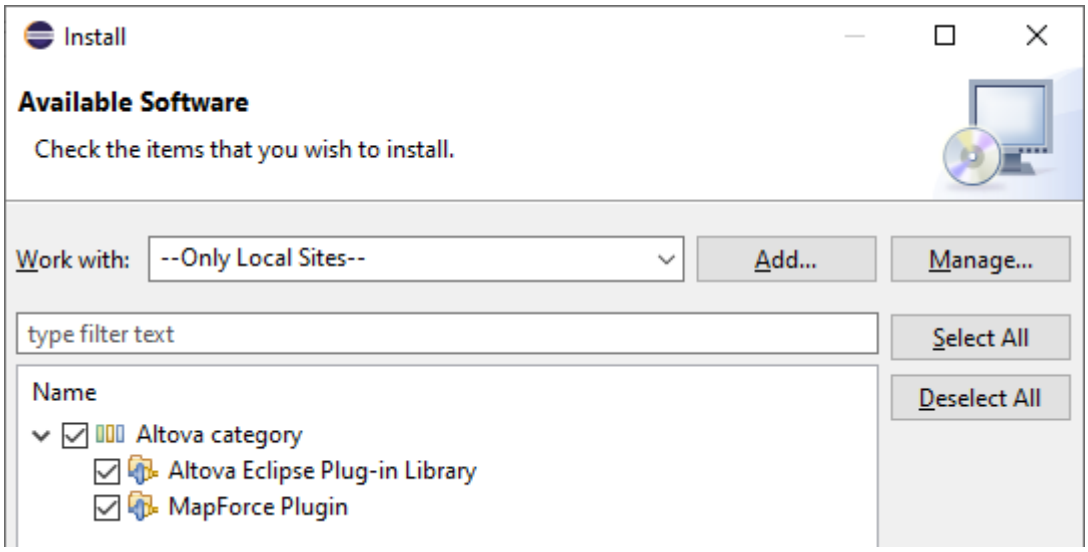
1. Eclipse にてメニューオプションから[ヘルプ | 新規ソフトウェアのインストール]を選択します。
2. [インストール] ダイアログボックス 内で [追加]をクリックします。



3. [レポジトリの追加] ダイアログボックス 内で [ローカル]をクリックします。フォルダー**C:\Program Files\Altova\Common2021\eclipse\UpdateSite** を参照して選択します。("Altova" などの) サイトの名前を与えます。



4. ステップ 2 から 3 を繰り返します、今度は、フォルダー**C:\Program Files\Altova\MapForce2021\eclipse\UpdateSite**を選択し、MapForce などの名前を与えます。
5. インストールダイアログボックスで **--Only Local Sites--** を選択します。次に[Altova category] フォルダーを選択し [次へ] をクリックします。



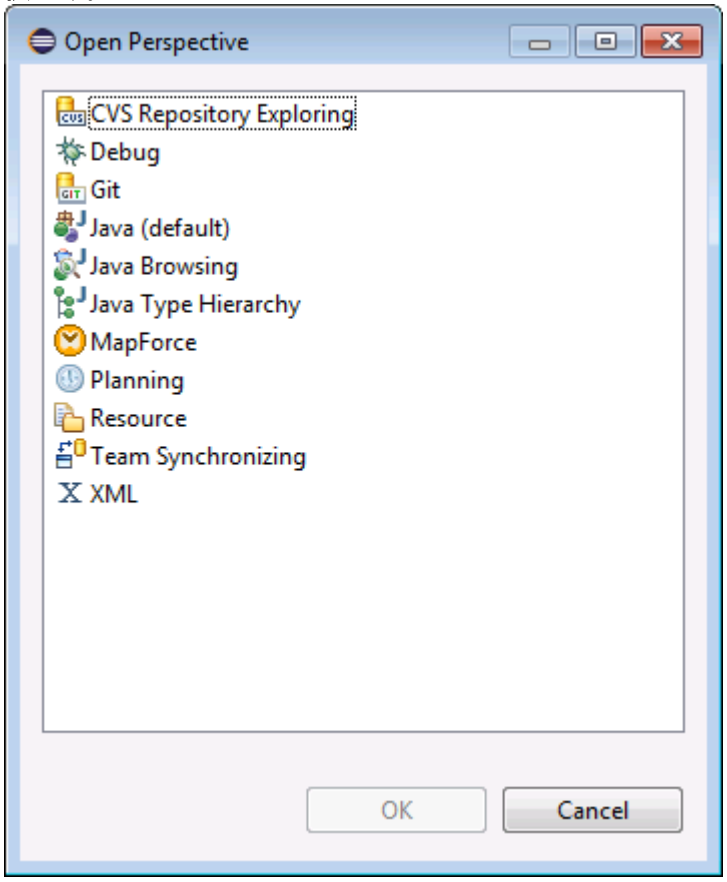
6. インストールされるアイテムをレビューし [次へ] をクリックして継続します。
7. 使用許諾契約書に合意するには対応するチェックボックスを選択します。
8. [完了] をクリックすることでインストールを完了します。

メモ （不足するアイコンなど）プラグインの使用に関する問題が発生した場合 `-clean` フラグを使用してコマンドラインから Eclipse を起動します。

(C) 2015-2021 Altova GmbH

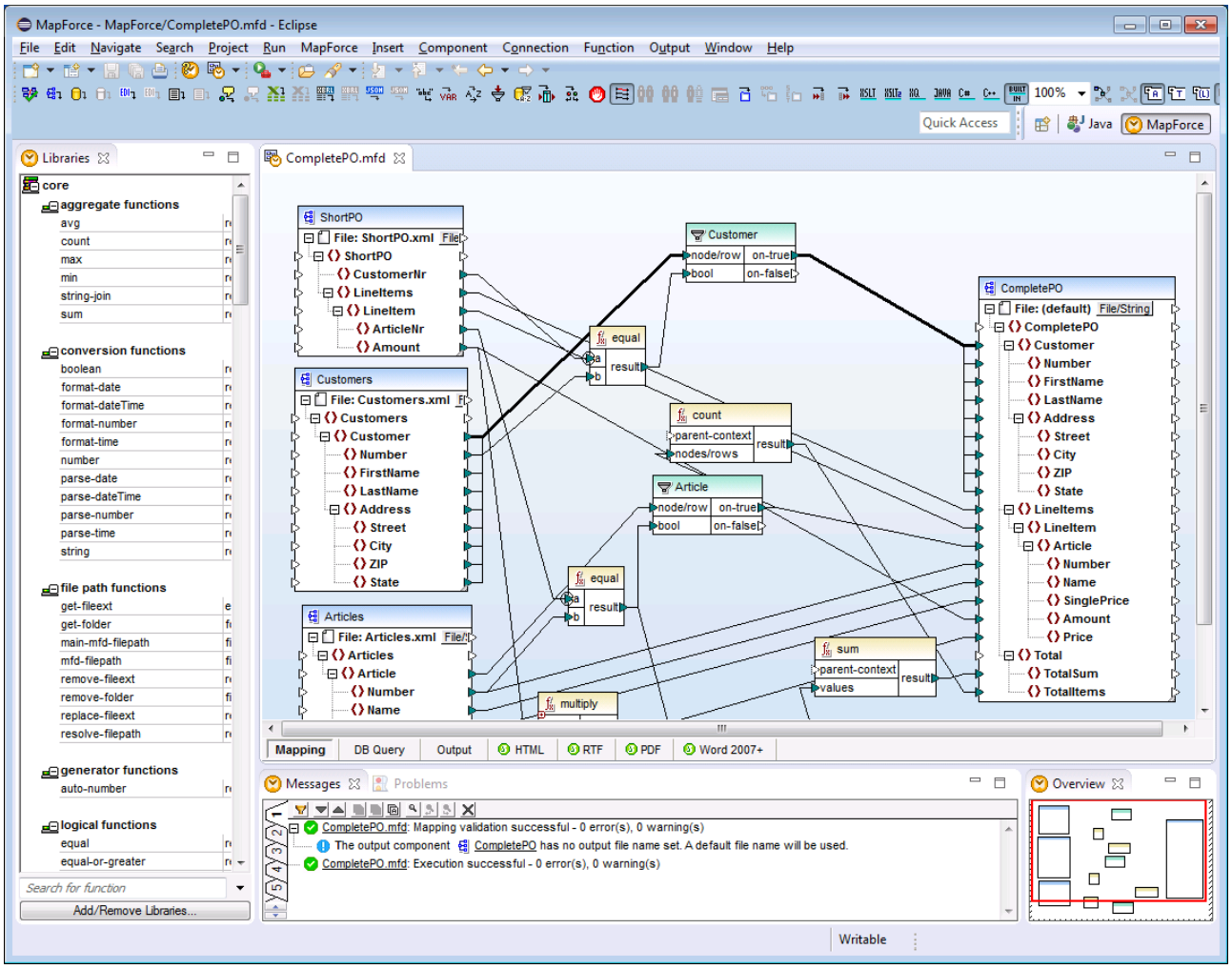
14.2 MapForce パースペクティブ

Eclipse のための MapForce プラグインをインストールすると、Eclipse 内で新しいパースペクティブ("MapForce")を使用することができるようになります。このパースペクティブのレイアウトは、MapForce のスタンドアロンエディションのインターフェイスに類似しています。MapForce パースペクティブに切り替えるには、**「ウィンドウ | パースペクティブを開く | その他」**をクリックし、MapForce をリストから選択します。



Eclipse 内で MapForce パースペクティブを選択する

MapForce パースペクティブは、Eclipse の他のパースペクティブと同じで、必要に応じて Eclipse 内(**「ウィンドウ | ナビゲーション | 次へパースペクティブ」**)で切り替えます。含まれるビューをカスタム化することもでき、Eclipse 設定から他の複数のオプションをカスタム化します。(MapForce パースペクティブをカスタム化するには、MapForce パースペクティブに[Window | **パースペクティブを開く | その他**]から切り替え、メニューコマンド[Window | **パースペクティブのカスタム化**]を選択します)。Eclipse パースペクティブに関しての詳しい情報は、Eclipse ドキュメントを参照してください。以下のスクリーンショットは MapForce パースペクティブのスイッチがオンされている Eclipse 環境を示しています。



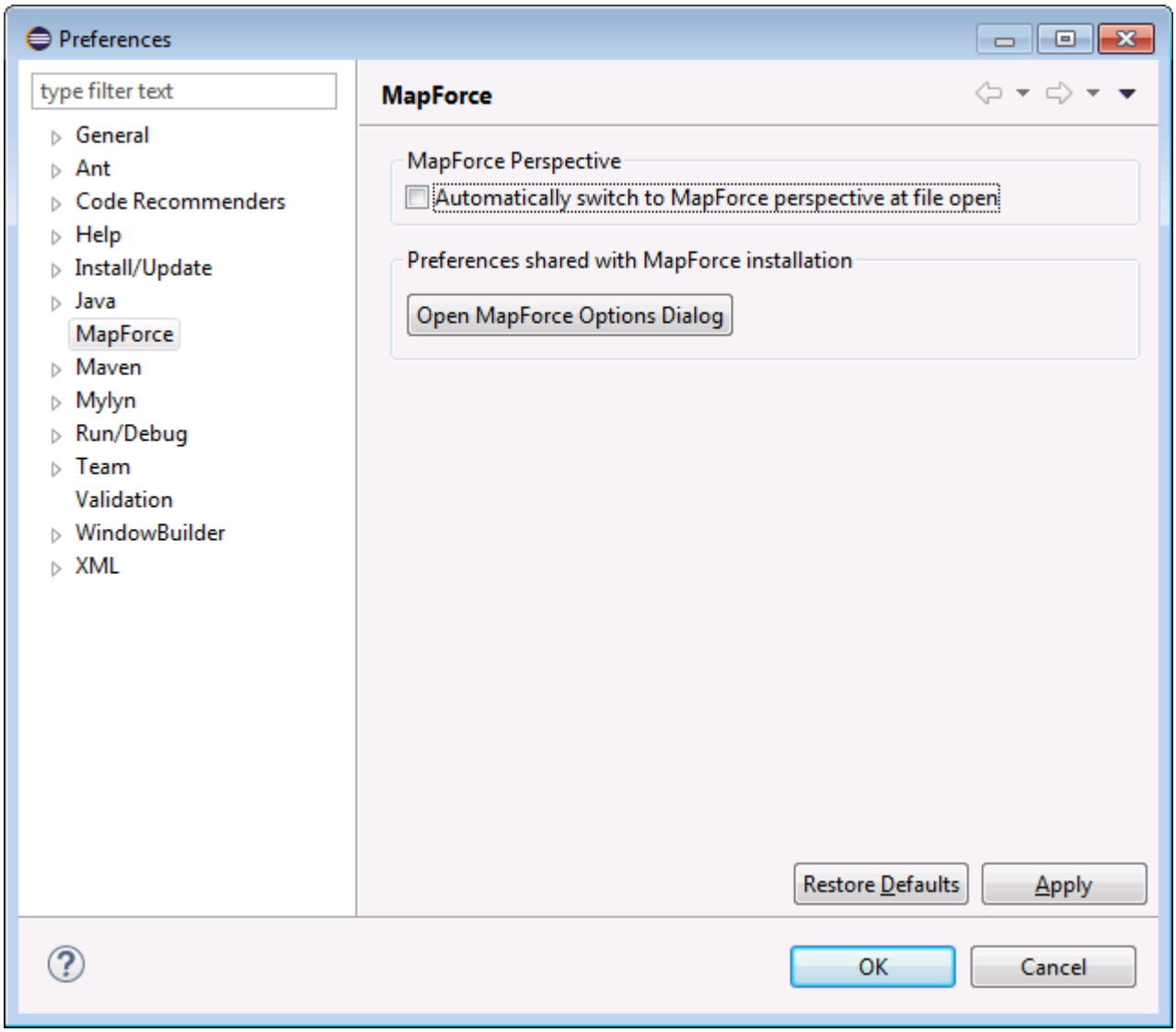
MapForce パースペクティブ(Eclipse のためのMapForce Enterprise Edition プラグイン)

デフォルトでは、Eclipse 内でのMapForce パースペクティブは以下のように整理されています:

- Eclipse エディターでマッピングデザイン ウィンドウを使用することができます。MapForcein のスタンドアロンエディション内と同じタブと機能が搭載されています。
- Eclipse ビューとして、メインマッピングエディターに左のライブラリウィンドウを使用することができます。このビューが表示されていない場合、MapForce パースペクティブに切り替えます。メニューコマンド「**ウィンドウ | ビューの表示 | ライブラリ**」を選択します。ライブラリビューにより、定義済みまたはカスタム定義された関数と関数ライブラリと作業できます。
- Eclipse ビューとして、メインマッピングエディター下のメッセージペインを使用することができます。メッセージビューが表示されていない場合、MapForce パースペクティブに切り替え、メニューコマンド「**ウィンドウ | ビューの表示 | メッセージ**」を選択します。メッセージは、検証メッセージ、エラー、と警告を表示します。
- Eclipse ビューとして、概要ペインを使用することができます。アウトラインビューが表示されていない場合は、MapForce パースペクティブに切り替え メニューコマンド「**ウィンドウ | ビューの表示 | アウトライン**」を選択します。このビューを使用することにより、大きなマッピングのマッピングデザインの特定の箇所に素早く移動することができます。

MapForce マッピングを開くと、Eclipse がMapForce パースペクティブに自動的に切り替えるように構成することもできます。これを行うには、メニューコマンド「**ウィンドウ | 設定**」を開き、MapForce を選択して、「**ファイルが開かれると自動的にMapForce パースペクティブに切り替える**」チェックボックスを選択します。

(C) 2015-2021 Altova GmbH



設定 ダイアログボックス

14.3 共通メニューと関数へのアクセス

Eclipse 内では、下にリストされるEclipse 固有のバリエーションを除き、スタンドアロンバージョン内と同様にMapForce 機能の同じメニューの多数にアクセスすることができます。これはデフォルトのセットアップですが、 Eclipse からのインターフェイスの基本設定を更にカスタマイズする場合は、（ MapForce パースペクティブを参照してください）。

メモ コンテキストが関連しない場合、Eclipse 内では、MapForce メニューグループまたはコマンドの一部が無効化（または使用できない）場合があります。例: Eclipse 内でマッピングデザインファイル（.mfd）がアクティブな場合のみ、「**挿入**」メニューを使用することができます。

MapForce 標準メニューに関する詳細は、 メニューレファレンスを参照してください。

一般的な MapForce コマンド

MapForcein のスタンドアロンエディション内で、マッピングデザインファイルに適用することのできる（「**検証**」、「**FlowForce Server にデプロイ**」、「**コードの生成**」などの）コマンドは、「**ファイル**」メニュー内で使用することができます。Eclipse 内でこれらのコマンドは **MapForce** メニュー内または MapForce ツールバーで使用することができます。（ MapForce プロジェクトファイルを含む）ファイルを開いたり閉じたりするコマンドは、Eclipse の「**ファイル**」メニューで使用することができることにご注意してください。

Eclipse 内の MapForce ツールバー

ツールバーボタンは、MapForce ヘルプファイルを開きます。

ツールバーボタンは、 MapForce ファイル固有のコマンドを表示します。このボタンを展開すると、現在 Eclipse エディター上でアクティブなファイルにより決定される使用可能なコマンドが表示されます。例: マッピングデザイン（.mfd）ファイル固有のコマンドは、このようなファイルがでアクティブな場合、Eclipse エディター内で使用することができます。

グローバルリソース

グローバルリソースへアクセスまたはグローバルリソースの管理は、 以下を行います:

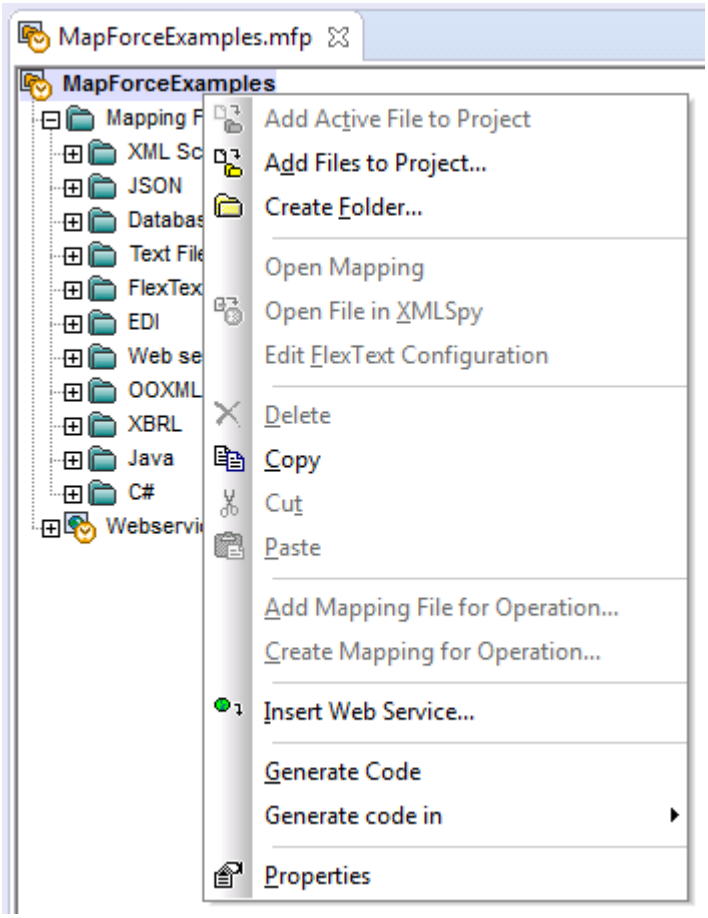
- MapForce ツールバーボタンをクリックして展開し、**グローバルリソース** をクリックします。
- MapForce メニューから**グローバルリソース**をクリックします。

MapForce プロジェクト

MapForce のスタンドアロンのエディションでは、**プロジェクト** メニューは、マッピング プロジェクト（.mfp）ファイルに適用することのできる複数のコマンドを含んでいます。Eclipse 内では、これらのコマンドは以下のように存在します:

- プロジェクトを開くまたは閉じるコマンドは、 Eclipse「**ファイル**」メニューから使用することができます。
- 他のプロジェクト専用のコマンドは、Eclipse 内でコンテキストコマンドとして使用できます。コンテキストコマンドを表示、MapForce プロジェクト（.mfp）ファイルを作成または開く場合は、プロジェクトを右クリックしてください。

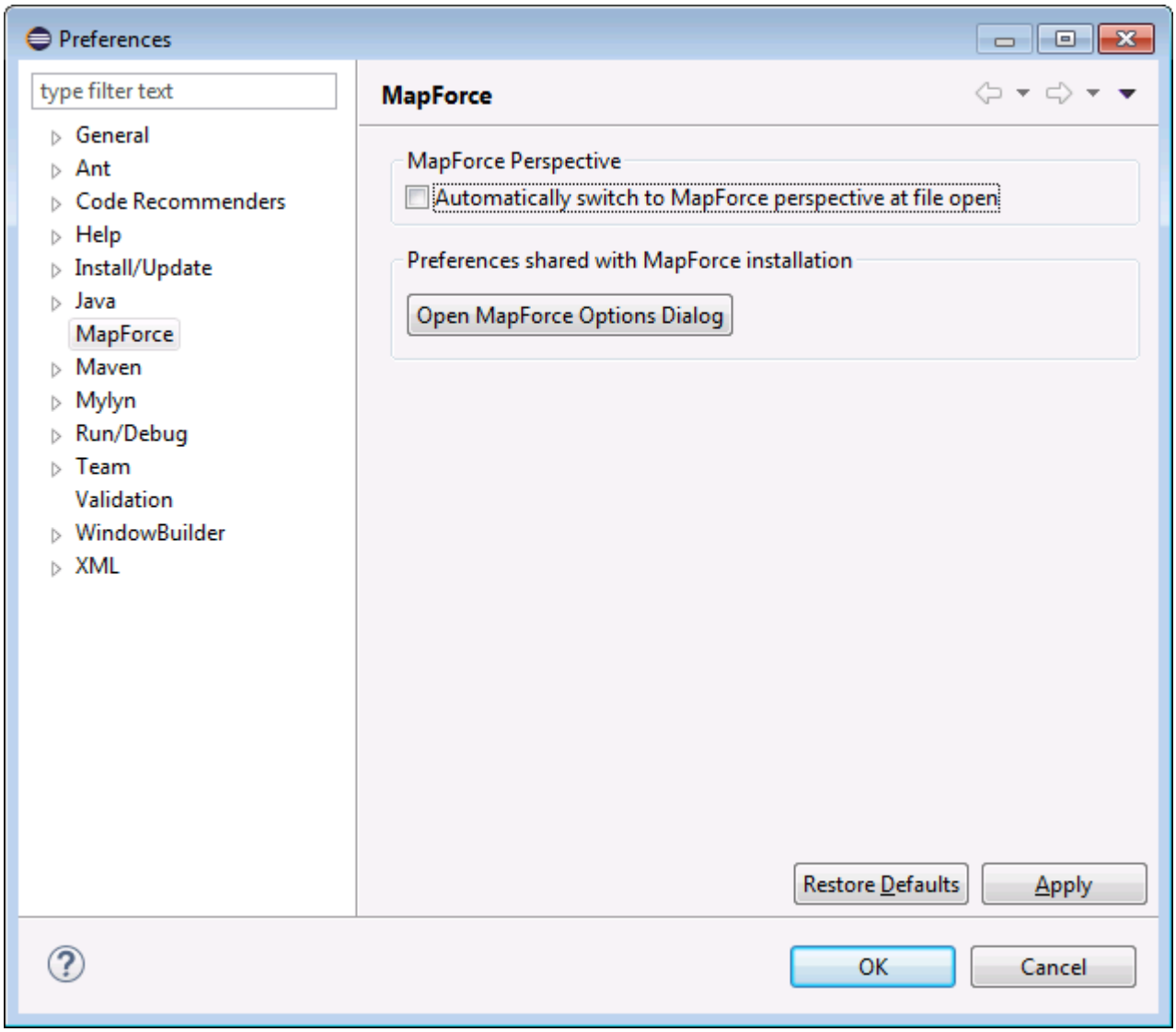
(C) 2015-2021 Altova GmbH



標準 MapForce プロジェクト (.mfp) に加え、Eclipse 内では、"MapForce/Eclipse" の型のプロジェクトも作成することができます。このようなプロジェクトは、2面の特質のため、MapForce コードの自動的なビルドと生成のために構成することができます。マッピングおよびプロジェクトと作業を参照してください。

MapForce オプション

MapForce オプションは、「ウィンドウ | 優先」メニューから使用することができます。優先ダイアログボックスで、**MapForce** を選択し、「**MapForce オプションダイアログを開く**」をクリックします。



設定ダイアログボックス

ライブラリウィンドウ

Eclipse 内で、MapForce ライブラリウィンドウは、ビューとして使用することができます。このビューはデフォルトでは、メインエディターウィンドウの左に配置されています。(MapForce に関連したすべてのビューは、MapForce パースペクティブ がオンに切り替えられると、Eclipse インターフェイスで表示されます MapForce パースペクティブも参照)。

MapForce プラグイン バージョン

Eclipse のための MapForce プラグインの現在インストールされているバージョンを確認するには、Eclipse メニューオプション「**ヘルプ | Eclipse プラットフォームについて**」を選択して、そして、MapForce アイコンを選択します。

ヘルプとサポート

MapForce ヘルプ、サポートセンター、最新情報 メニューは、Eclipse の「**ヘルプ | MapForce ヘルプ**」メニューで使用することができます。

14.4 マッピングおよびプロジェクトと作業

When Eclipse のためのMapForce プラグイン がインストールされている場合、Eclipse のEclipse プロジェクトからMapForce のスタンドアロンエディション内と同じマッピング とマッピングプロジェクト型 を作成することができます。マッピングをデザイン、コンパイル、デプロイし、マッピングコードを生成するには、新規のEclipse プロジェクトを作成するか、または既存のEclipse プロジェクトを使用することができます (例: MapForce マッピングを追加するJava プロジェクト)。

更に、 **MapForce/Eclipse プロジェクト** 内の MapForce プラグインをインストール後にEclipse 内で使用することのできる、特別なプロジェクト型内で全てのマッピングと作業することができます。カスタマイズを選択しない限り、MapForce/Eclipse プロジェクトはデフォルトでJava ビルダーとMapForce コード生成ビルダーが割り当てられています。更に、2つのEclipse 性質が存在します: MapForce 性質と JDT (Java 開発ツール) 性質。この結果、MapForce/Eclipse プロジェクトは、(マッピングデザインファイルなどの)リソースを保存または変更する場合、以下に記載されるように振る舞います:

- **プロジェクト** > **「自動的にビルド」** メニューオプションが有効化されていると、マッピングコードは自動的に生成されます。1つまたは複数のMapForce プロジェクトファイルMapForce/Eclipse プロジェクト内に存在する場合、コード生成言語と出力ターゲットフォルダーは、各プロジェクトファイル内の設定により決定されます。それ以外の場合、Eclipse は、場所を選択するようにプロンプトします。
- エラーと出力メッセージは、メッセージと問題ビューに表示されます。

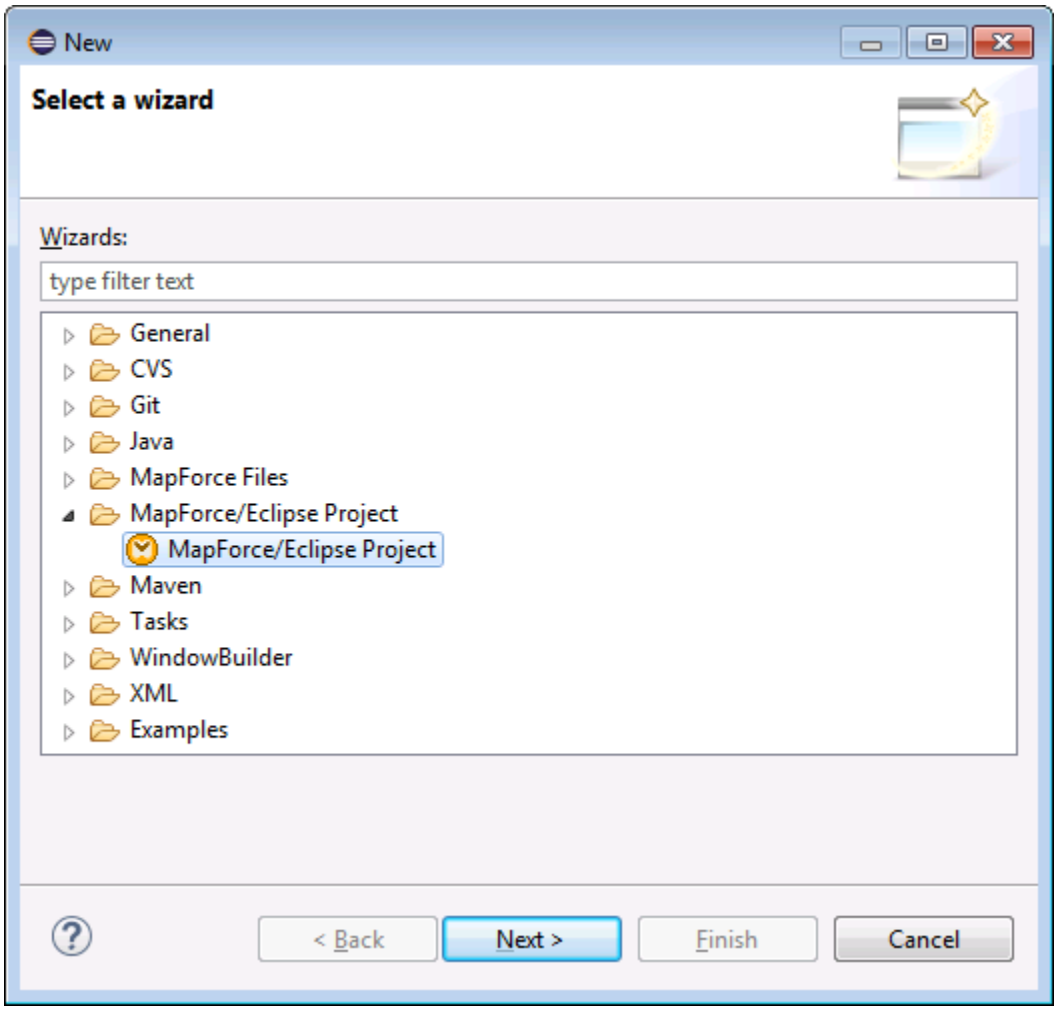
このセクションは、以下のトピックを含みます:

- MapForce/Eclipse プロジェクトの作成
- 新しいマッピングの作成
- Eclipse プロジェクトに既存のマッピングをインポートする
- 自動ビルドの構成とMapForce コードの生成

14.4.1 MapForce/Eclipse プロジェクトの作成

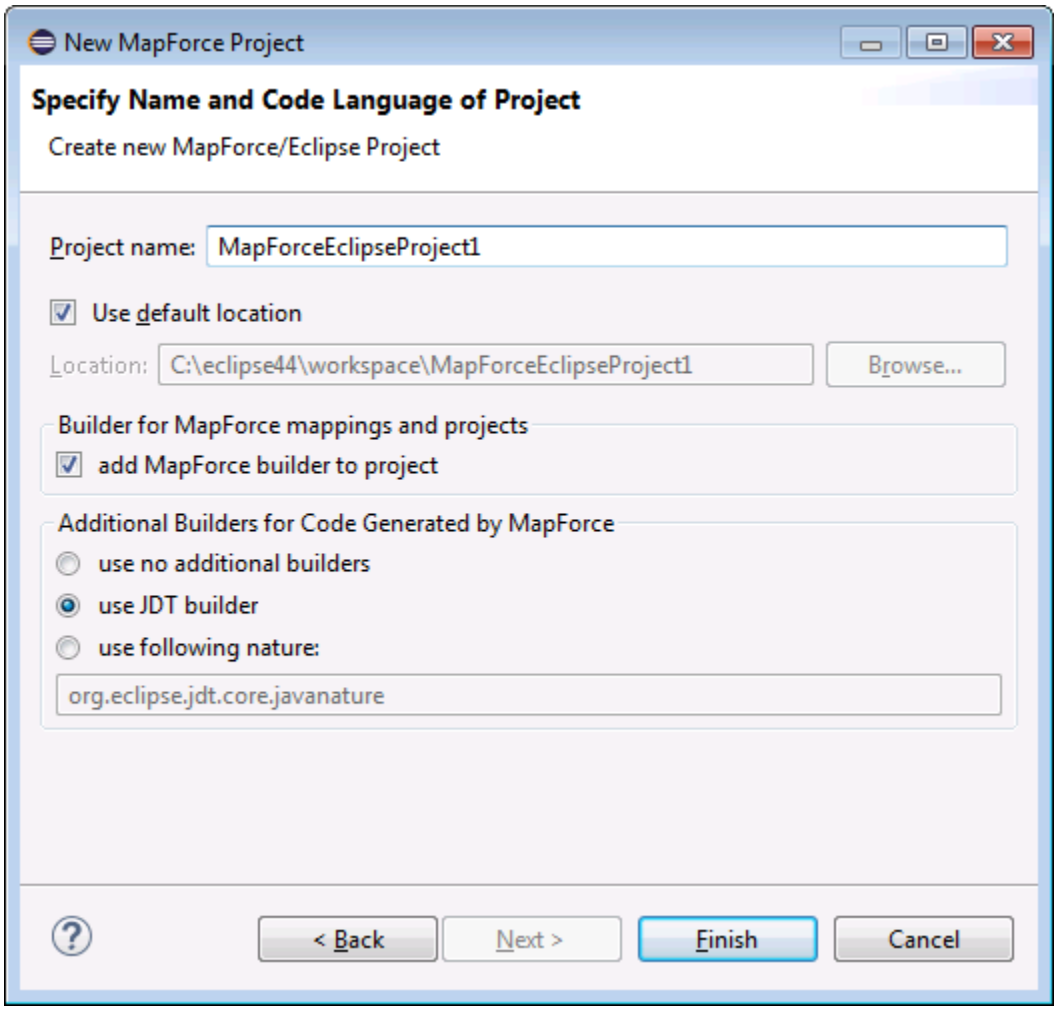
MapForce/Eclipse プロジェクトの作成方法:

1. **「ファイル」** メニューから**「新規作成 | その他」** をクリックします。
2. **「MapForce/Eclipse プロジェクト」** カテゴリーを選択します。



3. 「次へ」をクリックします。

(C) 2015-2021 Altova GmbH



4. プロジェクト名を入力して、プロジェクトの保存場所を選択します。「MapForce ビルダーをプロジェクトへ追加」そのままにして、「JDT ビルダーを使用」オプションをそのままにします。
5. 「完了」をクリックします。

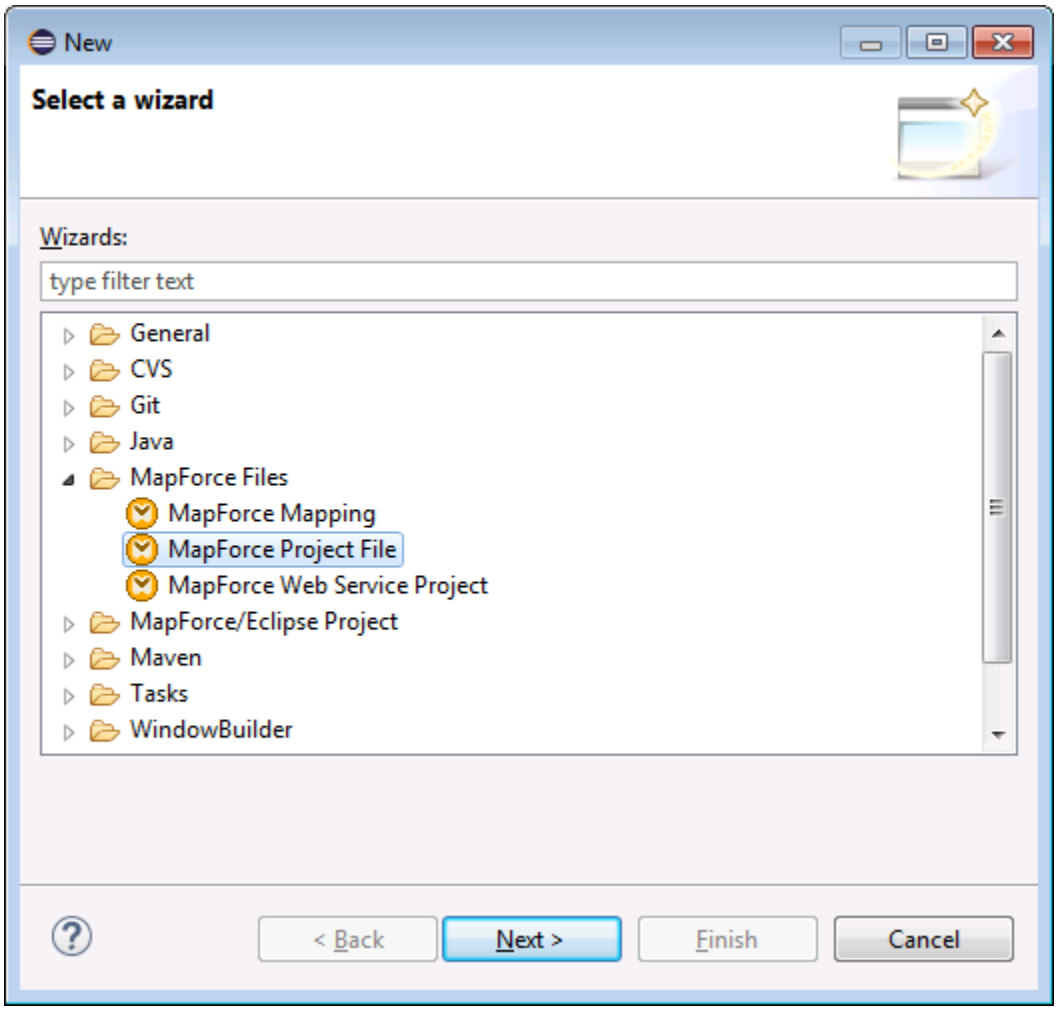
14.4.2 新しいマッピングの作成

以下のMapForce ファイル型をEclipse プロジェクト内で作成することができます:

- MapForce マッピング
- MapForce プロジェクトファイル
- MapForce Web サービスプロジェクト (MapForce Enterprise Edition で使用することができます)

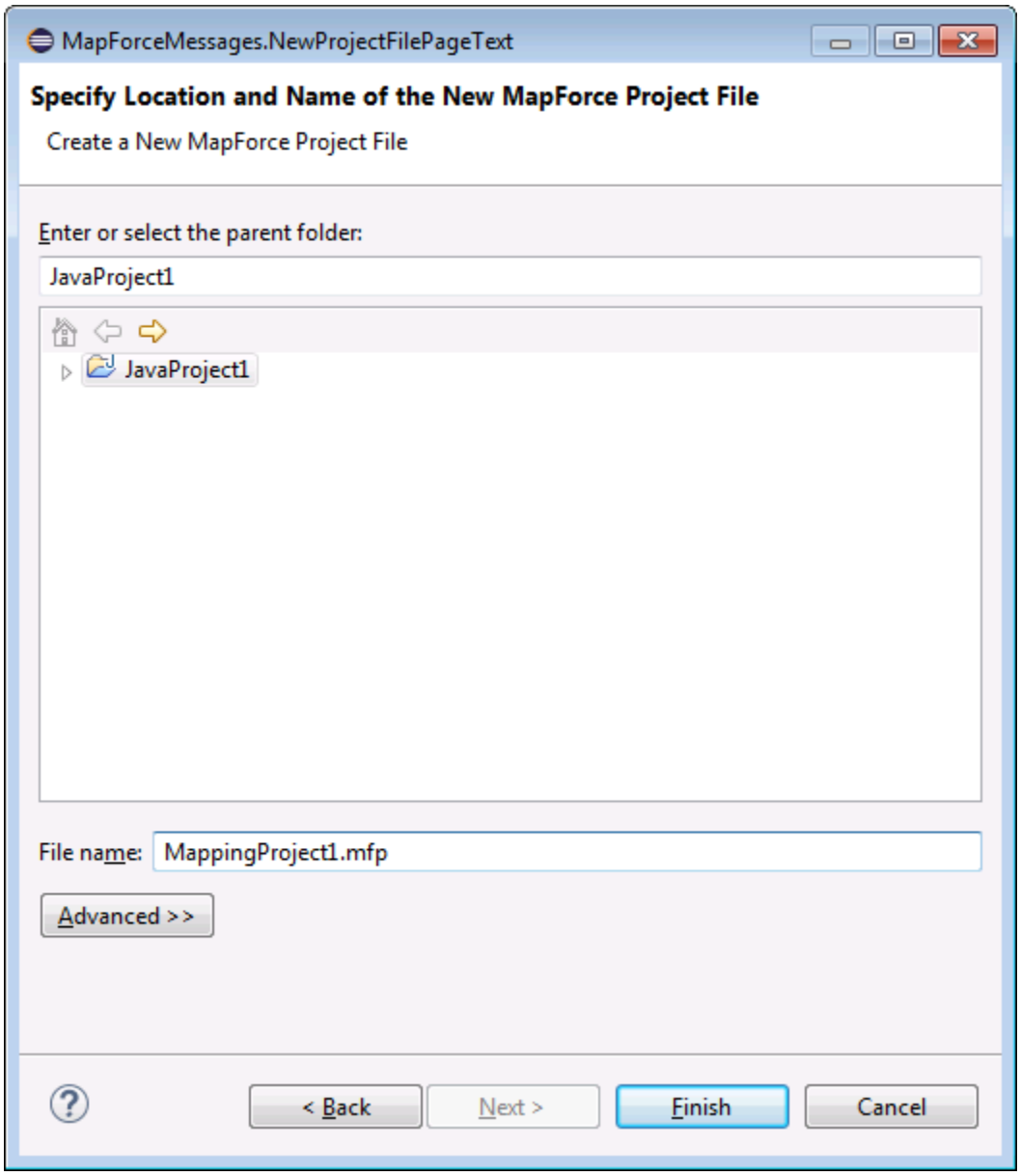
Eclipse プロジェクト内でこれらのファイル方を作成する:

1. 新規のEclipse プロジェクトを作成、または既存のプロジェクトを開きます。
2. 「ファイル」メニューから「新規作成」をクリックし、「その他」をクリックします。



3. ウィザードダイアログボックスから必要なファイル方を選択し、「次へ」をクリックします。

(C) 2015-2021 Altova GmbH

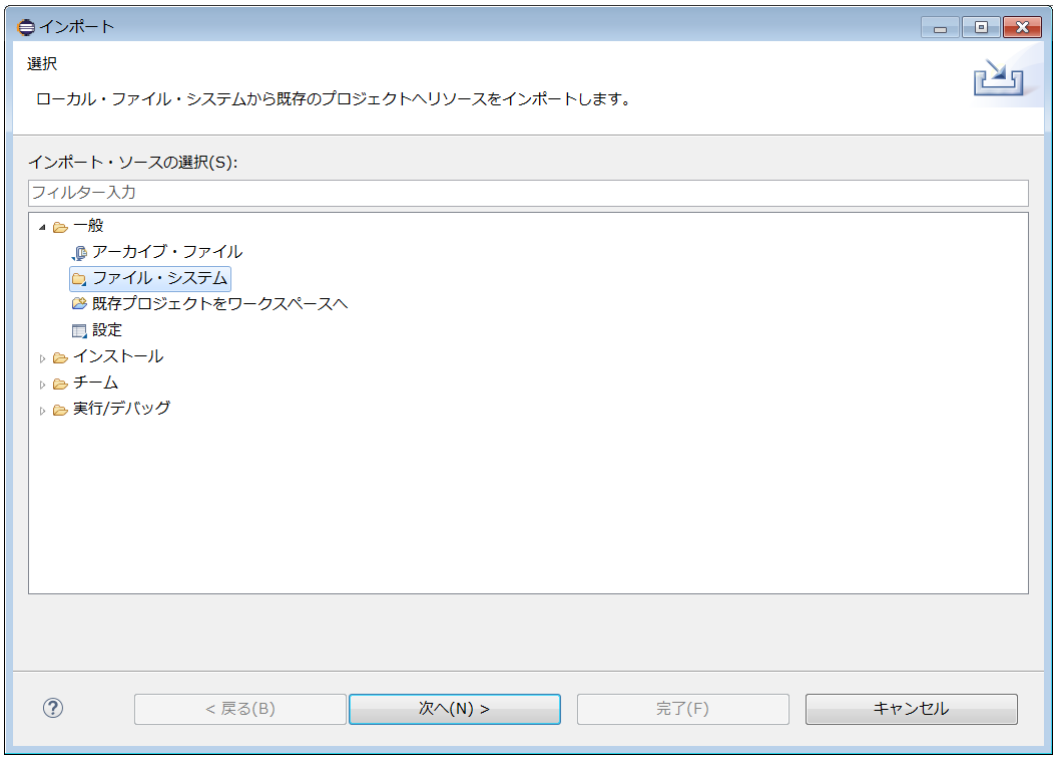


4. 既存のプロジェクト内の親フォルダーを選択して、「完了」をクリックします。

14.4.3 Eclipse プロジェクトに既存のマッピングをインポートする

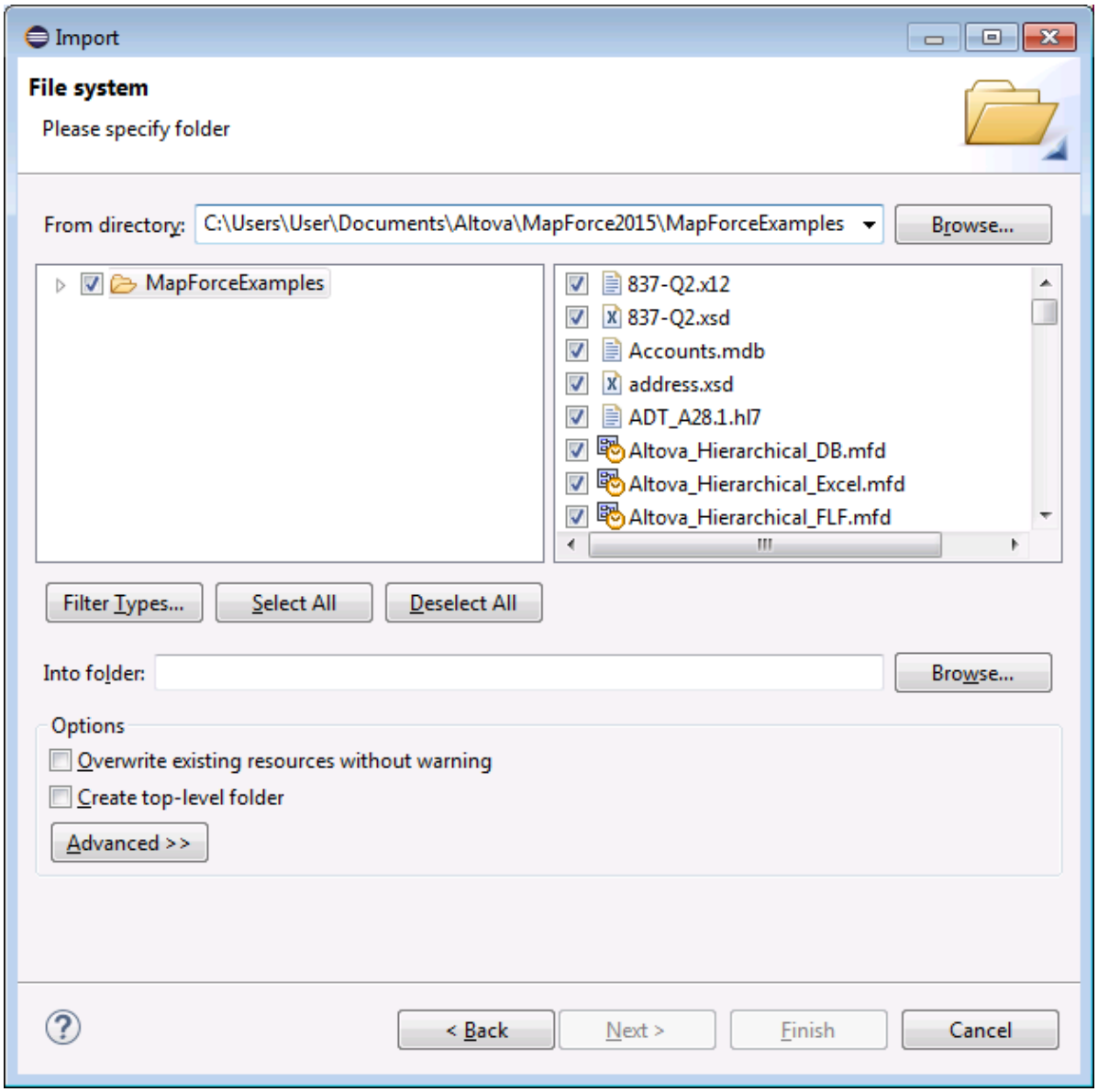
MapForce マッピングと依存ファイルを既存の Eclipse プロジェクトにインポートする:

1. ファイルをインポートするプロジェクトを開きます。
2. 「ファイル」メニューから「インポート」をクリックします。

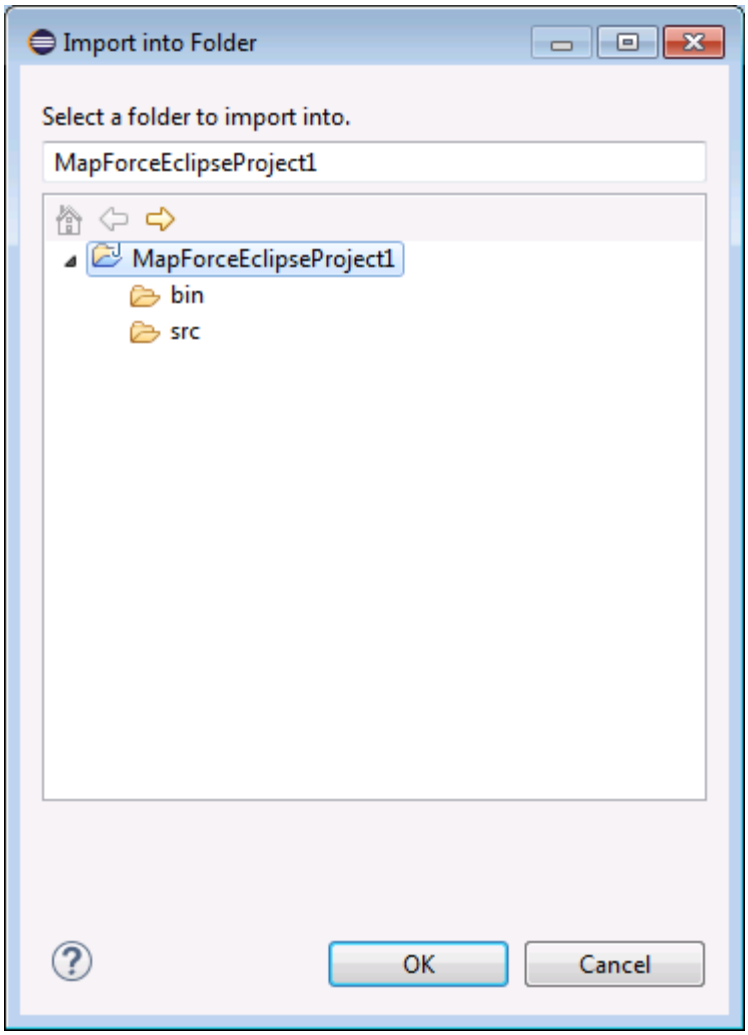


3. 「**ファイルシステム**」を選択して、「**次へ**」 をクリックします。

(C) 2015-2021 Altova GmbH



4. 「次のディレクトリから」の横で、インポートするファイルの場所を参照し、必要なファイルを選択します。
5. 「宛先フォルダー」の横の、「参照」をクリックして、ファイルを追加するプロジェクトを選択します（このサンプルでは、*MapForceEclipseプロジェクト1*）。



6. 「OK」をクリックして、「完了」をクリックします。

14.4.4 自動ビルドの構成と MapForce コードの生成

自動的な MapForce コードのビルドと生成は、MapForce/Eclipse プロジェクト内でデフォルトとして有効化されています (MapForce/Eclipse プロジェクトの作成 を参照してください)。*MapForce/Eclipse* 型ではない既存のプロジェクト内の MapForce コードの自動ビルドとコードの作成を行うには、*MapForce コード生成ビルダー*と *MapForce ネイチャー*を手動で追加して行います。

MapForce コード生成ビルダーをプロジェクトに追加する方法:

- Eclipse **.project** ファイル、以下でハイライトされているラインを追加します:

```
<buildSpec>
    <buildCommand>
       <name>org.eclipse.jdt.core.javabuilder</name>
       <arguments>
```

```
        </arguments>
      </buildCommand>
      <buildCommand>
        <name>com.altova.mapforceeclipseplugin.MapForceBuilder</name>
        <arguments>
        </arguments>
      </buildCommand>
   </buildSpec>
```

MapForce のネイチャーをプロジェクトに追加する:

- Eclipse **.project** ファイル、以下でハイライトされているラインを追加します:

```
   <natures>
      <nature>org.eclipse.jdt.core.javanature</nature>
      <nature>com.altova.mapforceeclipseplugin.MapForceNature</nature>
   </natures>
```

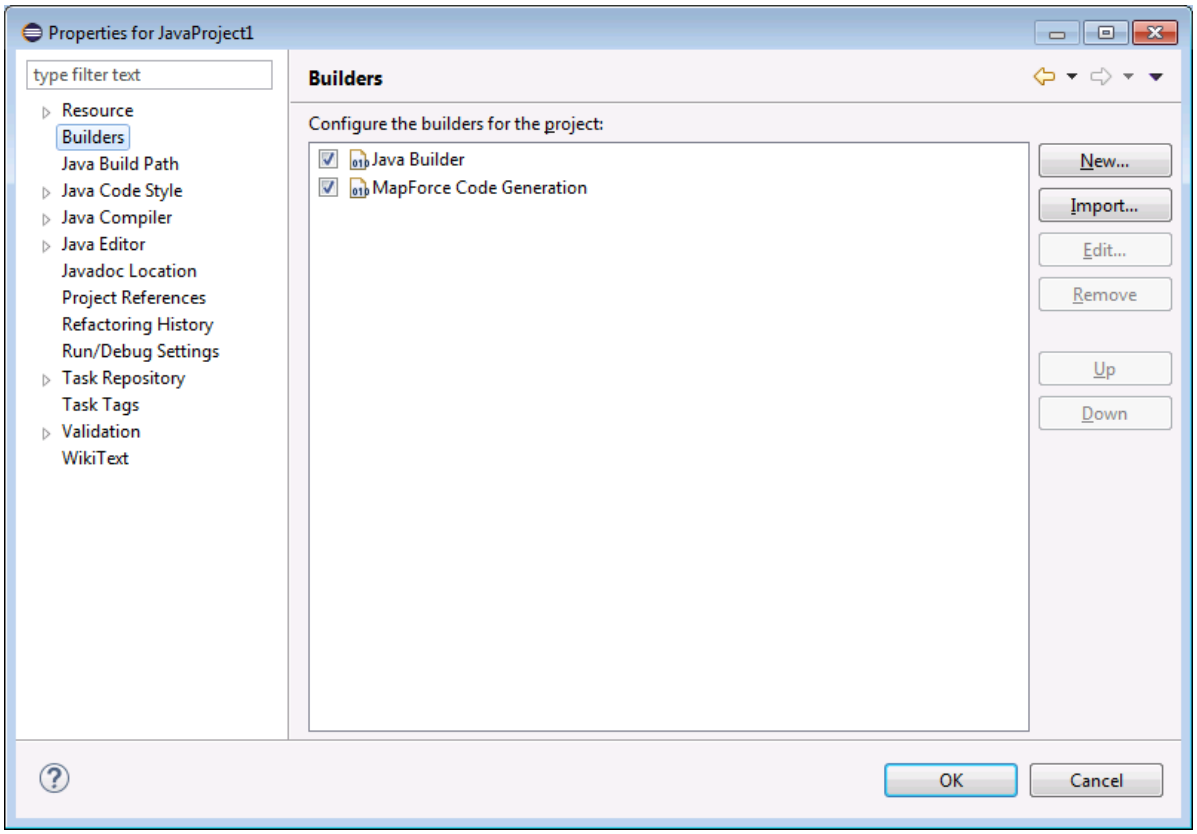
ヒント: **.project** ファイルを素早く Eclipse のナビゲータービューから開くことができます（このビューを有効化するにはメニューコマンド「**ウィンドウ | ビューの表示 | ナビゲーター**」を選択します）。

自動的な MapForce コード生成のオン/オフの切り替え:

- **プロジェクト** メニューから**自動的にビルド**をクリックします。

MapForce コード生成ビルダーの無効化:

1. **プロジェクト** メニューから「**プロパティ**」をクリックします。
2. 「**ビルダー**」をクリックします。



3. 「MapForce コード生成」チェックボックスのチェックを解除するためにクリックします。

(C) 2015-2021 Altova GmbH

14.5 Eclipse のための MapForce プラグインの有効化

Eclipse のためのMapForce プラグインは、ID を持つEclipse 拡張点 **com.altova.mapforceeclipseplugin** を提供します。この拡張点を使用して、MapForce プラグインの機能性を適用または拡張します。拡張点は、MapForce コントロールとMapForce API のCOM-インターフェイス へのアクセスを提供します。

MapForce Eclipse インストール パッケージは、プラグインの簡単なサンプルを含んでいます。新規 MapForce マッピングのファイルを開くイベントをチェックし、マッピングビューのズームのレベルを 70% に設定します。

拡張ポイントのJavaDoc ドキュメントは、MapForce プラグイン インストールディレクトリ（**C:\Program Files\Altova\MapForce2021\eclipse\docs**）で検索することができます。

サンプルMapForce プラグインをインストールして実行する前に、以下の必要条件が満たされていることを確認してください：

- 64-ビット Java を使用している場合は、64-ビット Eclipse、64-ビット MapForce および64-ビット MapForce Integration Package が必要です。
- JDT（Java Development Tools）プラグインのインストール。
- Eclipse PDE（プラグイン 開発環境）のインストール。

サンプルMapForce プラグイン プロジェクトをワークプレースにインポートする:

1. Eclipse を開始します。
2. 「**ファイル**」メニューから「**インポート**」をクリックします。
3. 「**一般 | 既存のプロジェクトをワークプレースへ**」を選択して、「**次へ**」をクリックします。
4. 「ルートディレクトリの選択」フィールドの横の「**参照**」ボタンをクリックして、サンプルプロジェクトディレクトリを選択します。例: **C:\Program Files\Altova\MapForce2021\eclipse\workspace\MapForceExtension.**
5. 「**プロジェクトをワークプレースへコピー**」オプションを選択して、「**完了**」をクリックします。"MapForceExtension" という名前の新規プロジェクトがワークプレイス内に作成されます。

サンプル拡張 プラグインの実行:

1. Java パースペクティブに切り替えます。
2. 「**実行**」メニューから「**構成を実行**」をクリックします。
3. 「**Eclipse アプリケーション**」を右クリックし、「**新規作成**」を選択します。（リスト内に "Eclipse アプリケーション" が存在しない場合、Eclipse プラグイン開発ツールがEclipse 環境にインストールされていないことを意味します。Eclipse プラグインプラグイン開発ツールをインストールするには、「**ヘルプ**」メニュー内の「**新しいソフトウェアをインストールする**」をクリックして、"Eclipse プラグイン開発ツール" を "Eclipse プロジェクト更新" ダウンロードサイトからインストールします。
4. 名前と新しい構成を入力し、（このサンプルでは、*SampleMapForcePlugin*）「**適用**」をクリックします。
5. **MapForceClient** ワークプレイス プラグインが 'プラグイン' タブ内で選択されていることを確認してください。
6. 「**実行**」をクリックして、新しいEclipse ワークベンチが開かれます。
7. MapForce マッピングを新しいワークベンチで開きます。マッピングは、70% のズームレベルで開かれます。

15 メニューレファレンス

以下のセクションでは、MapForce に用意されているメニューやメニューオプションのリストと、それらの簡単な説明を記します。

(C) 2015-2021 Altova GmbH

15.1 ファイル

新規作成

新規マッピングドキュメントまたはマッピングプロジェクト（*.mfp）を開きます。

開く

既に保存されたマッピング（*.mfd）またはマッピングプロジェクト（*.mfp）ファイルを開きます。使用中のMapForce エディションで使用できない機能が含まれているファイルは、開くことができません。

上書き保存

現在アクティブなファイル名で、現在アクティブになっているマッピングを保存します。

名前をつけて保存

現在アクティブになっているマッピングを、別の名前で保存します。まだ保存されていないファイルの場合、ファイルに新たな名前をつけることができます。

全て保存

現在開かれている全てのマッピングファイルを保存します。

再ロード

現在アクティブになっているマッピングを再ロードします。それまでに作成された変更点を破棄してもよいか尋ねられます。

閉じる

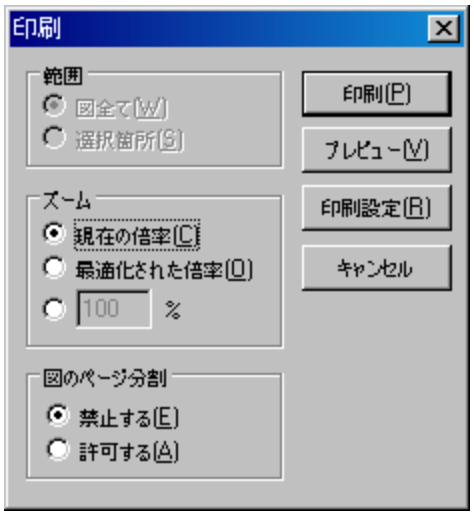
現在アクティブになっているマッピングを閉じます。ファイルを閉じる前に保存するか尋ねられます。

全て閉じる

現在開かれている全てのマッピングファイルを閉じます。保存されていないファイルを保存するか尋ねられます。

印刷

印刷ダイアログボックスが開かれ、マッピングを印刷することができます。



印刷ダイアログボックス

「**現在の倍率**」を選択すると、現在マッピングに対して指定されているズーム倍率が使用されます。「**最適化された倍率**」を選択すると、マッピングがページの大きさに収められるよう、印刷が最適化されます。コンポーネントのスクロールバーは印刷されません。1 つのコンポーネントが複数のページに切り分けて印刷してもよいか指定することもできます。

印刷プレビュー

上に記述された機能を持つ印刷ダイアログボックスが表示されます。

印刷設定

印刷設定ダイアログボックスが表示され、使用するプリンターや用紙の設定などを指定することができます。

マッピングの検証

マッピングを検証により、全てのマッピング(接続)が妥当であるかと、エラーや警告が表示されます。
詳細については[マッピングの検証]を参照ください。

マッピングの設定

マッピングの設定ダイアログボックスではドキュメントごとの設定を定義することができます。[マッピングの設定の変更] を参照してください。

資格情報マネージャーを開く

基本的な HTTP 認証または OAuth 2.0 認証を実行するマッピング内で必要とされる資格情報を管理する [資格情報マネージャー] を開きます。

選択された言語でコードを生成する

現在選択されているマッピングの言語によりコードが生成されます。現在選択されている言語は、ツールバーでハイライトされているプログラミング言語のアイコン(XSLT、XSLT2、XSLT3、XQuery、Java、C#、または C++ で確認することができます。

コード生成 | XSLT (XSLT2、XSLT3)

このコマンドにより、ソースファイルからの変換に必要な XSLT ファイルが生成されます。このオプションを選択すると、フォルダーの参照ダイアログボックスが表示され、生成された XSLT ファイルを配置する場所を指定することができます。[マッピング設定の変更] を参照してください。

(C) 2015-2021 Altova GmbH

コード生成 | XQuery

このコマンドにより、ソースファイルからの変換に必要な XQuery ファイルが生成されます。このオプションを選択すると、フォルダーの参照ダイアログ ボックスが表示され、生成された XQuery ファイルを配置する場所を指定することができます。マッピング設定の変更 を参照してください。

コード生成 | Java | C# | C++

これらのコマンドにより、ソースファイルから変換を行うのに必要なアプリケーションプログラムのコードが生成されます。このコマンドを選択すると、フォルダーの参照ダイアログボックスが表示され、生成されたファイルが配置される場所を指定することができます。生成されるアプリケーションファイル（ならびにプロジェクトファイル: *.csproj C# プロジェクトファイル、*.sln ソリューションファイル、*.vcproj Visual C++ プロジェクトファイル）の名前は、マッピング設定ダイアログボックスのアプリケーション名フィールドにより定義されます。メニューオプションの「ファイル | マッピングの設定」を選択することで、このダイアログへアクセスすることができます。マッピング設定の変更 を参照してください。

ターゲットが XML/スキーマドキュメントの場合、コードの実行により作成されるファイル名は、コンポーネント設定ダイアログボックスの出力 XML ファイルフィールドにて与えられている名前となります。

MapForce Server 実行ファイルのコンパイル

マッピングの変換を実行するために MapForce Server で実行されるファイルを生成します。MapForce マッピングのコンパイル を参照してください。

FlowForce Server へデプロイ

現在アクティブなマッピングを FlowForce Server へデプロイします。MapForce マッピングのデプロイ を参照してください。

ドキュメントの生成

様々な出力フォーマットにより、マッピングプロジェクトの詳細なドキュメンテーションを生成します（マッピングドキュメントの生成とカスタム化 を参照してください）。

最近のプロジェクト

最近開かれたプロジェクトのリストを表示します。

終了

アプリケーションを終了します。保存されていないドキュメントがある場合、それらを保存するか尋ねられます。

15.2 編集

このメニュー以下にあるコマンドの殆どは、出力タブにてマッピングの結果を参照している時、またはXSLT タブにてXSLT コードのプレビューを行なっている際に利用することができます。

元に戻す

MapForce では使用回数に制限の無い「元に戻す」操作を行うことができ、マッピングのデザインステップを順を追って確認することができます。

やり直し

やり直しコマンドにより、元に戻すコマンドにより戻された動作を進めることができます。これらのコマンドを使用することで、操作の履歴を確認することができます。

検索

XSLT、XSLT2、XSLT3、XQuery または出力タブにて、指定されたテキストを検索することができます。

次を検索 F3

同じ検索文字列が次に出現する箇所を検索することができます。

前を検索 Shift F3

同じ検索文字列が前に出現する箇所を検索することができます。

切り取り/コピー/貼り付け/削除

標準的な Windows の編集コマンドです。マッピングウィンドウ内に表示されているコンポーネントや関数を切り取ったりコピーすることができます。

すべて選択

マッピングタブにある全てのコンポーネントを選択する、または**XSLT、XSLT2、XSLT3、XQuery** または出力タブに表示されているテキスト/コード全体を選択することができます。

(C) 2015-2021 Altova GmbH

15.3 挿入

XML スキーマ / ファイル

XML スキーマまたはインスタンスをマッピングに追加します。スキーマを参照するXML ファイルを選択すると、追加情報をマッピングに追加する必要はありません。スキーマ参照のないXML ファイルを選択すると、自動的に一致するXML スキーマを生成するかを問われます。(XML スキーマの生成 を参照してください)。XML スキーマファイルを選択すると、プレビューのためのデータを提供するXML インスタンスを任意で追加するかを問われます。

データベース

データソースまたはターゲットコンポーネントとしてデータベースコンポーネントを挿入することができます。データベースによりスキーマ構造が与えられ、ツリー形式により表示されます。

EDI

データソースまたはターゲットコンポーネントとして使用することができるEDI ドキュメントを挿入することができます。詳細についてはEDI のセクションを参照ください。

テキストファイル

フラットなテキストファイルドキュメント(例: CSV)、または固定長のテキストファイルを挿入することができます。両方の種類とも、ソースならびにターゲットコンポーネントとして使用することができます。更に、CSV または固定長とは異なる構造でテキストファイルを処理する場合、を使用することができます。(MapForce FlexText を参照してください。

Web サービス関数

WSDL ファイルにより定義されたWeb サービス関数を挿入します。入力ならびに出力コンポーネントへ接続することで、出力ウィンドウにてWeb サービス関数の結果を素早く確認することができます。詳細については" Web サービスの呼び出し" を参照ください。

Excel 2007+ ファイル

Microsoft Excel 2007+ (.xlsx) ファイルをマッピングに追加します(Microsoft OOXML Excel 2007+ を参照)。If you don't have Excel 2007 を使用していない場合、Excel 2007+ からマップ、またはExcel 2007+ へマップすることができます。この場合、出力タブで結果をプレビューすることはできませんが、出力メニューの出力ファイルの保存をクリックしてファイルを保存することができます。

XBRL ドキュメント

XBRL インスタンスまたはタクソノミドキュメントを挿入します。タクソノミならびにインスタンスファイルのマッピングに関する詳細については、XBRL マッピング を参照ください。

JSON スキーマ/ファイル

JSON スキーマまたはファイルを挿入します。詳細に関しては、JSON ファイルをマッピングコンポーネントとして追加するを参照してください。

プロトコルバッファーファイル

マッピングにプロトコルバッファーフォーマットのバイナリファイルを追加します。プロトコルバッファーを参照してください。

入力の挿入

マッピングウィンドウがマッピングを表示すると、このコマンドにより入力コンポーネントをマッピングに追加することができます。(マッピングにパラメーターを与える を参照してください)。マッピングウィンドウがユーザー定義関数を表示すると、このコマンドはユーザー定義関数の入力コンポーネントを追加します。(ユーザー定義関数内のパラメーター を参照してください)。

出力の挿入

マッピングウィンドウがマッピングを表示すると、このコマンドにより出力コンポーネントをマッピングに追加することができます(マッピングから文字列の値を返す を参照してください)。マッピングウィンドウがユーザー定義関数を表示すると、このコマンドはユーザー定義関数の出力コンポーネントを追加します。 ユーザー定義関数内のパラメーター を参照してください。

定数

コネクタ に固定のデータを提供する定数を挿入します。コンポーネントを作成する際にデータはダイアログボックスに入力されます。以下のデータ型から選択することができます: 文字列、数値、その他から選択することができます。

変数

標準的な(非インライン型の) ユーザー定義関数と等価な中間変数を挿入します。変数はインスタンスファイルを持たない構造コンポーネントで、マッピング処理を単純にするために使用されます。詳細については 中間変数 を参照ください。

並べ替え: node/row

ノードの並べ替えを許可するコンポーネントを挿入します。詳細に関しては、ノード/行の並べ替えを参照してください。

フィルター: node/row

node/row とbool、そしてon-true とon-false という2 つの入力と出力パラメーターを備えたコンポーネントを挿入します。Boolean がtrue となっている場合、node/row パラメーターの値がon-true パラメーターから渡され、Boolean がfalse となっている場合、node/row パラメーターの値がon-false パラメーターへ渡されます。フィルターの使用方法については フィルターと条件を参照してください。

SQL-WHERE/ORDER

データベースのデータに対して使用することができるフィルターコンポーネントを挿入することができます。MapForce により生成されたクエリに対して、任意のSQL W HERE 句を追加することができます。詳細については SQL W HERE / ORDER コンポーネント を参照ください。

Value-Map

ルックアップテーブルを使って入力された値を特定の値へ変換するコンポーネントを挿入します。(例えば、月の数値を月の名前にマップするなど)他の値のセットに値のセットをマップする場合に役に立ちます。詳細については Value-Map の使用 を参照ください。

IF-Else 条件

"If-Else 条件"のかたのコンポーネントを挿入します。詳細に関しては、 フィルターと条件 を参照してください。

(C) 2015-2021 Altova GmbH

例外

例外コンポーネントによりマッピングの処理を、特定の条件が満たされた場合に中断することができます。WSDL マッピングプロジェクトを使用してフォールトメッセージを定義することができます。例外の追加とWeb サービスのフォールトを参照してください。

15.4 プロジェクト

MapForce では、マルチドキュメントインターフェースがサポートされており、複数のマッピングを 1 つのマッピングプロジェクトへまとめることができます。詳細に関しては、マッピングプロジェクトとの作業 を参照してください。

プロジェクトの再ロード

現在アクティブなプロジェクトを再ロードして、プロジェクトタブを表示します。

プロジェクトを閉じる

現在アクティブになっているプロジェクトを閉じます。

プロジェクトの保存

現在アクティブになっているプロジェクトを保存します。

プロジェクトへファイルを追加する

ファイルを開くダイアログボックスにより、現在開かれているプロジェクトにマッピングを追加することができます。

アクティブなファイルをプロジェクトへ追加する

現在アクティブなファイルを、現在開かれているプロジェクトに追加することができます。

フォルダーの作成

このオプションにより、現在開かれているプロジェクト構造へ、新たなフォルダーが追加されます。このメニューオプションは、使用できるときだけ選択可能になります。詳細に関しては、プロジェクトフォルダーの管理を参照してください。

マッピングを開く

プロジェクトタブにて現在ハイライト/選択されているマッピングを開きます。

オペレーションのためにマッピングを作成する

現在選択されているW SDL プロジェクトのオペレーションに対してマッピングを作成します。名前を付けて保存ダイアログボックスが自動的に表示され、W SDL ファイル内で定義されているオペレーション名が与えられます

オペレーションにマッピングファイルを追加する

既に保存されているマッピングファイルを現在アクティブな W SDL オペレーションへ追加することができます。ファイルを開くダイアログボックスからマッピングファイルを選択してください。

W eb サービスの挿入...

既存のW SDL ファイルをベースにW eb サービスを挿入することができます。

XMLSpy のファイルを開く

XMLSpy のプロジェクトウィンドウで選択されているW SDL ファイルを XMLSpy で開きます。

(C) 2015-2021 Altova GmbH

プロジェクト全体のコードの生成

プロジェクトウィンドウに現在表示されているプロジェクト全体に対してプロジェクトコードを生成します。現在選択されているデフォルトの言語で、各フォルダーにある*.mfd マッピングファイルのコードが生成されます。

コードの生成..

サブメニューから選択した言語にてプロジェクトコードを生成します。

プロパティ

プロジェクト全体の設定を定義することのできるダイアログボックスを開きます。詳細に関しては、コード生成設定の設定を参照してください。

最近のプロジェクト - 1. 2. etc.

最近に開かれたプロジェクトをリストします。

15.5 コンポーネント

ルート要素の変更

XML インスタンスドキュメントのルート要素を変更することができます。

XMLSpy でスキーマ定義を編集

XML スキーマコンポーネントが選択された状態でこのオプションを選択すると、XMLSpy のスキーマビューで XML スキーマファイルが開かれ、編集を行うことができます

FlexText 構成の編集

FlexText を開き、選択されている FlexText ファイルを編集することができます。

データベースオブジェクトの追加/削除/編集

データベースコンポーネント内のデータベースオブジェクトを追加、削除または変更することができます。

EDI X12 997 へのマッピングを作成

X12 997 Functional Acknowledgement により、EDI インターチェンジのステータスが報告されます。ドキュメントを処理する際に発生したエラーは全て、この中で報告されます。X12 997 Functional Acknowledgment の生成 を参照ください。

EDI X12 999 へのマッピングを作成

X12 999 Implementation Acknowledgment Transaction Set により、HIPAA 実装ガイド不順守、またはアプリケーションエラーが報告されます。企業などの組織に送信された各 EDI トランザクションは、999 トランザクションを送信することで応答する必要があります。詳細については X12 999 Implementation Acknowledgment の生成 を参照ください。

更新

現在アクティブになっているデータベースコンポーネントの構造を、データベースから再ロードします。

前に入力を複製して追加

選択されたアイテムのコピー/クローンを、現在選択されているアイテムの前に挿入します。コピーされたアイテムには出力アイコンが含まれておらず、データ構造として使用することはできません。この機能を使用した例については 複数のノースから1 つのターゲットにマップ セクションを参照ください。複製されたアイテムを右クリックすると、コンテキストメニューから複製アイテムを移動することができます。必要に応じて複製されたアイテムを上/下へ移動させてください。

後に入力を複製して追加

選択されたアイテムのコピー/クローンを、現在選択されているアイテムの後に挿入します。コピーされたアイテムには出力アイテムが含まれておらず、データ構造として使用することはできません。この機能を使用した例については 複数のノースから1 つのターゲットにマップ セクションを参照ください。複製されたアイテムを右クリックすると、コンテキストメニューから複製アイテムを移動することができます。必要に応じて複製されたアイテムを上/下へ移動させてください。

複製を削除

既に作成された複製アイテムを削除します。この機能を使用した例については 複数のノースから1 つのターゲットにマップ セクションを参照ください。

(C) 2015-2021 Altova GmbH

データベーステーブルのアクション

選択されているターゲットデータベーステーブルへ、データがマッピングされた際に取られるアクションを定義することができます。詳細については[データベーステーブルアクションの設定](#) のセクションを参照ください。

データベースのクエリ

データベースコンポーネント内で選択されているテーブル/フィールドをベースに、SELECT ステートメントを作成します。テーブル/フィールドが選択されている状態でこのコマンドが使用可能になり、SELECT ステートメントがSELECT ウィンドウに自動的に配置されます。

ツリーを左揃えにする

全てのアイテムをコンポーネントの左端へ揃えます。

ツリーを右揃えにする

全てのアイテムをコンポーネントの右端へ揃えます。この表示方法はターゲットスキーマへのマッピングを行う際に利用することができます。

プロパティ

現在選択されているコンポーネントの設定を表示するダイアログボックスを表示します。[コンポーネント設定を変更する](#)を参照してください。

15.6 接続

子要素の自動接続

子要素の自動接続機能を有効化または無効化します。ツールバーにあるアイコンからも選択を行うことができます。

子要素の接続設定

一致する子要素の接続設定ダイアログボックスが表示され、接続設定を行うことができます。(一致する子要素の接続を参照)。

一致する子要素の接続

このコマンドにより、ソースとターゲットスキーマ間で同じ名前を持つ複数のアイテムに対して接続を作成することができます。このダイアログボックスで定義した設定はそのまま保持され、ツールバーにある**子要素の自動接続アイコン** が有効になっている状態で2 つのアイテムを接続した際に適用されます。このアイコンをクリックすることで、アイコンの有効/無効状態を切り替えることができます。詳細については一致する子要素の接続のセクションを参照ください。

標準マッピング(ターゲット優先マッピング)

接続の種類を標準的なマッピングへ変更します。詳細については標準マッピング(ターゲット優先マッピング)を参照してください。

全てコピー(子要素)

親コネクタのサブツリーとして表示されている子アイテムのコネクタが存在する状況で、マッチする全ての子要素に対してコネクタを作成します。詳細については すべてコピー接続 を参照ください。

ソース優先マッピング(混合コンテンツ)

接続をソース優先 / 混合コンテンツ型のコネクタへ変更して、新たに要素を選択してマッピングすることができるようになります。詳細に関しては、ソース優先マッピング(混合コンテンツ)を参照してください。

プロパティ

現在のコネクタに対して固有(混合コンテンツ)の設定を定義するためのダイアログボックスが表示されます。利用することができないオプションはグレーアウトされます。これらの設定はテキストノードを持たない複合型のアイテムに対しても適用されます。詳細は接続設定を参照してください。

(C) 2015-2021 Altova GmbH

15.7 関数

ユーザー定義関数の作成

新たなユーザー定義関数を作成します。(詳細については ユーザー定義関数 を参照ください)。

選択からユーザー定義関数作成

マッピングウィンドウにて現在選択されている要素をベースに、新たなユーザー定義関数を作成します。

関数設定

現在アクティブなユーザー定義関数の設定ダイアログボックスを開き、現在の設定を変更することができます。

関数削除

既存のユーザー定義関数を開き、その関数の名前がタブに表示されている状態で、現在アクティブなユーザー定義関数を削除します。

入力の挿入

マッピングウィンドウがマッピングを表示すると、このコマンドは、マッピングに入力コンポーネントを追加します(詳細については 単純型入力 を参照ください)。マッピングウィンドウがユーザー定義関数を表示すると、このコマンドは、マッピングに入力コンポーネントをユーザー定義関数に追加します(詳細については ユーザー定義関数内のパラメーター を参照ください)。

出力の挿入

マッピングウィンドウがマッピングを表示すると、このコマンドは、マッピングに出力コンポーネントを追加します(詳細については 単純型出力 を参照ください)。マッピングウィンドウがユーザー定義関数を表示すると、このコマンドは、マッピングに出力コンポーネントをユーザー定義関数に追加します(詳細については ユーザー定義関数内のパラメーター を参照ください)。

15.8 出力

XSLT 1.0、XSLT 2.0、XSLT 3.0、XQuery、Java、C#、C++、Built-in 実行エンジン
マッピングが実行される変換言語を設定します。変換言語の選択を参照してください。

出力ファイルの検証
参照されているスキーマに対して出力 XML ファイルを検証します。

生成された出力の保存
出力タブに現在表示されているデータを保存します。

生成されたすべての出力を保存
動的マッピングにより生成された出力をファイルに保存します。詳細については 複数の入力または出力ファイルを動的に処理 を参照ください。

出力を再生成
現在のマッピングを出力ウィンドウから再生成します。

SQL スクリプトを実行
SQL スクリプトが現在出力ウィンドウで表示されている場合、テーブルアクションの定義に従ったかたちで、ターゲットデータベースに対してマッピングを実行します。

ブックマークの挿入/削除
出力ウィンドウのカーソル位置にブックマークを挿入します

次のブックマーク
出力ウィンドウ内にある次のブックマークへ移動します。

前のブックマーク
出力ウィンドウ内にある前のブックマークへ移動します。

全てのブックマークを削除
出力ウィンドウにて現在定義されている全てのブックマークを削除します。

XML テキストの整形
出力ペインに表示されている XML ドキュメントの表示を、ドキュメントの構造に従ったかたちに再構成します。各子ノードが、親ノードよりインデントされたかたちで表示されます。この機能では、タググループにて指定されたタブサイズの設定（タブまたはスペースを挿入）が使用されます。

(C) 2015-2021 Altova GmbH

テキストビューの設定

テキストビュー設定ダイアログボックスを表示します。このダイアログボックスにより出力 ペイン、**XSLT** ペイン、と**XQuery** ペインでテキストビュー設定をカスタマイズすることができます。またこのダイアログボックスは、現在ウィンドウ内で適用されている定義済みのホットキーを表示しています。詳細に関しては、次を参照してください テキストビュー機能。

15.9 デバッグ

デバッグの開始 (F11)

デバッグを開始または、ブレークポイントに到達またはマッピングが完了するまで継続します。

デバッグの停止 (Shift + F5)

デバッグを停止します。このコマンドはデバッグモードを終了して、MapForce を標準モードに切り替えます。

ステップイン (F11)

マッピング内で単一のステップを終了するまでマッピングを実行します。マッピングでデバッガーは、シーケンスの単一アイテムを生成する独立した計算の理論的なグループです。

マッピングコンテキストにより、このコマンドは「左へ移動/ターゲットに移動 子/ソースの親へ移動」 と解釈することができます。

ステップオーバー (F10)

現在のステップが終了するまで、または他のシーケンスのアイテムを終了するまで実行を継続します。このコマンドは、現在のステップの入力である計算をステップオーバーします。

ステップアウト (Shift + F11)

現在のステップの結果が使用されるまで、または入力または使用の子ではないステップが実行されるまで実行を継続します。このコマンドは現在の計算をステップアウトします。

マッピングコンテキストにより、このコマンドは「右へ移動/ターゲットへ移動 親/ソースの子へ移動」と解釈することができます。

最小ステップ (Ctrl + F11)

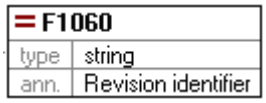
値が生成または使用されるまで、実行を継続します。このコマンドは、ステップを細分化し、接続ごとに通常2回停止します: ソースが値を生成した際、ターゲットが値を使用した際。MapForce は、マッピングの提示に従い値を計算する必要はなく、生成と使用イベントは順序に従わない場合があります。

(C) 2015-2021 Altova GmbH

15.10 表示

注釈の表示

XML スキーマの注釈（ならびにEDI 情報）をコンポーネントウィンドウにて表示します。
型の表示アイコンも有効になっている場合、両方の情報がグリッド形式で表示されます。



型の表示

各要素または属性に対してスキーマのデータ型を表示します。
注釈の表示アイコンも有効になっている場合、両方の情報がグリッド形式で表示されます。

関数ヘッダーにライブラリ名を表示

関数タイトルに括弧付きでライブラリ名を表示します。

ヒントの表示

マウスポインターが関数の上に配置された時に、ツールチップ内に関数の説明が表示されます。

XBRL 表示オプション

詳細に関しては、XBRL ラベル表示オプションの構成を参照してください。

選択されたコンポーネントの接続線の表示

すべてのマッピングの接続線または、現在選択されているコンポーネントに関連した接続線を切り替えることができます。

ソースからターゲットへの接続線表示

以下のように接続線の表示を切り替えることができます:

- 現在選択されているコンポーネントに直接接続されているコネクタ
- 現在選択されているコネクタで、ソースからターゲットコンポーネントまで

ズーム

ズームダイアログボックスが表示されます。ズームの倍率を手動で入力することができるほか、スライダーをドラッグして、ズームの倍率を変更することができます。

戻る

マッピングタブにて現在開かれているマッピングで、前に表示されていたマッピングを表示します。

進む
マッピングタブにて現在開かれているマッピングで、戻るコマンドにより表示される前のマッピングを表示します。

ステータスバー
メッセージウィンドウの下に表示されているステータスバーを表示/非表示にします。

ライブラリウィンドウ
ライブラリ関数が含まれるライブラリウィンドウを表示/非表示にします。

ライブラリの管理
ライブラリの管理ウィンドウを表示/非表示にします。

メッセージウィンドウ
検証出力ウィンドウを表示/隠します。コードの生成を行う場合、メッセージ出力ウィンドウが自動的に有効になり、検証結果が表示されます。

概要ウィンドウ
概要ウィンドウを表示/非表示にします。長方形をドラッグすることで、マッピングの表示範囲を移動することができます

プロジェクトウィンドウ
プロジェクトウィンドウを表示/非表示にします。

(C) 2015-2021 Altova GmbH

15.11 ツール

グローバルリソース

グローバルリソースの管理ダイアログボックスを表示して、グローバルリソースXMLファイル内にあるグローバルリソース情報を追加、編集、または削除することができます。詳細については グローバルリソース - プロパティ を参照ください。

アクティブな構成

現在アクティブになっているグローバルリソースを、グローバルリソースの構成リストから選択/変更することができます。サブメニューから目的の構成を選択してください。

反転マッピングの作成

MapForce で現在アクティブになっているマッピングをベースに、「反転した」マッピングを作成します。反転した結果生成されるマッピングは完成したものではなく、コンポーネント間における直接接続だけが保持されるという点に注意してください。恐らく反転したマッピングは妥当なものではなく、更なる編集を行うこと無く出力タブをクリックしても、実行されません。

マッピングを反転させると、ソースコンポーネントがターゲットコンポーネントになり、ターゲットコンポーネントがソースコンポーネントになります。入力または出力 XML インスタンスファイルがコンポーネントに割り当てられている場合、これらも交換されます。

以下のデータが保持されます:

- コンポーネント間の直接接続
- チェーンマッピング内のコンポーネント間の直接接続
- 接続の種類: 標準、混合コンテンツ、全てコピー
- パススルーコンポーネント設定
- データベースコンポーネント

以下のデータが保持されません

- 関数やフィルターなどを経由した接続は、関数などとともに削除されます
- ユーザー定義関数
- Web サービスコンポーネント

ツールバーとウィンドウの復元

ツールバーやドックウィンドウなどをデフォルトの状態にリセットします。変更を反映するには、MapForce を再起動する必要があります。

カスタム化...

MapForce グラフィカルなユーザーインターフェイスをカスタム化することのできるダイアログボックスを開きます。これにはツールバーの表示、非表示、およびメニューとキーボードのショートカットのカスタム化が含まれます。

オプション

デフォルトのMapForce 設定を変更することのできるダイアログボックスを開きます（MapForce オプションの変更 を参照してください）。

15.12 ウィンドウ

重ねて表示

開かれている全てのウィンドウを重ねて表示されるように再配置します。

上下に並べて表示

開かれている全てのウィンドウを上下に並べて表示することで、同時に表示されるように再配置します。

左右に並べて表示

開かれている全てのウィンドウを左右に並べて表示することで、同時に表示されるように再配置します。

1
2
現在開かれているウィンドウが表示され、これらウィンドウ間で素早い切り替えを行うことができます。
Ctrl + tab または Ctrl + F6 キーボードショートカットを使用することでも、開かれているウィンドウを切り替えることができます。

(C) 2015-2021 Altova GmbH

15.13 ヘルプメニュー

- 目次
  - *説明*

    ヘルプウィンドウの左側のペインに目次を表示した、MapForce の画面上のヘルプマニュアルを開きます。目次はヘルプドキュメント全体の概要を表示しています。目次のエントリをクリックしてトピックに移動することができます。

- インデックス
  - *説明*

    ヘルプウィンドウの左側のペインにキーワード インデックスを表示したMapForce の画面上のヘルプマニュアルを開きます。目次はヘルプドキュメント全体の概要を表示しています。インデックスはキーワードをリストし、キーワードをダブルクリックすることでトピックへ移動することができます。キーワードが 1 つ以上のトピックにリンクされている場合は、トピックのリストが表示されます。

- 検索
  - *説明*

    ヘルプウィンドウの左側のペインに検索ダイアログを表示したMapForce の画面上のヘルプマニュアルを開きます。単語を検索するには、入力フィールドに検索対象を入力して、(i) 「Return」 を押す、または(ii) 「**トピックのリスト**」をクリックします。 を押します。ヘルプシステムは、ヘルプドキュメント全体で全文検索を行いヒットしたリストを返します。アイテムを表示するためにはアイテムをダブルクリックします。

- ソフトウェアのライセンス認証
  - *説明*

    Altova 製品ソフトウェアをダウンロードすると、無料評価キーまたは購入されたライセンスキーを使用して、製品にライセンスを供与、または、ライセンスの認証を行うことができます。

    - *無料 評価ライセンス*初めて製品のダウンロードとインストールを行うと、ソフトウェアライセンス認証ダイアログが表示されます。ダイアログでは無料 評価 ライセンスをリクエストすることができます。ユーザーの名前、所属会社名、そして電子メールアドレスを表示されるダイアログに入力し **リクエスト** をクリックします。ライセンスファイルが入力された電子メールアドレスに送信されます。この手順には数分を要します。ライセンスファイルを適切な場所に保存します。**リクエスト** をクリックすると、リクエストダイアログの下に入力フィールドが表示されます。このフィールドはライセンスファイルの パスを取ります。ライセンスファイルを参照 またはライセンスファイルへの パスを入力し「OK」をクリックします。(「ソフトウェアのライセンス認証ダイアログ」内で **新規のライセンスをアップロードする**」 をクリックしてライセンスファイルへの パスを入力するダイアログにアクセスすることができます。)ソフトウェアは30 日の間アンロックされます。
    - *永続的なライセンスキー*:ソフトウェアライセンス認証ダイアログには永続的なライセンスキーを購入するためのボタンが含まれています。このボタンをクリックすると、製品の永続的なライセンスキーを購入することのできる Altova オンラインショップに移動することができます。受信する電子メールはライセンスデータを含むライセンスファイルの書式で送信されています。3 つの種類の永続的なライセンスが存在します: インストール済み、同時使用ユーザー、名前を持つユーザー。*インストール済みのライセンス* は単一のコンピューター上のソフトウェアのロックを解除します。*インストール済みのライセンス* を N 台のコンピューターのために購入すると N 台までのコンピューター上でソフトウェアを使用することができます。*同時使用ユーザーライセンス* は N 人の同時使用ユーザーに N 人のユーザーに同時にソフトウェアを使用することを許可します 。(10N 台のコンピューターにソフトウェアをインストールすることができます)。 *名前を持つユーザーライセンス*は*特定のユーザー*が 5 台の異なるコンピューター上でソフトウェアを使用することを許可します。ソフトウェアのライセンスを認証するには「**新規のライセンスをアップロードする**」 をクリックして表示されるダイアログ内でライセンスファイルを参照、またはライセンスファイルの パスを入力し「OK」をクリックします。

メモ 複数のユーザーライセンスのために各ユーザーは各自の名前を入力するようにプロンプトされます。

ライセンスの電子メールとAltova 製品へのライセンス供与(有効化) の異なる方法
Altova から受信するライセンス電子メールにはライセンスファイルが添付ファイルとして含まれています。ライセンスファイルは `.altova_licenses` ファイル拡張子を有しています。

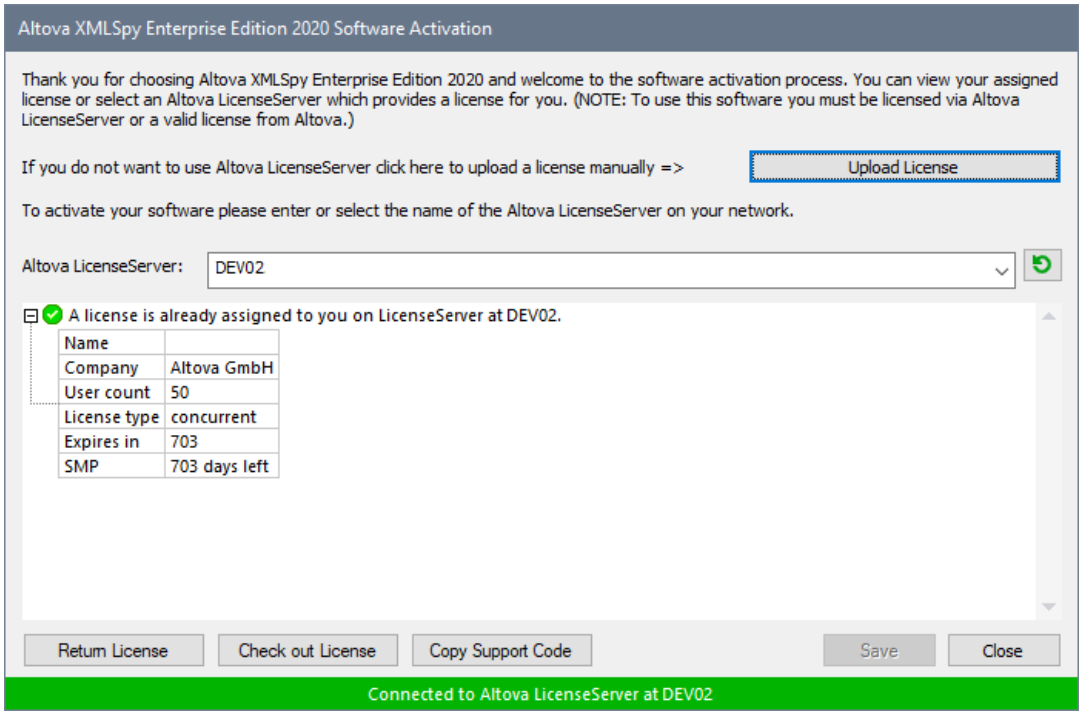
Altova 製品のライセンスを認証するには、以下のうち1つ行ってください:

- 適切な場所にライセンスファイル(`.altova_licenses`) を保存し、ライセンスファイルをダブルクリックし、表示されるダイアログに必要な情報を入力し、**「キーの適用」**をクリックして完了します。
- ライセンスファイル(`.altova_licenses`) を適切な場所に保存します。Altova 製品内では、**「ヘルプ | ライセンス登録メニューコマンド**を選択し、**新規のライセンスをアップロード**します。ライセンスファイルへのパスを入力し「OK」をクリックします。
- 適切な場所にライセンスファイル(`.altova_licenses`) を保存し、Altova LicenseServer のライセンスプールにアップロードします。以下を行うことができます: (i) 製品のソフトウェアライセンス認証ダイアログから Altova 製品からライセンスを取得します。*(以下を参照)*または(ii) Altova LicenseServer から製品へのライセンスを割り当てます。*LicenseServer の使用の詳細に関しては、下記のトピックを参照してください*。

ソフトウェアライセンス認証ダイアログ(*下のスクリーンショット*) は、**「ヘルプ | ソフトウェア アクティベーション」**をクリックすることにより常にアクセスすることができます。

以下の方法によりソフトウェアをアクティブ化することができます:

- *ソフトウェアライセンス認証ダイアログでライセンスを登録する方法*。ダイアログ内で、**「新規のライセンスをアップロード」**をクリックして、ライセンスファイルを参照し選択します。「OK」をクリックしてライセンスファイルへのパスを確認します(複数のユーザーライセンスの場合は個人の名前です)。**「保存」**をクリックして完了します。
- *ネットワーク上のAltova LicenseServer を使用してライセンス供与する方法:* ネットワーク上の Altova LicenseServer を使用してライセンスを取得するにはソフトウェアのライセンスの認証ダイアログの下にある **Altova LicenseServer を使用する**をクリックします。使用する LicenseServer がインストールされているマシンを選択します。License Servers の自動検知は LAN 上で配信が送信されることを意味します。これらの配信がサブネットに制限されているため、License Server は自動検知のためのクライアントマシンと同じサブネット上に存在する必要があります。自動検知が作動しない場合、サーバーの名前を入力します。Altova LicenseServer はライセンスプール内で Altova 製品のためのライセンスを有している必要があります。LicenseServer プール内に存在する場合、ソフトウェアライセンス認証ダイアログ内に表示されます(*Altova XMLSpy 内のダイアログで表示されている例を参照してください*)。**「保存」**をクリックしてライセンスを取得します。



マシン固有のライセンスがLicenseServer からインストールされると、7日間は、LicenseServer に戻すことができません。7日過ぎると、(「**ライセンスを戻す**」をクリックして) マシンのライセンスをLicenseServer に戻すことができ、このライセンスは、他のクライアントによりLicenseServer から取得することができます。LicenseServer 管理者は、LicenseServer のW eb UI を使用して、取得されたライセンスの割り当てを解除することができます。ライセンスの返却は、マシン固有のライセンスのみに適用され、現在使用中のライセンスには適用されないことにご注意ください。

ライセンスのチェックアウト

ライセンスが製品マシン上に保管されるように、ライセンスをライセンスプールから30 日間チェックアウトすることができます。これにより、オフラインで作業することが可能になります。この機能はとても役に立ちます。Altova LicenseServer にアクセスできない環境(例えば、旅行中にAltova 製品がインストールされたラップトップコンピューターで作業する)場合などが挙げられます。ライセンスはチェックアウトされていますが、LicenseServer は、ライセンスが使用中と表示し、ライセンスは他のマシンで使用することができません。ライセンスはチェックアウトの期間が終わると自動的にチェックインされた状態を戻します。または、チェックアウトされたライセンスはソフトウェアのライセンスの認証ダイアログのボタンを使用して「**チェックイン**」することができます。

ライセンスをチェックアウトするには以下をおこないます: (i) ソフトウェアのライセンスの認証ダイアログで「**ライセンスのチェックアウト**」をクリックします(*上のスクリーンショット参照*)。(ii) ライセンスのチェックアウトダイアログ内から、チェックアウトの期間を選択し、「**チェックアウト**」をクリックします。ライセンスがチェックアウトされます。ライセンスのチェックアウト後2つの状態が発生します: (i) ソフトウェアのライセンス認証ダイアログは時刻およびチェックアウトの期限を含むチェックアウトに関する情報を表示します。(ii) ダイアログ内の「**ライセンスのチェックアウト**」ボタンは「**チェックイン**」ボタンに変更されます。「**チェックイン**」ボタンをクリックして、ライセンスをチェックインすることができます。チェックアウト期間の期限が切れると、ライセンスは自動的にチェックイン状態に戻されるため、選択したチェックアウトの期間がオフラインで作業する期間をカバーするように確認してください。

メモ　ライセンスのチェックアウトを可能にするには、LicenseServer 上でチェックアウト機能が有効化されている必要があります。チェックアウトを試みる際この機能が有効化されていない場合、エラーメッセージが表示されます。この場合、LicenseServer 管理者に連絡してください。

サポートコードのコピー
「**サポートコードのコピー**」をクリックして、ライセンスの詳細をクリップボードにコピーしてください。これはオンラインサポートフォームを使用してサポートをリクエストする際に必要なデータです。

Altova LicenseServer を使用することにより、IT 管理者は、リアルタイムでネットワーク上の全てのライセンスの概要、および、クライアントの割り当てと、クライアントのライセンスの使用状況を確認することができます。LicenseServer を使用する利点は、ですから、多数の Altova ライセンスを管理することのできる管理機能です。Altova LicenseServer は、Altova Web サイトで無料で提供されています。Altova LicenseServer および Altova LicenseServer を使用したライセンスの供与に関する詳細は、Altova LicenseServer ドキュメントを参照してください。

▼ 注文フォーム

- *説明*

  ソフトウェア製品のライセンス許与バージョンを注文する準備が整っている場合、(*前のセクション参照*) ソフトウェアライセンス認証ダイアログ内の「**永久ライセンスの購入**」ボタン、または「**注文フォーム**」コマンドを使用して Altova オンラインショップに移動して注文することができます。

▼ 登録

- *説明*

  Altova 製品登録ページをブラウザーのタブに表示します。Altova ソフトウェアを登録することにより、最新の製品の情報が得られます。

▼ 更新のチェック

- *説明*

  Altova サーバーに接続して、より新しいバージョンの製品が利用可能かどうかチェックし、その結果を表示します。

▼ サポートセンター

- *説明*

  インターネット上にある Altova サポートセンターへのリンクとなっています。サポートセンターには FAQ やディスカッションフォーラムが含まれており、問題の解決方法を探り、Altova の技術サポートスタッフへアクセスすることができます(現在英語のみの提供となります)。

▼ WEB 上の FAQ

- *説明*

  インターネット上にある Altova の FAQ へのリンクとなっています。FAQ データベースは Altova のサポートスタッフにより常時更新されています。

▼ コンポーネントのダウンロード

- *説明*

  インターネット上にある Altova のコンポーネントダウンロードセンターへのリンクとなっています。このリンク先から様々なコンポーネン

トソフトウェアをダウンロードして、Altova 製品とともに使用することができます。ソフトウェアコンポーネントはXSLT やXSL-FO プロセッサーからアプリケーションサービスプラットフォームまで、幅広く提供されています。コンポーネントダウンロードセンターにてご利用になれるソフトウェアは、通常無料でご利用になれます。

インターネット上のMapForce

説明

インターネット上にあるAltova W eb サイトへのリンクとなっています。Altova W eb サイトでは、MapForce や関連するテクノロジーについて確認することができます。

MapForce トレーニング

説明

Altova W eb サイトでオンライントレーニングページを見つけることができますAltova の専門家トレーナーによるオンラインコースを選択することができます。

MapForce について

説明

スプラッシュ画面と製品のバージョン番号が表示されます。MapForce の64 ビットバージョンを使用している場合、これはアプリケーション名の後のサフィックス(x64) により示されています。32 ビットバージョンにはサフィックスは存在しません。

16 Code Generator

Code Generator is a MapForce built-in feature which enables you to generate Java, C++ or C# code from mapping files designed with MapForce. You can generate code not only from simple mappings with a single data source and target, but also from mappings with multiple sources and multiple targets. The result is a fully-featured and complete application which performs the mapping operation for you. Once you generate the code, you can execute the mapping by running the application directly as generated. You can also import the generated code into your own application, or extend it with your own functionality.

The primary goal of the generated code is to execute a MapForce mapping. If your mapping uses XML schemas or DTDs, you can optionally generate schema wrapper libraries for XML/DTD schemas used by the mapping, which enables you to read or write data to or from XML instances. The schema wrapper libraries enable you to work with XML data programmatically, using types generated from the schema, without too much concern for the underlying XML API. For details, see Generating Code from XML Schemas or DTDs.

The generated code is expressed in C++, Java or C# programming languages.

Target Language	C++	C#	Java
Development environments	Microsoft Visual Studio 2010, 2013, 2015, 2017, 2019	Microsoft Visual Studio 2010, 2013, 2015, 2017, 2019 Target frameworks: • .NET Framework • .NET Core 3.1 • .NET 5.0	(OpenJDK を含む) Java SE JDK 7、8、または11 Eclipse 4.4 or later Apache Ant (build.xml file)
XML DOM implementations	MSXML 6.0 Apache Xerces 3	System.Xml	JAXP
Database API	ADO	ADO.NET	JDBC

Note: The MapForce-generated code can be considered thread-safe only if the underlying third-party XML DOM and database API libraries are. Although the thread safety of the generated code cannot be realistically proven or guaranteed, it is likely that multiple concurrent instances of the mapping code will run successfully in most cases.

C++

You can configure whether the C++ generated output should use MSXML 6.0 or Apache Xerces 3. MapForce generates complete project (.vcproj) and solution (.sln) files for all supported versions of Visual Studio (see table above). The generated code optionally supports MFC.

Prerequisites:

1. To compile the generated C++ code, Windows SDK must be installed on your computer.
2. To use Xerces 3 for C++, you will need to install and build it by following instructions from http://xerces.apache.org/xerces-c/. Make sure to add the XERCES3 environment variable that points to the directory where Xerces was installed, for example: `C:\xerces-c-3.2.2`. Also, the PATH

(C) 2015-2021 Altova GmbH

environment variable must include the path where the Xerces binaries are, for example: `%XERCES3%\bin`.

3. When building C++ code for Visual Studio and using a Xerces library precompiled for Visual C++, a compiler setting has to be changed in all projects of the solution:

   a) Select all projects in the Solution Explorer.
   b) On the **Project** menu, click **Properties**.
   c) Click **Configuration Properties | C/C++ | Language**.
   d) In the list of configurations, select *All Configurations*.
   e) Change **Treat wchar_t as Built-in Type** to **No (/Zc:wchar_t-)**.

C#

The generated C# code can be used from any .NET capable programming language, such as VB.NET, Managed C++, or J#. Project files can be generated for all supported versions of Visual Studio (see table above).

Java

The generated Java output is written against the Java API for XML Processing (JAXP) and includes an Ant build file and project files for supported versions of Java and Eclipse (see table above).

Generated output

The designated destination folder will include all the libraries and files required to execute the mapping, namely:

- A variable number of Altova libraries required by the mapping (for example, Altova function libraries, database libraries, EDI libraries)
- A complete mapping application. When compiled and run, the application performs the mapping transformation.

Code generator templates

The generated code supports customization via a template language called SPL (Spy Programming Language). This can be useful, for example, when you want to customize code according to your company's writing conventions, or replace specific libraries in the generated code.

16.1 What's new ...

Version 2021 R2

- When generating C# code, you can select .NET Core 3.1 and .NET 5.0 as target frameworks from code generation options (this adds to existing support for .NET Framework projects).

Version 2020 R2

- Code generated for XML schema wrapper libraries now provides more control over element namespaces and prefixes. New methods are available to declare or override namespaces for an element, or to append an element with a prefixed namespace. See Example: Purchase Order.

Version 2020

- Added support for Visual Studio 2019
- End of support for Visual Studio 2008

Version 2018

- Added support for Microsoft Visual Studio 2013, 2015, 2017
- End of support for Visual Studio 2005 and Xerces 2.x

Version 2014

- Removal of compatibility mode option for code generation

Version 2011

- Contains bug fixes and enhancements

Version 2010 R3

- Support for Microsoft Visual Studio 2010
- Support for MSXML 6.0 in generated C++ code
- Support for 64-bit targets for C++ and C# projects

Version 2010

- Enumeration facets from XML schemas are now available as symbolic constants in the generated classes (using 2007r3 templates)

Version 2009 sp1

- Apache Xerces version 3.x support added (older versions starting from Xerces 2.6.x are still supported)

Version 2009

- The generated mapping implementation was redesigned to support sequences and grouping. The API has not changed.

(C) 2015-2021 Altova GmbH

Version 2008 R2

- Support for generation of Visual Studio 2008 project files for C# and C++ has been added
- Generated MapForce mapping code in C# and Java can use readers/writers, streams, strings or DOM documents as sources and targets

Version 2008

- The new 2007 R3-style SPL templates have been further enhanced:
  - It is now possible to remove single elements
  - Access to schema metadata (e.g. element names, facets, enumerations, occurrence, etc.) is provided
  - Complex types derived by extension are now generated as derived classes

Version 2007 R3

Code Generator has been redesigned for version 2007 release 3 to simplify usage of the generated code, reduce code volume and increase performance.

- Handling of XML documents and nodes with explicit ownership, to avoid memory leaks and to enable multi-threading
- New syntax to avoid name collisions
- New data types for simpler usage and higher performance (native types where possible, new null handling, ...)
- Attributes are no longer generated as collections
- Simple element content is now also treated like a special attribute, for consistency
- New internal object model (important for customized SPL templates)
- Compatibility mode to generate code in the style of older releases
- Type wrapper classes are now only generated on demand for smaller code

16.2 Generating C++ code

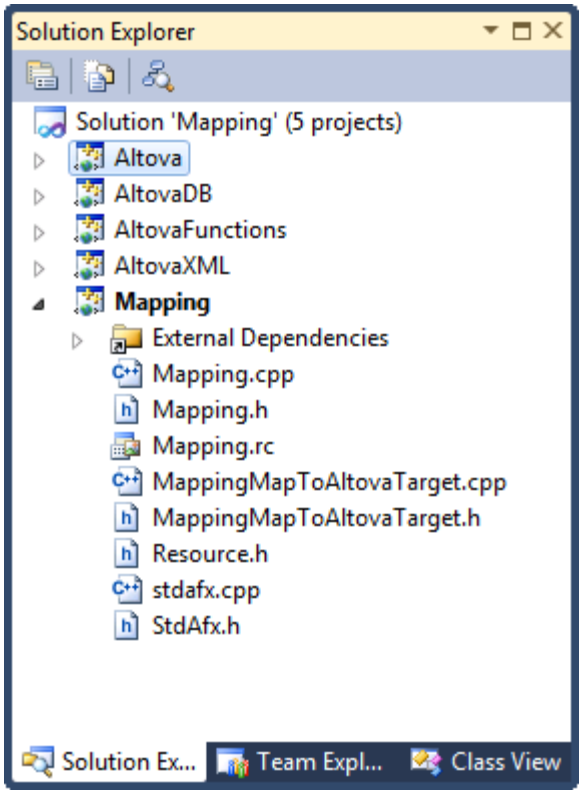
You can generate C++ code for Visual Studio 2010, 2013, 2015, 2017, 2019. The generated code includes **.sln** and .**vcproj** files for Visual Studio. Note the following when generating code:

- Certain MapForce features are not supported in generated program code. For details, see Supported features in generated code.
- You can generate code either from a single mapping design (.mfd), or from a mapping project (.mfp). If you generate code from a single mapping, the resulting application executes the respective mapping transformation. If you generate code from a MapForce project (.mfp) which includes multiple mappings, the resulting application executes in bulk all mappings included in the project.
- You can change the general code generation options from the **Tools | Options** menu, **Generation** tab. See Code Generator Options.
- You can change the name of the generated mapping application and other settings from the **File | Mapping settings** menu. The default application name is **Mapping**. See also Changing the Mapping Settings.
- For information about handling paths in generated code, see Paths in Various Execution Environments.
- If your mapping contains database components, you can view database specific settings by clicking a database component, and then selecting the menu option **Component | Properties**. See Database Component Settings.
- Support for database connections varies by platform and there are connection kinds that are not supported on all platforms. If your mapping connects to a database, choose a database connection that is compatible with the target environment for which you generate code. For details, see Database mappings in various execution environments.

A typical C++ solution generated by MapForce includes the following:

- Several Altova-signed libraries required by the mapping (all prefixed with **Altova**).
- The main mapping project (in this example, **Mapping**), which includes the mapping application and dependent files.

(C) 2015-2021 Altova GmbH



Sample C++ solution generated with MapForce

This section includes the following topics:

- Generating code from a mapping
- Generating code from a mapping project
- Building the project
- Running the application

16.2.1 Generating code from a mapping

To generate C++ code from a mapping design file (.mfd):

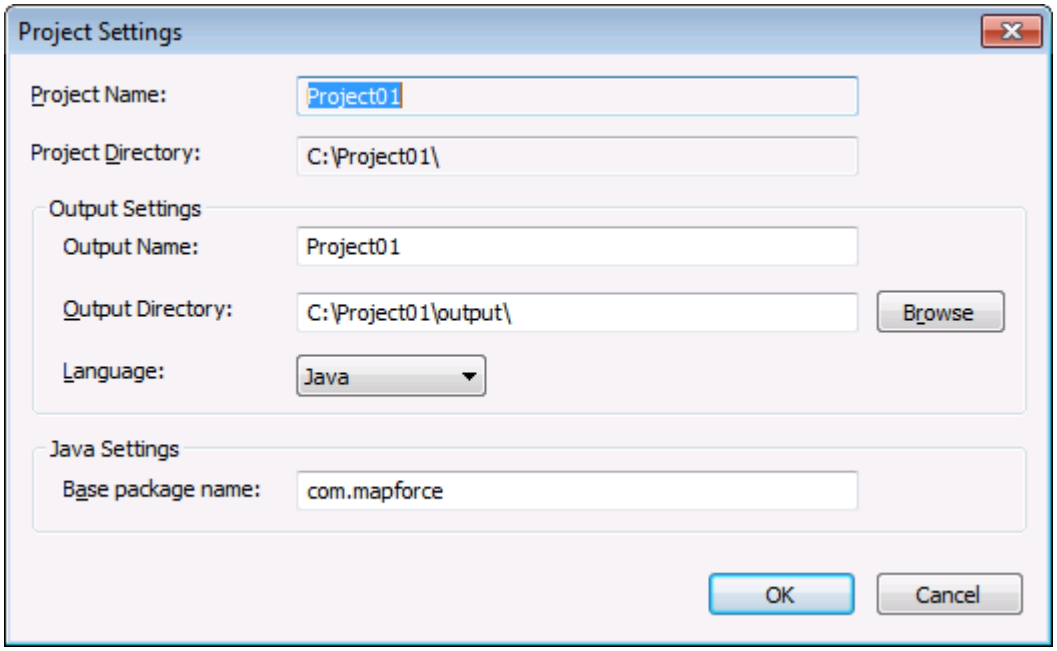
1. Review and select the code generator options.
2. On the **File** menu, click **Generate code in | C++**.
3. Select a destination directory for the generated files, and then click OK to confirm. The result of code generation (error or success message) is displayed in the Messages window.

The default name of the generated application is **Mapping**. If required, you can change this, and other settings, from the Mapping Settings dialog box.

16.2.2 Generating code from a mapping project

To generate code from a mapping project (.mfp):

1. If you haven't done so already, open the mapping project in MapForce.
2. Right-click the project in the Project window, and then click **Properties**.



3. Review and change the project settings if required (in particular, ensure that the target language and the output directory are set correctly), and then click **OK**.
4. On the **Project menu, click Generate code for the Entire Project**.

The progress and result of the code generation process (error or success message) is displayed in the Messages window.

By default, the name of the generated application is the same as the project name. If the project name contains spaces, these are converted to underscores in the generated code. By default, code is generated in the same directory as the MapForce project, in the **output** sub-directory.

To change the output directory and the name of the project, click the Project in the Project window, and then select **Project | Properties** from the menu. If your MapForce project contains folders, you can change the code generation settings for each individual folder (right click on the folder, and then select **Properties**). Otherwise, all project folders inherit the settings from the MapForce project.

16.2.3 Building the project

Once you generated the C++ code, building it in Visual Studio is the next step. To build the generated code:

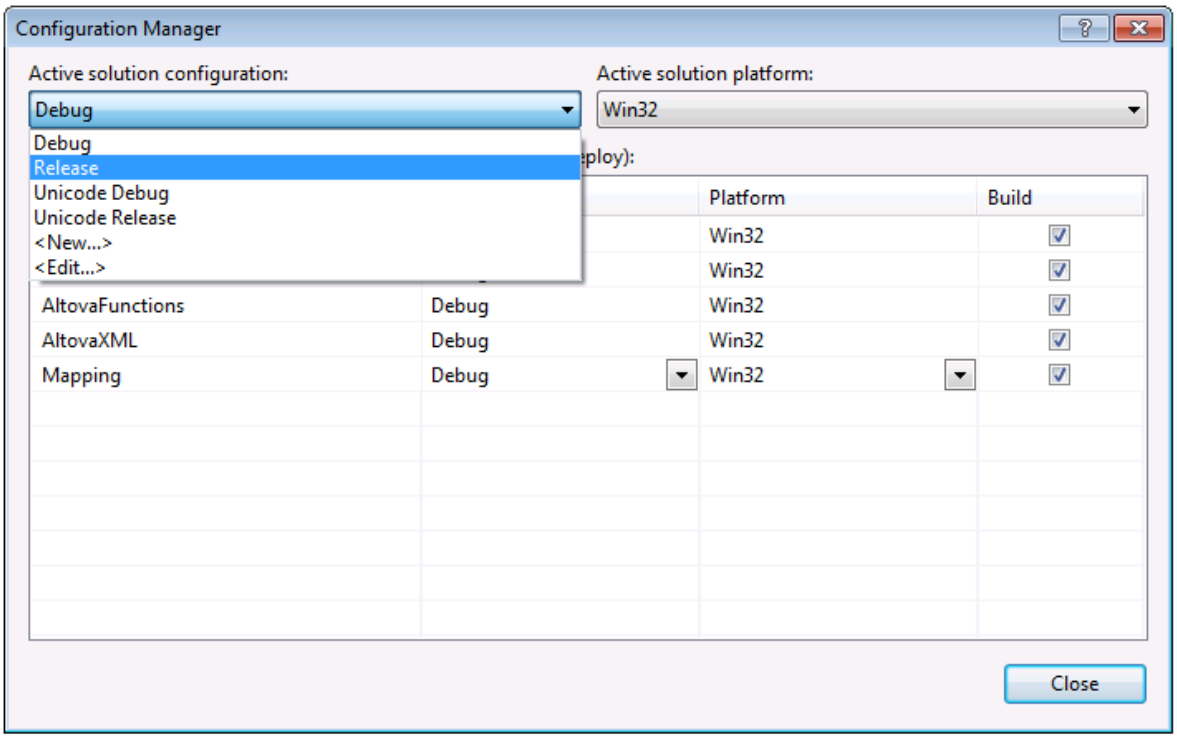
1. Open the generated solution (.sln) file in Visual Studio.

By default, the name of the solution file is **Mapping.sln**, and it is located in the **Mapping** subdirectory relative to the directory where you saved the generated code. If you changed the application name from the mapping

(C) 2015-2021 Altova GmbH

settings, then the name of the .sln file is changed accordingly. For example, if you changed the application name to **MyApplication**, then the solution file is called **MyApplication.sln**, and it is located in the **MyApplication** subdirectory.

2. On the **Build** menu, click **Configuration Manager**.



3. Select the required build configuration (Debug, Release, Unicode Debug, Unicode Release). Note that only Unicode builds support the full Unicode character set in XML and other files. The non-Unicode builds work with the local codepage of your Windows installation.
4. On the **Build** menu, click **Build Solution**.

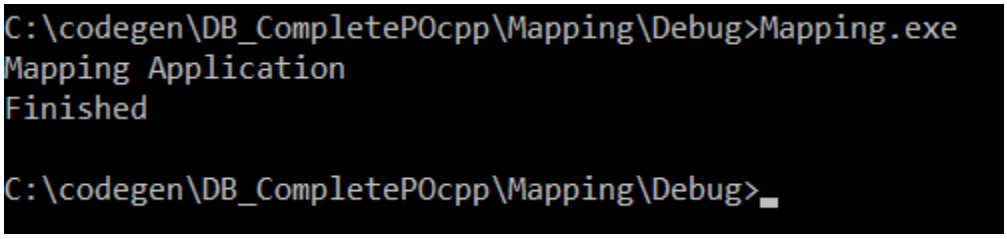
16.2.4 Running the application

Once you compile the Visual Studio project, a command-line application is produced, called **Mapping.exe**. (Note that if you changed the application name from the mapping settings, then the executable name is changed accordingly.)

You can locate the mapping application in one of the following subdirectories relative to the .sln file, depending on the build option you chose:

- Debug
- Release
- Unicode Debug
- Unicode Release

To run the application, open a command prompt, change the current directory to the path of the executable, and run it, for example:



(C) 2015-2021 Altova GmbH

16.3 Generating C# code

You can generate .NET C# code for any mapping or mapping project where all the mapping components are supported in C#. For an overview of components supported in each language, including C#, see Supported features in generated code. Note that MapForce typically issues notification or warning messages whenever you attempt to add components that are not supported in a specific language.

When generating C# code, you can target not only the .NET Framework platform but also the .NET Core 3.1 and .NET 5.0 platforms. If you choose the latter, the result is a .NET (Core) cross-platform desktop application that can run on Windows, macOS, and Linux.

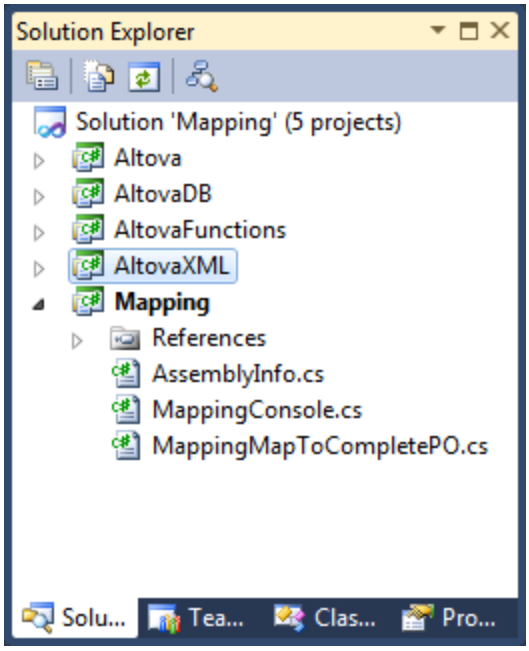
Note the following when generating code:

- Certain MapForce features are not supported in generated program code. For details, see Supported features in generated code.
- You can generate code either from a single mapping design (.mfd), or from a mapping project (.mfp). If you generate code from a single mapping, the resulting application executes the respective mapping transformation. If you generate code from a MapForce project (.mfp) which includes multiple mappings, the resulting application executes in bulk all mappings included in the project.
- You can change the general code generation options from the **Tools | Options** menu, **Generation** tab. See Code Generator Options.
- You can change the name of the generated mapping application and other settings from the **File | Mapping settings** menu. The default application name is **Mapping**. See also Changing the Mapping Settings.
- For information about handling paths in generated code, see Paths in Various Execution Environments.
- If your mapping contains database components, you can view database specific settings by clicking a database component, and then selecting the menu option **Component | Properties**. See Database Component Settings.
- Support for database connections varies by platform and there are connection kinds that are not supported on all platforms. If your mapping connects to a database, choose a database connection that is compatible with the target environment for which you generate code. For details, see Database mappings in various execution environments.

If your mapping uses database components, make sure to create the database connection using database drivers compatible with the platform on which you intend to run the generated program. For example, ADO database connections and some ODBC drivers are not supported on Linux.

A typical C# solution generated by MapForce includes the following:

- Solution (.sln) and project (.csproj) files that can be opened in Visual Studio.
- Several Altova-signed libraries required by the mapping (all prefixed with **Altova**).
- The main mapping project (in this example, **Mapping**), which includes the mapping application and dependent files.

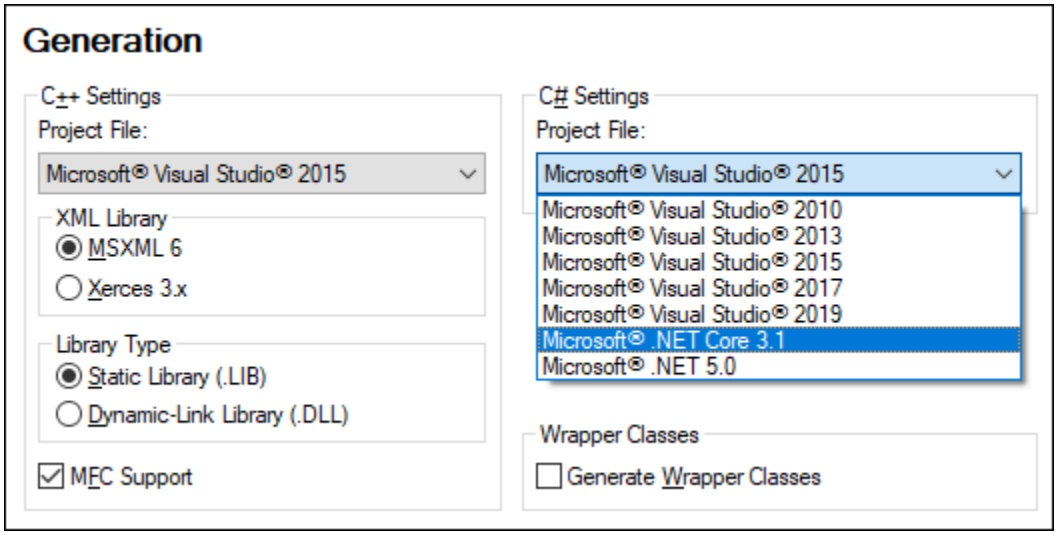


Sample C# solution generated with MapForce

16.3.1 Selecting the target platform

Before generating C# code, you can set the target platform at program level, as follows:

1. On the **Tools** menu, click **Options**.
2. Click the **Generation** tab.
3. In the "C# settings" group, select a value from the drop-down list, as applicable.



Select the option **Microsoft .NET Core 3.1** or **Microsoft .NET 5.0** to generate a Visual Studio solution targeting the respective platforms. If you need to target the **.NET Framework** platform for a specific Visual

(C) 2015-2021 Altova GmbH

Studio version, select any of the **Microsoft Visual Studio 2010-2019** options—in this case, the generated solution will target the .NET Framework version corresponding to the respective Visual Studio version.

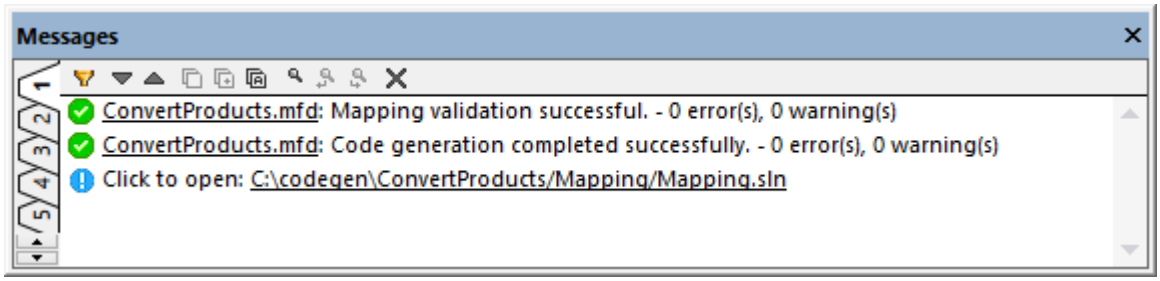
The **Generate Wrapper Classes** check box, if selected, creates additional projects in the generated code, one for each XML schema or DTD component present on the mapping. The generated wrapper projects enable you to manipulate programmatically the respective XML instances outside of the mapping. The wrapper libraries are optional and independent of the mapping program, although you can use them from the mapping program if necessary. For more information about wrapper classes, see Generating Code from XML Schemas or DTDs.

16.3.2 Generating code from a mapping

To generate C# code from a mapping design file (.mfd):

1. If you haven't done so already, create a new mapping or open an existing one in MapForce.
2. Select C# as mapping transformation language. To do this, either click the C# toolbar button or select the menu command **Output | C# (Sharp)**.
3. Set or change the C# code generation options as described in Selecting the target platform.
4. Select the menu command **File | Generate code in | C# (Sharp)**. You are now prompted to select a target directory.
5. Browse for the directory of your choice (for example, "C:\codegen\ConvertProducts") and click **OK**.

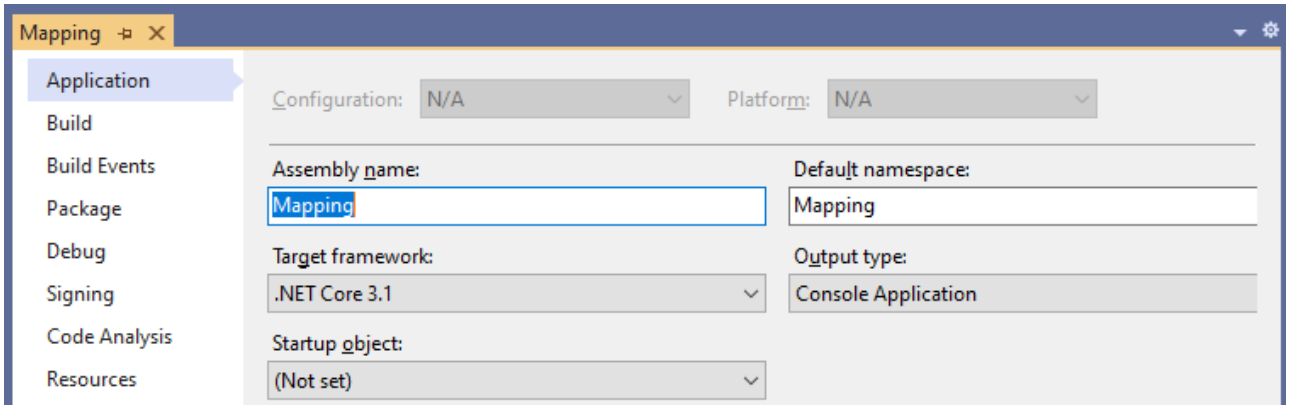
The code generation outcome is displayed in the Messages window, for example:



Optionally, click the link in the Messages window to open the generated solution in Visual Studio.

Note that, if you chose to generate code targeting .NET 5 or .NET Core platforms, you need the respective SDKs and a compatible Visual Studio version. For the download package applicable to your operating system and platform, refer to the Microsoft website (https://dotnet.microsoft.com/download).

To double-check the project's target framework from Visual Studio, right-click the "Mapping" project in the Solution Explorer, select **Properties**, and observe the value displayed in the **Target Framework** drop-down list, for example:



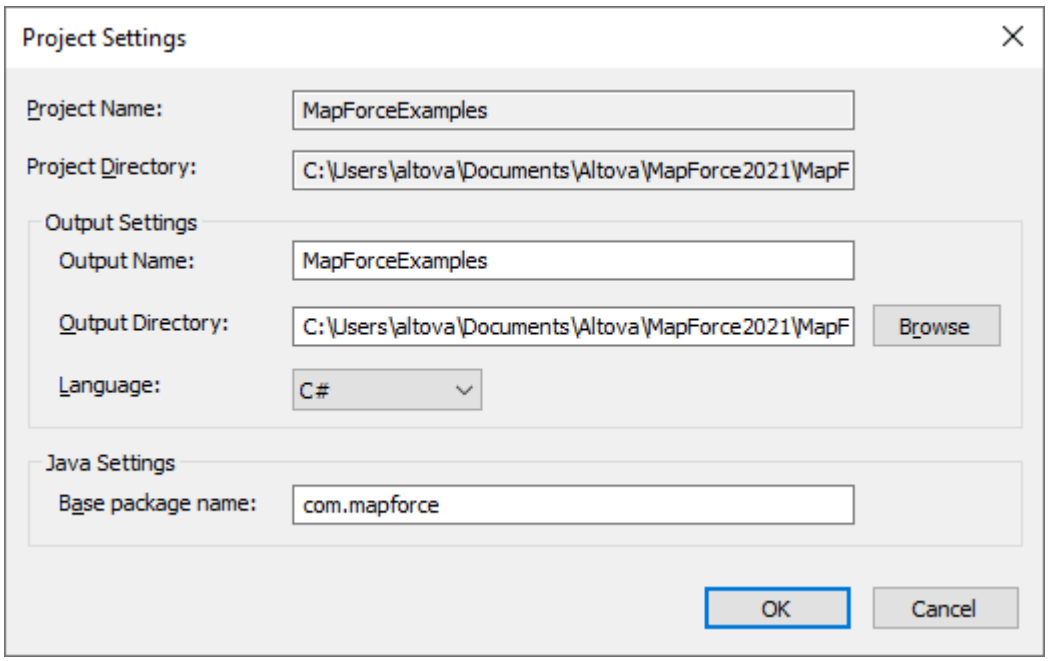
The default name of the generated application is **Mapping**. Optionally, you can change this and other settings before generating code, from the Mapping Settings dialog box.

16.3.3 Generating code from a mapping project

You can generate C# code from a mapping project (.mfp) that consists of multiple mapping design (.mfd) files. Note that all mapping design files in the project must qualify for C# generation, that is, all their components must be supported in C#, as described in Supported features in generated code.

To generate code from a mapping project (.mfp):

1. If you haven't done so already, open the mapping project in MapForce.
2. Right-click the project in the Project window, and then click **Properties**.



(C) 2015-2021 Altova GmbH

3. Review and change the project settings if required (in particular, ensure that the target language and the output directory are set correctly), and then click **OK**.
4. On the **Project** menu, click **Generate code for the Entire Project**.

The progress and result of the code generation process (error or success message) is displayed in the Messages window.

By default, the name of the generated application is the same as the project name. If the project name contains spaces, these are converted to underscores in the generated code. By default, code is generated in the same directory as the MapForce project, in the **output** sub-directory.

To change the output directory and the name of the project, click the Project in the Project window, and then select **Project | Properties** from the menu. If your MapForce project contains folders, you can change the code generation settings for each individual folder (right click on the folder, and then select **Properties**). Otherwise, all project folders inherit the settings from the MapForce project.

16.3.4 Building .NET Framework projects

If your generated C# targets the .NET Framework platform, the easiest way to build the project is as follows:

1. Open the generated solution (.sln) file in Visual Studio.

By default, the name of the solution file is **Mapping.sln**, and it is located in the **Mapping** subdirectory relative to the directory where you saved the generated code. If you changed the application name from the mapping settings, then the name of the .sln file is changed accordingly. For example, if you changed the application name to **MyApplication**, then the solution file is called **MyApplication.sln**, and it is located in the **MyApplication** subdirectory.

2. On the **Build** menu, click **Configuration Manager**.
3. Select the required build configuration (Debug, Release).
4. On the **Build** menu, click **Build Solution**.

Running the application

Once you compile the Visual Studio project, a command-line application is produced, called **Mapping.exe**. Note that if you changed the application name from the Mapping Settings dialog box, then the executable name is changed accordingly.

You can find the mapping application in one of the following subdirectories relative to the .sln file, depending on the build option you chose earlier:

- bin\Debug
- bin\Release

To run the application, open a command prompt, change the current directory to the path of the executable, and run it, for example:

```
C:\codegen\DB_CompletePOcs\Mapping\bin\Release>Mapping.exe
Mapping Application
Connecting to CustomersAndArticles database...

Finished

C:\codegen\DB_CompletePOcs\Mapping\bin\Release>
```

16.3.5 Building .NET 5 and .NET Core projects

To build and run the generated .NET Core 3.1 and .NET 5.0 solution, you must have the respective SDK installed. For the download package applicable to your operating system and platform, refer to the Microsoft website (https://dotnet.microsoft.com/download). After downloading and installing the required package, yon can easily view the installed SDKs by entering the following command at the command prompt:

```
dotnet --list-sdks
```

If you have the .NET Core 3.1 or .NET 5.0 SDK, you can proceed to building the generated code, either at the command line or in Visual Studio. If using Visual Studio, note that, at the time of this writing (December 2020), the .NET Core 3.1 and .NET 5.0 SDKs are compatible with Visual Studio 2019 (v16.7).

Building at the command line

To build the generated code at the command line:

1. Open a command prompt window and change directory to the one where the mapping code was generated. For example, if you selected **C:\codegen\ConvertProducts** as target directory and the default mapping settings, run:

   ```
   cd C:\codegen\ConvertProducts
   ```

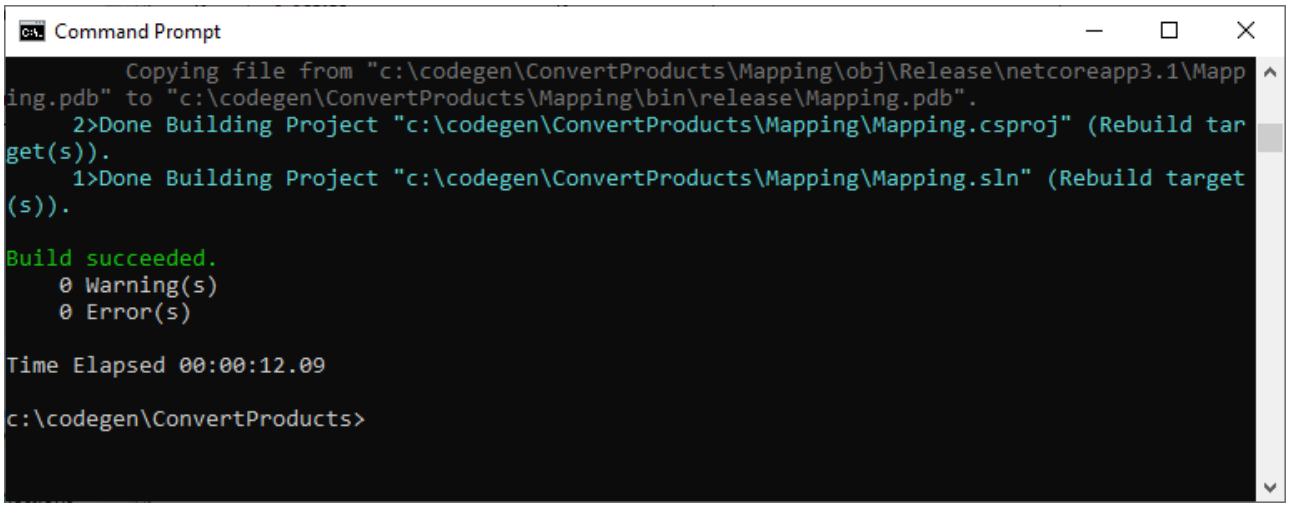
2. Run the following command:

   ```
   dotnet build Mapping\Mapping.sln --no-incremental --configuration Release --
   verbosity normal --output Mapping\bin\release
   ```

Note: The path to the mapping solution and its name may be different if you have changed the **Application Name** property from the Mapping Settings dialog box.

The outcome for a successful build could be similar to the one below:

(C) 2015-2021 Altova GmbH



Building with Visual Studio

To build the generated code in Visual Studio:

1. Double-click **Mapping.sln** (or the respective custom solution name) to open it in Visual Studio.
2. On the **Build** menu, click **Configuration Manager**.
3. Select the required build configuration (Debug, Release) and solution platform (x64, x86, Any CPU).
4. On the **Build** menu, select **Build Solution (Ctrl+Shift+B)**.

Running the generated program

After you build the generated code, a **Mapping.exe** (or a program with a custom name if configured) is produced in the project's output directory. The path of the output directory depends on the build configuration you have chosen earlier (Debug, Release).

If you build the generated code on Linux, the generated executable will be called simply **Mapping**, without any extension.

On Windows, you can execute the program by double-clicking the executable in Windows Explorer or by calling the executable from a command prompt window. On other platforms, you will need to call the executable from the terminal with a command like:

```
./Mapping
```

Note that, if the mapping has references to any input or output files on the disk, the program will execute successfully only if all the paths can be resolved. For example, if you configured the mapping so that a component should look for some input file in the same directory using a relative path, then the respective file must be present in the program's execution directory. For information about handling paths at mapping design time, see Using Relative and Absolute Paths.

16.4 Generating Java code

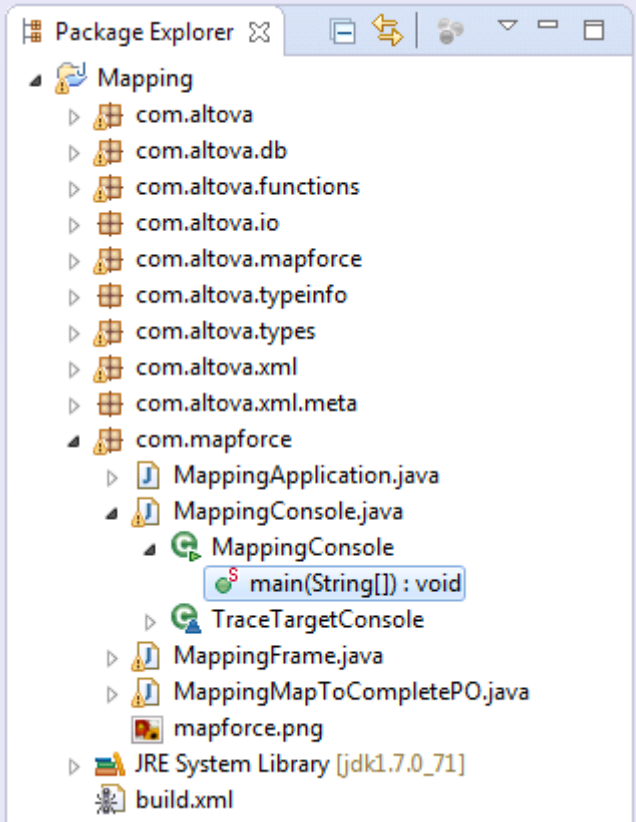
You can generate program code for (OpenJDK を含む) Java SE JDK 7、8、または11. Note the following when generating code:

- Certain MapForce features are not supported in generated program code. For details, see Supported features in generated code.
- You can generate code either from a single mapping design (.mfd), or from a mapping project (.mfp). If you generate code from a single mapping, the resulting application executes the respective mapping transformation. If you generate code from a MapForce project (.mfp) which includes multiple mappings, the resulting application executes in bulk all mappings included in the project.
- You can change the general code generation options from the **Tools | Options** menu, **Generation** tab. See Code Generator Options.
- You can change the name of the generated mapping application and other settings from the **File | Mapping settings** menu. The default application name is **Mapping**. See also Changing the Mapping Settings.
- For information about handling paths in generated code, see Paths in Various Execution Environments.
- If your mapping contains database components, you can view database specific settings by clicking a database component, and then selecting the menu option **Component | Properties**. See Database Component Settings.
- Support for database connections varies by platform and there are connection kinds that are not supported on all platforms. If your mapping connects to a database, choose a database connection that is compatible with the target environment for which you generate code. For details, see Database mappings in various execution environments.

A typical Java project generated by MapForce includes the following:

- Several Altova-signed Java packages required by the mapping (all prefixed with **com.altova**).
- The **com.mapforce** package, which includes the mapping application and dependent files (as shown below, it is possible to change the name of this package). The two most important files in this package are as follows:
  - The Java mapping application as a dialog application (**MappingApplication.java**).
  - The Java mapping application as a console application (**MappingConsole.java**).
- A **build.xml** file which you can execute with Apache Ant to compile the project and generate JAR files.

(C) 2015-2021 Altova GmbH



Sample MapForce-generated Java application (Eclipse IDE)

This section includes the following topics:

- Generating code from a mapping
- Generating code from a mapping project
- Handling JDBC references
- Building the project with Ant
- Example: Generate and Run Java Code

16.4.1 Generating code from a mapping

To generate Java code from a mapping design file (.mfd):

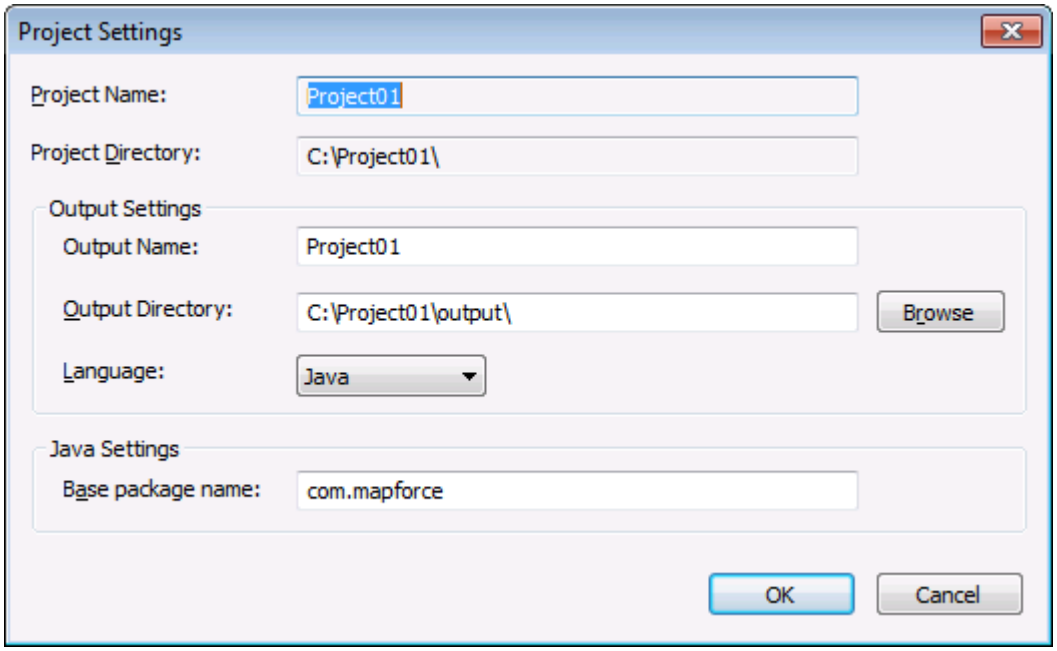
1. Review and select the code generator options.
2. On the **File** menu, click **Generate code in | Java**.
3. Select a destination directory for the generated files, and then click **OK** to confirm. The result of code generation (error or success message) is displayed in the Messages window.

The default name of the generated application is **Mapping**, and the default name of the base package is **com.mapforce**. If required, you can change these from the Mapping Settings dialog box.

16.4.2 Generating code from a mapping project

To generate code from a mapping project (.mfp):

1. If you haven't done so already, open the mapping project in MapForce.
2. Right-click the project in the Project window, and then click **Properties**.



3. Review and change the project settings if required (in particular, ensure that the target language and the output directory are set correctly), and then click **OK**.
4. On the **Project menu, click Generate code for the Entire Project**.

The progress and result of the code generation process (error or success message) is displayed in the Messages window.

By default, the name of the generated application is the same as the project name. If the project name contains spaces, these are converted to underscores in the generated code. By default, code is generated in the same directory as the MapForce project, in the **output** sub-directory.

To change the output directory and the name of the project, click the Project in the Project window, and then select **Project | Properties** from the menu. If your MapForce project contains folders, you can change the code generation settings for each individual folder (right click on the folder, and then select **Properties**). Otherwise, all project folders inherit the settings from the MapForce project.

16.4.3 Building the project with Ant

Apache Ant is an open source tool which automates building and compilation of Java projects (see http://ant.apache.org/manual/). Ant works with build files (such files define the sources and targets from which code must be compiled, as well as any specific build options). Since any MapForce-generated project includes a **build.xml** file recognized by Ant, you can easily build MapForce-generated projects with Ant.

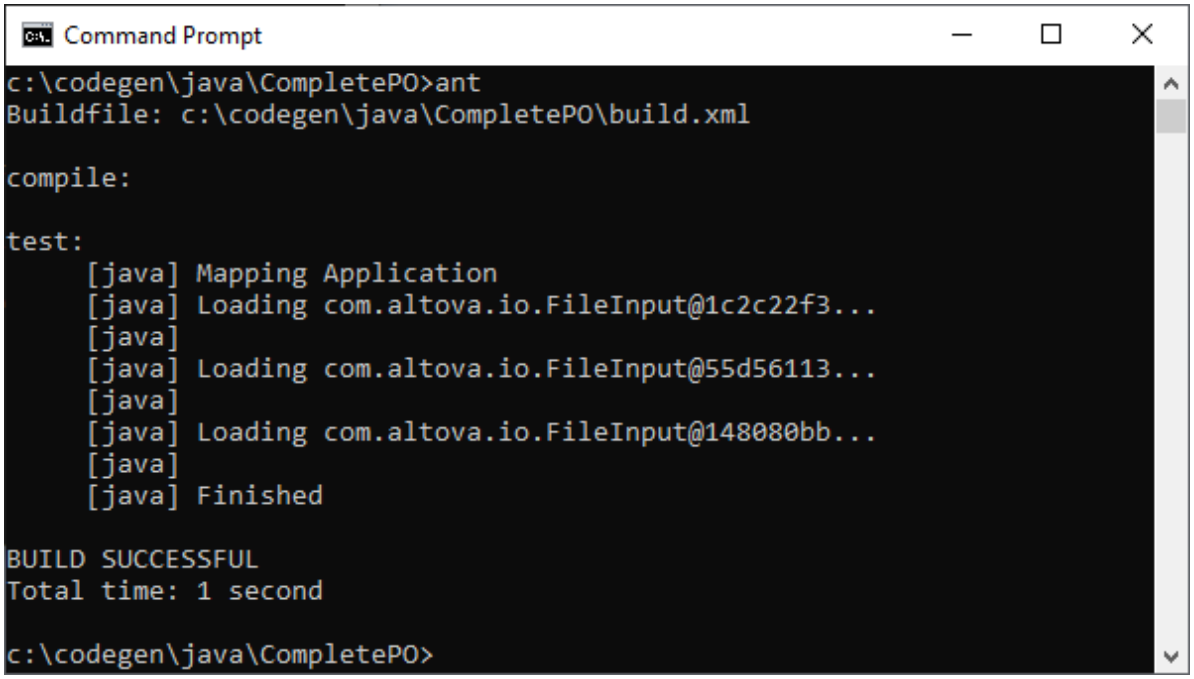
(C) 2015-2021 Altova GmbH

Ant may be available on your system either as a standalone installation, or bundled within Eclipse (or other Java IDEs). For instructions on how to install Ant on your system, see http://ant.apache.org/manual/. For instructions on how to use Ant in Eclipse, refer to the Eclipse documentation.

You can quickly check whether the standalone version of Ant (not the one bundled with Eclipse) is available on your system by opening a command prompt and typing `ant` at the command line. When Ant is already available, the resulting message will be similar to: `Buildfile: buildxml does not exist!` This message indicates that Ant is installed and it is attempting to build a **build.xml** file, but the latter does not exist in the current directory. If you run Ant from a directory which includes a **build.xml** file, Ant executes the **build.xml** file instead, with whatever build options are defined in it.

To build a MapForce-generated Java project with Ant:

1. Open a command prompt and navigate to the directory where the Java project was generated (note that the directory must contain the **build.xml** file).
2. At the command prompt, enter `ant`. This will compile and execute the Java code according to the default options defined in the **build.xml** file, for example:



To generate a JAR file with Ant:

- At the command prompt, enter `ant jar`.

For help with Ant command syntax and options, enter `ant -help` at the command line. For a demo, see Example: Generate and Run Java Code.

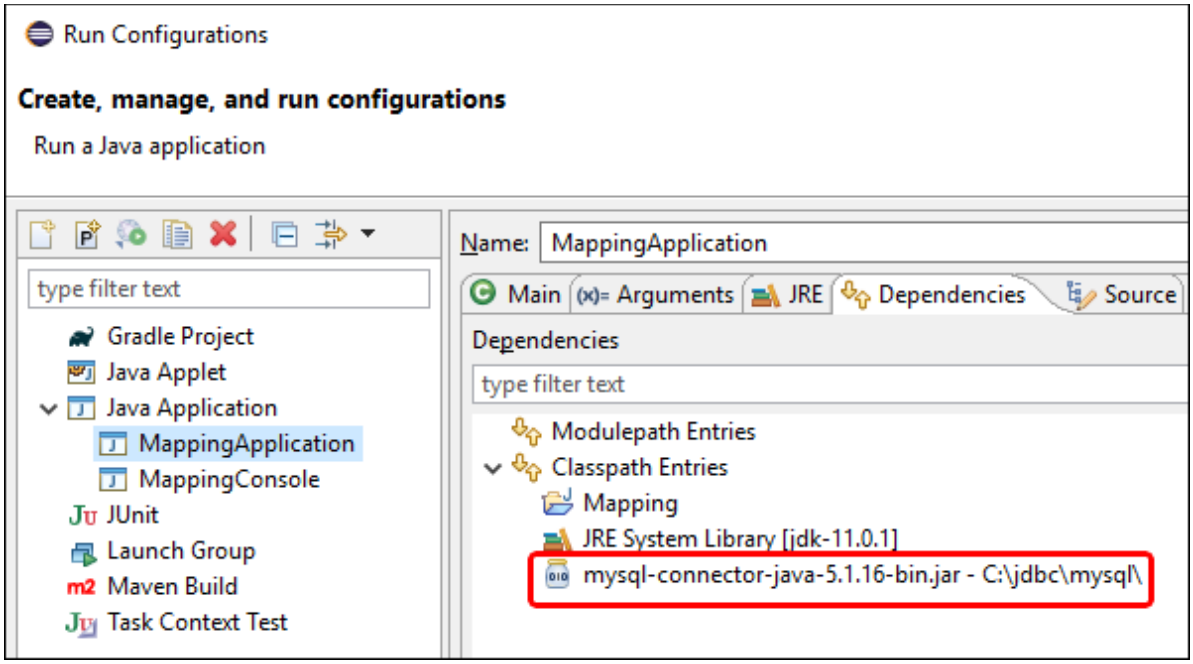
16.4.4 Handling JDBC references

If the mapping connects to a database through JDBC, ensure that the JDBC drivers used by the mapping are installed on your system. To view the current JDBC settings of any database component in MapForce, click it, and then select **Component | Properties** from the menu. For more information, see Creating a JDBC connection.

> If the mapping uses a non-JDBC database connection, the connection may be converted to JDBC during Java code generation, to provide compatibility in a Java environment. For example, ADO, ADO.NET, and ODBC connections are converted to JDBC connections when you generate Java code from the mapping. For details, see Database mappings in various execution environments.

If you test run the generated Java application in Eclipse, the JDBC driver may need to be added as a "classpath" entry to the current run configuration. Otherwise, test running the application in Eclipse could fail with an error message such as: `java.lang.ClassNotFoundException: com.mysql.jdbc.Driver`. Taking MySQL database as an example, you can add the JDBC driver as a dependency in Eclipse as follows:

1. Generate the Java code from MapForce and import the project into Eclipse using the menu command **File | Import | Existing Projects into Workspace**.
2. On the **Run** menu, click **Run Configurations**. On the left side of the dialog box, under "Java Application", there are two configurations called **MappingApplication** and **MappingConsole**.
3. Click the desired configuration (for example, **MappingApplication**).
4. On the **Dependencies** tab, click **Classpath entries**, and then click **Add External JARs**.
5. Browse for the .jar file of the JDBC driver, for example:



6. Click **Run** to run the program with the database JDBC driver added as a dependency.

If you get the JDBC driver error above when test running the Ant **build.xml** file, add the JDBC driver to the classpath of the "test" task. The following is an example of an Ant "test" task that includes a reference to the .jar file of the JDBC driver.

(C) 2015-2021 Altova GmbH

```
<target name="test" depends="compile">
   <java classpath="C:\codegen\java\mysql_mapping"
classname="com.mapforce.MappingConsole" fork="true" failonerror="true">
      <classpath>
         <pathelement path="${classpath}"/>
         <pathelement location="C:\jdbc\mysql\mysql-connector-java-5.1.16-bin.jar"/>
      </classpath>
      <arg line="${cmdline}"/>
   </java>
</target>
```

Note: The example above assumes that the .jar file exists at the following path: **C:\jdbc\mysql\mysql-connector-java-5.1.16-bin.jar**. Make sure to change the .jar file path as applicable to your environment.

Including JDBC drivers in the application's manifest

If you build JAR files from the generated Java application, add a reference to the database driver in the "manifest" section of the **build.xml** file. This ensures that the reference to the database driver is available in the manifest (**MANIFEST.MF**) file after you build the project.

To add the database reference to the manifest file:

1. Locate the **manifest** element in the **build.xml** file,
2. Add a new element called **attribute** where the "name" attribute is "Class-Path" and the "value" attribute is the name of the .jar file. For example, for MySQL 5.1.16, the new element could look as follows:

   ```
   <attribute name="Class-Path" value="mysql-connector-java-5.1.16-bin.jar"/>
   ```

   The **manifest** element of the **build.xml** file now looks as follows:

   ```
   <manifest file="C:\codegen\java\mysql_mapping/META-INF/MANIFEST.MF"
   mode="replace">
      <attribute name="Created-By" value="MapForce 2021"/>
      <attribute name="Main-Class" value="com.mapforce.MappingConsole"/>
      <attribute name="Class-Path" value="mysql-connector-java-5.1.16-bin.jar"/>
   </manifest>
   ```

3. Copy the JAR file of the JDBC driver to the folder that contains the JAR file of the generated application.

16.4.5 Example: Generate and run Java code

This example walks you through the steps required to generate a Java application with MapForce, and compile it outside of MapForce using the Eclipse Integrated Development Environment (IDE) and Apache Ant. After completing this example, you will have created and compiled a complete Java application which executes one of the mapping samples available by default in MapForce.

If you can already compile successfully other Java applications with Eclipse and Ant, there are no special requirements to run this example. Otherwise, note the following prerequisites:

- Java Development Kit (JDK), Eclipse, and Ant must be installed on your system. Eclipse typically includes a bundled version of Ant but you can also install Ant separately, see also Building the project with Ant.
- To run Eclipse with OpenJDK, the PATH environment variable typically needs to include the path to the JDK **bin** directory (for example, **C:\Java\jdk-11.0.1\bin**).
- If you run Apache Ant standalone like in this example, the PATH environment variable should also include the path to Ant **bin** directory (for example, **C:\apache-ant-1.10.5\bin**), so that you can conveniently run Ant without having to type the full path to the executable at the command line.

This example uses the following configuration:

- MapForce 64-bit
- OpenJDK 11.0.1 64-bit
- Eclipse IDE for Java Developers, version 2019-03 (4.11.0), 64-bit
- Apache Ant 1.10.6 (installed as standalone) and Apache Ant 1.10.5 (bundled with Eclipse)

Step 1: Generate Java code

To generate the Java code from MapForce:

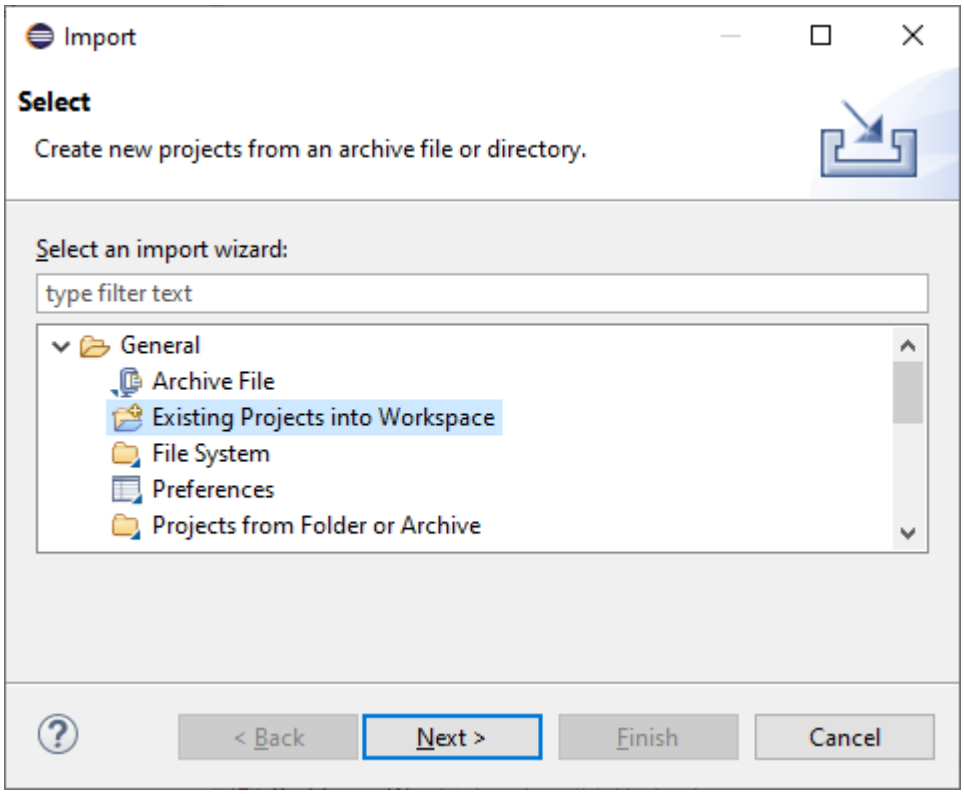
1. On the **File** menu, click **Open**, and browse for the **CompletePO.mfd** mapping available in the <マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥directory.
2. On the **Output** menu, click **Java**. This changes the transformation language to Java.
3. On the **File** menu, click **Generate code in | Java**. When prompted, browse for the directory where the Java project should be saved. For the purpose of this example, you might want to generate Java code to **C:\codegen\java\CompletePO**. You will need to refer to this path in subsequent steps.

Step 2: Import the project into Eclipse

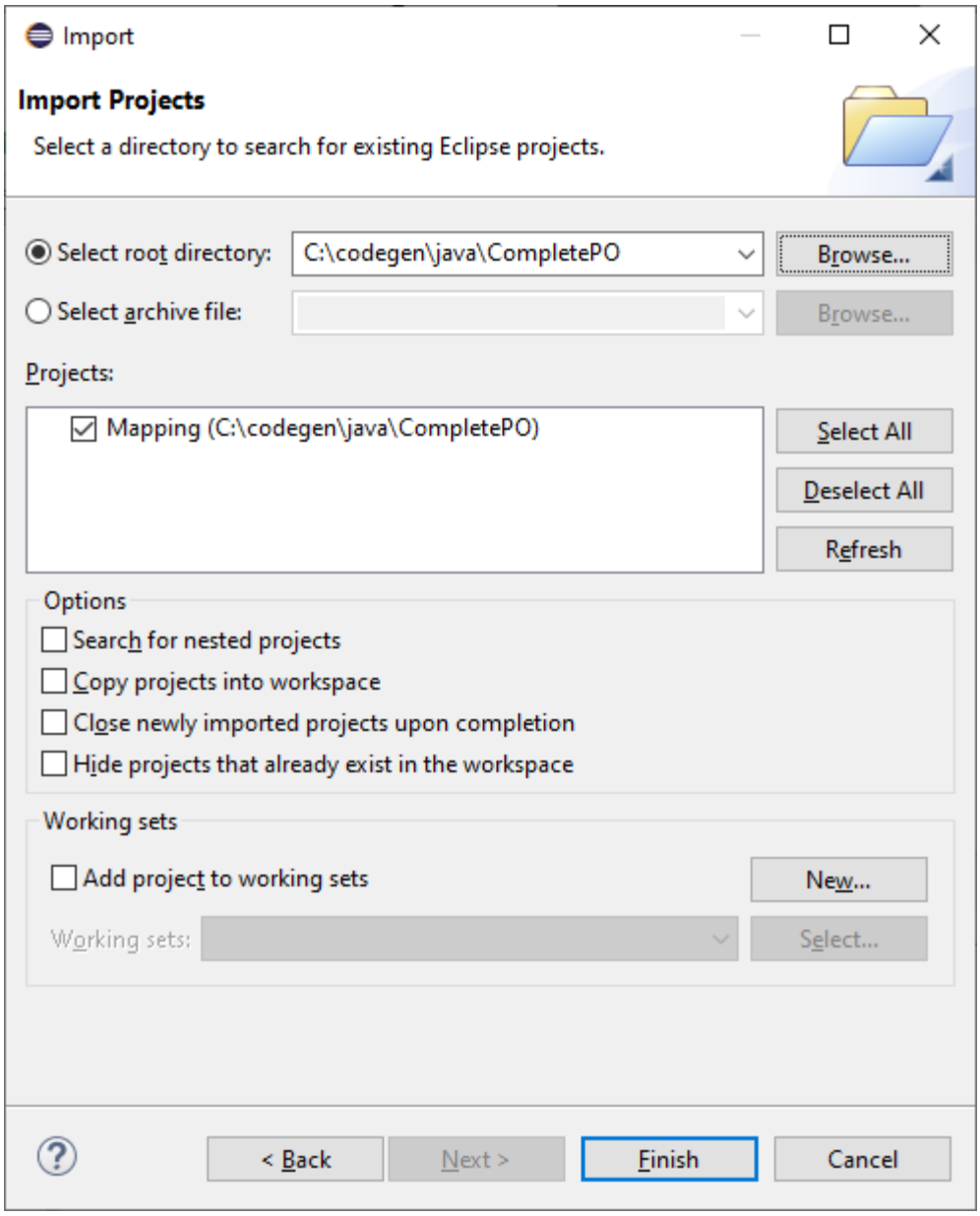
To import the project into Eclipse:

1. If you haven't done so already, run Eclipse and switch to the default Java perspective using the menu command **Window | Open Perspective**.
2. On the **File** menu, click **Import**, and then select **General | Existing Projects into Workspace**.

(C) 2015-2021 Altova GmbH

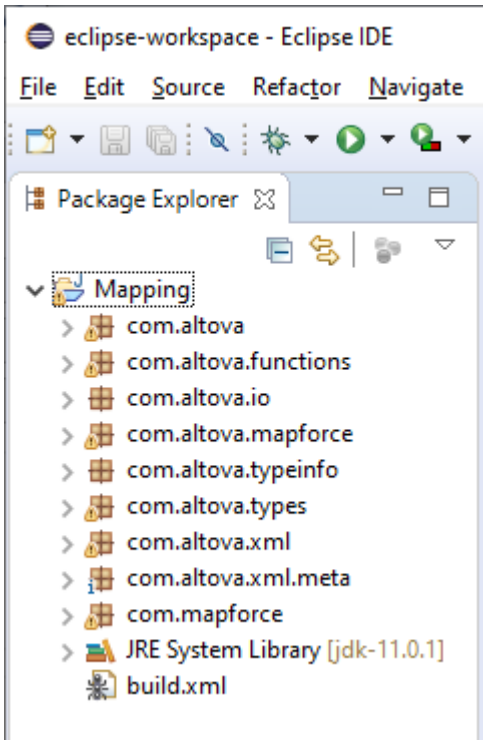

Import
Select
Create new projects from an archive file or directory.
Select an import wizard:
type filter text
General
Archive File
Existing Projects into Workspace
File System
Preferences
Projects from Folder or Archive
< Back
Next >
Finish
Cancel

3. Click **Next**.



4. Browse for the folder where you have previously saved the generated code, and then click **Finish**. The Java project created by MapForce is now available in the Package Explorer view. If you cannot see the Package Explorer view, display it using the menu command **Window | Show View | Package Explorer**.

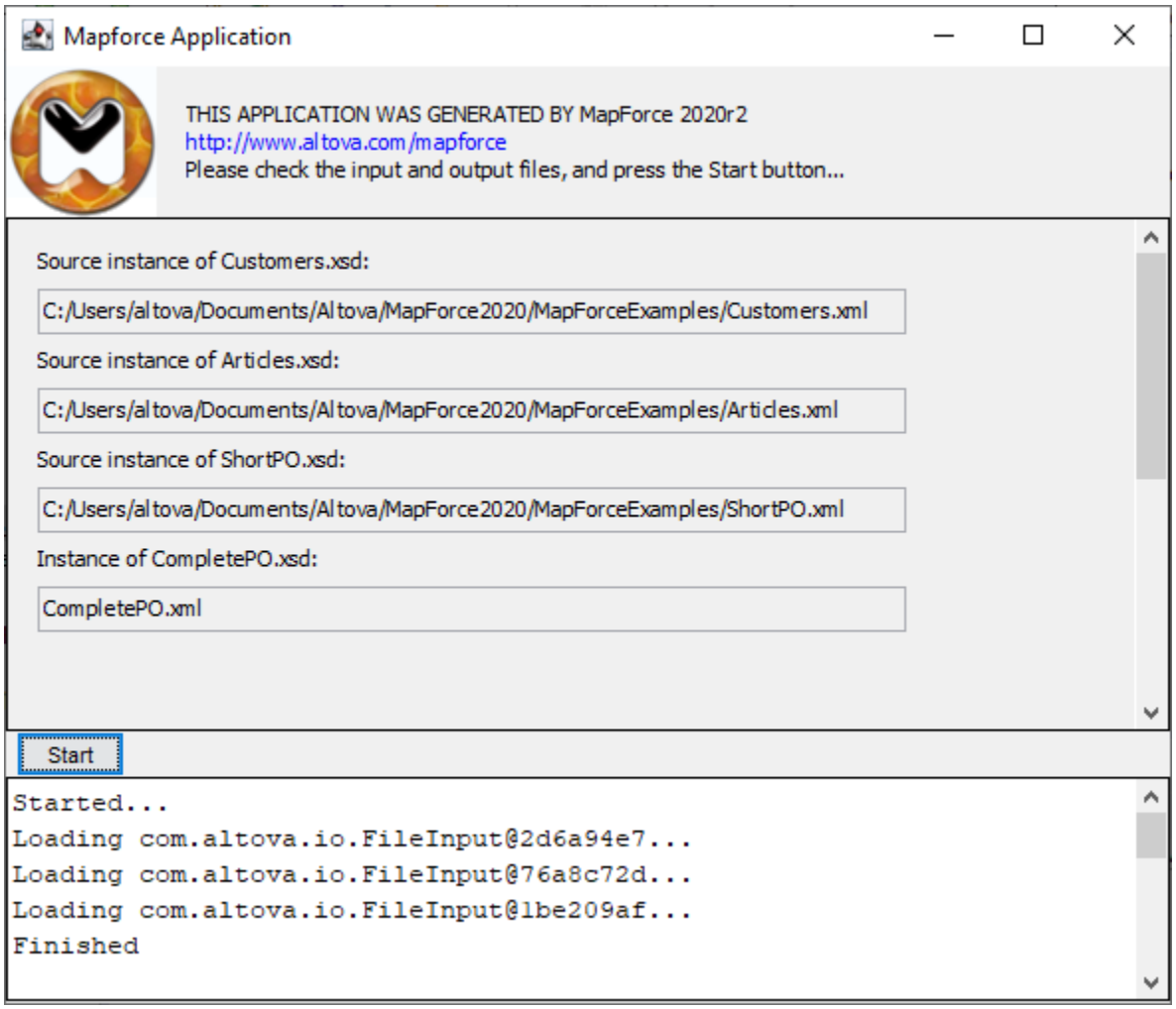
(C) 2015-2021 Altova GmbH



Step 3: Run the project as dialog application

To run the Java project as a GUI application:

1. In the Package Explorer view of Eclipse, click the **MappingApplication.java** file available in the **com.mapforce** package.
2. On the **Run** menu, click **Run As | Java application**.
3. On the MapForce application window, click **Start** to execute the mapping.



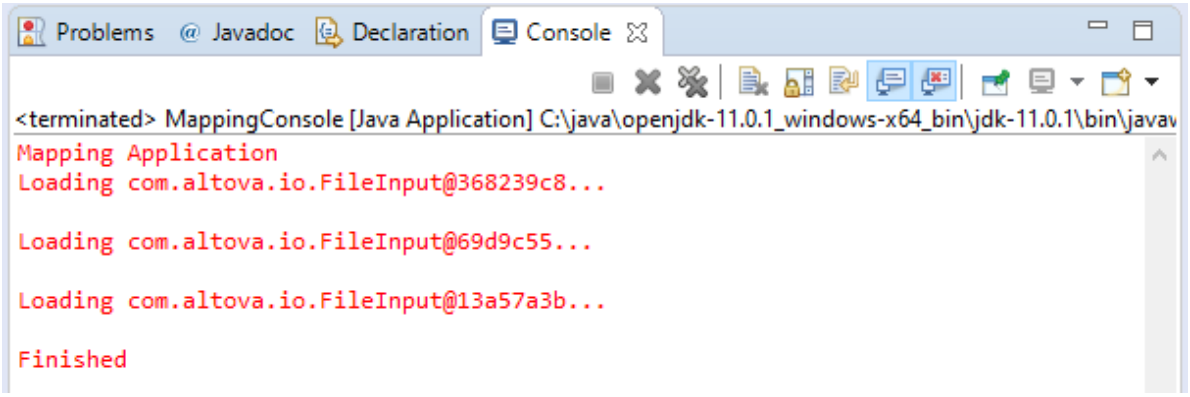
If Eclipse encounters system configuration or run-time errors, you will be prompted, see also Handling JDBC references. Otherwise, the Java application executes the mapping transformation and generates the **CompletePO.xml** at the output path (in this example, the output path is **C:\codegen\Java\CompletePO**).

Step 4: Run the project as console application

To run the Java project as a console application:

1. In the Package Explorer view of Eclipse, click the **MappingConsole.java** file available in the **com.mapforce** package.
2. On the **Run** menu, click **Run As | Java application**.

(C) 2015-2021 Altova GmbH



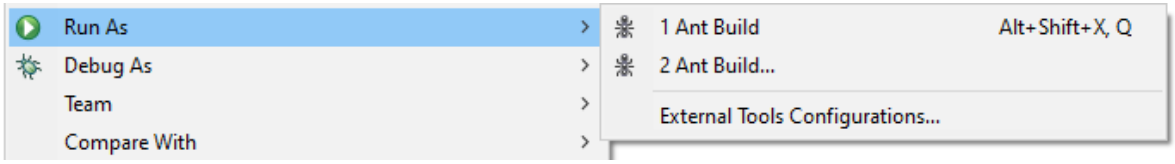
If Eclipse detects system configuration or run-time errors, you will be prompted, see also Handling JDBC references. Otherwise, the Java application executes the mapping transformation and generates the **CompletePO.xml** at the output path (in this example, the output path is **C:\codegen\Java\CompletePO**).

Step 5: Build the JAR file

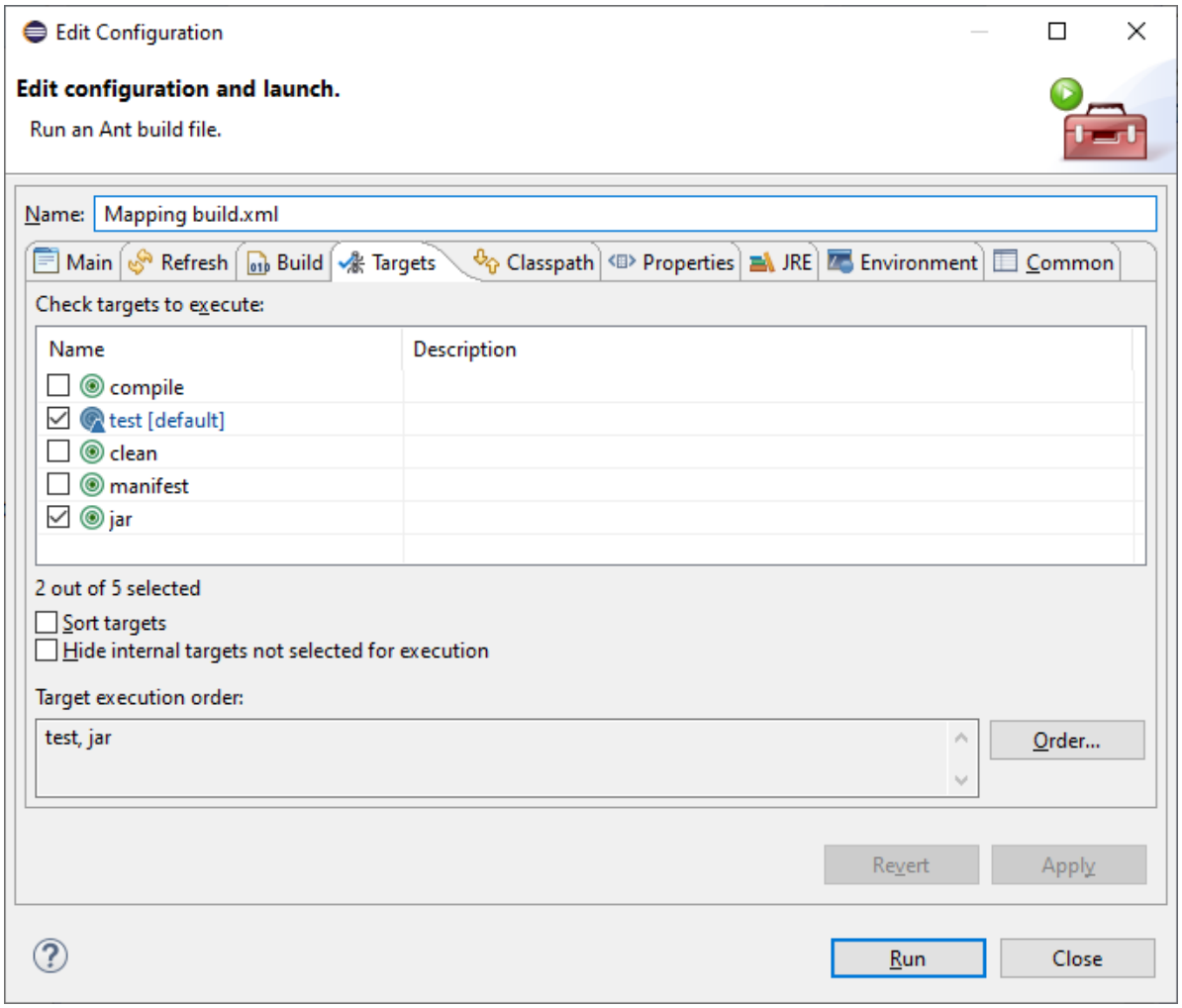
If Apache Ant is bundled with Eclipse, you can build the JAR file directly from the Eclipse IDE. Otherwise, to build the JAR file at the command line outside of Eclipse, Apache Ant must be installed separately.

To build the .jar file with Ant in Eclipse:

1. In the Package Explorer view of Eclipse, right-click the **build.xml** file, and point to the **Run As** menu. In the **Run As** menu, two possible options to run the Ant build file are displayed. If you choose the first option, Eclipse launches the Ant build with the default settings. If you choose the second option, you can change the settings of the Ant build before launching it. Select the second option.



2. Click to enable the targets that you wish to include in the Ant build. In this example, the targets **test** and **jar** are selected.



3. Click **Run**. Eclipse executes the Ant build file and displays the result in the Console view.

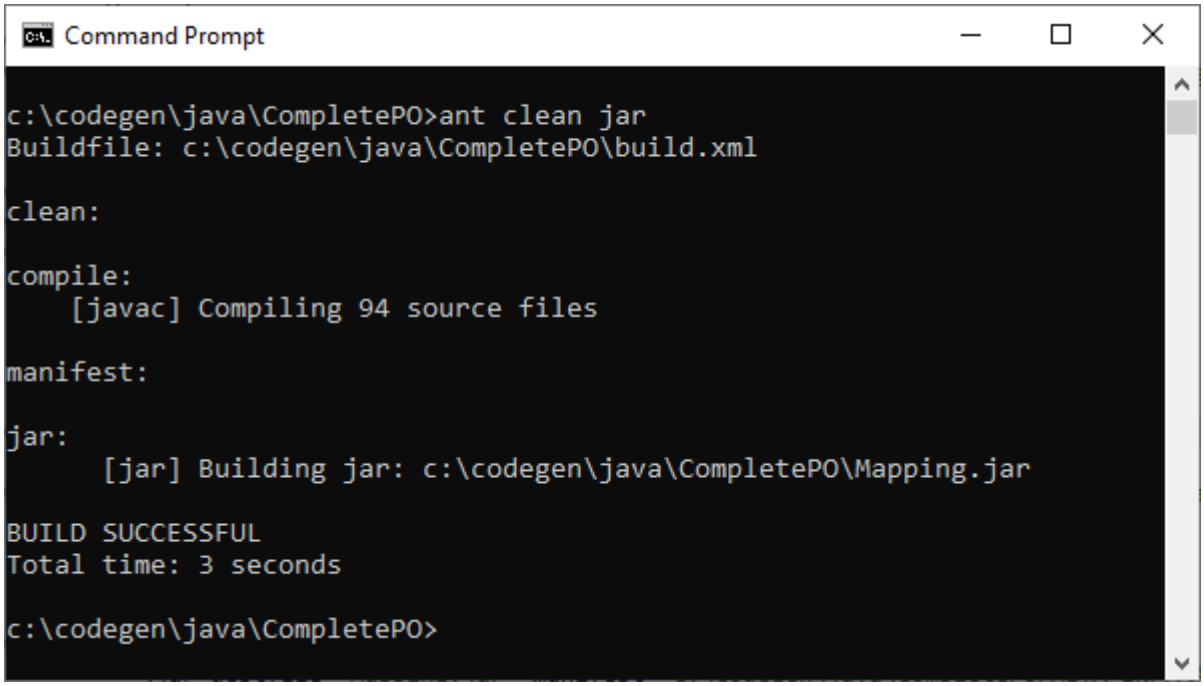
To build the JAR file at the command line:

1. Make sure that the directory where Apache Ant is installed (for example, **C:\apache-ant-1.10.5\bin**) is added to the PATH environment variable.
2. Open a Command Prompt window and change the current directory to the one where you generated Java code (in this example, **C:\codegen\java\CompletePO**). This is also the directory where the **build.xml** file was generated.
3. At the command line, enter:

```
ant clean jar
```

Ant runs the "clean" and "jar" targets from the **build.xml** file (including any dependent targets), and displays the outcome at the command line:

(C) 2015-2021 Altova GmbH

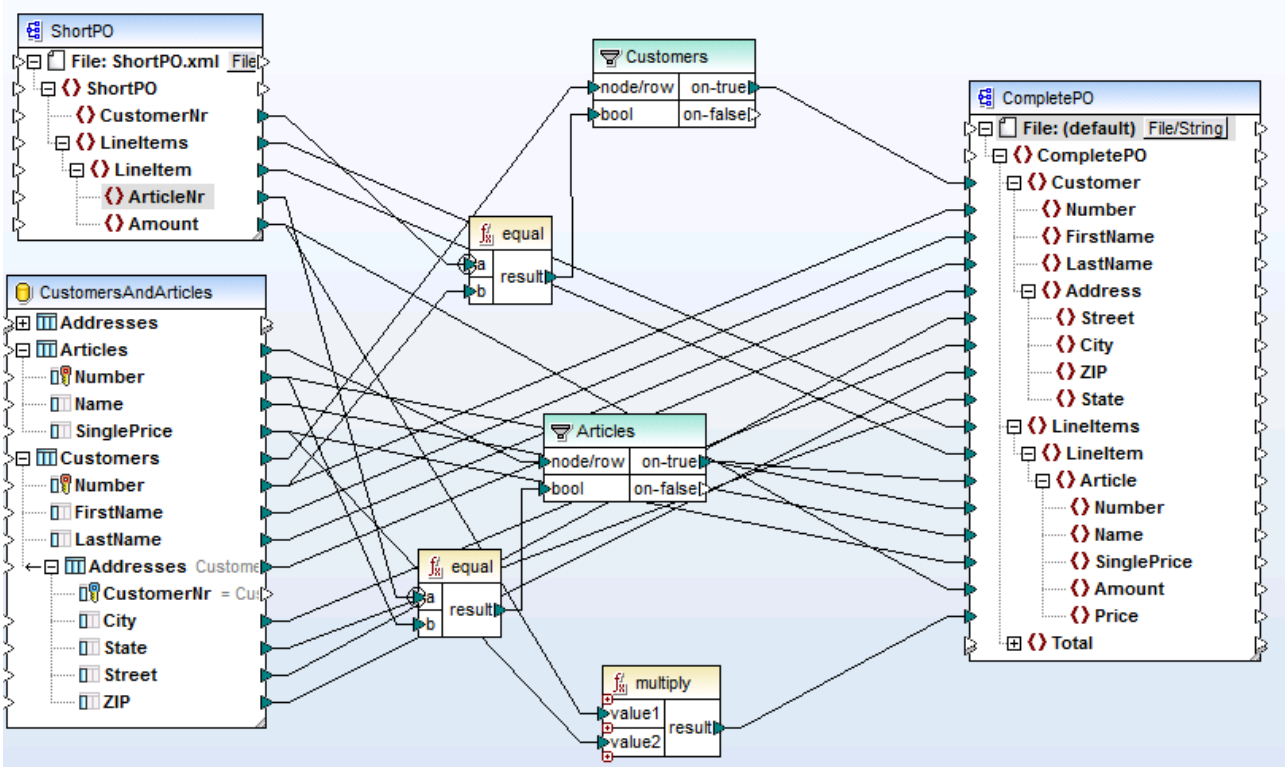

Command Prompt
c:\codegen\java\CompletePO>ant clean jar
Buildfile: c:\codegen\java\CompletePO\build.xml
clean:
compile:
[javac] Compiling 94 source files
manifest:
jar:
[jar] Building jar: c:\codegen\java\CompletePO\Mapping.jar
BUILD SUCCESSFUL
Total time: 3 seconds
c:\codegen\java\CompletePO>

16.5 Integrating MapForce-Generated Code

MapForce-generated code can be integrated, or adapted to your specific application, even though the result of code generation is a complete and fully-functioning application. Some typical scenarios where you might want to change the generated code are as follows:

- Define custom source or target files for the mapping application
- Add custom error handling code
- In C# or Java generated code, you can also change the data type of the mapping input programmatically (for example, from string to stream).

This section provides instructions on how to achieve these goals, based on the **DB_CompletePO.mfd** sample mapping available in the <マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥directory.



DB_CompletePO.mfd mapping sample in MapForce

As illustrated above, the sample mapping consists of two sources and one target:

- **ShortPO.xml** is a source XML file
- **CustomersAndArticles.mdb** is a source database
- **CompletePO.xml** is the target XML file.

In the generated code, these sources and targets will translate to two input and one output parameters supplied to the run method which executes the mapping (as described in the subsequent topics). For now, note the following basic points about code generation:

(C) 2015-2021 Altova GmbH

- The number of source and targets in the mapping design corresponds to the number of mapping parameters to the **run** method in the generated code.
- If you change the number of sources or targets of the mapping, then you will need to re-generate the code accordingly.
- If you make changes to the generated code, and then re-generate the code at the same location, all changes will be overwritten.

If a mapping includes database components, the generated **run** method includes the database connection object at the appropriate location. For example, if the mapping uses three sources (text content, XML content and a database) to map to a single output file, MapForce generates the following **run** method:

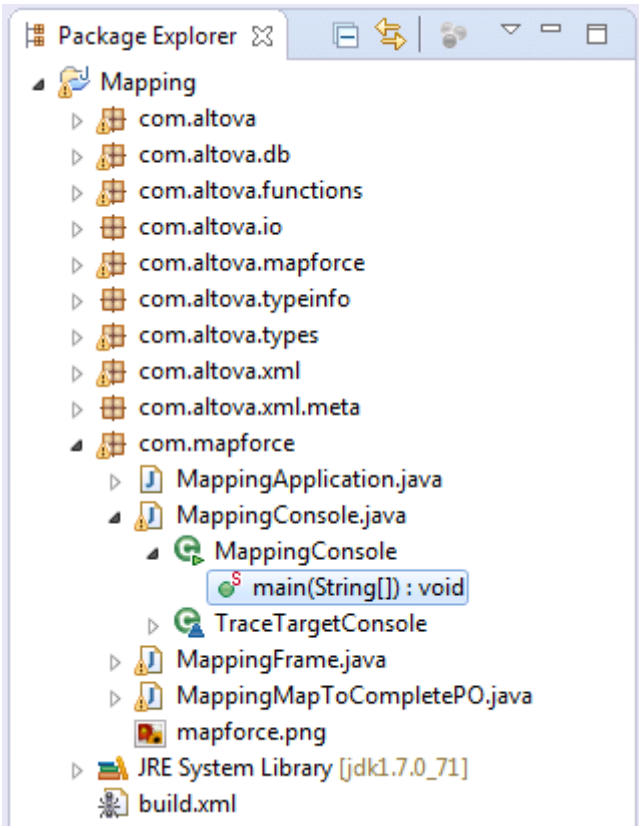
Java

```
void run(Input in1, Input in2, java.sql.Connection dbConn, Output out1);
```

The argument order is important. As you will see in the subsequent examples, you can modify `dbConn` parameters, or use the default parameters generated by MapForce when integrating your code.

16.5.1 Java example

This example uses Eclipse as Java IDE. To begin, generate Java code from the **DB_CompletePO** sample mapping available in the <マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥directory, and then import the project into Eclipse.



Sample MapForce-generated Java application (Eclipse IDE)

To edit the generated Java console application, locate the `main(String[] args)` method of your generated application (see the screen shot above). If you did not change the default base package name before generating code, this method is in the `MappingConsole` class of the `com.mapforce` package. Otherwise, it is in the `MappingConsole` class of your custom defined package.

To edit the generated Java dialog application, locate the place in the code where the `run` method is invoked from your generated application. If you did not change the default base package name before generating code, the run method is invoked from the class called `MappingFrame.java` of the `com.mapforce` package.

The following code sample illustrates an extract from the `main` method in the generated Java console application. The mapping sources and targets are highlighted in yellow and are defined as parameters to the run method. Since this mapping uses a database connection, the corresponding parameter has a special structure. Namely, the connection consists of the connection string (in this case, `jdbc:odbc:;DRIVER=Microsoft Access Driver (*.mdb);DBQ=CustomersAndArticles.mdb`), as well as two empty arguments intended for the **Username** and **Password** (in clear text) for those databases where this data is necessary.

Note that the file paths in the code below have been changed from absolute to relative.

```
com.altova.io.Input ShortPO2Source =
com.altova.io.StreamInput.createInput("ShortPO.xml");
com.altova.io.Output CompletePO2Target = new
com.altova.io.FileOutput("CompletePO.xml");
```


(C) 2015-2021 Altova GmbH

```
MappingMapToCompletePOObject.run(
                  com.altova.db.Dbs.newConnection(
                     "jdbc:odbc:;DRIVER=Microsoft Access Driver
(*.mdb);DBQ=CustomersAndArticles.mdb",
                     "",
                     ""),
                  ShortPO2Source,
                  CompletePO2Target);
```

To define custom mapping source or target files:

- Locate the parameters passed to the `run` method and edit them as required. In the sample above, `com.altova.db.Dbs.newConnection` and `ShortPO2Source` is the mapping input and `CompletePO2Target` is the mapping output.

To add extra error handling code:

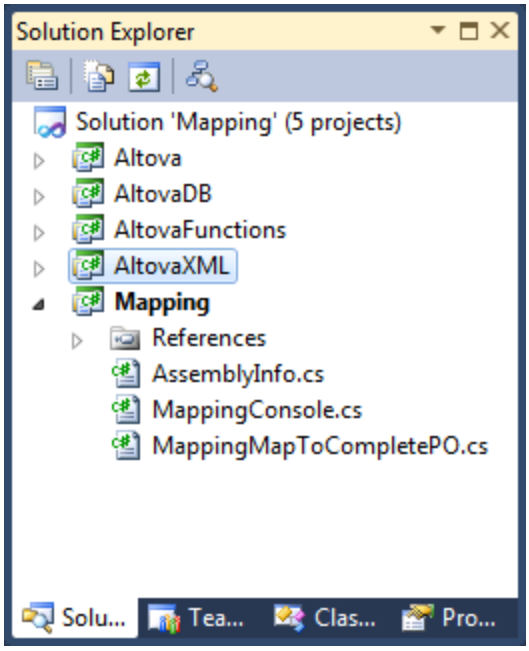
- Edit the code below the `catch (Exception e)` code (in case of a Java console application)
- Edit the code below the `catch (Exception ex)` code (in case of a Java dialog application)

For instructions on how to change the data type of parameters supplied as mapping input/output, see Changing Input and Output Programmatically.

16.5.2 C# example

This example uses the Visual Studio 2010 IDE. To begin, generate C# code from the **DB_CompletePO** sample mapping available in the <マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥directory, and then open the solution in Visual Studio.

By default, the name of the solution file is **Mapping.sln**, and it is located in the **Mapping** subdirectory relative to the directory where you saved the generated code. If you changed the application name from the mapping settings, then the name of the .sln file is changed accordingly. For example, if you changed the application name to **MyApplication**, then the solution file is called **MyApplication.sln**, and it is located in the **MyApplication** subdirectory.



Sample C# solution generated with MapForce

Open the MappingConsole.cs file, and locate the `main(String[] args)` method. The following code sample illustrates an extract from the `main` method. The mapping sources and targets are defined as parameters to the `Run` method. Since this mapping reads data from a database, there is also an input parameter which is a database connection string. If necessary, you can modify the connection string of the database.

Note that the file paths in the code below have been changed from absolute to relative.

```
Altova.IO.Input ShortPO2Source = Altova.IO.StreamInput.createInput("ShortPO.xml");
Altova.IO.Output CompletePO2Target = new Altova.IO.FileOutput("CompletePO.xml");

MappingMapToCompletePOObject.Run(
               "Provider=Microsoft.Jet.OLEDB.4.0; Data Source=CustomersAndArticles.mdb;
",
               ShortPO2Source,
               CompletePO2Target);
```

To define custom mapping source or target files:

- Locate the parameters passed to the Run method and edit them as required. In the sample above, the mapping input is a connection string to the **CustomersAndArticles.mdb** and `ShortPO2Source` . The mapping output is `CompletePO2Target`.

To add extra error handling code:

- Edit the code below the `catch (Exception e)` code

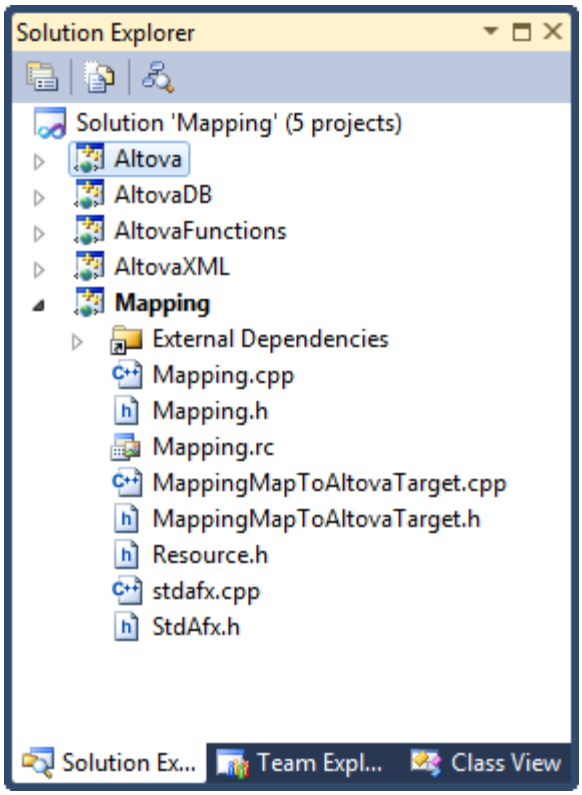
(C) 2015-2021 Altova GmbH

For instructions on how to change the data type of parameters supplied as mapping input/output, see Changing Input and Output Programmatically.

16.5.3 C++ example

This example uses the Visual Studio 2010 IDE. To begin, generate C++ code from the **DB_CompletePO** sample mapping available in the <マイドキュメント>¥Altova¥MapForce2021¥MapForceExamples¥directory, and then open the solution in Visual Studio.

By default, the name of the solution file is **Mapping.sln**, and it is located in the **Mapping** subdirectory relative to the directory where you saved the generated code. If you changed the application name from the mapping settings, then the name of the .sln file is changed accordingly. For example, if you changed the application name to **MyApplication**, then the solution file is called **MyApplication.sln**, and it is located in the **MyApplication** subdirectory.



Sample C++ solution generated with MapForce

Open the **Mapping.cpp** file, and locate the `_tmain` method. The following code sample illustrates an extract from this method. The mapping sources and targets are defined as parameters to the `Run` method. Since this mapping reads data from a database, there is also an input parameter which is a database connection string. If necessary, you can modify the connection string of the database.

Note that the file paths in the code below have been changed from absolute to relative.

```
MappingMapToCompletePO MappingMapToCompletePOObject;
        MappingMapToCompletePOObject.Run(
```

```
            _T("Provider=Microsoft.Jet.OLEDB.4.0; Data Source=CustomersAndArticles.mdb;
"),
            _T("ShortPO.xml"),
            _T("CompletePO.xml"));
```

To define custom mapping source or target files:

- Locate the parameters passed to the `Run` method and edit them as required. In the code sample above, the mapping input is a connection string to the **CustomersAndArticles.mdb** database and `_T("ShortPO.xml")`. The mapping output is `_T("CompletePO.xml")`.

To add extra error handling code:

- Edit the code below the `catch (CAltovaException& e)` code.

16.5.4 Changing Input and Output Programmatically

After generating Java or C# code with MapForce, you can optionally change the data type of the mapping input or output by editing the generated code. More specifically, you can use as mapping parameters objects of types other than those generated by default. For example, instead of having the mapping read the input from a file on the disk, you can provide a string or a stream object as input. Note that this feature is specific to code generated in C# or Java only.

The object types supported as input or output are listed in the first column of the table below. Each subsequent column specifies data formats where that specific type is supported. For a more precise definition of each type, see the "Type definitions" section below.

	XML	Microsoft Excel*	EDI (includes X12, HL7)*	FlexText*	CSV/Text
Files	Yes	Yes	Yes	Yes	Yes
Streams	Yes	Yes	Yes	Yes	Yes
Strings	Yes	–	Yes	Yes	Yes
Reader/Writer	Yes	–	Yes	Yes	Yes
DOM documents	Yes	–	–	–	–

** Formats supported only in MapForce Enterprise Edition*

To change the data type of the mapping input or output:

1. Generate C# or Java code from a mapping.
2. In the generated code, find the call to the `run` method (in Java) or `Run` method (in C#), as follows:

   a. If using C#, open the **MappingConsole.cs** file.

(C) 2015-2021 Altova GmbH

b. If using Java, open the **MappingConsole.java** (the console program) or the **MappingFrame.java** file (the GUI program).

Note: The name of the file may be different if you have changed the application name in the mapping settings. For example, if you changed it to "MyApp", then name of the generated file becomes **MyAppConsole.js** and **MyAppConsole.java**, and **MyAppFrame.java**, respectively.

3. Create an instance of the required type (see the "Type definitions" section).
4. Supply the declared objects as parameters to the `run` method (in Java) or `Run` method (in C#), as shown in the examples below.

The `run` method is the most important method of generated mapping classes. It has one parameter for each *static* source or input component in the mapping, and a final parameter for the output component. If your mapping contains components that process multiple files dynamically, the respective parameters do not appear in generated code, because in this case the file names are processed dynamically inside the mapping.

Type definitions

In C#, the types that you can provide as parameters to the `Run` method are classes defined in the `Altova.IO` namespace. The base classes are `Altova.IO.Input` and `Altova.IO.Output`, respectively.

C# types

Files	`Altova.IO.FileInput(string filename)` `Altova.IO.FileOutput(string filename)`
Streams	`Altova.IO.StreamInput(System.IO.Stream stream)` `Altova.IO.StreamOutput(System.IO.Stream stream)`
Strings	`Altova.IO.StringInput(string content)` `Altova.IO.StringOutput(System.Text.StringBuilder sbuilder)`
Reader/Writer	`Altova.IO.ReaderInput(System.IO.TextReader reader)` `Altova.IO.WriterOutput(System.IO.TextWriter writer)`
DOM documents	`Altova.IO.DocumentInput(System.Xml.XmlDocument document)` `Altova.IO.DocumentOutput(System.Xml.XmlDocument document)`

In Java, the types that you can provide as parameters to the `run` method are classes defined in the `com.altova.io` package. The base classes are `com.altova.io.Input` and `com.altova.io.Output`, respectively.

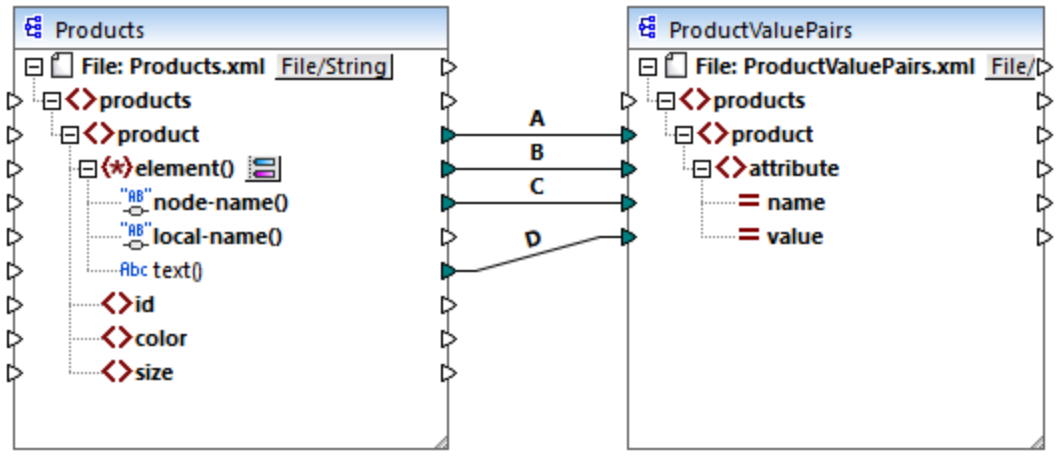
Java types

Files	`com.altova.io.FileInput(String filename)` `com.altova.io.FileOutput(String filename)`
Streams	`com.altova.io.StreamInput(java.io.InputStream stream)` `com.altova.io.StreamOutput(String filename)`
Strings	`com.altova.io.StringInput(String content)` `com.altova.io.StringOutput()`
Reader/Writer	`com.altova.io.ReaderInput(java.io.Reader reader)` `com.altova.io.WriterOutput(java.io.Writer writer)`
DOM documents	`com.altova.io.DocumentInput(org.w3c.dom.Document document)`

	`com.altova.io.DocumentOutput(org.w3c.dom.Document document)`

Example

To illustrate changing the input and output programmatically, we will use the **ConvertProducts.mfd** mapping as a model. After installing MapForce and running it at least once, you can find this mapping in the following directory: **C:\Users\<username>\Documents\Altova\MapForce2021\MapForceExamples\Tutorials**.

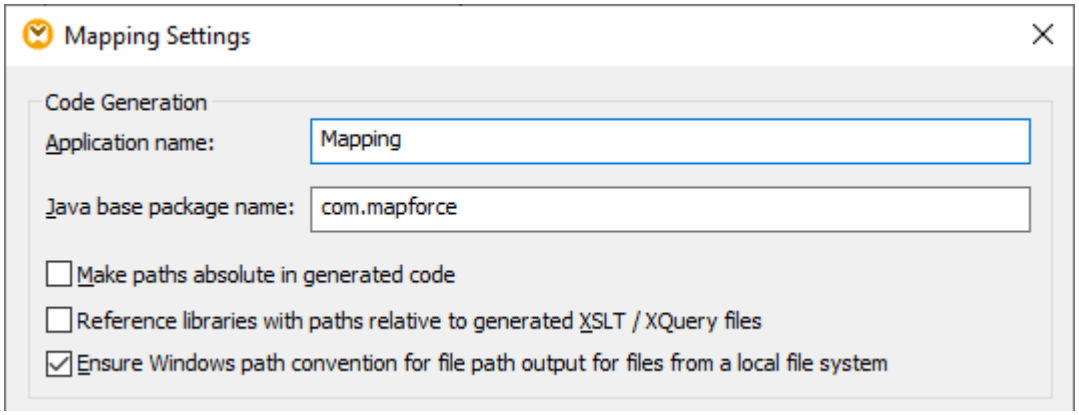


ConvertProducts.mfd

As illustrated above, the mapping converts data from a source XML document to another XML document. Our goals are as follows:

1. Generate Java and C# program code from this mapping.
2. Change the data type of the source component to a string type.
3. Change the data type of the target component to a string writer type.

To generate the program code, open the **ConvertProducts.mfd** mapping and run the **File | Generate code in | C#** (or **Java**) command. For the scope of this example, we will assume that the mapping settings of **ConvertProducts.mfd** are the default ones.



This example uses the following target directories for the generated code (feel free to change the path if necessary):

(C) 2015-2021 Altova GmbH

- **C:\codegen\cs\ConvertProducts**, for C#
- **C:\codegen\java\ConvertProducts**, for Java

Having generated the program code, open the **MappingConsole.cs** (in C#) or **MappingConsole.java** (in Java) and find the following lines:

C#

```
Altova.IO.Input Products2Source = Altova.IO.StreamInput.createInput("Products.xml");
Altova.IO.Output ProductValuePairs2Target = new
Altova.IO.FileOutput("ProductValuePairs.xml");
```

Java

```
com.altova.io.Input Products2Source =
com.altova.io.StreamInput.createInput("Products.xml");
com.altova.io.Output ProductValuePairs2Target = new
com.altova.io.FileOutput("ProductValuePairs.xml");
```

Comment out the lines above and change the code as follows:

C#

```
//Altova.IO.Input Products2Source = Altova.IO.StreamInput.createInput("Products.xml");
//Altova.IO.Output ProductValuePairs2Target = new
Altova.IO.FileOutput("ProductValuePairs.xml");

Altova.IO.Input Products2Source = new Altova.IO.StringInput("<?xml version=\"1.0\"
encoding=\"UTF-8\"?>\r\n" +
                                        "<products
xmlns:xsi=\"http://www.w3.org/2001/XMLSchema-instance\"
xsi:noNamespaceSchemaLocation=\"products.xsd\">\r\n" +
                                        "   <product>\r\n" +
                                        "      <id>100</id>\r\n" +
                                        "      <color>blue</color>\r\n" +
                                        "      <size>XXL</size>\r\n" +
                                        "   </product>\r\n" +
                                        "</products>\r\n");

System.IO.StringWriter writer = new System.IO.StringWriter(new
System.Text.StringBuilder());
Altova.IO.Output ProductValuePairs2Target = new Altova.IO.WriterOutput(writer);

try
{
    MappingMapToProductValuePairsObject.Run(Products2Source, ProductValuePairs2Target);

    // Print out the writer object
    Console.Write(writer.ToString());
}
finally
{
    Products2Source.Close();
    ProductValuePairs2Target.Close();
}
```

Java

```
//com.altova.io.Input Products2Source =
com.altova.io.StreamInput.createInput("Products.xml");
//com.altova.io.Output ProductValuePairs2Target = new
com.altova.io.FileOutput("ProductValuePairs.xml");

com.altova.io.Input Products2Source = new com.altova.io.StringInput("<?xml
version=\"1.0\" encoding=\"UTF-8\"?>\r\n" +
                "<products xmlns:xsi=\"http://www.w3.org/2001/XMLSchema-instance\"
xsi:noNamespaceSchemaLocation=\"products.xsd\">\r\n" +
                "   <product>\r\n" +
                "      <id>100</id>\r\n" +
                "      <color>blue</color>\r\n" +
                "      <size>XXL</size>\r\n" +
                "   </product>\r\n" +
                "</products>\r\n");

java.io.StringWriter writer = new java.io.StringWriter();
com.altova.io.Output ProductValuePairs2Target = new com.altova.io.WriterOutput(writer);

try {
   MappingMapToProductValuePairsObject.run(Products2Source, ProductValuePairs2Target);

   // Print out the writer object
   System.out.print(writer.toString());

   } finally {
      (Products2Source).close();
      ProductValuePairs2Target.close();
}
```

In the C# and Java code listings above, the following happens:

- The two original lines that provide the input and output to the `run` method were commented out. Consequently, the mapping application no longer reads data from **Products.xml**. In fact, we did not even need to copy this file to the program's working directory.
- The type `Products2Source` has been declared as a `StringInput` that provides the content of the input XML file to be processed.
- The type `ProductValuePairs2Target` has been declared as a `WriterOutput` type that takes a string writer as argument.
- After the mapping completes running, the contents of the string writer is printed out to the console.

Usage guidelines for streams and Reader/Writer objects

When using binary streams or Reader/Writer objects as input or output to the mapping, note the following:

- Binary stream objects and Reader/Writer objects are expected to be opened and ready to use before calling the `run` method.
- By default, the `run` method closes the stream when finished. To prevent this behavior, insert (or uncomment) the following line before calling the `run` method:

C#

```
MappingMapToSomething.CloseObjectsAfterRun = false;
```


(C) 2015-2021 Altova GmbH

Java

```
MappingMapToSomething.setCloseObjectsAfterRun(false);
```

Note: Make sure to change `MappingMapToSomething` to the name of the mapping object as applicable to your generated code.

Usage guidelines for strings

In Java, the constructor of `StringOutput` does not take an argument. The string content produced by the mapping can be accessed by calling the `getString()` method, for example:

Java

```
com.altova.io.Input Products2Source =
com.altova.io.StreamInput.createInput("Products.xml");
com.altova.io.StringOutput ProductValuePairs2Target = new com.altova.io.StringOutput();

try {
   // Run the mapping
   MappingMapToProductValuePairsObject.run(Products2Source, ProductValuePairs2Target);
   // Get the string object
   String str = ProductValuePairs2Target.getString().toString();
}
```

In C#, the constructor of `StringOutput` takes a parameter of type `StringBuilder` which you need to declare beforehand. If the `StringBuilder` object already contains data, the mapping output will be appended to it.

C#

```
Altova.IO.Input Products2Source = Altova.IO.StreamInput.createInput("Products.xml");
System.Text.StringBuilder sb = new System.Text.StringBuilder();
Altova.IO.Output ProductValuePairs2Target = new Altova.IO.StringOutput(sb);

try
{
   // Run the mapping
   MappingMapToProductValuePairsObject.Run(Products2Source, ProductValuePairs2Target);
   // Get the string output
   String str = sb.ToString();
}
```

To run these code listings, you can use the same generated project as in the previous example. Make sure, however, to copy the file **Products.xml** from **C:\Users\<username>\Documents\Altova\MapForce2021\MapForceExamples\Tutorials** to your program's working directory, since the mapping code reads data from this file.

Usage guidelines for DOM documents

When using DOM documents as mapping input or output, note the following:

- The document instance supplied as parameter to the `DocumentOutput` constructor must be empty.
- After calling `run`, the DOM Document generated by the constructor of `DocumentOutput` already contains the mapping output, and you can manipulate the document as necessary.

16.6 Generating Code from XML Schemas or DTDs

When you generate code from a mapping, MapForce generates a complete application that executes all steps of the mapping automatically. Optionally, you can generate libraries for all the XML schemas used in the mapping. These allow your code to easily create or read XML instances that are used or created by the mapping code.

To generate libraries for all the XML schemas used in the mapping, select the **Generate Wrapper Classes** check box from the code generator options. Next time when you generate code, MapForce will create not only the mapping application, but also wrapper classes for all schemas used in the mapping, as follows:

C++ or C#	Java	Purpose
Altova	`com.altova`	Base library containing common runtime support, identical for every schema.
AltovaXML	`com.altova.xml`	Base library containing runtime support for XML, identical for every schema.
[YourSchema]	`com.YourSchema`	A library containing declarations generated from the input schema, named as the schema file or DTD. This library is a DOM (W3C Document Object Model) wrapper that allows you to read, modify and create XML documents easily and safely. All data is held inside the DOM, and there are methods for extracting data from the DOM, and to update and create data into the DOM. The generated C++ code supports either Microsoft MSXML or Apache Xerces 3. The syntax for using the generated code is generally similar for both DOM implementations, except for a few slight differences (for example, Xerces supports more overloaded functions). The generated C# code uses the .NET standard System.XML library as the underlying DOM implementation. The generated Java code uses JAXP (Java API for XML Processing) as the underlying DOM interface.

While prototyping an application from a frequently changing XML schema, you may need to frequently generate code to the same directory, so that the schema changes are immediately reflected in the code. Note that the generated test application and the Altova libraries are overwritten every time when you generate code into the same target directory. Therefore, do not add code to the generated test application. Instead, integrate the Altova libraries into your project (see Integrating Schema Wrapper Libraries).

In addition to the base libraries listed above, some supporting libraries are also generated. The supporting libraries are used by the Altova base libraries and are not meant for custom integrations, since they are subject to change.

(C) 2015-2021 Altova GmbH

Name generation and namespaces

MapForce generates classes corresponding to all declared elements or complex types which redefine any complex type in your XML Schema, preserving the class derivation as defined by extensions of complex types in your XML Schema. In the case of complex schemas which import schema components from multiple namespaces, MapForce preserves this information by generating the appropriate C# or C++ namespaces or Java packages.

Generally, the code generator tries to preserve the names for generated namespaces, classes and members from the original XML Schema. Characters that are not valid in identifiers in the target language are replaced by a "_". Names that would collide with other names or reserved words are made unique by appending a number. Name generation can be influenced by changing the default settings in the SPL template.

The namespaces from the XML Schema are converted to packages in Java or namespaces in C# or C++ code, using the namespace prefix from the schema as code namespace. The complete library is enclosed in a package or namespace derived from the schema file name, so you can use multiple generated libraries in one program without name conflicts.

Data Types

XML Schema has a more elaborate data type model than Java, C# or C++. Code Generator converts the built-in XML Schema types to language-specific primitive types, or to classes delivered with the Altova library. Complex types and derived types defined in the schema are converted to classes in the generated library. Enumeration facets from simple types are converted to symbolic constants.

The mapping of simple types can be configured in the SPL template, see SPL Reference.

If your XML instance files use schema types related to time and duration, these are converted to Altova native classes in the generated code. For information about the Altova library classes, see:

- Reference to Generated Classes (C++)
- Reference to Generated Classes (C#)
- Reference to Generated Classes (Java)

For information about type conversion and other details applicable to each language, see:

- About Schema Wrapper Libraries (C++)
- About Schema Wrapper Libraries (C#)
- About Schema Wrapper Libraries (Java)

Memory management

A DOM tree is comprised of nodes, which are always owned by a specific DOM document - even if the node is not currently part of the document's content. All generated classes are references to the DOM nodes they represent, not values. This means that assigning an instance of a generated class does not copy the value, it only creates an additional reference to the same data.

XML Schema support

The following XML Schema constructs are translated into code:

a) XML namespaces

b) Simple types:

- Built-in XML schema types
- Simple types derived by extension
- Simple types derived by restriction
- Facets
- Enumerations
- Patterns

c) Complex types:

- Built-in anyType node
- User-defined complex types
- Derived by extension: Mapped to derived classes
- Derived by restriction
- Complex content
- Simple content
- Mixed content

The following advanced XML Schema features are not supported (or not fully supported) in generated wrapper classes:

- Wildcards: `xs:any` and `xs:anyAttribute`
- Content models (sequence, choice, all). Top-level compositor is available in SPL, but is not enforced by generated classes.
- Default and fixed values for attributes. These are available in SPL, but are not set or enforced by generated classes.
- The attributes `xsi:type`, abstract types. When you need to write the xsi:type attribute, use the `SetXsiType()` method of the generated classes.
- Union types: not all combinations are supported.
- Substitution groups are partially supported (resolved like "choice").
- Attribute `nillable="true"` and `xsi:nil`
- Uniqueness constraints
- Identity constraints (`key` and `keyref`)

16.6.1 About Schema Wrapper Libraries (C++)

Character Types

The generated C++ code can be compiled with or without Unicode support. Depending on this setting, the types `string_type` and `tstring` will both be defined as `std::string` or `std::wstring`, consisting of narrow or wide characters. To use Unicode characters in your XML file that are not representable with the current 8-bit character set, Unicode support must be enabled. Pay special attention to the `_T()` macros. This macro ensures that string constants are stored correctly, whether you're compiling for Unicode or non-Unicode programs.

Data Types

The default mapping of XML Schema types to C++ data types is:

(C) 2015-2021 Altova GmbH

XML Schema	C++	Remarks
xs:string	string_type	string_type is defined as std::string or std:wstring
xs:boolean	bool	
xs:decimal	double	C++ does not have a decimal type, so double is used.
xs:float, xs:double	double	
xs:integer	__int64	xs:integer has unlimited range, mapped to __int64 for efficiency reasons.
xs:nonNegativeInteger	unsigned __int64	see above
xs:int	int	
xs:unsignedInt	unsigned int	
xs:dateTime, date, time, gYearMonth, gYear, gMonthDay, gDay, gMonth	altova::DateTime	
xs:duration	altova::Duration	
xs:hexBinary and xs:base64Binary	std::vector<unsigned char>	Encoding and decoding of binary data is done automatically.
xs:anySimpleType	string_type	

All XML Schema types not contained in this list are derived types, and mapped to the same C++ type as their respective base type.

Generated Classes

For each type in the schema, a class is generated that contains a member for each attribute and element of the type. The members are named the same as the attributes or elements in the original schema (in case of possible collisions, a number is appended). For simple types, assignment and conversion operators are generated. For simple types with enumeration facets, the methods `GetEnumerationValue()` and `SetEnumerationValue(int)` can be used together with generated constants for each enumeration value. In addition, the method `StaticInfo()` allows accessing schema information as one of the following types:

altova::meta::SimpleType
altova::meta::ComplexType

Classes generated from complex types include the method `SetXsiType()`, which enables you to set the `xsi:type` attribute of the type. This method is useful when you want to create XML instance elements of a derived type.

In addition to the classes for the types declared in the XML Schema, a document class (identified with "CDoc" below) is generated. It contains all possible root elements as members, and various other methods. For more information about the class, see [YourSchema]::[CDoc].

Note: The actual class name depends on the name of the .xsd schema.

For each member attribute or element of a schema type, a new class is generated. For more information about such classes, see:

[YourSchema]::MemberAttribute
[YourSchema]::MemberElement

Note: The actual class names depend on the name of the schema attribute or element.

See also Example: Using the Schema Wrapper Libraries.

Error Handling

Errors are reported by exceptions. The following exception classes are defined in the namespace `altova`:

Class	Base Class	Description
`Error`	`std::logic_error`	Internal program logic error (independent of input data).
`Exception`	`std::runtime_error`	Base class for runtime errors.
`InvalidArgumentsException`	`Exception`	A method was called with invalid argument values.
`ConversionException`	`Exception`	Exception thrown when a type conversion fails.
`StringParseException`	`ConversionException`	A value in the lexical space cannot be converted to value space.
`ValueNotRepresentableException`	`ConversionException`	A value in the value space cannot be converted to lexical space.
`OutOfRangeException`	`ConversionException`	A source value cannot be represented in target domain.
`InvalidOperationException`	`Exception`	An operation was attempted that is not valid in the given context.
`DataSourceUnavailableException`	`Exception`	A problem occurred while loading an XML instance.
`DataTargetUnavailableException`	`Exception`	A problem occurred while saving an XML instance.

All exception classes contain a message text and a pointer to a possible inner exception.

Method	Purpose
`string_type message()`	Returns a textual description of the exception.
`std::exception inner()`	Returns the exception that caused this exception, if available, or NULL.

(C) 2015-2021 Altova GmbH

Accessing schema information

The generated library allows accessing static schema information via the following classes. All methods are declared as `const`. The methods that return one of the metadata classes return a NULL object if the respective property does not exist.

`altova::meta::Attribute`
`altova::meta::ComplexType`
`altova::meta::Element`
`altova::meta::SimpleType`

16.6.2 About Schema Wrapper Libraries (C#)

The default mapping of XML Schema types to C# data types is as follows.

XML Schema	C#	Remarks
`xs:string`	`string`	
`xs:boolean`	`bool`	
`xs:decimal`	`decimal`	xs:decimal has unlimited range and precision, mapped to decimal for efficiency reasons.
`xs:float, xs:double`	`double`	
`xs:long`	`long`	
`xs:unsignedLong`	`ulong`	
`xs:int`	`int`	
`xs:unsignedInt`	`uint`	
`xs:dateTime, date, time, gYearMonth, gYear, gMonthDay, gDay, gMonth`	`Altova.Types.DateTime`	
`xs:duration`	`Altova.Types.Duration`	
`xs:hexBinary and xs:base64Binary`	`byte[]`	Encoding and decoding of binary data is done automatically.
`xs:anySimpleType`	`string`	

All XML Schema types not contained in this list are derived types, and mapped to the same C# type as their respective base type.

Generated Classes

For each type in the schema, a class is generated that contains a member for each attribute and element of the type. The members are named the same as the attributes or elements in the original schema (in case of possible collisions, a number is appended). For simple types, assignment and conversion operators are

generated. For simple types with enumeration facets, the methods `GetEnumerationValue()` and `SetEnumerationValue(int)` can be used together with generated constants for each enumeration value. In addition, the method `StaticInfo()` allows accessing schema information as one of the following types:

`Altova.Xml.Meta.SimpleType`
`Altova.Xml.Meta.ComplexType`

Classes generated from complex types include the method `SetXsiType()`, which enables you to set the `xsi:type` attribute of the type. This method is useful when you want to create XML instance elements of a derived type.

In addition to the classes for the types declared in the XML Schema, a document class (identified with "Doc" below) is generated. It contains all possible root elements as members, and various other methods. For more information about the class, see `[YourSchema].[Doc]`.

Note: The actual class name depends on the name of the .xsd schema.

For each member attribute or element of a schema type, a new class is generated. For more information about such classes, see:

`[YourSchemaType].MemberAttribute`
`[YourSchemaType].MemberElement`

Note: The actual class names depend on the name of the schema attribute or element.

Error Handling

Errors are reported by exceptions. The following exception classes are defined in the namespace Altova:

Class	Base Class	Description
`ConversionException`	`Exception`	Exception thrown when a type conversion fails
`StringParseException`	`ConversionException`	A value in the lexical space cannot be converted to value space.
`DataSourceUnavailableException`	`System.Exception`	A problem occurred while loading an XML instance.
`DataTargetUnavailableException`	`System.Exception`	A problem occurred while saving an XML instance.

In addition, the following .NET exceptions are commonly used:

Class	Description
`System.Exception`	Base class for runtime errors
`System.ArgumentException`	A method was called with invalid argument values, or a type conversion failed.
`System.FormatException`	A value in the lexical space cannot be converted to value space.

(C) 2015-2021 Altova GmbH

Class	Description
System.InvalidCastException	A value cannot be converted to another type.
System.OverflowException	A source value cannot be represented in target domain.

Accessing schema information

The generated library allows accessing static schema information via the following classes:

Altova.Xml.Meta.Attribute
Altova.Xml.Meta.ComplexType
Altova.Xml.Meta.Element
Altova.Xml.Meta.SimpleType

The properties that return one of the metadata classes return null if the respective property does not exist.

16.6.3 About Schema Wrapper Libraries (Java)

The default mapping of XML Schema types to Java data types is as follows:

XML Schema	Java	Remarks
xs:string	String	
xs:boolean	boolean	
xs:decimal	java.math.BigDecimal	
xs:float, xs:double	double	
xs:integer	java.math.BigInteger	
xs:long	long	
xs:unsignedLong	java.math.BigInteger	Java does not have unsigned types.
xs:int	int	
xs:unsignedInt	long	Java does not have unsigned types.
xs:dateTime, date, time, gYearMonth, gYear, gMonthDay, gDay, gMonth	com.altova.types.DateTime	
xs:duration	com.altova.types.Duration	
xs:hexBinary and xs:base64Binary	byte[]	Encoding and decoding of binary data is done automatically.
xs:anySimpleType	string	

All XML Schema types not contained in this list are derived types, and mapped to the same Java type as their respective base type.

Generated Classes

For each type in the schema, a class is generated that contains a member for each attribute and element of the type. The members are named the same as the attributes or elements in the original schema (in case of possible collisions, a number is appended). For simple types, assignment and conversion operators are generated. For simple types with enumeration facets, the methods `GetEnumerationValue()` and `SetEnumerationValue(int)` can be used together with generated constants for each enumeration value. In addition, the method `StaticInfo()` allows accessing schema information as one of the following types:

`com.altova.xml.meta.SimpleType`
`com.altova.xml.meta.ComplexType`

Classes generated from complex types include the method `SetXsiType()`, which enables you to set the `xsi:type` attribute of the type. This method is useful when you want to create XML instance elements of a derived type.

In addition to the classes for the types declared in the XML Schema, a document class (identified with "Doc" below) is generated. It contains all possible root elements as members, and various other methods. For more information about the class, see `com.[YourSchema].[Doc]`.

Note: The actual class name depends on the name of the .xsd schema.

For each member attribute or element of a schema type, a new class is generated. For more information about such classes, see:

`com.[YourSchema].[YourSchemaType].MemberAttribute`
`com.[YourSchema].[YourSchemaType].MemberElement`

Note: The actual class names depend on the name of the schema attribute or element.

Error Handling

Errors are reported by exceptions. The following exception classes are defined in the namespace com.altova:

Class	Base Class	Description
`SourceInstanceUnvailableException`	Exception	A problem occurred while loading an XML instance.
`TargetInstanceUnavailableException`	Exception	A problem occurred while saving an XML instance.

In addition, the following Java exceptions are commonly used:

Class	Description
`java.lang.Error`	Internal program logic error (independent of input data)
`java.lang.Exception`	Base class for runtime errors

(C) 2015-2021 Altova GmbH

Class	Description
java.lang.IllegalArgumentsException	A method was called with invalid argument values, or a type conversion failed.
java.lang.ArithmeticException	Exception thrown when a numeric type conversion fails.

Accessing schema information

The generated library allows accessing static schema information via the following classes:

com.altova.xml.meta.Attribute
com.altova.xml.meta.ComplexType
com.altova.xml.meta.Element
com.altova.xml.meta.SimpleType

The properties that return one of the metadata classes return null if the respective property does not exist.

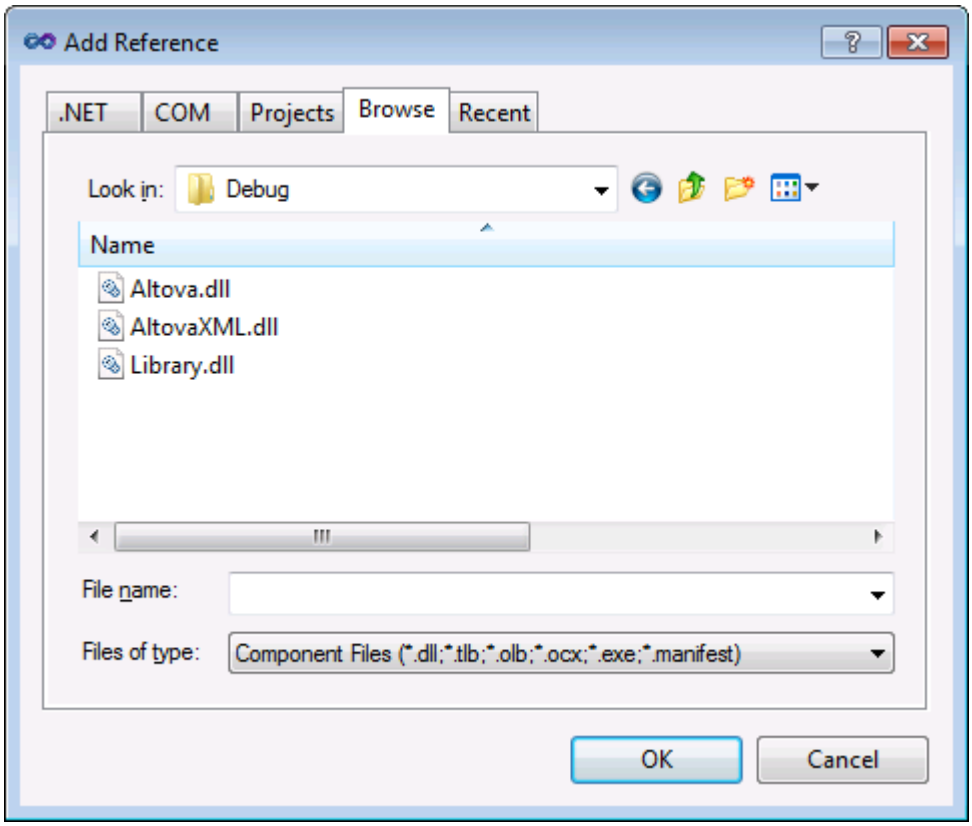
16.6.4 Integrating Schema Wrapper Libraries

To use the Altova libraries in your custom project, refer to the libraries from your project (or include them into your project), as shown below for each language.

C#

To integrate the Altova libraries into an existing C# project:

1. After MapForce generates code from a schema (for example, **YourSchema.xsd**), build the generated **YourSchema.sln** solution in Visual Studio. This solution is in a project folder with the same name as the schema.
2. Right-click your existing project in Visual Studio, and select **Add Reference**.
3. On the Browse tab, browse for the following libraries: **Altova.dll**, **AltovaXML.dll**, and **YourSchema.dll** located in the output directory of the generated projects (for example, **bin\Debug**).



C++

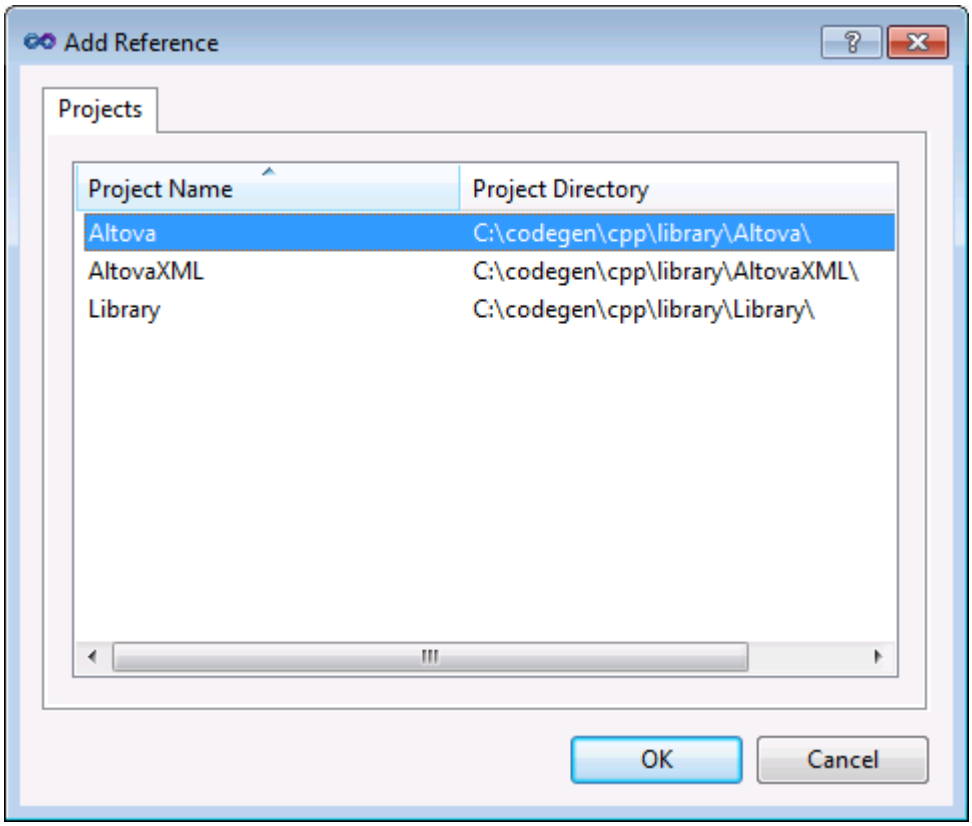
The easiest way to integrate the libraries into an existing C++ project is to add the generated project files to your solution. For example, let's assume that you generated code from a schema called **Library.xsd** and selected **c:\codegen\cpp\library** as target directory. The generated libraries in this case are available at:

- c:\codegen\cpp\library\Altova.vcxproj
- c:\codegen\cpp\library\AltovaXML\AltovaXML.vcxproj
- c:\codegen\cpp\library\Library.vcxproj

First, open the generated **c:\codegen\cpp\library\Library.sln** solution and build it in Visual Studio.

Next, open your existing Visual Studio solution (in Visual Studio 2010, in this example), right-click it, select **Add | Existing Project**, and add the project files listed above, one by one. Be patient while Visual Studio parses the files. Next, right-click your project and select **Properties**. In the Property Pages dialog box, select **Common Properties | Framework and References**, and then click **Add New Reference**. Next, select and add each of the following projects: *Altova*, *AltovaXML*, and *Library*.

(C) 2015-2021 Altova GmbH



See also the MSDN documentation for using functionality from a custom library, as applicable to your version of Visual Studio, for example:

- If you chose to generate static libraries, see https://msdn.microsoft.com/en-us/library/ms235627(v=vs.100).aspx
- If you chose to generate dynamic libraries, see https://msdn.microsoft.com/en-us/library/ms235636(v=vs.100).aspx

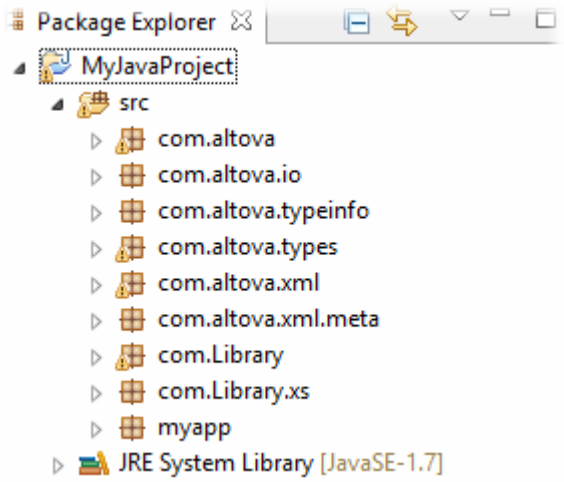
The option to generate static or dynamic libraries is available in code generation options (see Code generator options).

Java

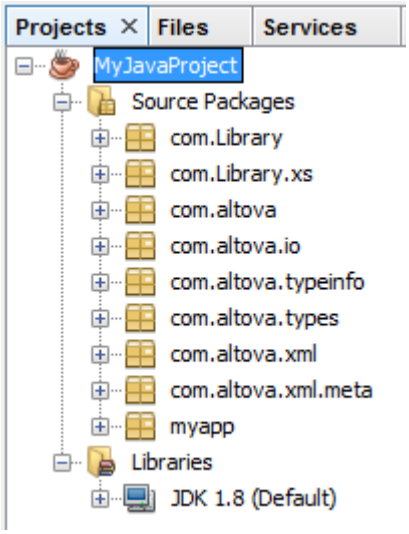
One of the ways to integrate the Altova packages into your Java project is to copy the **com** directory of the generated code to the directory which stores the source packages of your Java project (for example, **C:\Workspace\MyJavaProject\src**). For example, let's assume that you generated code in **c:\codegen\java\library**. The generated Altova classes in this case are available at **c:\codegen\java\library\com**.

After copying the libraries, refresh the project. To refresh the project in Eclipse, select it in the Package Explorer, and press **F5**. To refresh the project in NetBeans IDE 8.0, select the menu command **Source | Scan for External Changes**.

Once you perform the copy operation, the Altova packages are available in the Package Explorer (in case of Eclipse), or under "Source Packages" in the Projects pane (in case of NetBeans IDE).



Altova packages in Eclipse 4.4



Altova packages in NetBeans IDE 8.0.2

16.6.5 Example: Book Library

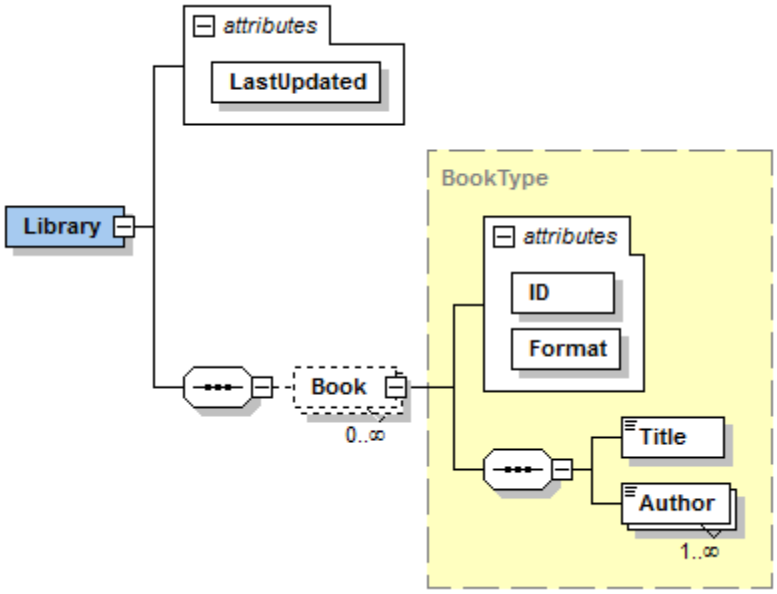
This example illustrates how to use the generated schema wrapper libraries in order to write or read programmatically XML documents conformant to the schema. Before using the sample code, take some time to understand the structure of the schema below.

The schema used in this example describes a library of books. The complete definition of the schema is shown below. Save this code listing as **Doc.xsd** if you want to get the same results as this example. You will need this schema to generate the code libraries used in this example.

(C) 2015-2021 Altova GmbH

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns="http://www.nanonull.com/LibrarySample"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.nanonull.com/LibrarySample" elementFormDefault="qualified"
attributeFormDefault="unqualified">
   <xs:element name="Library">
      <xs:complexType>
         <xs:sequence>
            <xs:element name="Book" type="BookType" minOccurs="0"
maxOccurs="unbounded"/>
         </xs:sequence>
         <xs:attribute name="LastUpdated" type="xs:dateTime"/>
      </xs:complexType>
   </xs:element>
   <xs:complexType name="BookType">
      <xs:sequence>
         <xs:element name="Title" type="xs:string"/>
         <xs:element name="Author" type="xs:string" maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:attribute name="ID" type="xs:integer" use="required"/>
      <xs:attribute name="Format" type="BookFormatType" use="required"/>
   </xs:complexType>
   <xs:complexType name="DictionaryType">
      <xs:complexContent>
         <xs:extension base="BookType">
            <xs:sequence>
               <xs:element name="FromLang" type="xs:string"/>
               <xs:element name="ToLang" type="xs:string"/>
            </xs:sequence>
         </xs:extension>
      </xs:complexContent>
   </xs:complexType>
   <xs:simpleType name="BookFormatType">
      <xs:restriction base="xs:string">
         <xs:enumeration value="Hardcover"/>
         <xs:enumeration value="Paperback"/>
         <xs:enumeration value="Audiobook"/>
         <xs:enumeration value="E-book"/>
      </xs:restriction>
   </xs:simpleType>
</xs:schema>
```

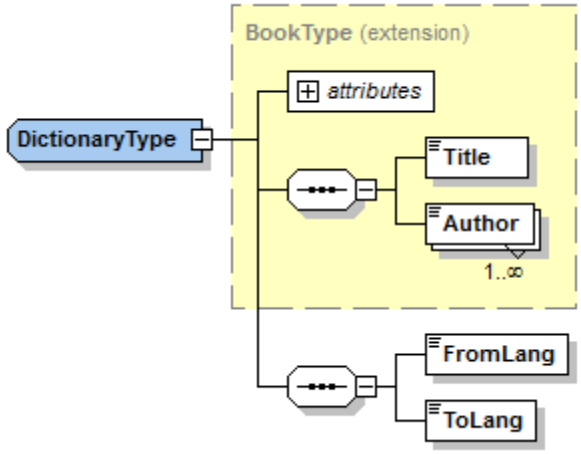
Library is a root element of a `complexType` which can be graphically represented as follows in the schema view of XMLSpy:



As shown above, the library has a **LastUpdated** attribute (defined as `xs:dateTime`), and stores a sequence of books. Each book is an `xs:complexType` and has two attributes: an **ID** (defined as `xs:integer`), and a **Format**. The format of any book can be hardcover, paperback, audiobook, or e-book. In the schema, **Format** is defined as `xs:simpleType` which uses an enumeration of the above-mentioned values.

Each book also has a **Title** element (defined as `xs:string`), as well as one or several **Author** elements (defined as `xs:string`).

The library may also contain books that are dictionaries. Dictionaries have the type `DictionaryType`, which is derived by extension from the `BookType`. In other words, a dictionary inherits all attributes and elements of a Book, plus two additional elements: **FromLang** and **ToLang**, as illustrated below.



The **FromLang** and **ToLang** elements store the source and destination language of the dictionary.

(C) 2015-2021 Altova GmbH

An XML instance file valid according to the schema above could therefore look as shown in the listing below (provided that it is in the same directory as the schema file):

```
<?xml version="1.0" encoding="utf-8"?>
<Library xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://www.nanonull.com/LibrarySample"
xsi:schemaLocation="http://www.nanonull.com/LibrarySample Library.xsd"
LastUpdated="2016-02-03T17:10:08.4977404">
   <Book ID="1" Format="E-book">
      <Title>The XMLSpy Handbook</Title>
      <Author>Altova</Author>
   </Book>
   <Book ID="2" Format="Paperback" xmlns:n1="http://www.nanonull.com/LibrarySample"
xsi:type="n1:DictionaryType">
      <Title>English-German Dictionary</Title>
      <Author>John Doe</Author>
      <FromLang>English</FromLang>
      <ToLang>German</ToLang>
   </Book>
</Library>
```

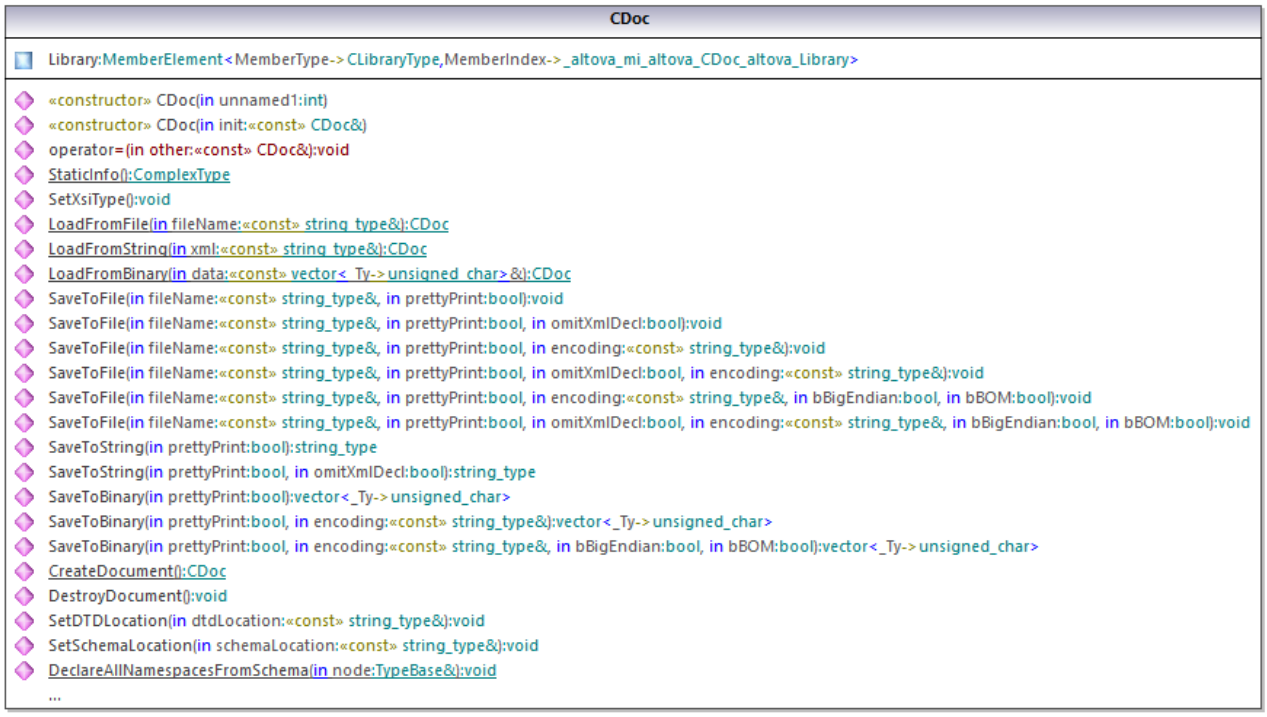
The next topics illustrate how to read from such a file programmatically, or write to such a file programmatically. To begin, generate the schema wrapper code from the schema above, using the steps described in Generating Code from XML Schemas or DTD.

16.6.5.1 Reading and Writing XML Documents (C++)

After you generate code from the example schema, a test C++ application is created, along with several supporting Altova libraries.

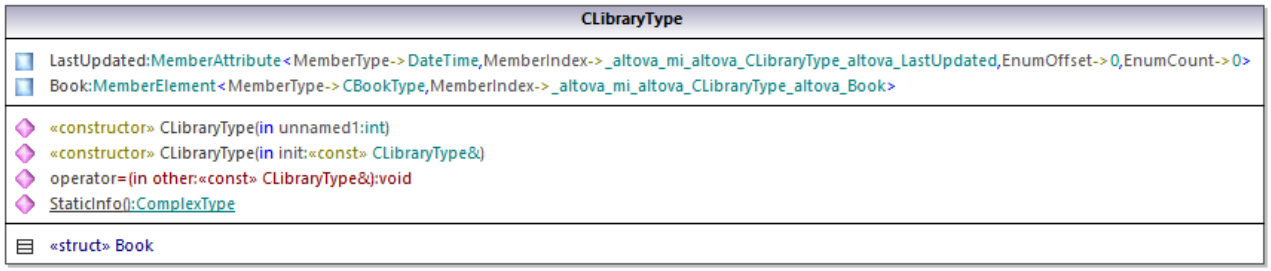
About the generated C++ libraries

The central class of the generated code is the `CDoc` class, which represents the XML document. Such a class is generated for every schema and its name depends on the schema file name (**Doc.xsd**, in this example). As shown in the diagram, this class provides methods for loading documents from files, binary streams, or strings (or saving documents to files, streams, strings). For a description of all members exposed by this class, see the class reference (`[YourSchema]::[CDoc]`).



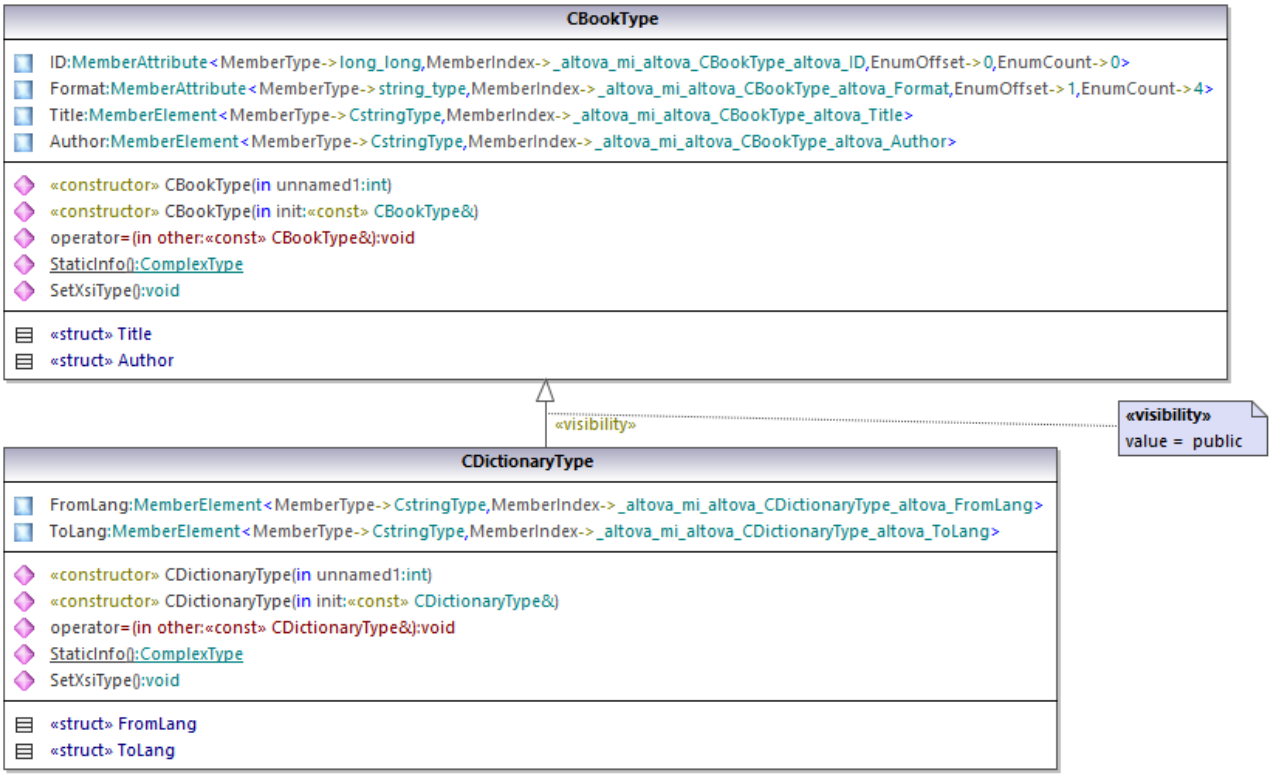
The `Library` field of the `CDoc` class represents the actual root of the document. **Library** is an element in the XML file, so in the C++ code it has a template class as type (`MemberElement`). The template class exposes methods and properties for interacting with the **Library** element. In general, each attribute and each element of a type in the schema is typed in the generated code with the `MemberAttribute` and `MemberElement` template classes, respectively. For more information, see `[YourSchema]::MemberAttribute` and `[YourSchema]::MemberElement` class reference.

The class `CLibraryType` is generated from the **LibraryType** complex type in the schema. Notice that the `CLibraryType` class contains two fields: `Book` and `LastUpdated`. According to the logic already mentioned above, these correspond to the **Book** element and **LastUpdated** attribute in the schema, and enable you to manipulate programmatically (append, remove, etc) elements and attributes in the instance XML document.



The `DictionaryType` is a complex type derived from **BookType** in the schema, so this relationship is also reflected in the generated classes. As illustrated in the diagram, the class `CDictionaryType` inherits the `CBookType` class.

(C) 2015-2021 Altova GmbH



If your XML schema defines simple types as enumerations, the enumerated values become available as `enum` values in the generated code. In the schema used in this example, a book format can be hardcover, paperback, e-book, and so on. Therefore, in the generated code, these values would be available through an `enum` that is a member of the `CBookFormatType` class.

Writing an XML document

1. Open the **LibraryTest.sln** solution in Visual Studio generated from the Library schema mentioned earlier in this example.

> While prototyping an application from a frequently changing XML schema, you may need to frequently generate code to the same directory, so that the schema changes are immediately reflected in the code. Note that the generated test application and the Altova libraries are overwritten every time when you generate code into the same target directory. Therefore, do not add code to the generated test application. Instead, integrate the Altova libraries into your project (see Integrating Schema Wrapper Libraries).

2. In Solution Explorer, open the **LibraryTest.cpp** file, and edit the `Example()` method as shown below.

```
#include <ctime> // required to get current time
using namespace Doc; // required to work with Altova libraries

void Example()
{
```

```
   // Create a new, empty XML document
   CDoc libDoc = CDoc::CreateDocument();

   // Create the root element <Library> and add it to the document
   CLibraryType lib = libDoc.Library.append();

   // Get current time and set the "LastUpdated" attribute using Altova classes
   time_t t = time(NULL);
   struct tm * now = localtime( & t );
   altova::DateTime dt = altova::DateTime(now->tm_year + 1900, now->tm_mon + 1, now-
>tm_mday, now->tm_hour, now->tm_min, now->tm_sec);
   lib.LastUpdated = dt;

   // Create a new <Book> and add it to the library
   CBookType book = lib.Book.append();

   // Set the "ID" attribute of the book
   book.ID = 1;

   // Set the "Format" attribute of the <Book> using an enumeration constant
   book.Format.SetEnumerationValue( CBookFormatType::k_Paperback );

   // Add the <Title> and <Author> elements, and set values
   book.Title.append() = _T("The XML Spy Handbook");
   book.Author.append() = _T("Altova");

   // Append a dictionary (book of derived type) and populate its attributes and
elements
   CDictionaryType dictionary = CDictionaryType(lib.Book.append().GetNode());
   dictionary.ID = 2;
   dictionary.Format.SetEnumerationValue( CBookFormatType::k_E_book);
   dictionary.Title.append() = _T("English-German Dictionary");
   dictionary.Author.append() = _T("John Doe");
   dictionary.FromLang.append() = _T("English");
   dictionary.ToLang.append() = _T("German");

   // Since dictionary a derived type, set the xsi:type attribute of the book element
   dictionary.SetXsiType();

   // Optionally, set the schema location
   libDoc.SetSchemaLocation(_T("Library.xsd"));

   // Save the XML document to a file with default encoding (UTF-8),
   // "true" causes the file to be pretty-printed.
   libDoc.SaveToFile(_T("GeneratedLibrary.xml"), true);

   // Destroy the document
   libDoc.DestroyDocument();
}
```

3. Press **F5** to start debugging. If the code was executed successfully, a **GeneratedLibrary.xml** file is created in the solution output directory.

Reading an XML document

1. Open the **LibraryTest.sln** solution in Visual Studio.
2. Save the code below as **Library1.xml** to a directory that can be read by the program code (for example, the same directory as **LibraryTest.sln**).

(C) 2015-2021 Altova GmbH

```
<?xml version="1.0" encoding="utf-8"?>
<Library xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://www.nanonull.com/LibrarySample"
xsi:schemaLocation="http://www.nanonull.com/LibrarySample Library.xsd"
LastUpdated="2016-02-03T17:10:08.4977404">
   <Book ID="1" Format="E-book">
      <Title>The XMLSpy Handbook</Title>
      <Author>Altova</Author>
   </Book>
   <Book ID="2" Format="Paperback" xmlns:n1="http://www.nanonull.com/LibrarySample"
xsi:type="n1:DictionaryType">
      <Title>English-German Dictionary</Title>
      <Author>John Doe</Author>
      <FromLang>English</FromLang>
      <ToLang>German</ToLang>
   </Book>
</Library>
```

3. In Solution Explorer, open the **LibraryTest.cpp** file, and edit the `Example()` method as shown below.

```
using namespace Doc;
void Example()
{
   // Load XML document
   CDoc libDoc = CDoc::LoadFromFile(_T("Library1.xml"));

   // Get the first (and only) root element <Library>
   CLibraryType lib = libDoc.Library.first();

   // Check whether an element exists:
   if (!lib.Book.exists())
   {
      tcout << "This library is empty." << std::endl;
      return;
   }

   // iteration: for each <Book>...
   for (Iterator<CBookType> itBook = lib.Book.all(); itBook; ++itBook)
   {
      // output values of ISBN attribute and (first and only) title element
      tcout << "ID: " << itBook->ID << std::endl;
      tcout << "Title: " << tstring(itBook->Title.first()) << std::endl;

      // read and compare an enumeration value
      if (itBook->Format.GetEnumerationValue() == CBookFormatType::k_Paperback)
         tcout << "This is a paperback book." << std::endl;

      // for each <Author>...
      for (CBookType::Author::iterator itAuthor = itBook->Author.all(); itAuthor; +
+itAuthor)
         tcout << "Author: " << tstring(itAuthor) << std::endl;

      // alternative: use count and index
      for (unsigned int j = 0; j < itBook->Author.count(); ++j)
         tcout << "Author: " << tstring(itBook->Author[j]) << std::endl;
   }

   // Destroy the document
```

```
    libDoc.DestroyDocument();
}
```

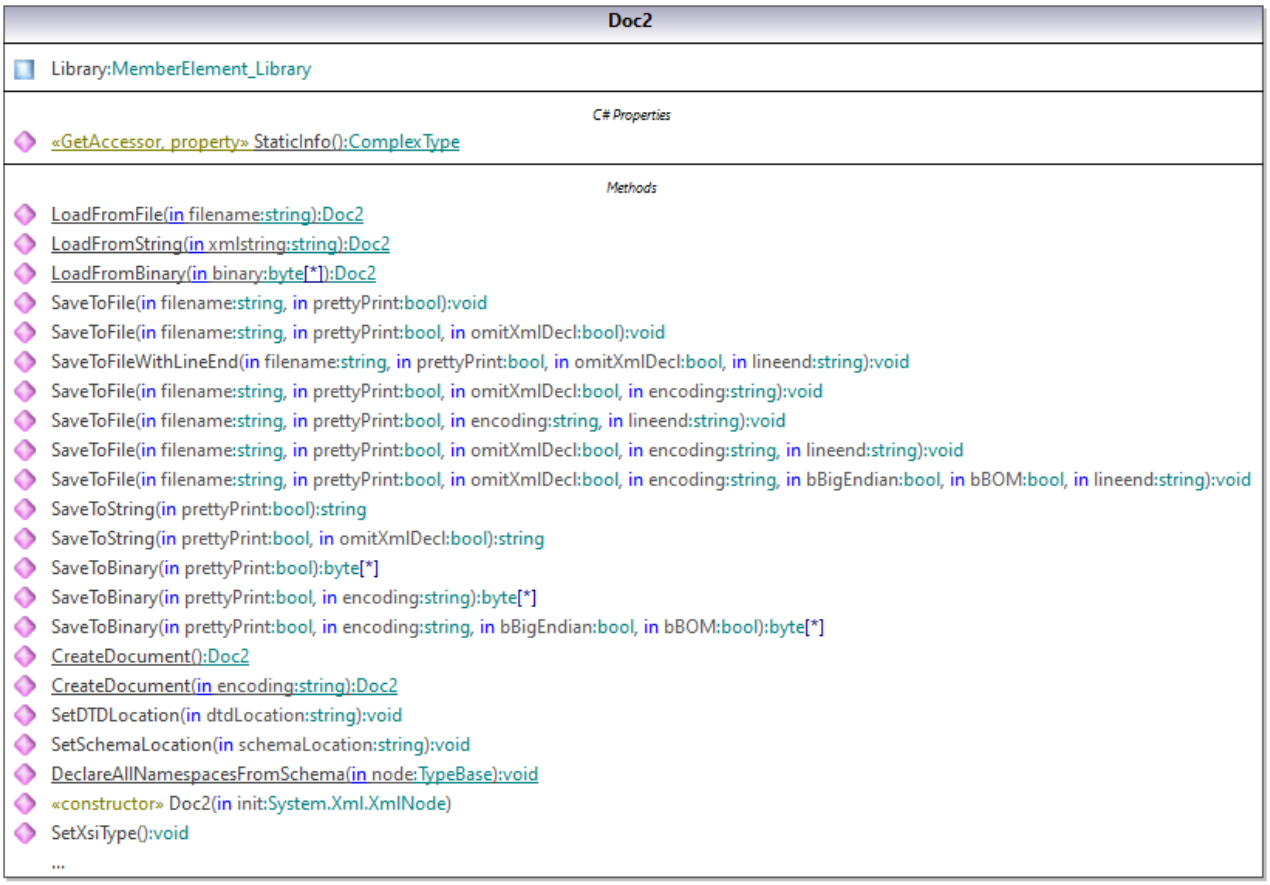
4. Press **F5** to start debugging.

16.6.5.2 Reading and Writing XML Documents (C#)

After you generate code from the example schema, a test C# application is created, along with several supporting Altova libraries.

About the generated C# libraries

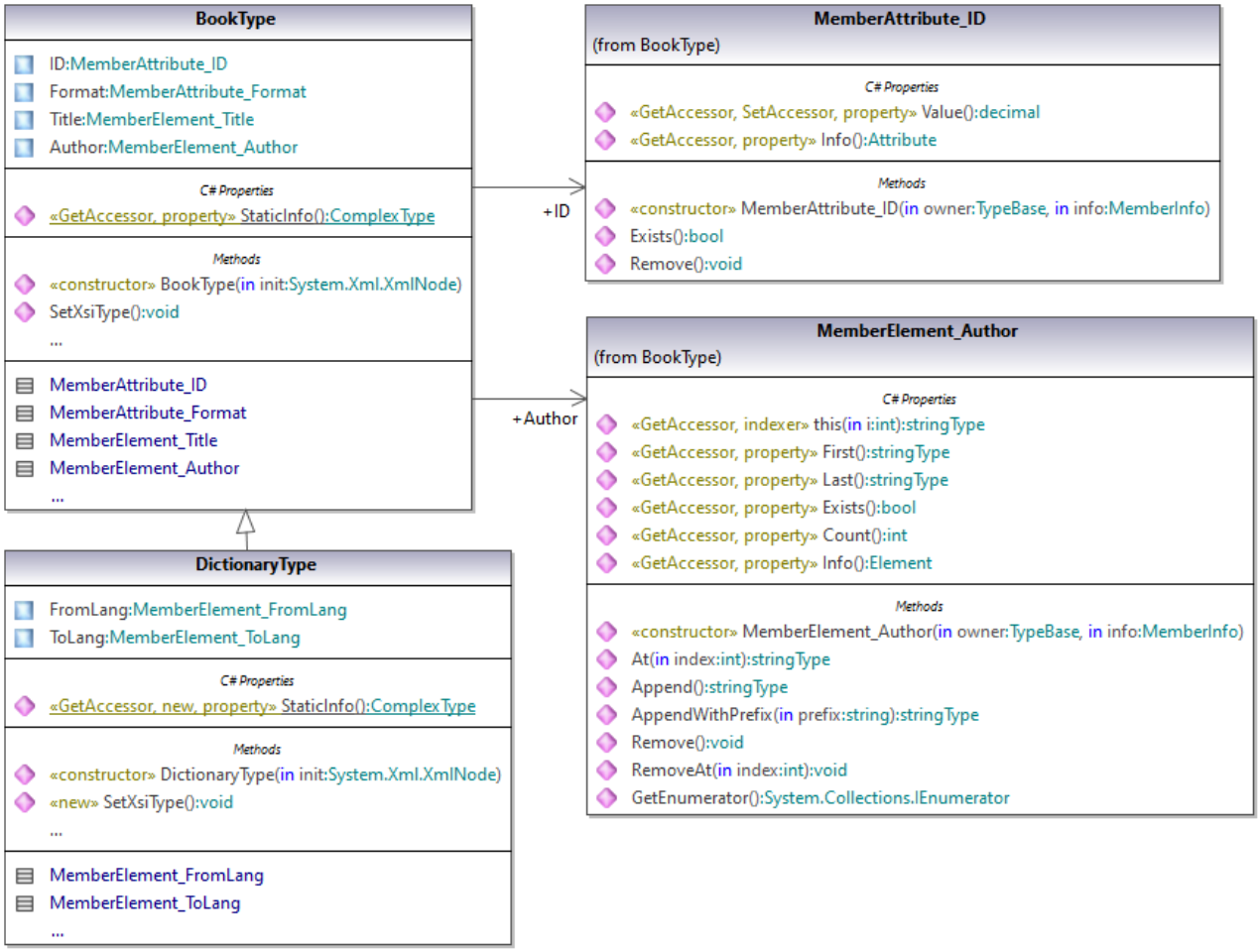
The central class of the generated code is the `Doc2` class, which represents the XML document. Such a class is generated for every schema and its name depends on the schema file name (**Doc.xsd**, in this example). Note that this class is called `Doc2` to avoid a possible conflict with the namespace name. As shown in the diagram, this class provides methods for loading documents from files, binary streams, or strings (or saving documents to files, streams, strings). For a description of this class, see the class reference ([YourSchema].[Doc]).



The `Library` member of the `Doc2` class represents the actual root of the document.

According to the code generation rules mentioned in About Schema Wrapper Libraries (C#), member classes are generated for each attribute and for each element of a type. In the generated code, the name of such member classes is prefixed with `MemberAttribute_` and `MemberElement_`, respectively. Examples of such classes are `MemberAttribute_ID` and `MemberElement_Author`, generated from the **Author** element and **ID** attribute of a book, respectively (in the diagram below, they are classes nested under `BookType`). Such classes enable you to manipulate programmatically the corresponding elements and attributes in the instance XML document (for example, append, remove, set value, etc). For more information, see the `[YourSchemaType].MemberAttribute` and `[YourSchemaType].MemberElement` class reference.

Since the **DictionaryType** is a complex type derived from **BookType** in the schema, this relationship is also reflected in the generated classes. As illustrated in the diagram below, the class `DictionaryType` inherits the `BookType` class.



If your XML schema defines simple types as enumerations, the enumerated values become available as `Enum` values in the generated code. In the schema used in this example, a book format can be hardcover, paperback, e-book, and so on. Therefore, in the generated code, these values would be available through an `Enum` that is a member of the `BookFormatType` class.

Writing an XML document

1. Open the **LibraryTest.sln** solution in Visual Studio generated from the Library schema mentioned earlier in this example.

While prototyping an application from a frequently changing XML schema, you may need to frequently generate code to the same directory, so that the schema changes are immediately reflected in the code. Note that the generated test application and the Altova libraries are overwritten every time when you generate code into the same target directory. Therefore, do not add code to the generated test application. Instead, integrate the Altova libraries into your project (see Integrating Schema Wrapper Libraries).

2. In Solution Explorer, open the **LibraryTest.cs** file, and edit the `Example()` method as shown below.

```
protected static void Example()
{
    // Create a new XML document
    Doc2 doc = Doc2.CreateDocument();
    // Append the root element
    LibraryType root = doc.Library.Append();

    // Create the generation date using Altova DateTime class
    Altova.Types.DateTime dt = new Altova.Types.DateTime(System.DateTime.Now);
    // Append the date to the root
    root.LastUpdated.Value = dt;

    // Add a new book
    BookType book = root.Book.Append();
    // Set the value of the ID attribute
    book.ID.Value = 1;
    // Set the format of the book (enumeration)
    book.Format.EnumerationValue = BookFormatType.EnumValues.eHardcover;
    // Set the Title and Author elements
    book.Title.Append().Value = "The XMLSpy Handbook";
    book.Author.Append().Value = "Altova";

    // Append a dictionary (book of derived type) and populate its attributes
and elements
    DictionaryType dictionary = new DictionaryType(root.Book.Append().Node);
    dictionary.ID.Value = 2;
    dictionary.Title.Append().Value = "English-German Dictionary";
    dictionary.Format.EnumerationValue = BookFormatType.EnumValues.eE_book;
    dictionary.Author.Append().Value = "John Doe";
    dictionary.FromLang.Append().Value = "English";
    dictionary.ToLang.Append().Value = "German";
    // Since it's a derived type, make sure to set the xsi:type attribute of the
book element
    dictionary.SetXsiType();

    // Optionally, set the schema location (adjust the path if
    // your schema is not in the same folder as the generated instance file)
    doc.SetSchemaLocation("Library.xsd");

    // Save the XML document with the "pretty print" option enabled
    doc.SaveToFile("GeneratedLibrary.xml", true);
}
```

3. Press **F5** to start debugging. If the code was executed successfully, a **GeneratedLibrary.xml** file is created in the solution output directory (typically, **bin/Debug**).

(C) 2015-2021 Altova GmbH

Reading an XML document

1. Open the **LibraryTest.sln** solution in Visual Studio.
2. Save the code below as **Library.xml** to the output directory of the project (by default, **bin/Debug**). This is the file that will be read by the program code.

```
<?xml version="1.0" encoding="utf-8"?>
<Library xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://www.nanonull.com/LibrarySample"
xsi:schemaLocation="http://www.nanonull.com/LibrarySample Library.xsd"
LastUpdated="2016-02-03T17:10:08.4977404">
   <Book ID="1" Format="E-book">
      <Title>The XMLSpy Handbook</Title>
      <Author>Altova</Author>
   </Book>
   <Book ID="2" Format="Paperback" xmlns:n1="http://www.nanonull.com/LibrarySample"
xsi:type="n1:DictionaryType">
      <Title>English-German Dictionary</Title>
      <Author>John Doe</Author>
      <FromLang>English</FromLang>
      <ToLang>German</ToLang>
   </Book>
</Library>
```

3. In Solution Explorer, open the **LibraryTest.cs** file, and edit the `Example()` method as shown below.

```
protected static void Example()
{
    // Load the XML file
    Doc2 doc = Doc2.LoadFromFile("Library.xml");
    // Get the root element
    LibraryType root = doc.Library.First;

    // Read the library generation date
    Altova.Types.DateTime dt = root.LastUpdated.Value;
    string dt_as_string = dt.ToString(DateTimeFormat.W3_dateTime);
    Console.WriteLine("The library generation date is: " + dt_as_string);

    // Iteration: for each <Book>...
    foreach (BookType book in root.Book)
    {
      // Output values of ID attribute and (first and only) title element
      Console.WriteLine("ID:    " + book.ID.Value);
      Console.WriteLine("Title: " + book.Title.First.Value);

      // Read and compare an enumeration value
      if (book.Format.EnumerationValue == BookFormatType.EnumValues.ePaperback)
         Console.WriteLine("This is a paperback book.");

      // Iteration: for each <Author>
      foreach (xs.stringType author in book.Author)
         Console.WriteLine("Author: " + author.Value);

      // Determine if this book is of derived type
      if (book.Node.Attributes.GetNamedItem("xsi:type") != null)
      {
```

```
         // Find the value of the xsi:type attribute
         string xsiTypeValue =
book.Node.Attributes.GetNamedItem("xsi:type").Value;
         // Get the namespace URI and the lookup prefix of this namespace
         string namespaceUri = book.Node.NamespaceURI;
         string prefix = book.Node.GetPrefixOfNamespace(namespaceUri);

         // if this book has DictionaryType
         if (namespaceUri == "http://www.nanonull.com/LibrarySample" &&
xsiTypeValue.Equals(prefix + ":DictionaryType"))
         {
            // output additional fields
            DictionaryType dictionary = new DictionaryType(book.Node);
            Console.WriteLine("Language from: " +
dictionary.FromLang.First.Value);
            Console.WriteLine("Language to: " + dictionary.ToLang.First.Value);
         }
         else
         {
            throw new Exception("Unexpected book type");
         }
      }
   }

   Console.ReadLine();
}
```

4. Press **F5** to start debugging. If the code was executed successfully, **Library.xml** will be read by the program code, and its contents displayed as console output.

Reading and writing elements and attributes

Values of attributes and elements can be accessed using the `Value` property of the generated member element or attribute class, for example:

```
// Output values of ID attribute and (first and only) title element
Console.WriteLine("ID:    " + book.ID.Value);
Console.WriteLine("Title: " + book.Title.First.Value);
```

To get the value of the **Title** element in this particular example, we also used the `First()` method, since this is the first (and only) **Title** element of a book. For cases when you need to pick a specific element from a list by index, use the `At()` method.

The class generated for each member element of a type implements the standard `System.Collections.IEnumerable` interface. This makes it possible to loop through multiple elements of the same type. In this particular example, you can loop through all books of a Library object as follows:

```
// Iteration: for each <Book>...
foreach (BookType book in root.Book)
{
   // your code here...
}
```

To add a new element, use the `Append()` method. For example, the following code appends the root element to the document:

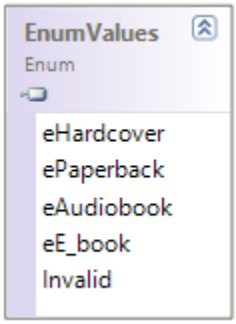
```
// Append the root element to the library
LibraryType root = doc.Library.Append();
```

You can set the value of an attribute (like ID in this example) as follows:

```
// Set the value of the ID attribute
book.ID.Value = 1;
```

Reading and writing enumeration values

If your XML schema defines simple types as enumerations, the enumerated values become available as `Enum` values in the generated code. In the schema used in this example, a book format can be hardcover, paperback, e-book, and so on. Therefore, in the generated code, these values would be available through an `Enum`:



To assign enumeration values to an object, use code such as the one below:

```
// Set the format of the book (enumeration)
book.Format.EnumerationValue = BookFormatType.EnumValues.eHardcover;
```

You can read such enumeration values from XML instance documents as follows:

```
// Read and compare an enumeration value
if (book.Format.EnumerationValue == BookFormatType.EnumValues.ePaperback)
Console.WriteLine("This is a paperback book.");
```

When an "if" condition is not enough, create a switch to determine each enumeration value and process it as required.

Working with xs:dateTime and xs:duration types

If the schema from which you generated code uses time and duration types such as `xs:dateTime`, or `xs:duration`, these are converted to Altova native classes in generated code. Therefore, to write a date or duration value to the XML document, do the following:

1. Construct an `Altova.Types.DateTime` or `Altova.Types.Duration` object (either from `System.DateTime`, or by using parts such as hours and minutes, see `Altova.Types.DateTime` and `Altova.Types.Duration` for more information).
2. Set the object as value of the required element or attribute, for example:

```
// Create the library generation date using Altova DateTime class
Altova.Types.DateTime dt = new Altova.Types.DateTime(System.DateTime.Now);
// Append the date to the root
root.LastUpdated.Value = dt;
```

To read a date or duration from an XML document, do the following:

1. Declare the element value (or attribute) as `Altova.Types.DateTime` or `Altova.Types.Duration` object.
2. Format the required element or attribute, for example:

```
// Read the library generation date
Altova.Types.DateTime dt = root.LastUpdated.Value;
string dt_as_string = dt.ToString(DateTimeFormat.W3_dateTime);
Console.WriteLine("The library generation date is: " + dt_as_string);
```

For more information, see `Altova.Types.DateTime` and `Altova.Types.Duration` class reference.

Working with derived types

If your XML schema defines derived types, you can preserve type derivation in XML documents that you create or load programmatically. Taking the schema used in this example, the following code listing illustrates how to create a new book of derived type `DictionaryType`:

```
// Append a dictionary (book of derived type) and populate its attributes and elements
DictionaryType dictionary = new DictionaryType(root.Book.Append().Node);
dictionary.ID.Value = 2;
dictionary.Title.Append().Value = "English-German Dictionary";
dictionary.Author.Append().Value = "John Doe";
dictionary.FromLanguage.Append().Value = "English";
dictionary.ToLanguage.Append().Value = "German";

// Since it's a derived type, make sure to set the xsi:type attribute of the book
element
dictionary.SetXsiType();
```

Note that it is important to set the `xsi:type` attribute of the newly created book. This ensures that the book type will be interpreted correctly by the schema when the XML document is validated.

When you load data from an XML document, the following code listing shows how to identify a book of derived type `DictionaryType` in the loaded XML instance. First, the code finds the value of the `xsi:type` attribute of the book node. If the namespace URI of this node is `http://www.nanonull.com/LibrarySample`, and if the URI lookup prefix and type matches the value of the `xsi:type` attribute, then this is a dictionary:

```
    // Determine if this book is of derived type
    if (book.Node.Attributes.GetNamedItem("xsi:type") != null)
    {
        // Find the value of the xsi:type attribute
        string xsiTypeValue = book.Node.Attributes.GetNamedItem("xsi:type").Value;
        // Get the namespace URI and the lookup prefix of this namespace
        string namespaceUri = book.Node.NamespaceURI;
        string prefix = book.Node.GetPrefixOfNamespace(namespaceUri);

        // if this book has DictionaryType
```


(C) 2015-2021 Altova GmbH

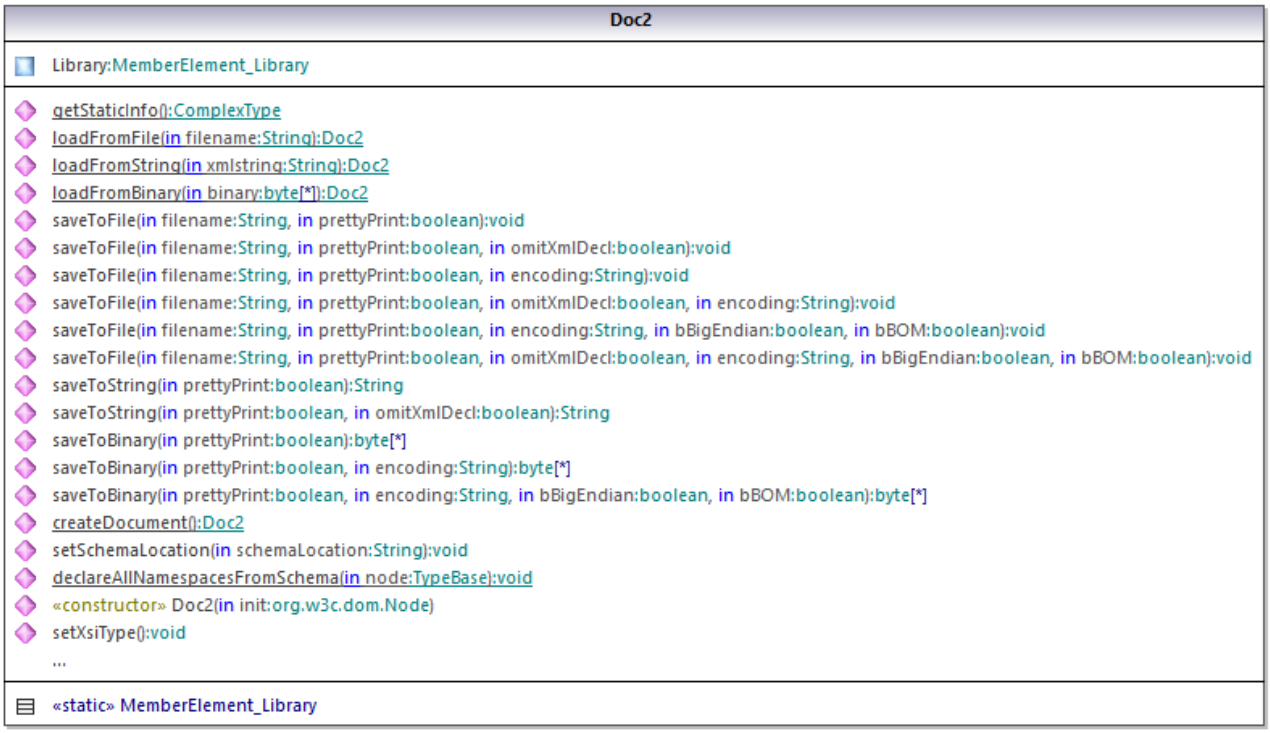
```
          if (namespaceUri == "http://www.nanonull.com/LibrarySample" &&
xsiTypeValue.Equals(prefix + ":DictionaryType"))
          {
             // output additional fields
             DictionaryType dictionary = new DictionaryType(book.Node);
             Console.WriteLine("Language from: " + dictionary.FromLang.First.Value);
             Console.WriteLine("Language to: " + dictionary.ToLang.First.Value);
          }
          else
          {
              throw new Exception("Unexpected book type");
          }
      }
```

16.6.5.3 Reading and Writing XML Documents (Java)

After you generate code from the example schema, a test Java project is created, along with several supporting Altova libraries.

About the generated Java libraries

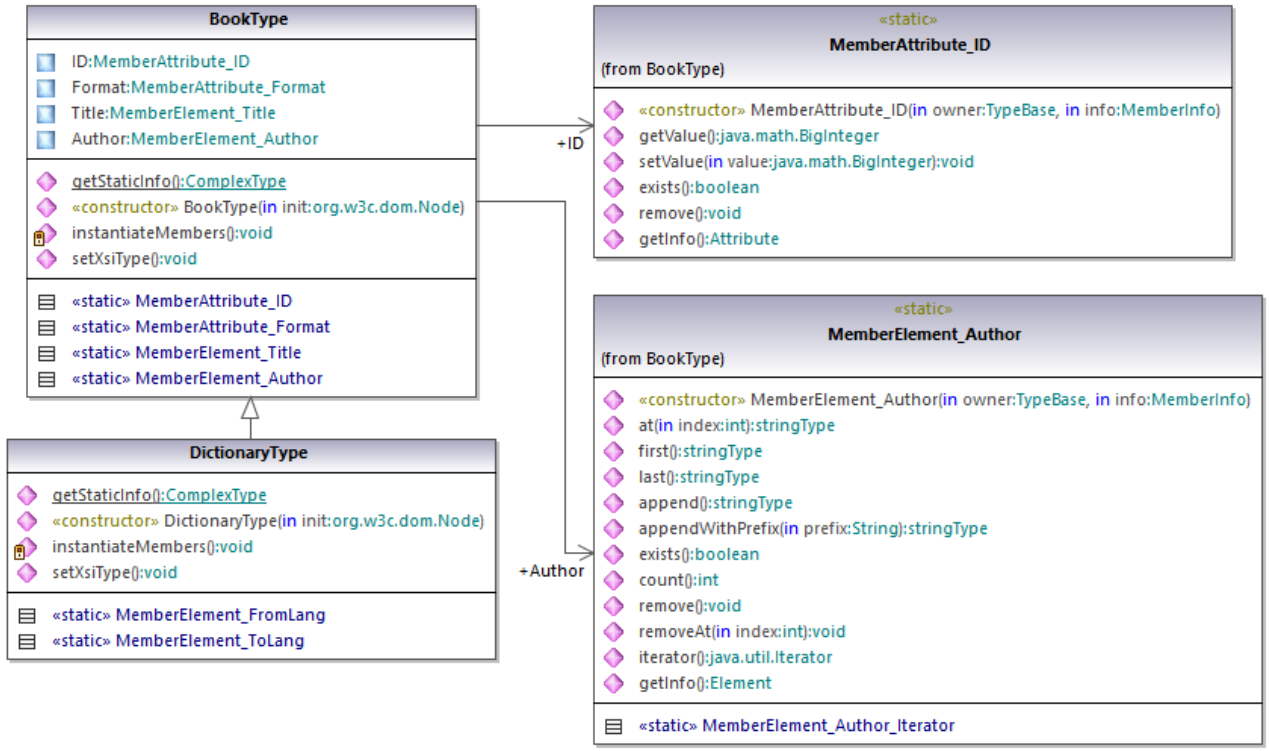
The central class of the generated code is the `Doc2` class, which represents the XML document. Such a class is generated for every schema and its name depends on the schema file name (**Doc.xsd**, in this example). Note that this class is called `Doc2` to avoid a possible conflict with the namespace name. As shown in the diagram, this class provides methods for loading documents from files, binary streams, or strings (or saving documents to files, streams, strings). For a description of this class, see the `com.[YourSchema].[Doc]` class reference.



The `Library` member of the `Doc2` class represents the actual root of the document.

According to the code generation rules mentioned in About Generated Java Code, member classes are generated for each attribute and for each element of a type. In the generated code, the name of such member classes is prefixed with `MemberAttribute_` and `MemberElement_`, respectively. In the diagram below, examples of such classes are `MemberAttribute_ID` and `MemberElement_Author`, generated from the **Author** element and **ID** attribute of a book, respectively. Such classes enable you to manipulate programmatically the corresponding elements and attributes in the instance XML document (for example, append, remove, set value, etc). For more information, see the `com.[YourSchema].[YourSchemaType].MemberAttribute` and `com.[YourSchema].[YourSchemaType].MemberElement` class reference.

Since the **DictionaryType** is a complex type derived from **BookType** in the schema, this relationship is also reflected in the generated classes. As illustrated in the diagram below, the class `DictionaryType` inherits the `BookType` class.



If your XML schema defines simple types as enumerations, the enumerated values become available as `Enum` values in the generated code. In the schema used in this example, a book format can be hardcover, paperback, e-book, and so on. Therefore, in the generated code, these values would be available through an `Enum` that is a member of the `BookFormatType` class.

Writing an XML document

1. On the **File** menu of Eclipse, click **Import**, select **Existing Projects into Workspace**, and click **Next**.
2. Next to **Select root directory**, click **Browse**, select the directory to which you generated the Java code, and then click **Finish**.
3. In the Eclipse Package Explorer, expand the **com.LibraryTest** package and open the **LibraryTest.java** file.

(C) 2015-2021 Altova GmbH

While prototyping an application from a frequently changing XML schema, you may need to frequently generate code to the same directory, so that the schema changes are immediately reflected in the code. Note that the generated test application and the Altova libraries are overwritten every time when you generate code into the same target directory. Therefore, do not add code to the generated test application. Instead, integrate the Altova libraries into your project (see Integrating Schema Wrapper Libraries).

4. Edit the `Example()` method as shown below.

```
protected static void example() throws Exception {
      // create a new, empty XML document
      Doc2 libDoc = Doc2.createDocument();

      // create the root element <Library> and add it to the document
      LibraryType lib = libDoc.Library.append();

      // set the "LastUpdated" attribute
      com.altova.types.DateTime dt = new com.altova.types.DateTime(DateTime.now());
      lib.LastUpdated.setValue(dt);

      // create a new <Book> and populate its elements and attributes
      BookType book = lib.Book.append();
      book.ID.setValue(java.math.BigInteger.valueOf(1));
      book.Format.setEnumerationValue( BookFormatType.EPAPERBACK );
      book.Title.append().setValue("The XML Spy Handbook");
      book.Author.append().setValue("Altova");

      // create a dictionary (book of derived type)  and populate its elements and
attributes
      DictionaryType dict = new DictionaryType(lib.Book.append().getNode());
      dict.ID.setValue(java.math.BigInteger.valueOf(2));
      dict.Title.append().setValue("English-German Dictionary");
      dict.Format.setEnumerationValue(BookFormatType.EE_BOOK);
      dict.Author.append().setValue("John Doe");
      dict.FromLang.append().setValue("English");
      dict.ToLang.append().setValue("German");
      dict.setXsiType();

      // set the schema location (this is optional)
      libDoc.setSchemaLocation("Library.xsd");

      // save the XML document to a file with default encoding (UTF-8). "true" causes
the file to be pretty-printed.
      libDoc.saveToFile("Library1.xml", true);
   }
```

5. Build the Java project and run it. If the code is executed successfully, a **Library1.xml** file is created in the project directory.

Reading an XML document

1. On the **File** menu of Eclipse, click **Import**, select **Existing Projects into Workspace**, and click **Next**.
2. Next to **Select root directory**, click **Browse**, select the directory to which you generated the Java code, and then click **Finish**.
3. Save the code below as **Library1.xml** to a local directory (you will need to refer to the path of the **Library1.xml** file from the sample code below).

```
<?xml version="1.0" encoding="utf-8"?>
<Library xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://www.nanonull.com/LibrarySample"
xsi:schemaLocation="http://www.nanonull.com/LibrarySample Library.xsd"
LastUpdated="2016-02-03T17:10:08.4977404">
   <Book ID="1" Format="E-book">
      <Title>The XMLSpy Handbook</Title>
      <Author>Altova</Author>
   </Book>
   <Book ID="2" Format="Paperback" xmlns:n1="http://www.nanonull.com/LibrarySample"
xsi:type="n1:DictionaryType">
      <Title>English-German Dictionary</Title>
      <Author>John Doe</Author>
      <FromLang>English</FromLang>
      <ToLang>German</ToLang>
   </Book>
</Library>
```

4. In the Eclipse Package Explorer, expand the **com.LibraryTest** package and open the **LibraryTest.java** file.
5. Edit the `Example()` method as shown below.

```
protected static void example() throws Exception {
     // load XML document from a path, make sure to adjust the path as necessary
     Doc2 libDoc = Doc2.loadFromFile("Library1.xml");

     // get the first (and only) root element <Library>
     LibraryType lib = libDoc.Library.first();

     // check whether an element exists:
     if (!lib.Book.exists()) {
        System.out.println("This library is empty.");
        return;
     }

     // read a DateTime schema type
     com.altova.types.DateTime dt = lib.LastUpdated.getValue();
     System.out.println("The library was last updated on: " + dt.toDateString());

     // iteration: for each <Book>...
     for (java.util.Iterator itBook = lib.Book.iterator(); itBook.hasNext();) {
        BookType book = (BookType) itBook.next();
        // output values of ID attribute and (first and only) title element
        System.out.println("ID: " + book.ID.getValue());
        System.out.println("Title: " + book.Title.first().getValue());

        // read and compare an enumeration value
        if (book.Format.getEnumerationValue() == BookFormatType.EPAPERBACK)
           System.out.println("This is a paperback book.");

        // for each <Author>...
        for (java.util.Iterator itAuthor = book.Author.iterator(); itAuthor
              .hasNext();)
           System.out.println("Author: " + ((com.Doc.xs.stringType)
itAuthor.next()).getValue());
```


(C) 2015-2021 Altova GmbH

```
        // find the derived type of this book
        // by looking at the value of the xsi:type attribute, using DOM
        org.w3c.dom.Node bookNode = book.getNode();
        if (bookNode.getAttributes().getNamedItem("xsi:type") != null) {
           // Get the value of the xsi:type attribute
           String xsiTypeValue =
bookNode.getAttributes().getNamedItem("xsi:type").getNodeValue();

           // Get the namespace URI and lookup prefix of this namespace
           String namespaceUri = bookNode.getNamespaceURI();
           String lookupPrefix = bookNode.lookupPrefix(namespaceUri);

           // If xsi:type matches the namespace URI and type of the book node
           if (namespaceUri == "http://www.nanonull.com/LibrarySample"
                 && (  xsiTypeValue.equals(lookupPrefix + ":DictionaryType" )))    {
              // ...then this is a book of derived type (dictionary)
              DictionaryType dictionary = new DictionaryType(   book.getNode());
              // output the value of the "FromLang" and "ToLang" elements
              System.out.println("From language: " +
dictionary.FromLang.first().getValue());
              System.out.println("To language: " +
dictionary.ToLang.first().getValue());
           }
           else
           {
              // throw an error
              throw new java.lang.Error("This book has an unknown type.");
           }
        }
     }
   }
```

6. Build the Java project and run it. If the code is executed successfully, **Library1.xml** will be read by the program code, and its contents displayed in the Console view.

Reading and writing elements and attributes

Values of attributes and elements can be accessed using the `getValue()` method of the generated member element or attribute class, for example:

```
// output values of ID attribute and (first and only) title element
System.out.println("ID: " + book.ID.getValue());
System.out.println("Title: " + book.Title.first().getValue());
```

To get the value of the **Title** element in this particular example, we also used the `first()` method, since this is the first (and only) **Title** element of a book. For cases when you need to pick a specific element from a list by index, use the `at()` method.

To iterate through multiple elements, use either index-based iteration or `java.util.Iterator`. For example, you can iterate through the books of a library as follows:

```
// index-based iteration
for (int j = 0; j < lib.Book.count(); ++j ) {
   // your code here
}
```

```
// alternative iteration using java.util.Iterator
for (java.util.Iterator itBook = lib.Book.iterator(); itBook.hasNext();) {
      // your code here
   }
```

To add a new element, use the `append()` method. For example, the following code appends an empty root **Library** element to the document:

```
// create the root element <Library> and add it to the document
LibraryType lib = libDoc.Library.append();
```

Once an element is appended, you can set the value of any of its elements or an attributes by using the setValue() method.

```
// set the value of the Title element
book.Title.append().setValue("The XML Spy Handbook");
// set the value of the ID attribute
book.ID.setValue(java.math.BigInteger.valueOf(1));
```

Reading and writing enumeration values

If your XML schema defines simple types as enumerations, the enumerated values become available as `Enum` values in the generated code. In the schema used in this example, a book format can be hardcover, paperback, e-book, and so on. Therefore, in the generated code, these values would be available through an `Enum` (see the `BookFormatType` class diagram above). To assign enumeration values to an object, use code such as the one below:

```
// set an enumeration value
book.Format.setEnumerationValue( BookFormatType.EPAPERBACK );
```

You can read such enumeration values from XML instance documents as follows:

```
// read an enumeration value
if (book.Format.getEnumerationValue() == BookFormatType.EPAPERBACK)
         System.out.println("This is a paperback book."
```

When an "if" condition is not enough, create a switch to determine each enumeration value and process it as required.

Working with xs:dateTime and xs:duration types

If the schema from which you generated code uses time and duration types such as `xs:dateTime`, or `xs:duration`, these are converted to Altova native classes in generated code. Therefore, to write a date or duration value to the XML document, do the following:

1. Construct a `com.altova.types.DateTime` or `com.altova.types.Duration` object.
2. Set the object as value of the required element or attribute, for example:

```
// set the value of an attribute of DateTime type
com.altova.types.DateTime dt = new com.altova.types.DateTime(DateTime.now());
lib.LastUpdated.setValue(dt);
```


(C) 2015-2021 Altova GmbH

To read a date or duration from an XML document:

1. Declare the element value (or attribute) as com.altova.types.DateTime or com.altova.types.Duration object.
2. Format the required element or attribute, for example:

```
// read a DateTime type
com.altova.types.DateTime dt = lib.LastUpdated.getValue();
   System.out.println("The library was last updated on: " + dt.toDateString());
```

For more information, see com.altova.types.DateTime and com.altova.types.Duration class reference.

Working with derived types

If your XML schema defines derived types, you can preserve type derivation in XML documents that you create or load programmatically. Taking the schema used in this example, the following code listing illustrates how to create a new book of derived type `DictionaryType`:

```
// create a dictionary (book of derived type)  and populate its elements and attributes
DictionaryType dict = new DictionaryType(lib.Book.append().getNode());
dict.ID.setValue(java.math.BigInteger.valueOf(2));
dict.Title.append().setValue("English-German Dictionary");
dict.Format.setEnumerationValue(BookFormatType.EE_BOOK);
dict.Author.append().setValue("John Doe");
dict.FromLang.append().setValue("English");
dict.ToLang.append().setValue("German");
dict.setXsiType();
```

Note that it is important to set the `xsi:type` attribute of the newly created book. This ensures that the book type will be interpreted correctly by the schema when the XML document is validated.

When you load data from an XML document, the following code listing shows how to identify a book of derived type `DictionaryType` in the loaded XML instance. First, the code finds the value of the `xsi:type` attribute of the book node. If the namespace URI of this node is `http://www.nanonull.com/LibrarySample`, and if the URI lookup prefix and type matches the value of the `xsi:type` attribute, then this is a dictionary:

```
         // find the derived type of this book
         // by looking at the value of the xsi:type attribute, using DOM
         org.w3c.dom.Node bookNode = book.getNode();
         if (bookNode.getAttributes().getNamedItem("xsi:type") != null) {
            // Get the value of the xsi:type attribute
            String xsiTypeValue =
bookNode.getAttributes().getNamedItem("xsi:type").getNodeValue();

            // Get the namespace URI and lookup prefix of the book node
            String namespaceUri = bookNode.getNamespaceURI();
            String lookupPrefix = bookNode.lookupPrefix(namespaceUri);

            // If xsi:type matches the namespace URI and type of the book node
            if (namespaceUri == "http://www.nanonull.com/LibrarySample"
                  && (  xsiTypeValue.equals(lookupPrefix + ":DictionaryType" )))     {
               // ...then this is a book of derived type (dictionary)
               DictionaryType dictionary = new DictionaryType(   book.getNode());
```

```
            // output the value of the "FromLang" and "ToLang" elements
            System.out.println("From language: " +
dictionary.FromLang.first().getValue());
            System.out.println("To language: " +
dictionary.ToLang.first().getValue());
         }
         else
         {
            // throw an error
            throw new java.lang.Error("This book has an unknown type.");
         }
      }
```

16.6.6 Example: Purchase Order

This example illustrates how to work with program code generated from a "main" XML schema that imports other schemas. Each of the imported schema has a different target namespace. The goal here is to create programmatically an XML document where all elements are prefixed according to their namespace. More specifically, the XML document created from your C++, C#, or Java code should look like the one below:

```
<?xml version="1.0" encoding="utf-8"?>
<p:Purchase xsi:schemaLocation="http://NamespaceTest.com/Purchase Main.xsd"
            xmlns:p="http://NamespaceTest.com/Purchase"
            xmlns:o="http://NamespaceTest.com/OrderTypes"
            xmlns:c="http://NamespaceTest.com/CustomerTypes"
            xmlns:cmn="http://NamespaceTest.com/CommonTypes"
            xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <p:OrderDetail>
        <o:Item>
            <o:ProductName>Lawnmower</o:ProductName>
            <o:Quantity>1</o:Quantity>
            <o:UnitPrice>148.42</o:UnitPrice>
        </o:Item>
    </p:OrderDetail>
    <p:PaymentMethod>VISA</p:PaymentMethod>
    <p:CustomerDetails>
        <c:Name>Alice Smith</c:Name>
        <c:DeliveryAddress>
            <cmn:Line1>123 Maple Street</cmn:Line1>
            <cmn:Line2>Mill Valley</cmn:Line2>
        </c:DeliveryAddress>
        <c:BillingAddress>
            <cmn:Line1>8 Oak Avenue</cmn:Line1>
            <cmn:Line2>Old Town</cmn:Line2>
        </c:BillingAddress>
    </p:CustomerDetails>
</p:Purchase>
```

The main schema used in this example is called **Main.xsd**. As illustrated in the code listing below, it imports three other schemas: **CommonTypes.xsd**, **CustomerTypes.xsd**, and **OrderTypes.xsd**. To get the same results as in this example, save all the code listings below to files, and use the same file names as above. Notice that the schema maps each of the prefixes `ord`, `pur`, `cmn`, and `cust` to some namespace (Order types, Purchase types, Common types, and Customer types, respectively). This means that, in the generated code,

(C) 2015-2021 Altova GmbH

the classes corresponding to Orders, Purchases, Customers, and so on, will be available under their respective namespace.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
      targetNamespace="http://NamespaceTest.com/Purchase"
      xmlns:ord="http://NamespaceTest.com/OrderTypes"
      xmlns:pur="http://NamespaceTest.com/Purchase"
      xmlns:cmn="http://NamespaceTest.com/CommonTypes"
      xmlns:cust="http://NamespaceTest.com/CustomerTypes"
      elementFormDefault="qualified">
    <xs:import schemaLocation="CommonTypes.xsd"
namespace="http://NamespaceTest.com/CommonTypes" />
    <xs:import schemaLocation="CustomerTypes.xsd"
namespace="http://NamespaceTest.com/CustomerTypes" />
    <xs:import schemaLocation="OrderTypes.xsd"
namespace="http://NamespaceTest.com/OrderTypes" />
    <xs:element name="Purchase">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="OrderDetail" type="ord:OrderType" />
                <xs:element name="PaymentMethod" type="cmn:PaymentMethodType" />
                <xs:element ref="pur:CustomerDetails" />
            </xs:sequence>
        </xs:complexType>
    </xs:element>
    <xs:element name="CustomerDetails" type="cust:CustomerType" />
</xs:schema>
```

Main.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
      targetNamespace="http://NamespaceTest.com/CommonTypes"
      elementFormDefault="qualified">
   <xs:complexType name="AddressType">
      <xs:sequence>
         <xs:element name="Line1" type="xs:string"/>
         <xs:element name="Line2" type="xs:string"/>
      </xs:sequence>
   </xs:complexType>
   <xs:simpleType name="PriceType">
      <xs:restriction base="xs:decimal">
         <xs:fractionDigits value="2"/>
      </xs:restriction>
   </xs:simpleType>
   <xs:simpleType name="PaymentMethodType">
      <xs:restriction base="xs:string">
         <xs:enumeration value="VISA"/>
         <xs:enumeration value="MasterCard"/>
         <xs:enumeration value="Cash"/>
         <xs:enumeration value="AMEX"/>
      </xs:restriction>
   </xs:simpleType>
</xs:schema>
```

CommonTypes.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
      targetNamespace="http://NamespaceTest.com/CustomerTypes"
      xmlns:cmn="http://NamespaceTest.com/CommonTypes"
      elementFormDefault="qualified">
    <xs:import schemaLocation="CommonTypes.xsd"
namespace="http://NamespaceTest.com/CommonTypes" />
    <xs:complexType name="CustomerType">
        <xs:sequence>
            <xs:element name="Name" type="xs:string" />
            <xs:element name="DeliveryAddress" type="cmn:AddressType" />
            <xs:element name="BillingAddress" type="cmn:AddressType" />
        </xs:sequence>
    </xs:complexType>
</xs:schema>
```

CustomerTypes.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
      targetNamespace="http://NamespaceTest.com/OrderTypes"
      xmlns:cmn="http://NamespaceTest.com/CommonTypes"
      elementFormDefault="qualified">
    <xs:import schemaLocation="CommonTypes.xsd"
namespace="http://NamespaceTest.com/CommonTypes" />
    <xs:complexType name="OrderType">
        <xs:sequence>
            <xs:element maxOccurs="unbounded" name="Item">
                <xs:complexType>
                    <xs:sequence>
                        <xs:element name="ProductName" type="xs:string" />
                        <xs:element name="Quantity" type="xs:int" />
                        <xs:element name="UnitPrice" type="cmn:PriceType" />
                    </xs:sequence>
                </xs:complexType>
            </xs:element>
        </xs:sequence>
    </xs:complexType>
</xs:schema>
```

OrderTypes.xsd

To complete this example, take the following steps:

1. Save all schemas from the code listings above to files on the disk, making sure that you preserve the indicated file names.
2. Generate the schema wrapper code from the **Main.xsd** schema above, using the steps described in Generating Code from XML Schemas or DTD. After completing this step, you should have generated a compilable program in the language of your choice (C++, C#, or Java).
3. Add code to your C++, C#, or Java program from one the following example code listings, as required:

   - XML Namespaces and Prefixes (C++)

(C) 2015-2021 Altova GmbH

- XML Namespaces and Prefixes (C#)
- XML Namespaces and Prefixes (Java)

16.6.6.1 XML Namespaces and Prefixes (C++)

After you generate code from the example schema, a test C++ application is created, along with several supporting Altova libraries. Recall that the example schema (**Main.xsd**) has multiple namespace declarations. Consequently, the generated code includes namespaces that correspond to namespace aliases (prefixes) from the schema, namely: `Main::ord`, `Main::pur`, `Main::cmn`, and `Main::cust`.

In general, in order to control XML namespaces and prefixes with the help of the schema wrapper libraries, you have the following methods at your disposal:

- DeclareAllNamespacesFromSchema(). Call this method if you want to declare the same namespaces in your XML instance as in the schema. Otherwise, if you need different namespaces as in this example, then `DeclareNamespace()` should be used. The method `DeclareAllNamespacesFromSchema()` is not used in this example because we specifically want to create XML elements with prefixes that are slightly different from those declared in the schema.
- DeclareNamespace(). Call this method to create or override the existing namespace prefix attribute on an element. The element must already be created using either the `append()` or `appendWithPrefix()` methods, as further illustrated below.
- appendWithPrefix(). Use this method to append an instance element with a specific prefix. To create the XML instance illustrated in this example, it was sufficient to call this method for the root element only. All other elements were appended using just append(), and their prefixes were added automatically based on their namespaces, according to the rules above.

The code listing below shows you how to create an XML document with multiple namespace declarations and prefixed element names. Specifically, it generates a Purchase Order instance as illustrated in the Example: Purchase Order. Importantly, for illustrative purposes, some prefixes are overridden in the XML instance (that is, they are not exactly the same as the ones declared in the schema).

```
void Example()
{
   // Create the XML document and append the root element
   Main::pur::CMain doc = Main::pur::CMain::CreateDocument();
   Main::pur::CPurchaseType purchase = doc.Purchase.appendWithPrefix(_T("p"));

   // Set schema location
   doc.SetSchemaLocation(_T("Main.xsd"));

   // Declare namespaces on root element
   purchase.DeclareNamespace(_T("o"), _T("http://NamespaceTest.com/OrderTypes"));
   purchase.DeclareNamespace(_T("c"), _T("http://NamespaceTest.com/CustomerTypes"));
   purchase.DeclareNamespace(_T("cmn"), _T("http://NamespaceTest.com/CommonTypes"));

   // Append the OrderDetail element
   Main::ord::COrderType order = purchase.OrderDetail.append();
   Main::ord::CItemType item = order.Item.append();
   item.ProductName.append() = _T("Lawnmower");
   item.Quantity.append() = 1;
   item.UnitPrice.append() = 148.42;

   // Append the PaymentMethod element
   Main::cmn::CPaymentMethodTypeType paymentMethod = purchase.PaymentMethod.append();
```

```
   paymentMethod.SetEnumerationValue(Main::cmn::CPaymentMethodTypeType::k_VISA);

   // Append the CustomerDetails element
   Main::cust::CCustomerType customer = purchase.CustomerDetails.append();
   customer.Name.append() = _T("Alice Smith");
   Main::cmn::CAddressType deliveryAddress = customer.DeliveryAddress.append();
   deliveryAddress.Line1.append() = _T("123 Maple Street");
   deliveryAddress.Line2.append() = _T("Mill Valley");
   Main::cmn::CAddressType billingAddress = customer.BillingAddress.append();
   billingAddress.Line1.append() = _T("8 Oak Avenue");
   billingAddress.Line2.append() = _T("Old Town");

   // Save to file and release object from memory
   doc.SaveToFile(_T("Main1.xml"), true);
   doc.DestroyDocument();
}
```

16.6.6.2 XML Namespaces and Prefixes (C#)

After you generate code from the example schema, a test C# application is created, along with several supporting Altova libraries. Recall that the example schema (**Main.xsd**) has multiple namespace declarations. Consequently, the generated code includes namespaces that correspond to namespace aliases (prefixes) from the schema, namely: `Main.ord`, `Main.pur`, `Main.cmn`, and `Main.cust`.

In general, in order to control XML namespaces and prefixes with the help of the schema wrapper libraries, you have the following methods at your disposal:

- DeclareAllNamespacesFromSchema(). Call this method if you want to declare the same namespaces in your XML instance as in the schema. Otherwise, if you need different namespaces as in this example, then `DeclareNamespace()` should be used. The method `DeclareAllNamespacesFromSchema()` is not used in this example because we specifically want to create XML elements with prefixes that are slightly different from those declared in the schema.
- DeclareNamespace(). Call this method to create or override the existing namespace prefix attribute on an element. The element must already be created using either the `Append()` or `AppendWithPrefix()` methods, as further illustrated below.
- AppendWithPrefix(). Use this method to append an instance element with a specific prefix. To create the XML instance illustrated in this example, it was sufficient to call this method for the root element only. All other elements were appended using just Append(), and their prefixes were added automatically based on their namespaces, according to the rules above.

The code listing below shows you how to create an XML document with multiple namespace declarations and prefixed element names. Specifically, it generates a Purchase Order instance as illustrated in the Example: Purchase Order. Importantly, for illustrative purposes, some prefixes are overridden in the XML instance (that is, they are not exactly the same as the ones declared in the schema).

```
protected static void Example()
{
   // Create the XML document and append the root element
   pur.Main2 doc = pur.Main2.CreateDocument();
   pur.PurchaseType purchase = doc.Purchase.AppendWithPrefix("p");

   // Set schema location
   doc.SetSchemaLocation(@"Main.xsd");
```


(C) 2015-2021 Altova GmbH

```
    // Declare namespaces on root element
    purchase.DeclareNamespace("o", "http://NamespaceTest.com/OrderTypes");
    purchase.DeclareNamespace("c", "http://NamespaceTest.com/CustomerTypes");
    purchase.DeclareNamespace("cmn", "http://NamespaceTest.com/CommonTypes");

    // Append the OrderDetail element
    ord.OrderType order = purchase.OrderDetail.Append();
    ord.ItemType item = order.Item.Append();
    item.ProductName.Append().Value = "Lawnmower";
    item.Quantity.Append().Value = 1;
    item.UnitPrice.Append().Value = 148.42M;

    // Append the PaymentMethod element
    cmn.PaymentMethodTypeType paymentMethod = purchase.PaymentMethod.Append();
    paymentMethod.EnumerationValue = cmn.PaymentMethodTypeType.EnumValues.eVISA;

    // Append the CustomerDetails element
    cust.CustomerType customer = purchase.CustomerDetails.Append();
    customer.Name.Append().Value = "Alice Smith";
    cmn.AddressType deliveryAddress = customer.DeliveryAddress.Append();
    deliveryAddress.Line1.Append().Value = "123 Maple Street";
    deliveryAddress.Line2.Append().Value = "Mill Valley";
    cmn.AddressType billingAddress = customer.BillingAddress.Append();
    billingAddress.Line1.Append().Value = "8 Oak Avenue";
    billingAddress.Line2.Append().Value = "Old Town";

    // Save to file
    doc.SaveToFile("PurchaseOrder.xml", true);
}
```

16.6.6.3 XML Namespaces and Prefixes (Java)

After you generate code from the example schema, a test Java application is created, along with several supporting Altova libraries. Recall that the example schema (**Main.xsd**) has multiple namespace declarations. Consequently, the generated code includes namespaces that correspond to namespace aliases (prefixes) from the schema, namely: `com.Main.ord`, `com.Main.pur`, `com.Main.cmn`, and `com.Main.cust`.

In general, in order to control XML namespaces and prefixes with the help of the schema wrapper libraries, you have the following methods at your disposal:

- declareAllNamespacesFromSchema(). Call this method if you want to declare the same namespaces in your XML instance as in the schema. Otherwise, if you need different namespaces as in this example, then `declareNamespace()` should be used. The method `declareAllNamespacesFromSchema()` is not used in this example because we specifically want to create XML elements with prefixes that are slightly different from those declared in the schema.
- declareNamespace(). Call this method to create or override the existing namespace prefix attribute on an element. The element must already be created using either the `append()` or `appendWithPrefix()` methods, as further illustrated below.
- appendWithPrefix(). Use this method to append an instance element with a specific prefix. To create the XML instance illustrated in this example, it was sufficient to call this method for the root element only. All other elements were appended using just append(), and their prefixes were added automatically based on their namespaces, according to the rules above.

The code listing below shows you how to create an XML document with multiple namespace declarations and prefixed element names. Specifically, it generates a Purchase Order instance as illustrated in the Example:

Purchase Order. Importantly, for illustrative purposes, some prefixes are overridden in the XML instance (that is, they are not exactly the same as the ones declared in the schema).

```
protected static void example() throws Exception {
   // Create the XML document and append the root element
   com.Main.pur.Main2 doc = com.Main.pur.Main2.createDocument();
   com.Main.pur.PurchaseType purchase = doc.Purchase.appendWithPrefix("p");

   // Set schema location
   doc.setSchemaLocation("Main.xsd");

   // Declare namespaces on root element
   purchase.declareNamespace("o", "http://NamespaceTest.com/OrderTypes");
   purchase.declareNamespace("c", "http://NamespaceTest.com/CustomerTypes");
   purchase.declareNamespace("cmn", "http://NamespaceTest.com/CommonTypes");

   // Append the OrderDetail element
   com.Main.ord.OrderType order = purchase.OrderDetail.append();
   com.Main.ord.ItemType item = order.Item.append();
   item.ProductName.append().setValue("Lawnmower");
   item.Quantity.append().setValue(1);
   java.math.BigDecimal price = new java.math.BigDecimal("148.42");
   item.UnitPrice.append().setValue(price);

   // Append the PaymentMethod element
   com.Main.cmn.PaymentMethodTypeType paymentMethod = purchase.PaymentMethod.append();
   paymentMethod.setEnumerationValue(com.Main.cmn.PaymentMethodTypeType.EVISA);

   // Append the CustomerDetails element
   com.Main.cust.CustomerType customer = purchase.CustomerDetails.append();
   customer.Name.append().setValue("Alice Smith");
   com.Main.cmn.AddressType deliveryAddress = customer.DeliveryAddress.append();
   deliveryAddress.Line1.append().setValue("123 Maple Street");
   deliveryAddress.Line2.append().setValue("Mill Valley");
   com.Main.cmn.AddressType billingAddress = customer.BillingAddress.append();
   billingAddress.Line1.append().setValue("8 Oak Avenue");
   billingAddress.Line2.append().setValue("Old Town");

   // Save to file
   doc.saveToFile("PurchaseOrder.xml", true);
}
```


(C) 2015-2021 Altova GmbH

16.7 Reference to Generated Classes (C++)

This chapter includes a description of C++ classes generated with MapForce from a DTD or XML schema (see Generating Code from XML Schemas or DTDs). You can integrate these classes into your code to read, modify, and write XML documents.

Note: The generated code does include other supporting classes, which are not listed here and are subject to modification.

16.7.1 altova::DateTime

This class enables you to process XML attributes or elements that have date and time types, such as `xs:dateTime`.

Constructors

Name	Description
`DateTime()`	Initializes a new instance of the `DateTime` class to 12:00:00 midnight, January 1, 0001.
`DateTime(__int64 value, short timezone)`	Initializes a new instance of the `DateTime` class. The `value` parameter represents the number of ticks (100-nanosecond intervals) that have elapsed since 12:00:00 midnight, January 1, 0001.
`DateTime(int year, unsigned char month, unsigned char day, unsigned char hour, unsigned char minute, double second)`	Initializes a new instance of the `DateTime` class to the year, month, day, hour, minute, and second supplied as argument.
`DateTime(int year, unsigned char month, unsigned char day, unsigned char hour, unsigned char minute, double second, short timezone)`	Initializes a new instance of the `DateTime` class to the year, month, day, hour, minute, second and timezone supplied as argument. The timezone is expressed in minutes and can be positive or negative. For example, the timezone "UTC-01:00" is expressed as "-60".

Methods

Name	Description
`unsigned char Day() const`	Returns the day of month of the current `DateTime` object. The return values range from 1 through 31.
`int DayOfYear() const`	Returns the day of year of the current `DateTime` object. The return values range from 1 through 366.
`bool HasTimezone() const`	Returns Boolean **true** if the current `DateTime` object has a timezone defined; **false** otherwise.

Name	Description
`unsigned char Hour() const`	Returns the hour of the current `DateTime` object. The return values range from 0 through 23.
`static bool IsLeapYear(int year)`	Returns Boolean **true** if the year of the `DateTime` class is a leap year; **false** otherwise.
`unsigned char Minute() const`	Returns the minute of the current `DateTime` object. The return values range from 0 through 59.
`unsigned char Month() const`	Returns the month of the current `DateTime` object. The return values range from 1 through 12.
`__int64 NormalizedValue() const`	Returns the value of the `DateTime` object expressed as the Coordinated Universal Time (UTC).
`double Second() const`	Returns the second of the current `DateTime` object. The return values range from 0 through 59.
`void SetTimezone(short tz)`	Sets the timezone of the current `DateTime` object to the timezone value supplied as argument. The **tz** argument is expressed in minutes and can be positive or negative.
`short Timezone() const`	Returns the timezone, in minutes, of the current `DateTime` object. Before using this method, make sure that the object actually has a timezone, by calling the `HasTimezone()` method.
`__int64 Value() const`	Returns the value of the `DateTime` object, expressed in the number of ticks (100-nanosecond intervals) that have elapsed since 12:00:00 midnight, January 1, 0001.
`int Weekday() const`	Returns the day of week of the current `DateTime` object, as an integer. Values range from 0 through 6, where 0 is Monday (ISO-8601).
`int Weeknumber() const`	Returns the number of week in the year of the current `DateTime` object. The return values are according to ISO-8601.
`int WeekOfMonth() const`	Returns the number of week in the month of the current `DateTime` object. The return values are according to ISO-8601.
`int Year() const`	Returns the year of the current `DateTime` object.

Example

```
void Example()
{
   // initialize a new DateTime instance to 12:00:00 midnight, January 1st, 0001
   altova::DateTime dt1 = altova::DateTime();

   // initialize a new DateTime instance using the year, month, day, hour, minute, and
second
   altova::DateTime dt2 = altova::DateTime(2015, 11, 10, 9, 8, 7);

   // initialize a new DateTime instance using the year, month, day, hour, minute,
```

```
second, and UTC +01:00 timezone
    altova::DateTime dt = altova::DateTime(2015, 11, 22, 13, 53, 7, 60);

   // Get the value of this DateTime object
   std::cout << "The number of ticks of the DateTime object is: " << dt.Value() <<
std::endl;

   // Get the year
   cout << "The year is: " << dt.Year() << endl;
   // Get the month
   cout << "The month is: " << (int)dt.Month() << endl;
   // Get the day of the month
   cout << "The day of the month is: " << (int) dt.Day() << endl;
   // Get the day of the year
   cout << "The day of the year is: " << dt.DayOfYear() << endl;
   // Get the hour
   cout << "The hour is: " << (int) dt.Hour() << endl;
   // Get the minute
   cout << "The minute is: " << (int) dt.Minute() << endl;
   // Get the second
   cout << "The second is: " << dt.Second() << endl;
   // Get the weekday
   cout << "The weekday is: " << dt.Weekday() << endl;
   // Get the week number
   cout << "The week of year is: " << dt.Weeknumber() << endl;
   // Get the week in month
   cout << "The week of month is: " << dt.WeekOfMonth() << endl;

   // Check whether a DateTime instance has a timezone
   if (dt.HasTimezone() == TRUE)
   {
      // output the value of the Timezone
      cout <<  "The timezone is: " << dt.Timezone() << endl;
   }
   else
   {
      cout <<  "No timezone has been defined." << endl;
   }

   // Construct a DateTime object with a timezone UTC+01:00 (Vienna)
   altova::DateTime vienna_dt = DateTime(2015, 11, 23, 14, 30, 59, +60);
   // Output the result in readable format
   cout << "The Vienna time: "
       << (int) vienna_dt.Month()
       << "-" << (int) vienna_dt.Day()
       << " " << (int) vienna_dt.Hour()
       << ":" << (int) vienna_dt.Minute()
       << ":" << (int) vienna_dt.Second()
       << endl;

   // Convert the value to UTC time
   DateTime utc_dt = DateTime(vienna_dt.NormalizedValue());
   // Output the result in readable format
   cout << "The UTC time:    "
      << (int) utc_dt.Month()
      << "-" << (int) utc_dt.Day()
      << " " << (int) utc_dt.Hour()
      << ":" << (int) utc_dt.Minute()
      << ":" << (int) utc_dt.Second()
```

```
      << endl;

   // Check if a year is a leap year
   int year = 2016;
   if( altova::DateTime::IsLeapYear(year) )
   { cout << year << " is a leap year" << endl; }
   else
   { cout << year << " is not a leap year" << endl; }
}
```

16.7.2 altova::Duration

This class enables you to process XML attributes or elements of type `xs:duration`.

Constructors

Name	Description
`Duration()`	Initializes a new instance of the `Duration` class to an empty value.
`Duration(const DayTimeDuration& dt)`	Initializes a new instance of the `Duration` class to a duration defined by the `dt` argument (see altova::DayTimeDuration).
`Duration(const YearMonthDuration& ym)`	Initializes a new instance of the `Duration` class to the duration defined by the `ym` argument (see altova::YearMonthDuration).
`Duration(const YearMonthDuration& ym, const DayTimeDuration& dt)`	Initializes a new instance of the `Duration` class to the duration defined by both the `dt` and the `ym` arguments (see altova::YearMonthDuration and altova::DayTimeDuration).

Methods

Name	Description
`int Days() const`	Returns the number of days in the current `Duration` instance.
`DayTimeDuration DayTime() const`	Returns the day and time duration in the current `Duration` instance, expressed as a `DayTimeDuration` object (see altova::DayTimeDuration).
`int Hours() const`	Returns the number of hours in the current `Duration` instance.
`bool IsNegative() const`	Returns Boolean **true** if the current `Duration` instance is negative.
`bool IsPositive() const`	Returns Boolean **true** if the current `Duration` instance is positive.
`int Minutes() const`	Returns the number of minutes in the current `Duration` instance.
`int Months() const`	Returns the number of months in the current `Duration` instance.
`double Seconds() const`	Returns the number of seconds in the current `Duration` instance.

(C) 2015-2021 Altova GmbH

Name	Description
`YearMonthDuration YearMonth() const`	Returns the year and month duration in the current `Duration` instance, expressed as a `YearMonthDuration` object (see altova::YearMonthDuration).
`int Years() const`	Returns the number of years in the current `Duration` instance.

Example

The following code listing illustrates creating a new `Duration` object, as well as reading values from it.

```
void ExampleDuration()
{
   // Create an empty Duration object
   altova::Duration empty_duration = altova::Duration();

   // Create a Duration object using an existing duration value
   altova::Duration duration1 = altova::Duration(empty_duration);

   // Create a YearMonth duration of six years and five months
   altova::YearMonthDuration yrduration = altova::YearMonthDuration(6, 5);

   // Create a DayTime duration of four days, three hours, two minutes, and one second
   altova::DayTimeDuration dtduration = altova::DayTimeDuration(4, 3, 2, 1);

   // Create a Duration object by combining the two previously created durations
   altova::Duration duration = altova::Duration(yrduration, dtduration);

   // Get the number of years in this Duration instance
   cout << "Years:  " << duration.Years() << endl;

   // Get the number of months in this Duration instance
   cout << "Months: " << duration.Months() << endl;

   // Get the number of days in this Duration instance
   cout << "Days:   " << duration.Days() << endl;

   // Get the number of hours in this Duration instance
   cout << "Hours:  " << duration.Hours() << endl;

   // Get the number of hours in this Duration instance
   cout << "Minutes: " << duration.Minutes() << endl;

   // Get the number of seconds in this Duration instance
   cout << "Seconds: " << duration.Seconds() << endl;
}
```

16.7.3 altova::DayTimeDuration

This class enables you to process XML schema duration types that consist of a day and time part.

Constructors

Name	Description
`DayTimeDuration()`	Initializes a new instance of the `DayTimeDuration` class to an empty value.
`DayTimeDuration(int days, int hours, int minutes, double seconds)`	Initializes a new instance of the `DayTimeDuration` class to the number of days, hours, minutes, and seconds supplied as arguments.
`explicit DayTimeDuration(__int64 value)`	Initializes a new instance of the `DayTimeDuration` class to a duration that consists of as many ticks (100-nanosecond intervals) as supplied in the **value** argument.

Methods

Name	Description
`int Days() const`	Returns the number of days in the current `DayTimeDuration` instance.
`int Hours() const`	Returns the number of hours in the current `DayTimeDuration` instance.
`bool IsNegative() const`	Returns Boolean **true** if the current `DayTimeDuration` instance is negative.
`bool IsPositive() const`	Returns Boolean **true** if the current `DayTimeDuration` instance is positive.
`int Minutes() const`	Returns the number of minutes in the current `DayTimeDuration` instance.
`double Seconds() const`	Returns the number of seconds in the current `DayTimeDuration` instance.
`__int64 Value() const`	Returns the value (in ticks) of the current `DayTimeDuration` instance.

(C) 2015-2021 Altova GmbH

16.7.4 altova::YearMonthDuration

This class enables you to process XML schema duration types that consist of a year and month part.

Constructors

Name	Description
`YearMonthDuration()`	Initializes a new instance of the `YearMonthDuration` class to an empty value.
`YearMonthDuration(int years, int months)`	Initializes a new instance of the `YearMonthDuration` class to the number of years and months supplied in the **years** and **months** arguments.
`explicit YearMonthDuration(int value)`	Initializes a new instance of the `YearMonthDuration` class to a duration that consists of as many ticks (100-nanosecond intervals) as supplied in the **value** argument.

Methods

Name	Description
`bool IsNegative() const`	Returns Boolean **true** if the current `YearMonthDuration` instance is negative.
`bool IsPositive() const`	Returns Boolean **true** if the current `YearMonthDuration` instance is positive.
`int Months() const`	Returns the number of months in the current `YearMonthDuration` instance.
`int Value() const`	Returns the value (in ticks) of the current `YearMonthDuration` instance.
`int Years()`	Returns the number of years in the current `YearMonthDuration` instance.

16.7.5 altova::meta::Attribute

This class enables you to access schema information about classes generated from attributes. Note that this class is not meant to provide dynamic information about particular instances of an attribute in an XML document. Instead, it enables you to obtain programmatically information about a particular attribute defined in the XML schema.

Methods

Name	Description
`SimpleType GetDataType()`	Returns the type of the attribute content.

Name	Description
`string_type GetLocalName()`	Returns the local name of the attribute.
`string_type GetNamespaceURI()`	Returns the namespace URI of the attribute.
`bool IsRequired()`	Returns true if the attribute is required.

Operators

Name	Description
`bool operator()`	Returns true if this is not the NULL Attribute.
`bool operator!()`	Returns true if this is the NULL Attribute.

16.7.6 altova::meta::ComplexType

This class enables you to access schema information about classes generated from complex types. Note that this class is not meant to provide dynamic information about particular instances of a complex type in an XML document. Instead, it enables you to obtain programmatically information about a particular complex type defined in the XML schema.

Methods

Name	Description
`Attribute FindAttribute(const char_type* localName, const char_type* namespaceURI)`	Finds the attribute with the specified local name and namespace URI.
`Element FindElement(const char_type* localName, const char_type* namespaceURI)`	Finds the element with the specified local name and namespace URI.
`std::vector<Attribute> GetAttributes()`	Returns a list of all attributes.
`ComplexType GetBaseType()`	Returns the base type of this type.
`SimpleType GetContentType()`	Returns the simple type of the content.
`std::vector<Element> GetElements()`	Returns a list of all elements.
`string_type GetLocalName()`	Returns the local name of the type.
`string_type GetNamespaceURI()`	Returns the namespace URI of the type.

(C) 2015-2021 Altova GmbH

Operators

Name	Description
`bool operator()`	Returns true if this is not the NULL ComplexType.
`bool operator!()`	Returns true if this is the NULL ComplexType.

16.7.7 altova::meta::Element

This class enables you to access information about classes generated from schema elements. Note that this class is not meant to provide dynamic information about particular instances of an element in an XML document. Instead, it enables you to obtain programmatically information about a particular element defined in the XML schema.

Methods

Name	Description
`ComplexType GetDataType()`	Returns the type of the element. Note that this is always a complex type even if declared as simple in the original schema. Use GetContentType() of the returned object to get the simple content type.
`string_type GetLocalName()`	Returns the local name of the element.
`unsigned int GetMaxOccurs()`	Returns the maxOccurs value defined in the schema.
`unsigned int GetMinOccurs()`	Returns the minOccurs value defined in the schema.
`string_type GetNamespaceURI()`	Returns the namespace URI of the element.

Operators

Name	Description
`bool operator()`	Returns true if this is not the NULL Element.
`bool operator!()`	Returns true if this is the NULL Element.

16.7.8 altova::meta::SimpleType

This class enables you to access schema information about classes generated from simple types. Note that this class is not meant to provide dynamic information about particular instances of simple types in an XML document. Instead, it enables you to obtain programmatically information about a particular simple type defined in the XML schema.

Methods

Name	Description
`SimpleType GetBaseType()`	Returns the base type of this type.
`std::vector<string_type> GetEnumerations()`	Returns a list of all enumeration facets.
`unsigned int GetFractionDigits()`	Returns the value of this facet.
`unsigned int GetLength()`	Returns the value of this facet.
`string_type GetLocalName()`	Returns the local name of the type.
`string_type GetMaxExclusive()`	Returns the value of this facet.
`string_type GetMaxInclusive()`	Returns the value of this facet.
`unsigned int GetMaxLength()`	Returns the value of this facet.
`string_type GetMinExclusive()`	Returns the value of this facet.
`string_type GetMinInclusive()`	Returns the value of this facet.
`unsigned int GetMinLength()`	Returns the value of this facet.
`string_type GetNamespaceURI()`	Returns the namespace URI of the type.
`std::vector<string_type> GetPatterns()`	Returns a list of all pattern facets.
`unsigned int GetTotalDigits()`	Returns the value of this facet.
`WhitespaceType GetWhitespace()`	Returns the value of the whitespace facet, which is one of: • Whitespace_Unknown • Whitespace_Preserve • Whitespace_Replace • Whitespace_Collapse

Operators

Name	Description
`bool operator()`	Returns true if this is not the NULL SimpleType.
`bool operator!()`	Returns true if this is the NULL SimpleType.

16.7.9 [YourSchema]::[CDoc]

When code is generated from an XML Schema, the generated code provides a document class with the same name as the schema. This class contains all possible root elements as members, as well as the following

(C) 2015-2021 Altova GmbH

methods. Note that, in the method names below, "CDoc" stands for the name of the generated document class itself.

Methods

Name	Description
`static CDoc CreateDocument()`	Creates a new, empty XML document. Must be released using DestroyDocument().
`static void DeclareAllNamespacesFromSchema(ElementType& node)`	Declares all namespaces from the XML Schema on the element supplied as argument (typically, the XML root element). Calling this method is useful if your schema has multiple namespace declarations, each mapped to a prefix, and you would like to declare all of them on the element supplied as argument.
`void DestroyDocument()`	Destroys a document. All references to the document and its nodes are invalidated. This must be called when you finished working with a document.
`static CDoc LoadFromBinary(const std:vector<unsigned char>& xml)`	Loads an XML document from a byte array.
`static CDoc LoadFromFile(const string_type& fileName)`	Loads an XML document from a file.
`static CDoc LoadFromString(const string_type& xml)`	Loads an XML document from a string.
`std::vector<unsigned char> SaveToBinary(bool prettyPrint)`	Saves an XML document to a byte array. When set to true, the `prettyPrint` argument re-formats the XML document for better readability.
`std::vector<unsigned char> SaveToBinary(bool prettyPrint, const string_type & encoding)`	Saves an XML document to a byte array, with optional "pretty-print" formatting, with the specified encoding.
`std::vector<unsigned char> SaveToBinary(bool prettyPrint, const string_type & encoding, bool bBigEndian, bool bBOM)`	Saves an XML document to a byte array, with optional "pretty-print" formatting, with the specified encoding. Byte order and Unicode byte-order mark can be specified for Unicode encodings.
`void SaveToFile(const string_type & fileName, bool prettyPrint)`	Saves an XML document to a file, with optional "pretty-print" formatting.
`void SaveToFile(const string_type & fileName, bool omitXmlDecl)`	Saves an XML document to a file. If the `omitXmlDecl` argument is set to true, the XML declaration will not be written.
`void SaveToFile(const string_type & fileName, bool omitXmlDecl, const string_type & encoding)`	Saves an XML document to a file with the specified encoding. If the `omitXmlDecl` argument is set to true, the XML declaration will not be written.
`void SaveToFile(const string_type & fileName, bool prettyPrint, bool omitXmlDecl, const string_type & encoding, bool bBigEndian, bool bBOM)`	Saves an XML document to a file, with optional "pretty-print" formatting, with the specified encoding. Byte order and Unicode byte-order mark can be specified for Unicode encodings.

Name	Description
`void SaveToFile(const string_type & fileName, bool prettyPrint, bool omitXmlDecl, const string_type & encoding, bool bBigEndian, bool bBOM, const string_type & lineend)`	Saves an XML document to a file, with optional "pretty-print" formatting, with the specified encoding and the specified line end. Byte order and Unicode byte-order mark can be specified for Unicode encodings. This method is only available if you generated the code for the Xerces3 XML library (see Code Generator Options).
`void SaveToFile(const string_type& fileName, bool prettyPrint, bool omitXmlDecl, const string_type & encoding, const string_type & lineend)`	Saves an XML document to a file, with optional "pretty-print" formatting, with the specified encoding and the specified line end. This method is only available if you generated the code for the Xerces3 XML library (see Code Generator Options).
`void SaveToFile(const string_type & fileName, bool prettyPrint, const string_type & encoding)`	Saves an XML document to a file, with optional "pretty-print" formatting, with the specified encoding.
`void SaveToFile(const string_type& fileName, bool prettyPrint, const string_type & encoding, bool bBigEndian, bool bBOM)`	Saves an XML document to a file, with optional "pretty-print" formatting, with the specified encoding. Byte order and Unicode byte-order mark can be specified for Unicode encodings.
`void SaveToFile(const string_type& fileName, bool prettyPrint, const string_type & encoding, bool bBigEndian, bool bBOM, const string_type & lineend)`	Saves an XML document to a file with the specified encoding and the specified line end. Byte order and Unicode byte-order mark can be specified for Unicode encodings. This method is only available if you generated the code for the Xerces3 XML library (see Code Generator Options).
`void SaveToFile(const string_type& fileName, bool prettyPrint, const string_type & encoding, const string_type & lineend)`	Saves an XML document to a file, with optional "pretty-print" formatting, with the specified encoding and the specified line end. This method is only available if you generated the code for the Xerces3 XML library (see Code Generator Options).
`string_type SaveToString(bool prettyPrint)`	Saves an XML document to a string, with optional "pretty-print" formatting.
`string_type SaveToString(bool prettyPrint, bool omitXmlDecl)`	Saves an XML document to a string, with optional "pretty-print" formatting. If the `omitXmlDecl` argument is set to true, the XML declaration will not be written.
`void SetDTDLocation(const string_type & dtdLocation)`	Adds a DOCTYPE declaration with the specified system ID. A root element must already exist. This method is not supported for MSXML, since it is not possible to add a DOCTYPE declaration to a document in memory.
`void SetSchemaLocation(const string_type & schemaLocation)`	Adds an xsi:schemaLocation or xsi:noNamespaceSchemaLocation attribute to the root element. A root element must already exist.

(C) 2015-2021 Altova GmbH

16.7.10 [YourSchema]::[ElementType]

This class provides methods for manipulating XML elements from your schema. Methods of this class can be called on elements, not on the XML document itself. Note that, in order to call methods of this class, you don't need to instantiate the class directly. Any element created using the **append()** or **appendWithPrefix()** methods is of `[ElementType]` type.

Methods

Name	Description

`void DeclareNamespace(const string_type prefix, const string_type nsURI)`	This method takes two arguments that are both of string type: the prefix and the namespace URI that you want to use. The prefix supplied as argument will be mapped to the namespace URI value supplied as argument. If the prefix supplied as argument is empty, the method creates or overrides the default namespace declaration in the element.
	
	For example, let's assume that the XML document has an XML element called "purchase". If you call
	
	`purchase.DeclareNamespace(_T("ord"), _T("http://OrderTypes"));`
	
	then the XML document becomes
	
	`<purchase xmlns:ord="http://OrderTypes" />`
	
	Another example, if you call:
	
	`purchase.DeclareNamespace(_T(""), _T("http://OrderTypes"));`
	
	then the XML document becomes
	
	`<purchase xmlns="http://OrderTypes" />`
	
	Note: The declared namespace is used when appending subsequent child elements or attributes, according to the following rules:
	
	1. If the child namespace is the default, then use empty prefix.
	2. If the child namespace is equal to the parent one, then use the parent prefix.
	3. Otherwise, search for nearest prefix from parent to top, using the lookup algorithm described in section "B.2: Namespace Prefix Lookup" at https://www.w3.org/TR/2002/WD-DOM-Level-3-Core-20021022/namespaces-algorithms.html.

Name	Description
	4. If there is no prefix for element namespace found, then use empty prefix.

16.7.11 [YourSchema]::MemberAttribute

When code is generated from an XML schema, a class such as this one is created for each member attribute of a type.

Methods

Name	Description
`bool exists()`	Returns true if the attribute exists.
`int GetEnumerationValue()`	Generated for enumeration types only. Returns one of the constants generated for the possible values, or "Invalid" if the value does not match any of the enumerated values in the schema.
`altova::meta::Attribute info()`	Returns an object for querying schema information (see `altova::meta::Attribute`).
`void remove()`	Removes the attribute from its parent element.
`void SetEnumerationValue(int)`	Generated for enumeration types only. Pass one of the constants generated for the possible values to this method to set the value.

16.7.12 [YourSchema]::MemberElement

When code is generated from an XML schema, a class such as this one is created for each member element of a type. In the descriptions below, "MemberType" stands for the name of the member element itself.

Methods

Name	Description
`Iterator<MemberType> all()`	Returns an object for iterating instances of the member element.
`MemberType append()`	Creates a new element and appends it to its parent.
`MemberType appendWithPrefix(string_type prefix)`	Creates a new element having the prefix supplied as argument, and appends it to its parent. For an example, see Example: Purchase Order.
`unsigned int count()`	Returns the count of elements.

(C) 2015-2021 Altova GmbH

Name	Description
`int GetEnumerationValue()`	Generated for enumeration types only. Returns one of the constants generated for the possible values, or **Invalid** if the value does not match any of the enumerated values in the schema.
`bool exists()`	Returns true if at least one element exists.
`MemberType first()`	Returns the first instance of the member element.
`MemberType operator[](unsigned int index)`	Returns the member element specified by the index.
`altova::meta::Element info()`	Returns an object for querying schema information (see `altova::meta::Element`).
`MemberType last()`	Returns the last instance of the member element.
`void remove()`	Deletes all occurrences of the element from its parent.
`void removeAt(unsigned int index)`	Deletes the occurrence of the element specified by the index.
`void SetEnumerationValue(int)`	Generated for enumeration types only. Pass one of the constants generated for the possible values to this method to set the value.

16.8 Reference to Generated Classes (C#)

This chapter includes a description of C# classes generated with MapForce from a DTD or XML schema (see Generating Code from XML Schemas or DTDs). You can integrate these classes into your code to read, modify, and write XML documents.

Note: The generated code does include other supporting classes, which are not listed here and are subject to modification.

16.8.1 Altova.Types.DateTime

This class enables you to process XML attributes or elements that have date and time types, such as `xs:dateTime`.

Constructors

	Name	Description
	`DateTime(DateTime obj)`	Initializes a new instance of the `DateTime` class to the `DateTime` object supplied as argument.
	`DateTime(System.DateTime newvalue)`	Initializes a new instance of the `DateTime` class to the `System.DateTime` object supplied as argument.
	`DateTime(int year, int month, int day, int hour, int minute, double second, int offsetTZ)`	Initializes a new instance of the `DateTime` class to the year, month, day, hour, minute, second, and timezone offset supplied as arguments.
	`DateTime(int year, int month, int day, int hour, int minute, double second)`	Initializes a new instance of the `DateTime` class to the year, month, day, hour, minute, and second supplied as arguments.
	`DateTime(int year, int month, int day)`	Initializes a new instance of the `DateTime` class to the year, month and day supplied as arguments.

Properties

	Name	Description
	`bool HasTimezone`	Gets a Boolean value which indicates if the `DateTime` has a timezone.
	`static DateTime Now`	Gets a `DateTime` object that is set to the current date and time on this computer.
	`short TimezoneOffset`	Gets or sets the timezone offset, in minutes, of the `DateTime` object.
	`System.DateTime Value`	Gets or sets the value of the `DateTime` object as a `System.DateTime` value.

(C) 2015-2021 Altova GmbH

Methods

	Name	Description
	`int CompareTo(object obj)`	The `DateTime` class implements the `IComparable` interface. This method compares the current instance of `DateTime` to another object and returns an integer that indicates whether the current instance precedes, follows, or occurs in the same position in the sort order as the other object. See also https://msdn.microsoft.com/en-us/library/system.icomparable.compareto(v=vs.110).aspx
	`override bool Equals(object obj)`	Returns **true** if the specified object is equal to the current object; **false** otherwise.
	`System.DateTime GetDateTime(bool correctTZ)`	Returns a `System.DateTime` object from the current `Altova.Types.DateTime` instance. The `correctTZ` Boolean argument specifies whether the time of the returned object must be adjusted according to the timezone of the current `Altova.Types.DateTime` instance.
	`override int GetHashCode()`	Returns the hash code of the current instance.
	`int GetWeekOfMonth()`	Returns the number of the week in month as an integer.

	`static DateTime Parse( string s )`	Creates a `DateTime` object from the string supplied as argument. For example, the following sample string values would be converted successfully to a `DateTime` object:
		
		`2015-01-01T23:23:23`
		`2015-01-01`
		`2015-11`
		`23:23:23`
		
		An exception is raised if the string cannot be converted to a `DateTime` object.
		
		Note that this method is static and can only be called on the `Altova.Types.DateTime` class itself, not on an instance of the class.

	`static DateTime Parse(string s, DateTimeFormat format)`	Creates a `DateTime` object from a string, using the format supplied as argument. For the list of possible formats, see Altova.Types.DateTimeFormat. An exception is raised if the string cannot be converted to a `DateTime` object. Note that this method is static and can only be called on the `Altova.Types.DateTime` class itself, not on an instance of the class.
	`override string ToString()`	Converts the `DateTime` object to a string.

	Name	Description
	`string ToString(DateTimeFormat format)`	Converts the `DateTime` object to a string, using the format supplied as argument. For the list of possible formats, see Altova.Types.DateTimeFormat.

Operators

Name	Description
`!=`	Determines if `DateTime` a is not equal to `DateTime` b.
`<`	Determines if `DateTime` a is less than `DateTime` b.
`<=`	Determines if `DateTime` a is less than or equal to `DateTime` b.
`==`	Determines if `DateTime` a is equal to `DateTime` b.
`>`	Determines if `DateTime` a is greater than `DateTime` b.
`>=`	Determines if `DateTime` a is greater than or equal to `DateTime` b.

Examples

Before using the following code listings in your program, ensure the Altova types are imported:

```
using Altova.Types;
```

The following code listing illustrates various ways to create `DateTime` objects:

```
protected static void DateTimeExample1()
{
   // Create a DateTime object from the current system time
   Altova.Types.DateTime dt = new Altova.Types.DateTime(System.DateTime.Now);
   Console.WriteLine("The current time is: " + dt.ToString());

   // Create an Altova DateTime object from parts (no timezone)
   Altova.Types.DateTime dt1 = new Altova.Types.DateTime(2015, 10, 12, 10, 50, 33);
   Console.WriteLine("My custom time is : " + dt1.ToString());

   // Create an Altova DateTime object from parts (with UTC+60 minutes timezone)
   Altova.Types.DateTime dt2 = new Altova.Types.DateTime(2015, 10, 12, 10, 50, 33, 60);
   Console.WriteLine("My custom time with timezone is : " + dt2.ToString());

   // Create an Altova DateTime object by parsing a string
   Altova.Types.DateTime dt3 = Altova.Types.DateTime.Parse("2015-01-01T23:23:23");
   Console.WriteLine("Time created from string: " + dt3.ToString());

   // Create an Altova DateTime object by parsing a string formatted as schema date
   Altova.Types.DateTime dt4 = Altova.Types.DateTime.Parse("2015-01-01",
DateTimeFormat.W3_date);
   Console.WriteLine("Time created from string formatted as schema date: " +
dt4.ToString());
}
```


(C) 2015-2021 Altova GmbH

The following code listing illustrates various ways to format `DateTime` objects:

```
protected static void DateTimeExample2()
{
   // Create a DateTime object from the current system time
   Altova.Types.DateTime dt = new Altova.Types.DateTime(System.DateTime.Now);

   // Output the unformatted DateTime
   Console.WriteLine("Unformatted time: " + dt.ToString());

   // Output this DateTime formatted using various formats
   Console.WriteLine("S_DateTime:       " + dt.ToString(DateTimeFormat.S_DateTime));
   Console.WriteLine("S_Days:           " + dt.ToString(DateTimeFormat.S_Days));
   Console.WriteLine("S_Seconds:        " + dt.ToString(DateTimeFormat.S_Seconds));
   Console.WriteLine("W3_date:          " + dt.ToString(DateTimeFormat.W3_date));
   Console.WriteLine("W3_dateTime:      " + dt.ToString(DateTimeFormat.W3_dateTime));
   Console.WriteLine("W3_gDay:          " + dt.ToString(DateTimeFormat.W3_gDay));
   Console.WriteLine("W3_gMonth:        " + dt.ToString(DateTimeFormat.W3_gMonth));
   Console.WriteLine("W3_gMonthDay:     " + dt.ToString(DateTimeFormat.W3_gMonthDay));
   Console.WriteLine("W3_gYear:         " + dt.ToString(DateTimeFormat.W3_gYear));
   Console.WriteLine("W3_gYearMonth:    " + dt.ToString(DateTimeFormat.W3_gYearMonth));
   Console.WriteLine("W3_time:          " + dt.ToString(DateTimeFormat.W3_time));
}
```

16.8.2 Altova.Types.DateTimeFormat

The `DateTimeFormat` enum type has the following constant values:

Value	Description	Example
S_DateTime	Formats the value as standard dateTime, with a precision of a ten-millionth of a second, including timezone.	`2015-11-12 12:19:03.9019132+01:00`
S_Days	Formats the value as number of days elapsed since the UNIX epoch.	`735913.631897345108796296 2963`
S_Seconds	Formats the value as number of seconds elapsed since the UNIX epoch, with a precision of a ten-millionth of a second.	`63582937678.0769062`
W3_date	Formats the value as schema date.	`2015-11-12`
W3_dateTime	Formats the value as schema dateTime.	`2015-11-12T15:12:14.5194251`
W3_gDay	Formats the value as schema gDay.	`---12` (assuming that the date is 12th of the month)
W3_gMonth	Formats the value as schema gMonth.	`--11`

Value	Description	Example
		(assuming that the month is November)
W3_gMonthDay	Formats the value as schema gMonthDay.	`--11-12` (assuming that the date is 12th of November)
W3_gYear	Formats the value as schema gYear.	`2015` (assuming that the year is 2015)
W3_gYearMonth	Formats the value as schema gYearMonth.	`2015-11` (assuming that the year is 2015 and the month is November)
W3_time	Formats the value as schema time, with a precision of a ten-millionth of a second.	`15:19:07.5582719`

(C) 2015-2021 Altova GmbH

16.8.3 Altova.Types.Duration

This class enables you to process XML attributes or elements of type `xs:duration`.

Constructors

	Name	Description
	`Duration(Duration obj)`	Initializes a new instance of the `Duration` class to the `Duration` object supplied as argument.
	`Duration(System.TimeSpan newvalue)`	Initializes a new instance of the `Duration` class to the `System.TimeSpan` object supplied as argument.
	`Duration(long ticks)`	Initializes a new instance of the `Duration` class to the number of ticks supplied as argument.
	`Duration(int newyears, int newmonths, int days, int hours, int minutes, int seconds, double partseconds, bool bnegative)`	Initializes a new instance of the `Duration` class to a duration built from parts supplied as arguments.

Properties

	Name	Description
	`int Months`	Gets or sets the number of months of the current instance of `Duration`.
	`System.TimeSpan Value`	Gets or sets the value (as `System.TimeSpan`) of the current instance of `Duration`.
	`int Years`	Gets or sets the number of years of the current instance of `Duration`.

Methods

	Name	Description
	`override bool Equals(object other)`	Returns **true** if the specified object is equal to the current object; **false** otherwise.
	`override int GetHashCode()`	Returns the hash code of the current instance.
	`bool IsNegative()`	Returns **true** if the current instance of `Duration` represents a negative duration.
	`static Duration Parse( string s, ParseType pt )`	Returns an `Altova.Types.Duration` object parsed from the string supplied as argument, using the parse type supplied as argument. Valid parse type values:

	Name	Description
		DURATION Parse duration assuming that year, month, day, as well as time duration parts exist. **YEARMONTH** Parse duration assuming that only year and month parts exist. **DAYTIME** Parse duration assuming that only the day and time parts exist. Note that this method is static and can only be called on the class itself, not on an instance of the class.
	`override string ToString()`	Converts the current `Duration` instance to string. For example, a time span of 3 hours, 4 minutes, and 5 seconds would be converted to "PT3H4M5S".
	`string ToYearMonthString()`	Converts the current `Duration` instance to string, using the "Year and Month" parse type.

Operators

Name	Description
`!=`	Determines if `Duration` a is not equal to `Duration` b.
`==`	Determines if `Duration` a is equal to `Duration` b.

Examples

Before using the following code listings in your program, ensure the Altova types are imported:

```
using Altova.Types;
```

The following code listing illustrates various ways to create `Duration` objects:

```
protected static void DurationExample1()
{
   // Create a new time span of 3 hours, 4 minutes, and 5 seconds
   System.TimeSpan ts = new TimeSpan(3, 4, 5);
   // Create a Duration from the time span
   Duration dr = new Duration(ts);
   // The output is: PT3H4M5S
   Console.WriteLine("Duration created from TimeSpan: " + dr.ToString());

   // Create a negative Altova.Types.Duration from 6 years, 5 months, 4 days, 3 hours,
       // 2 minutes, 1 second, and .33 of a second
   Duration dr1 = new Duration(6, 5, 4, 3, 2, 1, .33, true);
   // The output is: -P6Y5M4DT3H2M1.33S
   Console.WriteLine("Duration created from parts: " + dr1.ToString());

   // Create a Duration from a string using the DAYTIME parse type
```

```
   Duration dr2 = Altova.Types.Duration.Parse("-P4DT3H2M1S",
Duration.ParseType.DAYTIME);
   // The output is -P4DT3H2M1S
   Console.WriteLine("Duration created from string: " + dr2.ToString());

   // Create a duration from ticks
   Duration dr3 = new Duration(System.DateTime.UtcNow.Ticks);
   // Output the result
   Console.WriteLine("Duration created from ticks: " + dr3.ToString());
}
```

The following code listing illustrates getting values from `Duration` objects:

```
protected static void DurationExample2()
{
   // Create a negative Altova.Types.Duration from 6 years, 5 months, 4 days, 3 hours,
       // 2 minutes, 1 second, and .33 of a second
   Duration dr = new Duration(6, 5, 4, 3, 2, 1, .33, true);
   // The output is: -P6Y5M4DT3H2M1.33S
   Console.WriteLine("The complete duration is: " + dr.ToString());

   // Get only the year and month part as string
   string dr1 = dr.ToYearMonthString();
   Console.WriteLine("The YEARMONTH part is: " + dr1);

   // Get the number of years in duration
   Console.WriteLine("Years: " + dr.Years);

   // Get the number of months in duration
   Console.WriteLine("Months: " + dr.Months);
}
```

16.8.4 Altova.Xml.Meta.Attribute

This class enables you to access schema information about classes generated from attributes. Note that this class is not meant to provide dynamic information about particular instances of an attribute in an XML document. Instead, it enables you to obtain programmatically information about a particular attribute defined in the XML schema.

Properties

	Name	Description
	`SimpleType DataType`	Returns the type of the attribute content.
	`string LocalName`	Returns the local name of the attribute.
	`string NamespaceURI`	Returns the namespace URI of the attribute.
	`XmlQualifiedName QualifiedName`	Returns the qualified name of the attribute.
	`bool Required()`	Returns true if the attribute is required.

16.8.5 Altova.Xml.Meta.ComplexType

This class enables you to access schema information about classes generated from complex types. Note that this class is not meant to provide dynamic information about particular instances of a complex type in an XML document. Instead, it enables you to obtain programmatically information about a particular complex type defined in the XML schema.

Properties

	Name	Description
	`Attribute[] Attributes`	Returns a list of all attributes.
	`ComplexType BaseType`	Returns the base type of this type or null if no base type exists.
	`SimpleType ContentType`	Returns the simple type of the content.
	`Element[] Elements`	Returns a list of all elements.
	`string LocalName`	Returns the local name of the type.
	`string NamespaceURI`	Returns the namespace URI of the type.
	`XmlQualifiedName QualifiedName`	Returns the qualified name of this type.

Methods

	Name	Description
	`ComplexType BaseType`	Returns the base type of this type.
	`bool Equals(obj)`	Checks if two info objects refer to the same type, based on qualified name comparison. Returns true if the type has the same qualified name.
	`Attribute FindAttribute(string localName, string namespaceURI)`	Finds the attribute with the specified local name and namespace URI.
	`Element FindElement(string localName, string namespaceURI)`	Finds the element with the specified local name and namespace URI.

16.8.6 Altova.Xml.Meta.Element

This class enables you to access information about classes generated from schema elements. Note that this class is not meant to provide dynamic information about particular instances of an element in an XML document. Instead, it enables you to obtain programmatically information about a particular element defined in the XML schema.

(C) 2015-2021 Altova GmbH

Properties

	Name	Description
	`ComplexType DataType`	Returns the type of the element. Note that this is always a complex type even if declared as simple in the original schema. Use the `ContentType` property of the returned object to get the simple content type.
	`string` `LocalName`	Returns the local name of the element.
	`int` `MaxOccurs`	Returns the `maxOccurs` value defined in the schema.
	`int` `MinOccurs`	Returns the `minOccurs` value defined in the schema.
	`string` `NamespaceURI`	Returns the namespace URI of the element.
	`XmlQualifiedName QualifiedName`	Returns the qualified name of the element.

16.8.7 Altova.Xml.Meta.SimpleType

This class enables you to access schema information about classes generated from simple types. Note that this class is not meant to provide dynamic information about particular instances of simple types in an XML document. Instead, it enables you to obtain programmatically information about a particular simple type defined in the XML schema.

Properties

	Name	Description
	`SimpleType BaseType`	Returns the base type of this type.
	`string``[] Enumerations`	Returns a list of all enumeration facets.
	`int` `FractionDigits`	Returns the value of this facet.
	`int` `Length`	Returns the value of this facet.
	`string` `LocalName`	Returns the local name of the type.
	`string` `MaxExclusive`	Returns the value of this facet.
	`string` `MaxInclusive`	Returns the value of this facet.
	`int` `MaxLength`	Returns the value of this facet.
	`string` `MinExclusive`	Returns the value of this facet.
	`string` `MinInclusive`	Returns the value of this facet.
	`int` `MinLength`	Returns the value of this facet.

	Name	Description
	`string NamespaceURI`	Returns the namespace URI of the type.
	`string[] Patterns`	Returns the pattern facets, or null if no patterns are specified.
	`XmlQualifiedName QualifiedName`	Returns the qualified name of this type.
	`int TotalDigits`	Returns the value of this facet.
	`WhitespaceType Whitespace`	Returns the whitespace normalization facet.

16.8.8 [YourSchema].[Doc]

When code is generated from an XML Schema, the generated code provides a document class with the same name as the schema. This class contains all possible root elements as members, as well as the members listed below. Note that, in the method names below, "Doc" stands for the name of the generated document class itself.

Methods

	Name	Description
	`static Doc CreateDocument()`	Creates a new, empty XML document.
	`static Doc CreateDocument(string encoding)`	Creates a new, empty XML document, with encoding of type "encoding".
	`static void DeclareAllNamespacesFromSchema(Altova.Xml.ElementType node)`	Declares all namespaces from the XML Schema on the element supplied as argument (typically, the XML root element). Calling this method is useful if your schema has multiple namespace declarations, each mapped to a prefix, and you would like to declare all of them on the element supplied as argument.
	`static Doc LoadFromBinary(byte[] binary)`	Loads an XML document from a byte array.
	`static Doc LoadFromFile(string filename)`	Loads an XML document from a file.
	`static Doc LoadFromString(string xmlstring)`	Loads an XML document from a string.
	`byte[] SaveToBinary(bool prettyPrint)`	Saves an XML document to a byte array, with optional "pretty-print" formatting.
	`byte[] SaveToBinary(bool prettyPrint, string encoding)`	Saves an XML document to a byte array, with optional "pretty-print" formatting, with the specified encoding.

(C) 2015-2021 Altova GmbH

	Name	Description
	`byte[] SaveToBinary(bool prettyPrint, string encoding, bool bBigEndian, bool bBOM)`	Saves an XML document to a byte array, with optional "pretty-print" formatting, with the specified encoding, byte order, and BOM (Byte Order Mark).
	`void SaveToFile(string fileName, bool prettyPrint)`	Saves an XML document to a file, with optional "pretty-print" formatting.
	`void SaveToFile(string fileName, bool prettyPrint, bool omitXmlDecl)`	Saves an XML document to a file, with optional "pretty-print" formatting. When `omitXmlDecl` is true, the XML declaration will not be written.
	`void SaveToFile(string fileName, bool prettyPrint, bool omitXmlDecl, string encoding)`	Saves an XML document to a file, with optional "pretty-print" formatting, with the specified encoding. When `omitXmlDecl` is true, the XML declaration will not be written.
	`void SaveToFile(string fileName, bool prettyPrint, string encoding, string lineend)`	Saves an XML document to a file, with optional "pretty-print" formatting, with the specified encoding, and line ending character(s).
	`void SaveToFile(string fileName, bool prettyPrint, bool omitXmlDecl, string encoding, string lineend)`	Saves an XML document to a file, with optional "pretty-print" formatting, with the specified encoding, and line ending character(s). When `omitXmlDecl` is true, the XML declaration will not be written.
	`void SaveToFile(string fileName, bool prettyPrint, bool omitXmlDecl, string encoding, bool bBigEndian, bool bBOM, string lineend)`	Saves an XML document to a file, with optional "pretty-print" formatting, with the specified encoding, byte order, BOM (Byte Order Mark), and line ending character(s). When `omitXmlDecl` is true, the XML declaration will not be written.
	`void SaveToFileWithLineEnd(string fileName, bool prettyPrint, bool omitXmlDecl, string lineend)`	Saves an XML document to a file, with optional "pretty-print" formatting, and line ending character(s). When `omitXmlDecl` is true, the XML declaration will not be written.
	`string SaveToString(bool prettyPrint)`	Saves an XML document to a file, with optional "pretty-print" formatting.
	`string SaveToString(bool prettyPrint, bool omitXmlDecl)`	Saves an XML document to a file, with optional "pretty-print" formatting. When `omitXmlDecl` is true, the XML declaration will not be written.
	`void SetDTDLocation(string dtdLocation)`	Adds a DOCTYPE declaration with the specified system ID. A root element must already exist.
	`void SetSchemaLocation(string schemaLocation)`	Adds an xsi:schemaLocation or xsi:noNamespaceSchemaLocation attribute to the root element. A root element must already exist.

16.8.9 [YourSchema].[ElementType]

This class provides methods for manipulating XML elements from your schema. Methods of this class can be called on elements, not on the XML document itself. Note that, in order to call methods of this class, you don't need to instantiate the class directly. Any element created using the **`Append()`** or **`AppendWithPrefix()`** methods is of `[ElementType]` type.

Methods

	Name	Description

	`void DeclareNamespace(string prefix, string nsURI)`	This method takes two arguments that are both of string type: the prefix and the namespace URI that you want to use. The prefix supplied as argument will be mapped to the namespace URI value supplied as argument. If the prefix supplied as argument is empty, the method creates or overrides the default namespace declaration in the element.
		
		For example, let's assume that the XML document has an XML element called "purchase". If you call
		
		`purchase.DeclareNamespace("ord", "http://OrderTypes");`
		
		then the XML document becomes
		
		`<purchase xmlns:ord="http://OrderTypes" />`
		
		Another example, if you call:
		
		`purchase.DeclareNamespace("", "http://OrderTypes");`
		
		then the XML document becomes
		
		`<purchase xmlns="http://OrderTypes" />`
		
		Note: The declared namespace is used when appending subsequent child elements or attributes, according to the following rules:
		
		1. If the child namespace is the default, then use empty prefix.
		2. If the child namespace is equal to the parent one, then use the parent prefix.
		3. Otherwise, search for nearest prefix from parent to top, using the lookup algorithm described in section "B.2: Namespace Prefix Lookup" at https://www.w3.org/TR/2002/WD-DOM-Level-3-Core-20021022/namespaces-algorithms.html.

(C) 2015-2021 Altova GmbH

	Name	Description
		4. If there is no prefix for element namespace found, then use empty prefix.

16.8.10 [YourSchemaType].MemberAttribute

When code is generated from an XML schema, a class is created for each member attribute of a type. In the descriptions below, "AttributeType" stands for the type of the member attribute itself.

Methods

	Name	Description
	`bool Exists()`	Returns true if the attribute exists.
	`void Remove()`	Removes the attribute from its parent element.

Properties

	Name	Description
	`int` `EnumerationValue`	Generated for enumeration types only. Sets or gets the attribute value using one of the constants generated for the possible values. Returns **Invalid** if the value does not match any of the enumerated values in the schema.
	`Altova.Xml.Meta.Attribute Info`	Returns an object for querying schema information (see `Altova.Xml.Meta.Attribute`).
	`AttributeType Value`	Sets or gets the attribute value.

16.8.11 [YourSchemaType].MemberElement

When code is generated from an XML schema, a class with the following members is created for each member element of a type. The class implements the standard `System.Collections.IEnumerable` interface, so it can be used with the `foreach` statement.

In the descriptions below, "MemberType" stands for the type of the member element itself.

Methods

	Name	Description
	`MemberType Append()`	Creates a new element and appends it to its parent.
	`MemberType AppendWithPrefix(string prefix)`	Creates a new element having the prefix supplied as argument, and appends it to its parent. For an example, see Example: Purchase Order.

	Name	Description
	`MemberType At(int index)`	Returns the member element specified by the index.
	`System.Collections.IEnumerator GetEnumerator()`	Returns an object for iterating instances of the member element.
	`void Remove()`	Deletes all occurrences of the element from its parent.
	`void RemoveAt(int index)`	Deletes the occurrence of the element specified by the index.

Properties

	Name	Description
	`int Count`	Returns the count of elements.
	`int EnumerationValue`	Generated for enumeration types only. Sets or gets the element value using one of the constants generated for the possible values. Returns **Invalid** if the value does not match any of the enumerated values in the schema.
	`bool Exists`	Returns true if at least one element exists.
	`MemberType First`	Returns the first instance of the member element.
	`Altova.Xml.Meta.Element Info`	Returns an object for querying schema information (see `Altova.Xml.Meta.Element`).
	`MemberType Last`	Returns the last instance of the member element.
	`MemberType this[int index]`	Returns the member element specified by the index.
	`MemberType Value`	Sets or gets the element content (only generated if element can have mixed or simple content).

(C) 2015-2021 Altova GmbH

16.9 Reference to Generated Classes (Java)

This chapter includes a description of Java classes generated with MapForce from a DTD or XML schema (see Generating Code from XML Schemas or DTDs). You can integrate these classes into your code to read, modify, and write XML documents.

Note: The generated code does include other supporting classes, which are not listed here and are subject to modification.

16.9.1 com.altova.types.DateTime

This class enables you to process XML attributes or elements that have date and time types, such as `xs:dateTime`.

Constructors

	Name	Description
C	`public DateTime()`	Initializes a new instance of the `DateTime` class to an empty value.
C	`public DateTime(DateTime newvalue)`	Initializes a new instance of the `DateTime` class to the `DateTime` value supplied as argument.
C	`public DateTime(int newyear, int newmonth, int newday, int newhour, int newminute, int newsecond, double newpartsecond, int newoffsetTZ)`	Initializes a new instance of the `DateTime` class to the year, month, day, hour, minute, second, the fractional part of the second, and timezone supplied as arguments. The fractional part of the second newpartsecond must be between 0 and 1. The timezone offset newoffsetTZ can be either positive or negative and is expressed in minutes.
C	`public DateTime(int newyear, int newmonth, int newday, int newhour, int newminute, int newsecond, double newpartsecond)`	Initializes a new instance of the `DateTime` class to the year, month, day, hour, minute, second, and the fractional part of a second supplied as arguments.
C	`public DateTime(int newyear, int newmonth, int newday)`	Initializes a new instance of the `DateTime` class to the year, month, and day supplied as arguments.
C	`public DateTime(Calendar newvalue)`	Initializes a new instance of the `DateTime` class to the `java.util.Calendar` value supplied as argument.

Methods

	Name	Description
S	`static DateTime now()`	Returns the current time as a `DateTime` object.
S	`static DateTime parse(String s)`	Returns a `DateTime` object parsed from the string value supplied as argument. For example, the following sample string values

	Name	Description

		would be converted successfully to a `DateTime` object:
		
		`2015-11-24T12:54:47.969+01:00`
		`2015-11-24T12:54:47`
		`2015-11-24`

●	**int** `getDay()`	Returns the day of the current `DateTime` instance.
●	**int** `getHour()`	Returns the hour of the current `DateTime` instance.
●	**int** `getMillisecond()`	Returns the millisecond of the current `DateTime` instance, as an integer value.
●	**int** `getMinute()`	Returns the minute of the current `DateTime` instance.
●	**int** `getMonth()`	Returns the month of the current `DateTime` instance.
●	**double** `getPartSecond()`	Returns the fractional part of the second of the current `DateTime` instance, as a **double** value. The return value is greater than zero and smaller than one, for example: `0.313`
●	**int** `getSecond()`	Returns the second of the current `DateTime` instance.
●	**int** `getTimezoneOffset()`	Returns the timezone offset, in minutes, of the current `DateTime` instance. For example, the timezone "UTC-01:00" would be returned as: `-60`
●	`Calendar getValue()`	Returns the current `DateTime` instance as a `java.util.Calendar` value.
●	**int** `getWeekday()`	Returns the day in week of the current `DateTime` instance. Values range from 0 through 6, where 0 is Monday (ISO-8601).
●	**int** `getYear()`	Returns the year of the current `DateTime` instance.
●	**int** `hasTimezone()`	Returns information about the timezone of the current `DateTime` instance. Possible return values are: `CalendarBase.TZ_MISSING` — A timezone offset is not defined. `CalendarBase.TZ_UTC` — The timezone is UTC. `CalendarBase.TZ_OFFSET` — A timezone offset has been defined.
●	**void** `setDay( int nDay )`	Sets the day of the current `DateTime` instance to the value supplied as argument.
●	**void** `setHasTimezone( int nHasTZ )`	Sets the timezone information of the current `DateTime` instance to the value supplied as argument. This method can be used to

(C) 2015-2021 Altova GmbH

	Name	Description
		strip the timezone information or set the timezone to UTC (Coordinated Universal Time). Valid values for the nHasTZ argument: `CalendarBase.TZ_MISSING` Set the timezone offset to undefined. `CalendarBase.TZ_UTC` Set the timezone to UTC. `CalendarBase.TZ_OFFSET` If the current object has a timezone offset, leave it unchanged.
●	`void setHour( int nHour )`	Sets the hour of the current `DateTime` instance to the value supplied as argument.
●	`void setMinute( int nMinute )`	Sets the minute of the current `DateTime` instance to the value supplied as argument.
●	`void setMonth( int nMonth )`	Sets the month of the current `DateTime` instance to the value supplied as argument.
●	`void setPartSecond( double nPartSecond )`	Sets the fractional part of the second of the current `DateTime` instance to the value supplied as argument.
●	`void setSecond( int nSecond )`	Sets the second of the current `DateTime` instance to the value supplied as argument.
●	`void setTimezoneOffset( int nOffsetTZ )`	Sets the timezone offset of the current `DateTime` instance to the value supplied as argument. The value nOffsetTZ must be an integer (positive or negative) and must be expressed in minutes.
●	`void setYear( int nYear )`	Sets the year of the current `DateTime` instance to the value supplied as argument.
●	`String toString()`	Returns the string representation of the current `DateTime` instance, for example: `2015-11-24T15:50:56.968+01:00`

Examples

Before using the following code listings in your program, ensure the Altova types are imported:

```
import com.altova.types.*;
```

The following code listing illustrates various ways to create `DateTime` objects:

```
protected static void DateTimeExample1()
{
   // Initialize a new instance of the DateTime class to the current time
   DateTime dt = new DateTime(DateTime.now());
```

```
   System.out.println("DateTime created from current date and time: " + dt.toString());

   // Initialize a new instance of the DateTime class by supplying the parts
   DateTime dt1 = new DateTime(2015, 11, 23, 14, 30, 24, .459);
   System.out.println("DateTime from parts (no timezone): " + dt1.toString());

   // Initialize a new instance of the DateTime class by supplying the parts
   DateTime dt2 = new DateTime(2015, 11, 24, 14, 30, 24, .459, -60);
   System.out.println("DateTime from parts (with negative timezone): " +
dt2.toString());

   // Initialize a new instance of the DateTime class by parsing a string value
 DateTime dt3 = DateTime.parse("2015-11-24T12:54:47.969+01:00");
   System.out.println("DateTime parsed from string: " + dt3.toString());
}
```

The following code listing illustrates getting values from `DateTime` objects:

```
protected static void DateTimeExample2()
   {
      // Initialize a new instance of the DateTime class to the current time
      DateTime dt = new DateTime(DateTime.now());

      // Output the formatted year, month, and day of this DateTime instance
      String str1 = String.format("Year: %d; Month: %d; Day: %d;", dt.getYear(),
dt.getMonth(), dt.getDay());
      System.out.println(str1);

      // Output the formatted hour, minute, and second of this DateTime instance
      String str2 = String.format("Hour: %d; Minute: %d; Second: %d;", dt.getHour(),
dt.getMinute(), dt.getSecond());
      System.out.println(str2);

      // Return the timezone (in minutes) of this DateTime instance
      System.out.println("Timezone: " + dt.getTimezoneOffset());

      // Get the DateTime as a java.util.Calendar value
      java.util.Calendar dt_java = dt.getValue();
      System.out.println("" + dt_java.toString());

      // Return the day of week of this DateTime instance
      System.out.println("Weekday: " + dt.getWeekday());

      // Check whether the DateTime instance has a timezone defined
      switch(dt.hasTimezone())
      {
         case CalendarBase.TZ_MISSING:
            System.out.println("No timezone.");
            break;
         case CalendarBase.TZ_UTC:
            System.out.println("The timezone is UTC.");
            break;
         case CalendarBase.TZ_OFFSET:
            System.out.println("This object has a timezone.");
            break;
         default:
            System.out.println("Unable to determine whether a timezone is defined.");
            break;
```


(C) 2015-2021 Altova GmbH

```
      }
   }
```

The following code listing illustrates changing the timezone offset of a `DateTime` object:

```
protected static void DateTimeExample3()
{
   // Create a new DateTime object with timezone -0100 UTC
   DateTime dt = new DateTime(2015, 11, 24, 14, 30, 24, .459, -60);
   // Output the value before the change
   System.out.println("Before: " + dt.toString());
   // Change the offset to +0100 UTC
   dt.setTimezoneOffset(60);
   // Output the value after the change
   System.out.println("After:  " + dt.toString());
}
```

16.9.2 com.altova.types.Duration

This class enables you to process XML attributes or elements of type `xs:duration`.

Constructors

	Name	Description
C	`Duration(Duration newvalue)`	Initializes a new instance of the `Duration` class to the `Duration` object supplied as argument.
C	`Duration(int newyear, int newmonth, int newday, int newhour, int newminute, int newsecond, double newpartsecond, boolean newisnegative)`	Initializes a new instance of the `Duration` class to a duration built from parts supplied as arguments.

Methods

	Name	Description
S	`static Duration getFromDayTime( int newday, int newhour, int newminute, int newsecond, double newpartsecond )`	Returns a `Duration` object created from the number of days, hours, minutes, seconds, and fractional second parts supplied as argument.
S	`static Duration getFromYearMonth( int newyear, int newmonth )`	Returns a `Duration` object created from the number of years and months supplied as argument.
S	`static Duration parse( String s )`	Returns a `Duration` object created from the string supplied as argument. For example, the string -P1Y1M1DT1H1M1.333S can be used to create a negative duration of one year, one month,

	Name	Description
		one day, one hour, one minute, one second, and 0.333 fractional parts of a second. To create a negative duration, append the minus sign (-) to the string.
S	**static** Duration parse(String s, ParseType pt)	Returns a `Duration` object created from the string supplied as argument, using a specific parse format. The parse format can be any of the following: **ParseType.DAYTIME** — May be used when the string **s** consists of any of the following: days, hours, minutes, seconds, fractional second parts, for example -P4DT4H4M4.774S. **ParseType.DURATION** — May be used when the string **s** consists of any of the following: years, months, days, hours, minutes, seconds, fractional second parts, for example P1Y1M1DT1H1M1.333S. **ParseType.YEARMONTH** — May be used when the string **s** consists of any of the following: years, months. For example: P3Y2M.
	int getDay()	Returns the number of days in the current `Duration` instance.
	long getDayTimeValue()	Returns the day and time value (in milliseconds) of the current `Duration` instance. Years and months are ignored.
	int getHour()	Returns the number of hours in the current `Duration` instance.
	int getMillisecond()	Returns the number of milliseconds in the current `Duration` instance.
	int getMinute()	Returns the number of minutes in the current `Duration` instance.
	int getMonth()	Returns the number of months in the current `Duration` instance.
	double getPartSecond()	Returns the number of fractional second parts in the current `Duration` instance.
	int getSecond()	Returns the number of seconds in the current `Duration` instance.
	int getYear()	Returns the number of years in the current `Duration` instance.
	int getYearMonthValue()	Returns the year and month value (in months) of the current `Duration` instance. Days, hours, seconds, and milliseconds are ignored.

(C) 2015-2021 Altova GmbH

	Name	Description
	`boolean isNegative()`	Returns Boolean **true** if the current `Duration` instance is negative.
	`void setDayTimeValue(long l)`	Sets the duration to the number of milliseconds supplied as argument, affecting only the day and time part of the duration.
	`void setNegative( boolean isnegative )`	Converts the current `Duration` instance to a negative duration.
	`void setYearMonthValue(int l)`	Sets the duration to the number of months supplied as argument. Only the years and months part of the duration is affected.
	`String toString()`	Returns the string representation of the current `Duration` instance, for example: `-P4DT4H4M4.774S`
	`String toYearMonthString()`	Returns the string representation of the YearMonth part of the current `Duration` instance, for example: `P1Y2M`

Examples

Before using the following code listings in your program, ensure the Altova types are imported:

```
import com.altova.types.*;
import com.altova.types.Duration.ParseType;
```

The following code listing illustrates various ways to create `Duration` objects:

```
protected static void ExampleDuration()
{
   // Create a negative duration of 1 year, 1 month, 1 day, 1 hour, 1 minute, 1
second,
   // and 0.333 fractional second parts
   Duration dr = new Duration(1, 1, 1, 1, 1, 1, .333, true);

   // Create a duration from an existing Duration object
   Duration dr1 = new Duration(dr);

   // Create a duration of 4 days, 4 hours, 4 minutes, 4 seconds, .774 fractional
second parts
   Duration dr2 = Duration.getFromDayTime(4, 4, 4, 4, .774);

   // Create a duration of 3 years and 2 months
   Duration dr3 = Duration.getFromYearMonth(3, 2);

   // Create a duration from a string
   Duration dr4 = Duration.parse("-P4DT4H4M4.774S");

   // Create a duration from a string, using specific parse formats
   Duration dr5 = Duration.parse("-P1Y1M1DT1H1M1.333S", ParseType.DURATION);
```

```
   Duration dr6 = Duration.parse("P3Y2M", ParseType.YEARMONTH);
   Duration dr7 = Duration.parse("-P4DT4H4M4.774S", ParseType.DAYTIME);
}
```

The following code listing illustrates getting and setting the value of `Duration` objects:

```
protected static void DurationExample2()
{
   // Create a duration of 1 year, 2 month, 3 days, 4 hours, 5 minutes, 6 seconds,
       // and 333 milliseconds
   Duration dr = new Duration(1, 2, 3, 4, 5, 6, .333, false);
   // Output the number of days in this duration
   System.out.println(dr.getDay());

   // Create a positive duration of one year and 333 milliseconds
   Duration dr1 = new Duration(1, 0, 0, 0, 0, 0, .333, false);
   // Output the day and time value in milliseconds
   System.out.println(dr1.getDayTimeValue());

   // Create a positive duration of 1 year, 1 month, 1 day, 1 hour, 1 minute, 1
second,
       // and 333 milliseconds
   Duration dr2 = new Duration(1, 1, 1, 1, 1, 1, .333, false);
   // Output the year and month value in months
   System.out.println(dr2.getYearMonthValue());

   // Create a positive duration of 1 year and 1 month
   Duration dr3 = new Duration(1, 1, 0, 0, 0, 0, 0, false);
   // Output the value
   System.out.println("The duration is now: " + dr3.toString());
   // Set the DayTime part of duration to 1000 milliseconds
   dr3.setDayTimeValue(1000);
   // Output the value
   System.out.println("The duration is now: " + dr3.toString());
   // Set the YearMonth part of duration to 1 month
   dr3.setYearMonthValue(1);
   // Output the value
   System.out.println("The duration is now: " + dr3.toString());
   // Output the year and month part of the duration
   System.out.println("The YearMonth part of the duration is: " +
dr3.toYearMonthString());
}
```

16.9.3 com.altova.xml.meta.Attribute

This class enables you to access schema information about classes generated from attributes. Note that this class is not meant to provide dynamic information about particular instances of an attribute in an XML document. Instead, it enables you to obtain programmatically information about a particular attribute defined in the XML schema.

(C) 2015-2021 Altova GmbH

Methods

	Name	Description
	`SimpleType getDataType()`	Returns the type of the attribute content.
	`String getLocalName()`	Returns the local name of the attribute.
	`String getNamespaceURI()`	Returns the namespace URI of the attribute.
	`boolean isRequired()`	Returns true if the attribute is required.

16.9.4 com.altova.xml.meta.ComplexType

This class enables you to access schema information about classes generated from complex types. Note that this class is not meant to provide dynamic information about particular instances of a complex type in an XML document. Instead, it enables you to obtain programmatically information about a particular complex type defined in the XML schema.

Methods

	Name	Description
	`Attribute findAttribute(String localName, String namespaceURI)`	Finds the attribute with the specified local name and namespace URI.
	`Element findElement(String localName, String namespaceURI)`	Finds the element with the specified local name and namespace URI.
	`Attribute[] GetAttributes()`	Returns a list of all attributes.
	`ComplexType getBaseType()`	Returns the base type of this type.
	`SimpleType getContentType()`	Returns the simple type of the content.
	`Element[] GetElements()`	Returns a list of all elements.
	`String getLocalName()`	Returns the local name of the type.
	`String getNamespaceURI()`	Returns the namespace URI of the type.

16.9.5 com.altova.xml.meta.Element

This class enables you to access information about classes generated from schema elements. Note that this class is not meant to provide dynamic information about particular instances of an element in an XML document. Instead, it enables you to obtain programmatically information about a particular element defined in the XML schema.

Methods

	Name	Description
	`ComplexType getDataType()`	Returns the type of the element. Note that this is always a complex type even if declared as simple in the original schema. Use `getContentType()` of the returned object to get the simple content type.
	`String getLocalName()`	Returns the local name of the element.
	`int` `getMaxOccurs()`	Returns the maxOccurs value defined in the schema.
	`int` `getMinOccurs()`	Returns the minOccurs value defined in the schema.
	`String getNamespaceURI()`	Returns the namespace URI of the element.

16.9.6 com.altova.xml.meta.SimpleType

This class enables you to access schema information about classes generated from simple types. Note that this class is not meant to provide dynamic information about particular instances of simple types in an XML document. Instead, it enables you to obtain programmatically information about a particular simple type defined in the XML schema.

Methods

	Name	Description
	`SimpleType getBaseType()`	Returns the base type of this type.
	`String[] getEnumerations()`	Returns an array of all enumeration facets.
	`int` `getFractionDigits()`	Returns the value of this facet.
	`int` `getLength()`	Returns the value of this facet.
	`String getLocalName()`	Returns the local name of the type.
	`String getMaxExclusive()`	Returns the value of this facet.
	`String getMaxInclusive()`	Returns the value of this facet.
	`int` `getMaxLength()`	Returns the value of this facet.
	`String getMinExclusive()`	Returns the value of this facet.
	`String getMinInclusive()`	Returns the value of this facet.
	`int` `getMinLength()`	Returns the value of this facet.
	`String getNamespaceURI()`	Returns the namespace URI of the type.
	`String[] getPatterns()`	Returns an array of all pattern facets.

(C) 2015-2021 Altova GmbH

	Name	Description
	`int getTotalDigits()`	Returns the value of this facet.

	`int getWhitespace()`	Returns the value of the whitespace facet, which is one of:
		`com.altova.typeinfo.WhitespaceType.Whitespace_Unknown`
		`com.altova.typeinfo.WhitespaceType.Whitespace_Preserve`
		`com.altova.typeinfo.WhitespaceType.Whitespace_Replace`
		`com.altova.typeinfo.WhitespaceType.Whitespace_Collapse`

16.9.7 com.[YourSchema].[Doc]

When code is generated from an XML Schema, the generated code provides a document class with the same name as the schema. This class contains all possible root elements as members, as well as the members listed below. Note that, in the method names below, "Doc" stands for the name of the generated document class itself.

Methods

	Name	Description
S	`static Doc createDocument()`	Creates a new, empty XML document.
S	`static void declareAllNamespacesFromSchema(com.altova.xml.ElementType node)`	Declares all namespaces from the XML Schema on the element supplied as argument (typically, the XML root element). Calling this method is useful if your schema has multiple namespace declarations, each mapped to a prefix, and you would like to declare all of them on the element supplied as argument.
S	`static Doc loadFromBinary(byte[] xml)`	Loads an XML document from a byte array.
S	`static Doc loadFromFile(String fileName)`	Loads an XML document from a file.
S	`static Doc loadFromString(String xml)`	Loads an XML document from a string.
	`byte[] saveToBinary(boolean prettyPrint)`	Saves an XML document to a byte array, with optional "pretty-print" formatting.
	`byte[] saveToBinary(boolean prettyPrint, String encoding)`	Saves an XML document to a byte array, with optional "pretty-print" formatting, with the specified encoding.
	`byte[] saveToBinary(boolean prettyPrint, String encoding, boolean bigEndian, boolean writeBOM)`	Saves an XML document to a byte array, with optional "pretty-print" formatting, with the specified encoding. Byte order and Unicode byte-order mark can be specified for Unicode encodings.

	Name	Description
	`void saveToFile(String fileName, boolean prettyPrint)`	Saves an XML document to a file, with optional "pretty-print" formatting.
	`void saveToFile(String fileName, boolean prettyPrint, boolean omitXmlDecl)`	Saves an XML document to a file, with optional "pretty-print" formatting, with UTF-8 encoding. When `omitXmlDecl` is true, the XML declaration will not be written.
	`void saveToFile(String fileName, boolean prettyPrint, boolean omitXmlDecl, String encoding)`	Saves an XML document to a file, with optional "pretty-print" formatting, with the specified encoding. When `omitXmlDecl` is true, the XML declaration will not be written.
	`void saveToFile(String fileName, boolean prettyPrint, boolean omitXmlDecl, String encoding, boolean bBigEndian, boolean bBOM)`	Saves an XML document to a file, with optional "pretty-print" formatting, with the specified encoding. When `omitXmlDecl` is true, the XML declaration will not be written. Byte order and Unicode byte-order mark can be specified for Unicode encodings.
	`void saveToFile(String fileName, boolean prettyPrint, String encoding)`	Saves an XML document to a file, with optional "pretty-print" formatting, with the specified encoding.
	`void saveToFile(String fileName, boolean prettyPrint, String encoding, boolean bBigEndian, boolean bBOM)`	Saves an XML document to a file, with optional "pretty-print" formatting, with the specified encoding. Byte order and Unicode byte-order mark can be specified for Unicode encodings.
	`String saveToString(boolean prettyPrint)`	Saves an XML document to a string, with optional "pretty-print" formatting.
	`String saveToString(boolean prettyPrint, boolean omitXmlDecl)`	Saves an XML document to a string, with optional "pretty-print" formatting. When `omitXmlDecl` is true, the XML declaration will not be written.
	`void setSchemaLocation(String schemaLocation)`	Adds an `xsi:schemaLocation` or `xsi:noNamespaceSchemaLocation` attribute to the root element. A root element must already exist.

16.9.8 com.[YourSchema].[ElementType]

This class provides methods for manipulating XML elements from your schema. Methods of this class can be called on elements, not on the XML document itself. Note that, in order to call methods of this class, you don't need to instantiate the class directly. Any element created using the **`append()`** or **`appendWithPrefix()`** methods is of [`ElementType`] type.

(C) 2015-2021 Altova GmbH

Methods

	Name	Description

	`void declareNamespace(String prefix, String nsURI)`	This method takes two arguments that are both of string type: the prefix and the namespace URI that you want to use. The prefix supplied as argument will be mapped to the namespace URI value supplied as argument. If the prefix supplied as argument is empty, the method creates or overrides the default namespace declaration in the element.
		
		For example, let's assume that the XML document has an XML element called "purchase". If you call
		
		`purchase.declareNamespace("ord", "http://OrderTypes");`
		
		then the XML document becomes
		
		`<purchase xmlns:ord="http://OrderTypes" />`
		
		Another example, if you call:
		
		`purchase.declareNamespace("", "http://OrderTypes");`
		
		then the XML document becomes
		
		`<purchase xmlns="http://OrderTypes" />`
		
		Note: The declared namespace is used when appending subsequent child elements or attributes, according to the following rules:
		
		1. If the child namespace is the default, then use empty prefix.
		2. If the child namespace is equal to the parent one, then use the parent prefix.
		3. Otherwise, search for nearest prefix from parent to top, using the lookup algorithm described in section "B.2: Namespace Prefix Lookup" at https://www.w3.org/TR/2002/WD-DOM-Level-3-Core-20021022/namespaces-algorithms.html.
		4. If there is no prefix for element namespace found, then use empty prefix.

16.9.9 com.[YourSchema].[YourSchemaType].MemberAttribute

When code is generated from an XML schema, a class is created for each member attribute of a type. In the descriptions below, "AttributeType" stands for the type of the member attribute itself.

Methods

	Name	Description
	`boolean exists()`	Returns true if the attribute exists.
	`int getEnumerationValue()`	Generated for enumeration types only. Returns one of the constants generated for the possible values, or **Invalid** if the value does not match any of the enumerated values in the schema.
	`com.altova.xml.meta.Attribute getInfo()`	Returns an object for querying schema information (see `com.altova.xml.meta.Attribute`).
	`AttributeType getValue()`	Gets the attribute value.
	`void remove()`	Removes the attribute from its parent element.
	`void setEnumerationValue(int)`	Generated for enumeration types only. Pass one of the constants generated for the possible values to this method to set the value.
	`void setValue(AttributeType value)`	Sets the attribute value.

16.9.10 com.[YourSchema].[YourSchemaType].MemberElement

When code is generated from an XML schema, a class with the following members is created for each member element of a type. In the descriptions below, "MemberType" stands for the type of the member element itself.

Methods

	Name	Description
	`MemberType append()`	Creates a new element and appends it to its parent.
	`MemberType appendWithPrefix(String prefix)`	Creates a new element having the prefix supplied as argument, and appends it to its parent. For an example, see Example: Purchase Order.
	`MemberType at(int index)`	Returns the instance of the member element at the specified index.
	`int count()`	Returns the count of elements.

(C) 2015-2021 Altova GmbH

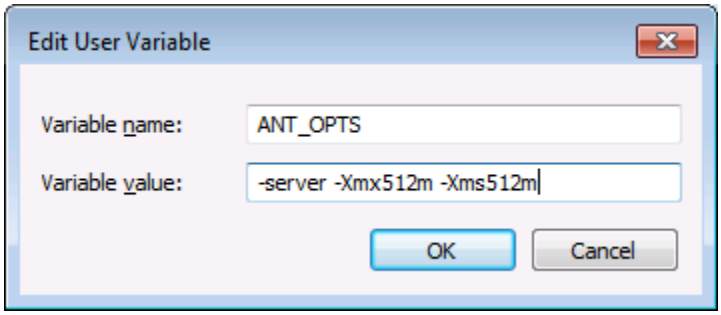
	Name	Description
	`boolean exists()`	Returns true if at least one element exists.
	`MemberType first()`	Returns the first instance of the member element.
	`int getEnumerationValue()`	Generated for enumeration types only. Returns one of the constants generated for the possible values, or **Invalid** if the value does not match any of the enumerated values in the schema.
	`com.altova.xml.meta.Element getInfo()`	Returns an object for querying schema information (see `com.altova.xml.meta.Element`).
	`MemberType getValue()`	Gets the element content (only generated if element can have simple or mixed content).
	`java.util.Iterator iterator()`	Returns an object for iterating instances of the member element.
	`MemberType last()`	Returns the last instance of the member element.
	`void remove()`	Deletes all occurrences of the element from its parent.
	`void removeAt(int index)`	Deletes the occurrence of the element specified by the index.
	`void setEnumerationValue(int index)`	Generated for enumeration types only. Pass one of the constants generated for the possible values to this method to set the value.
	`void setValue(MemberType value)`	Sets the element content (only generated if element can have simple or mixed content).

16.10 Code Generation Tips

Resolving "Out of memory" exceptions during Java compilation

Complex mappings with large schemas can produce a large amount of code, which might cause a `java.lang.OutofMemory` exception during compilation using Ant. To rectify this:

- Add the environment variable `ANT_OPTS`, which sets specific Ant options such as the memory to be allocated to the compiler, and set its value as shown below.



- To make sure that the compiler and the generated code run in the same process as Ant, change the `fork` attribute, in **build.xml**, to `false`.

You may need to customize the values depending on the amount of memory in your machine and the size of the project you are working with. For more details, see your Java VM documentation.

When running the `ant jar` command, you may get an error message similar to "`[...] archive contains more than 65535 entities`". To prevent this, it is recommended that you use Ant 1.9 or later, and, in the **build.xml** file, add `zip64mode="as-needed"` to the `<jar>` element.

Reserving method names

When customizing code generation using the supplied SPL files, it might be necessary to reserve names to avoid collisions with other symbols. To do this:

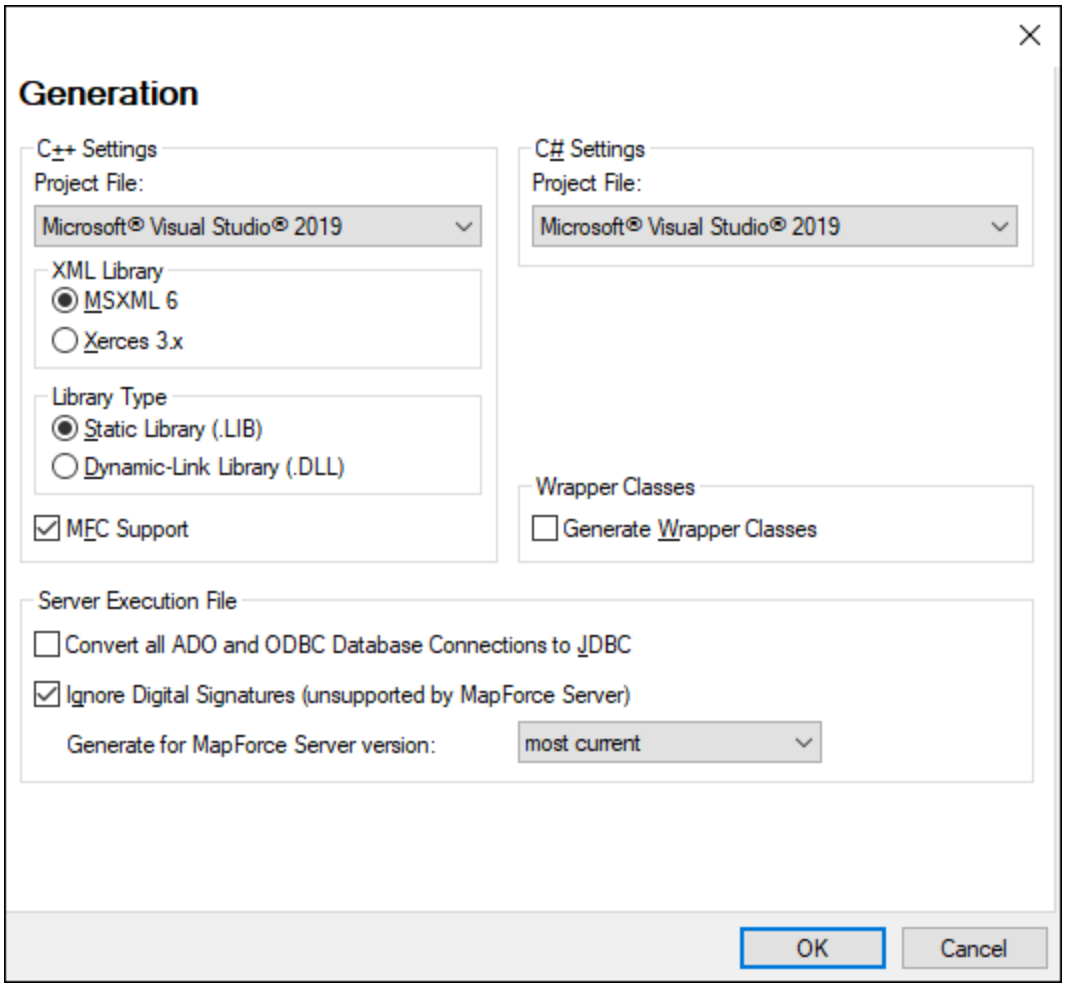
1. Navigate to the program installation directory, for example, `C:\Program Files\Altova\MapForce2021`.
2. In the **spl** subdirectory, locate the directory corresponding to the programming language, for example, **..\spl\java**.
3. Open the **settings.spl** file and insert a new line into the "reserve" section, for example, `reserve "myReservedWord"`.
4. Regenerate the program code.

(C) 2015-2021 Altova GmbH

16.11 Code Generator Options

To view or change the MapForce settings applicable to code generation:

- On the **Tools** menu, click **Options**, and then click **Generation**.



The available settings are as follows.

C++ Settings	Defines the specific compiler settings for the C++ environment, namely: • The Visual Studio version (2010, 2013, 2015, 2017, 2019) • The XML library (MSXML, Xerces 3.x) • Whether static or dynamic libraries must be generated • Whether code must be generated with or without MFC support
C# Settings	Select the option **Microsoft .NET Core 3.1** or **Microsoft .NET 5.0** to generate a Visual Studio solution targeting the respective platforms.

	If you need to target the **.NET Framework** platform for a specific Visual Studio version, select any of the **Microsoft Visual Studio 2010-2019** options—in this case, the generated solution will target the .NET Framework version corresponding to the respective Visual Studio version.
Wrapper Classes	Allows you to generate wrapper classes for XML schemas, see Generating Code from XML Schemas or DTDs. These wrapper classes can be used by custom code that includes the code generated by MapForce.
Server Execution File	These options are applicable when you compile mappings to MapForce Server execution files. They do not affect generation of C#, C++, or Java code. For more information, see Compiling Mappings to MapForce Server Execution Files.

(C) 2015-2021 Altova GmbH

16.12 SPL Reference

This section gives an overview of SPL (Spy Programming Language), the code generator's template language.

It is assumed that you have prior programming experience, and are familiar with operators, functions, variables and classes, as well as the basics of object-oriented programming - which is used heavily in SPL.

The templates used by MapForce are supplied in the **...\MapForce\spl** folder. You can use these files as an aid to help you in developing your own templates.

How code generator works

Inputs to the code generator are the template files (.spl) and the object model provided by MapForce. The template files contain SPL instructions for creating files, reading information from the object model and performing calculations, interspersed with literal code fragments in the target programming language.

The template file is interpreted by the code generator and outputs **.cpp, .java, .cs** source code files, project files, or any other type of file depending on the template. The source code can then be compiled into an executable file that accesses XML data described by the schema file.

SPL files have access to a wide variety of information that is collated from the source schemas. Note that an SPL file is not tied to a specific schema, but allows access to all schemas. Make sure you write your SPL files generically and avoid structures which apply to specific schemas.

Example: Creating a new file in SPL

This is a very basic SPL file. It creates a file named **test.cpp,** and places the include statement within it. The close command completes the template.

```
[create "test.cpp"]
#include "stdafx.h"
[close]
```

16.12.1 Basic SPL structure

An SPL file contains literal text to output, interspersed with code generator instructions.

Code generator instructions are enclosed in square brackets '**[**' and '**]**'. Multiple statements can be included in a bracket pair. Additional statements have to be separated by a new line or a colon '**:**'.

Valid examples are:

```
[$x = 42
$x = $x + 1]
```

or

```
[$x = 42: $x = $x + 1]
```

Adding text to files

Text not enclosed by **[** and **]**, is written directly to the current output file. If there is no current output file, the text is ignored (see Using files how to create an output file).

To output literal square brackets, escape them with a backslash: **\[** and **\]**; to output a backslash use ****.

Comments

Comments inside an instruction block always begin with a **'** character, and terminate on the next line, or at a block close character **]**.

16.12.2 Declarations

The following statements are evaluated while parsing the SPL template file. They are **not** affected by flow control statements like conditions, loops or subroutines, and are always evaluated exactly once.

These keywords, like all keywords in SPL, are not case sensitive.

Remember that all of these declarations must be inside a block delimited by square brackets.

map ... to ...

```
map mapname key to value [, key to value ]...
```

This statement adds information to a map. See below for specific uses.

```
map schemanativetype schematype to typespec
```

The specified built-in XML Schema type will be mapped to the specified native type or class, using the specified formatter. This setting applies only to code generation for version 2007r3 and higher. Typespec is a native type or class name, followed by a comma, followed by the formatter class instance.

Example:

```
map schemanativetype "double" to "double,Altova::DoubleFormatter"
```

map type ... to ...

```
map type schematype to classname
```

The specified built-in XML Schema type will be mapped to the specified class. This setting applies only to code generation for version 2007 or lower.

(C) 2015-2021 Altova GmbH

Example:

```
map type "float" to "CSchemaFloat"
```

default ... is ...

```
default setting is value
```

This statement allows you to affect how class and member names are derived from the XML Schema. Note that the setting names are case sensitive.

Example:

```
default "InvalidCharReplacement" is "_"
```

Setting name	Explanation
ValidFirstCharSet	Allowed characters for starting an identifier
ValidCharSet	Allowed characters for other characters in an identifier
InvalidCharReplacement	The character that will replace all characters in names that are not in the ValidCharSet
AnonTypePrefix	Prefix for names of anonymous types*
AnonTypeSuffix	Suffix for names of anonymous types*
ClassNamePrefix	Prefix for generated class names
ClassNameSuffix	Suffix for generated class names
EnumerationPrefix	Prefix for symbolic constants declared for enumeration values
EnumerationUpperCase	"on" to convert the enumeration constant names to upper case
FallbackName	If a name consists only of characters that are not in ValidCharSet, use this one

* Names of anonymous types are built from AnonTypePrefix + element name + AnonTypeSuffix

reserve

```
reserve word
```

Adds the specified word to the list of reserved words. This ensures that it will never be generated as a class or member name.

Example:

```
reserve "while"
```

include

includes the specified file as SPL source. This allows you to split your template into multiple files for easier editing and handling.

```
include filename
```

Example:

```
include "Module.cpp"
```

16.12.3 Variables

Any non-trivial SPL file will require variables. Some variables are predefined by the code generator, and new variables may be created simply by assigning values to them.

The **$** character is used when **declaring** or **using** a variable, a variable name is always prefixed by **$**. Variable names are **case sensitive**.

Variables types:

- integer - also used as boolean, where 0 is false and everything else is true
- string
- object - provided by MapForce
- iterator - see foreach statement

Variable types are declared by first assignment:

```
[$x = 0]
```

x is now an integer.

```
[$x = "teststring"]
```

x is now treated as a string.

Strings

String constants are always enclosed in double quotes, like in the example above. **\n** and **\t** inside double quotes are interpreted as newline and tab, **\"** is a literal double quote, and \\ is a backslash. String constants can also span multiple lines.

String concatenation uses the **&** character:

(C) 2015-2021 Altova GmbH

```
[$BasePath = $outputpath & "/" & $JavaPackageDir]
```

Objects

Objects represent the information contained in the XML schemas, database structures, text files and mappings. Objects have **properties**, which can be accessed using the **.** operator. It is not possible to create new objects in SPL (they are predefined by the code generator, derived from the input mapping), but it is possible to assign objects to variables.

Example:

```
class [=$class.Name]
```

This example outputs the word "class", followed by a space and the value of the **Name** property of the **$class** object.

16.12.4 Predefined variables

After a Schema file is analyzed by the code generator, the objects in the table below exist in the Template Engine.

Name	Type	Description
$schematype	integer	1 for DTD, 2 for XML Schema
$TheLibrary	Library	The library derived from the XML Schema or DTD
$module	string	Name of the source Schema without extension
$outputpath	string	The output path specified by the user, or the default output path

For C++ generation only:

Name	Type	Description
$domtype	integer	1 for MSXML, 2 for Xerces
$libtype	integer	1 for static LIB, 2 for DLL
$mfc	boolean	True if MFC support is enabled
$VSVersion	integer	Specifies the Visual Studio version. Valid values: **0** No Visual Studio project **2010** Visual Studio 2010 **2013** Visual Studio 2013 **2015** Visual Studio 2015

Name	Type	Description
		2017 Visual Studio 2017 **2019** Visual Studio 2019

For C# generation only:

Name	Type	Description
$VSVersion	integer	Specifies the Visual Studio version. Valid values: **0** No Visual Studio project **2010** Visual Studio 2010 **2013** Visual Studio 2013 **2015** Visual Studio 2015 **2017** Visual Studio 2017 **2019** Visual Studio 2019

16.12.5 Creating output files

These statements are used to create output files from the code generation. Remember that all of these statements must be inside a block delimited by square brackets.

create

```
create filename
```

creates a new file. The file has to be closed with the **close** statement. All following output is written to the specified file.

Example:

(C) 2015-2021 Altova GmbH

```
[create $outputpath & "/" & $JavaPackageDir & "/" & $application.Name & ".java"]
package [=$JavaPackageName];

public class [=$application.Name]Application {
...
}
[close]
```

close

closes the current output file.

```
=$variable
```

writes the value of the specified variable to the current output file.

Example:

```
[$x = 20+3]
The result of your calculation is [=$x] - so have a nice day!
```

The file output will be:

```
The result of your calculation is 23 - so have a nice day!
```

write

```
write string
```

writes the string to the current output file.

Example:

```
[write "C" & $name]
```

This can also be written as:

```
C[=$name]
```

filecopy ... to ...

```
filecopy source to target
```

copies the source file to the target file, without any interpretation.

Example:

```
filecopy "java/mapforce/mapforce.png" to $outputpath & "/" & $JavaPackageDir &
"/mapforce.png"
```

16.12.6 Operators

Operators in SPL work like in most other programming languages.

List of SPL operators in descending precedence order:

.	Access object property
( )	Expression grouping
true	boolean constant "true"
false	boolean constant "false"
&	String concatenation
-	Sign for negative number
not	Logical negation
*	Multiply
/	Divide
%	Modulo
+	Add
-	Subtract
<=	Less than or equal
<	Less than
>=	Greater than or equal
>	Greater than
=	Equal
<>	Not equal
and	Logical conjunction (with short circuit evaluation)
or	Logical disjunction (with short circuit evaluation)
=	Assignment

16.12.7 Conditions

SPL allows you to use standard "if" statements. The syntax is as follows:

```
if condition
        statements
else
        statements
```


(C) 2015-2021 Altova GmbH

```
endif
```

or, without else:

```
if condition
	statements
endif
```

Note: There are no round brackets enclosing the condition.

As in any other programming language, conditions are constructed with logical and comparison operators.

Example:

```
[if $namespace.ContainsPublicClasses and $namespace.Prefix <> ""]
	whatever you want ['inserts whatever you want, in the resulting file]
[endif]
```

Switch

SPL also contains a multiple choice statement.

Syntax:

```
switch $variable
	case X:
		statements
	case Y:
	case Z:
		statements
	default:
		statements
endswitch
```

The case labels must be constants or variables.

The switch statement in SPL does not fall through the cases (as in C), so there is no need for a "break" statement.

16.12.8 Collections and foreach

Collections and iterators

A collection contains multiple objects - like a ordinary array. Iterators solve the problem of storing and incrementing array indexes when accessing objects.

Syntax:

```
foreach iterator in collection
	statements
```

```
next
```

Example:

```
[foreach $class in $classes
       if not $class.IsInternal
              ]      class [=$class.Name];
[      endif
next]
```

Example 2:

```
[foreach $i in 1 To 3
      Write "// Step " & $i & "\n"
      ' Do some work
next]
```

In the first line:

$classes is the global object of all generated types. It is a collection of single class objects.

Foreach steps through all the items in $classes, and executes the code following the instruction, up to the **next** statement, for each of them.

In each iteration, **$class** is assigned to the next class object. You simply work with the class object instead of using, classes[i]->Name(), as you would in C++.

All collection iterators have the following additional properties:

Index	The current index, starting with 0
IsFirst	true if the current object is the first of the collection (index is 0)
IsLast	true if the current object is the last of the collection
Current	The current object (this is implicit if not specified and can be left out)

Example:

```
[foreach $enum in $facet.Enumeration
       if not $enum.IsFirst
              ], [
       endif
       ]"[=$enum.Value]"[
next]
```

16.12.9 Subroutines

Code generator supports subroutines in the form of procedures or functions.

Features:

(C) 2015-2021 Altova GmbH

- By-value and by-reference passing of values
- Local/global parameters (local within subroutines)
- Local variables
- Recursive invocation (subroutines may call themselves)

16.12.9.1 Subroutine declaration

Subroutines

Syntax example:

```
Sub SimpleSub()
        … lines of code
 EndSub
```

- **Sub** is the keyword that denotes the procedure.
- **SimpleSub** is the name assigned to the subroutine.
- Round **parenthesis** can contain a parameter list.
- The code block of a subroutine starts immediately after the closing parameter parenthesis.
- **EndSub** denotes the end of the code block.

Note: Recursive or cascaded subroutine **declaration** is not permitted, i.e. a subroutine may not contain another subroutine.

Parameters

Parameters can also be passed by procedures using the following syntax:

- All parameters must be variables
- Variables must be prefixed by the **$** character
- Local variables are defined in a subroutine
- Global variables are declared explicitly, outside of subroutines
- Multiple parameters are separated by the comma character "," within round parentheses
- Parameters can pass values

Parameters - passing values

Parameters can be passed in two ways, by value and by reference, using the keywords **ByVal** and **ByRef** respectively.

Syntax:

```
' define sub CompleteSub()
[Sub CompleteSub( $param, ByVal $paramByValue, ByRef $paramByRef )
] ...
```

- **ByVal** specifies that the parameter is passed by value. Note that most objects can only be passed by reference.
- **ByRef** specifies that the parameter is passed by reference. This is the default if neither ByVal nor ByRef is specified.

Function return values

To return a value from a subroutine, use the **return** statement. Such a function can be called from within an expression.

Example:

```
' define a function
[Sub MakeQualifiedName( ByVal $namespacePrefix, ByVal $localName )
if $namespacePrefix = ""
  return $localName
else
  return $namespacePrefix & ":" & $localName
endif
EndSub
]
```

16.12.9.2 Subroutine invocation

Use **call** to invoke a subroutine, followed by the procedure name and parameters, if any.

```
Call SimpleSub()
```

or

```
Call CompleteSub( "FirstParameter", $ParamByValue, $ParamByRef )
```

Function invocation

To invoke a function (any subroutine that contains a **return** statement), simply use its name inside an expression. Do not use the **call** statement to call functions. Example:

```
$QName = MakeQualifiedName($namespace, "entry")
```

16.12.9.3 Subroutine example

The following example shows subroutine declaration and invocation.

```
[create $outputpath & $module & "output.txt"

' define sub SimpleSub()
Sub SimpleSub()
]SimpleSub() called
[endsub

' execute sub SimpleSub()
Call SimpleSub()
```


(C) 2015-2021 Altova GmbH

```
$ParamByValue       = "Original Value"
]ParamByValue       = [=$ParamByValue]
[$ParamByRef  = "Original Value"
]ParamByRef   = [=$ParamByRef]

' define sub CompleteSub()
[Sub CompleteSub( $param, ByVal $paramByValue, ByRef $paramByRef )
]CompleteSub called.
      param = [=$param]
      paramByValue = [=$paramByValue]
      paramByRef = [=$paramByRef]
[$ParamByRef = "Local Variable"
$paramByValue = "new value"
$paramByRef = "new value"
]     Set values inside Sub
[$ParamByRef = "Local Variable"
$paramByValue = "new value"
$paramByRef = "new value"
]CompleteSub finished.
[endsub

' run sub CompleteSub()
Call CompleteSub( "FirstParameter", $ParamByValue, $ParamByRef )
]
ParamByValue=[=$ParamByValue]
ParamByRef=[=$ParamByRef]
[
Close
]
```

16.12.10 Built in Types

The section describes the properties of the built-in types used in the predefined variables which describe the parsed schema.

16.12.10.1 Library

This object represents the whole library generated from the XML Schema or DTD.

Property	Type	Description
SchemaNamespaces	Namespace collection	Namespaces in this library
SchemaFilename	string	Name of the XSD or DTD file this library is derived from
SchemaType	integer	1 for DTD, 2 for XML Schema
Guid	string	A globally unique ID
CodeName	string	Generated library name (derived from schema file name)

16.12.10.2 Namespace

One namespace object per XML Schema namespace is generated. Schema components that are not in any namespace are contained in a special namespace object with an empty NamespaceURI. Note that for DTD, namespaces are also derived from attributes whose names begin with "xmlns".

Property	Type	Description
CodeName	string	Name for generated code (derived from prefix)
LocalName	string	Namespace prefix
NamespaceURI	string	Namespace URI
Types	Type collection	All types contained in this namespace
Library	Library	Library containing this namespace

16.12.10.3 Type

This object represents a complex or simple type. It is used to generate a class in the target language. There is one additional type per library that represents the document, which has all possible root elements as members.

Anonymous types have an empty LocalName.

Property	Type	Description
CodeName	string	Name for generated code (derived from local name or parent declaration)
LocalName	string	Original name in the schema
Namespace	Namespace	Namespace containing this type
Attributes	Member collection	Attributes contained in this type*
Elements	Member collection	Child elements contained in this type
IsSimpleType	boolean	True for simple types, false for complex types
IsDerived	boolean	True if this type is derived from another type, which is also represented by a Type object
IsDerivedByExtension	boolean	True if this type is derived by extension
IsDerivedByRestriction	boolean	True if this type is derived by restriction
IsDerivedByUnion	boolean	True if this type is derived by union
IsDerivedByList	boolean	True if this type is derived by list
BaseType	Type	The base type of this type (if IsDerived is true)

(C) 2015-2021 Altova GmbH

Property	Type	Description
IsDocumentRootType	boolean	True if this type represents the document itself
Library	Library	Library containing this type
IsFinal	boolean	True if declared as final in the schema
IsMixed	boolean	True if this type can have mixed content
IsAbstract	boolean	True if this type is declared as abstract
IsGlobal	boolean	True if this type is declared globally in the schema
IsAnonymous	boolean	True if this type is declared locally in an element

For simple types only:

Property	Type	Description
IsNativeBound	boolean	True if native type binding exists
NativeBinding	NativeBinding	Native binding for this type
Facets	Facets	Facets of this type
Whitespace	string	Shortcut to the Whitespace facet

* Complex types with text content (these are types with mixed content and complexType with simpleContent) have an additional unnamed attribute member that represents the text content.

16.12.10.4 Member

This object represents an attribute or element in the XML Schema. It is used to create class members of types.

Property	Type	Description
CodeName	string	Name for generated code (derived from local name or parent declaration)
LocalName	string	Original name in the schema. Empty for the special member representing text content of complex types.
NamespaceURI	string	The namespace URI of this Element/Attribute within XML instance documents/streams.
DeclaringType	Type	Type originally declaring the member (equal to ContainingType for non-inherited members)
ContainingType	Type	Type where this is a member of

Property	Type	Description
DataType	Type	Data type of this member's content
Library	Library	Library containing this member's DataType
IsAttribute	boolean	True for attributes, false for elements
IsOptional	boolean	True if minOccurs = 0 or optional attribute
IsRequired	boolean	True if minOccurs > 0 or required attribute
IsFixed	boolean	True for fixed attributes, value is in Default property
IsDefault	boolean	True for attributes with default value, value is in Default property
IsNillable	boolean	True for nillable elements
IsUseQualified	boolean	True if NamespaceURI is not empty
MinOccurs	integer	minOccurs, as in schema. 1 for required attributes
MaxOccurs	integer	maxOccurs, as in schema. 0 for prohibited attributes, -1 for unbounded
Default	string	Default value

16.12.10.5 NativeBinding

This object represents the binding of a simple type to a native type in the target programming language, as specified by the "schemanativetype" map.

Property	Type	Description
ValueType	string	Native type
ValueHandler	string	Formatter class instance

16.12.10.6 Facets

This object represents all facets of a simple type. Inherited facets are merged with the explicitly declared facets. If a Length facet is in effect, MinLength and MaxLength are set to the same value.

Property	Type	Description
DeclaringType	Type	Type facets are declared on
Whitespace	string	"preserve", "collapse" or "replace"

(C) 2015-2021 Altova GmbH

Property	Type	Description
MinLength	integer	Facet value
MaxLength	integer	Facet value
MinInclusive	integer	Facet value
MinExclusive	integer	Facet value
MaxInclusive	integer	Facet value
MaxExclusive	integer	Facet value
TotalDigits	integer	Facet value
FractionDigits	integer	Facet value
List	Facet collection	All facets as list

Facet

This object represents a single facet with its computed value effective for a specific type.

Property	Type	Description
LocalName	string	Facet name
NamespaceURI	string	Facet namespace
FacetType	string	one of "normalization", "lexicalspace", "valuespace-length", "valuespace-enum" or "valuespace-range"
DeclaringType	Type	Type this facet is declared on
FacetCheckerName	string	Name of facet checker (from schemafacet map)
FacetValue	string or integer	Actual value of this facet

17 The MapForce API

The COM-based API of MapForce enables clients to access the functionality of MapForce from a custom code or application, and automate a wide range of tasks.

The MapForce COM API follows the common specifications for automation servers as set out by Microsoft. MapForce is automatically registered as a COM server object during installation. Once the COM server object is registered, you can invoke it from within applications and scripting languages that have programming support for COM calls. This makes it possible to access the MapForce API not only from development environments using .NET, C++ and Visual Basic, but also from scripting languages like JScript and VBScript.

Note the following:

- If you use the MapForce API to create an application that you intend to distribute to other clients, MapForce must be installed on each client computer. Also, your custom integration code must be deployed to (or your application installed on) each client computer.
- Certain API methods, such as `Document.GenerateOutput`, require that the MapForce main window is visible, or MapForce (running as a COM server) is embedded within a graphical user interface. If you need to run mappings in a fully unattended manner, across various platforms, consider using MapForce Server (https://www.altova.com/mapforce-server).

(C) 2015-2021 Altova GmbH

17.1 Accessing the API

To access the MapForce COM API, a new instance of the `Application` object must be created in your application (or script). Once this object is created, you can interact with MapForce by invoking its methods and properties as required (for example, create a new document, open an existing document, generate mapping code, etc).

Prerequisites

To make the MapForce COM object available in your Visual Studio project, add a reference to the MapForce type library (.tlb) file. The following instructions are applicable to Visual Studio 2013, but are similar in other Visual Studio versions:

1. On the **Project** menu, click **Add Reference**.
2. Click **Browse** and select the **MapForce.tlb** file located in the MapForce installation folder.

A sample MapForce API client in C# is available at: **C:\Users\<username>\Documents\Altova\MapForce2021\MapForceExamples\API\C#**.

In Java, the MapForce API is available through Java-COM bridge libraries. These libraries are available in the MapForce installation folder: **C:\Program Files (x86)\Altova\MapForce2021\JavaAPI** (note this path is valid when 32-bit MapForce runs on 64-bit Windows, otherwise adjust the path accordingly).

- `AltovaAutomation.dll`: a JNI wrapper for Altova automation servers
- `AltovaAutomation.jar`: Java classes to access Altova automation servers
- `MapForceAPI.jar`: Java classes that wrap the MapForce automation interface
- `MapForceAPI_JavaDoc.zip`: a `Javadoc` file containing help documentation for the Java API

To allow access to the MapForce automation server directly from Java code, the libraries above must be in the Java `classpath`.

A sample MapForce API client in Java is available at: **C:\Users\<username>\Documents\Altova\MapForce2021\MapForceExamples\API\Java**.

In scripting languages such as JScript or VBScript, the MapForce COM object is accessible through the Microsoft Windows Script Host (see https://msdn.microsoft.com/en-us/library/9bbdkx3k.aspx). Such scripts can be written with a text editor, and do not need compilation, since they are executed by the Windows Script Host packaged with Windows. (To check that the Windows Script Host is running, type `wscript.exe /?` at the command prompt). Several JScript example files that call the MapForce API are available at: **C:\Users\<username>\Documents\Altova\MapForce2021\MapForceExamples\API\JScript**.

メモ 32-ビット MapForce では、登録された名前、またはCOM オブジェクトのプログラム識別子（ProgId）は以下の通りです: `MapForce.Application`。64-ビット MapForce では、名前は以下の通りです: `MapForce_x64.Application`。プログラムの呼び出しは自身のレジストリハイブ、またはグループ（32-bit または64-bit）内のCLASSES レジストリエントリにアクセスすることにご注意してください。ですから、標準のコマンドプロンプトと上 64-bit Windows のWindows Explorerを使用してスクリプトを実行すると、64-ビット MapForce. を指す 64-ビットレジストリエントリにアクセスされます。この理由のため、MapForce 32-ビットと64-ビットの両方がインストールされている場合、32-ビット MapForce. を呼び出すために特別な処理が必要になります。例えば、スクリプトホストがプログラムを呼び出す場合以下を行います:

1. ディレクトリを **C:\Windows\SysWOW64** に変更します。
2. コマンドライン上で **wscript.exe** を入力し実行するスクリプトパスを以下のように入力します:

```
wscript.exe "C:\Users\...
```

```
\Documents\Altova\MapForce2021\MapForceExamples\API\JScript\start.js"
```

Guidelines

The following guidelines should be considered in your client code:

- Do not hold references to objects in memory longer than you need them. If a user interacts between two calls of your client, then there is no guarantee that these references are still valid.
- Be aware that if your client code crashes, instances of MapForce may still remain in the system. For details on how to avoid error messages, see Error handling.
- Free references explicitly, if using languages such as C++.

Creating the Application object

The syntax to create the starting `Application` object depends on the programming language, as shown in the examples below:

C#

```
// Create a new instance of MapForce via its automation interface.
MapForceLib.Application objMapForce = new MapForceLib.Application();
```

Java

```
// Start MapForce as COM server.
com.altova.automation.MapForce.Application objMapForce = new Application();
// COM servers start up invisible so we make it visible
objMapForce.setVisible(true);
```

JScript

```
// Access a running instance, or create a new instance of MapForce.
try
{
	objMapForce = WScript.GetObject ("", "MapForce.Application");
	// unhide application if it is a new instance
	objMapForce.Visible = true;
}
catch(err) { WScript.Echo ("Can't access or create MapForce.Application"); }
```

VBA

```
' Create a new instance of MapForce.
Dim objMapForce As Application
Set objMapForce = CreateObject("MapForce.Application")
```

VBScript

```
' Access a running instance, or create a new instance of MapForce.
Set objMapForce = GetObject("MapForce.Application");
```

Visual Basic

(C) 2015-2021 Altova GmbH

```
Dim objMapForce As MapForceLib.Application = New MapForceLib.Application
```

17.2 The Object Model

The starting point for every application which uses the MapForce API is the `Application` object. All other interfaces are accessed through the `Application` object as the starting point.

The object model of the MapForce API can be represented as follows (each indentation level indicates a child–parent relationship with the level directly above):

Application
- Options
- Project
  - ProjectItem
- Documents
  - Document
    - MapForceView
    - Mapping
      - Component
        - Datapoint
      - Components
      - Connection
    - Mappings
- ErrorMarkers
  - ErrorMarker

AppOutputLines
- AppOutputLine
  - AppOutputLines
    - ...
  - AppOutputLineSymbol

For information about creating an instance the `Application` object, see Accessing the API. For reference to the objects exposed by the API, see Object Reference.

(C) 2015-2021 Altova GmbH

17.3 Error Handling

The MapForce API returns errors in two different ways. Every API method returns an `HRESULT`. This return value informs the caller about any malfunctions during the execution of the method. If the call was successful, the return value is equal to `S_OK`. C/C++ programmers generally use `HRESULT` to detect errors.

Visual Basic, scripting languages, and other high-level development environments do not give the programmer access to the returning `HRESULT` of a COM call. They use the second error-raising mechanism supported by the MapForce API, the `IErrorInfo` interface. If an error occurs, the API creates a new object that implements the `IErrorInfo` interface. The development environment takes this interface and fills its own error-handling mechanism with the provided information.

The following text describes how to deal with errors raised from the MapForce API in different development environments.

Visual Basic

A common way to handle errors in Visual Basic is to define an error handler. This error handler can be set with the `On Error` statement. Usually the handler displays an error message and does some cleanup to avoid spare references and any kind of resource leaks. Visual Basic fills its own `Err` object with the information from the `IErrorInfo` interface.

```
Sub Validate()
  'place variable declarations here

  'set error handler
  On Error GoTo ErrorHandler

  'if generation fails, program execution continues at ErrorHandler:
  objMapForce.ActiveDocument.GenerateXSLT()

  'additional code comes here

  'exit
  Exit Sub

  ErrorHandler:
  MsgBox("Error: " & (Err.Number - vbObjectError) & Chr(13) &
     "Description: " & Err.Description)
End Sub
```

JavaScript

The Microsoft implementation of JavaScript (JScript) provides a try-catch mechanism to deal with errors raised from COM calls. It is very similar to the VisualBasic approach, in that you also declare an error object containing the necessary information.

```
function Generate() {
    // please insert variable declarations here

    try {
        objMapForce.ActiveDocument.GenerateXSLT();
    }
    catch (Error) {
        sError = Error.description;
        nErrorCode = Error.number & 0xffff;
        return false;
    }

    return true;
}
```

C/C++

C/C++ gives you easy access to the `HRESULT` of the COM call and to the `IErrorInterface`.

```
   HRESULT hr;

   // Call GenerateXSLT() from the MapForce API
   if(FAILED(hr = ipDocument->GenerateXSLT()))
   {
      IErrorInfo *ipErrorInfo = Null;

      if(SUCCEEDED(::GetErrorInfo(0, &ipErrorInfo)))
      {
         BSTR   bstrDescr;
         ipErrorInfo->GetDescription(&bstrDescr);

         // handle Error information
         wprintf(L"Error message:\t%s\n",bstrDescr);
         ::SysFreeString(bstrDescr);

         // release Error info
         ipErrorInfo->Release();
      }
   }
```


(C) 2015-2021 Altova GmbH

17.4 Example C# Project

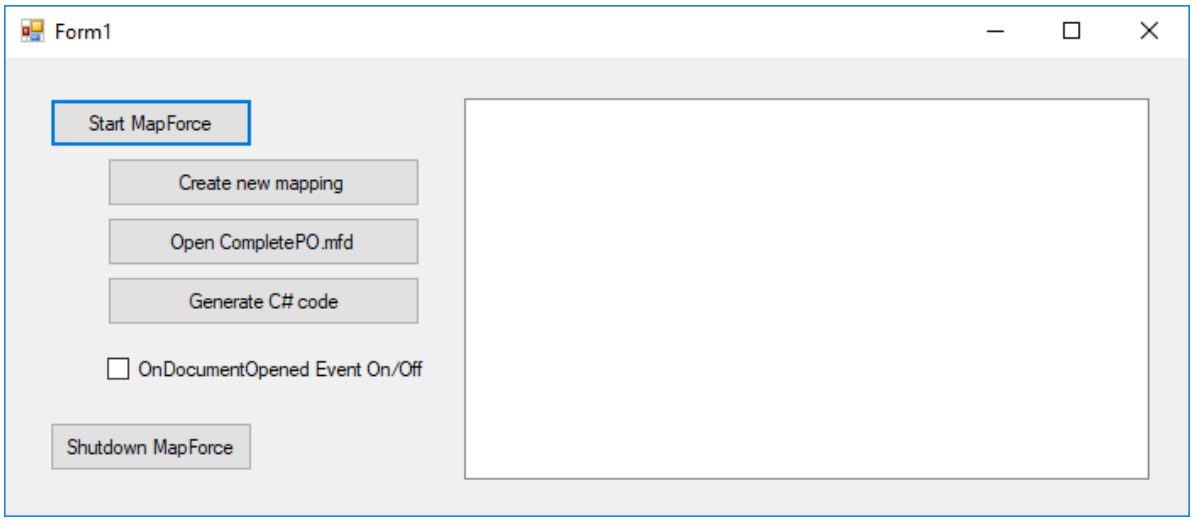
After you install MapForce, an example MapForce API client project for C# is available in the directory **C:\Users\<username>\Documents\Altova\MapForce2021\MapForceExamples\API**.

To compile and run the example, open the solution .sln file in Visual Studio and run **Debug | Start Debugging** or **Debug | Start Without Debugging**.

Note: If you have a 64-bit operating system and are using a 32-bit installation of MapForce, add the **x86** platform in the solution's Configuration Manager and build the sample using this configuration. A new x86 platform (for the active solution in Visual Studio) can be created in the New Solution Platform dialog (**Build | Configuration Manager | Active solution platform | <New...>**).

When you run the example, a Windows form is displayed, containing buttons that invoke basic MapForce operations:

- Start MapForce
- Create a new mapping design
- Open the CompletePO.mfd file from the **...\MapForceExamples** folder (note that you may need to adjust the path to point to the **\MapForceExamples** folder on your machine)
- Generate C# code in a temp directory
- Shut down MapForce



Code listing

The listing is commented for ease of understanding. The code essentially consists of a series of handlers for the buttons in the user interface shown above.

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
```

```
using System.Text;
using System.Windows.Forms;

namespace WindowsFormsApplication2
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

        // An instance of MapForce accessed via its automation interface.
        MapForceLib.Application MapForce;

        // Location of examples installed with MapForce
        String strExamplesFolder;

        private void Form1_Load(object sender, EventArgs e)
        {
        }

        // handler for the "Start MapForce" button
        private void StartMapForce_Click(object sender, EventArgs e)
        {
            if (MapForce == null)
            {
                Cursor.Current = Cursors.WaitCursor;

                // if we have no MapForce instance, we create one a nd make it visible.
                MapForce = new MapForceLib.Application();
                MapForce.Visible = true;

                // locate examples installed with MapForce.
                int majorVersionYear = MapForce.MajorVersion + 1998;
                strExamplesFolder = Environment.GetEnvironmentVariable("USERPROFILE") +
 "\\My Documents\\Altova\\MapForce" + Convert.ToString(majorVersionYear) + "\
\MapForceExamples\\";

                Cursor.Current = Cursors.Default;
            }
            else
            {
                // if we have already an MapForce instance running we toggle its
visibility flag.
                MapForce.Visible = !MapForce.Visible;
            }
        }

        // handler for the "Open CompletePO.mfd" button
        private void openCompletePO_Click(object sender, EventArgs e)
        {
            if (MapForce == null)
                StartMapForce_Click(null, null);

            // Open one of the sample files installed with the product.
            MapForce.OpenDocument(strExamplesFolder + "CompletePO.mfd");
        }
```


(C) 2015-2021 Altova GmbH

```
        // handler for the "Create new mapping" button
        private void newMapping_Click(object sender, EventArgs e)
        {
            if (MapForce == null)
                StartMapForce_Click(null, null);

            // Create a new mapping
            MapForce.NewMapping();
        }

        // handler for the "Shutdown MapForce" button
        // shut-down application instance by explicitly releasing the COM object.
        private void shutdownMapForce_Click(object sender, EventArgs e)
        {
            if (MapForce != null)
            {
                // allow shut-down of MapForce by releasing UI
                MapForce.Visible = false;

                // explicitly release COM object
                try
                {
                    while
(System.Runtime.InteropServices.Marshal.ReleaseComObject(MapForce) > 0) ;
                }
                finally
                {
                    // avoid later access to this object.
                    MapForce = null;
                }
            }
        }

        // handler for button "Generate C# Code"
        private void generateCppCode_Click(object sender, EventArgs e)
        {
            if (MapForce == null)
                listBoxMessages.Items.Add("start MapForce first.");
            // COM errors get returned to C# as exceptions. We use a try/catch block to
handle them.
            try
            {
                MapForceLib.Document doc = MapForce.ActiveDocument;

                listBoxMessages.Items.Add("Active document " + doc.Name);
                doc.GenerateCHashCode();

            }
            catch (Exception ex)
            {
                // The COM call was not successful.
                // Probably no application instance has been started or no document is
open.
                MessageBox.Show("COM error: " + ex.Message);
            }
        }

        delegate void addListBoxItem_delegate(string sText);
        // called from the UI thread
```

```
        private void addListBoxItem(string sText)
        {
            listBoxMessages.Items.Add(sText);
        }
        // wrapper method to allow to call UI controls methods from a worker thread
        void syncWithUIthread(Control ctrl, addListBoxItem_delegate methodToInvoke,
String sText)
        {
            // Control.Invoke: Executes on the UI thread, but calling thread waits for
completion before continuing.
            // Control.BeginInvoke: Executes on the UI thread, and calling thread
doesn't wait for completion.
            if (ctrl.InvokeRequired)
                ctrl.BeginInvoke(methodToInvoke, new Object[] { sText });
        }

        // event handler for OnDocumentOpened event
        private void handleOnDocumentOpened(MapForceLib.Document i_ipDocument)
        {
            String sText = "";

            if (i_ipDocument.Name.Length > 0)
                sText = "Document " + i_ipDocument.Name + " was opened!";
            else
                sText = "A new mapping was created.";

            // we need to synchronize the calling thread with the UI thread because
            // the COM events are triggered from a working thread
            addListBoxItem_delegate methodToInvoke = new
addListBoxItem_delegate(addListBoxItem);
            // call syncWithUIthread with the following arguments:
            // 1 - listBoxMessages - list box control to display messages from COM
events
            // 2 - methodToInvoke  - a C# delegate which points to the method which
will be called from the UI thread
            // 3 - sText           - the text to be displayed in the list box
            syncWithUIthread(listBoxMessages, methodToInvoke, sText);
        }

        private void checkBoxEventOnOff_CheckedChanged(object sender, EventArgs e)
        {
            if (MapForce != null)
            {
                if (checkBoxEventOnOff.Checked)
                    MapForce.OnDocumentOpened += new
MapForceLib._IApplicationEvents_OnDocumentOpenedEventHandler(handleOnDocumentOpened);
                else
                    MapForce.OnDocumentOpened -= new
MapForceLib._IApplicationEvents_OnDocumentOpenedEventHandler(handleOnDocumentOpened);
            }
        }
    }
}
```


(C) 2015-2021 Altova GmbH

17.5 Example Java Project

After you install MapForce, an example MapForce API client project for Java is available in the directory **C:\Users\<username>\Documents\Altova\MapForce2021\MapForceExamples\API**.

You can test the Java example directly from the command line, using the batch file `BuildAndRun.bat`, or you can compile and run the example project from within Eclipse. See below for instructions on how to use these procedures.

File list

The Java examples folder contains all the files required to run the example project. These files are listed below:

`AltovaAutomation.dll`	Java-COM bridge: DLL part
`AltovaAutomation.jar`	Java-COM bridge: Java library part
`MapForceAPI.jar`	Java classes of the MapForce API
`RunMapForce.java`	Java example source code
`BuildAndRun.bat`	Batch file to compile and run example code from the command line prompt. Expects folder where Java Virtual Machine resides as parameter.
`.classpath`	Eclipse project helper file
`.project`	Eclipse project file
`MapForceAPI_JavaDoc.zip`	`Javadoc` file containing help documentation for the Java API

What the example does

The example starts up MapForce and performs a few operations, including opening and closing documents. When done, MapForce stays open. You must close it manually.

Running the example from the command line

To run the example from the command line, open a command prompt window, go to the Java folder of the API Examples folder (*see above for location*), and then type:

```
buildAndRun.bat "<Path-to-the-Java-bin-folder>"
```

The Java binary folder must be that of a Java Development Kit (JDK) 7 or later installation on your computer.

Press the **Return** key. The Java source in `RunMapForce.java` will be compiled and then executed.

Loading the example in Eclipse

Open Eclipse and use the **File | Import... | General | Existing Projects into Workspace** command to add the Eclipse project file (`.project`) located in the Java folder of the API Examples folder (*see above for location*). The project `RunMapForce` will then appear in your Package Explorer or Navigator.

Select the project and then the command **Run as | Java Application** to execute the example.

Note: You can select a class name or method of the Java API and press F1 to get help for that class or method.

Java source code listing

The Java source code in the example file `RunMapForce.java` is listed below with comments.

```
// access general JAVA-COM bridge classes
import java.util.Iterator;

import com.altova.automation.libs.*;

// access XMLSpy Java-COM bridge
import com.altova.automation.MapForce.*;
import com.altova.automation.MapForce.Enums.ENUMProgrammingLanguage;

/**
 * A simple example that starts the COM server and performs a few operations on it.
 * Feel free to extend.
 */
public class RunMapForce
{
   public static void main(String[] args)
   {
      // an instance of the application.
      Application mapforce = null;

      // instead of COM error handling use Java exception mechanism.
      try
      {
         // Start MapForce as COM server.
         mapforce = new Application();
         // COM servers start up invisible so we make it visible
         mapforce.setVisible(true);

         // The following lines attach to the application events using a default
implementation
         // for the events and override one of its methods.
         // If you want to override all document events it is better to derive your
listener class
         // from DocumentEvents and implement all methods of this interface.
         mapforce.addListener(new ApplicationEventsDefaultHandler()
         {
            @Override
            public void onDocumentOpened(Document i_ipDoc) throws AutomationException
            {
               String name = i_ipDoc.getName();

               if (name.length() > 0)
                  System.out.println("Document " + name + " was opened.");
               else
                  System.out.println("A new mapping was created.");
            }
         });

         // Locate samples installed with the product.
         int majorVersionYear = mapforce.getMajorVersion() + 1998;
         String strExamplesFolder = System.getenv("USERPROFILE") + "\\Documents\
\Altova\\MapForce" + Integer.toString(majorVersionYear) + "\\MapForceExamples\\";
         // create a new MapForce mapping and generate c++ code
```


(C) 2015-2021 Altova GmbH

```
         Document newDoc = mapforce.newMapping();
         ErrorMarkers err1 = newDoc.generateCodeEx(ENUMProgrammingLanguage.eCpp);
         display(err1);
         // open CompletePO.mfd and generate c++ code
         Document doc = mapforce.openDocument(strExamplesFolder + "CompletePO.mfd");
         ErrorMarkers err2 = doc.generateCodeEx(ENUMProgrammingLanguage.eCpp);
         display(err2);

         doc.close();
         doc = null;

         System.out.println("Watch MapForce!");
      }
      catch (AutomationException e)
      {
         // e.printStackTrace();
      }
      finally
      {
         // Make sure that MapForce can shut down properly.
         if (mapforce != null)
            mapforce.dispose();

         // Since the COM server was made visible and still is visible, it will keep
running
         // and needs to be closed manually.
         System.out.println("Now close MapForce!");
      }
   }

   public static void display(ErrorMarkers err) throws AutomationException
   {
      Iterator<ErrorMarker> itr = err.iterator();

      if (err.getCount() == 0)
          System.out.print("Code generation completed successfully.\n");

      while (itr.hasNext())
      {
         String sError = "";
          Object element = itr.next();
          if (element instanceof ErrorMarker)
             sError = ((ErrorMarker)element).getText();
          System.out.print("Error text: " + sError + "\n");
      }
   }
}
```

17.6 JScript Examples

After you install MapForce, several JScript example files are available in the directory **C:\Users\<username>\Documents\Altova\MapForce2021\MapForceExamples\API**.

The example files can be run in one of two ways:

- *From the command line:*
  Open a command prompt window and type the name of one of the example scripts (for example, `Start.js`). The Windows Scripting Host that is packaged with Windows will execute the script.

- *From Windows Explorer:*
  In Windows Explorer, browse for the JScript file and double-click it. The Windows Scripting Host that is packaged with Windows will execute the script. After the script is executed, the command console gets closed automatically.

The following example files are included:

`Start.js`	Start Mapforce registered as an automation server or connect to a running instance. You can also view this code listing in Start Application.
`DocumentAccess.js`	Shows how to open, iterate and close documents. You can also view this code listing in Simple Document Access.
`GenerateCode.js`	Shows how to invoke code generation using JScript. You can also view this code listing in Generate Code.
`Readme.txt`	Provides basic help to run the scripts.

This documentation additionally includes a few extra JScript code listings:

- Example: Code Generation
- Example: Mapping Execution
- Example: Project Support

17.6.1 Start Application

The JScript code listing below starts the application and shuts it down. If an instance of the application is already running, the running instance will be returned. To run the script, start it from a command prompt window or from Windows Explorer, see also Accessing the API.

```
// Initialize application's COM object. This will start a new instance of the
application and
// return its main COM object. Depending on COM settings, a the main COM object of an
already
// running application might be returned.

try {   objMapForce = WScript.GetObject("", "MapForce.Application");   }
catch(err) {}
```


(C) 2015-2021 Altova GmbH

```
if( typeof( objMapForce ) == "undefined" )
{
   try   {   objMapForce = WScript.GetObject("", "MapForce_x64.Application")   }
   catch(err)
   {
      WScript.Echo( "Can't access or create MapForce.Application" );
      WScript.Quit();
   }
}

// if newly started, the application will start without its UI visible. Set it to
visible.
objMapForce.Visible = true;

WScript.Echo(objMapForce.Edition + " has successfully started. ");

objMapForce.Visible = false; // will shutdown application if it has no more COM
connections
//objMapForce.Visible = true;   // will keep application running with UI visible
```

17.6.2 Simple Document Access

The JScript listing below shows how to open documents, set a document as the active document, iterate through the open documents, and close documents.

```
// Initialize application's COM object. This will start a new instance of the
application and
// return its main COM object. Depending on COM settings, a the main COM object of an
already
// running application might be returned.
try {   objMapForce = WScript.GetObject("", "MapForce.Application");   }
catch(err) {}

if( typeof( objMapForce ) == "undefined" )
{
   try   {   objMapForce = WScript.GetObject("", "MapForce_x64.Application")   }
   catch(err)
   {
      WScript.Echo( "Can't access or create MapForce.Application" );
      WScript.Quit();
   }
}

// if newly started, the application will start without its UI visible. Set it to
visible.
objMapForce.Visible = true;

// ************************** code snippet for "Simple Document Access"
**********************

// Locate examples via USERPROFILE shell variable. The path needs to be adapted to
major release versions.
objWshShell = WScript.CreateObject("WScript.Shell");
majorVersionYear = objMapForce.MajorVersion + 1998
strExampleFolder = objWshShell.ExpandEnvironmentStrings("%USERPROFILE%") + "\
\Documents\\Altova\\MapForce" + majorVersionYear + "\\MapForceExamples\\";
```

```
objMapForce.Documents.OpenDocument(strExampleFolder + "CompletePO.mfd");
objMapForce.Documents.OpenDocument(strExampleFolder + "Altova_Hierarchical_DB.mfd");

// ************************** code snippet for "Simple Document Access"
**********************

// ************************** code snippet for "Iteration"
**********************************

// go through all open documents using a JScript Enumerator
for (var iterDocs = new Enumerator(objMapForce.Documents); !iterDocs.atEnd();
iterDocs.moveNext())
{
   objName = iterDocs.item().Name;
   WScript.Echo("Document name: " + objName);
}

// go through all open documents using index-based access to the document collection
for (i = objMapForce.Documents.Count; i > 0; i--)
   objMapForce.Documents.Item(i).Close();


// ************************** code snippet for "Iteration"
**********************************

//objMapForce.Visible = false;      // will shutdown application if it has no more COM
connections
objMapForce.Visible = true;   // will keep application running with UI visible
```

The code listed above is available as a sample file (see JScript Examples). To run the script, start it from a command prompt window or from Windows Explorer.

17.6.3 Generate Code

The JScript listing below shows how to open documents, set a document as the active document, iterate through the open documents, and generate C++ code.

```
// Initialize application's COM object. This will start a new instance of the
application and
// return its main COM object. Depending on COM settings, a the main COM object of an
already
// running application might be returned.
try {   objMapForce = WScript.GetObject("", "MapForce.Application");   }
catch(err) {}

if( typeof( objMapForce ) == "undefined" )
{
   try   {   objMapForce = WScript.GetObject("", "MapForce_x64.Application")   }
   catch(err)
   {
      WScript.Echo( "Can't access or create MapForce.Application" );
      WScript.Quit();
   }
}

// if newly started, the application will start without its UI visible. Set it to
```

```
visible.
objMapForce.Visible = true;

// ************************** code snippet for "Simple Document Access"
************************

// Locate examples via USERPROFILE shell variable. The path needs to be adapted to
major release versions.
objWshShell = WScript.CreateObject("WScript.Shell");
majorVersionYear = objMapForce.MajorVersion + 1998
strExampleFolder = objWshShell.ExpandEnvironmentStrings("%USERPROFILE%") + "\
\Documents\\Altova\\MapForce" + majorVersionYear + "\\MapForceExamples\\";

objMapForce.Documents.OpenDocument(strExampleFolder + "CompletePO.mfd");
//objMapForce.Documents.OpenDocument(strExampleFolder + "Altova_Hierarchical_DB.mfd");
objMapForce.Documents.NewDocument();

// ************************** code snippet for "Simple Document Access"
************************

// ************************** code snippet for "Iteration"
**********************************

objText = "";
// go through all open documents using a JScript Enumerator and generate c++ code
for (var iterDocs = new Enumerator(objMapForce.Documents); !iterDocs.atEnd();
iterDocs.moveNext())
{
   objText += "Generated c++ code result for document " + iterDocs.item().Name + " :
\n";
   objErrorMarkers = iterDocs.item().generateCodeEx(1); // ENUMProgrammingLanguage.eCpp
= 1

   bSuccess = true;
   for (var iterErrorMarkers = new
Enumerator(objErrorMarkers); !iterErrorMarkers.atEnd(); iterErrorMarkers.moveNext())
   {
      bSuccess = false;
      objText += "\t" + iterErrorMarkers.item().Text + "\n";
   }

   if (bSuccess)
       objText += "\tCode generation completed successfully.\n";

   objText += "\n";
}

WScript.Echo(objText);

// go through all open documents using index-based access to the document collection
for (i = objMapForce.Documents.Count; i > 0; i--)
   objMapForce.Documents.Item(i).Close();


// ************************** code snippet for "Iteration"
**********************************

//objMapForce.Visible = false;      // will shutdown application if it has no more COM
connections
objMapForce.Visible = true;   // will keep application running with UI visible
```

The code listed above is available as a sample file (see JScript Examples). To run the script, start it from a command prompt window or from Windows Explorer.

17.6.4 Generate Code (Alternative)

The following JScript example shows how to load an existing document and generate different kinds of mapping code for it.

```
// ------------------- begin JScript example --------------------
// Generate Code for existing mapping.
// works with Windows scripting host.

// ----------------- helper function -----------------
function Exit(strErrorText)
{
   WScript.Echo(strErrorText);
   WScript.Quit(-1);
}

function ERROR(strText, objErr)
{
   if (objErr != null)
      Exit ("ERROR: (" + (objErr.number & 0xffff) + ")" + objErr.description + " - " +
strText);
   else
      Exit ("ERROR: " + strText);
}
// ---------------------------------------------------

// ----------------- MAIN ------------------

// ----- create the Shell and FileSystemObject of the windows scripting
try
{
   objWshShell = WScript.CreateObject("WScript.Shell");
   objFSO = WScript.CreateObject("Scripting.FileSystemObject");
}
catch(err)
   { Exit("Can't create WScript.Shell object"); }

// ----- open MapForce or access running instance and make it visible
try
{
   objMapForce = WScript.GetObject ("", "MapForce.Application");
   objMapForce.Visible = true;      // remove this line to perform background
processing
}
catch(err) { WScript.Echo ("Can't access or create MapForce.Application"); }

// ----- open an existing mapping. adapt this to your needs!
objMapForce.OpenDocument(objFSO.GetAbsolutePathName ("Test.mfd"));

// ----- access the mapping to have access to the code generation methods
var objDoc = objMapForce.ActiveDocument;

// ----- set the code generation output properties and call the code generation
methods.
```


(C) 2015-2021 Altova GmbH

```
// ----- adapt the output directories to your needs
try
{
   // ----- code generation uses some of these options
   var objOptions = objMapForce.Options;

   // ----- generate XSLT -----
   objOptions.XSLTDefaultOutputDirectory = "C:\\test\\TestCOMServer\\XSLT";
   objDoc.GenerateXSLT();

   // ----- generate Java Code -----
   objOptions.CodeDefaultOutputDirectory = "C:\\test\\TestCOMServer\\Java";
   objDoc.GenerateJavaCode();

   // ----- generate CPP Code, use same cpp code options as the last time -----
   objOptions.CodeDefaultOutputDirectory = "C:\\test\\TestCOMServer\\CPP";
   objDoc.GenerateCppCode();

   // ----- generate C# Code, use options C# code options as the last time -----
   objOptions.CodeDefaultOutputDirectory = "C:\\test\\TestCOMServer\\CHash";
   objDoc.GenerateCHashCode();
}
catch (err)
   { ERROR ("while generating XSL or program code", err); }


// hide MapForce to allow it to shut down
objMapForce.Visible = false;

// -------------------- end example ---------------------
```

17.6.5 Run a Mapping

The following JScript example shows how to load an existing document with a simple mapping, access its components, set input- and output-instance file names and execute the mapping.

```
/*
   This sample file performs the following operations:

   Load existing MapForce mapping document.
   Find source and target component.
   Set input and output instance filenames.
   Execute the transformation.

   Works with Windows scripting host.
*/

// ---- general helpers ------------------------------

function Exit( message )
{
   WScript.Echo( message );
   WScript.Quit(-1);
}

function ERROR( message, err )
{
```

```
   if( err != null )
      Exit( "ERROR: (" + (err.number & 0xffff) + ") " + err.description + " - " +
message );
   else
      Exit( "ERROR: " + message );
}



// ---- MapForce constants -----------------------

var eComponentUsageKind_Unknown     = 0;
var eComponentUsageKind_Instance    = 1;
var eComponentUsageKind_Input       = 2;
var eComponentUsageKind_Output      = 3;



// ---- MapForce helpers -----------------------

// Searches in the specified mapping for a component by name and returns it.
// If not found, throws an error.
function FindComponent( mapping, component_name )
{
   var components = mapping.Components;
   for( var i = 0 ; i < components.Count ; ++i )
   {
      var component = components.Item( i + 1 );
      if( component.Name == component_name )
         return component;
   }
   throw new Error( "Cannot find component with name " + component_name );
}



// Browses components in a mapping and returns the first one found acting as
// source component (i.e. having connections on its right side).
function GetFirstSourceComponent( mapping )
{
   var components = mapping.Components;
   for( var i = 0 ; i < components.Count ; ++i )
   {
      var component = components.Item( i + 1 );
      if( component.UsageKind == eComponentUsageKind_Instance &&
         component.HasOutgoingConnections )
      {
         return component;
      }
   }
   throw new Error( "Cannot find a source component" );
}

// Browses components in a mapping and returns the first one found acting as
// target component (i.e. having connections on its left side).
function GetFirstTargetComponent( mapping )
{
   var components = mapping.Components;
   for( var i = 0 ; i < components.Count ; ++i )
   {
```


(C) 2015-2021 Altova GmbH

```
      var component = components.Item( i + 1 );
      if( component.UsageKind == eComponentUsageKind_Instance &&
         component.HasIncomingConnections )
      {
         return component;
      }
   }
   throw new Error( "Cannot find a target component" );
}



function IndentTextLines( s )
{
   return "\t" + s.replace( /\n/g, "\n\t" );
}

function GetAppoutputLineFullText( oAppoutputLine )
{
   var s = oAppoutputLine.GetLineText();
   var oAppoutputChildLines = oAppoutputLine.ChildLines;
   var i;

   for( i = 0 ; i < oAppoutputChildLines.Count ; ++i )
   {
      oAppoutputChildLine = oAppoutputChildLines.Item( i + 1 );
      sChilds = GetAppoutputLineFullText( oAppoutputChildLine );
      s += "\n" + IndentTextLines( sChilds );
   }

   return s;
}

// Create a nicely formatted string from AppOutputLines
function GetResultMessagesString( oAppoutputLines )
{
   var s1 = "Transformation result messages:\n";
   var oAppoutputLine;
   var i;

   for( i = 0 ; i < oAppoutputLines.Count ; ++i )
   {
      oAppoutputLine = oAppoutputLines.Item( i + 1 );
      s1 += GetAppoutputLineFullText( oAppoutputLine );
      s1 += "\n";
   }

   return s1;
}



// ---- MAIN -----------------------------------

var wshShell;
var fso;
var mapforce;

// create the Shell and FileSystemObject of the windows scripting system
try
```

```
{
   wshShell = WScript.CreateObject( "WScript.Shell" );
   fso = WScript.CreateObject( "Scripting.FileSystemObject" );
}
catch( err )
   { ERROR( "Can't create windows scripting objects", err ); }

// open MapForce or access currently running instance
try
{
   mapforce = WScript.GetObject( "", "MapForce.Application" );
}
catch( err )
   { ERROR( "Can't access or create MapForce.Application", err ); }

try
{
   // Make MapForce UI visible. This is an API requirement for output generation.
  mapforce.Visible = true;

   // open an existing mapping.
   // **** adjust the examples path to your needs ! **************
   var sMapForceExamplesPath = fso.BuildPath(
            wshShell.SpecialFolders( "MyDocuments" ),
            "Altova\\MapForce2021\\MapForceExamples" );
   var sDocFilename = fso.BuildPath( sMapForceExamplesPath, "PersonList.mfd" );
   var doc = mapforce.OpenDocument( sDocFilename );

   // Find existing components by name in the main mapping.
   // Note, the names of components may not be unique as a schema component's name
   // is derived from its schema file name.
   var source_component = FindComponent( doc.MainMapping, "Employees" );
   var target_component = FindComponent( doc.MainMapping, "PersonList" );
   // If you do not know the names of the components for some reason, you could
   // use the following functions instead of FindComponent.
   //var source_component = GetFirstSourceComponent( doc.MainMapping );
   //var target_component = GetFirstTargetComponent( doc.MainMapping );

   // specify the desired input and output files.
   source_component.InputInstanceFile = fso.BuildPath( sMapForceExamplesPath,
"Employees.xml" );
   target_component.OutputInstanceFile = fso.BuildPath( sMapForceExamplesPath,
"test_transformation_results.xml" );

   // Perform the transformation.
   // You can use doc.GenerateOutput() if you do not need result messages.
   // If you have a mapping with more than one target component and you want
   // to execute the transformation only for one specific target component,
   // call target_component.GenerateOutput() instead.
   var result_messages = doc.GenerateOutputEx();

   var summary_info =
         "Transformation performed from " + source_component.InputInstanceFile + "\n" +
         "to " + target_component.OutputInstanceFile + "\n\n" +
         GetResultMessagesString( result_messages );
   WScript.Echo( summary_info );
}
catch( err )
{
```


(C) 2015-2021 Altova GmbH

```
   ERROR( "Failure", err );
}
```

17.6.6 Project Tasks

The following JScript example shows how to use the MapForce API to automate tasks pertaining to MapForce projects. Before running the example, make sure to edit the variable strSamplePath so that it points to the **MapForceExamples** folder of your MapForce installation.

To successfully run all operations in this example below, you will need the Enterprise edition of MapForce. If you have the Professional edition, comment out the lines that insert the WebService project.

```
// ############ global variables #################
var objMapForce = null;
var objWshShell = null;
var objFSO = null;

// !!! adapt the following path to your needs. !!!
var strSamplePath = "C:\\Users\\<username>\\Documents\\Altova\\MapForce2021\
\MapForceExamples\\";

// ###################### Helpers ##############################

function Exit(strErrorText)
{
   WScript.Echo(strErrorText);
   WScript.Quit(-1);
}

function ERROR(strText, objErr)
{
   if (objErr != null)
      Exit ("ERROR: (" + (objErr.number & 0xffff) + ")" + objErr.description + " - " +
strText);
   else
      Exit ("ERROR: " + strText);
}

function CreateGlobalObjects ()
{
   // the Shell and FileSystemObject of the windows scripting host often useful
   try
   {
      objWshShell = WScript.CreateObject("WScript.Shell");
      objFSO = WScript.CreateObject("Scripting.FileSystemObject");
   }
   catch(err)
      { Exit("Can't create WScript.Shell object"); }

   // create the MapForce connection
   // if there is a running instance of MapForce (that never had a connection) - use it
   // otherwise, we automatically create a new instance
   try
   {
      objMapForce = WScript.GetObject("", "MapForce.Application");
   }
```

```
   catch(err)
   {
      { Exit("Can't access or create MapForce.Application"); }
   }
}

// ------------------------------------------------------
// print project tree items and their properties recursively.
// ------------------------------------------------------
function PrintProjectTree( objProjectItemIter, strTab )
{
   while ( ! objProjectItemIter.atEnd() )
   {
      // get current project item
      objItem = objProjectItemIter.item();

      try
      {
         // ----- print common properties
         strGlobalText += strTab + "[" + objItem.Kind + "]" + objItem.Name + "\n";

         // ----- print code generation properties, if available
         try
         {
            if ( objItem.CodeGenSettings_UseDefault )
               strGlobalText += strTab + "  Use default code generation settings\n";
            else
               strGlobalText += strTab + "  code generation language is " +
                                                    objItem.CodeGenSettings_Language +
                                " output folder is " +
objItem.CodeGenSettings_OutputFolder + "\n";
         }
         catch( err ) {}

         // ----- print WSDL settings, if available
         try
         {
            strGlobalText += strTab + "  WSDL File is " + objItem.WSDLFile +
                             " Qualified Name is " + objItem.QualifiedName + "\n";
         }
         catch( err ) {}
      }
      catch( ex )
         { strGlobalText += strTab + "[" + objItem.Kind + "]\n" }

      // ---- recurse
      PrintProjectTree( new Enumerator( objItem ), strTab + '   ' );

      objProjectItemIter.moveNext();
   }
}

// ------------------------------------------------------
// Load example project installed with MapForce.
// ------------------------------------------------------
function LoadSampleProject()
{
   // close open project
   objProject = objMapForce.ActiveProject;
   if ( objProject != null )
```

```
      objProject.Close();

   // open sample project and iterate through it.
   objProject = objMapForce.OpenProject(strSamplePath + "MapForceExamples.mfp");
   // dump properties of all project items
   strGlobalText = '';
   PrintProjectTree( new Enumerator (objProject), ' ' )
   WScript.Echo( strGlobalText );

   objProject.Close();
}

// ------------------------------------------------------------
// Create a new project with some folders, mappings and a
// Web service project.
// ------------------------------------------------------------
function CreateNewProject()
{
   try
   {
      // create new project and specify file to store it.
      objProject = objMapForce.NewProject(strSamplePath + "Sample.mfp");

      // create a simple folder structure
      objProject.CreateFolder( "New Folder 1");
      objFolder1 = objProject.Item(1);
      objFolder1.CreateFolder( "New Folder 2");
      objFolder2 = ( new Enumerator( objFolder1 ) ).item();   // an alternative to
Item(0)

      // add two different mappings to folder structure
      objFolder1.AddFile( strSamplePath + "DB_Altova_SQLXML.mfd");
      objMapForce.Documents.OpenDocument(strSamplePath + "InspectionReport.mfd");
      objFolder2.AddActiveFile();

      // override code generation settings for this folder
      objFolder2.CodeGenSettings_UseDefault = false;
      objFolder2.CodeGenSettings_OutputFolder = strSamplePath + "SampleOutput"
      objFolder2.CodeGenSettings_Language = 1;      //C++

      // insert Web service project based on a wsdl file from the installed examples
      objProject.InsertWebService( strSamplePath + "TimeService/TimeService.wsdl",
                                   "{http://www.Nanonull.com/TimeService/}TimeService",
                           "TimeServiceSoap",
                           true );
      objProject.Save();
      if ( ! objProject.Saved )
         WScript.Echo("problem occurred when saving project");

      // dump project tree
      strGlobalText = '';
      PrintProjectTree( new Enumerator (objProject), ' ' )
      WScript.Echo( strGlobalText );
   }
   catch (err)
   { ERROR("while creating new project", err ); }
}

// ------------------------------------------------------------
// Generate code for a project's sub-tree. Mix default code
```

```
// generation parameters and overloaded parameters.
// -----------------------------------------------------------
function GenerateCodeForNewProject()
{
   // since the Web service project contains only initial mappings,
   // we generate code only for our custom folder.
   // code generation parameters from project are used for Folder1,
   // whereas Folder2 provides overwritten values.
   objFolder = objProject.Item(1);
   objFolder1.GenerateCode();
}

// ////////////////////// MAIN ////////////////////////////////

CreateGlobalObjects();
objMapForce.Visible = true;

LoadSampleProject();
CreateNewProject();
GenerateCodeForNewProject();

// uncomment to shut down application when script ends
// objMapForce.Visible = false;
```


(C) 2015-2021 Altova GmbH

17.7 Object Reference

This section provides reference to the objects of the MapForce COM API. The objects are described in a generic manner, since the API may be used with virtually any language that supports calling a COM object. For language-specific examples, see:

- Example C# Project
- Example Java Project
- JScript Examples

The API reference contains two main sections, each describing the interfaces and the enumeration types used in the API, respectively. The enumeration values contain both the string name and a numeric value. If your scripting environment does not support enumerations, use the number-values instead.

In .NET, for every interface of the MapForce COM automation interface, a .NET class exists with the same name. Also, COM types will be converted to the appropriate .NET type. For example, a type such as `Long` in the COM API would appear as `System.Int32` in .NET.

In Java, note the following syntax variations:

- **Classes and class names**. For every interface of the MapForce automation interface, a Java class exists with the name of the interface.
- **Method names**. Method names on the Java interface are the same as used on the COM interfaces, but start with a small letter to conform to Java naming conventions. To access COM properties, Java methods that prefix the property name with get and set can be used. If a property does not support write-access, no setter method is available. For example, for the `Name` property of the `Document` interface, the Java methods `getName` and `setName` are available.
- **Enumerations**. For every enumeration defined in the automation interface, a Java enumeration is defined with the same name and values.
- **Events and event handlers**. For every interface in the automation interface that supports events, a Java interface with the same name plus 'Event' is available. To simplify the overloading of single events, a Java class with default implementations for all events is provided. The name of this Java class is the name of the event interface plus 'DefaultHandler'. For example:

```
Application // Java class to access the application
ApplicationEvents // Events interface for the application
ApplicationEventsDefaultHandler // Default handler for "ApplicationEvents"
```

17.7.1 Interfaces

17.7.1.1 Application

The `Application` interface is the interface to a MapForce application object. It represents the main access point for the MapForce application itself. This interface is the starting point to do any further operations with MapForce or to retrieve or create other MapForce related automation objects. For information about creating an instance the Application object, see Accessing the API.

Properties to navigate the object model:

- Application
- Parent
- Options
- Project
- Documents

Application status:

- Visible
- Name
- Quit
- Status
- WindowHandle

MapForce designs:

- NewDocument
- OpenDocument
- OpenURL
- ActiveDocument

MapForce projects:

- NewProject
- OpenProject
- ActiveProject

MapForce code generation:

- HighlightSerializedMarker

Global resources:

- GlobalResourceConfig
- GlobalResourceFile

Version information:

- Edition
- IsAPISupported
- MajorVersion
- MinorVersion

Properties

Name	Description
ActiveDocument	Read-only. Returns the automation object of the currently active document. This property returns the same as Documents.ActiveDocument.
ActiveProject	Read-only. Returns the automation object of the currently active project.

(C) 2015-2021 Altova GmbH

Name	Description
Application	Read-only. Retrieves the application's top-level object.
Documents	Read-only. Returns a collection of all currently open documents.
Edition	Read-only. Returns the edition of the application, e.g. "Altova MapForce Enterprise Edition" for the enterprise edition.
GlobalResourceConfig	Gets or sets the name of the active global resource configuration file. By default, the file is called **GlobalResources.xml**. The configuration file can be renamed and saved to any location. You can therefore have multiple Global Resources XML files. However, only one of these Global Resources XML File can be active, per application, at one time, and only the definitions contained in this file will be available to the application.
GlobalResourceFile	Gets or sets the global resource definition file. By default, the file is called GlobalResources.xml.
IsAPISupported	Read-only. Returns true if the API is supported in this version of MapForce.
LibraryImports	Read-only. Gets a collection of imported libraries. In the MapForce graphical user interface, these correspond to entries from the **Manage Libraries** window, added at application level.
MajorVersion	Read-only. Gets the major version number of MapForce. The version is calculated starting from 1998, and is incremented by one every year. For example, the major version is "18" for the release 2016.
MinorVersion	Read-only. The minor version number of the product, e.g. 2 for 2006 R2 SP1.
Name	Read-only. The name of the application.
Options	Read-only. This property gives access to options that configure the generation of code.
Parent	Read-only. The parent object according to the object model.
ServicePackVersion	Read-only.

Name	Description
	The service pack version number of the product, e.g. 1 for 2016 R2 SP1.
Status	Read-only. The status of the application. It is one of the values of the ENUMApplicationStatus enumeration.
Visible	**True** if MapForce is displayed on the screen (though it might be covered by other applications or be iconized). **False** if MapForce is hidden. The default value for MapForce when automatically started due to a request from the automation server Application is false. In all other cases, the property is initialized to **true**. An application instance that is visible is said to be controlled by the user (and possibly by clients connected via the automation interface). It will only shut down due to an explicit user request. To shut down an application instance, set its visibility to false and clear all references to this instance within your program. The application instance will shut down automatically when no further COM clients are holding references to it.
WindowHandle	Read-only. Retrieves the application's Window Handle.

Methods

Name	Description
HighlightSerializedMarker	Use this method to highlight a location in a mapping file that has been previously serialized. If the corresponding document has not already been loaded, it will be loaded first. See Document.GenerateCodeEx for a method to retrieve a serialized marker.
NewDocument	Creates a new empty document. The newly opened document becomes the ActiveDocument. This method is a shortened form of Documents.NewDocument.
NewProject	Creates a new empty project. The current project is closed. The new project is accessible under ActiveProject.
NewWebServiceProject	Creates a new empty Web Service project. The new project is accessible under ActiveProject. This method is available in MapForce Enterprise Edition only.
OpenDocument	Loads a previously saved document file and continues working on it. The newly opened document becomes the ActiveDocument. This method is a shorter form of Documents.OpenDocument.

(C) 2015-2021 Altova GmbH

Name	Description
OpenProject	Opens an existing Mapforce project (*.mfp). The current project is closed. The newly opened project is accessible under ActiveProject.
OpenURL	Loads a previously saved document file from an URL location. Allows user name and password to be supplied.
Quit	Disconnects from MapForce to allow the application to shutdown. Calling this method is optional since MapForce keeps track of all external COM connections and automatically recognizes a disconnection. For more information on automatic shutdown see the Visible property.

Events

Name	Description
OnDocumentOpened	This event is triggered when an existing or new document is opened. The corresponding close event is Document.OnDocumentClosed.
OnProjectOpened	This event is triggered when an existing or new project is loaded into the application. The corresponding close event is Project.OnProjectClosed.
OnShutdown	This event is triggered when the application is shutting down.

17.7.1.1.1 Properties

17.7.1.1.1.1 ActiveDocument

Returns the automation object of the currently active document. This property returns the same as Documents.ActiveDocument.

Signature

```
ActiveDocument : Document
```

Errors

Error code	Description
1000	The application object is no longer valid.
1001	Invalid address for the return parameter was specified.

17.7.1.1.1.2 ActiveProject

Returns the automation object of the currently active project.

Signature

```
ActiveProject : Project
```

Errors

Error code	Description
1000	The application object is no longer valid.
1001	Invalid address for the return parameter was specified.

17.7.1.1.1.3 Application

Retrieves the application's top-level object.

Signature

```
Application : Application
```

Errors

Error code	Description
1000	The application object is no longer valid.
1001	Invalid address for the return parameter was specified.

17.7.1.1.1.4 Documents

Returns a collection of all currently open documents.

Signature

```
Documents : Documents
```

Errors

Error code	Description
1000	The application object is no longer valid.

(C) 2015-2021 Altova GmbH

Error code	Description
1001	Invalid address for the return parameter was specified.

17.7.1.1.1.5 Edition

Returns the edition of the application, e.g. "Altova MapForce Enterprise Edition" for the enterprise edition.

Signature

```
Edition : String
```

Errors

Error code	Description
1000	The application object is no longer valid.
1001	Invalid address for the return parameter was specified.

17.7.1.1.1.6 GlobalResourceConfig

Gets or sets the name of the active global resource configuration file. By default, the file is called **GlobalResources.xml**.

The configuration file can be renamed and saved to any location. You can therefore have multiple Global Resources XML files. However, only one of these Global Resources XML File can be active, per application, at one time, and only the definitions contained in this file will be available to the application.

Signature

```
GlobalResourceConfig : String
```

Errors

Error code	Description
1000	The application object is no longer valid.
1001	Invalid address for the return parameter was specified.

17.7.1.1.1.7 GlobalResourceFile

Gets or sets the global resource definition file. By default, the file is called GlobalResources.xml.

Signature

```
GlobalResourceFile : String
```

Errors

Error code	Description
1000	The application object is no longer valid.
1001	Invalid address for the return parameter was specified.

17.7.1.1.1.8 IsAPISupported

Returns true if the API is supported in this version of MapForce.

Signature

```
IsAPISupported : Boolean
```

Errors

Error code	Description
1001	Invalid address for the return parameter was specified.

17.7.1.1.1.9 LibraryImports

Gets a collection of imported libraries. In the MapForce graphical user interface, these correspond to entries from the **Manage Libraries** window, added at application level.

Signature

```
LibraryImports : LibraryImports
```

Errors

Error code	Description
1200	The object is no longer valid.

(C) 2015-2021 Altova GmbH

Error code	Description
1201	Invalid address for the return parameter was specified.

17.7.1.1.1.10 MajorVersion

Gets the major version number of MapForce. The version is calculated starting from 1998, and is incremented by one every year. For example, the major version is "18" for the release 2016.

Signature

```
MajorVersion : Long
```

Errors

Error code	Description
1000	The application object is no longer valid.
1001	Invalid address for the return parameter was specified.

17.7.1.1.1.11 MinorVersion

The minor version number of the product, e.g. 2 for 2006 R2 SP1.

Signature

```
MinorVersion : Long
```

Errors

Error code	Description
1000	The application object is no longer valid.
1001	Invalid address for the return parameter was specified.

17.7.1.1.1.12 Name

The name of the application.

Signature

```
Name : String
```

Errors

Error code	Description
1000	The application object is no longer valid.
1001	Invalid address for the return parameter was specified.

17.7.1.1.1.13 Options

This property gives access to options that configure the generation of code.

Signature

```
Options : Options
```

Errors

Error code	Description
1000	The application object is no longer valid.
1001	Invalid address for the return parameter was specified.

17.7.1.1.1.14 Parent

The parent object according to the object model.

Signature

```
Parent : Application
```

Errors

Error code	Description
1000	The application object is no longer valid.

(C) 2015-2021 Altova GmbH

Error code	Description
1001	Invalid address for the return parameter was specified.

17.7.1.1.1.15 ServicePackVersion

The service pack version number of the product, e.g. 1 for 2016 R2 SP1.

Signature

```
ServicePackVersion : Long
```

Errors

Error code	Description
1000	The application object is no longer valid.
1001	Invalid address for the return parameter was specified.

17.7.1.1.1.16 Status

The status of the application. It is one of the values of the `ENUMApplicationStatus` enumeration.

Signature

```
Status : ENUMApplicationStatus
```

Errors

Error code	Description
1001	Invalid address for the return parameter was specified.

17.7.1.1.1.17 Visible

True if MapForce is displayed on the screen (though it might be covered by other applications or be iconized).

False if MapForce is hidden. The default value for MapForce when automatically started due to a request from the automation server `Application` is false. In all other cases, the property is initialized to **true**.

An application instance that is visible is said to be controlled by the user (and possibly by clients connected via the automation interface). It will only shut down due to an explicit user request. To shut down an application instance, set its visibility to false and clear all references to this instance within your program. The application instance will shut down automatically when no further COM clients are holding references to it.

Signature

```
Visible : Boolean
```

Errors

Error code	Description
1000	The application object is no longer valid.
1001	Invalid address for the return parameter was specified.

17.7.1.1.1.18 WindowHandle

Retrieves the application's Window Handle.

Signature

```
WindowHandle : Long
```

Errors

Error code	Description
1000	The application object is no longer valid.
1001	Invalid address for the return parameter was specified.

17.7.1.1.2 Methods

17.7.1.1.2.1 HighlightSerializedMarker

Use this method to highlight a location in a mapping file that has been previously serialized. If the corresponding document has not already been loaded, it will be loaded first. See `Document.GenerateCodeEx` for a method to retrieve a serialized marker.

Signature

```
HighlightSerializedMarker(in i_strSerializedMarker:String) -> Void
```


(C) 2015-2021 Altova GmbH

Parameters

Name	Type	Description
i_strSerializedMarker	`String`	The `ErrorMarker` object to highlight. Use `ErrorMaker.Serialized` to obtain this value.

Errors

Error code	Description
1000	The application object is no longer valid.
1001	Invalid address for the return parameter was specified.
1007	The string passed in **i_strSerializedMarker** is not recognized as a serialized MapForce marker.
1008	The marker points to a location that is no longer valid.

17.7.1.1.2.2 NewDocument

Creates a new empty document. The newly opened document becomes the `ActiveDocument`. This method is a shortened form of `Documents.NewDocument`.

Signature

```
NewDocument() -> Document
```

Errors

Error code	Description
1000	The application object is no longer valid.
1001	Invalid address for the return parameter was specified.

17.7.1.1.2.3 NewProject

Creates a new empty project. The current project is closed. The new project is accessible under `ActiveProject`.

Signature

```
NewProject() -> Project
```

Errors

Error code	Description
1000	The application object is no longer valid.
1001	Invalid address for the return parameter was specified.

17.7.1.1.2.4 NewWebServiceProject

Creates a new empty Web Service project. The new project is accessible under `ActiveProject`. This method is available in MapForce Enterprise Edition only.

Signature

```
NewWebServiceProject() -> Project
```

Errors

Error code	Description
1004	Error creating new project.
1005	Wrong edition of MapForce.

17.7.1.1.2.5 OpenDocument

Loads a previously saved document file and continues working on it. The newly opened document becomes the `ActiveDocument`. This method is a shorter form of `Documents.OpenDocument`.

Signature

```
OpenDocument(in i_strFileName:String) -> Document
```

Parameters

Name	Type	Description
i_strFileName	`String`	The path of the document to open.

Errors

Error code	Description
1000	The application object is no longer valid.

(C) 2015-2021 Altova GmbH

Error code	Description
1001	Invalid address for the return parameter was specified.

17.7.1.1.2.6 OpenProject

Opens an existing Mapforce project (*.mfp). The current project is closed. The newly opened project is accessible under `ActiveProject`.

Signature

```
OpenProject(in i_strFileName:String) -> Project
```

Parameters

Name	Type	Description
i_strFileName	`String`	The path of the project to open.

Errors

Error code	Description
1000	The application object is no longer valid.
1001	Invalid address for the return parameter was specified.
1002	The supplied filename is not valid.

17.7.1.1.2.7 OpenURL

Loads a previously saved document file from an URL location. Allows user name and password to be supplied.

Signature

```
OpenURL(in strURL:String, in strUser:String, in strPassword:String) -> Void
```

Parameters

Name	Type	Description
strURL	`String`	The URL from which the document should be loaded.
strUser	`String`	The username required to access the URL.

Name	Type	Description
strPassword	`String`	The password required to access the URL.

Errors

Error code	Description
1000	The application object is no longer valid.
1001	Invalid address for the return parameter was specified.
1002	The supplied URL is not valid.
1006	Error while opening the URL file.

17.7.1.1.2.8 Quit

Disconnects from MapForce to allow the application to shutdown. Calling this method is optional since MapForce keeps track of all external COM connections and automatically recognizes a disconnection. For more information on automatic shutdown see the `Visible` property.

Signature

```
Quit() -> Void
```

Errors

Error code	Description
1000	The application object is no longer valid.
1001	Invalid address for the return parameter was specified.

17.7.1.1.3 Events

17.7.1.1.3.1 OnDocumentOpened

This event is triggered when an existing or new document is opened. The corresponding close event is `Document.OnDocumentClosed`.

Signature

```
OnDocumentOpened(in i_ipDocument:Document) : Void
```


(C) 2015-2021 Altova GmbH

17.7.1.1.3.2 OnProjectOpened

This event is triggered when an existing or new project is loaded into the application. The corresponding close event is `Project.OnProjectClosed`.

Signature

```
OnProjectOpened(in i_ipProject:Project) : Void
```

17.7.1.1.3.3 OnShutdown

This event is triggered when the application is shutting down.

Signature

```
OnShutdown : Void
```

17.7.1.2 AppOutputLine

Represents a message line. In contrast to `ErrorMarker`, its structure is more detailed and can contain a collection of child lines, therefore forming a tree of message lines.

Properties to navigate the object model:

- `Application`
- `Parent`

Line access:

- `GetLineSeverity`
- `GetLineSymbol`
- `GetLineText`
- `GetLineTextEx`
- `GetLineTextWithChildren`
- `GetLineTextWithChildrenEx`

A single `AppOutputLine` consists of one or more sub-lines. Sub-line access:

- `GetLineCount`

A sub-line consists of one or more cells. Cell access:

- `GetCellCountInLine`
- `GetCellIcon`
- `GetCellSymbol`
- `GetCellText`
- `GetCellTextDecoration`
- `GetIsCellText`

Below an AppOutputLine there can be zero, one, or more child lines which themselves are of type AppOutputLine, which thus form a tree structure.

Child lines access:

- ChildLines

Properties

Name	Description
Application	Read-only. Retrieves the application's top-level object.
ChildLines	Read-only. Returns a collection of the current line's direct child lines.
Parent	Read-only. The parent object according to the object model.

Methods

Name	Description
GetCellCountInLine	Gets the number of cells in the sub-line indicated by nLine in the current AppOutputLine.
GetCellIcon	Gets the icon of the cell indicated by nCell in the current AppOutputLine's sub-line indicated by nLine
GetCellSymbol	Gets the symbol of the cell indicated by nCell in the current AppOutputLine's sub-line indicated by nLine
GetCellText	Gets the text of the cell indicated by nCell in the current AppOutputLine's sub-line indicated by nLine
GetCellTextDecoration	Gets the decoration of the text cell indicated by nCell in the current AppOutputLine's sub-line indicated by nLine. It can be one of the ENUMAppOutputLine_TextDecoration values.
GetIsCellText	Returns true if the cell indicated by nCell in the current AppOutputLine's sub-line indicated by nLine is a text cell.
GetLineCount	Gets the number of sub-lines the current line consists of.
GetLineSeverity	Gets the severity of the line. It can be one of the ENUMAppOutputLine_Severity values.
GetLineSymbol	Gets the symbol assigned to the whole line.
GetLineText	Gets the contents of the line as text.
GetLineTextEx	Gets the contents of the line as text using the specified part and line separators.

(C) 2015-2021 Altova GmbH

Name	Description
GetLineTextWithChildren	Gets the contents of the line including all child and descendant lines as text.
GetLineTextWithChildrenEx	Gets the contents of the line including all child and descendant lines as text using the specified part, line, tab and item separators.

17.7.1.2.1 Properties

17.7.1.2.1.1 Application

Retrieves the application's top-level object.

Signature

```
Application : Application
```

Errors

Error code	Description
4100	The object is no longer valid.
4101	Invalid address for the return parameter was specified.

17.7.1.2.1.2 ChildLines

Returns a collection of the current line's direct child lines.

Signature

```
ChildLines : AppOutputLines
```

Errors

Error code	Description
4100	The object is no longer valid.
4101	Invalid address for the return parameter was specified.

17.7.1.2.1.3 Parent

The parent object according to the object model.

Signature

```
Parent : AppOutputLines
```

Errors

Error code	Description
4100	The object is no longer valid.
4101	Invalid address for the return parameter was specified.

17.7.1.2.2 Methods

17.7.1.2.2.1 GetCellCountInLine

Gets the number of cells in the sub-line indicated by `nLine` in the current `AppOutputLine`.

Signature

```
GetCellCountInLine(in nLine:Long) -> Long
```

Parameters

Name	Type	Description
nLine	`Long`	Specifies the zero-based index of the line.

Errors

Error code	Description
4100	The object is no longer valid.
4101	Invalid address for the return parameter was specified.

(C) 2015-2021 Altova GmbH

17.7.1.2.2.2 GetCellIcon (obsolete)

Gets the icon of the cell indicated by `nCell` in the current `AppOutputLine`'s sub-line indicated by `nLine`

Signature

```
GetCellIcon(in nLine:Long, in nCell:Long) -> Long
```

Parameters

Name	Type	Description
nLine	**Long**	
nCell	**Long**	

Errors

Error code	Description
4100	The object is no longer valid.
4101	Invalid address for the return parameter was specified.

17.7.1.2.2.3 GetCellSymbol

Gets the symbol of the cell indicated by `nCell` in the current `AppOutputLine`'s sub-line indicated by `nLine`

Signature

```
GetCellSymbol(in nLine:Long, in nCell:Long) -> AppOutputLineSymbol
```

Parameters

Name	Type	Description
nLine	**Long**	Specifies the zero-based index of the line.
nCell	**Long**	Specifies the zero-based index of the cell.

Errors

Error code	Description
4100	The object is no longer valid.

Error code	Description
4101	Invalid address for the return parameter was specified.

17.7.1.2.2.4 GetCellText

Gets the text of the cell indicated by `nCell` in the current `AppOutputLine`'s sub-line indicated by `nLine`

Signature

```
GetCellText(in nLine:Long, in nCell:Long) -> String
```

Parameters

Name	Type	Description
nLine	`Long`	Specifies the zero-based index of the line.
nCell	`Long`	Specifies the zero-based index of the cell.

Errors

Error code	Description
4100	The object is no longer valid.
4101	Invalid address for the return parameter was specified.

17.7.1.2.2.5 GetCellTextDecoration

Gets the decoration of the text cell indicated by `nCell` in the current `AppOutputLine`'s sub-line indicated by `nLine`. It can be one of the `ENUMAppOutputLine_TextDecoration` values.

Signature

```
GetCellTextDecoration(in nLine:Long, in nCell:Long) -> Long
```

Parameters

Name	Type	Description
nLine	`Long`	Specifies the zero-based index of the line.

(C) 2015-2021 Altova GmbH

Name	Type	Description
nCell	`Long`	Specifies the zero-based index of the cell.

Errors

Error code	Description
4100	The object is no longer valid.
4101	Invalid address for the return parameter was specified.

17.7.1.2.2.6 GetIsCellText

Returns true if the cell indicated by `nCell` in the current `AppOutputLine`'s sub-line indicated by `nLine` is a text cell.

Signature

```
GetIsCellText(in nLine:Long, in nCell:Long) -> Boolean
```

Parameters

Name	Type	Description
nLine	`Long`	Specifies the zero-based index of the line.
nCell	`Long`	Specifies the zero-based index of the cell.

Errors

Error code	Description
4100	The object is no longer valid.
4101	Invalid address for the return parameter was specified.

17.7.1.2.2.7 GetLineCount

Gets the number of sub-lines the current line consists of.

Signature

```
GetLineCount() -> Long
```

Errors

Error code	Description
4100	The object is no longer valid.
4101	Invalid address for the return parameter was specified.

17.7.1.2.2.8 GetLineSeverity

Gets the severity of the line. It can be one of the `ENUMAppOutputLine_Severity` values.

Signature

```
GetLineSeverity() -> Long
```

Errors

Error code	Description
4100	The object is no longer valid.
4101	Invalid address for the return parameter was specified.

17.7.1.2.2.9 GetLineSymbol

Gets the symbol assigned to the whole line.

Signature

```
GetLineSymbol() -> AppOutputLineSymbol
```

Errors

Error code	Description
4100	The object is no longer valid.

(C) 2015-2021 Altova GmbH

Error code	Description
4101	Invalid address for the return parameter was specified.

17.7.1.2.2.10 GetLineText

Gets the contents of the line as text.

Signature

```
GetLineText() -> String
```

Errors

Error code	Description
4100	The object is no longer valid.
4101	Invalid address for the return parameter was specified.

17.7.1.2.2.11 GetLineTextEx

Gets the contents of the line as text using the specified part and line separators.

Signature

```
GetLineTextEx(in psTextPartSeperator:String, in psLineSeperator:String) -> String
```

Parameters

Name	Type	Description
psTextPartSeperator	`String`	
psLineSeperator	`String`	

Errors

Error code	Description
4100	The object is no longer valid.
4101	Invalid address for the return parameter was specified.

17.7.1.2.2.12 GetLineTextWithChildren

Gets the contents of the line including all child and descendant lines as text.

Signature

```
GetLineTextWithChildren() -> String
```

Errors

Error code	Description
4100	The object is no longer valid.
4101	Invalid address for the return parameter was specified.

17.7.1.2.2.13 GetLineTextWithChildrenEx

Gets the contents of the line including all child and descendant lines as text using the specified part, line, tab and item separators.

Signature

```
GetLineTextWithChildrenEx(in psPartSep:String, in psLineSep:String, in psTabSep:String,
in psItemSep:String) -> String
```

Parameters

Name	Type	Description
psPartSep	String	
psLineSep	String	
psTabSep	String	
psItemSep	String	

Errors

Error code	Description
4100	The object is no longer valid.
4101	Invalid address for the return parameter was specified.

(C) 2015-2021 Altova GmbH

17.7.1.3 AppOutputLines

Represents a collection of `AppOutputLine` message lines.

Properties to navigate the object model:

- `Application`
- `Parent`

Iterating through the collection:

- `Count`
- `Item`

Properties

Name	Description
`Application`	Read-only. Retrieves the application's top-level object.
`Count`	Read-only. Retrieves the number of lines in the collection.
`Item`	Read-only. Retrieves the line at index `n` from the collection. Indices start with 1.
`Parent`	Read-only. The parent object according to the object model.

17.7.1.3.1 Properties

17.7.1.3.1.1 Application

Retrieves the application's top-level object.

Signature

```
Application : Application
```

Errors

Error code	Description
4000	The object is no longer valid.
4001	Invalid address for the return parameter was specified.

17.7.1.3.1.2 Count

Retrieves the number of lines in the collection.

Signature

```
Count : Integer
```

Errors

Error code	Description
4000	The object is no longer valid.
4001	Invalid address for the return parameter was specified.

17.7.1.3.1.3 Item

Retrieves the line at index `n` from the collection. Indices start with 1.

Signature

```
Item(in n:Integer) : AppOutputLine
```

Errors

Error code	Description
4000	The object is no longer valid.
4001	Invalid address for the return parameter was specified.

17.7.1.3.1.4 Parent

The parent object according to the object model.

Signature

```
Parent : AppOutputLine
```

Errors

Error code	Description
4000	The object is no longer valid.

(C) 2015-2021 Altova GmbH

Error code	Description
4001	Invalid address for the return parameter was specified.

17.7.1.4 AppOutputLineSymbol

An `AppOutputLineSymbol` represents a link in an `AppOutputLine` message line which can be clicked in the MapForce Messages window. It is applied to a cell of an `AppOutputLine` or to the whole line itself.

Properties to navigate the object model:

- `Application`
- `Parent`

Access to AppOutputLineSymbol methods:

- `GetSymbolHREF`
- `GetSymbolID`
- `IsSymbolHREF`

Properties

Name	Description
`Application`	Read-only. Retrieves the application's top-level object.
`Parent`	Read-only. The parent object according to the object model.

Methods

Name	Description
`GetSymbolHREF`	If the symbol is of type URL, returns the URL as a string.
`GetSymbolID`	Gets the ID of the symbol.
`IsSymbolHREF`	Returns true if the symbol is of kind URL.

17.7.1.4.1 Properties

17.7.1.4.1.1 Application

Retrieves the application's top-level object.

Signature

```
Application : Application
```

Errors

Error code	Description
4200	The object is no longer valid.
4201	Invalid address for the return parameter was specified.

17.7.1.4.1.2 Parent

The parent object according to the object model.

Signature

```
Parent : Application
```

Errors

Error code	Description
4200	The object is no longer valid.
4201	Invalid address for the return parameter was specified.

(C) 2015-2021 Altova GmbH

17.7.1.4.2 Methods

17.7.1.4.2.1 GetSymbolHREF

If the symbol is of type URL, returns the URL as a string.

Signature

```
GetSymbolHREF() -> String
```

Errors

Error code	Description
4200	The object is no longer valid.
4201	Invalid address for the return parameter was specified.

17.7.1.4.2.2 GetSymbolID

Gets the ID of the symbol.

Signature

```
GetSymbolID() -> Long
```

Errors

Error code	Description
4200	The object is no longer valid.
4201	Invalid address for the return parameter was specified.

17.7.1.4.2.3 IsSymbolHREF

Returns true if the symbol is of kind URL.

Signature

```
IsSymbolHREF() -> Boolean
```

Errors

Error code	Description
4200	The object is no longer valid.
4201	Invalid address for the return parameter was specified.

17.7.1.5 Component

A `Component` represents a MapForce component.

To navigate the control, use the `Application` and `Parent` properties.

Component properties:

- `HasIncomingConnections`
- `HasOutgoingConnections`
- `CanChangeInputInstanceFile`
- `CanChangeOutputInstanceFile`
- `ComponentName.`
- `ID`
- `IsParameterInputRequired`
- `IsParameterSequence`
- `Name`
- `Preview`
- `Schema`
- `SubType`
- `Type`

Instance related properties:

- `InputInstanceFile`
- `OutputInstanceFile`

Datapoints:

- `GetRootDatapoint`

Execution:

- `GenerateOutput`

Properties

Name	Description
`Application`	Read-only. Retrieves the application's top-level object.
`CanChangeInputInstanceFile`	Read-only.

(C) 2015-2021 Altova GmbH

Name	Description
	Indicates if the input instance file name can be changed. Returns false if the component has a filename node and this node has a connection on its left (input) side, otherwise returns true. If the component does not have a filename node, false is returned.
CanChangeOutputInstanceFile	Read-only. Indicates if the output instance file name can be changed. Returns false if the component has a filename node and this node has a connection on its left (input) side, otherwise returns true. If the component does not have a filename node, false is returned.
ComponentName	Gets or sets the component's name.
HasIncomingConnections	Read-only. Indicates if the component has any incoming connections (on its left side) not including the filename node. An incoming connection on the filename node does not have any effect on the returned value.
HasOutgoingConnections	Read-only. Indicates if the component has any outgoing connections (on its right side).
ID	Read-only. Retrieves the component ID.
InputInstanceFile	Gets or sets the component's input instance file.
IsParameterInputRequired	Gets or sets, if the input parameter component requires an ingoing connection on the function call component of the user-defined function this input parameter component is in. This property works only for components, which are input parameter components.
IsParameterSequence	Gets or sets, if the input or output parameter component supports sequences. This property works only for components, which are input or output parameter components.
Name	Read-only. Gets the component's name.
OutputInstanceFile	Gets or sets the component's output instance file. Trying to access the `OutputInstanceFile` of a component via the API does not return any data if the "File" connector of the component has been connected to another item in the mapping.
Parent	Read-only. The parent object according to the object model.

Name	Description
Preview	Gets or sets if the component is the current preview component. This property works only for components, which are target components in the document's main mapping. Only one target component in the main mapping can be the preview component at any time. When setting this property, it is only possible to set it to true. This then will also implicitly set the `Preview` property of all other components to false. If there is just a single target component in the main mapping, it is also the preview component.
Schema	Read-only. Retrieves the component's schema file name.
SubType	Read-only. Retrieves the component's subtype.
Type	Read-only. Retrieves the component's type.
UsageKind	Read-only. Retrieves the component's usage kind.

Methods

Name	Description
GenerateOutput	Generates the output file(s) defined in the mapping for the current component only, using a MapForce internal mapping language. The name(s) of the output file(s) are defined as property of the current component which is the output item in the mapping for this generation process.
GetRootDatapoint	Gets a root datapoint on the left (input) or right (output) side of a component. To access children and descendants, the `Datapoint` object provides further methods.

(C) 2015-2021 Altova GmbH

17.7.1.5.1 Properties

17.7.1.5.1.1 Application

Retrieves the application's top-level object.

Signature

```
Application : Application
```

Errors

Error code	Description
1200	The object is no longer valid.
1201	Invalid address for the return parameter was specified.

17.7.1.5.1.2 CanChangeInputInstanceFile

Indicates if the input instance file name can be changed.

Returns false if the component has a filename node and this node has a connection on its left (input) side, otherwise returns true. If the component does not have a filename node, false is returned.

Signature

```
CanChangeInputInstanceFile : Boolean
```

Errors

Error code	Description
1200	The object is no longer valid.
1201	Invalid address for the return parameter was specified.

17.7.1.5.1.3 CanChangeOutputInstanceFile

Indicates if the output instance file name can be changed.

Returns false if the component has a filename node and this node has a connection on its left (input) side, otherwise returns true.
If the component does not have a filename node, false is returned.

Signature

```
CanChangeOutputInstanceFile : Boolean
```

Errors

Error code	Description
1200	The object is no longer valid.
1201	Invalid address for the return parameter was specified.

17.7.1.5.1.4 ComponentName

Gets or sets the component's name.

Signature

```
ComponentName : String
```

Errors

Error code	Description
1200	The object is no longer valid.
1201	Invalid address for the return parameter was specified.
1246	The component does not support setting its name.
1247	Invalid component name.

17.7.1.5.1.5 HasIncomingConnections

Indicates if the component has any incoming connections (on its left side) not including the filename node. An incoming connection on the filename node does not have any effect on the returned value.

Signature

```
HasIncomingConnections : Boolean
```

Errors

Error code	Description
1200	The object is no longer valid.

(C) 2015-2021 Altova GmbH

Error code	Description
1201	Invalid address for the return parameter was specified.

17.7.1.5.1.6 HasOutgoingConnections

Indicates if the component has any outgoing connections (on its right side).

Signature

```
HasOutgoingConnections : Boolean
```

Errors

Error code	Description
1200	The object is no longer valid.
1201	Invalid address for the return parameter was specified.

17.7.1.5.1.7 ID

Retrieves the component ID.

Signature

```
ID : Long
```

Errors

Error code	Description
1200	The object is no longer valid.
1201	Invalid address for the return parameter was specified.

17.7.1.5.1.8 InputInstanceFile

Gets or sets the component's input instance file.

Signature

```
InputInstanceFile : String
```

Errors

Error code	Description
1200	The object is no longer valid.
1201	Invalid address for the return parameter was specified.

17.7.1.5.1.9 IsParameterInputRequired

Gets or sets, if the input parameter component requires an ingoing connection on the function call component of the user-defined function this input parameter component is in. This property works only for components, which are input parameter components.

Signature

```
IsParameterInputRequired : Boolean
```

Errors

Error code	Description
1200	The object is no longer valid.
1201	Invalid address for the return parameter was specified.
1232	This operation works only for an input parameter component.
1240	Changing the document not allowed. It is read-only.

17.7.1.5.1.10 IsParameterSequence

Gets or sets, if the input or output parameter component supports sequences. This property works only for components, which are input or output parameter components.

Signature

```
IsParameterSequence : Boolean
```


(C) 2015-2021 Altova GmbH

Errors

Error code	Description
1200	The object is no longer valid.
1201	Invalid address for the return parameter was specified.
1233	This operation works only for an input or output parameter component.
1240	Changing the document not allowed. It is read-only.

17.7.1.5.1.11 Name

Gets the component's name.

Signature

```
Name : String
```

Errors

Error code	Description
1200	The object is no longer valid.
1201	Invalid address for the return parameter was specified.

17.7.1.5.1.12 OutputInstanceFile

Gets or sets the component's output instance file.

Trying to access the `OutputInstanceFile` of a component via the API does not return any data if the "File" connector of the component has been connected to another item in the mapping.

Signature

```
OutputInstanceFile : String
```

Errors

Error code	Description
1200	The object is no longer valid.
1201	Invalid address for the return parameter was specified.

17.7.1.5.1.13 Parent

The parent object according to the object model.

Signature

```
Parent : Mapping
```

Errors

Error code	Description
1200	The object is no longer valid.
1201	Invalid address for the return parameter was specified.

17.7.1.5.1.14 Preview

Gets or sets if the component is the current preview component.

This property works only for components, which are target components in the document's main mapping. Only one target component in the main mapping can be the preview component at any time.

When setting this property, it is only possible to set it to true. This then will also implicitly set the `Preview` property of all other components to false.

If there is just a single target component in the main mapping, it is also the preview component.

Signature

```
Preview : Boolean
```

Errors

Error code	Description
1200	The object is no longer valid.
1201	Invalid address for the return parameter was specified.
1234	Only a target component in the main mapping can be set as preview component.
1235	A component cannot be set as non-preview component. Set another component as preview component instead.

(C) 2015-2021 Altova GmbH

17.7.1.5.1.15 Schema

Retrieves the component's schema file name.

Signature

```
Schema : String
```

Errors

Error code	Description
1200	The object is no longer valid.
1201	Invalid address for the return parameter was specified.

17.7.1.5.1.16 SubType

Retrieves the component's subtype.

Signature

```
SubType : ENUMComponentSubType
```

Errors

Error code	Description
1200	The object is no longer valid.
1201	Invalid address for the return parameter was specified.

17.7.1.5.1.17 Type

Retrieves the component's type.

Signature

```
Type : ENUMComponentType
```

17.7.1.5.1.18 UsageKind

Retrieves the component's usage kind.

Signature

```
UsageKind : ENUMComponentUsageKind
```

Errors

Error code	Description
1200	The object is no longer valid.
1201	Invalid address for the return parameter was specified.

17.7.1.5.2 Methods

17.7.1.5.2.1 GenerateOutput

Generates the output file(s) defined in the mapping for the current component only, using a MapForce internal mapping language. The name(s) of the output file(s) are defined as property of the current component which is the output item in the mapping for this generation process.

Signature

```
GenerateOutput(out pbError:Boolean) -> AppOutputLines
```

Parameters

Name	Type	Description
pbError	Boolean	This is an output-only parameter. You will receive a value only if the calling language supports output parameters. If not, the value you pass here will remain unchanged when the function has finished.

Errors

Error code	Description
1200	The object is no longer valid.
1201	Invalid address for the return parameter was specified.

(C) 2015-2021 Altova GmbH

Error code	Description
1248	Generating output is only supported when the graphical user interface is visible.

17.7.1.5.2.2 GetRootDatapoint

Gets a root datapoint on the left (input) or right (output) side of a component. To access children and descendants, the `Datapoint` object provides further methods.

Signature

```
GetRootDatapoint(in side:ENUMComponentDatapointSide, in strNamespace:String, in
strLocalName:String, in strParameterName:String) -> Datapoint
```

Parameters

Name	Type	Description
side	ENUMComponentDatapointSide	The side parameter indicates if an input, or output, datapoint of a component is to be retrieved.
strNamespace	String	The specified namespace and local name, indicate the specific name of the node whose datapoint is to be retrieved. For components with structural information such as schema components, you will have to provide the namespace together with the local name, or you can just pass an empty string for the namespace. File-based components like the schema component contain a special node on their root, the filename node. There, `GetRootDatapoint` can only find the filename node. You will have to pass namespace `"http://www.altova.com/mapforce"` and local name `"FileInstance"` to retrieve a datapoint of this node.
strLocalName	String	See above.
strParameterName	String	The specified parameter name should be an empty string unless the component in question is a function call component. Since a user-defined function might

Name	Type	Description
		contain input or output parameters of the same structure, the function call component calling this user-defined function can have more than one root node with an identical namespace and local name. They will then differ only by their parameter names, which are in fact the names of the according parameter components in the user-defined function mapping itself. It is not mandatory to specify the parameter name, though. In that case, the method will return the first root datapoint matching the specified namespace and local name.

Errors

Error code	Description
1200	The object is no longer valid.
1201	Invalid address for the return parameter was specified.
1248	Datapoint not found.

17.7.1.6 Components

Represents a collection of `Component` objects.

To navigate the object model, use the following properties:

- `Application`
- `Parent`

To iterate through the collection:

- `Count`
- `Item`

(C) 2015-2021 Altova GmbH

Properties

Name	Description
Application	Read-only. Retrieves the application's top-level object.
Count	Read-only. Retrieves the number of components in the collection.
Item	Read-only. Retrieves the component at index `n` from the collection. Indices start with 1.
Parent	Read-only. The parent object according to the object model.

17.7.1.6.1 Properties

17.7.1.6.1.1 Application

Retrieves the application's top-level object.

Signature

```
Application : Application
```

Errors

Error code	Description
1200	The object is no longer valid.
1201	Invalid address for the return parameter was specified.

17.7.1.6.1.2 Count

Retrieves the number of components in the collection.

Signature

```
Count : Integer
```

Errors

Error code	Description
1200	The object is no longer valid.
1201	Invalid address for the return parameter was specified.

17.7.1.6.1.3 Item

Retrieves the component at index `n` from the collection. Indices start with 1.

Signature

```
Item(in n:Integer) : Component
```

Errors

Error code	Description
1200	The object is no longer valid.
1201	Invalid address for the return parameter was specified.

17.7.1.6.1.4 Parent

The parent object according to the object model.

Signature

```
Parent : Mapping
```

Errors

Error code	Description
1200	The object is no longer valid.
1201	Invalid address for the return parameter was specified.

17.7.1.7 Connection

A `Connection` object represents a connector between two components.

To navigate the object model, use the following properties:

(C) 2015-2021 Altova GmbH

- Application
- Parent

To get or set the connection's type, use ConnectionType.

Properties

Name	Description
Application	Read-only. Retrieves the application's top-level object.
ConnectionType	Gets or sets the connection's type.
Parent	Read-only. The parent object according to the object model.

17.7.1.7.1 Properties

17.7.1.7.1.1 Application

Retrieves the application's top-level object.

Signature

```
Application : Application
```

Errors

Error code	Description
2100	The object is no longer valid.
2101	Invalid address for the return parameter was specified.

17.7.1.7.1.2 ConnectionType

Gets or sets the connection's type.

Signature

```
ConnectionType : ENUMConnectionType
```

Errors

Error code	Description
2100	The object is no longer valid.
2101	Invalid address for the return parameter was specified.
2102	Changing the document is not allowed. It is read-only.
2103	Failed changing connection type.

17.7.1.7.1.3 Parent

The parent object according to the object model.

Signature

```
Parent : Mapping
```

Errors

Error code	Description
2100	The object is no longer valid.
2101	Invalid address for the return parameter was specified.

17.7.1.8 Datapoint

A `Datapoint` object represents an input or output icon of a component.

Properties

Name	Description
Application	Read-only. Retrieves the application's top-level object.
Parent	Read-only. The parent object according to the object model.

(C) 2015-2021 Altova GmbH

Methods

Name	Description
GetChild	Scans for a direct child datapoint of the current datapoint, by namespace and local name. If a schema component has elements that contain mixed content, each displays an additional child node, the so-called **text()** node. To retrieve a datapoint of a **text()** node, you will have to pass an empty string in `strNamespace` as well as `"#text"` in `strLocalName` and `eSearchDatapointElement` in `searchFlags`.

17.7.1.8.1 Properties

17.7.1.8.1.1 Application

Retrieves the application's top-level object.

Signature

```
Application : Application
```

Errors

Error code	Description
2000	The object is no longer valid.
2001	Invalid address for the return parameter was specified.

17.7.1.8.1.2 Parent

The parent object according to the object model.

Signature

```
Parent : Component
```

Errors

Error code	Description
2000	The object is no longer valid.

Error code	Description
2001	Invalid address for the return parameter was specified.

17.7.1.8.2 Methods

17.7.1.8.2.1 GetChild

Scans for a direct child datapoint of the current datapoint, by namespace and local name.

If a schema component has elements that contain mixed content, each displays an additional child node, the so-called **text()** node. To retrieve a datapoint of a **text()** node, you will have to pass an empty string in `strNamespace` as well as `"#text"` in `strLocalName` and `eSearchDatapointElement` in `searchFlags`.

Signature

```
GetChild(in strNamespace:String, in strLocalName:String, in
searchFlags:ENUMSearchDatapointFlags) -> Datapoint
```

Parameters

Name	Type	Description
strNamespace	`String`	The namespace of the direct child datapoint.
strLocalName	`String`	The name of the direct child datapoint.
searchFlags	`ENUMSearchDatapointFlags`	Search flags can be passed as combination of values (combined using binary OR) of the `ENUMSearchDatapointFlags` enumeration.

Errors

Error code	Description
2000	The object is no longer valid.
2001	Invalid address for the return parameter was specified.
2002	Datapoint not found.

(C) 2015-2021 Altova GmbH

17.7.1.9 Document

A `Document` object represents a MapForce document (a loaded MFD file). A document contains a main mapping and zero or more local user-defined-function mappings.

To navigate the object model, use the following properties:

- `Application`
- `Parent`

For file handling, use:

- `Activate`
- `Close`
- `FullName`
- `Name`
- `Path`
- `Saved`
- `Save`
- `SaveAs`

For mapping handling, use:

- `MainMapping`
- `Mappings`
- `CreateUserDefinedFunction`

For component handling, use:

- `FindComponentByID`

For code generation, use:

- `OutputSettings_ApplicationName`
- `JavaSettings_BasePackageName`
- `GenerateCHashCode`
- `GenerateCodeEx`
- `GenerateCppCode`
- `GenerateJavaCode`
- `GenerateXQuery`
- `GenerateXSLT`
- `GenerateXSLT2`
- `GenerateXSLT3`
- `HighlightSerializedMarker`

For mapping execution, use:

- `GenerateOutput`
- `GenerateOutputEx`

View access:

- `MapForceView`

Obsolete:

- OutputSettings_Encoding

Properties

Name	Description
Application	Read-only. Retrieves the application's top-level object.
FullName	Path and name of the document file.
JavaSettings_BasePackageName	Sets or retrieves the base package name used when generating Java code. In the MapForce graphical user interface, this setting is available in the **Mapping Settings** dialog box (right-click the mapping and select **Mapping Settings** from the context menu).
LibraryImports	Read-only. Gets a collection of imported libraries. In the MapForce graphical user interface, these correspond to entries from the **Manage Libraries** window, added at document level.
MainMapping	Read-only. Retrieves the main mapping of the document.
MapForceView	Read-only. This property gives access to functionality specific to the MapForce view.
Mappings	Read-only. Returns a collection of mappings contained in the document.
Name	Read-only. Name of the document file without file path.
OutputSettings_ApplicationName	Sets or retrieves the application name available in the **Mapping Settings** dialog box (To display this dialog box in MapForce, right-click the mapping and select **Mapping Settings** from the context menu).
OutputSettings_Encoding	This property is no longer supported. Mapping output encoding settings do no longer exist. Components have individual output encoding settings.
Parent	Read-only. The parent object according to the object model.
Path	Read-only. Path of the document file without name.
Saved	Read-only. **True** if the document was not modified since the last save operation, **false** otherwise.

(C) 2015-2021 Altova GmbH

Methods

Name	Description
Activate	Makes this document the active document.
Close	Closes the document without saving.
CreateUserDefinedFunction	Creates a user defined function in the current document.
FindComponentByID	Searches in the whole document, also all its mappings, for the component with the specified id.
GenerateCHashCode	Generates C# code that will perform the mapping. Uses the properties defined in `Application.Options` to configure code generation.
GenerateCodeEx	Generates code that will perform the mapping. The parameter **i_nLanguage** specifies the target language. The method returns an object that can be used to enumerate all messages created by the code generator. These are the same messages that get displayed in the Messages window of MapForce.
GenerateCppCode	Generates C++ code that will perform the mapping. Uses the properties defined in `Application.Options` to configure code generation.
GenerateJavaCode	Generates Java code that will perform the mapping. Uses the properties defined in `Application.Options` to configure code generation.
GenerateOutput	Generates all output files defined in the mapping using a MapForce internal mapping language. The names of the output files are defined as properties of the output items in the mapping. **Note:** This method can only be used when the MapForce (running as a COM server) main window is visible, or is embedded with a graphical user interface. If the method is called while MapForce is not visible, then an error will occur.
GenerateOutputEx	Generates all output files defined in the mapping using a MapForce internal mapping language. The names of the output files are defined as properties of the output items in the mapping. This method is identical to `GenerateOutput` except for its return value containing the resulting messages, warnings and errors arranged as trees of `AppOutputLines`. **Note:** This method can only be used when the MapForce (running as a COM server) main window is visible, or is embedded with a graphical user interface. If the method is called while MapForce is not visible, then an error will occur.

Name	Description
GenerateXQuery	Generates mapping code as XQuery. Uses the properties defined in `Application.Options` to configure code generation.
GenerateXSLT	Generates mapping code as XSLT. Uses the properties defined in `Application.Options` to configure code generation.
GenerateXSLT2	Generates mapping code as XSLT2. Uses the properties defined in `Application.Options` to configure code generation.
GenerateXSLT3	Generates XSLT 3.0 mapping code. Uses the properties defined in `Application.Options` to configure code generation.
HighlightSerializedMarker	Use this method to highlight a location in a mapping file that has been previously serialized. If the corresponding document is not already loaded, it will be loaded first. See `GenerateCodeEx` for a method to retrieve a serialized marker.
Save	Saves the document to the file defined by `Document.FullName`.
SaveAs	Saves the document to the specified file name, and sets `Document.FullName` to this value if the save operation was successful.

Events

Name	Description
OnDocumentClosed	This event is triggered when a document is closed. The document object passed into the event handler should not be accessed. The corresponding open event is `Application.OnDocumentOpened`.
OnModifiedFlagChanged	This event is triggered when a document's modification status changes.

(C) 2015-2021 Altova GmbH

17.7.1.9.1 Properties

17.7.1.9.1.1 Application

Retrieves the application's top-level object.

Signature

```
Application : Application
```

Errors

Error code	Description
1200	The object is no longer valid.
1201	Invalid address for the return parameter was specified.

17.7.1.9.1.2 FullName

Path and name of the document file.

Signature

```
FullName : String
```

Errors

Error code	Description
1200	The object is no longer valid.
1201	Invalid address for the return parameter was specified.

17.7.1.9.1.3 JavaSettings_BasePackageName

Sets or retrieves the base package name used when generating Java code. In the MapForce graphical user interface, this setting is available in the **Mapping Settings** dialog box (right-click the mapping and select **Mapping Settings** from the context menu).

Signature

```
JavaSettings_BasePackageName : String
```

Errors

Error code	Description
1200	The object is no longer valid.
1201	Invalid address for the return parameter was specified.

17.7.1.9.1.4 LibraryImports

Gets a collection of imported libraries. In the MapForce graphical user interface, these correspond to entries from the **Manage Libraries** window, added at document level.

Signature

```
LibraryImports : LibraryImports
```

Errors

Error code	Description
1200	The object is no longer valid.
1201	Invalid address for the return parameter was specified.

17.7.1.9.1.5 MainMapping

Retrieves the main mapping of the document.

Signature

```
MainMapping : Mapping
```

Errors

Error code	Description
1200	The object is no longer valid.
1201	Invalid address for the return parameter was specified.

(C) 2015-2021 Altova GmbH

17.7.1.9.1.6 MapForceView

This property gives access to functionality specific to the MapForce view.

Signature

```
MapForceView : MapForceView
```

Errors

Error code	Description
1200	The object is no longer valid.
1201	Invalid address for the return parameter was specified.

17.7.1.9.1.7 Mappings

Returns a collection of mappings contained in the document.

Signature

```
Mappings : Mappings
```

Errors

Error code	Description
1200	The object is no longer valid.
1201	Invalid address for the return parameter was specified.

17.7.1.9.1.8 Name

Name of the document file without file path.

Signature

```
Name : String
```

Errors

Error code	Description
1200	The object is no longer valid.

Error code	Description
1201	Invalid address for the return parameter was specified.

17.7.1.9.1.9 OutputSettings_ApplicationName

Sets or retrieves the application name available in the **Mapping Settings** dialog box (To display this dialog box in MapForce, right-click the mapping and select **Mapping Settings** from the context menu).

Signature

```
OutputSettings_ApplicationName : String
```

Errors

Error code	Description
1200	The object is no longer valid.
1201	Invalid address for the return parameter was specified.

17.7.1.9.1.10 OutputSettings_Encoding (obsolete)

This property is no longer supported. Mapping output encoding settings do no longer exist. Components have individual output encoding settings.

Signature

```
OutputSettings_Encoding : String
```

17.7.1.9.1.11 Parent

The parent object according to the object model.

Signature

```
Parent : Documents
```

Errors

Error code	Description
1200	The object is no longer valid.
1201	Invalid address for the return parameter was specified.

(C) 2015-2021 Altova GmbH

17.7.1.9.1.12 Path

Path of the document file without name.

Signature

```
Path : String
```

Errors

Error code	Description
1200	The object is no longer valid.
1201	Invalid address for the return parameter was specified.

17.7.1.9.1.13 Saved

True if the document was not modified since the last save operation, **false** otherwise.

Signature

```
Saved : Boolean
```

Errors

Error code	Description
1200	The object is no longer valid.
1201	Invalid address for the return parameter was specified.

17.7.1.9.2 Methods

17.7.1.9.2.1 Activate

Makes this document the active document.

Signature

```
Activate() -> Void
```

Errors

Error code	Description
1200	The object is no longer valid.

17.7.1.9.2.2 Close

Closes the document without saving.

Signature

```
Close() -> Void
```

Errors

Error code	Description
1200	The object is no longer valid.
1201	Invalid address for the return parameter was specified.

17.7.1.9.2.3 CreateUserDefinedFunction

Creates a user defined function in the current document.

Signature

```
CreateUserDefinedFunction(in strFunctionName:String, in strLibraryName:String, in
strSyntax:String, in strDetails:String, in bInlinedUse:Boolean) -> Mapping
```

Parameters

Name	Type	Description
strFunctionName	`String`	The name of the function.
strLibraryName	`String`	The name of the library to which this function belongs.
strSyntax	`String`	A string that describes the syntax of this function (this is for information purpose only).
strDetails	`String`	A description of this function.

(C) 2015-2021 Altova GmbH

Name	Type	Description
bInlinedUse	`Boolean`	Boolean flag that specifies if the function has inlined use.

Errors

Error code	Description
1200	The object is no longer valid.
1201	Invalid address for the return parameter was specified.
1208	Failed creating user-defined function.
1209	Changing the document not allowed. It is read-only.

17.7.1.9.2.4 FindComponentByID

Searches in the whole document, also all its mappings, for the component with the specified id.

Signature

```
FindComponentByID(in nID:Unsigned Long) -> Component
```

Parameters

Name	Type	Description
nID	`Unsigned Long`	The ID of the component to search for.

Errors

Error code	Description
1200	The object is no longer valid.
1201	Invalid address for the return parameter was specified.

17.7.1.9.2.5 GenerateCHashCode

Generates C# code that will perform the mapping. Uses the properties defined in `Application.Options` to configure code generation.

Signature

```
GenerateCHashCode() -> Void
```

Errors

Error code	Description
1200	The object is no longer valid.
1201	Invalid address for the return parameter was specified.
1205	Error during code generation.

17.7.1.9.2.6 GenerateCodeEx

Generates code that will perform the mapping. The parameter **i_nLanguage** specifies the target language. The method returns an object that can be used to enumerate all messages created by the code generator. These are the same messages that get displayed in the Messages window of MapForce.

Signature

```
GenerateCodeEx(in i_nLanguage:ENUMProgrammingLanguage) -> ErrorMarkers
```

Parameters

Name	Type	Description
i_nLanguage	ENUMProgrammingLanguage	Specifies the target code generation language.

Errors

Error code	Description
1200	The object is no longer valid.
1201	Invalid address for the return parameter was specified.
1205	Error during code generation.

17.7.1.9.2.7 GenerateCppCode

Generates C++ code that will perform the mapping. Uses the properties defined in `Application.Options` to configure code generation.

(C) 2015-2021 Altova GmbH

Signature

```
GenerateCppCode() -> Void
```

Errors

Error code	Description
1200	The object is no longer valid.
1201	Invalid address for the return parameter was specified.
1205	Error during code generation.

17.7.1.9.2.8 GenerateJavaCode

Generates Java code that will perform the mapping. Uses the properties defined in `Application.Options` to configure code generation.

Signature

```
GenerateJavaCode() -> Void
```

Errors

Error code	Description
1200	The object is no longer valid.
1201	Invalid address for the return parameter was specified.
1205	Error during code generation.

17.7.1.9.2.9 GenerateOutput

Generates all output files defined in the mapping using a MapForce internal mapping language. The names of the output files are defined as properties of the output items in the mapping.

Note: This method can only be used when the MapForce (running as a COM server) main window is visible, or is embedded with a graphical user interface. If the method is called while MapForce is not visible, then an error will occur.

Signature

```
GenerateOutput() -> Void
```

Errors

Error code	Description
1200	The object is no longer valid.
1201	Invalid address for the return parameter was specified.
1206	Error during execution of mapping algorithm.
1210	Generating output is only supported when the graphical user interface is visible.

17.7.1.9.2.10 GenerateOutputEx

Generates all output files defined in the mapping using a MapForce internal mapping language. The names of the output files are defined as properties of the output items in the mapping. This method is identical to `GenerateOutput` except for its return value containing the resulting messages, warnings and errors arranged as trees of `AppOutputLines`.

Note: This method can only be used when the MapForce (running as a COM server) main window is visible, or is embedded with a graphical user interface. If the method is called while MapForce is not visible, then an error will occur.

Signature

```
GenerateOutputEx() -> AppOutputLines
```

Errors

Error code	Description
1200	The object is no longer valid.
1201	Invalid address for the return parameter was specified.
1206	Error during execution of mapping algorithm.
1210	Generating output is only supported when the graphical user interface is visible.

17.7.1.9.2.11 GenerateXQuery

Generates mapping code as XQuery. Uses the properties defined in `Application.Options` to configure code generation.

Signature

```
GenerateXQuery() -> Void
```


(C) 2015-2021 Altova GmbH

Errors

Error code	Description
1200	The object is no longer valid.
1201	Invalid address for the return parameter was specified.
1204	Error during XSLT/XSLT2/XSLT3/XQuery code generation.

17.7.1.9.2.12 GenerateXSLT

Generates mapping code as XSLT. Uses the properties defined in `Application.Options` to configure code generation.

Signature

```
GenerateXSLT() -> Void
```

Errors

Error code	Description
1200	The object is no longer valid.
1201	Invalid address for the return parameter was specified.
1204	Error during XSLT/XSLT2/XSLT3/XQuery code generation.

17.7.1.9.2.13 GenerateXSLT2

Generates mapping code as XSLT2. Uses the properties defined in `Application.Options` to configure code generation.

Signature

```
GenerateXSLT2() -> Void
```

Errors

Error code	Description
1200	The object is no longer valid.
1201	Invalid address for the return parameter was specified.
1204	Error during XSLT/XSLT2/XSLT3/XQuery code generation.

17.7.1.9.2.14 GenerateXSLT3

Generates XSLT 3.0 mapping code. Uses the properties defined in `Application.Options` to configure code generation.

Signature

```
GenerateXSLT3() -> Void
```

Errors

Error code	Description
1200	The object is no longer valid.
1201	Invalid address for the return parameter was specified.
1204	Error during XSLT/XSLT2/XSLT3/XQuery code generation.

17.7.1.9.2.15 HighlightSerializedMarker

Use this method to highlight a location in a mapping file that has been previously serialized. If the corresponding document is not already loaded, it will be loaded first. See `GenerateCodeEx` for a method to retrieve a serialized marker.

Signature

```
HighlightSerializedMarker(in i_strSerializedMarker:String) -> Void
```

Parameters

Name	Type	Description
i_strSerializedMarker	`String`	The `ErrorMarker` object to highlight. Use `ErrorMaker.Serialized` to obtain this value.

Errors

Error code	Description
1000	The object is no longer valid.
1001	Invalid address for the return parameter was specified.
1007	The string passed in `i_strSerializedMarker` is not recognized a serialized MapForce marker.

(C) 2015-2021 Altova GmbH

Error code	Description
1008	The marker points to a location that is no longer valid.

17.7.1.9.2.16 Save

Saves the document to the file defined by `Document.FullName`.

Signature

```
Save() -> Void
```

Errors

Error code	Description
1200	The object is no longer valid.
1201	Invalid address for the return parameter was specified.

17.7.1.9.2.17 SaveAs

Saves the document to the specified file name, and sets `Document.FullName` to this value if the save operation was successful.

Signature

```
SaveAs(in i_strFileName:String) -> Void
```

Parameters

Name	Type	Description
i_strFileName	`String`	Specifies the path where to save the document.

Errors

Error code	Description
1200	The object is no longer valid.
1201	Invalid address for the return parameter was specified.

17.7.1.9.3 Events

17.7.1.9.3.1 OnDocumentClosed

This event is triggered when a document is closed. The document object passed into the event handler should not be accessed. The corresponding open event is `Application.OnDocumentOpened`.

Signature

```
OnDocumentClosed(in i_ipDocument:Document) : Void
```

17.7.1.9.3.2 OnModifiedFlagChanged

This event is triggered when a document's modification status changes.

Signature

```
OnModifiedFlagChanged(in i_bIsModified:Boolean) : Void
```

17.7.1.10 Documents

Represents a collection of `Document` objects.

Properties to navigate the object model:

- `Application`
- `Parent`

Open and create mappings:

- `OpenDocument`
- `NewDocument`

Iterating through the collection:

- `Count`
- `Item`
- `ActiveDocument`

Properties

Name	Description
`ActiveDocument`	Read-only. Retrieves the active document. If no document is open, `null` is returned.

(C) 2015-2021 Altova GmbH

Name	Description
Application	Read-only. Retrieves the application's top-level object.
Count	Read-only. Retrieves the number of documents in the collection.
Item	Read-only. Retrieves the document at index `n` from the collection. Indices start with 1.
Parent	Read-only. The parent object according to the object model.

Methods

Name	Description
NewDocument	Creates a new document, adds it to the end of the collection, and makes it the active document.
OpenDocument	Opens an existing mapping document (*.mfd). Adds the newly opened document to the end of the collection and makes it the active document.

17.7.1.10.1 Properties

17.7.1.10.1.1 ActiveDocument

Retrieves the active document. If no document is open, `null` is returned.

Signature

```
ActiveDocument : Document
```

Errors

Error code	Description
1600	The object is no longer valid.
1601	Invalid address for the return parameter was specified.

17.7.1.10.1.2 Application

Retrieves the application's top-level object.

Signature

```
Application : Application
```

Errors

Error code	Description
1600	The object is no longer valid.
1601	Invalid address for the return parameter was specified.

17.7.1.10.1.3 Count

Retrieves the number of documents in the collection.

Signature

```
Count : Integer
```

Errors

Error code	Description
1600	The object is no longer valid.
1601	Invalid address for the return parameter was specified.

17.7.1.10.1.4 Item

Retrieves the document at index `n` from the collection. Indices start with 1.

Signature

```
Item(in n:Integer) : Document
```

Errors

Error code	Description
1600	The object is no longer valid.

(C) 2015-2021 Altova GmbH

Error code	Description
1601	Invalid address for the return parameter was specified.

17.7.1.10.1.5 Parent

The parent object according to the object model.

Signature

```
Parent : Application
```

Errors

Error code	Description
1600	The object is no longer valid.
1601	Invalid address for the return parameter was specified.

17.7.1.10.2 Methods

17.7.1.10.2.1 NewDocument

Creates a new document, adds it to the end of the collection, and makes it the active document.

Signature

```
NewDocument() -> Document
```

Errors

Error code	Description
1600	The object is no longer valid.
1601	Invalid address for the return parameter was specified.

17.7.1.10.2.2 OpenDocument

Opens an existing mapping document (*.mfd). Adds the newly opened document to the end of the collection and makes it the active document.

Signature

```
OpenDocument(in strPath:String) -> Document
```

Parameters

Name	Type	Description
strPath	`String`	The path of the mapping file.

Errors

Error code	Description
1600	The object is no longer valid.
1601	Invalid address for the return parameter was specified.

17.7.1.11 ErrorMarker

Represents a simple message line. Unlike `AppOutputLine` objects, error markers do not have a hierarchical structure.

Properties to navigate the object model:

- `Application`
- `Parent`

Access to message information:

- `DocumentFileName`
- `ErrorLevel`
- `Highlight`
- `Serialization`
- `Text`

Properties

Name	Description
`Application`	Read-only. Retrieves the application's top-level object.
`DocumentFileName`	Read-only. Retrieves the name of the mapping file that the error marker is associated with.
`ErrorLevel`	Read-only. Retrieves the severity of the error.

(C) 2015-2021 Altova GmbH

Name	Description
Parent	Read-only. The parent object according to the object model.
Serialization	Read-only. Serialize error marker into a string. Use this string in calls to `Application.HighlightSerializedMarker` or `Document.HighlightSerializedMarker` to highlight the marked item in the mapping. The string can be persisted and used in other instantiations of MapForce or its Control.
Text	Read-only. Retrieves the message text.

Methods

Name	Description
Highlight	Highlights the item that the error marker is associated with. If the corresponding document is not open, it will be opened.

17.7.1.11.1 Properties

17.7.1.11.1.1 Application

Retrieves the application's top-level object.

Signature

```
Application : Application
```

Errors

Error code	Description
1900	The object is no longer valid.
1901	Invalid address for the return parameter was specified.

17.7.1.11.1.2 DocumentFileName

Retrieves the name of the mapping file that the error marker is associated with.

Signature

```
DocumentFileName : String
```

Errors

Error code	Description
1900	The object is no longer valid.
1901	Invalid address for the return parameter was specified.

17.7.1.11.1.3 ErrorLevel

Retrieves the severity of the error.

Signature

```
ErrorLevel : ENUMCodeGenErrorLevel
```

Errors

Error code	Description
1900	The object is no longer valid.
1901	Invalid address for the return parameter was specified.

17.7.1.11.1.4 Parent

The parent object according to the object model.

Signature

```
Parent : ErrorMarkers
```

Errors

Error code	Description
1900	The object is no longer valid.
1901	Invalid address for the return parameter was specified.

17.7.1.11.1.5 Serialization

Serialize error marker into a string. Use this string in calls to `Application.HighlightSerializedMarker` or `Document.HighlightSerializedMarker` to highlight the marked item in the mapping. The string can be persisted and used in other instantiations of MapForce or its Control.

(C) 2015-2021 Altova GmbH

Signature

```
Serialization : String
```

Errors

Error code	Description
1900	The object is no longer valid.
1901	Invalid address for the return parameter was specified.

17.7.1.11.1.6 Text

Retrieves the message text.

Signature

```
Text : String
```

Errors

Error code	Description
1900	The object is no longer valid.
1901	Invalid address for the return parameter was specified.

17.7.1.11.2 Methods

17.7.1.11.2.1 Highlight

Highlights the item that the error marker is associated with. If the corresponding document is not open, it will be opened.

Signature

```
Highlight() -> Void
```

Errors

Error code	Description
1900	The object is no longer valid.

Error code	Description
1901	Invalid address for the return parameter was specified.
1008	The marker points to a location that is no longer valid.

17.7.1.12 ErrorMarkers

Represents a collection of `ErrorMarker` objects.

Properties to navigate the object model:

- `Application`
- `Parent`

Iterating through the collection:

- `Count`
- `Item`

Properties

Name	Description
`Application`	Read-only. Retrieves the application's top-level object.
`Count`	Read-only. Retrieves the number of error markers in the collection.
`Item`	Read-only. Retrieves the error marker at index `n` from the collection. Indices start with 1.
`Parent`	Read-only. The parent object according to the object model.

(C) 2015-2021 Altova GmbH

17.7.1.12.1 Properties

17.7.1.12.1.1 Application

Retrieves the application's top-level object.

Signature

```
Application : Application
```

Errors

Error code	Description
1800	The object is no longer valid.
1801	Invalid address for the return parameter was specified.

17.7.1.12.1.2 Count

Retrieves the number of error markers in the collection.

Signature

```
Count : Integer
```

Errors

Error code	Description
1800	The object is no longer valid.
1801	Invalid address for the return parameter was specified.

17.7.1.12.1.3 Item

Retrieves the error marker at index `n` from the collection. Indices start with 1.

Signature

```
Item(in n:Integer) : ErrorMarker
```

Errors

Error code	Description
1800	The object is no longer valid.
1801	Invalid address for the return parameter was specified.

17.7.1.12.1.4 Parent

The parent object according to the object model.

Signature

```
Parent : Application
```

Errors

Error code	Description
1800	The object is no longer valid.
1801	Invalid address for the return parameter was specified.

17.7.1.13 LibraryImport

A `LibraryImport` represents an imported library file (an entry from the **Manage Libraries** window).

Properties

Name	Description
`Application`	Read-only. Retrieves the application's top level object.
`Parent`	Read-only. Retrieves the parent object, according to the object model.
`Path`	Read-only. Gets the path of the imported library.
`SaveRelativePath`	When you save the document, this property specifies whether the library path should be saved as absolute or relative. When **true**, the path of the library will be relative to the document. When **false**, the library path will be absolute. Do not rely on this property to determine whether the path is absolute or relative, since the path may have been changed

(C) 2015-2021 Altova GmbH

Name	Description
	(either from the user interface or via API) since the document was loaded from the .mfd file. If you set this property (either via API or via user interface), the "Manage Libraries" window does immediately show the correct path's state in the user interface. Internally, however, the `Path` of the `ImportedLibrary` object will not be changed until the document is saved. Libraries imported globally cannot be saved with a relative path. Only libraries imported at document level can.

17.7.1.13.1 Properties

17.7.1.13.1.1 Application

Retrieves the application's top level object.

Signature

```
Application : Application
```

Errors

Error code	Description
2500	The object is no longer valid.
2501	Invalid address for the return parameter was specified.

17.7.1.13.1.2 Parent

Retrieves the parent object, according to the object model.

Signature

```
Parent : LibraryImports
```

Errors

Error code	Description
2500	The object is no longer valid.

Error code	Description
2501	Invalid address for the return parameter was specified.

17.7.1.13.1.3 Path

Gets the path of the imported library.

Signature

```
Path : String
```

Errors

Error code	Description
2500	The object is no longer valid.
2501	Invalid address for the return parameter was specified.

17.7.1.13.1.4 SaveRelativePath

When you save the document, this property specifies whether the library path should be saved as absolute or relative. When **true**, the path of the library will be relative to the document. When **false**, the library path will be absolute.

Do not rely on this property to determine whether the path is absolute or relative, since the path may have been changed (either from the user interface or via API) since the document was loaded from the .mfd file.

If you set this property (either via API or via user interface), the "Manage Libraries" window does immediately show the correct path's state in the user interface. Internally, however, the `Path` of the `ImportedLibrary` object will not be changed until the document is saved.

> Libraries imported globally cannot be saved with a relative path. Only libraries imported at document level can.

Signature

```
SaveRelativePath : Boolean
```

Errors

Error code	Description
2500	The object is no longer valid.

(C) 2015-2021 Altova GmbH

Error code	Description
2501	Invalid address for the return parameter was specified.
2502	Global imports cannot be saved with a relative path.

17.7.1.14 LibraryImports

Represents a collection of imported libraries (`LibraryImport` objects). Use the `Application` and `Parent` properties to navigate the object model. Use the `Count` and `Item` properties to iterate through the collection. You can get this collection as follows:

- Locally (at document level), through the `Document.LibraryImports` property
- Globally (at application level), through the `Application.LibraryImports` property.

> If you get the `LibraryImports` collection from the application object, the `Parent` property of the collection will be null.

Properties

Name	Description
`Application`	Read-only. Gets the application's top level object.
`Count`	Read-only. Gets the count of `LibraryImport` objects in this collection.
`Item`	Read-only. Retrieves a library entry at index *n* from this collection. The index is 1-based.
`Parent`	Read-only. Gets the parent document for local library imports. If you get the `LibraryImports` collection from the application object, the `Parent` property of the collection will be null.

Methods

Name	Description
`Add`	Adds a new library to this `LibraryImports` object. The new library will have the path supplied by the **i_strFileName** parameter.
`Find`	Returns a library reference given the path to the library file.
`Remove`	Removes a library reference from the **Manage Libraries** window.

17.7.1.14.1 Properties

17.7.1.14.1.1 Application

Gets the application's top level object.

Signature

```
Application : Application
```

Errors

Error code	Description
2400	The object is no longer valid.
2401	Invalid address for the return parameter was specified.

17.7.1.14.1.2 Count

Gets the count of `LibraryImport` objects in this collection.

Signature

```
Count : Integer
```

Errors

Error code	Description
2400	The object is no longer valid.
2401	Invalid address for the return parameter was specified.

17.7.1.14.1.3 Item

Retrieves a library entry at index *n* from this collection. The index is 1-based.

Signature

```
Item(in n:Integer) : LibraryImport
```


(C) 2015-2021 Altova GmbH

Errors

Error code	Description
2400	The object is no longer valid.
2401	Invalid address for the return parameter was specified.

17.7.1.14.1.4 Parent

Gets the parent document for local library imports. If you get the `LibraryImports` collection from the application object, the `Parent` property of the collection will be null.

Signature

```
Parent : Document
```

Errors

Error code	Description
2400	The object is no longer valid.
2401	Invalid address for the return parameter was specified.

17.7.1.14.2 Methods

17.7.1.14.2.1 Add

Adds a new library to this `LibraryImports` object. The new library will have the path supplied by the **i_strFileName** parameter.

Signature

```
Add(in i_strFileName:String) -> LibraryImport
```

Parameters

Name	Type	Description
i_strFileName	`String`	Specifies the path of the library file. This path can be either absolute or relative to the mapping, depending on the state in which it was passed to the

Name	Type	Description
		object. When the document is saved, the path will be made relative if the `LibraryImport.SaveRelativePath` flag is **true**; otherwise, it will be made absolute.

Errors

Error code	Description
2400	The object is no longer valid.
2401	Invalid address for the return parameter was specified.
2402	Adding library file failed.

17.7.1.14.2.2 Find

Returns a library reference given the path to the library file.

Signature

```
Find(in i_strFileName:String) -> LibraryImport
```

Parameters

Name	Type	Description
i_strFileName	`String`	The path of the library file to search for. For locally imported libraries, you can specify either the absolute or the relative path to the library file (unlike the `Remove` method, which requires the exact path). For globally imported libraries, the path must always be absolute (since globally imported libraries cannot have a relative path).

Errors

Error code	Description
2400	The object is no longer valid.

(C) 2015-2021 Altova GmbH

Error code	Description
2401	Invalid address for the return parameter was specified.

17.7.1.14.2.3 Remove

Removes a library reference from the **Manage Libraries** window.

Signature

```
Remove(in i_strFileName:String) -> Void
```

Parameters

Name	Type	Description
i_strFileName	`String`	The path of the library file to remove. Note that the path must reflect exactly the current (most recent) state of the `LibraryImport` object. Remember that the path may be either relative or absolute, and it may have changed if you saved the document, depending on the `LibraryImport.SaveRelativePath` flag. Therefore, if the `LibraryImport` object currently contains a relative path, then you should supply a relative path as value of this parameter. Otherwise, the library is not found and the `Remove` method fails. The above applies only for locally imported libraries. For globally imported libraries, the path must always be absolute (since globally imported libraries cannot have a relative path)

Errors

Error code	Description
2400	The object is no longer valid.

17.7.1.15 MapForceView

Represents the current view in the MapForce Mapping tab for a document. A document has exactly one `MapForceView` which displays the currently active mapping.

Properties to navigate the object model:

- `Application`
- `Parent`

View activation and view properties:

- `Active`
- `ShowItemTypes`
- `ShowLibraryInFunctionHeader`
- `HighlightMyConnections`
- `HighlightMyConnectionsRecursivly`

Mapping related properties:

- `ActiveMapping`
- `ActiveMappingName`

Adding items:

- `InsertWSDLCall`
- `InsertXMLFile`
- `InsertXMLSchema`
- `InsertXMLSchemaWithSample`

Properties

Name	Description
`Active`	Use this property to query if the mapping view is the active view, or set this view to be the active one.
`ActiveMapping`	Gets or sets the currently active mapping in the document this `MapForceView` belongs to.
`ActiveMappingName`	Gets or sets the currently active mapping by name in the document this MapForceView belongs to.
`Application`	Read-only. Retrieves the application's top-level object.
`HighlightMyConnections`	This property defines whether connections from the selected item only should be highlighed.
`HighlightMyConnectionsRecursively`	This property defines if only the connections coming directly or indirectly from the selected item should be highlighted.
`Parent`	Read-only. The parent object according to the object model.

(C) 2015-2021 Altova GmbH

Name	Description
ShowItemTypes	This property defines if types of items should be shown in the mapping diagram.
ShowLibraryInFunctionHeader	This property defines whether the name of the function library should be part of function names.

Methods

Name	Description
InsertWSDLCall	Adds a new WSDL call component to the mapping.
InsertXMLFile	`MapForceView.InsertXMLFile` is obsolete. Use `Mapping.InsertXMLFile` instead.
InsertXMLSchema	`MapForceView.InsertXMLSchema` is obsolete. Use `Mapping.InsertXMLSchema` instead.
InsertXMLSchemaWithSample	`MapForceView.InsertXMLSchemaWithSample` is obsolete. Use `Mapping.InsertXMLFile` instead. Notice, `Mapping.InsertXMLFile` does not require a parameter for passing the root element. The root element is automatically set as the XML file's root element name.

17.7.1.15.1 Properties

17.7.1.15.1.1 Active

Use this property to query if the mapping view is the active view, or set this view to be the active one.

Signature

```
Active : Boolean
```

Errors

Error code	Description
1300	The object is no longer valid.
1301	Invalid address for the return parameter was specified.

17.7.1.15.1.2 ActiveMapping

Gets or sets the currently active mapping in the document this `MapForceView` belongs to.

Signature

```
ActiveMapping : Mapping
```

Errors

Error code	Description
1300	The object is no longer valid.
1301	Invalid address for the return parameter was specified.

17.7.1.15.1.3 ActiveMappingName

Gets or sets the currently active mapping by name in the document this MapForceView belongs to.

Signature

```
ActiveMappingName : String
```

Errors

Error code	Description
1300	The object is no longer valid.
1301	Invalid address for the return parameter was specified.

17.7.1.15.1.4 Application

Retrieves the application's top-level object.

Signature

```
Application : Application
```

Errors

Error code	Description
1300	The object is no longer valid.

(C) 2015-2021 Altova GmbH

Error code	Description
1301	Invalid address for the return parameter was specified.

17.7.1.15.1.5 HighlightMyConnections

This property defines whether connections from the selected item only should be highlighed.

Signature

```
HighlightMyConnections : Boolean
```

Errors

Error code	Description
1300	The object is no longer valid.
1301	Invalid address for the return parameter was specified.

17.7.1.15.1.6 HighlightMyConnectionsRecursively

This property defines if only the connections coming directly or indirectly from the selected item should be highlighted.

Signature

```
HighlightMyConnectionsRecursively : Boolean
```

Errors

Error code	Description
1300	The object is no longer valid.
1301	Invalid address for the return parameter was specified.

17.7.1.15.1.7 Parent

The parent object according to the object model.

Signature

```
Parent : Document
```

Errors

Error code	Description
1300	The object is no longer valid.
1301	Invalid address for the return parameter was specified.

17.7.1.15.1.8 ShowItemTypes

This property defines if types of items should be shown in the mapping diagram.

Signature

```
ShowItemTypes : Boolean
```

Errors

Error code	Description
1300	The object is no longer valid.
1301	Invalid address for the return parameter was specified.

17.7.1.15.1.9 ShowLibraryInFunctionHeader

This property defines whether the name of the function library should be part of function names.

Signature

```
ShowLibraryInFunctionHeader : Boolean
```

Errors

Error code	Description
1300	The object is no longer valid.

(C) 2015-2021 Altova GmbH

Error code	Description
1301	Invalid address for the return parameter was specified.

17.7.1.15.2 Methods

17.7.1.15.2.1 InsertWSDLCall

Adds a new WSDL call component to the mapping.

Signature

```
InsertWSDLCall(in i_strWSDLFileName:String) -> Void
```

Parameters

Name	Type	Description
i_strWSDLFileName	`String`	Specifies the path of the WSDL file to add to the mapping.

Errors

Error code	Description
1300	The object is no longer valid.
1301	Invalid address for the return parameter was specified.

17.7.1.15.2.2 InsertXMLFile (obsolete)

`MapForceView.InsertXMLFile` is obsolete. Use `Mapping.InsertXMLFile` instead.

Signature

```
InsertXMLFile(in i_strFileName:String, in i_strXMLRootName:String) -> Void
```

Parameters

Name	Type	Description
i_strFileName	`String`	
i_strXMLRootName	`String`	

17.7.1.15.2.3 InsertXMLSchema (obsolete)

`MapForceView.InsertXMLSchema` is obsolete. Use `Mapping.InsertXMLSchema` instead.

Signature

```
InsertXMLSchema(in i_strSchemaFileName:String, in i_strXMLRootName:String) -> Void
```

Parameters

Name	Type	Description
i_strSchemaFileName	`String`	
i_strXMLRootName	`String`	

17.7.1.15.2.4 InsertXMLSchemaWithSample (obsolete)

`MapForceView.InsertXMLSchemaWithSample` is obsolete. Use `Mapping.InsertXMLFile` instead. Notice, `Mapping.InsertXMLFile` does not require a parameter for passing the root element. The root element is automatically set as the XML file's root element name.

Signature

```
InsertXMLSchemaWithSample(in i_strSchemaFileName:String, in i_strXMLExampleFile:String,
in i_strXMLRootName:String) -> Void
```

Parameters

Name	Type	Description
i_strSchemaFileName	`String`	
i_strXMLExampleFile	`String`	
i_strXMLRootName	`String`	

17.7.1.16 Mapping

A `Mapping` object represents a mapping in a document, either the main mapping, or a local user-defined-function mapping.

Properties to navigate the object model:

- `Application`
- `Parent`

(C) 2015-2021 Altova GmbH

Mapping properties:

- IsMainMapping
- Name

Components in the mapping:

- Components

Adding items:

- CreateConnection
- InsertFunctionCall
- InsertXMLFile
- InsertXMLSchema
- InsertXMLSchemaInputParameter
- InsertXMLSchemaOutputParameter

Properties

Name	Description
Application	Read-only. Retrieves the application's top-level object.
Components	Read-only. Returns a collection of all components in the current mapping.
IsMainMapping	Read-only. Indicates if the current mapping is the main mapping of the document the mapping is in. **True** means it is the main mapping. **False** means it is a user-defined function (UDF).
Name	Read-only. The name of the mapping or user defined-function (UDF).
Parent	Read-only. The parent object according to the object model.

Methods

Name	Description
CreateConnection	Creates a connection between the two supplied datapoints (DatapointFrom & DatapointTo). It will fail to do so if the DatapointFrom is not an output-side datapoint, the DatapointTo is not an input-side datapoint, or a connection between these two datapoints already exists.
InsertFunctionCall	Inserts a function call component into the current mapping.

Name	Description
	The specified library and function names indicate the function or user-defined function to be called.
InsertXMLFile	Adds a new XML schema component to the mapping. The component's internal structure is determined by the schema referenced in the specified XML file (`i_strFileName`) or, if the XML file does not reference a schema file, by the separately specified schema file (`i_strSchemaFileName`). If the XML file has a schema file reference, then the parameter `i_strSchemaFileName` is ignored. The root element of the XML file will be used in the component. The specified XML file is used as the input sample to evaluate the mapping.
InsertXMLSchema	Adds a new XML schema component to the mapping. The component's internal structure is determined by the schema file specified in the first parameter. The second parameter defines the root element of this schema if there is more than one candidate. If the passed root element is an empty string and more candidates are available, a **Select Root Element** dialog box will pop up if MapForce is visible. If MapForce is invisible, no dialog box will pop up and only an error is returned. No XML input sample is assigned to this component.
InsertXMLSchemaInputParameter	Inserts an XML schema input parameter component into the current mapping. The current mapping has to be a user-defined function. Trying to insert it (the schema input parameter) into the main mapping will fail.
InsertXMLSchemaOutputParameter	Inserts an XML schema output parameter component into the current mapping. The current mapping has to be a user-defined function. Trying to insert it (the schema output paramter) into the main mapping will fail.

(C) 2015-2021 Altova GmbH

17.7.1.16.1 Properties

17.7.1.16.1.1 Application

Retrieves the application's top-level object.

Signature

```
Application : Application
```

Errors

Error code	Description
1200	The object is no longer valid.
1201	Invalid address for the return parameter was specified.

17.7.1.16.1.2 Components

Returns a collection of all components in the current mapping.

Signature

```
Components : Components
```

Errors

Error code	Description
1200	The object is no longer valid.
1201	Invalid address for the return parameter was specified.

17.7.1.16.1.3 IsMainMapping

Indicates if the current mapping is the main mapping of the document the mapping is in.

True means it is the main mapping.
False means it is a user-defined function (UDF).

Signature

```
IsMainMapping : Boolean
```

Errors

Error code	Description
1200	The object is no longer valid.
1201	Invalid address for the return parameter was specified.

17.7.1.16.1.4 Name

The name of the mapping or user defined-function (UDF).

Signature

```
Name : String
```

Errors

Error code	Description
1200	The object is no longer valid.
1201	Invalid address for the return parameter was specified.

17.7.1.16.1.5 Parent

The parent object according to the object model.

Signature

```
Parent : Document
```

Errors

Error code	Description
1200	The object is no longer valid.
1201	Invalid address for the return parameter was specified.

(C) 2015-2021 Altova GmbH

17.7.1.16.2 Methods

17.7.1.16.2.1 CreateConnection

Creates a connection between the two supplied datapoints (`DatapointFrom` & `DatapointTo`).

It will fail to do so if the `DatapointFrom` is not an output-side datapoint, the `DatapointTo` is not an input-side datapoint, or a connection between these two datapoints already exists.

Signature

```
CreateConnection(in DatapointFrom:Datapoint, in DatapointTo:Datapoint) -> Connection
```

Parameters

Name	Type	Description
DatapointFrom	`Datapoint`	The datapoint from which the connection is to be created.
DatapointTo	`Datapoint`	The destination datapoint.

Errors

Error code	Description
1200	The object is no longer valid.
1201	Invalid address for the return parameter was specified.
1240	Changing the document not allowed. It is read-only.
1241	Failed creating the connection.

17.7.1.16.2.2 InsertFunctionCall

Inserts a function call component into the current mapping.

The specified library and function names indicate the function or user-defined function to be called.

Signature

```
InsertFunctionCall(in strFunctionName:String, in strLibraryName:String) -> Component
```

Parameters

Name	Type	Description
strFunctionName	`String`	The name of the function to be inserted.
strLibraryName	`String`	The library name of the function to be inserted

Errors

Error code	Description
1200	The object is no longer valid.
1201	Invalid address for the return parameter was specified.
1240	Changing the document not allowed. It is read-only.
1242	Failed creating function call component.

17.7.1.16.2.3 InsertXMLFile

Adds a new XML schema component to the mapping.

The component's internal structure is determined by the schema referenced in the specified XML file (`i_strFileName`) or, if the XML file does not reference a schema file, by the separately specified schema file (`i_strSchemaFileName`).

If the XML file has a schema file reference, then the parameter `i_strSchemaFileName` is ignored.

The root element of the XML file will be used in the component.

The specified XML file is used as the input sample to evaluate the mapping.

Signature

```
InsertXMLFile(in i_strFileName:String, in i_strSchemaFileName:String) -> Component
```

Parameters

Name	Type	Description
i_strFileName	`String`	The path of the instance XML file to add.
i_strSchemaFileName	`String`	The path of the XML Schema Definition file to add.

(C) 2015-2021 Altova GmbH

Errors

Error code	Description
1200	The object is no longer valid.
1201	Invalid address for the return parameter was specified.
1240	Changing the document not allowed. It is read-only.
1244	Failed creating component.

17.7.1.16.2.4 InsertXMLSchema

Adds a new XML schema component to the mapping.

The component's internal structure is determined by the schema file specified in the first parameter.

The second parameter defines the root element of this schema if there is more than one candidate.

If the passed root element is an empty string and more candidates are available, a **Select Root Element** dialog box will pop up if MapForce is visible. If MapForce is invisible, no dialog box will pop up and only an error is returned.

No XML input sample is assigned to this component.

Signature

```
InsertXMLSchema(in i_strSchemaFileName:String, in i_strXMLRootName:String) -> Component
```

Parameters

Name	Type	Description
i_strSchemaFileName	`String`	The path of the XML Schema Definition file to add.
i_strXMLRootName	`String`	The root element of the schema (applicable when the schema has more than one root element).

Errors

Error code	Description
1200	The object is no longer valid.
1201	Invalid address for the return parameter was specified.
1240	Changing the document not allowed. It is read-only.

Error code	Description
1244	Failed creating component.

17.7.1.16.2.5 InsertXMLSchemaInputParameter

Inserts an XML schema input parameter component into the current mapping.

The current mapping has to be a user-defined function. Trying to insert it (the schema input parameter) into the main mapping will fail.

Signature

```
InsertXMLSchemaInputParameter(in strParamName:String, in strSchemaFileName:String, in
strXMLRootElementName:String) -> Component
```

Parameters

Name	Type	Description
strParamName	`String`	The name of the input parameter component to create.
strSchemaFileName	`String`	The path of the XML Schema Definition file to add.
strXMLRootElementName	`String`	The root element of the schema (applicable when the schema has more than one root element). If the passed root element is an empty string and more candidates are available, a **Select Root Element** dialog will pop up if MapForce is visible. If MapForce is invisible, no dialog box will pop up and only an error is returned.

Errors

Error code	Description
1200	The object is no longer valid.
1201	Invalid address for the return parameter was specified.
1240	Changing the document not allowed. It is read-only.
1243	Failed creating parameter component.
1245	This operation is not supported for the main mapping.

(C) 2015-2021 Altova GmbH

17.7.1.16.2.6 InsertXMLSchemaOutputParameter

Inserts an XML schema output parameter component into the current mapping.

The current mapping has to be a user-defined function. Trying to insert it (the schema output paramter) into the main mapping will fail.

Signature

```
InsertXMLSchemaOutputParameter(in strParamName:String, in strSchemaFileName:String, in
strXMLRootElementName:String) -> Component
```

Parameters

Name	Type	Description
strParamName	`String`	The name of the output parameter component to create.
strSchemaFileName	`String`	The path of the XML Schema Definition file to add.
strXMLRootElementName	`String`	The root element of the schema (applicable when the schema has more than one root element). If the passed root element is an empty string and more candidates are available, a **Select Root Element** dialog will pop up if MapForce is visible. If MapForce is invisible, no dialog box will pop up and only an error is returned.

Errors

Error code	Description
1200	The object is no longer valid.
1201	Invalid address for the return parameter was specified.
1240	Changing the document not allowed. It is read-only.
1243	Failed creating parameter component.
1245	This operation is not supported for the main mapping.

17.7.1.17 Mappings

Represents a collection of `Mapping` objects.

Properties to navigate the object model:

- `Application`
- `Parent`

Iterating through the collection:

- `Count`
- `Item`

Properties

Name	Description
`Application`	Read-only. Retrieves the application's top-level object.
`Count`	Read-only. Retrieves the number of mappings in the collection.
`Item`	Read-only. Retrieves the mapping at index `n` from the collection. Indices start with 1.
`Parent`	Read-only. The parent object according to the object model.

17.7.1.17.1 Properties

17.7.1.17.1.1 Application

Retrieves the application's top-level object.

Signature

```
Application : Application
```

Errors

Error code	Description
1200	The object is no longer valid.
1201	Invalid address for the return parameter was specified.

(C) 2015-2021 Altova GmbH

17.7.1.17.1.2 Count

Retrieves the number of mappings in the collection.

Signature

```
Count : Integer
```

Errors

Error code	Description
1200	The object is no longer valid.
1201	Invalid address for the return parameter was specified.

17.7.1.17.1.3 Item

Retrieves the mapping at index `n` from the collection. Indices start with 1.

Signature

```
Item(in n:Integer) : Mapping
```

Errors

Error code	Description
1200	The object is no longer valid.
1201	Invalid address for the return parameter was specified.

17.7.1.17.1.4 Parent

The parent object according to the object model.

Signature

```
Parent : Document
```

Errors

Error code	Description
1200	The object is no longer valid.

Error code	Description
1201	Invalid address for the return parameter was specified.

17.7.1.18 Options

This object gives access to all MapForce options available in the **Tools | Options** dialog.

Properties to navigate the object model:

- `Application`
- `Parent`

General options:

- `ShowLogoOnPrint`
- `ShowLogoOnStartup`
- `UseGradientBackground`

Options for code generation:

- `DefaultOutputEncoding`
- `DefaultOutputByteOrder`
- `DefaultOutputByteOrderMark`
- `XSLTDefaultOutputDirectory`
- `CodeDefaultOutputDirectory`
- `CPPSettings_DOMType`
- `CPPSettings_GenerateVC6ProjectFile`
- `CppSettings_GenerateVSProjectFile`
- `CPPSettings_LibraryType`
- `CPPSettings_UseMFC`
- `CSharpSettings_ProjectType`

Properties

Name	Description
`Application`	Read-only. Retrieves the application's top-level object.
`CodeDefaultOutputDirectory`	Specifies the target directory where files generated by `Document.GenerateCppCode`, `Document.GenerateJavaCode` and `Document.GenerateCHashCode` are placed.
`CPPSettings_DOMType`	Specifies the DOM type used by `Document.GenerateCppCode`.
`CPPSettings_GenerateVC6ProjectFile`	Specifies if VisualC++ 6.0 project files should be generated by `Document.GenerateCppCode`.
`CppSettings_GenerateVSProjectFile`	Specifies the version of Visual Studio in which project files should be generated by `Document.GenerateCppCode`.

(C) 2015-2021 Altova GmbH

Name	Description
CPPSettings_LibraryType	Specifies the library type used by Document.GenerateCppCode.
CPPSettings_UseMFC	Specifies if MFC support should be used by C++ code generated by Document.GenerateCppCode.
CSharpSettings_ProjectType	Specifies the type of C# project used by Document.GenerateCHashCode.
DefaultOutputByteOrder	Byte order for the file encoding used for output files.
DefaultOutputByteOrderMark	Indicates if a byte order mark (BOM), is to be included in the file encoding of output files.
DefaultOutputEncoding	File encoding used for output files.
GenerateWrapperClasses	Indicates if wrapper classes are also to be generated when generating code.
JavaSettings_ApacheAxisVersion	This property is obsolete.
Parent	Read-only. The parent object according to the object model.
ShowLogoOnPrint	Show or hide the MapForce logo on printed outputs.
ShowLogoOnStartup	Show or hide the MapForce logo on application startup.
UseGradientBackground	Set or retrieve the background color mode for a mapping window.
XSLTDefaultOutputDirectory	Specifies the target directory where files generated by Document.GenerateXSLT are placed.

17.7.1.18.1 Properties

17.7.1.18.1.1 Application

Retrieves the application's top-level object.

Signature

```
Application : Application
```

Errors

Error code	Description
1400	The object is no longer valid.

Error code	Description
1401	Invalid address for the return parameter was specified.

17.7.1.18.1.2 CodeDefaultOutputDirectory

Specifies the target directory where files generated by `Document.GenerateCppCode`, `Document.GenerateJavaCode` and `Document.GenerateCHashCode` are placed.

Signature

```
CodeDefaultOutputDirectory : String
```

Errors

Error code	Description
1400	The object is no longer valid.
1401	Invalid address for the return parameter was specified.

17.7.1.18.1.3 CPPSettings_DOMType

Specifies the DOM type used by `Document.GenerateCppCode`.

Signature

```
CPPSettings_DOMType : ENUMDOMType
```

Errors

Error code	Description
1400	The object is no longer valid.
1401	Invalid address for the return parameter was specified.
1402	The parameter value is out of range.
1403	The parameter value is not available anymore.

(C) 2015-2021 Altova GmbH

17.7.1.18.1.4 CPPSettings_GenerateVC6ProjectFile (obsolete)

Specifies if VisualC++ 6.0 project files should be generated by `Document.GenerateCppCode`.

Signature

```
CPPSettings_GenerateVC6ProjectFile : Boolean
```

Errors

Error code	Description
1400	The object is no longer valid.
1401	Invalid address for the return parameter was specified.
1402	The parameter value is out of range.
1403	The parameter value is not available anymore.

17.7.1.18.1.5 CppSettings_GenerateVSProjectFile

Specifies the version of Visual Studio in which project files should be generated by `Document.GenerateCppCode`.

Signature

```
CppSettings_GenerateVSProjectFile : ENUMProjectType
```

Errors

Error code	Description
1400	The object is no longer valid.
1401	Invalid address for the return parameter was specified.
1402	The parameter value is out of range.
1403	The parameter value is not available anymore.

17.7.1.18.1.6 CPPSettings_LibraryType

Specifies the library type used by `Document.GenerateCppCode`.

Signature

```
CPPSettings_LibraryType : ENUMLibType
```

Errors

Error code	Description
1400	The object is no longer valid.
1401	Invalid address for the return parameter was specified.

17.7.1.18.1.7 CPPSettings_UseMFC

Specifies if MFC support should be used by C++ code generated by `Document.GenerateCppCode`.

Signature

```
CPPSettings_UseMFC : Boolean
```

Errors

Error code	Description
1400	The object is no longer valid.
1401	Invalid address for the return parameter was specified.

17.7.1.18.1.8 CSharpSettings_ProjectType

Specifies the type of C# project used by `Document.GenerateCHashCode`.

Signature

```
CSharpSettings_ProjectType : ENUMProjectType
```

Errors

Error code	Description
1400	The object is no longer valid.

(C) 2015-2021 Altova GmbH

Error code	Description
1401	Invalid address for the return parameter was specified.
1402	The parameter value is out of range.
1403	The parameter value is not available anymore.

17.7.1.18.1.9 DefaultOutputByteOrder

Byte order for the file encoding used for output files.

Signature

```
DefaultOutputByteOrder : String
```

Errors

Error code	Description
1400	The object is no longer valid.
1401	Invalid address for the return parameter was specified.

17.7.1.18.1.10 DefaultOutputByteOrderMark

Indicates if a byte order mark (BOM), is to be included in the file encoding of output files.

Signature

```
DefaultOutputByteOrderMark : Boolean
```

Errors

Error code	Description
1400	The object is no longer valid.
1401	Invalid address for the return parameter was specified.

17.7.1.18.1.11 DefaultOutputEncoding

File encoding used for output files.

Signature

```
DefaultOutputEncoding : String
```

Errors

Error code	Description
1400	The object is no longer valid.
1401	Invalid address for the return parameter was specified.

17.7.1.18.1.12 GenerateWrapperClasses

Indicates if wrapper classes are also to be generated when generating code.

Signature

```
GenerateWrapperClasses : Boolean
```

Errors

Error code	Description
1400	The object is no longer valid.
1401	Invalid address for the return parameter was specified.

17.7.1.18.1.13 JavaSettings_ApacheAxisVersion (obsolete)

This property is obsolete.

Signature

```
JavaSettings_ApacheAxisVersion : ENUMApacheAxisVersion
```


(C) 2015-2021 Altova GmbH

17.7.1.18.1.14 Parent

The parent object according to the object model.

Signature

```
Parent : Application
```

Errors

Error code	Description
1400	The object is no longer valid.
1401	Invalid address for the return parameter was specified.

17.7.1.18.1.15 ShowLogoOnPrint

Show or hide the MapForce logo on printed outputs.

Signature

```
ShowLogoOnPrint : Boolean
```

Errors

Error code	Description
1400	The object is no longer valid.
1401	Invalid address for the return parameter was specified.

17.7.1.18.1.16 ShowLogoOnStartup

Show or hide the MapForce logo on application startup.

Signature

```
ShowLogoOnStartup : Boolean
```

Errors

Error code	Description
1400	The object is no longer valid.

Error code	Description
1401	Invalid address for the return parameter was specified.

17.7.1.18.1.17 UseGradientBackground

Set or retrieve the background color mode for a mapping window.

Signature

```
UseGradientBackground : Boolean
```

Errors

Error code	Description
1400	The object is no longer valid.
1401	Invalid address for the return parameter was specified.

17.7.1.18.1.18 XSLTDefaultOutputDirectory

Specifies the target directory where files generated by `Document.GenerateXSLT` are placed.

Signature

```
XSLTDefaultOutputDirectory : String
```

Errors

Error code	Description
1400	The object is no longer valid.
1401	Invalid address for the return parameter was specified.

17.7.1.19 Project

A `Project` object represents a project and its tree of project items in MapForce.

Properties to navigate the object model:

- `Application`
- `Parent`

File handling:

(C) 2015-2021 Altova GmbH

- FullName
- Name
- Path
- Saved
- Save
- Close

Project tree navigation:

- Count
- Item
- _NewEnum

Project tree manipulation:

- AddActiveFile
- AddFile
- InsertWebService (Enterprise edition only)
- CreateFolder

Code generation:

- Output_Folder
- Output_Language
- Output_TextEncoding
- Java_BasePackageName
- GenerateCode
- GenerateCodeEx
- GenerateCodeIn
- GenerateCodeInEx

For examples of how to use the properties and methods listed above, see Example: Project Tasks. Note that, in order to perform operations that involve Web services, MapForce Enterprise edition is required.

Properties

Name	Description
_NewEnum	Read-only. This property supports language-specific standard enumeration.
Application	Read-only. Retrieves the top-level application object.
Count	Read-only. Retrieves number of children of the project's root item. For examples, see Item or _NewEnum
FullName	Path and name of the project file.
Item	Read-only.

Name	Description
	Returns the child at *n* position of the project's root. The index is 1-based (the first index is 1). The largest valid index is `Count`. For an alternative, see `_NewEnum`.
`Java_BasePackageName`	Sets or gets the base package name of the Java packages that will be generated. This property is used only when generating Java code.
`Name`	Read-only. Name of the project file without file path.
`Output_Folder`	Sets or gets the default output folder used with `GenerateCode` and `GenerateCodeIn`. Project items can overwrite this value in their `CodeGenSettings_OutputFolder` property, when `CodeGenSettings_UseDefault` is set to false.
`Output_Language`	Sets or gets the default language for code generation when using `GenerateCode`. Project items can overwrite this value in their `CodeGenSettings_OutputLanguage` property, when `CodeGenSettings_UseDefault` is set to false.
`Output_TextEncoding`	Sets or gets the text encoding used when generating XML-based code.
`Parent`	Read-only. The parent object according to the object model.
`Path`	Read-only. Path of the project file without name.
`Saved`	Read-only. **True** if the project was not modified since the last `Save` operation, **false** otherwise.

Methods

Name	Description
`AddActiveFile`	Adds the currently open document to the mapping folder of the project's root.
`AddFile`	Adds the specified document to the mapping folder of the project's root.
`Close`	Closes the project without saving.
`CreateFolder`	Creates a new folder as a child of the project's root item.
`GenerateCode`	Generates code for all project items of the project. The code language and output location is determined by properties of the project and project items.

(C) 2015-2021 Altova GmbH

Name	Description
GenerateCodeEx	Generates code for all project items of the project. The code language and output location are determined by properties of the project and project items. An object that can be used to iterate through all messages issued by the code generation process is returned. These messages are the same as those shown in the Messages window of MapForce.
GenerateCodeIn	Generates code for all project items of the project in the specified language. The output location is determined by properties of the project and project items.
GenerateCodeInEx	Generates code for all project items of the project in the specified language. The output location is determined by properties of the project and project items. Returns an object that can be used to iterate through all messages issued by the code generation process. These messages are the same as those shown in the **Messages** window of MapForce.
InsertWebService	Inserts a new Web service project into the project's Web service folder. If **i_bGenerateMappings** is true, initial mapping documents for all ports get generated automatically.
Save	Saves the project to the file defined by `FullName`.

Events

Name	Description
OnProjectClosed	This event is triggered when the project is closed. The project object passed into the event handler should not be accessed. The corresponding open event is `Application.OnProjectOpened`.

17.7.1.19.1 Properties

17.7.1.19.1.1 _NewEnum

This property supports language-specific standard enumeration.

Signature

```
_NewEnum : IUnknown
```

Errors

Error code	Description
1500	The object is no longer valid.

Examples

```
// ------------------------------------------------------------
// JScript sample - enumeration of a project's project items.
function AllChildrenOfProjectRoot()
{
    objProject = objMapForce.ActiveProject;
    if ( objProject != null )
    {

for ( objProjectIter = new Enumerator(objProject); ! objProjectIter.atEnd(); objProject
Iter.moveNext() )
        {
            objProjectItem = objProjectIter.item();

            // do something with project item here
        }
    }
}
```

```
// ------------------------------------------------------------
// JScript sample - iterate all project items, depth first.
function IterateProjectItemsRec(objProjectItemIter)
{
    while ( ! objProjectItemIter.atEnd() )
    {
        objProjectItem = objProjectItemIter.item();
        // do something with project item here

        IterateProjectItemsRec( new Enumerator(objProjectItem) );

        objProjectItemIter.moveNext();
    }
}
function IterateAllProjectItems()
{
```


(C) 2015-2021 Altova GmbH

```
    objProject = objMapForce.ActiveProject;
    if ( objProject != null )
    {
        IterateProjectItemsRec( new Enumerator(objProject) );
    }
}
```

17.7.1.19.1.2 Application

Retrieves the top-level application object.

Signature

```
Application : Application
```

Errors

Error code	Description
1500	The object is no longer valid.
1501	Invalid address for the return parameter was specified.

17.7.1.19.1.3 Count

Retrieves number of children of the project's root item. For examples, see `Item` or `_NewEnum`

Signature

```
Count : Integer
```

Errors

Error code	Description
1500	The object is no longer valid.

17.7.1.19.1.4 FullName

Path and name of the project file.

Signature

```
FullName : String
```

Errors

Error code	Description
1500	The object is no longer valid.
1501	Invalid address for the return parameter was specified.

17.7.1.19.1.5 Item

Returns the child at *n* position of the project's root. The index is 1-based (the first index is 1). The largest valid index is `Count`. For an alternative, see `_NewEnum`.

Signature

```
Item(in n:Integer) : ProjectItem
```

Errors

Error code	Description
1500	The object is no longer valid.

Examples

```
// ------------------------------------------------------------
// JScript code snippet - enumerate children using Count and Item.
for( nItemIndex = 1; nItemIndex <= objProject.Count; nItemIndex++ )
{
    objProjectItem = objProject.Item(nItemIndex);
    // do something with project item here
}
```

17.7.1.19.1.6 Java_BasePackageName

Sets or gets the base package name of the Java packages that will be generated. This property is used only when generating Java code.

(C) 2015-2021 Altova GmbH

Signature

```
Java_BasePackageName : String
```

Errors

Error code	Description
1500	The object is no longer valid.
1501	Invalid package name specified. Invalid address for the return parameter was specified.

17.7.1.19.1.7 Name

Name of the project file without file path.

Signature

```
Name : String
```

Errors

Error code	Description
1500	The object is no longer valid.
1501	Invalid address for the return parameter was specified.

17.7.1.19.1.8 Output_Folder

Sets or gets the default output folder used with `GenerateCode` and `GenerateCodeIn`. Project items can overwrite this value in their `CodeGenSettings_OutputFolder` property, when `CodeGenSettings_UseDefault` is set to false.

Signature

```
Output_Folder : String
```

Errors

Error code	Description
1500	The object is no longer valid.
1501	Invalid folder name specified. Invalid address for the return parameter was specified.

17.7.1.19.1.9 Output_Language

Sets or gets the default language for code generation when using `GenerateCode`. Project items can overwrite this value in their `CodeGenSettings_OutputLanguage` property, when `CodeGenSettings_UseDefault` is set to false.

Signature

```
Output_Language : ENUMProgrammingLanguage
```

Errors

Error code	Description
1500	The object is no longer valid.
1501	Invalid language specified. Invalid address for the return parameter was specified.

17.7.1.19.1.10 Output_TextEncoding

Sets or gets the text encoding used when generating XML-based code.

Signature

```
Output_TextEncoding : String
```

Errors

Error code	Description
1500	The object is no longer valid.
1501	Invalid text encoding specified. Invalid address for the return parameter was specified.

17.7.1.19.1.11 Parent

The parent object according to the object model.

Signature

```
Parent : Application
```


(C) 2015-2021 Altova GmbH

Errors

Error code	Description
1500	The object is no longer valid.
1501	Invalid address for the return parameter was specified.

17.7.1.19.1.12 Path

Path of the project file without name.

Signature

```
Path : String
```

Errors

Error code	Description
1500	The object is no longer valid.
1501	Invalid address for the return parameter was specified.

17.7.1.19.1.13 Saved

True if the project was not modified since the last `Save` operation, **false** otherwise.

Signature

```
Saved : Boolean
```

Errors

Error code	Description
1500	The object is no longer valid.
1501	Invalid address for the return parameter was specified.

17.7.1.19.2 Methods

17.7.1.19.2.1 AddActiveFile

Adds the currently open document to the mapping folder of the project's root.

Signature

```
AddActiveFile() -> ProjectItem
```

Errors

Error code	Description
1500	The object is no longer valid.
1501	Invalid address for the return parameter was specified.
1503	No active document is available.
1504	Active documents needs to be given a path name before it can be added to the project.
1705	Mapping could not be assigned to project. Maybe it is already contained in the target folder.

17.7.1.19.2.2 AddFile

Adds the specified document to the mapping folder of the project's root.

Signature

```
AddFile(in i_strFileName:String) -> ProjectItem
```

Parameters

Name	Type	Description
i_strFileName	`String`	Specifies the path of the document to add.

Errors

Error code	Description
1500	The object is no longer valid.

(C) 2015-2021 Altova GmbH

Error code	Description
1501	Invalid address for the return parameter was specified.
1705	Mapping could not be assigned to project. The file does not exist or is not a MapForce mapping. Maybe the file is already assigned to the target folder.

17.7.1.19.2.3 Close

Closes the project without saving.

Signature

```
Close() -> Void
```

Errors

Error code	Description
1500	The object is no longer valid.

17.7.1.19.2.4 CreateFolder

Creates a new folder as a child of the project's root item.

Signature

```
CreateFolder(in i_strFolderName:String) -> ProjectItem
```

Parameters

Name	Type	Description
i_strFolderName	`String`	The name of the folder to create.

Errors

Error code	Description
1500	The object is no longer valid.
1501	Invalid folder name or invalid address for the return parameter was specified.

17.7.1.19.2.5 GenerateCode

Generates code for all project items of the project. The code language and output location is determined by properties of the project and project items.

Signature

```
GenerateCode() -> Void
```

Errors

Error code	Description
1500	The object is no longer valid.
1706	Error during code generation.

17.7.1.19.2.6 GenerateCodeEx

Generates code for all project items of the project. The code language and output location are determined by properties of the project and project items. An object that can be used to iterate through all messages issued by the code generation process is returned. These messages are the same as those shown in the Messages window of MapForce.

Signature

```
GenerateCodeEx() -> ErrorMarkers
```

Errors

Error code	Description
1500	The object is no longer valid.
1501	Invalid address for the return parameter was specified.
1706	Error during code generation.

17.7.1.19.2.7 GenerateCodeIn

Generates code for all project items of the project in the specified language. The output location is determined by properties of the project and project items.

Signature

```
GenerateCodeIn(in i_nLanguage:ENUMProgrammingLanguage) -> Void
```


(C) 2015-2021 Altova GmbH

Parameters

Name	Type	Description
i_nLanguage	ENUMProgrammingLanguage	Specifies the programming language in which code should be generated.

Errors

Error code	Description
1500	The object is no longer valid.
1706	Error during code generation.

17.7.1.19.2.8 GenerateCodeInEx

Generates code for all project items of the project in the specified language. The output location is determined by properties of the project and project items. Returns an object that can be used to iterate through all messages issued by the code generation process. These messages are the same as those shown in the **Messages** window of MapForce.

Signature

```
GenerateCodeInEx(in i_nLanguage:ENUMProgrammingLanguage) -> ErrorMarkers
```

Parameters

Name	Type	Description
i_nLanguage	ENUMProgrammingLanguage	Specifies the programming language in which code should be generated.

Errors

Error code	Description
1500	The object is no longer valid.
1501	Invalid address for the return parameter was specified.
1706	Error during code generation.

17.7.1.19.2.9 InsertWebService

Inserts a new Web service project into the project's Web service folder. If **i_bGenerateMappings** is true, initial mapping documents for all ports get generated automatically.

Signature

```
InsertWebService(in i_strWSDLFile:String, in i_strService:String, in i_strPort:String,
in i_bGenerateMappings:Boolean) -> ProjectItem
```

Parameters

Name	Type	Description
i_strWSDLFile	`String`	Specifies the path of the WSDL file to add.
i_strService	`String`	Specifies the name of the Web service to add.
i_strPort	`String`	Specifies the port of the Web service to add.
i_bGenerateMappings	`Boolean`	If this parameter is true, initial mapping documents for all ports get generated automatically.

Errors

Error code	Description
1500	The object is no longer valid.
1501	WSDL file can not be found or is invalid. Service or port names are invalid. Invalid address for the return parameter was specified.
1503	Operation not supported by current edition.

17.7.1.19.2.10 Save

Saves the project to the file defined by `FullName`.

Signature

```
Save() -> Void
```


(C) 2015-2021 Altova GmbH

Errors

Error code	Description
1500	The object is no longer valid.
1502	Can't save to file.

17.7.1.19.3 Events

17.7.1.19.3.1 OnProjectClosed

This event is triggered when the project is closed. The project object passed into the event handler should not be accessed. The corresponding open event is `Application.OnProjectOpened`.

Signature

```
OnProjectClosed(in i_ipProject:Project) : Void
```

17.7.1.20 ProjectItem

A `ProjectItem` object represents one item in a project tree.

Properties to navigate the object model:

- `Application`
- `Parent`

Project tree navigation:

- `Count`
- `Item`
- `_NewEnum`

Project item properties:

- `Kind`
- `Name`
- `WSDLFile` (only available to Web service project items)
- `QualifiedName` (only available to Web service project items)

Project tree manipulation:

- `AddActiveFile` (only available to folder items)
- `AddFile` (only available to folder items)
- `CreateFolder` (only available to folder items)
- `CreateMappingForProject` (only available to Web service operations)

- Remove

Document access:

- Open (only available to mapping items and Web service operations)

Code-generation:

- CodeGenSettings_UseDefault
- CodeGenSettings_OutputFolder
- CodeGenSettings_Language
- GenerateCode
- GenerateCodeEx
- GenerateCodeIn
- GenerateCodeInEx

For examples of how to use the properties and methods listed above, see Example: Project Tasks. For operations with Web services, the MapForce Enterprise edition is required.

Properties

Name	Description
_NewEnum	Read-only. This property supports language specific standard enumeration. For examples, see Project.Item or Project._NewEnum.
Application	Read-only. Retrieves the top-level application object.
CodeGenSettings_Language	Gets or sets the language to be used with GenerateCode or Project.GenerateCode. This property is consulted only if CodeGenSettings_UseDefault is set to false.
CodeGenSettings_OutputFolder	Gets or sets the output directory to be used with GenerateCode, GenerateCodeIn, Project.GenerateCode or Project.GenerateCodeIn. This property is consulted only if CodeGenSettings_UseDefault is set to false.
CodeGenSettings_UseDefault	Gets or sets whether output directory and code language are used as defined by either (a) the parent folders, or (b) the project root. This property is used with calls to GenerateCode, GenerateCodeIn, Project.GenerateCode and Project.GenerateCodeIn. If this property is set to false, the values of CodeGenSettings_OutputFolder and CodeGenSettings_Language are used to generate code for this project item.
Count	Read-only. Retrieves the number of children of this project item. See also Item. For examples, see Project.Item or Project._NewEnum.
Item	Read-only.

(C) 2015-2021 Altova GmbH

Name	Description
	Returns the child at `n` position of this project item. The index is 1-based (the first index is 1). The largest valid index is `ProjectItem.Count`. For an alternative, see `ProjectItem._NewEnum`. For examples, see `Project.Item` or `Project._NewEnum`.
`Kind`	Read-only. Retrieves the kind of the project item. Availability of some properties and the applicability of certain methods is restricted to specific kinds of project items. The description of all methods and properties contains information about these restrictions.
`Name`	Retrieves or sets the name of a project item. The name of most items is read-only. Exceptions are user-created folders, the names of which can be altered after creation.
`Parent`	Read-only. Retrieves the project that this item is a child of. Has the same effect as `Application.ActiveProject`.
`QualifiedName`	Read-only. Retrieves the qualified name of a Web service item.
`WSDLFile`	Read-only. Retrieves the file name of the WSDL file defining the Web service that hosts the current project item.

Methods

Name	Description
`AddActiveFile`	Adds the currently active document to this project item if it is a valid child. Otherwise, the document is added to the Mapping Folder of the project's root.
`AddFile`	Adds the specified document to this project item if it is a valid child. Otherwise, the document is added to the Mapping Folder of the project's root.
`CreateFolder`	Creates a new folder as a child of this project item.
`CreateMappingForProject`	Creates an initial mapping document for a Web service operation and saves it to **i_strFileName**. When using `Project.InsertWebService` you can set the **i_bGenerateMappings** flag to let MapForce automatically generate initial mappings for all ports.
`GenerateCode`	Generates code for this project item and its children. The code language and output location is determined by `CodeGenSettings_UseDefault`, `CodeGenSettings_Language` and `CodeGenSettings_OutputFolder`. Children of this project item can have their own property settings related to code-generation.

Name	Description
GenerateCodeEx	Generates code for this project item and its children. The code language and output location are determined by `CodeGenSettings_UseDefault`, `CodeGenSettings_Language` and `CodeGenSettings_OutputFolder`. Children of this project item can have their own property settings related to code-generation.
GenerateCodeIn	Generates code for the project item and its children in the specified language. The output location is determined by `CodeGenSettings_UseDefault` and `CodeGenSettings_OutputFolder`. Children of this project item can have their own property settings related to code-generation.
GenerateCodeInEx	Generates code for the project item and its children in the specified language. The output location is determined by `CodeGenSettings_UseDefault` and `CodeGenSettings_OutputFolder`. Children of this project item can have their own property settings related to code-generation. An object that can be used to iterate through all messages issued by the code generation process is returned. These messages are the same as those shown in the Messages window of MapForce.
Open	Opens the project item as a document or makes the corresponding document the active one, if it is already open. The project item must be a MapForce mapping or, for Enterprise edition only, Web service operation.
Remove	Remove this project item and all its children from the project tree.

Events

Name	Description
OnModifiedFlagChanged	Occurs when the `ProjectItem`'s modification status changes.
OnProjectClosed	This event is triggered when the project is closed. The project object passed into the event handler should not be accessed. The corresponding open event is `Application.OnProjectOpened`.

17.7.1.20.1 Properties

17.7.1.20.1.1 _NewEnum

This property supports language specific standard enumeration. For examples, see `Project.Item` or `Project._NewEnum`.

(C) 2015-2021 Altova GmbH

Signature

```
_NewEnum : IUnknown
```

Errors

Error code	Description
1700	The object is no longer valid.

17.7.1.20.1.2 Application

Retrieves the top-level application object.

Signature

```
Application : Application
```

Errors

Error code	Description
1700	The object is no longer valid.
1701	Invalid address for the return parameter was specified.

17.7.1.20.1.3 CodeGenSettings_Language

Gets or sets the language to be used with `GenerateCode` or `Project.GenerateCode`. This property is consulted only if `CodeGenSettings_UseDefault` is set to false.

Signature

```
CodeGenSettings_Language : ENUMProgrammingLanguage
```

Errors

Error code	Description
1700	The object is no longer valid.
1701	Invalid language or invalid address for the return parameter was specified.

17.7.1.20.1.4 CodeGenSettings_OutputFolder

Gets or sets the output directory to be used with `GenerateCode`, `GenerateCodeIn`, `Project.GenerateCode` or `Project.GenerateCodeIn`. This property is consulted only if `CodeGenSettings_UseDefault` is set to false.

Signature

```
CodeGenSettings_OutputFolder : String
```

Errors

Error code	Description
1700	The object is no longer valid.
1701	An invalid output folder or an invalid address for the return parameter was specified.

17.7.1.20.1.5 CodeGenSettings_UseDefault

Gets or sets whether output directory and code language are used as defined by either (a) the parent folders, or (b) the project root. This property is used with calls to `GenerateCode`, `GenerateCodeIn`, `Project.GenerateCode` and `Project.GenerateCodeIn`. If this property is set to false, the values of `CodeGenSettings_OutputFolder` and `CodeGenSettings_Language` are used to generate code for this project item.

Signature

```
CodeGenSettings_UseDefault : Boolean
```

Errors

Error code	Description
1700	The object is no longer valid.
1701	Invalid address for the return parameter was specified.

17.7.1.20.1.6 Count

Retrieves the number of children of this project item. See also `Item`. For examples, see `Project.Item` or `Project._NewEnum`.

Signature

```
Count : Integer
```


(C) 2015-2021 Altova GmbH

Errors

Error code	Description
1700	The object is no longer valid.

17.7.1.20.1.7 Item

Returns the child at `n` position of this project item. The index is 1-based (the first index is 1). The largest valid index is `ProjectItem.Count`. For an alternative, see `ProjectItem._NewEnum`. For examples, see `Project.Item` or `Project._NewEnum`.

Signature

```
Item(in n:Integer) : ProjectItem
```

Errors

Error code	Description
1700	The object is no longer valid.

17.7.1.20.1.8 Kind

Retrieves the kind of the project item. Availability of some properties and the applicability of certain methods is restricted to specific kinds of project items. The description of all methods and properties contains information about these restrictions.

Signature

```
Kind : ENUMProjectItemType
```

Errors

Error code	Description
1700	The object is no longer valid.
1701	Invalid address for the return parameter was specified.

17.7.1.20.1.9 Name

Retrieves or sets the name of a project item. The name of most items is read-only. Exceptions are user-created folders, the names of which can be altered after creation.

Signature

```
Name : String
```

Errors

Error code	Description
1700	The object is no longer valid.
1701	Invalid address for the return parameter was specified.
1702	Project item does not allow to alter its name.

17.7.1.20.1.10 Parent

Retrieves the project that this item is a child of. Has the same effect as `Application.ActiveProject`.

Signature

```
Parent : Project
```

Errors

Error code	Description
1700	The object is no longer valid.
1701	Invalid address for the return parameter was specified.

17.7.1.20.1.11 QualifiedName

Retrieves the qualified name of a Web service item.

Signature

```
QualifiedName : String
```

Errors

Error code	Description
1700	The object is no longer valid.
1701	Invalid address for the return parameter was specified.

(C) 2015-2021 Altova GmbH

Error code	Description
1702	The project item is not a part of a Web service.

17.7.1.20.1.12 WSDLFile

Retrieves the file name of the WSDL file defining the Web service that hosts the current project item.

Signature

```
WSDLFile : String
```

Errors

Error code	Description
1700	The object is no longer valid.
1701	Invalid address for the return parameter was specified.
1702	The project item is not a part of a Web service.

17.7.1.20.2 Methods

17.7.1.20.2.1 AddActiveFile

Adds the currently active document to this project item if it is a valid child. Otherwise, the document is added to the Mapping Folder of the project's root.

Signature

```
AddActiveFile() -> ProjectItem
```

Errors

Error code	Description
1700	The object is no longer valid.
1701	The file name is empty. Invalid address for the return parameter was specified.
1703	No active document is available.
1704	Active documents needs to be given a path name before it can be added to the project.

Error code	Description
1705	Mapping could not be assigned to project. The file does not exist or is not a MapForce mapping. Maybe the file is already assigned to the target folder.

17.7.1.20.2.2 AddFile

Adds the specified document to this project item if it is a valid child. Otherwise, the document is added to the Mapping Folder of the project's root.

Signature

```
AddFile(in i_strFilePath:String) -> ProjectItem
```

Parameters

Name	Type	Description
i_strFilePath	`String`	The path of the document to add.

Errors

Error code	Description
1700	The object is no longer valid.
1701	The file name is empty. Invalid address for the return parameter was specified.
1705	Mapping could not be assigned to project. The file does not exist or is not a MapForce mapping. Maybe the file is already assigned to the target folder.

17.7.1.20.2.3 CreateFolder

Creates a new folder as a child of this project item.

Signature

```
CreateFolder(in i_strFolderName:String) -> ProjectItem
```

Parameters

Name	Type	Description
i_strFolderName	`String`	The name of the folder to create.

(C) 2015-2021 Altova GmbH

Errors

Error code	Description
1700	The object is no longer valid.
1701	Invalid folder name or invalid address for the return parameter was specified.
1702	The project item does not support children.

17.7.1.20.2.4 CreateMappingForProject

Creates an initial mapping document for a Web service operation and saves it to **i_strFileName**. When using `Project.InsertWebService` you can set the **i_bGenerateMappings** flag to let MapForce automatically generate initial mappings for all ports.

Signature

```
CreateMappingForProject(in i_strFileName:String) -> ProjectItem
```

Parameters

Name	Type	Description
i_strFileName	`String`	Specifies the path where the mapping should be saved.

Errors

Error code	Description
1700	The object is no longer valid.
1701	Invalid address for the return parameter was specified.
1707	Cannot create new mapping. The project item does not support auto-creation of initial mappings or a mapping already exists.
1708	Operation not supported in current edition.

17.7.1.20.2.5 GenerateCode

Generates code for this project item and its children. The code language and output location is determined by `CodeGenSettings_UseDefault`, `CodeGenSettings_Language` and `CodeGenSettings_OutputFolder`. Children of this project item can have their own property settings related to code-generation.

Signature

```
GenerateCode() -> Void
```

Errors

Error code	Description
1700	The object is no longer valid.
1706	Error during code generation.

17.7.1.20.2.6 GenerateCodeEx

Generates code for this project item and its children. The code language and output location are determined by `CodeGenSettings_UseDefault`, `CodeGenSettings_Language` and `CodeGenSettings_OutputFolder`. Children of this project item can have their own property settings related to code-generation.

Signature

```
GenerateCodeEx() -> ErrorMarkers
```

Errors

Error code	Description
1700	The object is no longer valid.
1701	Invalid address for the return parameter was specified.
1706	Error during code generation.

17.7.1.20.2.7 GenerateCodeIn

Generates code for the project item and its children in the specified language. The output location is determined by `CodeGenSettings_UseDefault` and `CodeGenSettings_OutputFolder`. Children of this project item can have their own property settings related to code-generation.

Signature

```
GenerateCodeIn(in i_nLanguage:ENUMProgrammingLanguage) -> Void
```


(C) 2015-2021 Altova GmbH

Parameters

Name	Type	Description
i_nLanguage	ENUMProgrammingLanguage	Specifies the programming language for code generation.

Errors

Error code	Description
1700	The object is no longer valid.
1701	Invalid language specified.
1706	Error during code generation.

17.7.1.20.2.8 GenerateCodeInEx

Generates code for the project item and its children in the specified language. The output location is determined by `CodeGenSettings_UseDefault` and `CodeGenSettings_OutputFolder`. Children of this project item can have their own property settings related to code-generation.

An object that can be used to iterate through all messages issued by the code generation process is returned. These messages are the same as those shown in the Messages window of MapForce.

Signature

```
GenerateCodeInEx(in i_nLanguage:ENUMProgrammingLanguage) -> ErrorMarkers
```

Parameters

Name	Type	Description
i_nLanguage	ENUMProgrammingLanguage	Specifies the programming language for code generation.

Errors

Error code	Description
1700	The object is no longer valid.
1701	Invalid language specified or invalid address for the return parameter was specified.
1706	Error during code generation.

17.7.1.20.2.9 Open

Opens the project item as a document or makes the corresponding document the active one, if it is already open. The project item must be a MapForce mapping or, for Enterprise edition only, Web service operation.

Signature

```
Open() -> Document
```

Errors

Error code	Description
1700	The object is no longer valid.
1701	Invalid address for the return parameter was specified.
1702	The project item does not refer to a MapForce mapping file.
1708	Operation not supported in current edition.

17.7.1.20.2.10 Remove

Remove this project item and all its children from the project tree.

Signature

```
Remove() -> Void
```

Errors

Error code	Description
1700	The object is no longer valid.

17.7.1.20.3 Events

17.7.1.20.3.1 OnModifiedFlagChanged

Occurs when the `ProjectItem`'s modification status changes.

Signature

```
OnModifiedFlagChanged(in i_bIsModified:Boolean) : Void
```


(C) 2015-2021 Altova GmbH

17.7.1.20.3.2 OnProjectClosed

This event is triggered when the project is closed. The project object passed into the event handler should not be accessed. The corresponding open event is `Application.OnProjectOpened`.

Signature

```
OnProjectClosed(in i_ipProject:Project) : Void
```

17.7.2 Enumerations

17.7.2.1 ENUMApacheAxisVersion (obsolete)

This enumeration type is obsolete.

Members

eApacheAxisVersion_Axis = 1

eApacheAxisVersion_Axis2 = 2

17.7.2.2 ENUMApplicationStatus

Enumeration values to indicate the status of the application.

Members

eApplicationRunning = 0

eApplicationAfterLicenseCheck = 1

eApplicationBeforeLicenseCheck = 2

eApplicationConcurrentLicenseCheckFailed = 3

eApplicationProcessingCommandLine = 4

17.7.2.3 ENUMAppOutputLine_Severity

Enumeration values to identify the severity of an `AppOutputLine`.

Members

eSeverity_Undefined = -1

eSeverity_Info = 0

eSeverity_Warning = 1

eSeverity_Error = 2

eSeverity_CriticalError = 3

eSeverity_Success = 4

eSeverity_Summary = 5

eSeverity_Progress = 6

eSeverity_DataEdit = 7

eSeverity_ParserInfo = 8

eSeverity_PossibleInconsistencyWarning = 9

eSeverity_Message = 10

eSeverity_Document = 11

eSeverity_Rest = 12

eSeverity_NoSelect = 13

eSeverity_Select = 14

eSeverity_Autoinsertion = 15

eSeverity_GlobalResources_DefaultWarning = 16

eSeverity_XPath_Styles_Changed = 17

eSeverity_XPath_Styles_Unchanged = 18

eSeverity_XPath_Styles_Skipped = 19

eSeverity_XPath_ComboBox_Values_Changed = 20

eSeverity_XPath_ComboBox_Values_Unchanged = 21

eSeverity_XPath_ComboBox_Values_Skipped = 22

eSeverity_XPath_Assertions_Changed = 23

eSeverity_XPath_Assertions_Unchanged = 24

eSeverity_XPath_Assertions_Skipped = 25

(C) 2015-2021 Altova GmbH

17.7.2.4 ENUMAppOutputLine_TextDecoration

Enumeration values for the different kinds of text decoration of an `AppOutputLine`.

Members

eTextDecorationDefault = 0

eTextDecorationBold = 1

eTextDecorationDebugValues = 2

eTextDecorationDB_ObjectName = 3

eTextDecorationDB_ObjectLink = 4

eTextDecorationDB_ObjectKind = 5

eTextDecorationDB_TimeoutValue = 6

eTextDecorationFind_MatchingString = 7

eTextDecorationValidation_Speclink = 8

eTextDecorationValidation_ErrorPosition = 9

eTextDecorationValidation_UnkownParam = 10

17.7.2.5 ENUMCodeGenErrorLevel

Enumeration values to identify severity of code generation messages.

Members

eCodeGenErrorLevel_Information = 0

eCodeGenErrorLevel_Warning = 1

eCodeGenErrorLevel_Error = 2

eCodeGenErrorLevel_Undefined = 3

17.7.2.6 ENUMComponentDatapointSide

Enumeration values to indicate the side of a datapoint on its component. See also `Component.GetRootDatapoint`.

Members

eDatapointSideInput = 0

eDatapointSideOutput = 1

17.7.2.7 ENUMComponentSubType

Enumeration values to indicate component sub types.

Members

eComponentSubType_None = 0

eComponentSubType_Text_EDI = 1

eComponentSubType_Text_Flex = 2

eComponentSubType_Text_CSVFLF = 3

17.7.2.8 ENUMComponentType

Enumeration values to indicate component types.

Members

eComponentType_Unknown = 0

eComponentType_XML = 1

eComponentType_DB = 2

eComponentType_Text = 3

eComponentType_Excel = 4

eComponentType_WSDL = 5

eComponentType_XBRL = 6

eComponentType_Input = 7

eComponentType_JSON = 8

17.7.2.9 ENUMComponentUsageKind

Enumeration values to indicate component usage kind.

Members

eComponentUsageKind_Unknown = 0

eComponentUsageKind_Instance = 1

(C) 2015-2021 Altova GmbH

eComponentUsageKind_Input = 2

eComponentUsageKind_Output = 3

eComponentUsageKind_Variable = 4

eComponentUsageKind_String = 5

17.7.2.10 ENUMConnectionType

Enumeration values to indicate the type of a connection. See also `Connection.ConnectionType`.

Members

eConnectionTypeTargetDriven = 0

eConnectionTypeSourceDriven = 1

eConnectionTypeCopyAll = 2

17.7.2.11 ENUMDOMType

Enumeration values to specify the DOM type used by generated C++ mapping code.

NOTE: The value `eDOMType_xerces` is obsolete. `eDOMType_xerces3` indicates Xerces 3.x usage. Obsolete in this context means that this value is not supported and should not be used.

Members

eDOMType_xerces = 1 (obsolete)

eDOMType_xerces3 = 2

eDOMType_msxml6 = 3

17.7.2.12 ENUMLibType

Enumeration values to specify the library type used by the generated C++ mapping code.

Members

eLibType_static = 0

eLibType_dll = 1

17.7.2.13 ENUMProgrammingLanguage

Enumeration values to select a programming language.

Members

eUndefinedLanguage = -1

eJava = 0

eCpp = 1

eCSharp = 2

eXSLT = 3

eXSLT2 = 4

eXQuery = 5

eXSLT3 = 6

17.7.2.14 ENUMProjectItemType

Enumeration to identify the different kinds of project items that can be children of `Project` or folder-like `ProjectItems`. See also `ProjectItem.Kind`.

Members

eProjectItemType_MappingFolder = 0

eProjectItemType_Mapping = 1

eProjectItemType_WebServiceFolder = 2

eProjectItemType_WebServiceRoot = 3

eProjectItemType_WebServiceService = 4

eProjectItemType_WebServicePort = 5

eProjectItemType_WebServiceOperation = 6

eProjectItemType_ExternalFolder = 7

eProjectItemType_LibraryFolder = 8

eProjectItemType_ResourceFolder = 9

eProjectItemType_VirtualFolder = 10

eProjectItemType_Count = 11

(C) 2015-2021 Altova GmbH

eProjectItemType_Invalid = -1

17.7.2.15 ENUMProjectType

Enumeration values to select a project type for generated C# and C++ mapping code.

Members

eVisualStudio2010Project = 6

eVisualStudio2013Project = 7

eVisualStudio2015Project = 8

eVisualStudio2017Project = 9

eVisualStudio2019Project = 10

eDotNetCore3_1 = 11

eDotNet5_0 = 12

17.7.2.16 ENUMSearchDatapointFlags

Enumeration values used as bit-flags; to be used as combination of flags when searching for a datapoint. See also
`GetChild`.

Members

eSearchDatapointElement = 1

eSearchDatapointAttribute = 2

17.7.2.17 ENUMViewMode

Enumeration values to select a MapForce view.

Members

eMapForceView = 0

eXSLView = 1

eOutputView = 2

18 ActiveX Integration

このセクションで説明されているMapForce ユーザーインターフェイスとActiveX コントロール機能を使用するアプリケーションに統合することができます。 ActiveX 技術によりC++、C#、VB.NET、HTML などの統合のために使用される広範囲の言語が有効化されます(HTML 内に統合されているActiveX コンポーネントは、Microsoft Internet Explorer バージョンとActiveX をサポートすると共に作動する必要があります)。全てのコンポーネントはフルOLE コントロールです。Java への統合はラッパークラスにより与えられます。

> ActiveX コントロールを、カスタムコードに統合するには、MapForce Integration Package をインストールしてください(次を参照してください: http://www.altova.com/ide_integration.html)。すべてのMapForce を最初に、そして、MapForce Integration Package そして他の必要条件を言語とプラットフォームに応じてインストールしてください(次を参照してください: 必要条件)。

統合の2つの異なるレベル間を柔軟に選択することができます: アプリケーションレベルとドキュメントレベル.

アプリケーションレベルの統合は、(メニュー、ツールバー、ペインを含む) MapForce のフルインターフェイスを ActiveX コントロールをカスタムアプリケーションとして埋め込むことを意味します。例えば、多くのシナリオでは、カスタムアプリケーションを、MapForce グラフィカルユーザーインターフェイスを埋め込む1つのフォームから構成することができます。

ドキュメントレベルの統合よりもこのアプローチより先簡単に実装することができますが、カスタム必要条件に従いを柔軟的にMapForce グラフィカルユーザーインターフェイスを構成するためには適切ではない可能性があります。ドキュメントレベルの統合は、アプリケーションにMapForce を1つ1つ埋め込むことを意味します。メインMapForce コントロールのみではなく、メインドキュメント エディターウインドウの実装も含まれます。また、オプションで、追加ウインドウの実装も含まれます。このアプローチはGUI の構成に柔軟性を与えますが、選択された言語によりActiveX コントロールとの高度なインタラクションが必要とされます。

セクション アプリケーションレベルの統合とドキュメントレベルの統合 は、それぞれのレベルの主要なステップに関して説明しています。ActiveX 統合のサンプル セクションは、C#、HTML、および、Java のサンプルを提供します。これらのサンプルを確認することにより、正しい選択が直ぐに行うことができます。オブジェクトレファレンス セクションは、プロパティとメソッドに加え統合のために使用できるすべてのCOM オブジェクトに付いて説明しています。

MapForce をVisual Studio プラグインとして使用するための情報に関しては、次を参照してください: Visual Studio 内のMapForce。

(C) 2015-2021 Altova GmbH

18.1 必要条件

MapForce ActiveX コントロールをカスタムアプリケーションに統合するには、以下が使用中のコンピューターにインストールされている必要があります:

- MapForce
- MapForce Integration Package は、次からダウンロードできます: http://www.altova.com/ide_integration.html

64 ビット ActiveX コントロールを統合するには、MapForce の64 ビット バージョンとMapForce Integration Package をインストールする必要があります。Visual Studio を使用して、Microsoft .NET プラットフォームの下で開発されたアプリケーションに関しては、MapForce の32 ビット と64 ビット バージョンとMapForce Integration Package について下で説明されているとおりインストールします。

メモ W indows XP とW indows Server 2003 で統合パッケージが正確に作動するためには、再頒布可能パッケージを手動でインストールしてください: https://www.microsoft.com/en-US/download/details.aspx?id=48145.

Microsoft .NET (C#, VB.NET) with Visual Studio

MapForce ActiveX コントロールを、Microsoft .NET の下で開発された32 ビット アプリケーションに統合するには、以下を使用中のコンピューターにインストールしてください:

- Microsoft .NET Framework 4.0 またはは以降
- Visual Studio 2010/2012/2013/2015/2017/2019
- MapForce 32 ビットとMapForce Integration Package 32 ビット
- ActiveX コントロールがVisual Studio ツールボックスに追加されている必要があります(次を参照してください: ActiveX コントロールをツールボックスに追加する).

64 ビット ActiveX コントロールを統合する場合、次の必要条件が上記に加え満たされる必要があります:

- MapForce 32 ビットとMapForce Integration Package 32 ビット をインストールしてください(Visual Studio は32 ビットで作動するため、32 ビット ActiveX コントロールを Visual Studio デザイナーに与える必要があります。)
- MapForce 64 ビットとMapForce Integration Package 64 ビット をインストールしてください(実際の64 ビット ActiveX コントロールをランタイムにカスタムアプリケーションに与えます。)
- Visual Studio 内で、64 ビットビルド更新を作成し、この更新を使用してアプリケーションをビルドします。サンプルに関しては、次を参照してください: サンプルC# ソリューションの実行。

Java

Eclipse 開発環境を使用して、MapForce ActiveX コントロールを Java アプリケーションに統合するには、次を使用中のコンピューターにインストールしてください:

- Java Runtime Environment(JRE) またはJava Development Kit(JDK) 7 またはは以降
- Eclipse
- MapForce and MapForce Integration Package

メモ MapForce ActiveX コントロールの64 ビット バージョンを作動するには、Eclipse の64 ビット バージョン、および、MapForce とMapForce Integration Package の64 ビット バージョン ofを使用してください。

クライアントコンピューター上でのMapForce 統合とデプロイ

.NET アプリケーションを作成し他のクライアントに配布する場合は、クライアントコンピューター上に次をインストールしてください：

- MapForce
- MapForce Integration Package
- カスタムの統合コード、または、アプリケーション。

(C) 2015-2021 Altova GmbH

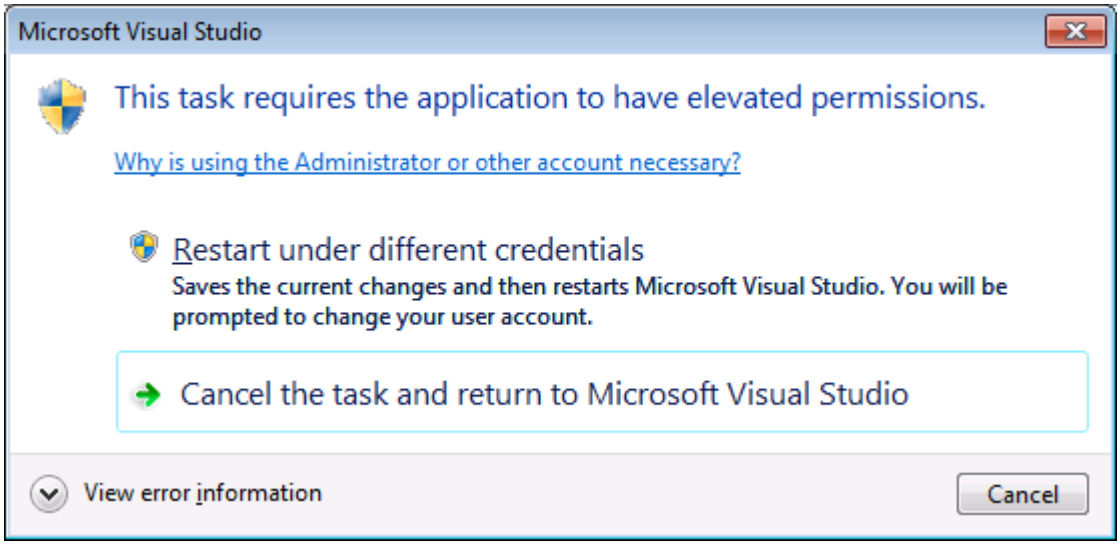
18.2 ActiveX コントロールをツールボックスに追加する

Visual Studio を使用して開発されたアプリケーション内で、MapForce ActiveX コントロールを使用するには、コントロールが最初に Visual Studio ツールボックスに追加される必要があります:

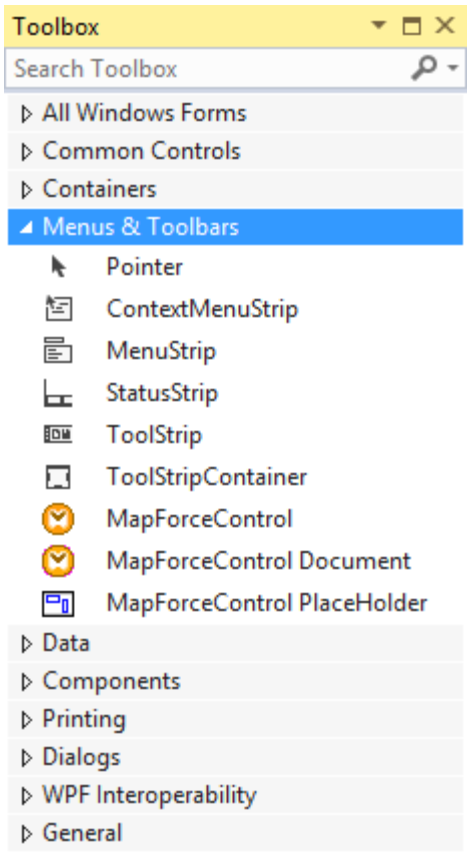
1. Visual Studio の「ツール」メニューから、「ツールボックスアイテムの選択」をクリックします。
2. 「COM コンポーネント」タブ、から、MapForceControl, MapForceControl ドキュメント、および、MapForceControl プレースホルダーの横のチェックボックスを選択します。

上記のコントロールが使用することができない場合、下のステップに従います:

1. 「COM コンポーネント」タブから「参照」をクリックします。**MapForceControl.ocx** を MapForce インストールからフォルダーを選択します。MapForce Integration Package をインストールすることを忘れないでください。それ以外の場合、このファイルを使用することはできません、次を参照してください: 必要条件.
2. を更に高いパーミッションと共に再起動するようにプロンプトされると、「異なる資格情報下で再起動する」をクリックします。



上記のステップが完了すると、MapForce ActiveX コントロールは、Visual Studio ツールボックス内で使用できるようになります。



メモ　アプリケーションレベルの統合では、**MapForceControl** ActiveX コントロールのみが使用されます（次を参照してください: アプリケーションレベルの統合）。 MapForceControl ドキュメント と MapForceControl プレースホルダー コントロールは、ドキュメントレベルの統合のために使用されます（次を参照してください: ドキュメントレベルの統合）。

(C) 2015-2021 Altova GmbH

18.3 アプリケーションレベルの統合

アプリケーションレベルの統合により使用中のアプリケーションのウィンドウにMapForce のインターフェイスを埋め込むことができます。 この種類の統合により、すべてのメニュー、ツールバー、ステータスバー、ドキュメントウィンドウ、および、ヘルパーウィンドウを含むMapForce のユーザーインターフェイスを使用することができます。アプリケーションのユーザーインターフェイスのカスタム化はMapForce が提供する機能に限られています。これには、ヘルパーウィンドウの整理とサイズ調整、及びメニューとツールバーのカスタム化が含まれています。

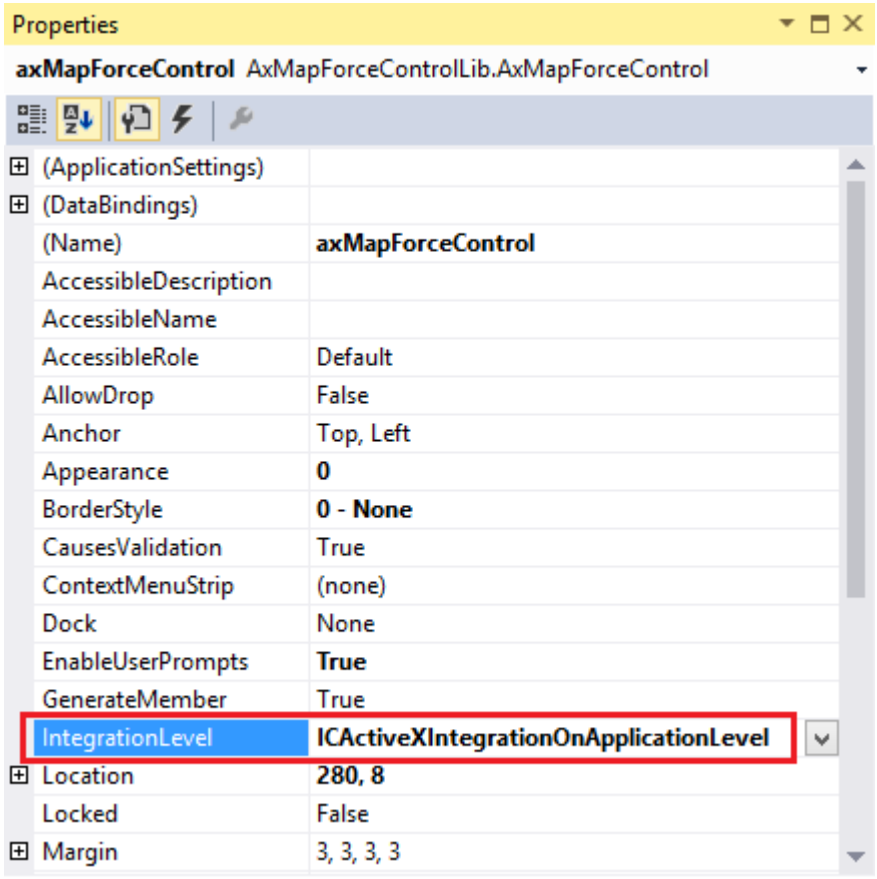
アプリケーションレベルで統合する場合、統合する必要のある唯一のActiveX コントロールは`MapForceControl` です。`MapForceControlDocument` または`MapForceControlPlaceHolder` ActiveX コントロールをインスタンス化、または、アクセス使用としないでください。

初期化、または、MapForce の振る舞いを自動化するには、`MapForceControl`.のために説明されているプロパティ、メソッド、とイベントを使用してください。MapForce 機能への複雑なアクセスのために`MapForceControl.application` を使用することを考慮してください。

MapForce アプリケーションをHTML ページ、に埋め込むサンプルに関しては、次を参照してください: アプリケーションレベルのHTML 統合。

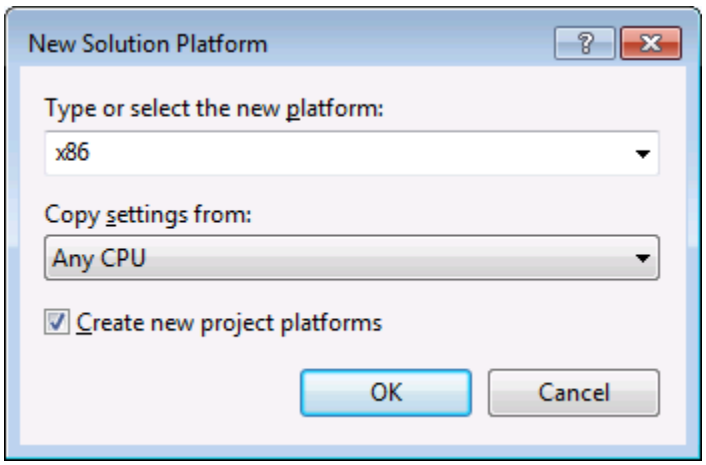
Visual Studio を使用するC# またはVB.NET でのMapForce ActiveX コントロールをアプリケーションレベルで統合する基本的なアプリケーションを作成するステップは、以下のとおりです:

1. すべての必要条件が満たされていることを確認してください(次を参照してください: 必要条件).
2. 新規のVisual Studio W indows Forms プロジェクトを新規の空のフォーム共に作成してください。
3. まだ行っていない場合、ActiveX コントロールをツールボックスに追加します(次を参照してください: ActiveX コントロールをツールボックスに追加する)。
4. ルーツボックスから、**MapForceControl** をドラッグして、新規のフォームにドロップしてください。
5. フォームで**MapForceControl** を選択し、プロパティウィンドウ内で**IntegrationLevel** プロパティを**ICActiveXIntegrationOnApplicationLevel** に設定します。



6. ビルド(x86、x64)するプラットフォームに一致するビルドプラットフォームの更新を作成します。ビルドの更新の作成の方法は以下のとおりです:

   a. Visual Studio 内のソリューションを右クリックし、「更新 マネージャー」を選択します。
   b. 「アクティブなソリューションプラットフォーム」から、「新規作成...」を選択し、x86 または x64 更新を選択します(このサンプルでは、**x86**)。



(C) 2015-2021 Altova GmbH

Visual Studio 内でソリューションをビルドして実行する準備が整うと、ターゲットプラットフォーム(x86、x64)に一致する更新をしようしてビルドすることを確認してください。

18.4 ドキュメントレベルの統合

アプリケーションレベルの統合に比べ、ドキュメントレベルの統合複雑ですが、ActiveX コントロールを使用することによりアプリケーションに MapForce 機能を柔軟的に埋め込むことができます。このアプローチにより、コードは MapForce ユーザーインターフェイスの次のパーツに部分的にアクセスすることができます:

- ドキュメントの編集ウィンドウ
- プロジェクトウィンドウ
- ライブラリウィンドウ
- 概要ウィンドウ
- メッセージウィンドウ

アプリケーションレベルの統合 で説明されているとおり、ActiveX アプリケーションレベルの統合のためには、**MapForceControl** のみが必要とされる唯一のコントロールです。しかしながら、ActiveX ドキュメントレベルの統合の場合は、機能 MapForce が次の ActiveX コントロールにより与えられます:

1. **MapForceControl**
2. MapForceControl ドキュメント
3. MapForceControl プレースホルダ

これらのコントロールは MapForce のアプリケーション インストール フォルダーで使用することのできる **MapForceControl.ocx** ファイルにより与えられます。Visual Studio を使用して ActiveX 統合を開発すると、これらのコントロールを Visual Studio ツールボックスに追加する必要があります (次を参照してください: ActiveX コントロールをツールボックスに追加する)。

ドキュメントレベルでの ActiveX コントロールのアプリケーションへの統合の基本のステップは、以下のとおりです:

1. 最初に、アプリケーション内で **MapForceControl** をインスタンス化します。「**MapForceControl ドキュメント**」と「**MapForceControl プレースホルダ**」コントロールへのサポートが有効化されるために、このコントロールの統合は必須です。`IntegrationLevel` プロパティを「ICActiveXIntegrationOnDocumentLevel」(または「1」)に設定することは重要です。ユーザーに対してコントロールを非表示にするには、「Visible」プロパティを「False」に設定します。

メモ ドキュメントレベルで統合する場合、未知の結果が生成される可能性があるため、**MapForceControl** の「開く」メソッドを使用しないでください。**MapForceControl ドキュメント** と **MapForceControl プレースホルダ** の対応するメソッドを代わりに使用して開いてください。

2. **MapForceControl ドキュメント** のインスタンスを少なくとも 1 つアプリケーション内に作成します。このコントロールは、アプリケーションに MapForce のドキュメントの編集ウィンドウを与え、必要であれば、複数回インスタンス化することができます。

   既存のファイルを開くために、メソッド「開く」を使用します。ドキュメントに関連した機能にアクセスするには、メソッドとプロパティ「ドキュメント」を介してアクセスすることのできる「パス」と「保存」を使用します。

メモ コントロールは、読み取り専用モードをサポートしません。プロパティ「**読み取り専用**」の値は無視されます。

3. 任意で、アプリケーションに、アプリケーションで使用することのできる(ドキュメント ウィンドウ以外の)各追加ウィンドウのために **MapForceControl プレースホルダ** コントロールを追加することができます。

   MapForceControl プレースホルダ のインスタンスによりアプリケーションウィンドウに選択して MapForce の追加ウィンドウを埋め込むことができます。ウィンドウの種類 (例えば、プロジェクトウィンドウ) はプロパティ「PlaceholderWindowID」により定義することができます。有効なウィンドウの識別子のためには、次を参照してください: MapForceControlPlaceholderWindow.

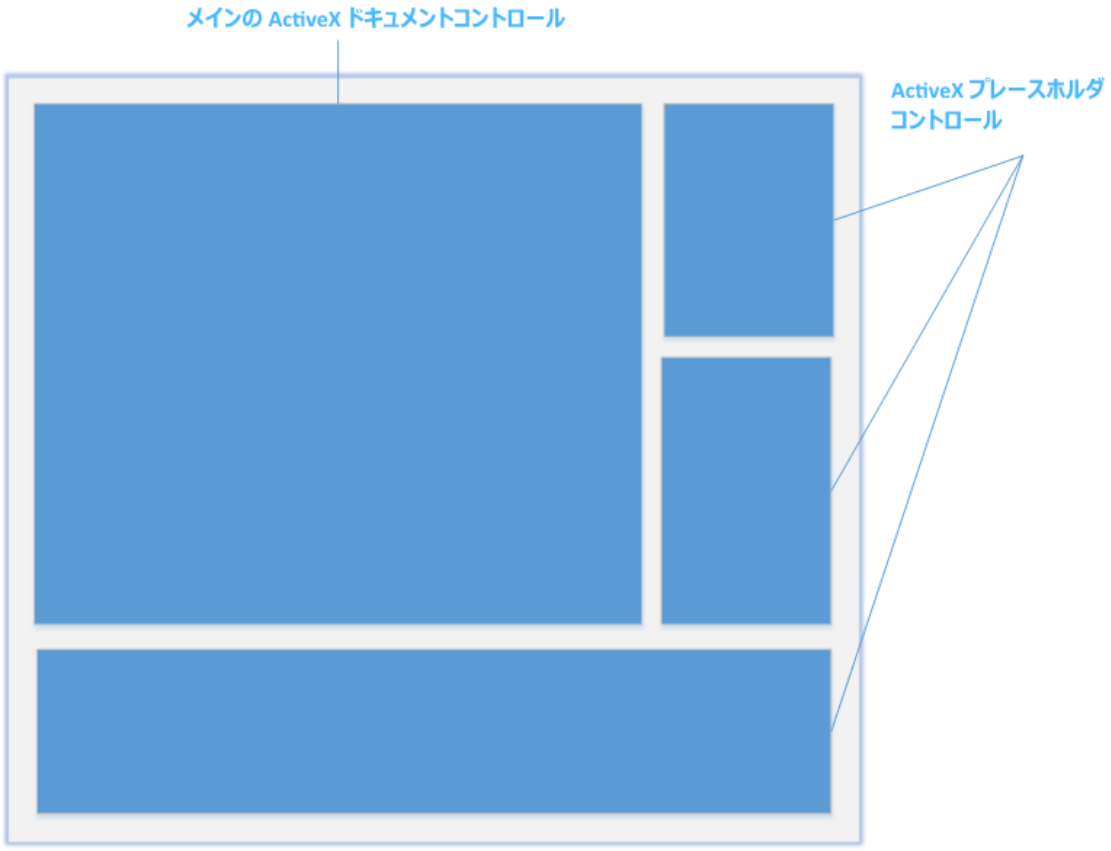
メモ 各ウィンドウ識別子のために「**MapForceControl プレースホルダ**」のみを使用してください。

(C) 2015-2021 Altova GmbH

MapForce プロジェクトウィンドウを選択するプレースホルダーコントロールのために、追加メソッドを使用することができます。「OpenProject」を使用して、MapForce プロジェクトをロードすることができます。MapForce 自動化インターフェイスからプロパティプロジェクトとメソッドを使用して、他のプロジェクトに関連したオペレーションを操作します。

例えば、Visual Studio を使用した C# または VB.NET 内では、MapForce ActiveX コントロールをドキュメントレベルで統合する基本的なアプリケーションの作成ステップは、以下にリストされるものに類似しています。必要に応じて適用は複雑になる可能性があります。しかしながら、ActiveX ドキュメントレベルの統合のための必要条件として、下の命令を理解することは重要です。

1. 新規の空のフォームと共に新規の Visual Studio W indows Forms プロジェクトを作成します。
2. まだ行っていない場合、ActiveX コントロールをツールボックスに追加します（次を参照してください: ActiveX コントロールをツールボックスに追加する）。
3. ツールボックスから **MapForceControl** をドラッグして新規のフォームにドラッグします。
4. **MapForceControl** の「IntegrationLevel」プロパティを「ICActiveXIntegrationOnDocumentLevel」に、と「Visible」プロパティを「False」に設定します。コードから、または、「プロパティ」ウィンドウから行うことができます。
5. MapForceControl ドキュメント をツールボックスからフォームにドラッグします。 このコントロールは、使用中のアプリケーションに MapForce のメインウィンドウを与え、ドキュメントのためにサイズを調整することができます。
6. 代わりに、フォームに 1 つまたは複数の MapForceControl プレースホルダ コントロールを追加することができます（アプリケーションが必要とする各追加ウィンドウの型など。例えば、「プロジェクト」ウィンドウなど）。このような追加プレースホルダーコントロールをメインドキュメントコントロールの下、または、左右に通常配置します。例:



7. 各「MapForceControl プレースホルダ」コントロールの「PlaceholderW indowID」プロパティを有効な ウィンドウ識別子に設定してください。有効な値のリストに関しては、次を参照してください: MapForceControlPlaceholderWindow。
8. アプリケーションにコマンドを以下のように追加します（少なくとも、ドキュメントを開き、保存し、閉じる必要があります）。

MapForce コマンドをクエリする

ドキュメントレベルで統合する場合は、MapForce メニューまたはツールバーをアプリケーションで使用することはできません。代わりに、必要とするコマンドを取得し、状態を確認し、プログラム的に実行することができます:

- 使用することのできるすべてのコマンドを取得するには、**MapForceControl** の CommandsList プロパティを使用します。
- メニュー構造により整理されているコマンドを取得するには、メイン Menu プロパティを使用します。
- 表示される順番に整理されているコマンドを取得するには、Toolbars プロパティを使用します。
- MapForce にコマンドを送信するには、Exec メソッドを使用します。
- コマンドが現在有効化されているか、または、無効化されているかクエリするために、QueryStatus メソッドを使用します。

これにより MapForce コマンドをアプリケーションのメニューとツールバーに柔軟的に統合することができます。

MapForce のインストールは、MapForce 内で使用されているコマンドラベルを与えます。GIF 書式内のアイコンのために MapForce インストールのフォルダー「<ApplicationFolder>¥Examples¥ActiveX¥Images」を参照してください。fコマンド名に対応するファイル名は、コマンド参照 セクション内にリストされています。

一般的な注意点

MapForce の振る舞いを自動化するには、**MapForceControl**、MapForceControl ドキュメント と MapForceControl プレースホルダ内で説明されている プロパティ、メソッド、とイベントを使用してください。

MapForce 機能への高度なアクセスに関しては、次の プロパティを考慮してください:

- MapForceControl.Application
- MapForceControlDocument.Document
- MapForceControlPlaceHolder.Project

これらのプロパティは MapForce 自動化インターフェイス(MapForceAPI)へのアクセスを提供します

メモ ドキュメントを開くには、適切なドキュメントコントロールの MapForceControlDocument.Open または MapForceControlDocument.New を使用してください。 プロジェクトを開くには、MapForce プロジェクトウィンドウを埋め込むプレースホルダ コントロール上の MapForceControlPlaceHolder.OpenProject を常に使用してください。

異なるプログラミング環境内で必要なコントロールをインスタント化しアクセスする方法を説明するサンプルに関しては、次を参照してください: ActiveX 統合のサンプル。

(C) 2015-2021 Altova GmbH

18.5 ActiveX 統合のサンプル

このセクションは、異なるコンテナー環境とプログラム言語を使用するMapForce ドキュメントレベルの統合のサンプル、を含みます（HTML セクションには、追加でアプリケーションレベルの統合のサンプルが含まれます）。全てのサンプルのソースコードは、MapForce インストール内のフォルダー <ApplicationFolder>\Examples\ActiveX で使用することができます。

18.5.1 C#

C# and Visual Studio のための基本のActiveX 統合サンプルソリューションは、フォルダー <ApplicationFolder>\Examples\ActiveX\C# 内にあります。ソースコードをコンパイルする前に、サンプルを実行して、全ての必要条件が満たされていることを確認してください（次を参照してください: サンプルC# ソリューションの実行）。

18.5.1.1 サンプル C# ソリューションの実行

フォルダー**<ApplicationFolder>\Examples\ActiveX\C#** 内で使用することのできるサンプルVisual Studio ソリューションは、MapForce ActiveX コントロールの使用方法について説明しています。このソリューションをビルドして実行する前に、次のステップに注意してください：

ステップ 1: 必要条件をチェックする

Visual Studio 2010 または以降は、サンプルソリューションを開く必要があります。必要条件の完全なリストは、次を参照してください: 必要条件。

ステップ 2: 書き込みのパーミッションを保有するディレクトリにサンプルをコピーする

Visual Studio を管理者として作動することを回避するには、デフォルトの場所から実行する代わりに、書き込みのパーミッションを持つディレクトリへのソースコードをコピーします。

ステップ 3: すべての必要とされるコントロールプロパティをチェックし設定する

サンプルアプリケーションは、MapForceControlDocument の1つのインスタンスとMapForceControlPlaceHolder コントロールの複数のインスタンスを含んでいます。下のテーブルに表示されているとおりにこれらのコントロールの次のプロパティが設定されていることを確認してください：

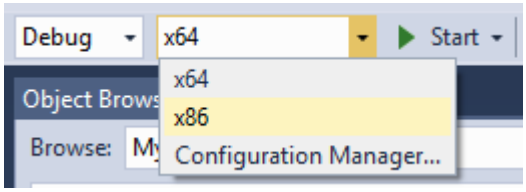
コントロール名	プロパティ	プロパティ値
axMapForceControl	IntegrationLevel	ICActiveXIntegrationOnDocumentLevel
axMapForceControlLibrary	PlaceholderWindowID	0

コントロール名	プロパティ	プロパティ値
axMapForceControlOutput	PlaceholderWindowID	2
axMapForceControlPreview	PlaceholderWindowID	1

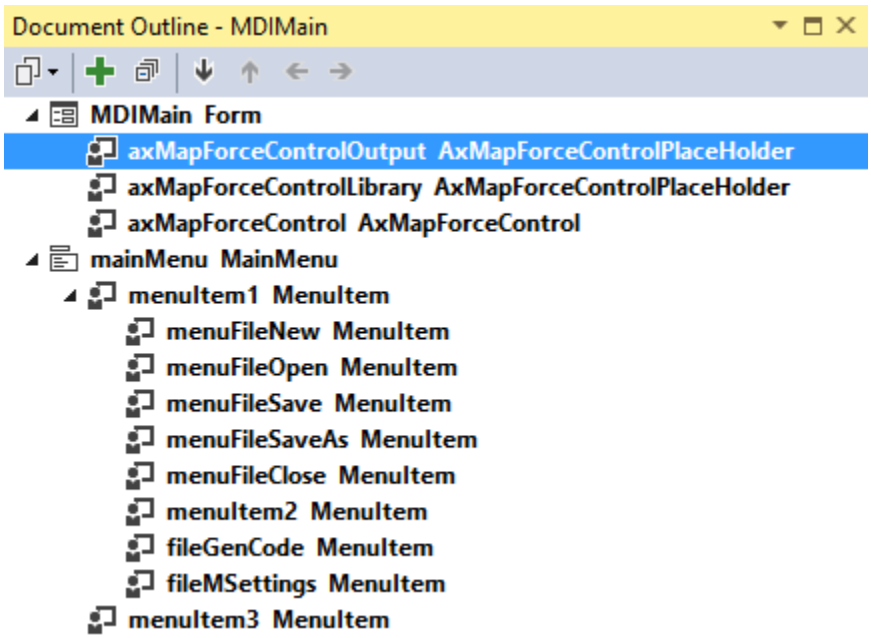
ActiveX コントロールのプロパティをビュー、または、設定する方法は、以下のとおりです:

1. デザイナーウィンドウ内で **MDIMain.cs** フォームを開きます。

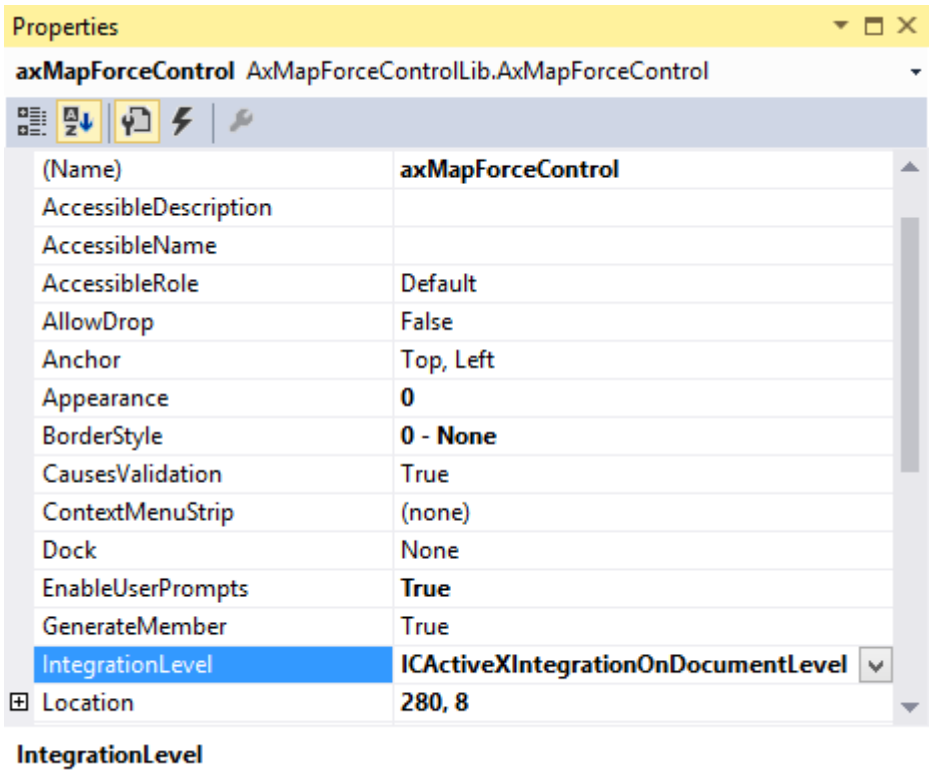
メモ 64 ビット Windows では、デザイナーウィンドウで開く前に Visual Studio ソリューションを「x86」にビルドを更新する必要がある可能性があります。64 ビットアプリケーションとしてサンプルをビルドする場合は、次を参照してください: 必要条件.



2. Visual Studio の**ドキュメント概要**ウィンドウを開きます(「**表示**」メニューから、「**他のウィンドウ | ドキュメントの概要**」をクリックします)。



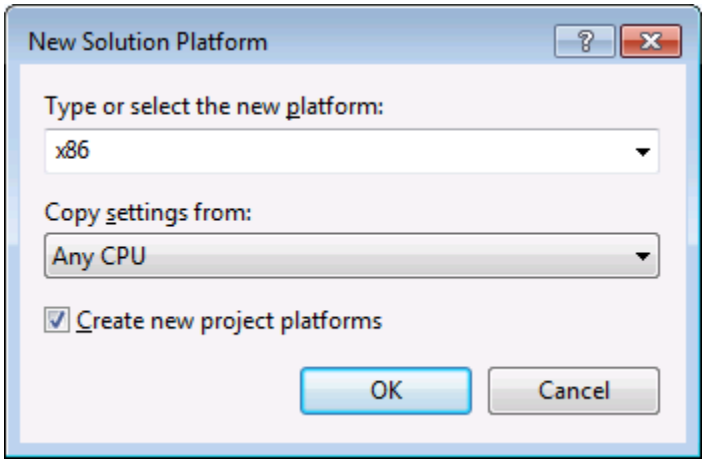
3. **ドキュメント概要**ウィンドウ内の ActiveX コントロールをクリックし、**プロパティ**ウィンドウ内の必要とされるプロパティを編集します。例:



ステップ 4: ビルドプラットフォームを設定する

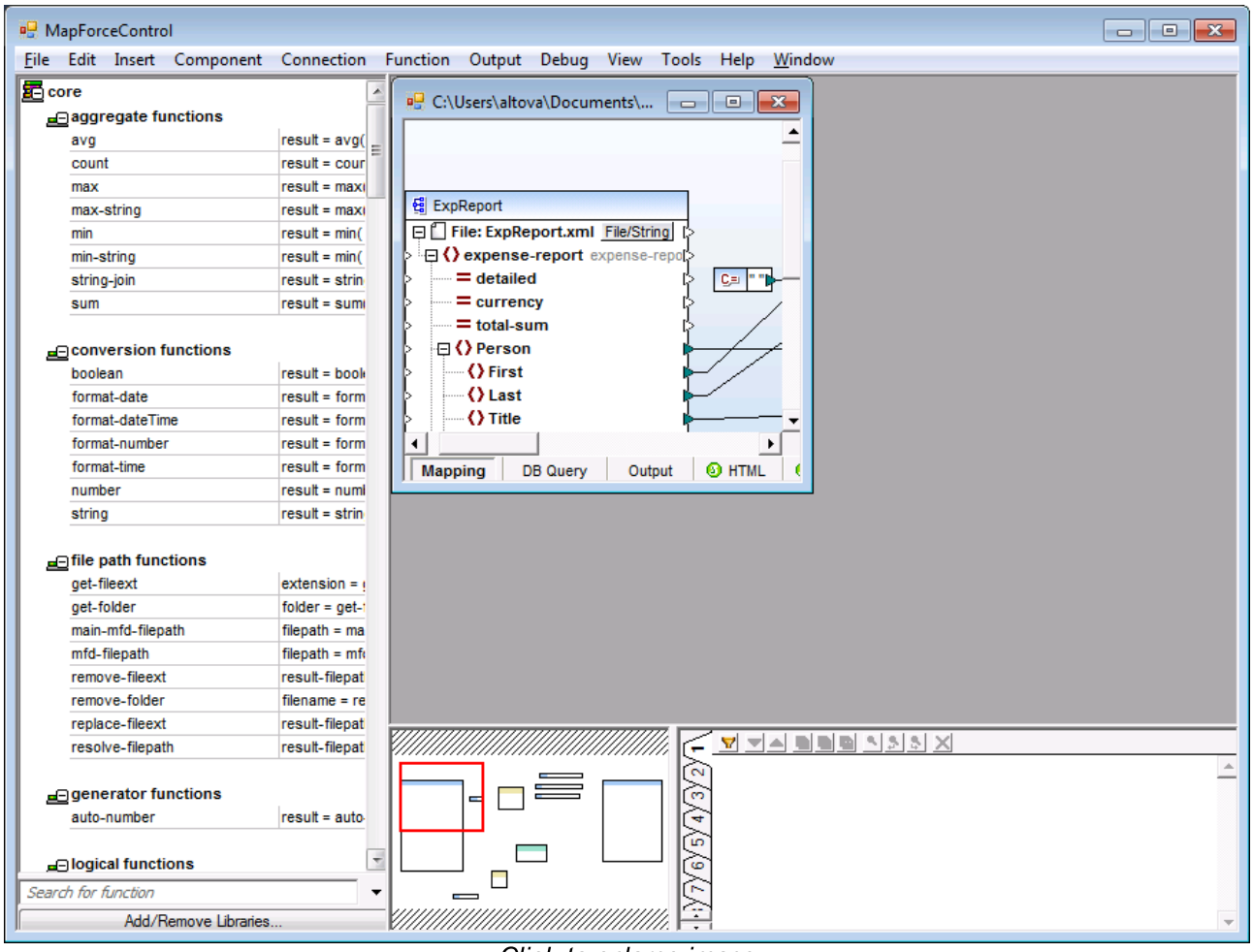
- ビルド(x86、x64)するプラットフォームに一致するビルドプラットフォームの更新を作成します。ビルドの更新の作成の方法は以下のとおりです:

  a. Visual Studio 内のソリューションを右クリックし、「更新マネージャー」を選択します。
  b. 「アクティブなソリューションプラットフォーム」から、「新規作成...」を選択し、x86 または x64 更新を選択します(このサンプルでは、**x86**)。



Visual Studio 内でソリューションをビルドして実行する準備が整うと、ターゲットプラットフォーム（x86、x64）に一致するように更新をしようしてビルドすることを忘れないでください。それ以外の場合、ランタイムエラーが起こる可能性があります。

作動中のサンプルでは、メイン MDI フレームウィンドウが表示されています。「ファイル | 開く」を使用してマッピングファイル（例えば、MapForce サンプルフォルダーである `MarketingExpenses.mfd`）を開きます。ファイルは、ロードされ新規ドキュメント 子ウィンドウ内で表示されます:



Click to enlarge image

ドキュメントをロードした後、メニューを使用してアクティブなドキュメントに対してコマンドを実行することができます。コンテキストメニューも使用し、追加ドキュメントをロードすることもできます。「ファイル | 保存」コマンドを使用して変更を保存できます。

18.5.1.2 コマンド情報の取得

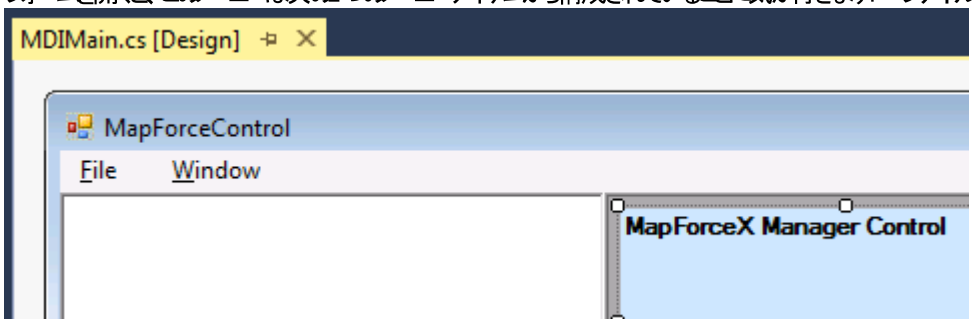
MapForceControl は、`CommandsList`、`MainMenu`、および `Toolbars` プロパティにより MapForce の全てのコマンドへのアクセスを与えます。フォルダー **<ApplicationFolder>\Examples\ActiveX\C#** 内にあるサンプルプロジェクトは、`MainMenu` プロパティを使用して MapForce メニュー構造を動的に作成します。

メニューコマンドを取得するコードは、`MDIMain.cs` ファイル内の `MDIMain` メソッドでみつけることができます:

```
public MDIMain()
```

```
{
    // ...

    // Get the MainMenu property of the control and create the menu structure from it.
    MFLib.MapForceCommand objCommand = this.axMapForceControl.MainMenu;
    InsertMenuStructure(mainMenu, objCommand);
}
```

コードリスト内で、mainMenu はMDI フレームウィンドウの既存の静的 メニューです。 Visual Studio Designer 内で**MDIMain.cs** フォームを開くと、このメニューは次の2 つのメニューアイテムから構成されていることに気が付きます: 「ファイル」とウィンドウ。



MDIMain.cs

メソッド InsertMenuStructure はmainMenu とobjCommand オブジェクトをパラメーターとして取ります。(前者は既存の静的なメニューを含み、後者は、MapForce ActiveX コントロールから取得されたフルメニュー構造を含みます) 取得されたMapForce メニュー構造は既存の静的 メニューにマージされます。メニュー「ファイル」、「プロジェクト」、および「ウィンドウ」は動的に追加されません。これらのメニューは開かれているドキュメントとアクティブに作業するため、また、このサンプルの範囲外のコードを必要とするため、これは意図的です。基本のファイル管理 コマンド (create、open、save、bring into focus) 既存の静的メニュー「ファイル」とウィンドウにより扱われます。すべての他のメニューのActiveX コントロールのMainMenu プロパティから取得された情報をベースに動的に挿入されます。新規のメニューは、「ファイル」の後、「ウィンドウ」の前に挿入されます。例えば、メニューインデックス1 から始まるなど。

メソッド InsertMenuStructure は、MapForceCommand オブジェクト内で見つけられたすべてのトップレベルメニュを介して反復し、新規のメニューをそれぞれに追加します。各トップレベルメニューは自身の子メニューアイテムを所有するため、メソッド InsertMenuCommand への呼び出しは、発生した子メニューアイテムのために実行されます。更に、各子メニューアイテムは自身の子ニューアイテムを持つことができるため、 InsertMenuCommand メソッドは子メニューアイテムが存在しなくなるまで再帰します。

動的に追加されるコマンドは、CustomMenuItem.cs 内で定義されるクラスCustomMenuItem のインスタンスです。このクラスは、System.Windows.Forms.MenuItem クラスから生成され、MapForce コマンド ID に保存するために追加メンバーが存在します。

```
public class CustomMenuItem : System.Windows.Forms.MenuItem
{
      public int m_MapForceCmdID;
}
```

(他のコマンドのためのコンテナー以外の)すべての動的に追加されたコマンド は、コマンドの処理をおこなう同じ イベントハンドラー AltovaMenuItem_Click を取得します:

```
private void AltovaMenuItem_Click(object sender, EventArgs e)
{
  if(sender.GetType() == System.Type.GetType("MapForceapplication.CustomMenuItem"))
  {
    CustomMenuItem    customItem = (CustomMenuItem)sender;
    ProcessCommand(customItem.m_MapForceCmdID);
```

```
    }
}
```

コマンドが(子コマンドを持つ場合)他のコマンド のためのコンテナーである場合、イベント ハンドラー AltovaSubMenu_Popup を取得します。このハンドラーは、各子コマンドの状態をクエリし、必要に応じて有効化、または、無効化することができます。

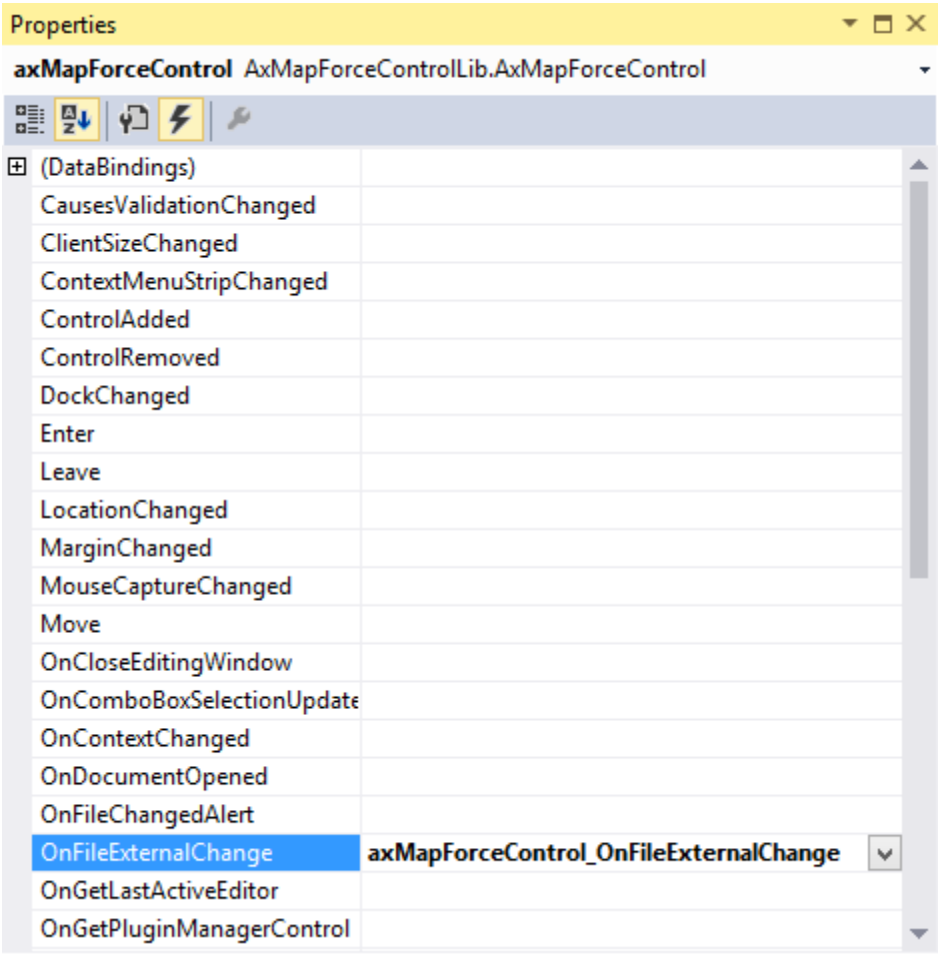
これにより各コマンドが意味のある時にのみ有効化されます。(例えば、「ファイル | 保存」メニューアイテムはアクティブなドキュメントが開かれていない場合無効化されている必要があります)。

メソッド ProcessCommand は、MapForceControl 自身または MapForceControlDocument コントロール内にロードされているアクティブな MapForce ドキュメントのどちらかに実行を委任します。MapForceControl はホストアプリケーション内で現在アクティブなドキュメントを知る方法がないため必要になります。

```
private void ProcessCommand(int nID)
{
   MapForceDoc docMapForce = GetCurrentMapForceDoc();

   if(docMapForce != null)
      docMapForce.axMapForceControlDoc.Exec(nID);
   else
      axMapForceControl.Exec(nID);
}
```

18.5.1.3 イベントの処理

MapForce ライブラリ内の全てのイベントは、接続ポイントをベースにしており、カスタムイベント ハンドラーを与えるために、C# 委任メカニズムを使用することができます。MapForce ライブラリのプロパティページ上でイベントの完全なリストを見つけることができます。下のイメージは、メイン MapForceControl イメージを表示しています:



サンプルプロジェクトは`OnFileExternalChange`イベントのみをオーバーライドします。C# フレームワークにより C# 委任は作成されます。空のイベントハンドラーをフィルするのみで行うことができます。

例えば、下に示されるハンドラーの実装は、ファイルの再ロードをオフに切り替え、MapForceControl によりロードされたファイルが外部により変更されたことをユーザーに通知するためにメッセージボックスを表示します:

```
private void axMapForceControl_OnFileExternalChange(object sender,
AxMapForceControlLib._DMapForceControlEvents_OnFileExternalChangeEvent e)
{
    MessageBox.Show("Attention: the file " + e.strPath + " has been changed from outside¥nbut reloading is turned
off in the sample application!");

    // This turns off any file reloading:
    e.varRet = false;
}
```

18.5.2 HTML

このセクションのコードリストは、アプリケーションレベルとドキュメントレベルでの MapForceControl の統合方法を示しています。全てのサンプルのためのソースコードは MapForce インストールのフォルダー`<ApplicationFolder>\Examples\ActiveX\HTML`内にあります。サンプルは、32 ビット アプリケーションとして作動する場合、Internet Explorer 内でのみ作動することができます。

メモ 64 ビット モード上で作動する場合、nternet Explorer 10 または以降は、ActiveX コントロールをロードしません。

18.5.2.1 アプリケーションレベルの HTML 統合

このサンプルは、アプリケーションレベルでのHTML ページへのMapForce コントロールの簡単な統合について説明しています。統合に関しては、次のセクションで説明されています:

- HTML コード内でMapForceControl を開始します。
- ボタンがドキュメントをロードするように実装し、コード生成タスクを自動化します。
- アプリケーションイベントのためのアクションを定義する。

このサンプルのためのコードは、MapForce インストール内の次の場所で使用することができます:
<ApplicationFolder>\Examples\ActiveX\HTML\MapForceActiveX_ApplicationLevel.htm.

18.5.2.1.1 コントロールインスタンスを作成する

HTML `Object` タグは、MapForceControl のインスタンスを作成するために使用されます。`Classid` は、MapForceControl です。幅と高さがウィンドウのサイズを指定します。アプリケーションレベルがデフォルトのため、追加のパラメーターは必要ありません。

```
<OBJECT id="objMapForceControl"
        Classid="clsid:A38637E9-5759-4456-A167-F01160CC22C1"
        width="800"
        height="500"
        VIEWASTEXT>
</OBJECT>
```

18.5.2.1.2 デフォルトのドキュメントを開くボタンを追加する

一部のタスクの自動化のシンプルなサンプルとして、ページにボタンを追加します:

```
<input type="button" value="Open Marketing Expenses" onclick="BtnOpenMEFile()">
```

クリックされると、定義済みドキュメントがMapForceControl 内で開かれます。異なるインストール上でサンプルを作動するため、MapForceControl に対して相対的なファイルを検索するためにメソッドを使用します。

```
<SCRIPT ID=Javahandlers LANGUAGE=javascript>
// --------------------------------
// open a pre-defined document
function BtnOpenMEFile()
{

             objMapForceControl.Open("C:\Documents and Settings\username\My
Documents\Altova\XMLSpy2021\Examples/MarketingExpenses.mfd");

}
</SCRIPT>
```

18.5.2.1.3 コード生成のためにボタンを追加する

アクティブなドキュメントのためのコード生成は、メニューを使用して使用することができますが、出力の場所をユーザーに問わずにコードが生成されるボタンを作成します。メソッドは前のセクションで使用されたメソッドに類似しています。

ボタンを追加します:

```
<input type="button" value="Generate XSLT" onclick="BtnGenerate( 0 )">
<input type="button" value="Generate Java" onclick="BtnGenerate( 1 )">
<input type="button" value="Generate C++" onclick="BtnGenerate( 2 )">
<input type="button" value="Generate C#" onclick="BtnGenerate( 3 )">
```

現在定義されている デフォルトの出力フォルダー のサブフォルダー内にコードを生成するスクリプトを与えます。

```
<SCRIPT ID=Javahandlers LANGUAGE=javascript>
// ---------------------------------------------------------------------
// generate code for active document into language-specific sub folders of
// the current default output directory. No user interaction necessary.
function BtnGenerate(languageID)
{
	// get top-level object of automation interface
	var objApp = objMapForceControl.Application;

	// get the active document
	var objDocument = objApp.ActiveDocument;

	// retrieve object to set the generation output path
	var objOptions = objApp.Options;

	if ( objDocument == null )
		alert( "no active document found" );
	else
	{
		if (languageID == 0)
		{
			objOptions.XSLTDefaultOutputDirectory =
objOptions.XSLTDefaultOutputDirectory + "\\XSLTGen";
			objDocument .GenerateXSLT();
		}
		else if (languageID == 1)
		{
			objOptions.CodeDefaultOutputDirectory =
objOptions.CodeDefaultOutputDirectory + "/JavaCode";
			objDocument .GenerateJavaCode();
		}
		else if (languageID == 2)
		{
			objOptions.CodeDefaultOutputDirectory =
objOptions.CodeDefaultOutputDirectory + "/CPPCode";
			objDocument .GenerateCppCode();
		}
		else if (languageID == 3)
		{
			objOptions.CodeDefaultOutputDirectory =
objOptions.CodeDefaultOutputDirectory + "/CSharpCode";
			objDocument .GenerateCHashCode();
		}
	}
}
</SCRIPT>
```

18.5.2.1.4 カスタムイベントへの接続

サンプルは、プリンシパルを表示するためにMapForceControl カスタムイベントのための2つのイベントのコールバックを実装します:

```
<!-- ------------------------------------------------------------ -->
<!--  custom event 'OnDocumentOpened" of MapForceControl object  -->
<SCRIPT LANGUAGE="javascript">
       function objMapForceControl::OnDocumentOpened( objDocument )
       {
       // alert("Document '" + objDocument.Name + "' opened!");
       }
</SCRIPT>

<!-- ------------------------------------------------------------ -->
<!--  custom event 'OnDocumentClosed" of MapForceControl object  -->
<SCRIPT LANGUAGE="javascript">
       function objMapForceControl::OnDocumentClosed( objDocument )
       {
       // alert("Document '" + objDocument.Name + "' closed!");
       }
</SCRIPT>
```

18.5.2.2 ドキュメントレベルの HTML 統合

このサンプルは、ドキュメントレベルで HTML ページにMapForce コントロールを統合する方法について説明しています。次のトピックについて説明されています:

- HTML コード内でMapForceControl ActiveX コントロールオブジェクトをインスタンス化します。
- MapForce ファイルの編集を許可する MapForceControlDocument ActiveX コントロールをインスタンス化します。
- MapForceControl プロジェクトウィンドウのためのMapForceControlPlaceHolder をインスタンス化します。
- MapForce ヘルパーウィンドウをホストするためにMapForceControlPlaceHolder をインスタンス化します。
- 頻繁に使用されるMapForce コマンドのためのシンプルなカスタマーツールバーを作成します。
- MapForce のCOM 自動化インターフェイスを使用するボタンを追加します。
- コマンドボタンを更新するためのイベントハンドラーを使用します。

このサンプルは、MapForce インストールの`<ApplicationFolder>\Examples\ActiveX\HTML\` フォルダー内のファイル `MapForceActiveX_ApplicationLevel.htm` 内で使用することがあります。

18.5.2.2.1 MapForceControl をインスタンス化する

MapForceControl HTML `OBJECT` タグは、MapForceControl のインスタンスを作成するために使用されます。クラスID は MapForceControl です。ユーザーインターフェイスのためのコントロール間ジェージャーとして使用される幅と高さは、0 に設定されています。統合レベルは`OBJECT` タグ内でパラメーターとして指定されています。

```
<OBJECT id="objMapForceXMapForceControl"
        Classid="clsid:A38637E9-5759-4456-A167-F01160CC22C1"
        width="0"
        height="0"
```

```
        VIEWASTEXT>
   <PARAM NAME="IntegrationLevel" VALUE="1">
</OBJECT>
```

18.5.2.2.2 エディターウィンドウの作成

HTML OBJECT タグは、編集ウィンドウを埋め込むために使用されます。追加のカスタムパラメーターは、新規のからのマッピングを使用してコントロールが初期化されるように指定します。

```
<OBJECT id="objDoc1"
        Classid="clsid:DFBB0871-DAFE-4502-BB66-08CEB7DF5255"
        width="600"
        height="500"
        VIEWASTEXT>
    <PARAM NAME="NewDocument">
</OBJECT>
```

18.5.2.2.3 プロジェクトウィンドウの作成

HTML OBJECT タグは、MapForceControlPlaceHolder ウィンドウを作成するために使用されます。最初の追加カスタムパラメーターは、MapForce プロジェクトウィンドウを表示するためのプレースホルダを定義します。2番目のパラメーターは、MapForce インストールで提供されたサンプルプロジェクトの1つをロードします（以下の場所で見つけることができます: <yourusername>/MyDocuments フォルダー）。

```
<OBJECT id="objProjectWindow"
        Classid="clsid:FDEC3B04-05F2-427d-988C-F03A85DE53C2"
        width="200"
        height="200"
        VIEWASTEXT>
    <PARAM name="PlaceholderWindowID" value="3">
          <PARAM name="FileName" value="MapForceExamples/MapForceExamples.mfp">
</OBJECT>
```

18.5.2.2.4 ヘルパーウィンドウのためにプレースホルダを作成する

HTML OBJECT タグは、異なる MapForce ヘルパーウィンドウをホストすることのできる MapForceControlPlaceHolder ActiveX コントロールをインスタンス化するために使用されます。最初から、ヘルパーウィンドウは表示されません。サンプルファイルを参照してください。

```
<OBJECT id="objEHWindow"
        Classid="clsid:FDEC3B04-05F2-427d-988C-F03A85DE53C2"
        width="200"
        height="200"
        VIEWASTEXT>
       <PARAM name="PlaceholderWindowID" value="0">
</OBJECT>
```

3つのボタンにより表示される実際のウィンドウを切り替えることができます。JavaScript 実行ボタンをクリックすることにより、プロパティ PlaceHolderWindowID を定義されている対応する値に設定します。

```
<input type="button" value="Library Window" onclick="BtnHelperWindow(0)">
<input type="button" value="Overview window" onclick="BtnHelperWindow(1)">
<input type="button" value="Validation Window" onclick="BtnHelperWindow(2)">

<SCRIPT ID="Javahandlers" LANGUAGE="javascript">
// ---------------------------------------------------------------------------
// specify which of the windows shall be shown in the placeholder control.
function BtnHelperWindow(i_ePlaceholderWindowID)
{
       objEHWindow.PlaceholderWindowID = i_ePlaceholderWindowID;
}
</SCRIPT>
```

18.5.2.2.5 カスタムツールバーの作成

カスタムツールバーは、MapForce コマンドのイメージを持つボタンにより構成されています。コマンド ID 番号は、表示されるボタン要素内で見つけることができます

```
<button id="btnInsertXML" title="Insert XML Schema/File" onclick="BtnDoCommand(13635)">
          <img src="..\Images\ID_INSERT_XSD.gif" width="16" height="16" />
</button>
<button id="btnInsertDB" title="Insert Database" onclick="BtnDoCommand(13590)">
          <img src="..\Images\ID_INSERT_DATABASE.gif" width="16" height="16" />
</button>
<button id="btnInsertEDI" title="Insert EDI" onclick="BtnDoCommand(13591)">
          <img src="..\Images\ID_INSERT_EDI.gif" width="16" height="16" />
</button>
```

ボタンをクリックすると、対応するコマンド ID がマネジャーコントロールに送信されます。

```
<SCRIPT ID="Javahandlers" LANGUAGE="javascript">
// ---------------------------------------------------------------
// perform any command specified by cmdID.
// command routing includes application, active document and view.
function BtnDoCommand(cmdID)
{
       objMapForceX.Exec( cmdID );
       msgtext.innerText = "Command " + cmdID + " performed.";
}
</SCRIPT>
```

18.5.2.2.6 ボタンを追加する

サンプルでは、自動化コードを表示するボタンを追加します。

```
<p>
    <input type="button" value="New File" onclick="BtnNewFile(objDoc1)">
    <input type="button" value="Save File" onclick="BtnSaveFile(objDoc1)">
    <input type="text" title="Path" id="strPath" width="150">
    <input type="button" value="Open MarketingExpenses" onclick="BtnOpenMEFile(objDoc1)">
</p>
<p>
```


(C) 2015-2021 Altova GmbH

```
    <input type="button" id="GenerateXSLT"   value="Generate XSLT"
onclick="BtnGenerate( objDoc1, 0 )">
    <input type="button" id="GenerateJava"   value="Generate Java"
onclick="BtnGenerate( objDoc1, 1 )">
    <input type="button" id="GenerateCpp"    value="Generate C++"
onclick="BtnGenerate( objDoc1, 2 )">
    <input type="button" id="GenerateCSharp" value="Generate C#"
onclick="BtnGenerate( objDoc1, 3 )">
</p>
```

対応するJavaScript修正は以下のようになります:

```
<SCRIPT ID="Javahandlers" LANGUAGE="javascript">
// ------------------------------------------------------
// open a document in the specified document control window.
function BtnOpenMEFile(objDocCtrl)
{
       // do not use MapForceX.Application.OpenDocument(...) to open a document,
       // since then MapForceControl wouldn't know a control window to show
       // the document in. Instead:

       objDocCtrl.OpenDocument("C:\Documents and Settings\username\My Documents\
              Altova\XMLSpy2021\Examples/MarketingExpenses.mfd");
       objDocCtrl.setActive();

}

// ------------------------------------------------------------------
// open a new empty document in the specified document control window.
function BtnNewFile(objDocCtrl)
{
       objDocCtrl.OpenDocument("");
       objDocCtrl.setActive();
}

// ------------------------------------------------------------------
// Saves the current file in the specified document control window.
function BtnSaveFile(objDocCtrl)
{
       if(objDocCtrl.Path.length > 0)
              objDocCtrl.SaveDocument();
       else
       {
              if(strPath.value.length > 0)
              {
                     objDocCtrl.Path = strPath.value;
                     objDocCtrl.SaveDocument();
              }
              else
              {
                     alert("Please set path for the document first!");
                     strPath.focus();
              }
       }

       objDocCtrl.setActive();
}
</SCRIPT>
```

18.5.2.2.7 ボタンステータスを更新するためのイベントハンドラーを作成する

コマンドの可用性はマウスクリック、または、キーボード操作により異なる場合があります。MapForceControl のカスタムイベント OnUpdateCmdUI は、MapForce コマンドに関連するボタンの有効化された/無効化された状態を更新することができます。コマンドの有効化、無効化に関わらず、メソッド MapForceControl.QueryStatus はクエリのために使用されます。

```
<SCRIPT LANGUAGE="javascript">

	function objMapForceX::OnUpdateCmdUI()
	{
	if ( document.readyState == "complete" )		// 'complete'
	{
		// update status of buttons
		GenerateXSLT.disabled =  ! (objDoc1.QueryStatus(13617) & 0x02);	// not
enabled
		GenerateJava.disabled =  ! (objDoc1.QueryStatus(13587) & 0x02);	// not
enabled
		GenerateCpp.disabled =   ! (objDoc1.QueryStatus(13589) & 0x02);	// not
enabled
		GenerateCSharp.disabled = ! (objDoc1.QueryStatus(13588) & 0x02);	// not
enabled

		btnFuncUserDef.disabled = ! (objDoc1.QueryStatus(13633) & 0x02);
		btnFuncUserDefSel.disabled = ! (objDoc1.QueryStatus(13634) & 0x02);
		btnFuncSettings.disabled = ! (objDoc1.QueryStatus(13632) & 0x02);
		btnInsertInput.disabled = ! (objDoc1.QueryStatus(13491) & 0x02);

		btnGenXSLT.disabled = ! (objDoc1.QueryStatus(13617) & 0x02);
		btnGenXSLT2.disabled = ! (objDoc1.QueryStatus(13618) & 0x02);
		btnGenXQuery.disabled = ! (objDoc1.QueryStatus(13586) & 0x02);
		btnGenCPP.disabled = ! (objDoc1.QueryStatus(13589) & 0x02);
		btnGenCSharp.disabled = ! (objDoc1.QueryStatus(13588) & 0x02);
		btnGenJava.disabled = ! (objDoc1.QueryStatus(13587) & 0x02);
	}
	}

	// set activity status of simulated toolbar
</SCRIPT>
```

18.5.3 Java

MapForce ActiveX コンポーネントに Java コードからアクセスすることができます。Java 統合は下にリストされるライブラリで提供されています。これらのライブラリは、MapForce と MapForce Integration Package の両方をインストールした後、MapForce インストール内のフォルダー <ApplicationFolder>\Examples\JavaAPI で使用することができます(次を参照してください: 必要条件)。

- AltovaAutomation.dll: Altova 自動サーバーのための JNI ラッパー(MapForce の 32 ビット インストールの場合)
- AltovaAutomation_x64.dll: Altova 自動サーバーのための JNI ラッパー(MapForce の 64 ビット インストールの場合)
- AltovaAutomation.jar: Altova 自動サーバーにアクセスします

(C) 2015-2021 Altova GmbH

- `MapForceActiveX.jar`: MapForce ActiveX インターフェイスをラップする Java クラス
- `MapForceActiveX_JavaDoc.zip`: Javadoc インターフェイスのためのヘルプドキュメントを含む Javadoc ファイル

メモ Java ActiveX 統合を使用するには、.dll と .jar ファイルが Java クラス検索パス内に含まれている必要があります。

サンプル Java プロジェクト

サンプル Java プロジェクトは製品のインストールにより提供されます。Java プロジェクト をテストし、使用用途に応じて変更することができます。詳細に関しては、次を参照してください: サンプル Java プロジェクト。

JavaRules に ActiveX コントロール名をマッピングする際のルール

ActiveX コントロールのドキュメントに関しては、次を参照してください: オブジェクトレファレンス。オブジェクトの名前変換は Java の場合、他の言語と比較して、若干異なります。具体的には、ActiveX コントロールと Java ラッパー間のマッピングのためのルールは、以下のとおりです:

- *クラスとクラス名*
  MapForce ActiveX インターフェイスの全てのコンポーネントのために、コンポーネントの名前を持つ Java クラスが存在します。

- *メソッド名*
  Javadoc インターフェイス上のメソッド名は、COM インターフェイス上で使用される名前と同じですが、Java 名前変換に準拠する小文字で開始します。COM プロパティにアクセスするには、プロパティ名をプレフィックスする `get` と `set` から始まる Java メソッド使用することができます。ふプロパティが書き込みをサポートしない場合、setter メソッドを使用することはできません。サンプル: MapForceControl の `IntegrationLevel` プロパティ、Java メソッド `getIntegrationLevel` と `setIntegrationLevel` を使用することができます。

- *列挙*
  ActiveX インターフェイス内で定義される全ての列挙のために、Java 列挙は同じ名前と値を使用して定義されます。

- *イベント とイベント ハンドラー*
  イベントをサポートする自動化インターフェイス内の全てのインターフェイスのために、同じ名前と'イベント' を使用した Javadoc インターフェイスを使用することができます。単一のイベントのオーバーロードを簡素化するために、全てのイベントのためのデフォルトの実装を使用する Java クラスが提供されます。この Java クラスの名前は、イベント インターフェイスの名前と `'DefaultHandler'` です。例:
  `MapForceControl`:アプリケーションにアクセスするための Java クラス
  `MapForceControlEvents`:MapForceControl のためのイベント インターフェイス
  `MapForceControlEventsDefaultHandler`:`MapForceControlEvents` のためのデフォルトのハンドラー

マッピングルールの例外

上記のルールの例外の一部は、以下のとおりです。これらは下にリストされています:

インターフェイス	Java クラス内の変更
`MapForceControlDocument`, method `New`	名前を付けて保存済み。`newDocument`
`MapForceControlDocument`, method `OpenDocument`	削除済み。次を使用してください: `Open` メソッド
`MapForceControlDocument`, method `NewDocument`	削除済み。次を使用してください: `newDocument` メソッド
`MapForceControlDocument`, method `SaveDocument`	削除済み。次を使用してください: `Save` メソッド

このセクション
このセクションは、Java コードからアクセスすることのできる基本のMapForce ActiveX 機能の一部を紹介しています。次のサブセクションに整理されています:

- サンプルJava プロジェクト
- ActiveX コントロールの作成
- コントロール内のデータのロード
- 基本のイベントハンドリング
- メニュー
- UI アップデートイベントハンドリング
- MapForce マッピングテーブルの作成

18.5.3.1 サンプル Java プロジェクト

MapForce インストールパッケージには、アプリケーションフォルダー: `<ApplicationFolder>\Examples\ActiveX\Java\` の ActiveX サンプルフォルダーにあるサンプルJava プロジェクトが含まれています。

Java サンプルは、Java. を使用して作成された共有されるデスクトップアプリケーション内にMapForceControl を統合する方法を示しています。

バッチファイル`BuildAndRun.bat` を使用してコマンドラインからテストすることができ、また、Eclipse 内からサンプルプロジェクトをコンパイルして実行することもできます。これらのプロシージャの使用方法を参照してください。

ファイルリスト
Java サンプルフォルダーには、サンプルプロジェクトを作動するために必要な全てのファイルが含まれています。これらのファイルは下にリストされています:

`.classpath`	Eclipse プロジェクトヘルパーファイル
`.project`	Eclipse プロジェクトファイル
`AltovaAutomation.dll`	Java-COM ブリッジ: DLL パート (32-ビットインストール用)
`AltovaAutomation_x64.dll`	Java-COM ブリッジ: DLL パート (64-ビットインストール用)
`AltovaAutomation.jar`	Java-COM ブリッジ: Java ライブラリパート
`BuildAndRun.bat`	コンパイルするバッチファイル、および、コマンドラインプロンプトからサンプルコードを実行します。Java 仮想マシンがパラメータとして存在する箇所にフォルダーが期待されます。
`MapForceActiveX.jar`	MapForce ActiveX コントロールのJava クラス
`MapForceActiveX_JavaDoc.zip`	Java API のためのヘルプドキュメントを含むJavadoc ファイル
`MapForceContainer.java`	Java サンプルソースコード
`MapForceContainerEventHandler.java`	Java サンプルソースコード

MapForceTable.java	Java サンプルソースコード

サンプルの内容

サンプルは、AWT フレームウィンドウ内で MapForce ドキュメント エディターウィンドウ、MapForce プロジェクト ウィンドウ、MapForce ライブラリ ウィンドウおよび MapForce 検証ウィンドウを表示します。MapForce のために定義されているメインメニューを読み取り、同じ構造を持つ AWT メニューを作成します。このメニューまたはプロジェクトを使用して、ドキュメント エディター内でファイルを開き作業することができます。

サンプルを必要に応じて変更することができます.

コードリスト内で次の特定の機能が説明されています:

- ActiveX コントロールの作成: 自動サーバーとして考えられている MapForce を返します。または、作動中の場合、MapForce をアクティブ化します。
- コントロール内のデータのロード: MapForce にインストールされているサンプルドキュメントの 1 つを検索し開きます。
- 基本のイベント ハンドリング: すべての開かれているドキュメントのビューをテキストビューに変えます。コードも開かれているドキュメント内での反復の方法を示しています。
- メニュー: アクティブなドキュメントを検証し、メッセージボックス内に結果を表示します。コードは出力パラメータの使用方法について説明しています。
- UI アップデート イベント ハンドリング: MapForce イベントの処理方法について説明しています。
- MapForce マッピングテーブルの作成: MapForce マッピングテーブルの作成とモーダルアクティブ化について説明しています。

サンプルフォルダーへのパスの更新

与えられているサンプルを作動する前に、**MapForceContainer.java** ファイルを編集します。具体的には、MapForce サンプルファイルが使用中のオペレーティングシステム上で保存されている実際のフォルダーを参照する次のパスを確認します:

```
// Locate samples installed with the product.
final String strExamplesFolder = System.getenv( "USERPROFILE" ) + "\\My Documents\
\Altova\\MapForce2021\\MapForceExamples\\";
```

コマンドラインからサンプルを実行する

コマンドラインからサンプルを作動します:

1. すべての必要条件が満たされていることを確認してください(次を参照してください: 必要条件).
2. コマンドプロンプト ウィンドウを開き、サンプル Java プロジェクト フォルダーに対して現在のディレクトリを変更し、次を入力します:

   ```
   buildAndRun.bat "<Path-to-the-Java-bin-folder>"
   ```

3. 「Enter」を押します。

MapForceContainer.java 内の Java ソースがコンパイルされ、実行されます。

Eclipse 内でサンプルをコンパイルし実行する

サンプル Java プロジェクトを Eclipse にインポートする

1. すべての必要条件が満たされていることを確認してください(次を参照してください: 必要条件).

2. ファイル メニューから、「インポート」をクリックします。
3. 「ワークスペースで既存のプロジェクトを開く」を選択し、次の場所にある Eclipse プロジェクト ファイルを参照します: <ApplicationFolder>¥Examples¥ActiveX¥Java¥。このフォルダーでは書き込みのアクセスがないため、インポート ダイアログボックス上の「ワークプレースにプロジェクトをコピーする」チェックボックスを選択することが奨励されます。

サンプル アプリケーションを作動するには、Package Explorer 内のプロジェクトを右クリックし、コマンド「実行 | Java アプリケーション」を選択します。

コード内のコメントを使用すると Java API クラスのためのヘルプを使用することができ、とEclipse 内で Javadoc ビューを有効化するには、メニューコマンド「Window | ビューの表示 | JavaDoc」を選択してください。

18.5.3.2 ActiveX コントロールの作成

下のコードリストは、ActiveX コントロールの作成方法を示しています。コンストラクターは、Java ラッパーオブジェクトを作成します。カンバスにより派生したオブジェクトをパネル、または、フレームに追加すると、ラップされた ActiveX の生成がトリガーされます。

```
01   /**
02    * MapForce manager control - always needed
03    */
04   public static MapForceControl         mapForceControl = null;
05
06   /**
07    * MapForceDocument editing control
08    */
09   public static MapForceControlDocument     mapForceDocument = null;
10
11   /**
12    * Tool windows - MapForce place-holder controls
13    */
14   private static MapForceControlPlaceHolder   mapForceProjectToolWindow = null;
15   private static MapForceControlPlaceHolder   mapForceValidationToolWindow = null;
16   private static MapForceControlPlaceHolder   mapForceLibraryToolWindow = null;
17
18   // Create the MapForce ActiveX control; the parameter determines that we want
     // to place document controls and place-holder controls individually.
19   // It gives us full control over the menu, as well.
20      mapForceControl = new MapForceControl(
        ICActiveXIntegrationLevel.ICActiveXIntegrationOnDocumentLevel.getValue(), false );
21
22      mapForceDocument = new MapForceControlDocument();
23      frame.add( mapForceDocument, BorderLayout.CENTER );
24
25
26   // Create a project window and open the sample project in it
27      mapForceProjectToolWindow = new MapForceControlPlaceHolder(
        MapForceControlPlaceholderWindow.MapForceXProjectWindow.getValue(),
        strExamplesFolder + "MapForceExamples.mfp" ) ;
28      mapForceProjectToolWindow.setPreferredSize( new Dimension( 200, 200 ) );
```

18.5.3.3 コントロール内のデータのロード

下のコードリストは、ActiveX コントロール内にデータをロードを表示する方法を表示しています。

```
1    // Locate samples installed with the product.
```

```
2       final String strExamplesFolder = System.getenv( "USERPROFILE" ) +
        "\\My Documents\\Altova\\MapForce2021\\MapForceExamples\\";
3       mapForceProjectToolWindow = new
MapForceControlPlaceHolder( MapForceControlPlaceholderWindow.MapForceXProjectWindow.getValue(),
strExamplesFolder + "MapForceExamples.mfp" ) ;
```

18.5.3.4 基本のイベントハンドリング

下のコードリストは、基本のイベントの処理方法について表示しています。MapForceControl の `open` メソッド、または、メニューまたはプロジェクトツリーを使用してファイルを開こうとすると、`onOpenedOrFocused` イベントは、アタッチ済みのイベントハンドラーに送信されます。このイベントのための基本の処理により、MapForceDocumentControl の `open` メソッドを呼び出すことによりファイルを開きます。

```
01       // Open the Marketing file when button is pressed
02       btnMarkExp.addActionListener( new ActionListener() {
03         public void actionPerformed(ActionEvent e) {
04           try {
05             // Instruct the Document control to open the file - avoid calling the open method of
MapForceControl (see help)
06             mapForceDocument.open( strExamplesFolder + "MarketingExpenses.mfd" );
07             mapForceDocument.requestFocusInWindow();
08           } catch (AutomationException e1) {
09             e1.printStackTrace();
10           }
11         }
12       } );
13       public void onOpenedOrFocused( String i_strFileName, boolean i_bOpenWithThisControl,
boolean i_bFileAlreadyOpened ) throws AutomationException
14   {
15     // Handle the New/Open events coming from the Project tree or from the menus
16     if ( !i_bFileAlreadyOpened )
17     {
18       // This is basically an SDI interface, so open the file in the already existing document
control
19       try {
20         MapForceContainer.mapForceDocument.open( i_strFileName );
21         MapForceContainer.mapForceDocument.requestFocusInWindow();
22       } catch (Exception e) {
23         e.printStackTrace();
24       }
25     }
26   }
```

18.5.3.5 メニュー

下のコードリストは、メニューアイテムの作成方法を表示しています。 各 `MapForceCommand` オブジェクトは、コマンドの ID へのセットされている `ActionCommand` を持つ、対応する `MenuItem` オブジェクト、を取得します。全てのメニューアイテムにより生成されているアクションは、(閉じられたメカニズムの)を再度解釈するなどの)特定の処理を行う同じ関数により処理されます。または、`exec` メソッドを呼び出すことにより `MapForceControl` オブジェクトに実行を委任することができます。メニューの作成中にフィルされる `menuMap` オブジェクトは、後で使用されます(次のセクションを参照してください: UI アップデートイベントハンドリング).

```
01
02       // Load the file menu when the button is pressed
03       btnMenu.addActionListener( new ActionListener() {
04         public void actionPerformed(ActionEvent e) {
```

```
05            try {
06              // Create the menubar that will be attached to the frame
07              MenuBar mb = new MenuBar();
08              // Load the main menu's first item - the File menu
09              MapForceCommand xmlSpyMenu =
mapForceControl.getMainMenu().getSubCommands().getItem( 0 );
10              // Create Java menu items from the Commands objects
11              Menu fileMenu = new Menu();
12              handlerObject.fillMenu( fileMenu, xmlSpyMenu.getSubCommands() );
13              fileMenu.setLabel( xmlSpyMenu.getLabel().replace( "&", "" ) );
14              mb.add( fileMenu );
15              frame.setMenuBar( mb );
16              frame.validate();
17            } catch (AutomationException e1) {
18              e1.printStackTrace();
19            }
20            // Disable the button when the action has been performed
21            ((AbstractButton) e.getSource()).setEnabled( false );
22          }
23        } ) ;
24    /**
25     * Populates a menu with the commands and submenus contained in an MapForceCommands object
26     */
27    public void fillMenu(Menu newMenu, MapForceCommands mapForceMenu) throws AutomationException
28    {
29      // For each command/submenu in the mapForceMenu
30      for ( int i = 0 ; i < mapForceMenu.getCount() ; ++i  )
31      {
32        MapForceCommand mapForceCommand = mapForceMenu.getItem( i );
33        if ( mapForceCommand.getIsSeparator() )
34          newMenu.addSeparator();
35        else
36        {
37          MapForceCommands subCommands = mapForceCommand.getSubCommands();
38          // Is it a command (leaf), or a submenu?
39          if ( subCommands.isNull() || subCommands.getCount() == 0 )
40          {
41            // Command -> add it to the menu, set its ActionCommand to its ID and store it in the
menuMap
42            MenuItem mi = new MenuItem( mapForceCommand.getLabel().replace( "&", "" ) );
43            mi.setActionCommand( "" + mapForceCommand.getID() );
44            mi.addActionListener( this );
45            newMenu.add( mi );
46            menuMap.put( mapForceCommand.getID(), mi );
47          }
48          else
49          {
50            // Submenu -> create submenu and repeat recursively
51            Menu newSubMenu = new Menu();
52            fillMenu( newSubMenu, subCommands );
53            newSubMenu.setLabel( mapForceCommand.getLabel().replace( "&", "" ) );
54            newMenu.add( newSubMenu );
55          }
56        }
57      }
58    }
59    /**
60     * Action handler for the menu items
61     * Called when the user selects a menu item; the item's action command corresponds to the
command table for MapForce
62     */
63    public void actionPerformed( ActionEvent e )
64    {
65      try
66      {
```

```
67        int iCmd = Integer.parseInt( e.getActionCommand() );
68        // Handle explicitly the Close commands
69        switch ( iCmd )
70        {
71          case 57602:       // Close
72          case 34050:       // Close All
73            MapForceContainer.initMapForceDocument();
74            break;
75          default:
76            MapForceContainer.mapForceControl.exec( iCmd );
77            break;
78        }
79      }
80      catch ( Exception ex )
81      {
82        ex.printStackTrace();
83      }
84
85    }
```

18.5.3.6 UI アップデートイベントハンドリング

下のコードリストは、UI-Update イベントハンドラーの作成方法を表示しています。

```
01 /**
02    * Call-back from the MapForceControl.
03    * Called to enable/disable commands
04    */
05   @Override
06   public void onUpdateCmdUI() throws AutomationException
07   {
08     // A command should be enabled if the result of queryStatus contains the Supported (1) and
Enabled (2) flags
09     for ( java.util.Map.Entry<Integer, MenuItem> pair : menuMap.entrySet() )
10       pair.getValue().setEnabled( MapForceContainer.mapForceControl.queryStatus( pair.getKey() )
> 2 );
11   }
12  /**
13    * Call-back from the MapForceControl.
14    * Usually called while enabling/disabling commands due to UI updates
15    */
16   @Override
17   public boolean onIsActiveEditor( String i_strFilePath ) throws AutomationException
18   {
19     try {
20       return
MapForceContainer.mapForceDocument.getDocument().getFullName().equalsIgnoreCase( i_strFilePath );
21     } catch ( Exception e ) {
22       return false;
23     }
24   }
```

18.5.3.7 MapForce マッピングのプロパティのリスト

下のリストは、MapForce 内のマッピングオブジェクトがテーブルとしてどのようにロードされるか、または、モーダルの有効化のために準備されるかについてを表示しています。

```
01 //access MapForce Java-COM bridge
02 import com.altova.automation.MapForce.*;
03 import com.altova.automation.MapForce.Component;
04 import com.altova.automation.MapForce.Enums.ENUMComponentUsageKind;
05
06 //access AWT and Swing components
07 import java.awt.*;
08 import javax.swing.*;
09 import javax.swing.table.*;
10
11
12 /**
13  * A simple example of a table control loading the structure from a Mapping object.
14  * The class receives an Mapping object, loads its components in a JTable, and prepares
15  * for modal activation.
16  *
17  * Feel free to modify and extend this sample.
18  *
19  * @author Altova GmbH
20  */
21 class MapForceTable extends JDialog
22 {
23   /**
24    * The table control
25    */
26   private JTable myTable;
27
28   /**
29    * Constructor that prepares the modal dialog containing the filled table control
30    * @param mapping The data to be displayed in the table
31    * @param parent  Parent frame
32    */
33   public MapForceTable( Mapping mapping, Frame parent )
34   {
35     // Construct the modal dialog
36     super( parent, "MapForce component table", true );
37     // Build up the tree
38     fillTable( mapping );
39     // Arrange controls in the dialog
40     setContentPane( new JScrollPane( myTable ) );
41   }
42
43   /**
44    * Loads the components of a Mapping object in the table
45    * @param mapping Source data
46    */
47   private void fillTable( Mapping mapping)
48   {
49     try
50     {
51       // count how many Instance components do we have
52       int size = 0;
53       for (Component comp : mapping.getComponents())
54         if ( comp.getUsageKind() == ENUMComponentUsageKind.eComponentUsageKind_Instance )
55           ++size;
56
57       // Prepare data
58       final String[] columnNames = { "Component", "Has inputs", "Has outputs", "Input file",
"Output file", "Schema" };
59       final Object[][] data = new Object[size ][ 7 ] ;
60       int index = 0 ;
61       for (Component comp : mapping.getComponents())
62         if ( comp.getUsageKind() == ENUMComponentUsageKind.eComponentUsageKind_Instance )
```

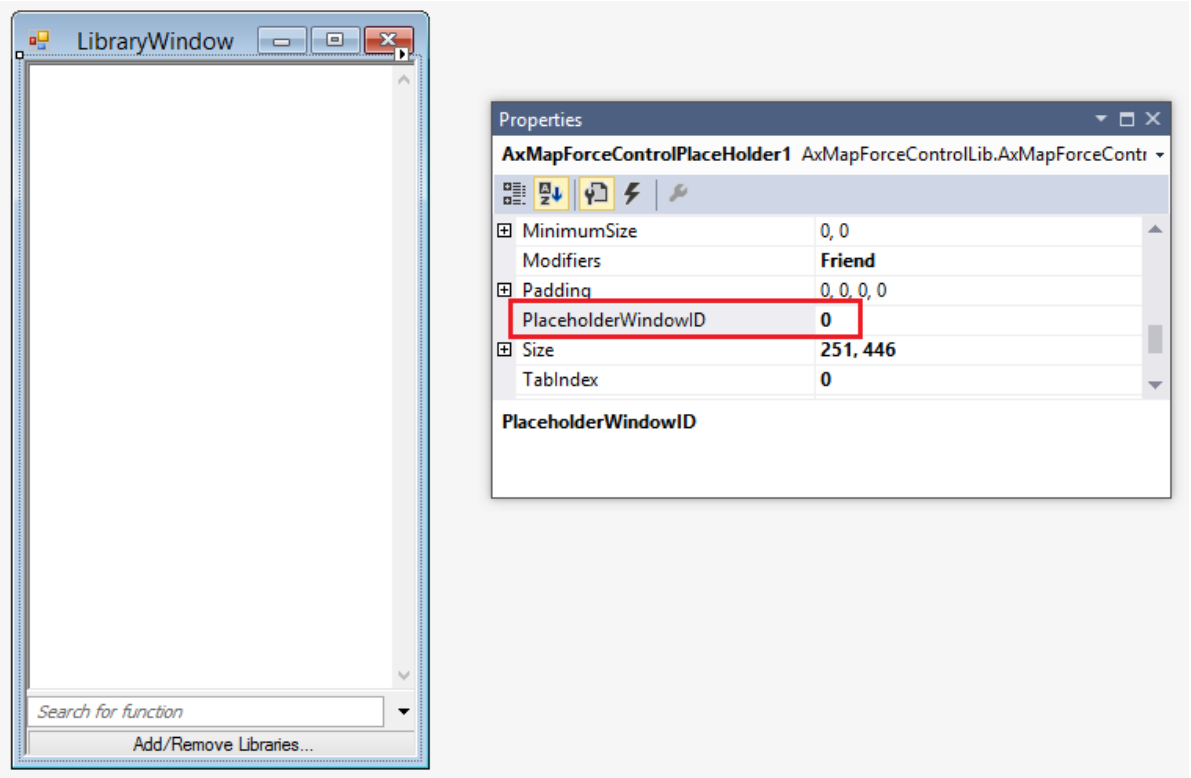

(C) 2015-2021 Altova GmbH

```
63          {
64            int i = 0;
65            data[ index ][ i++ ] = comp.getName() ;
66            data[ index ][ i++ ] = new Boolean( comp.getHasIncomingConnections() );
67            data[ index ][ i++ ] = new Boolean( comp.getHasOutgoingConnections() );
68            data[ index ][ i++ ] = comp.getInputInstanceFile();
69            data[ index ][ i++ ] = comp.getOutputInstanceFile();
70            data[ index++ ][ i ] = comp.getSchema() ;
71          }
72
73        // Set up table
74        myTable = new JTable( new AbstractTableModel() {
75            public String getColumnName(int col) { return columnNames[col]; }
76            public int getRowCount() { return data.length; }
77            public int getColumnCount() { return columnNames.length; }
78            public Object getValueAt(int row, int col) { return data[row][col]; }
79            public boolean isCellEditable(int row, int col) { return false; }
80            public Class getColumnClass(int c) { return getValueAt(0, c).getClass(); }
81        } );
82
83        // Set width
84        for( index = 0 ; index < columnNames.length ; ++index )
85          myTable.getColumnModel().getColumn( index ).setMinWidth( 80 );
86        myTable.getColumnModel().getColumn( 5 ).setMinWidth( 400 );
87      }
88      catch (Exception e)
89      {
90        e.printStackTrace();
91      }
92    }
93
94 }
```

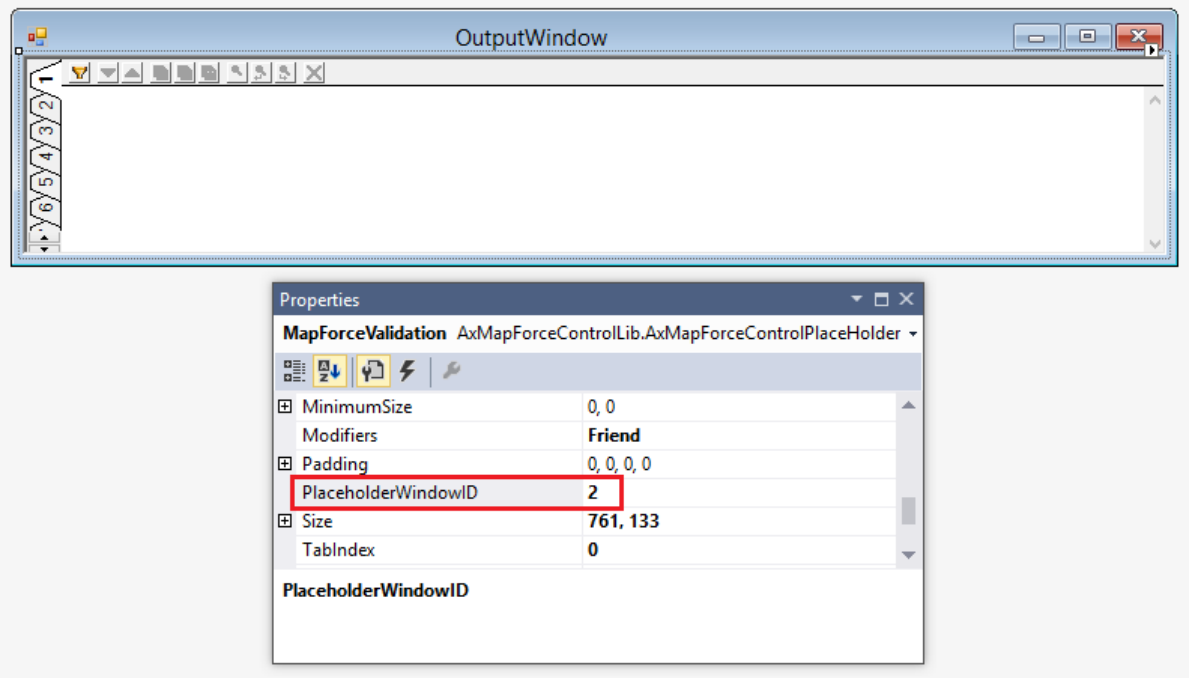
18.5.4 VB.NET

ソースコードは、VB.NET アプリケーションへのMapForceControl の統合を表しています。これはMapForce インストール内のフォルダー<ApplicationFolder>\Examples\ActiveX\VB.NET でみつけることができます。ソリューションは、3つのウィンドウにより以下のように構成されています:

1. **MainWindow.vb** – メインドキュメントウィンドウにも基本のアプリケーションメニューが含まれています。
2. **LibraryWindow.vb** – ライブラリウィンドウ。このウィンドウのコンテンツは、**PlaceholderWindowID** プロパティが0 に設定されているプレースホルダーコントロールにより作成されます(この値はコントロールに対してライブラリウィンドウに表示するように命令します)。



3. **OutputWindow.vb** – メッセージ（Output）ウィンドウ。このウィンドウのコンテンツは、**PlaceholderWindowID** プロパティが2に設定されているプレースホルダーコントロールにより作成されます（この値はコントロールに対して出力ウィンドウに表示するように命令します）。



(C) 2015-2021 Altova GmbH

このソリューションをビルドして実行する前に、以下のステップに注意してください:

ステップ 1: 必要条件をチェックする

必要条件のリストに関しては、次を参照してください: 必要条件.

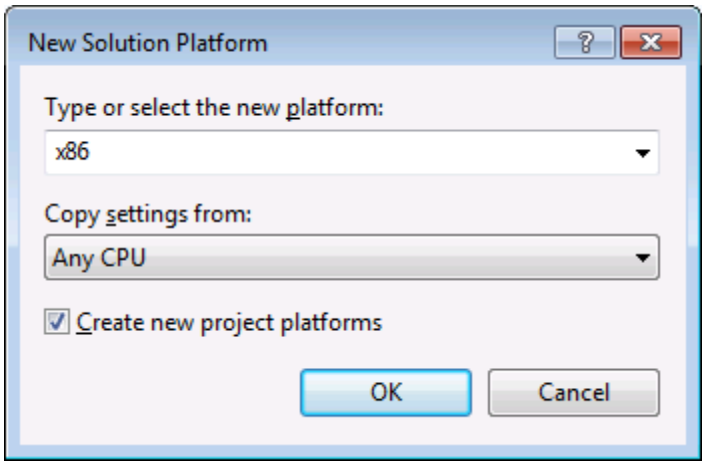
ステップ 2: 書き込みのパーミッションを持つ箇所でサンプルをディレクトリにコピーする

Visual Studio を管理者として作動することを回避するには、デフォルトの場所から実行する代わりに、書き込みのパーミッションを持つディレクトリへのソースコードをコピーします。

ステップ 3: ビルドプラットフォームを設定する

- ビルド(x86、x64)するプラットフォームに一致するビルドプラットフォームの更新を作成します。ビルドの更新の作成の方法は以下のとおりです:

  a. Visual Studio 内のソリューションを右クリックし、「**更新 マネージャー**」を選択します。
  b. 「**アクティブなソリューションプラットフォーム**」から、「**新規作成...**」を選択し、x86 または x64 更新を選択します(このサンプルでは、**x86**)。



Visual Studio 内でソリューションをビルドして実行する準備が整うと、ターゲットプラットフォーム(x86、x64) に一致するように更新をしようしてビルドすることを忘れないでください。それ以外の場合、ランタイムエラーが発生する可能性があります。

18.6 コマンド レファレンス

このセクションは、MapForce 内で使用することのできる全てのメニューコマンドの名前と識別子をリストしています。各サブセクションは MapForce. コマンドテーブルの対応するトップレベルメニューからのコマンドが以下のとおりに整理されています:

- 「メニューアイテム」列は、コマンドのメニューテキストを MapForce で表示される順番に表示します。コマンドの機能を簡単に識別することができます。
- 「コマンド名」列は、MapForce インストールディレクトリの **ActiveX\Images** フォルダーから同じ名前を持つアイコンを取得するために使用される文字列を指定します。
- 「ID」列は、このコマンドを実行またはクエリするメソッドの引数として与えられる列の数値識別子を表示します。

コマンドを実行するには、MapForceControl.Exec または MapForceControlDocument.Exec メソッドを利用します。コマンドのステータスをクエリするには、MapForceControl.QueryStatus または MapForceControlDocument.QueryStatus メソッドを使用します。

インストールされた MapForce のエディションにより、コマンドの一部がサポートされない場合があります。

18.6.1 「ファイル」メニュー

「ファイル」メニューには以下のコマンドが含まれます:

メニューアイテム	コマンド名	ID
新規作成...	ID_FILE_NEW	57600
開く...	ID_FILE_OPEN	57601
保存	ID_FILE_SAVE	57603
名前を付けて保存	ID_FILE_SAVE_AS	57604
全て保存する	ID_FILE_SAVEALL	32377
再ロード	IDC_FILE_RELOAD	32467
閉じる	ID_WINDOW_CLOSE	32453
全て閉じる	ID_WINDOW_CLOSEALL	32454
印刷	ID_FILE_PRINT	57607
印刷プレビュー	ID_FILE_PRINT_PREVIEW	57609
印刷設定	ID_FILE_PRINT_SETUP	57606
マッピングの検証	ID_MAPPING_VALIDATE	32347
マッピングの設定	ID_MAPPING_SETTINGS	32396
選択された言語でコードを生成する	ID_FILE_GENERATE_SELECTED_CODE	32362
XSLT 1.0	ID_FILE_GENERATEXSLT	32360

(C) 2015-2021 Altova GmbH

メニューアイテム	コマンド名	ID
XSLT 2.0	ID_FILE_GENERATEXSLT2	32361
XQuery	ID_FILE_GENERATEXQUERY	32359
Java	ID_FILE_GENERATEJAVACODE	32358
C# (Sharp)	ID_FILE_GENERATECSCODE	32357
C++	ID_FILE_GENERATECPPCODE	32356
MapForce Server 実行ファイルのコンパイル	ID_FILE_CREATE_SERVER_EXECUTION_FILE	32517
FlowForce Server にデプロイする	ID_FILE_DEPLOY_MAPPING	32506
ドキュメントの生成	ID_FILE_GENERATE_DOCUMENTATION	32468
最近使ったファイル	ID_FILE_MRU_FILE1	57616
終了	ID_APP_EXIT	57665

18.6.2 「編集」メニュー

「編集」メニューには以下のコマンドが含まれます:

メニューアイテム	コマンド名	ID
元に戻す	ID_EDIT_UNDO	57643
やり直し	ID_EDIT_REDO	57644
検索	ID_EDIT_FIND	57636
次を検索	ID_EDIT_FINDNEXT	32349
前を検索	ID_EDIT_FINDPREV	32350
切り取り	ID_EDIT_CUT	57635
コピー	ID_EDIT_COPY	57634
貼り付け	ID_EDIT_PASTE	57637
削除	ID_EDIT_CLEAR	57632
全て選択	ID_EDIT_SELECT_ALL	57642

18.6.3 「挿入」メニュー

「挿入」メニューには以下のコマンドが含まれます:

メニューアイテム	コマンド名	ID
XML スキーマ/ファイル	ID_INSERT_XSD	32393
データベース	ID_INSERT_DATABASE	32389
EDI...	ID_INSERT_EDI	32390
テキストファイル..	ID_INSERT_TXT	32392
ウェブサービス関数..	ID_INSERT_WEBSERVICE_FUNCTION	32319
Excel 2007+ ファイル..	ID_INSERT_EXCEL	32376
XBRL ドキュメント...	ID_INSERT_XBRL	32469
JSON スキーマ/ファイル	ID_INSERT_JSON	32531
入力の挿入...	ID_FUNCTION_INSERT_INPUT	32383
出力の挿入...	ID_FUNCTION_INSERT_OUTPUT	32402
定数..	ID_INSERT_CONSTANT	32388
変数..	ID_INSERT_VARIABLE	32500
ジョイン	ID_INSERT_JOIN	32581
並べ替え: Nodes/Rows	ID_INSERT_SORT	32444
フィルター: Nodes/Rows	ID_INSERT_FILTER	32391
SQL-WHERE/ORDER	ID_INSERT_SQLWHERE_CONDITION	32351
Value-Map	ID_INSERT_VALUEMAP	32354
IF-Else 条件	ID_INSRT_CONDITION	32394
例外	ID_INSERT_EXCEPTION	32311

18.6.4 「プロジェクト」メニュー

「プロジェクト」メニューには以下のコマンドが含まれます:

メニューアイテム	コマンド名	ID
プロジェクトを再ロードする	ID_PROJECT_RELOAD	32476

(C) 2015-2021 Altova GmbH

メニューアイテム	コマンド名	ID
プロジェクトを閉じる	ID_FILE_CLOSEPROJECT	32355
プロジェクトを保存する	ID_FILE_SAVEPROJECT	32378
プロジェクトにファイルを追加する	ID_PROJECT_ADDFILESTOPROJECT	32420
アクティブなファイルをプロジェクトに追加する	ID_PROJECT_ADDACTIVEFILETOPROJECT	32419
フォルダーの作成...	ID_PROJECT_CREATE_FOLDER	32310
マッピングを開く	ID_PROJECT_OPEN_MAPPING	32307
オペレーションマッピングの作成...	ID_PROJECT_CREATE_MAPPING_FOR_OPERATION	32399
オペレーションマッピングファイルの追加...	ID_PROJECT_ADD_MAPPING	32309
ウェブサービスを挿入...	ID_PROJECT_INSERT_WEBSERVICE	32306
XMLSpy のファイルを開く	ID_PROJECT_OPEN_IN_XMLSPY	32305
プロジェクト全体のコード生成	ID_PROJECT_GENERATE_ALL	32303
XSLT 1.0	ID_PROJECT_GENERATEXSLTCODE_ENTIRE	32408
XSLT 2.0	ID_PROJECT_GENERATEXSLT2CODE_ENTIRE	32409
XQuery	ID_PROJECT_GENERATEXQUERYCODE_ENTIRE	32410
Java	ID_PROJECT_GENERATEJAVACODE_ENTIRE	32411
C# (Sharp)	ID_PROJECT_GENERATECSCODE_ENTIRE	32412
C++	ID_PROJECT_GENERATECPPCODE_ENTIRE	32413
プロパティ	ID_PROJECT_PROPERTIES	32404
最近使ったプロジェクト	ID_FILE_MRU_PROJECT1	32364

18.6.5 「コンポーネント」メニュー

「コンポーネント」メニューには以下のコマンドが含まれます:

メニューアイテム	コマンド名	ID
ルート要素の変更...	ID_COMPONENT_CHANGEROOTELEMENT	32334

メニューアイテム	コマンド名	ID
XMLSpy でスキーマ定義を編集	ID_COMPONENT_EDIT_SCHEMA	32337
FlexText 構成の編集	ID_COMPONENT_EDIT_MFT	32301
データベースオブジェクトの追加/削除/編集...	ID_COMPONENT_SELECTTABLES	32346
EDI X12 997 へのマッピングを作成	ID_COMPONENT_CREATE_MAPPING_TO_997	32483
EDI X12 999 へのマッピングを作成	ID_COMPONENT_CREATE_MAPPING_TO_999	32484
更新	IDC_COMMAND_REFRESH_COMPONENT	32373
前に入力を複製する	ID_COMPONENT_CREATE_DUPLICATE_ICON_BEFORE	32503
後に入力を複製する	ID_COMPONENT_CREATE_DUPLICATE_ICON	32335
複製を削除	ID_COMPONENT_REMOVE_DUPLICATE_ICON	32339
前にコメントを追加する	ID_COMPONENT_ADD_COMMENT_BEFORE	32518
後にコメントを追加する	ID_COMPONENT_ADD_COMMENT_AFTER	32519
前に処理命令を追加する...	ID_COMPONENT_ADD_PI_BEFORE	32520
後に処理命令を追加する...	ID_COMPONENT_ADD_PI_AFTER	32521
処理命令を編集する...	ID_COMPONENT_EDIT_PI	32524
コメント/処理命令を削除する	ID_COMPONENT_REMOVE_COMMENT_PI	32522
CDATA セクションとしてコンテンツを書き込む	ID_COMPONENT_TOGGLE_CDATA	32525
データベーステーブルアクション	ID_POPUP_DATABASETABLEACTIONS	32400
Query データベース	ID_QUERY_DATABASE	32341
ツリーを左揃えにする	ID_COMPONENT_LEFTALIGNTREE	32338
ツリーを右揃えにする	ID_COMPONENT_RIGHTALIGNTREE	32340
プロパティ	ID_COMPONENT_PROPERTIES	32336

18.6.6 「接続」メニュー

「接続」メニューには以下のコマンドが含まれます:

(C) 2015-2021 Altova GmbH

メニューアイテム	コマンド名	ID
マッチした子に自動的に接続	ID_CONNECTION_AUTOCONNECTCHILDREN	32342
マッチした子に接続のための設定	ID_CONNECTION_SETTINGS	32344
マッチした子に接続...	ID_CONNECTION_MAPCHILDELEMENTS	32343
ターゲット優先(標準マッピング)	ID_POPUP_NORMALCONNECTION	32401
全てコピー(子アイテムをコピー)	ID_POPUP_NORMALWITHCHILDREN_CONNECTION	32460
ソース優先マッピング(複合型コンテンツ)	ID_POPUP_ORDERBYSOURCECONNECTION	32403
プロパティ	ID_POPUP_CONNECTION_SETTINGS	32398

18.6.7 「関数」メニュー

「関数」メニューには以下のコマンドが含まれます:

メニューアイテム	コマンド名	ID
ユーザー定義関数の作成...	ID_FUNCTION_CREATE_EMPTY	32380
選択からユーザー定義関数の作成...	ID_FUNCTION_CREATE_FROM_SELECTION	32381
関数設定	ID_FUNCTION_SETTINGS	32387
関数削除	ID_FUNCTION_REMOVE	32385
入力の挿入...	ID_FUNCTION_INSERT_INPUT	32383
出力の挿入...	ID_FUNCTION_INSERT_OUTPUT	32402

18.6.8 「出力」メニュー

「出力」メニューには以下のコマンドが含まれます:

メニューアイテム	コマンド名	ID
XSLT 1.0	ID_SELECT_LANGUAGE_XSLT	32433
XSLT 2.0	ID_SELECT_LANGUAGE_XSLT2	32434
XQuery	ID_SELECT_LANGUAGE_XQUERY	32432
Java	ID_SELECT_LANGUAGE_JAVA	32431
C# (Sharp)	ID_SELECT_LANGUAGE_CSHARP	32430

メニューアイテム	コマンド名	ID
C++	ID_SELECT_LANGUAGE_CPP	32429
ビルトイン実行エンジン	ID_SELECT_LANGUAGE_BUILTIN	32490
出力ファイルの検証	ID_XML_VALIDATE	32458
出力ファイルの保存...	IDC_FILE_SAVEGENERATEDOUTPUT	32321
全ての出力ファイルの保存...	IDC_FILE_SAVEALLGENERATEDOUTPUT	32374
出力を再生成	ID_REGENERATE_PREVIEW_OUTPUT	32480
SQL スクリプトを実行	ID_TRANSFORM_RUN_SQL	32442
ブックマークの挿入/削除	ID_TOGGLE_BOOKMARK	32317
次のブックマーク	ID_GOTONEXTBOOKMARK	32315
前のブックマーク	ID_GOTOPREVBOOKMARK	32314
全てのブックマークの削除	ID_REMOVEALLBOOKMARKS	32313
XML テキストの整形出力	ID_PRETTY_PRINT_OUTPUT	32363
テキストビューの設定	ID_TEXTVIEWSETTINGSDIALOG	32472

18.6.9 「デバッグ」メニュー

「デバッグ」メニューには以下のコマンドが含まれます:

メニューアイテム	コマンド名	ID
デバッグの開始	ID_DEBUG_START	32540
デバッグの停止	ID_DEBUG_STOP	32541
ステップイン	ID_DEBUG_STEP_INTO	32545
ステップオーバー	ID_DEBUG_STEP_OVER	32551
ステップアウト	ID_DEBUG_STEP_OUT	32552
最小ステップ	ID_DEBUG_STEP_NEXT_TRACE	32554

18.6.10 「表示」メニュー

「表示」メニューには以下のコマンドが含まれます:

(C) 2015-2021 Altova GmbH

メニューアイテム	コマンド名	ID
注釈の表示	ID_SHOW_ANNOTATION	32435
型の表示	ID_SHOW_TYPES	32437
関数のライブラリ名を表示	ID_VIEW_SHOWLIBRARYINFUNCTIONHEADER	32448
ヒントの表示	ID_SHOW_TIPS	32436
XBRL 表示オプション	ID_VIEW_XBRL_DISPLAY_OPTIONS	32473
選択コンポーネントの接続線の表示	ID_VIEW_AUTOHIGHLIGHTCOMPONENTCONNECTIONS	32443
ソースからターゲットへの接続線の表示	ID_VIEW_RECURSIVEAUTOHIGHLIGHT	32447
ズーム..	ID_VIEW_ZOOM	32451
戻る	ID_CMD_BACK	32479
進む	ID_CMD_FORWARD	32478
ステータスバー	ID_VIEW_STATUS_BAR	59393
ライブラリウィンドウ	ID_VIEW_LIBRARY_WINDOW	32445
メッセージ	ID_VIEW_VALIDATION_OUTPUT	32450
概要	ID_VIEW_OVERVIEW_WINDOW	32446
プロジェクトウィンドウ	ID_VIEW_PROJECT_WINDOW	32302
値	ID_DEBUG_VIEW_VALUES_WINDOW	32544
コンテキスト	ID_DEBUG_VIEW_CONTEXT_WINDOW	32546
ブレークポイント	ID_DEBUG_VIEW_DEBUGPOINTS_WINDOW	32547

18.6.11 「ツール」メニュー

「ツール」メニューには以下のコマンドが含まれています:

メニューアイテム	コマンド名	ID
グローバルリソース	IDC_GLOBALRESOURCES	37401
	IDC_GLOBALRESOURCES_SUBMENUENTRY1	37408
反転マッピングの作成	ID_CREATE_REVERSED_MAPPING	32489
カスタマイズ..	IDC_APP_TOOLS_CUSTOMIZE	32959

メニューアイテム	コマンド名	ID
オプション...	ID_TOOLS_OPTIONS	32441

18.6.12 「ヘルプ」メニュー

「ウィンドウ」メニューには以下のコマンドが含まれます:

メニューアイテム	コマンド名	ID
重ねて表示	ID_WINDOW_CASCADE	57650
上下に並べて表示	ID_WINDOW_TILE_HORZ	57651
左右に並べて表示	ID_WINDOW_TILE_VERT	57652

18.6.13 「ヘルプ」メニュー

「ヘルプ」メニューには以下のコマンドが含まれます:

メニューアイテム	コマンド名	ID
目次	IDC_HELP_CONTENTS	32966
インデックス	IDC_HELP_INDEX	32967
検索	IDC_HELP_SEARCH	32969
ライセンスの登録	IDC_ACTIVATION	32970
注文フォーム	IDC_OPEN_ORDER_PAGE	32971
登録	IDC_REGISTRATION	32972
最新情報のチェック	IDC_CHECK_FOR_UPDATES	32973
MapForce 製品比較	IDC_PRODUCT_COMPARISON	32955
サポートセンター	IDC_OPEN_SUPPORT_PAGE	32961
W eb のFAQ	IDC_SHOW_FAQ	32962
コンポーネントや無料ツールのダウンロード	IDC_OPEN_COMPONENTS_PAGE	32963
インターネット上のMapForce	IDC_OPEN_XML_SPY_HOME	32964
MapForce トレーニング	IDC_OPEN_MAPFORCE_TRAINING_PAGE	32965
MapForce について	ID_APP_ABOUT	57664

(C) 2015-2021 Altova GmbH

18.7 オブジェクトレファレンス

オブジェクト:
MapForceCommand
MapForceCommands
MapForceControl
MapForceControlDocument
MapForceControlPlaceHolder

標準のMapForce 機能へのアクセスを与えるには、**MapForce 自動化インターフェイス** のオブジェクトにもアクセスすることができます。詳細に関しては、次を参照してください: MapForceControl.Application, MapForceControlDocument.Document と MapForceControlPlaceHolder.Project 。

18.7.1 MapForceCommand

プロパティ:
ID
Label
Name
IsSeparator
ToolTip
StatusText
Accelerator
SubCommands

詳細:

コマンドオブジェクトは次のオプションであることができます: 実行可能なコマンド, コマンド コンテナー(例えば、メニュー、サブメニュー、または、ツールバー)、または、メニューセパレーター。現在の Command オブジェクト内に保管される情報の種類を決定するには、ID、IsSeparator、と SubCommands プロパティをクエリしてください。

コマンドオブジェクト	使用可能な場合...
実行可能なコマンド	• ID はゼロよりも大きい場合 • IsSeparator がfalse の場合 • SubCommands が空白の場合
コマンド コンテナー	• ID はゼロの場合 • IsSeparator がtrue の場合 • SubCommands が Command オブジェクトを含む場合
セパレーター	• ID はゼロの場合 • IsSeparator がtrue の場合

18.7.1.1 Accelerator

プロパティ: Accelerator を string として

詳細:

コマンドのために定義されたアクセラレータキーを返します。コマンドにアクセラレーターキーが割り当てられていない場合、このプロパティは、空の文字列を返します。 アクセラレータキーの文字列の表示は次のフォーマットを取ります:

```
[ALT+][CTRL+][SHIFT+]key
```

Windows プラットフォームSDK 関数 `GetKeyNameText` を使用して `key` が変換された箇所。

18.7.1.2 ID

プロパティ: `ID` を long として

詳細:
このプロパティは、コマンドの一意の識別子を取得します。コマンドのID は(`Exec` を使用して) コマンドを実行するために、または、(`QueryStatus` を使用して) ステータスをクエリするために必須です。コマンドが他のコマンド (例えば、トップレベルのメニュー) のためのコンテナー、または、セパレーターの場合、ID は0 です。

18.7.1.3 IsSeparator

プロパティ: `IsSeparator` を boolean として

詳細:
コマンドオブジェクトがメニューセパレーターの場合、の場合、プロパティは、`true` を返します。それ以外の場合は、`false` を返します。次も参照してください: コマンド.

18.7.1.4 Label

プロパティ: `Label` を string として

詳細:
このプロパティは、MapForce のグラフィカルなユーザーインターフェイス内に表示されるとおりのコマンドのテキストを取得します。コマンドがセパレーターの場合、「ラベル」は、空の文字列です。このプロパティは、関連したGUI テキストを持たない、一部のツールバーコマンドのために空の文字列を返します。

18.7.1.5 Name

プロパティ: `Name` を string として

詳細:
このプロパティは一意のコマンド名を取得します。使用できる箇所で、この値はコマンドのアイコンファイルを取得するために使用できます。使用することのできるアイコンファイルは、使用中のMapForce インストールのフォルダー **<ApplicationFolder>\Examples\ActiveX\Images** 内にあります。

18.7.1.6 StatusText

プロパティ: `Label` を string として

詳細:

コマンドが選択されると、ステータステキストは、MapForce のステータスバー内で表示されるテキストです。他のコマンドのセパレーターまたは、コンテナオではないコマンドオブジェクトに対してのみ適用することができます。それ以外の場合、プロパティは空の文字列です。

18.7.1.7 SubCommands

プロパティ: SubCommands を Commands として

詳細:
SubCommands プロパティは、現在のコマンドのサブコマンドである Command オブジェクトのコレクションを取得します。プロパティは、他のコマンド(メニュー、サブメニュー、または、ツールバー)のためのコンテナーであるコマンドにのみ適用されます。このようなコンテナーコマンドは、0 に設定された ID を持ち、および、false に設定された IsSeparator プロパティを持ちます。

18.7.1.8 ToolTip

プロパティ: ToolTip を string として

詳細:
このプロパティは、各コマンドのためのヒントを表示するテキストを取得します。コマンドがツールのヒントのテキストを持たない場合、プロパティは、空の文字列を返します。

18.7.2 MapForceCommands

プロパティ:
Count
Item

詳細:
MapForceControl のコマンドレベルと ID へのアクセスを取得するための Command オブジェクトのコレクションです。これらのコマンドは、Exec メソッドを使用して実行することができます。 また、ステータスを QueryStatus を使用してクエリすることができます。

18.7.2.1 Count

プロパティ: Count を long として

詳細:
このコレクションのレベル上の Command オブジェクトの数。

18.7.2.2 Item

プロパティ: Item (n を long として) を Command として

詳細:
このコレクション内のインデックス n を持つコマンドを取得します。インデックスは 1 ベースです。

18.7.3 MapForceControl

プロパティ:
IntegrationLevel
Appearance
Application
BorderStyle
CommandsList
EnableUserPrompts
MainMenu
Toolbars

メソッド:
Open
Exec
QueryStatus

イベント:
OnUpdateCmdUI
OnOpenedOrFocused
OnCloseEditingWindow
OnFileChangedAlert
OnContextChanged
OnDocumentOpened
OnValidationWindowUpdated

このオブジェクトは完全な ActiveX コントロールで、MapForce ライブラリがアプリケーションレベルモードで使用されている場合、表示されます。

```
CLSID: A38637E9-5759-4456-A167-F01160CC22C1
ProgID: Altova.MapForceControl
```

18.7.3.1 プロパティ

以下のプロパティが定義されています:

IntegrationLevel
EnableUserPrompts
Appearance
BorderStyle

コマンドに関連したプロパティ:
CommandsList
MainMenu
Toolbars

MapForceAPI へのアクセス:
Application

(C) 2015-2021 Altova GmbH

18.7.3.1.1 Appearance

プロパティ: Appearance を short として

ディスパッチ ID : -520

詳細:
0 に対して等しくない値は、コントロールの周りでクライアントエッジを表示します。デフォルトの値は 0 です。

18.7.3.1.2 Application

プロパティ: Application を Application として

ディスパッチ ID : 1

詳細:
Application プロパティは、完全な MapForce 自動サーバーAPI の Application オブジェクトへのアクセスを与えます。プロパティは読み取り専用です

18.7.3.1.3 BorderStyle

プロパティ: BorderStyle を short として

ディスパッチ ID : -504

詳細:
1 の値は、薄いボーダーを持つコントロールを表示します。デフォルトの値は 0 です。

18.7.3.1.4 CommandsList

プロパティ: CommandList を Commands として (読み取り専用)

ディスパッチ ID : 1004

詳細:
このプロパティは、MapForceControl を持つ定義された全てのコマンドのフラットリストを返します。メニュー構造に従い、コマンドを整理するには、MainMenu を使用します。ツールバーコマンドを取得するには、Toolbars を使用します。

```
public void GetAllMapForceCommands()
{
    // Get all commands from the MapForce ActiveX control assigned to the current form
    MapForceControlLib.MapForceCommands commands = this.axMapForceControl1.CommandList;
    // Iterate through all commands
    for (int i = 0; i < commands.Count; i++)
```

```
    {
      // Get each command by index and output it to the console
      MapForceControlLib.MapForceCommand cmd = axMapForceControl1.CommandList[i];
      Console.WriteLine("{0} {1} {2}", cmd.ID, cmd.Name, cmd.Label.Replace("&", ""));
    }
}
```

C# サンプル

18.7.3.1.5 EnableUserPrompts

プロパティ: `EnableUserPrompts` を boolean として

ディスパッチ ID : 1006

詳細:
このプロパティを *false* に設定すると、コントロール内のユーザープロンプトを無効化します。デフォルトの値は *true* です。

18.7.3.1.6 IntegrationLevel

プロパティ: `IntegrationLevel` を`ICActiveXIntegrationLevel` として

ディスパッチ ID : 1000

詳細:
The `IntegrationLevel` プロパティ determines コントロールのオペレーションモード。次も参照してください: アプリケーションレベルの統合 とドキュメントレベルの統合 。

メモ `MapForceControl` オブジェクトの作成後すぐに、このプロパティを設定することは重要です。

18.7.3.1.7 MainMenu

プロパティ: `MainMenu` を `Command` として(読み取り専用)

ディスパッチ ID : 1003

詳細:
このプロパティは、 MapForceControl メインメニュー内で使用することのできる構造の情報とコマンドを `Command` オブジェクトとして提供します。オブジェクトは、MapForce の使用することのできる全てのサブメニューを含んでいます(例「ファイル」、「編集」、「表示」など)。サブメニューオブジェクトにアクセスするには、`MainMenu` プロパティの `SubCommands` プロパティを使用します。各サブメニューは `Command` オブジェクトでもあります。各サブメニューのために、対応する子コマンドセパレーターを取得するために、`SubCommands` プロパティを使用して更に反復することができます(このテクニックが使用される可能性があります。例えば、アプリケーションメニューをプログラム的に作成するなど)。メニューコマンドの一部は、`SubCommands` プロパティを持つ他のメニューコマンドのためのコンテナー(「親」)としての役割を果たします。全てのメニューコマンドの構造をプログラム的に取得するには、取得されたコマンドの情報 内の C# で説明されている通り、.再帰的な関数を作成する必要があります。

(C) 2015-2021 Altova GmbH

```
public void GetMapForceMenus()
{
    // Get the main menu from the MapForce ActiveX control assigned to the current form
    MapForceControlLib.MapForceCommand mainMenu = this.axMapForceControl1.MainMenu;

    // Loop through entries of the main menu (e.g. File, Edit, etc.)
    for (int i = 0; i < mainMenu.SubCommands.Count; i++)
    {
      MapForceControlLib.MapForceCommand menu = mainMenu.SubCommands[i];
      Console.WriteLine("{0} menu has {1} children items (including separators)",
menu.Label.Replace("&", ""), menu.SubCommands.Count);
    }
}
```

C# サンプル

18.7.3.1.8 Toolbars

プロパティ: `Toolbars` を Commands として (読み取り専用)

ディスパッチ ID : 1005

詳細:
このプロパティは、MapForceControl ツールバーの構造の情報を `Command` オブジェクトとして与えます。`Command` オブジェクトには、MapForce の使用することのできる全てのツールバーが含まれています。ツールバーにアクセスするには、`Toolbars` プロパティの `SubCommands` プロパティを使用します。各ツールバーは、`Command` オブジェクトでもあります。各ツールバーのために、コマンドを使用するために `SubCommands` プロパティを更に反復します (例えば、アプリケーションのツールバープログラム的に作成するためなどにこのテクニックは使用される場合があります)。

```
public void GetMapForceToolbars()
{
    // Get the application toolbars from the MapForce ActiveX control assigned to the
current form
    MapForceControlLib.MapForceCommands toolbars = this.axMapForceControl1.Toolbars;

    // Iterate through all toolbars
    for (int i = 0; i < toolbars.Count; i++)
    {
      MapForceControlLib.MapForceCommand toolbar = toolbars[i];
      Console.WriteLine();
      Console.WriteLine("The toolbar \"{0}\" has the following commands:",
toolbar.Label);

      // Iterate through all commands of this toolbar
      for (int j = 0; j < toolbar.SubCommands.Count; j++)
      {
         MapForceControlLib.MapForceCommand cmd = toolbar.SubCommands[j];
         // Output only command objects that are not separators
         if (!cmd.IsSeparator)
         {
            Console.WriteLine("{0}, {1}, {2}", cmd.ID, cmd.Name, cmd.Label.Replace("&",
 ""));
         }
      }
    }
}
```

C# サンプル

18.7.3.2 メソッド

以下のメソッドが定義されます:

Open
Exec
QueryStatus

18.7.3.2.1 Exec

Method: Exec (nCmdID を long として) を boolean として

ディスパッチ ID : 6

詳細:
このメソッドは、ID nCmdID を持つ MapForce コマンドを呼び出し、コマンドが実行可能な場合、メソッドは、true を返します。使用可能な全てのコマンドのリストを取得するには、CommandsList を使用してください。.コマンドの状態を取得するには、QueryStatus を使用してください。

18.7.3.2.2 Open

Method: Open (strFilePath を string として) を boolean として

ディスパッチ ID : 5

詳細:
メソッドの結果は、引数 strFilePath 内でパスされる拡張子により異なります。ファイル拡張子が .sps の場合、新規ドキュメントが開かれます。ファイル拡張子が .svp の場合、対応するプロジェクトが開かれます。異なるファイル拡張子がメソッドにパスされると、コントロールはファイルを新規のコンポーネントとしてアクティブなドキュメントにロードします。

ドキュメントレベルの統合モード内でコントロール使用する際、このメソッドを使用して、ドキュメントまたはプロジェクトをロードしないでください。代わりに、MapForceControlDocument.Open と MapForceControlPlaceHolder.OpenProject を使用してください。

18.7.3.2.3 QueryStatus

Method: QueryStatus (nCmdID を long として) を long として

ディスパッチ ID : 7

詳細:
QueryStatus は、nCmdID により指定されたコマンドのチェックされた/チェックが解除されたステータスを返します。このステータスはビットマスクとして返されます。

ビット	値	名前	意味

(C) 2015-2021 Altova GmbH

0	1	サポートされている コマンドがサポートされている場合設定します。
1	2	有効化されている コマンドが有効化されている場合設定します(実行可能)。
2	4	チェックされている コマンドがチェックされている場合設定します。

これは、`QueryStatus` が `0` コマンドを返すと、ID は、有効な MapForce コマンドとして認識されないことを意味します。`QueryStatus` が `1` または `5` の値を返すとコマンドは無効化されます。

18.7.3.3 Events

MapForceControl ActiveX コントロールは次の接続ポイントイベントを提供します:

OnUpdateCmdUI
OnOpenedOrFocused
OnCloseEditingWindow
OnFileChangedAlert
OnContextChanged

OnDocumentOpened
OnValidationWindowUpdated

18.7.3.3.1 OnCloseEditingWindow

イベント: `OnCloseEditingWindow` (i_strFilePath を String として) を boolean として

ディスパッチ ID: 1002

詳細:
MapForce が既に開かれているドキュメントを閉じる場合、このイベントはトリガーされます。このイベントへの応答として、クライアントは *i_strFilePath* に関連するエディターウィンドウを閉じます。このイベントが *true* を返すと、クライアントがドキュメントを閉じたことを指します。特定の処理が必要とされておらず、MapForceControl がエディターを閉じようと試み、関連したドキュメントコントロールを破棄する場合、クライアント は、*false* を返すことができます。

18.7.3.3.2 OnContextChanged

イベント: `OnContextChanged` (`i_strContextName` を String として、`i_bActive` を bool として) を bool として

ディスパッチ ID: 1004

詳細:
このイベントは、is not used in MapForce

18.7.3.3.3 OnDocumentOpened

イベント: `OnDocumentOpened` (`objDocument` を Document として)

ディスパッチ ID: 1

詳細:
このイベントはドキュメントが開かれる都度トリガーされます。引数 `objDocument` は、MapForce 自動化インターフェイス、からの `Document` オブジェクトであり、ドキュメントの詳細をクエリするために使用することができ、または、追加オペレーションを行います。ドキュメントレベルで統合される場合、イベント `MapForceControlDocument.OnDocumentOpened` を代わりに使用することが奨励されます。

18.7.3.3.4 OnFileChangedAlert

イベント: `OnFileChangedAlert` (`i_strFilePath` を String として) を bool として

ディスパッチ ID: 1001

詳細:
MapForceControl を使用してファイルがロードされると、isこのイベントはトリガーされます。他のアプリケーションによりハードディスク上で変更されます。イベントを処理すると、クライアントは、true を返します。または、MapForce が通常の方法で処理すると、false を返します。例えば、ユーザーに再ロードを促すなど。

18.7.3.3.5 OnLicenseProblem

イベント: `OnLicenseProblem` (`i_strLicenseProblemText` を String として)

ディスパッチ ID: 1005

詳細:
MapForceControl がこのコントロールのための有効なライセンスを使用できないことを検知すると、このイベントはトリガーされます。制限されているユーザーファイセンスの場合、コントロールが初期化されると発生する可能性があります。インテグレーターはこのイベントをしようして、このコントロールの機能へのアクセスを無効化することができます。このイベントから戻ると、コントロールは機能へのアクセスをブロックします(例、コントロール内の空のウィンドウを表示し、リクエストに対してエラーを返すなど)。

18.7.3.3.6 OnOpenedOrFocused

イベント: `OnOpenedOrFocused` (`i_strFilePath` を String として、`i_bOpenWithThisControl` を bool として)

ディスパッチ ID: 1000

詳細:
アプリケーションレベルで統合する場合、このイベントはクライアントにドキュメントが開かれたこと、またはMapForce によりアクティブに設定されたことを通知します。

ドキュメントレベルで統合する場合、このイベントは、クライアントにドキュメントウィンドウを ファイル `i_strFilePath` で開くように命令します。ファイルが既に開かれている場合、対応するドキュメントウィンドウをアクティブなウィンドウにしてください。

`i_bOpenWithThisControl` が `true` の場合、内部アクセスが必要とされるため、ドキュメントは、MapForceControl により開かれる必要があります。それ以外の場合、ファイルは、異なるエディターで開くことができます。

(C) 2015-2021 Altova GmbH

18.7.3.3.7 OnToolWindowUpdated

イベント: `OnToolWindowUpdated(pToolWnd` を long として)

ディスパッチ ID: 1006

詳細:
ツールウィンドウが更新されると、このイベントはトリガーされます。

18.7.3.3.8 OnUpdateCmdUI

イベント: `OnUpdateCmdUI ()`

ディスパッチ ID: 1003

詳細:
頻繁に呼び出され、`MapForceControl.QueryStatus` を使用して MapForce コマンドの状況をチェックする良い機会をインテグレーターに与えます。このコールバック内で long オペレーションを実行しないでください。

18.7.3.3.9 OnValidationWindowUpdated

イベント: `OnValidationWindowUpdated ()`

ディスパッチ ID: 3

詳細:
新しい情報と共に検証出力ウィンドウが更新されると、このイベントはトリガーされます。

18.7.4 MapForceControlDocument

プロパティ:
Appearance
BorderStyle
Document
IsModified
Path
ReadOnly

メソッド:
Exec
New
Open
QueryStatus
Reload
Save
SaveAs

イベント:
OnDocumentOpened
OnDocumentClosed
OnModifiedFlagChanged
OnContextChanged
OnFileChangedAlert
OnActivate

MapForceControl がドキュメントレベルモード内で統合されている場合、各ドキュメントは、型 MapForceControlDocument のオブジェクト内で表示されています。MapForceControlDocument には、1度に1つのドキュメントのみが含まれていますが、場合によっては異なるファイルを表示するために使用される場合があります。

このオブジェクトは完全な ActiveX コントロールです。

```
CLSID: DFBB0871-DAFE-4502-BB66-08CEB7DF5255
ProgID: Altova.MapForceControlDocument
```

18.7.4.1 プロパティ

以下のプロパティが定義されています:

ReadOnly
IsModified
Path
Appearance
BorderStyle

MapForceAPI へのアクセス:
Document

18.7.4.1.1 Appearance

プロパティ: Appearance を short として

ディスパッチ ID: *-520*

詳細:
0 に対して等しくない値は、ドキュメント コントロールの周りにクライアント エッジを表示します。デフォルトの値は 0 です。

18.7.4.1.2 BorderStyle

プロパティ: BorderStyle を short として

ディスパッチ ID: *-504*

詳細:
1 の値は、薄いボーダー持つコントロールを表示します。デフォルトの値は 0 です。

18.7.4.1.3 Document

プロパティ: `Document` をドキュメントとして

ディスパッチ ID : 1

詳細:
`Document` プロパティは、MapForce 自動サーバーAPI の `Document` オブジェクト へのアクセスを与えます。このインターフェイスは、コントロール内にロードされるドキュメントと使用することのできる追加機能を与えます。プロパティは読み取り専用です

18.7.4.1.4 IsModified

プロパティ: `IsModified` を boolean として(読み取り専用)

ディスパッチ ID : 1006

詳細:
ドキュメントコンテンツが最後に開かれてから変更された場合、`IsModified` は *true* を返します。再ロードし、オペレーションを保存します。それ以外の場合は、`false` を返します。

18.7.4.1.5 Path

プロパティ: `Path` を string として

ディスパッチ ID : 1005

詳細:
コントロール内にロードされたドキュメントのフルパス名に設定、または、取得します。

18.7.4.1.6 ReadOnly

プロパティ: `ReadOnly` を boolean として

ディスパッチ ID : 1007

詳細:
このプロパティを使用すると、ドキュメントの読み取り専用モードをオンとオフに切り替えることができます。`ReadOnly` が `true` の場合、変更はできないことに注意してください。

18.7.4.2 メソッド

以下のメソッドが定義されています:

ドキュメントハンドリング:
New
Open
Reload
Save
SaveAs

コマンドハンドリング::
Exec
QueryStatus

18.7.4.2.1 Exec

メソッド: Exec (nCmdID を long として) を boolean として

ディスパッチ ID: 8

詳細:
このメソッドは、ID nCmdID を持つ MapForce コマンドを呼び出し、コマンドが実行可能な場合、メソッドは、true を返します。現在アクティブなドキュメントがアプリケーションで使用できる場合のみ、このメソッドを呼び出すことができます。

メニュー構造に従いコマンドを整理するには、MapForceControl の MainMenu プロパティを使用します。ツールバーコマンドを取得するには、MapForceControl の Toolbars プロパティを使用します。

18.7.4.2.2 New

メソッド: New () を boolean として

ディスパッチ ID: 1000

詳細:
このメソッドはコントロール内で新規ドキュメントを初期化します。

18.7.4.2.3 Open

メソッド: Open (strFileName を string として) を boolean として

ディスパッチ ID: 1001

詳細:
Open は、ファイル strFileName を新規のドキュメントとして、コントロールにロードします。

(C) 2015-2021 Altova GmbH

18.7.4.2.4 QueryStatus

メソッド: QueryStatus (nCmdID を long として) を long として
ディスパッチ ID: 9

詳細:
QueryStatus は、nCmdID により指定されたコマンドのチェックされた/チェックが解除されたステータスを返します。このステータスはビットマスクとして返されます。

ビット	値	名前	意味
0	1	サポートされている	コマンドがサポートされている場合設定します。
1	2	有効化されている	コマンドが有効化されている場合設定します(実行可能)。
2	4	チェックされている	コマンドがチェックされている場合設定します。

これは、QueryStatus が 0 コマンドを返すと、ID は、有効な MapForce コマンドとして認識されないことを意味します。QueryStatus が 1 または 5 の値を返すとコマンドは無効化されます。アプリケーション内で現在アクティブなドキュメントが存在する場合、クライアントは、ドキュメントコントロールの QueryStatus メソッドを呼び出します。

18.7.4.2.5 Reload

メソッド: Reload () を boolean として

ディスパッチ ID: 1002

詳細:
Reload ファイルシステムからドキュメントコンテンツを更新する。

18.7.4.2.6 Save

メソッド: Save () を boolean として

ディスパッチ ID: 1003

詳細:
Save は、現在のドキュメントをロケーション Path に保存する。

18.7.4.2.7 SaveAs

メソッド: SaveAs (strFileName を string として) を boolean として

ディスパッチ ID: 1004

詳細:
SaveAs は Path を *strFileName* に設定し、ドキュメントをこの場所に保存します。

18.7.4.3 Events

MapForceControlDocument ActiveX コントロールは次の接続ポイントイベントを提供します:

OnDocumentOpened
OnDocumentClosed
OnModifiedFlagChanged
OnContextChanged
OnFileChangedAlert
OnActivate
OnSetEditorTitle

18.7.4.3.1 OnActivate

イベント: OnActivate ()

ディスパッチ ID: *1005*

詳細:
ドキュメントコントロールが有効化されると、フォーカスのある、ユーザー入力として使用することのできるこのイベントがトリガーされます。

18.7.4.3.2 OnContextChanged

イベント: OnContextChanged (i_strContextName を String として、i_bActive を bool として) を bool として

ディスパッチ ID: *1004*

詳細: なし

18.7.4.3.3 OnDocumentClosed

イベント: OnDocumentClosed (objDocument を Document として)

ディスパッチ ID: *1001*

詳細:
このイベントはドキュメントがこのコントロールにロードされると閉じられます。引数 objDocument は、MapForce 自動化インターフェイスからの Document オブジェクトです、で慎重に使用してください。

(C) 2015-2021 Altova GmbH

18.7.4.3.4 OnDocumentOpened

イベント: `OnDocumentOpened` (`objDocument` を Document として)

ディスパッチ ID: 1000

詳細:
このイベントは、このコントロール内でドキュメントが開かれるとトリガーされます。引数 `objDocument` は、MapForce 自動化インターフェイスからの `Document` オブジェクトです。ドキュメントの詳細をクエリするために使用することができ、または、追加オペレーションを行います。

18.7.4.3.5 OnDocumentSaveAs

イベント: `OnContextDocumentSaveAs` (`i_strFileName` を String として)

ディスパッチ ID: 1007

詳細:
このドキュメントが新しい名前で内部で保存される場合このイベントがトリガーされます。

18.7.4.3.6 OnFileChangedAlert

イベント: `OnFileChangedAlert` () を bool として

ディスパッチ ID: 1003

詳細:
このドキュメントにロードされると、このイベントはトリガーされます。コントロールはハードディスク上で他のアプリケーションに変更されます。イベントを処理すると、クライアントは、true を返します。または、MapForce が通常の方法で処理すると、false を返します。例えば、ユーザーに再ロードを促すなど。

18.7.4.3.7 OnModifiedFlagChanged

イベント: `OnModifiedFlagChanged` (`i_bIsModified` を boolean として)

ディスパッチ ID: 1002

詳細:
このイベントはドキュメントの状態を、変更された、および、変更されていない状態に切り替えると、トリガーされます。ドキュメントコンテンツが元のコンテンツと異なる場合、パラメーター *i_bIsModifed* is *true* それ以外の場合は、`false` を返します。

18.7.4.3.8 OnSetEditorTitle

イベント: OnSetEditorTitle ()

ディスパッチ ID: 1006

詳細:
このイベントは、含まれているドキュメントが内部で名前を与えられると挙げられます。

18.7.5 MapForceControlPlaceHolder

すべてのプロジェクトプレースホルダーウィンドウに使用することのできるプロパティ:
PlaceholderWindowID

プロジェクトプレースホルダーウィンドウのためのプロパティ:
Project

プロジェクトプレースホルダーウィンドウのためのメソッド:
OpenProject
CloseProject

MapForceControlPlaceHolder コントロールは、概要、ライブラリまたはプロジェクトウィンドウなどの追加のMapForce ウィンドウを表示するために使用されます。他のActiveX コントロールと同様に使用することができ、クライアントアプリケーション内に配置することができます。

```
CLSID: FDEC3B04-05F2-427d-988C-F03A85DE53C2
ProgID: Altova.MapForceControlPlaceHolder
```

18.7.5.1 プロパティ

以下のプロパティが定義されています:

PlaceholderWindowID

MapForceAPI へのアクセス:
Project

18.7.5.1.1 Label

プロパティ: Label を String (読み取り専用)として

ディスパッチ ID: 1001

詳細:
このプロパティはプレースホルダのタイトルへのアクセスを与えます。プロパティは読み取り専用です

18.7.5.1.2 PlaceholderWindowID

プロパティ: PlaceholderWindowID を MapForceControlPlaceholderWindow として

ディスパッチ ID：1

詳細:
このプロパティを使用すると、コントロールのクライアントエリア内で表示される MapForce ウィンドウが通知されます。
`PlaceholderWindowID` は、`MapForceControlPlaceholderWindow` 列挙の有効な値に設定することができます。コントロールはすぐに状態を変更し、すぐに新規の MapForce ウィンドウを表示します。

18.7.5.1.3 Project

プロパティ: `Project` Project として（読み取り専用）

ディスパッチ ID：2

詳細:
`Project` プロパティは、MapForce 自動サーバー API の `Project` オブジェクト へのアクセスを与えます。このインターフェイスは、コントロールへロードされるプロジェクトに使用される追加機能を与えます。プレースホルダウィンドウが `MapForceXProjectWindow (=3)` の値を持つ `PlaceholderWindowID` の場合のみ、プロパティは有効なプロジェクトインターフェイスを返します。プロパティは読み取り専用です

18.7.5.2 メソッド

以下のメソッドが定義されています:

`OpenProject`
`CloseProject`

18.7.5.2.1 OpenProject

メソッド: `OpenProject` (`strFileName` を string として) を boolean として

ディスパッチ ID：3

詳細:
`OpenProject` は `strFileName` を新規のプロジェクトとしてコントロールにロードします。プレースホルダウィンドウが `XMLSpyXProjectWindow (=3)` と異なる `PlaceholderWindowID` を持つ場合、メソッドは失敗します。

18.7.5.2.2 CloseProject

メソッド: `CloseProject` ()

ディスパッチ ID：4

詳細:

CloseProject はプロジェクトにロードされたコントロールを閉じます。プレースホルダウィンドウが XMLSpyXProjectWindow (=3) と異なる PlaceholderWindowID を持つ場合、メソッドは失敗します。

18.7.5.3 イベント

MapForceControlPlaceholder ActiveX コントロールは次の接続ポイントイベントを提供します:

OnModifiedFlagChanged

18.7.5.3.1 OnModifiedFlagChanged

イベント: OnModifiedFlagChanged (i_bIsModified を boolean として)

ディスパッチ ID: 1

詳細:
このイベントは MapForceXProjectWindow (=3) の PlaceholderWindowID を持つプレースホルダコントロールのためのみにトリガーされます。プロジェクトのコンテンツが変更された、または、変更されていない状態の間で変更があった場合、イベントは実行されます。プロジェクトコンテンツが元のコンテンツと異なる場合、パラメーター *i_bIsModifed* は *true* を返し、それ以外の場合は、false を返します。

18.7.5.3.2 OnSetLabel

イベント: OnSetLabel (i_strNewLabel を string として)

ディスパッチ ID: 1000

詳細:
プレースホルダウィンドウのタイトルが変更されると、挙げられます。

18.7.6 列挙

以下の列挙が定義されています:

ICActiveXIntegrationLevel
MapForceControlPlaceholderWindow

18.7.6.1 ICActiveXIntegrationLevel

MapForceControl の IntegrationLevel プロパティのための可能な値。

```
ICActiveXIntegrationOnApplicationLevel  = 0
ICActiveXIntegrationOnDocumentLevel     = 1
```

18.7.6.2 MapForceControlPlaceholderWindow

この列挙にはサポートされる追加 MapForce ウィンドウのリストが含まれています。

```
MapForceXNoWindow                = -1
MapForceXLibraryWindow           = 0
MapForceXOverviewWindow          = 1
MapForceXValidationWindow        = 2
MapForceXProjectWindow           = 3
MapForceXDebuggerValuesWindow    = 4
MapForceXDebuggerContextWindow   = 5
MapForceXDebuggerPointsWindow    = 6
```

19 付録

以下の付録にはMapForce に関する技術的な情報や、ライセンスに関する重要な情報が収められています。各付録には以下のようにサブセクションが収められています:

技術的なデータ

- OS ならびにメモリの要件
- Altova XML パーサー
- Altova XSLT とXQuery エンジン
- Unicode のサポート
- インターネットの使用
- ライセンス使用状況測定

ライセンス情報

- 電子的なソフトウェアの配布
- 著作権
- 使用許諾契約書

(C) 2015-2021 Altova GmbH

19.1 エンジン情報

このセクションにはAltova XML バリデーター、Altova XSLT 1.0 エンジン、Altova XSLT 2.0 エンジン、そしてAltova XQuery エンジンの実装に特化した情報が含まれます。

19.1.1 XSLT および XQuery エンジンに関する情報

MapForce のXSLT およびXQuery エンジンは、W 3C 仕様に従っています、ですから、XMLSpy の以前のバージョン内のAltova エンジンより厳密です。この結果、以前のエンジンで無視されていた小さなエラーが、MapForce によりエラーとして挙げられます。

例えば

- パス演算子の結果がノードと非-ノードを両方含む場合、型エラー(`err:XPTY0018`) です。
- パス式 `E1/E2` 内の`E1` がノードのシーケンスを評価しない場合、型エラー(`err:XPTY0019`) です。

この種類のエラーが発生した場合、XSLT/XQuery ドキュメントまたはインスタンスドキュメントを必要に応じて修正してください。

このセクションは、エンジンの実装固有の機能を仕様別に整理して説明します。

- XSLT 1.0
- XSLT 2.0
- XQuery 1.0

19.1.1.1 XSLT 1.0

MapForce のXSLT 1.0 エンジンは、W orld W ide W eb Consortium(ワールド・ワイド・ウェブ・コンソーシアム) (W 3C) の1999 年 11 月 16 日版のXSLT 1.0 勧告 および1999 年 11 月 16 日版のXPath 1.0 勧告 に準拠します。実装に関しての以下の情報にご注意してください。

実装についての注意点

`xsl:output` の`method` 属性がHTMLに設定された場合、または、がHTML 出力 デフォルトで選択されている場合、XML またはXSLT ファイル内の特殊文字はHTML ドキュメントにHTML 文字参照として出力内に挿入されます。例えば、文字 U+00A0 (ブレーク無しのスペースのための16 進数レファレンス) がHTML コード内に文字の参照(` ` or ` `) 、または、エンティティ参照 ` ` として挿入されます

19.1.1.2 XSLT 2.0

このセクション:

- エンジン 適合性
- 下位互換性
- 名前空間
- スキーマ認識
- 実装固有の振る舞い

適合性

MapForce のXSLT 2.0 エンジンは、W orld W ide W eb Consortium（ワールド・ワイド・ウェブ・コンソーシアム）（W 3C）の2007 年 1 月 23 日版のXSLT 2.0 勧告 および2010 年 12 月 14 日版のXPath 2.0 勧告に準拠します。

下位互換性

XSLT 2.0 エンジン は下位互換性を有します。XSLT 2.0 エンジンの 下位互換性が有効になるのは、 XSLT 1.0 スタイルシートを処理するためにXSLT 2.0 エンジン が使用される際です。XSLT 1.0 エンジン と下位互換性を持つXSLT 2.0 エンジンにより作成される出力に違いがあるかもしれないことに注意してください。

名前空間

XSLT 2.0 スタイルシートは、 XSLT 2.0. プレフィックス内で使用することのできる型コンストラクタ および関数を使用するため、以下の名前空間を宣言する必要があります。 下のリストは通常使用されるリストです。希望する場合は、代替プレフィックスを使用することもできます。

名前空間	プレフィックス	名前空間 URI
XML スキーマ型	xs:	http://www.w3.org/2001/XMLSchema
XPath 2.0 関数	fn:	http://www.w3.org/2005/xpath-functions

通常これらの名前空間は、以下のリストで表示されるようにxsl：スタイルシートまたは、 xsl:transform 要素で宣言されます:

```
<xsl：スタイルシート version="2.0"
    xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    xmlns:fn="http://www.w3.org/2005/xpath-functions"
    ...
</xsl：スタイルシート>
```

次の点に注意してください:

- XSLT 2.0 エンジンは、（上のテーブルでリストされている）XPath 2.0 およびXQuery 1.0 関数 名前空間 をデフォルトの関数名前空間として使用します。XPath 2.0 およびXSLT 2.0 関数をプレフィックス無しでスタイルシート内で使用することができます。XPath 2.0 関数 名前空間 をスタイルシート内でプレフィックスと共に宣言すると、割り当てられた宣言内でプレフィックスを追加して使用することができます。
- XML スキーマ 名前空間から型コンストラクタと型を使用する場合、名前空間 宣言内で使用された、プレフィックスを使用して型コンストラクタを呼び出さなければなりません（例えば、 xs:date）。
- XPath 2.0 関数の一部は、XML スキーマ データ型と同じ名前を保有します。例えば、 XPath 関数 fn:string および fn:boolean のためには、同じロケーション名: xs:string および xs:boolean を持つ、XML スキーマ データ型 が存在します。ですから、 XPath 式 string('Hello')を使用する場合、式は、 xs:string('Hello') ではなくて fn:string('Hello') として検証します。

スキーマ認識

XSLT 2.0 エンジンは、スキーマを認識します。ですから、ユーザー定義 スキーマ型 および xsl:validate 命令を使用することができます。

(C) 2015-2021 Altova GmbH

実装固有の振る舞い

以下は、XSLT 2.0 エンジンが、特定のXSLT 2.0 関数の振る舞いの実装-特定のアスペクトをどのように扱うかの説明です。

xsl:result-document
追加してサポートされる エンコードは以下の通りです(Altova-固有): `x-base16tobinary` および `x-base64tobinary`.

function-available
インスコープ関数の使用をテストする関数(XSLT、XPath、および 拡張関数)。

unparsed-text
`href` 属性は、以下を受け入れます(i) ベース-uri フォルダー内のファイルの相対パス、および(ii) 相対パスを持つまたは持たない `file://`プロトコール。追加してサポートされる エンコードは以下の通りです(Altova-固有): `x-binarytobase16` および `x-binarytobase64`.

unparsed-text-available
`href` 属性は、以下を受け入れます(i) ベース-uri フォルダー内のファイルの絶対パス、および(ii) 絶対パスを持つまたは持たない `file://`プロトコール。追加してサポートされる エンコードは以下の通りです(Altova-固有): `x-binarytobase16` および `x-binarytobase64`.

メモ RaptorXML の先行製品である AltovaXML で実装されていた以下のエンコード値は使用しないでください: `base16tobinary`, `base64tobinary`, `binarytobase16` and `binarytobase64`.

19.1.1.3 XQuery 1.0

このセクション:

- エンジン 適合性
- スキーマ認識
- エンコード
- 名前空間
- XML ソースと検証
- 静的および動的な型のチェック
- ライブラリモジュール
- 外部関数
- 照合順序
- 数値データの精度
- XQuery 命令サポート

適合性

MapForce のXQuery 1.0 エンジンは、W orld W ide W eb Consortium(ワールド・ワイド・ウェブ・コンソーシアム) (W 3C) の2010 年 12 月 14 日版のXQuery 1.0 勧告に準拠します。XQuery 標準は、多数の機能の実装についての裁量を提供します。下には、XQuery 1.0 エンジンがどのようにこれらの機能を実装するかについて説明するリストが下に挙げられています。

スキーマ認識

XQuery 1.0 エンジン はスキーマを認識します。

エンコード

UTF-8 および UTF-16 文字のエンコードは、サポートされています。

名前空間

以下の名前空間 URI と関連するバインドは定義済みです。

名前空間	プレフィックス	名前空間 URI
XML スキーマ型	xs:	http://www.w3.org/2001/XMLSchema
スキーマインスタンス	xsi:	http://www.w3.org/2001/XMLSchema-instance
内蔵の関数	fn:	http://www.w3.org/2005/xpath-functions
Local 関数	local:	http://www.w3.org/2005/xquery-local-functions

次の点に注意してください：

- XQuery 1.0 エンジンは、上にリストされたプレフィックスを名前空間に対応するバウンドとして認識します。
- Since the 上にリストされた内蔵の関数 名前空間は、XQuery 内のデフォルトの関数です。内蔵の関数 が呼び出される際、名前空間、fn: プレフィックス を使用する必要はありません。（例えば、 `string("Hello")` が `fn:string` 関数を呼び出す場合。）しかし、プレフィックス `fn:` は、クエリプロローグ内で名前空間を宣言することなく内蔵の関数を呼び出す時に使用することができます。（サンプル: `fn:string("Hello")`）.
- クエリプロローグ内で `default function` 名前空間 式を宣言することにより、デフォルトの関数 名前空間をすることにより変更することができます。
- XML スキーマ 名前空間空の型を使用する場合、プレフィックス `xs:` は、名前空間を明確に宣言することなく、また、これらのプレフィックスをクエリプロローグ内でバインドすることなく使用することができます。（サンプル: `xs:date` および `xs:yearMonthDuration`。）XML スキーマ 名前空間のために、他のプレフィックスを使用する場合は、クエリプロローグ内で明確に宣言されている必要があります。（サンプル: `declare 名前空間 alt = "http://www.w3.org/2001/XMLSchema"; alt:date("2004-10-04")`.）
- `untypedAtomic`, `dayTimeDuration`, および `yearMonthDuration` データ型 が23 January 2007 のCR 共に、XPath データ型 名前空間から XML スキーマ 名前空間へ移動されていることに注意してください。ですから以下となります: `xs:yearMonthDuration`。

関数のための名前空間、型コンストラクタ、ノード テスト、が間違って割り当てられている場合、エラーが発生します。しかし、一部の関数はスキーマデータ型と同じ名前を持つことに注意してください。 例: `fn:string` および `fn:boolean`。(`xs:string` および `xs:boolean` は宣言されています) 名前空間 プレフィックス、関数または型コンストラクタが使用されるか決定します。

XML ソースドキュメントと検証

XQuery 1.0 エンジンを使用して、実行される XQuery ドキュメント内の XML ドキュメントは、整形式である必要があります。しかし、XML スキーマに従い有効である必要はありません。XML ファイルが外部スキーマと関連付けられ、また有効な場合、ポスト スキーマ検証情報が XML データのために生成され、クエリ検証のために使用されます。

静的および動的な型のチェック

静的分析フェーズは、外部レファレンスの存在（例、モジュールのため）、呼び出された関数と変数が定義済みであるか、など構文などのクエリのアスペクトをチェックします。静的分析フェーズでエラーが検知されると、実行は停止されます。

クエリが実際に作動中にランタイム中に動的な型チェックは実行されます。型がオペレーションの必要条件と整合性を持たない場合、エラーが報告されます。例えば、式 `xs:string("1") + 1` は、エラーを返します。型 `xs:string` のオペランドを足し算のオペレーションが実行できないためです。

(C) 2015-2021 Altova GmbH

ライブラリモジュール

ライブラリモジュールは、再利用のため関数と変数を保管します。XQuery 1.0 エンジンは単一の外部 XQuery ファイルに保管されているモジュールをサポートします。このようなモジュールファイルはプロローグにターゲット名前空間に関連するモジュール宣言を含む必要があります。 以下はモジュールサンプルです:

```
module namespace libns="urn:module-library";
declare variable $libns:company := "Altova";
declare function libns:webaddress() { "http://www.altova.com" };
```

すべての関数および変数は、モジュールに関連した名前空間に属するモジュール内で宣言されています。モジュールはクエリプロローグ内の `import module` ステートメントを使用して XQuery ファイルにインポートする際に使用されます。`import module` ステートメントは、ライブラリモジュールファイル内で直接宣言された、関数と変数のみをインポートします。例:

```
import module namespace modlib = "urn:module-library" at "modulefilename.xq";
if         ($modlib:company = "Altova")
then        modlib:webaddress()
else        error("No match found.")
```

外部関数

外部関数 は、サポートされていません。例えば、`external` キーワードを仕様する式など。以下参照:

```
declare function hoo($param as xs:integer) as xs:string external;
```

照合順序

デフォルトの照合順序は、Unicode コードポイントをベースにした文字列を比較する Unicode-コードポイント照合順序です。その他にサポートされる照合順序は、ICU 照合順序 はここ にリストされるとおりです。特定の照合順序を使用する場合、サポートされる照合順序のリスト 内に与えられているとおり URI を提供します。`fn:max` と `fn:min` 関数を含む文字列の比較は、指定された照合順序により行われます。 照合順序オプションが指定されていない場合、デフォルトの Unicode-コードポイント照合順序 が使用されます。

数値データの精度

- `xs:integer` データ型には任意の精度があり、表記できる桁数に制限はありません。
- `xs:decimal` データ型には小数点の後に 20 桁の制限があります。
- `xs:float` と `xs:double` データ型には 15 桁の精度の制限があります。

XQuery 命令サポート

`Pragma` 命令 は、サポートされていません。発生した場合、無視されフォールバックの式が検証されます。

19.1.2 XSLT と XPath/XQuery 関数

このセクションでは、XPath および/または XQuery 式で使用することのできる、Altova 拡張関数と他の拡張関数をリストします。Altova 拡張関数は Altova の XSLT および XQuery エンジンで使用することができ、W3C 標準で定義された関数ライブラリで使用することのできる機能に追加して機能を提供します。

一般的な情報

以下の一般的な情報に注意してください:

- W3C 仕様により定義されているコア関数ライブラリの関数は、関数の呼び出しにプレフィックスは必要ありません。これは、XSLT および XQuery エンジンが、 XPath/XQuery 関数仕様で指定されている `http://www.w3.org/2005/xpath-functions` プレフィックス無しの関数をデフォルト関数の名前空間に属するものとして読み込むためです。
- 関数において、各アイテムが引数となるようなシーケンスが期待されており、2つ以上のアイテムがシーケンスにより呼び出された場合、エラーが返されます。
- 全ての比較は Unicode コードポイントコレクションを使用することで行われます。
- QName の結果は `[prefix:]localname` という形式でシリアル化されます。

xs:decimal の精度

精度とは、数値内にある桁数のことで、仕様では少なくとも18桁が求められます。`xs:decimal` 型に結果が収められる除算の場合、端数処理を行うことなく精度は小数点以下の19桁になります。

黙示的なタイムゾーン

2つの `date`、`time`、または `dateTime` 値を比較する場合、比較する値のタイムゾーンを明らかにする必要があります。値の中にタイムゾーンが明示的に与えられていない場合、黙示的なタイムゾーンが使用されます。黙示的なタイムゾーンはシステムクロックから取得され `implicit-timezone()` 関数によりその値をチェックすることができます。

照合順序

デフォルトの照合順序は、Unicode コードポイントをベースに文字列を比較する Unicode コードポイント照合順序です。エンジンは Unicode 照合アルゴリズムを使用しています。他のサポートされる照合順序は下にリストされる ICU 照合順序 です。使用するには、サポートされる照合順序のリストの URI を提供してください（*下のテーブル*）。`max` と `min` 関数を含む、文字列の比較は、指定された照合順序に沿って行われます。照合順序オプションが指定されていない場合、デフォルトの Unicode コードポイント照合順序が使用されます。

言語	URI
`da`: デンマーク語	`da_DK`
`de`: ドイツ語	`de_AT, de_BE, de_CH, de_DE, de_LI, de_LU`
`en`: 英語	`en_AS, en_AU, en_BB, en_BE, en_BM, en_BW, en_BZ, en_CA, en_GB, en_GU, en_HK, en_IE, en_IN, en_JM, en_MH, en_MP, en_MT, en_MU, en_NA, en_NZ, en_PH, en_PK, en_SG, en_TT, en_UM, en_US, en_VI, en_ZA, en_ZW`
`es`: スペイン語	`es_419, es_AR, es_BO, es_CL, es_CO, es_CR, es_DO, es_EC, es_ES, es_GQ, es_GT, es_HN, es_MX, es_NI, es_PA, es_PE, es_PR, es_PY, es_SV, es_US, es_UY, es_VE`
`fr`: フランス語	`fr_BE, fr_BF, fr_BI, fr_BJ, fr_BL, fr_CA, fr_CD, fr_CF, fr_CG, fr_CH, fr_CI, fr_CM, fr_DJ, fr_FR, fr_GA, fr_GN, fr_GP, fr_GQ, fr_KM, fr_LU, fr_MC, fr_MF, fr_MG, fr_ML, fr_MQ, fr_NE, fr_RE, fr_RW, fr_SN, fr_TD, fr_TG`
`it`: イタリア語	`it_CH, it_IT`
`ja`: 日本語	`ja_JP`
`nb`: ノルウェー語（ブークモール）	`nb_NO`
`nl`: オランダ語	`nl_AW, nl_BE, nl_NL`
`nn`: ノルウェー語（ニーノシュク）	`nn_NO`

(C) 2015-2021 Altova GmbH

pt: ポルトガル語	pt_AO, pt_BR, pt_GW, pt_MZ, pt_PT, pt_ST
ru: ロシア語	ru_MD, ru_RU, ru_UA
sv: スウェーデン語	sv_FI, sv_SE

名前空間軸
名前空間軸はXPath 2.0 にて廃止されましたが、名前空間軸の使用はサポートされています。XPath 2.0 メカニズムにより名前空間情報へアクセスするには、in-scope-prefixes()、namespace-uri()、namespace-uri-for-prefix(関数を使用してください。

19.1.2.1 Altova 拡張関数

Altova 拡張関数はXPath/XQuery 式で使用することができ、XPath、XQuery、およびXSLT 関数の標準ライブラリで使用可能な機能に更なる機能性を与えます。Altova 拡張関数はAltova 拡張関数名前空間、**http://www.altova.com/xslt-extensions** に収められており、**altova:** プレフィックスが、このセクションでは使用されます。製品の今後のバージョンが拡張機能への継続的サポート、または個別の関数の振る舞いは変更する可能性があることご注意してください。Altova 拡張機能へのサポートに関しては、今後のリリースのドキュメントを参照してください。

W 3C のXPath/XQuery 関数仕様で定義された関数は、以下で使用することができます: (i) XSLT 子アンテキスト内のXPath 式と、(ii) XQuery 文書内のXQuery 式。このドキュメントでは、前者(XSLT 内のXPath) のコンテキストで使用することのできる関数を、**XP** シンボルと共に表示し、と称します。後者(XQuery) で使用することのできる関数はの**XQ** シンボルと共に表示され、XQuery 関数と共に作業することができます。W 3C のXSLT 仕様は、XPath/XQuery 関数の仕様ではなく、XSLT 文書内のXPath 式でも使用することのできる関数を定義します。これらの関数は、**XSLT** シンボルと共に表示され、XSLT 関数と称されます。関数を使用することのできるXPath/XQuery およびXSLT のバージョンは、関数の詳細に記載されています(*下のシンボルを参照してください*)。
XPath/XQuery およびXSLT 関数ライブラリからの関数は、プレフィックス無しでリストされています。Altova 拡張関数などの、他のライブラリからの関数はプレフィックスと共にリストされています。

XPath 関数 (XSLT 内のXPath 式で使用):	**XP1** **XP2** **XP3.1**
XSLT 関数 (XSLT 内のXPath 式で使用):	**XSLT1** **XSLT2** **XSLT3**
XQuery 関数 (XQuery 内のXQuery 式で使用):	**XQ1** **XQ3.1**

XSLT 関数
XSLT 関数はXSLT 2.0 のcurrent-group() やkey() 関数と同様に、XSLT コンテキストにて使用することができます。(例えば、XQuery コンテキストなどの) 非-XSLT コンテキストでは使用することができません。XBRL に対するXSLT 関数は、XBRL をサポートするエディションのAltova 製品でのみ使用することができます。

XPath/XQuery 関数
XPath/XQuery 関数は、 XSLT コンテキスト、 XQuery 関数のXPath 式で使用することができます:

- 日付/時刻
- 位置情報
- イメージに関連した
- 数値

- シーケンス
- 文字列
- その他

チャート関数 (Enterprise および Server Editions のみ)

チャート関数のためのAltova 拡張子は、**Enterprise** ならびに**サーバー エディション**のAltova 製品でしかサポートされていません。XML データからチャートを生成することができます。

19.1.2.1.1 XSLT 関数

XSLT **拡張関数** はXSLT コンテキスト内のXPath 式にて使用することができます。(例えば、XQuery コンテキストなどの) 非-XSLT コンテキストでは使用することができません。

関数の名前指定と言語の適用性に関するメモ

Altova 拡張関数はXPath/XQuery 式で使用することができ、XPath、XQuery、およびXSLT 関数の標準ライブラリで使用可能な機能に更なる機能性を与えます。Altova 拡張関数はAltova **拡張関数名前空間**、**http://www.altova.com/xslt-extensions** に収められており、**altova:** プレフィックスが、このセクションでは使用されます。製品の今後のバージョンが拡張機能への継続的サポート、または個別の関数の振る舞いは変更する可能性があることご注意してください。Altova 拡張機能へのサポートに関しては、今後のリリースのドキュメントを参照してください。

XPath 関数 (XSLT 内のXPath 式で使用):	**XP1** **XP2** **XP3.1**
XSLT 関数 (XSLT 内のXPath 式で使用):	**XSLT1** **XSLT2** **XSLT3**
XQuery 関数 (XQuery 内のXQuery 式で使用):	**XQ1** **XQ3.1**

標準関数

▼ distinct-nodes [altova:]

altova:distinct-nodes(*node()**)* を **node()*** とする **XSLT1** **XSLT2** **XSLT3**
入力として1つ以上のノードを必要とし、同じセットから重複した値を持つノードを除いたノードを返します。XPath/XQuery 関数 `fn:deep-equal` を使用して比較を行うことができます。

⊟ *サンプル*

- **altova:altova:distinct-nodes**(`country`) は重複した値を持つものを除く、全ての子 `country` ノード返します。

▼ evaluate [altova:]

altova:evaluate(XPathExpression *as xs:string*[**, ValueOf$p1, ... ValueOf$pN**]) **XSLT1** **XSLT2** **XSLT3**
XPath 式を必要とし、必須引数として文字列をパスします。評価された式の出力を返します。例えば **altova:evaluate('//Name[1]')** は、ドキュメント内の最初の`Name` 要素のコンテンツを返します。式 `//Name[1]` は、一重引用符を使用することにより、文字列としてパスされます。

`altova:evaluate` 関数は、オプションとして追加の引数を持つことができます。これらの引数は、`p1`, `p2`, `p3`... `pN` の名前を持つスコープ内の変数の値です。使用に関して以下の点にご注意してください: (i) 変数は、`X` が整数である箇所のフォーム`pX` の名前と

共に定義される必要があります。(ii) altova:evaluate 関数の引数は、(*上の署名参照*) 2 番目の引数からは、数値順の変数の*シーケンス*に対応した引数の*シーケンス*変数の値を与えます: p1 to pN: 第 2 引数は変数 p1 の値で、第 3 引数は、変数 p2 の値です。(iii) 変数の値は型 item* である必要があります。

- *サンプル*

```
<xsl:variable name="xpath" select="'$p3, $p2, $p1'" />
<xsl:value-of select="altova:evaluate($xpath, 10, 20, 'hi')" />
```

outputs "hi 20 10"

上のリストに関して、以下の点に注意してください:

- altova:evaluate 式の第 2 引数は、変数 $p1 に割り当てられた値で、第三の引数は変数 $p2 に割り当てられた値です。
- 関数の第 4 番目の引数は、引用符による囲いで表示された文字列の値です。
- xs:variable 要素の select 属性は、XPath 式を提供します。この式は xs:string の型である必要があり、一重引用符で囲まれています。

- *変数の使用方法を更に説明するサンプル*

-
```
<xsl:variable name="xpath" select="'$p1'" />
<xsl:value-of select="altova:evaluate($xpath, //Name[1])" />
```
*最初の*Name *要素の出力値*

-
```
<xsl:variable name="xpath" select="'$p1'" />
<xsl:value-of select="altova:evaluate($xpath, '//Name[1]')" />
```
Outputs "//Name[1]"

altova:evaluate() 拡張関数は、XSLT スタイルシート内のXPath 式が動的に評価される必要のあるシチュエーションで役に立ちます。例えば、ユーザーが並べ替えの必要条件をリクエストする場合、このシチュエーションは属性 UserReq/@sortkey に保管されます。スタイルシートでは、以下の式が使用できます: <xsl:sort select="**altova:evaluate(../UserReq/@sortkey)**" order="ascending"/>。altova:evaluate() 関数は、コンテキストノードの親の UserReq 子要素の sortkey 属性を読み込みます。sortkey 属性の値が Price の場合、Price は、altova:evaluate() 関数により返され、select 属性 <xsl:sort select="**Price**" order="ascending"/> の値になります。この sort 命令が、Order という要素のコンテキスト内で発生する場合、Order 要素は Price の子の値に従い並べ替えられます。また、@sortkey の値が、Date の場合、Order 要素は、Date の子の値に従い並べ替えられます。ですから、Order の並べ替えの条件は、ランタイムでの sortkey 属性から選択されます。これは、以下の式などでは達成することはできません: <xsl:sort select="**../UserReq/@sortkey**" order="ascending"/>。上の場合、並べ替え条件は sortkey 属性自身であり、Price または Date (または、現在の sortkey のコンテンツ) ではありません。

メモ 静的なコンテキストは、変数以外以外で、呼び出し環境の名前空間、型、機能、を含みます。ベース URI とデフォルトの名前空間は継承されます。

- *追加サンプル*

- 静的な変数: <xsl:value-of select="$i3, $i2, $i1" />
3 つの変数の値を出力します。

- 動的な変数を持つ動的 XPath 式:
```
<xsl:variable name="xpath" select="'$p3, $p2, $p1'" />
<xsl:value-of select="altova:evaluate($xpath, 10, 20, 30)" />
```
"30 20 10" *を出力します。*

- 動的な変数を持たないXPath 式:
```
<xsl:variable name="xpath" select="'$p3, $p2, $p1'" />
<xsl:value-of select="altova:evaluate($xpath)" />
```
出力エラー: $p3 に対して定義されている変数はありません。

▼ encode-for-rtf [altova:]

altova:encode-for-rtf(input *as xs:string*, **preserveallwhitespace** *as xs:boolean*, **preservenewlines** *as xs:boolean*) を **xs:string** とする **XSLT2** **XSLT3**
RTF のためのコードに入力文字列を変換します。空白と新しい行は、それぞれの引数により指定される boolean の値に基づき保管されます。

[トップ]

XBRL 関数

Altova XBRL 関数はXBRL をサポートするAltova 製品のエディションのみで使用することができます。

▼ xbrl-footnotes [altova:]

altova:xbrl-footnotes(*node()***)** を **node()*** とする **XSLT2** **XSLT3**
ノードを入力引数として必要とし、入力ノードに参照されるXBRL フットノート ノードを返します。

▼ xbrl-labels [altova:]

altova:xbrl-labels(*xs:QName*, *xs:string***)** を **node()*** とする **XSLT2** **XSLT3**
以下の2 つの入力引数を必要とします: ノード名とノードを含むタクソノミファイルロケーション。関数は、入力ノードと関連したXBRL ラベルノードを返します。

[トップ]

19.1.2.1.2 XPath/XQuery 関数: 日付と時刻

Altova の日付/時刻拡張関数はXPath とXQuery 式で使用することができ、XML スキーマの異なる日付および時刻データ型で保存されているデータを処理するための追加機能を提供します。このセクションの関数は、Altova の**XPath 3.0** および**XQuery 3.0** エンジンで使用することができます。これらの関数は、XPath/XQuery コンテキストで使用することができます。

関数の名前指定と言語の適用性に関するメモ

Altova 拡張関数はXPath/XQuery 式で使用することができ、XPath、XQuery、およびXSLT 関数の標準ライブラリで使用可能な機能に更なる機能性を与えます。Altova 拡張関数は**Altova 拡張関数名前空間**、**http://www.altova.com/xslt-extensions** に収められており、**altova:** プレフィックスが、このセクションでは使用されます。製品の今後のバージョンが拡張機能への継続的サポート、または個別の関数の振る舞いは変更する可能性があることご注意してください。Altova 拡張機能へのサポートに関しては、今後のリリースのドキュメントを参照してください。

XPath 関数 (XSLT 内の XPath 式で使用):	XP1 XP2 XP3.1
XSLT 関数 (XSLT 内の XPath 式で使用):	XSLT1 XSLT2 XSLT3
XQuery 関数 (XQuery 内の XQuery 式で使用):	XQ1 XQ3.1

▼ 機能によりグループ化

- xs:dateTime に期間を追加して、xs:dateTimeを返す
- xs:date に期間を追加して、xs:date を返す
- xs:time に期間を追加して、return xs:time を返す
- フォーマットと期間の取得
- 現在の日付/時刻を生成する関数からタイムゾーンを削除する
- 期間から日付、時刻、分数、および、秒数を返す
- 日付から整数を週の曜日として返す
- 日付から週数を整数として返す
- 各型の構文コンポーネントから日付、時刻、期間 の型を構築する
- 文字列入力から 日付、日付時刻 または 時刻 を構築する
- 年齢に関連した関数

▼ アルファベット順にグループ化

altova:add-days-to-date
altova:add-days-to-dateTime
altova:add-hours-to-dateTime
altova:add-hours-to-time
altova:add-minutes-to-dateTime
altova:add-minutes-to-time
altova:add-months-to-date
altova:add-months-to-dateTime
altova:add-seconds-to-dateTime
altova:add-seconds-to-time
altova:add-years-to-date
altova:add-years-to-dateTime
altova:age
altova:age-details
altova:build-date
altova:build-duration
altova:build-time
altova:current-dateTime-no-TZ
altova:current-date-no-TZ
altova:current-time-no-TZ
altova:date-no-TZ
altova:dateTime-no-TZ
altova:days-in-month
altova:hours-from-dateTimeDuration-accumulated
altova:minutes-from-dateTimeDuration-accumulated
altova:seconds-from-dateTimeDuration-accumulated
altova:format-duration
altova:parse-date
altova:parse-dateTime
altova:parse-duration
altova:parse-time
altova:time-no-TZ
altova:weekday-from-date
altova:weekday-from-dateTime
altova:weeknumber-from-date

altova:weeknumber-from-dateTime

[トップ]

xs:dateTime に期間を追加する XP3.1 XQ3.1

これらの関数は**xs:dateTime** に期間を追加し、**xs:dateTime** を返します。xs:dateTime 型はCCYY-MM-DDThh:mm:ss.sss のフォーマットです。これはxs:date とxs:time フォーマットの連結で、T により区切られています。タイムゾーンサフィックス+01:00 (for example)は任意です。

add-years-to-dateTime [altova:]

altova:add-years-to-dateTime(DateTime *as xs:dateTime*, **Years** を *xs:integer*)
asxs:dateTime とする XP3.1 XQ3.1
日付までの期間を年数で表示します。第 2 の引数は第 1 の引数として与えられたxs:date に追加される年数です。結果はxs:date 型です。

サンプル

- **altova:add-years-to-dateTime**(xs:dateTime("2014-01-15T14:00:00"), 10) は2024-01-15T14:00:00 を返します。
- **altova:add-years-to-dateTime**(xs:dateTime("2014-01-15T14:00:00"), -4) は2010-01-15T14:00:00 を返します。

add-months-to-dateTime [altova:]

altova:add-months-to-dateTime(DateTime *as xs:dateTime*, **Months** を *xs:integer*)
asxs:dateTime とする XP3.1 XQ3.1
xs:dateTime に月数での期間を追加します(*下のサンプル参照*)。第 2 の引数は、第 1 の引数として与えられたxs:dateTime に追加される月数です。結果はxs:dateTime 型です。

サンプル

- **altova:add-months-to-dateTime**(xs:dateTime("2014-01-15T14:00:00"), 10) 2014-11-15T14:00:00 を返します。
- **altova:add-months-to-dateTime**(xs:dateTime("2014-01-15T14:00:00"), -2) 2013-11-15T14:00:00 を返します。

add-days-to-dateTime [altova:]

altova:add-days-to-dateTime(DateTime *as xs:dateTime*, **Days** *as xs:integer*) を**xs:dateTime** とする XP3.1 XQ3.1
xs:dateTime に日数での期間を追加します(*下のサンプル参照*)。第 2 の引数は、第 1 の引数として与えられたxs:dateTime に追加される日数です。結果はxs:dateTime 型です。

サンプル

- **altova:add-days-to-dateTime**(xs:dateTime("2014-01-15T14:00:00"), 10) は2014-01-25T14:00:00 を返します。
- **altova:add-days-to-dateTime**(xs:dateTime("2014-01-15T14:00:00"), -8) は2014-01-25T14:00:00 を返します。

add-hours-to-dateTime [altova:]

(C) 2015-2021 Altova GmbH

altova:add-hours-to-dateTime(DateTime *as xs:dateTime*, **Hours** *as xs:integer*) を **xs:dateTime** とする **XP3.1** **XQ3.1**
xs:dateTime に時間数での期間を追加します（*下のサンプル参照*）。第 2 の引数は、第 1 の引数として与えられた xs:dateTime に追加される時間数です。結果は xs:dateTime 型です。

サンプル

- **altova:add-hours-to-dateTime**(xs:dateTime("2014-01-15T13:00:00"), 10) は 2014-01-15T23:00:00 を返します。
- **altova:add-hours-to-dateTime**(xs:dateTime("2014-01-15T13:00:00"), -8) は 2014-01-15T05:00:00 を返します。

add-minutes-to-dateTime [altova:]

altova:add-minutes-to-dateTime(DateTime *as xs:dateTime*, **Minutes** *as xs:integer*) を **xs:dateTime** とする **XP3.1** **XQ3.1**
xs:dateTime に分数での期間を追加します（*下のサンプル参照*）。第 2 の引数は、第 1 の引数として与えられた xs:dateTime に追加される分数です。結果は xs:dateTime 型です。

サンプル

- **altova:add-minutes-to-dateTime**(xs:dateTime("2014-01-15T14:10:00"), 45) 2014-01-15T14:55:00 を返します。
- **altova:add-minutes-to-dateTime**(xs:dateTime("2014-01-15T14:10:00"), -5) 2014-01-15T14:05:00 を返します。

add-seconds-to-dateTime [altova:]

altova:add-seconds-to-dateTime(DateTime *as xs:dateTime*, **Seconds** を *xs:integer*) as **xs:dateTime** とする **XP3.1** **XQ3.1**
xs:dateTime に秒数での期間を追加します（*下のサンプル参照*）。第 2 の引数は、第 1 の引数として与えられた xs:dateTime に追加される秒数です。結果は xs:dateTime 型です。

サンプル

- **altova:add-seconds-to-dateTime**(xs:dateTime("2014-01-15T14:00:10"), 20) 2014-01-15T14:00:30 を返します。
- **altova:add-seconds-to-dateTime**(xs:dateTime("2014-01-15T14:00:10"), -5) 2014-01-15T14:00:05 を返します。

[トップ]

xs:date に期間を追加する **XP3.1** **XQ3.1**

これらの関数は **xs:date** に期間を追加し、**xs:date** を返します。xs:date 型は CCYY-MM-DD フォーマットです。

add-years-to-date [altova:]

altova:add-years-to-date(Date *as xs:date*, **Years** *as xs:integer*) を **xs:date** とする **XP3.1** **XQ3.1**
日付までの期間を年数で表示します。第 2 の引数は第 1 の引数として与えられた xs:date に追加される年数です。結果は xs:date 型です。

サンプル

- **altova:add-years-to-date**(xs:date("2014-01-15"), 10) は2024-01-15 を返します。
- **altova:add-years-to-date**(xs:date("2014-01-15"), -4) は2010-01-15 を返します。

▼ add-months-to-date [altova:]

altova:add-months-to-date(Date *as xs:date*, **Months** *as xs:integer*) を**xs:date** とする **XP3.1** **XQ3.1**
日付までの期間を月数で表示します。第 2 の引数は第 1 の引数として与えられたxs:date に追加される月数です。結果はxs:date 型です。

▣ *サンプル*

- **altova:add-months-to-date**(xs:date("2014-01-15"), 10) 2014-11-15 を返します。
- **altova:add-months-to-date**(xs:date("2014-01-15"), -2) 2013-11-15 を返します。

▼ add-days-to-date [altova:]

altova:add-days-to-date(Date *as xs:date*, **Days** *as xs:integer*) を**xs:date** とする **XP3.1** **XQ3.1**
日付までの期間を日数で表示します。第 2 の引数は第 1 の引数として与えられたxs:date に追加される日数です。結果はxs:date 型です。

▣ *サンプル*

- **altova:add-days-to-date**(xs:date("2014-01-15"), 10) は 2014-01-25 を返します。
- **altova:add-days-to-date**(xs:date("2014-01-15"), -8) は2014-01-07 を返します。

[トップ]

フォーマットと期間の取得 XP3.1 XQ3.1

これらの関数は入力 **xs:duration** または**xs:string** を解析し、それぞれan **xs:string** または**xs:duration** を返します。

▼ format-duration [altova:]

altova:format-duration(Duration *as xs:duration*, **Picture** *as xs:string*) **asxs:string** とする **XP3.1** **XQ3.1**
第 1 の引数として提出された期間を、第 2 の引数として提出された文字列によりをフォーマットします。出力は、文字列によりフォーマットされたテキスト文字列です。

▣ *サンプル*

- **altova:format-duration**(xs:duration("P2DT2H53M11.7S"), "Days:[D01] Hours:[H01] Minutes:[m01] Seconds:[s01] Fractions:[f0]") は"Days:02 Hours:02 Minutes:53 Seconds:11 Fractions:7" を返します。
- **altova:format-duration**(xs:duration("P3M2DT2H53M11.7S"), "Months:[M01] Days:[D01] Hours:[H01] Minutes:[m01]") は"Months:03 Days:02 Hours:02 Minutes:53" を返します。

▼ parse-duration [altova:]

altova:parse-duration(InputString *as xs:string*, **Picture** *as xs:string*) を**xs:duration** と

する XP3.1 XQ3.1
パターン化された文字列を最初の引数として、文字を第 2 の引数とします。入力文字列は文字をベースに解析され、xs:duration が返されます。

サンプル

- `altova:parse-duration("Days:02 Hours:02 Minutes:53 Seconds:11 Fractions:7", "Days:[D01] Hours:[H01] Minutes:[m01] Seconds:[s01] Fractions:[f0]")` は `"P2DT2H53M11.7S"` を返します。
- `altova:parse-duration("Months:03 Days:02 Hours:02 Minutes:53 Seconds:11 Fractions:7", "Months:[M01] Days:[D01] Hours:[H01] Minutes:[m01]")` は `"P3M2DT2H53M"` を返します。

[トップ]

xs:time に期間を追加する XP3.1 XQ3.1

これらの関数は **xs:time** に期間を追加し、 **xs:time** を返します。 **xs:time** 型は hh:mm:ss.sss 構文フォームです。文字 Z は協定世界時（UTC）を表します。他のタイムゾーンは、UTC との差異を +hh:mm または -hh:mm のフォーマットで表示しています。タイムゾーンの値が無い場合は、UTC ではない未知のタイムゾーンとして見なされます。

add-hours-to-time [altova:]

altova:add-hours-to-time(Time *as xs:time*, **Hours** *as xs:integer*) を **xs:time** とする XP3.1 XQ3.1
日付までの期間を時間数で表示します。第 2 の引数は第 1 の引数として与えられた xs:time に追加される時間数です。結果は xs:time 型です。

サンプル

- `altova:add-hours-to-time(xs:time("11:00:00"), 10)` は `21:00:00` を返します。
- `altova:add-hours-to-time(xs:time("11:00:00"), -7)` は `04:00:00` を返します。

add-minutes-to-time [altova:]

altova:add-minutes-to-time(Time *as xs:time*, **Minutes** *as xs:integer*) を **xs:time** とする XP3.1 XQ3.1
日付までの期間を分数で表示します。第 2 の引数は第 1 の引数として与えられた xs:date に追加される分数です。結果は xs:date 型です。

サンプル

- `altova:add-minutes-to-time(xs:time("14:10:00"), 45)` `14:55:00` を返します。
- `altova:add-minutes-to-time(xs:time("14:10:00"), -5)` `14:05:00` を返します。

add-seconds-to-time [altova:]

altova:add-seconds-to-time(Time *as xs:time*, **Minutes** *as xs:integer*) を **xs:time** とする XP3.1 XQ3.1
時間までの期間を秒数で表示します。第 2 の引数は、第 1 の引数として与えられた xs:time に追加される秒数です。結果は xs:time 型です。第 2 のコンポーネントは 0 から 59.999 の範囲であることができます。

サンプル

- `altova:add-seconds-to-time(xs:time("14:00:00"), 20)` は `14:00:20` を返します。
- `altova:add-seconds-to-time(xs:time("14:00:00"), 20.895)` は `14:00:20.895` を返します。

[トップ]

日付/時刻データ型からタイムゾーンの部分を削除する XP3.1 XQ3.1

これらの関数は、現在の**xs:dateTime**、**xs:date**、または**xs:time** 値からそれぞれタイムゾーンを削除します。xs:dateTime と xs:dateTimeStamp の差異は、後者のタイムゾーンが必要な場合です。(前者の場合は任意です。) xs:dateTimeStamp 値のフォーマットは CCYY-MM-DDThh:mm:ss.sss±hh:mm または CCYY-MM-DDThh:mm:ss.sssZ です。、日付と時刻が xs:dateTimeStamp としてシステムクロックから読み込まれる場合、 current-dateTime-no-TZ() 関数がタイムゾーンを削除するために使用されます。

current-date-no-TZ [altova:]

altova:current-date-no-TZ() を **xs:date** とする XP3.1 XQ3.1
この関数は引数を必要としません。current-date() (システムクロックによる現在の時刻) のタイムゾーンの部分を削除し、xs:date の値を返します。

サンプル

現在の date が **2014-01-15+01:00** の場合:

- **altova:current-date-no-TZ()** は 2014-01-15 を返します。

current-dateTime-no-TZ [altova:]

altova:current-dateTime-no-TZ() を **xs:dateTime** とする XP3.1 XQ3.1
この関数は引数を必要としません。current-dateTime() (システムクロックによる現在の時刻) のタイムゾーンの部分を削除し、xs:dateTime の値を返します。

サンプル

現在の dateTime が **2014-01-15T14:00:00+01:00** の場合:

- **altova:current-dateTime-no-TZ()** は 2014-01-15T14:00:00 を返します。

current-time-no-TZ [altova:]

altova:current-time-no-TZ() を **xs:time** とする XP3.1 XQ3.1
この関数は引数を必要としません。current-time() (システムクロックによる現在の時刻) のタイムゾーンの部分を削除し、xs:time の値を返します。

サンプル

現在の time が **14:00:00+01:00** の場合:

- **altova:current-time-no-TZ()** は 14:00:00 を返します。

date-no-TZ [altova:]

altova:date-no-TZ(InputDate *as xs:date*) を **xs:date** とする XP3.1 XQ3.1
この関数は xs:date 引数を必要とし、タイムゾーンの部分を削除し、xs:date の値を返します。日付が変更されていない点に注意してください。

サンプル

(C) 2015-2021 Altova GmbH

- **altova:date-no-TZ**(xs:date("2014-01-15+01:00")) は2014-01-15 を返します。

▼ dateTime-no-TZ [altova:]

altova:dateTime-no-TZ(InputDateTime *as xs:dateTime*) を**xs:dateTime** とする **XP3.1** **XQ3.1**
この関数はxs:dateTime 引数を必要とし、タイムゾーンの部分を削除し、xs:dateTime の値を返します。日付が変更されていない点に注意してください。

サンプル

- **altova:dateTime-no-TZ**(xs:date("2014-01-15T14:00:00+01:00")) は2014-01-15T14:00:00 を返します。

▼ time-no-TZ [altova:]

altova:time-no-TZ(InputTime *as xs:time*) を**xs:time** とする **XP3.1** **XQ3.1**
この関数はxs:time 引数を必要とし、タイムゾーンの部分を削除し、xs:time 値を返します。時刻は変更されていないことご注意してください。

サンプル

- **altova:time-no-TZ**(xs:time("14:00:00+01:00")) は14:00:00 を返します。

[トップ]

期間から日数、時間数、分数、および、秒数を返す **XP3.1** **XQ3.1**

これらの関数は、期間から、月内の日数、時間数、分数、秒数をそれぞれ返します。

▼ days-in-month [altova:]

altova:days-in-month(Year *as xs:integer*, **Month** *as xs:integer*) **asxs:integer** **XP3.1** **XQ3.1**
は指定された月内の日数 を返します。Year とMonth 引数を使用して月を指定することができます。

サンプル

- **altova:days-in-month**(2018, 10) は31 を返します。
- **altova:days-in-month**(2018, 2) は28 を返します。
- **altova:days-in-month**(2020, 2) は29 を返します。

▼ hours-from-dayTimeDuration-accumulated

altova:hours-from-dayTimeDuration-accumulated(DayAndTime *as xs:duration*) **asxs:integer** **XP3.1** **XQ3.1**
DayAndTime 引数 (which is of type xs:duration) により提供される期間内の時間の総数を返します。Day とTime コンポーネント内の時間は、整数である結果に追加されます。60 分として新規の一時間は考えられます。ネガティブな時間の値で期間の結果をナビゲートすることができます。

サンプル

- **altova:hours-from-dayTimeDuration-accumulated**(xs:duration("P5D") は5 日間の時間の総計である120 を返します。
- **altova:hours-from-dayTimeDuration-accumulated**(xs:duration("P5DT2H") は5 日間の時間の総計プラス2 時間である122 を返します。

- **altova:hours-from-dayTimeDuration-accumulated**(xs:duration("P5DT2H60M") は5日間の時間の総計プラス2時間プラス60分である 123 を返します。
- **altova:hours-from-dayTimeDuration-accumulated**(xs:duration("P5DT2H119M") は5日間の時間の総計プラス2時間とプラス119分である 123 を返します。
- **altova:hours-from-dayTimeDuration-accumulated**(xs:duration("P5DT2H120M") は5日間の時間の総計プラス2時間プラス120分である 124 を返します。
- **altova:hours-from-dayTimeDuration-accumulated**(xs:duration("-P5DT2H") は -122 を返します。

minutes-from-dayTimeDuration-accumulated

altova:minutes-from-dayTimeDuration-accumulated(DayAndTime *as xs:duration*)
asxs:integer **XP3.1** **XQ3.1**
DayAndTime 引数 (which is of type xs:duration) により提出される期間内の分数の総数を返します Day と Time コンポーネントは、追加され、整数である結果に追加されます。ネガティブなマイナスの値で期間の結果をナビゲートします。

サンプル

- **altova:minutes-from-dayTimeDuration-accumulated**(xs:duration("PT60M") は を返します。 60
- **altova:minutes-from-dayTimeDuration-accumulated**(xs:duration("PT1H") は一時間内の分数である 60 を返します。
- **altova:minutes-from-dayTimeDuration-accumulated**(xs:duration("PT1H40M") は 100 を返します。
- **altova:minutes-from-dayTimeDuration-accumulated**(xs:duration("P1D") は一日の秒数の総数である 1440 を返します。
- **altova:minutes-from-dayTimeDuration-accumulated**(xs:duration("-P1DT60M") は を返します。 -1500

seconds-from-dayTimeDuration-accumulated

altova:seconds-from-dayTimeDuration-accumulated(DayAndTime *as xs:duration*)
asxs:integer **XP3.1** **XQ3.1**
(xs:duration の型である)DayAndTime 引数 により提出された期間内の秒数の総数を返します。Day と Time コンポーネント内の秒数は、整数である結果に追加されます。ネガティブな秒の値で期間をナビゲートします。

サンプル

- **altova:seconds-from-dayTimeDuration-accumulated**(xs:duration("PT1M") は一分内の秒数である 60 を返します。
- **altova:seconds-from-dayTimeDuration-accumulated**(xs:duration("PT1H") は一時間内の秒数の総数である 3600 を返します。
- **altova:seconds-from-dayTimeDuration-accumulated**(xs:duration("PT1H2M") は 3720 を返します。
- **altova:seconds-from-dayTimeDuration-accumulated**(xs:duration("P1D") は一日の秒数の総数である 86400 を返します。
- **altova:seconds-from-dayTimeDuration-accumulated**(xs:duration("-P1DT1M") は -86460 を返します。

(C) 2015-2021 Altova GmbH

xs:dateTime または xs:dateReturn から週数を返す XP3.1 XQ3.1

これらの関数は、xs:dateTime または xs:date から曜日を(整数として)を返します。曜日は(米国式フォーマットを使用して) 1 から 7 と番号づけられています。 日曜=1 と番号付けられます。欧州のフォーマットでは月曜(=1) として番号付けられます。日曜=1 である米国フォーマットは整数 0 がフォーマットを表示するために使用できる箇所で設定することができます。

▼ weekday-from-dateTime [altova:]

altova:weekday-from-dateTime(DateTime *as xs:dateTime*) を **xs:integer** とする XP3.1 XQ3.1

時刻付きの日付を単一の引数として、この日付の曜日を正数として返します。曜日は 日曜=1 から開始して番号を付けます。(月曜=1 とする) ヨーロッパのフォーマットが必要な場合、この関数の他の署名を使用します (*下の次の署名を参照*)。

▣ *サンプル*

- **altova:weekday-from-dateTime**(xs:dateTime("2014-02-03T09:00:00")) は月曜を表示する 2 を返します。

altova:weekday-from-dateTime(DateTime *as xs:dateTime*, **Format** *as xs:integer*) **asxs:integer** とする XP3.1 XQ3.1

時間月の日付を最初の引数として、この日付の曜日を正数として返します。曜日は 月曜=1 から開始して番号を付けます。第 2 の引数(整数)が 0 の場合、日曜=1 から開始して、曜日は 1 から 7 と番号を付けます。第 2 の引数が 0 以外の整数の場合、月曜=1 です。第 2 の引数がない場合、関数はこの関数の他の署名を持つと読み込まれます (*前の次の署名を参照*)。

▣ *サンプル*

- **altova:weekday-from-dateTime**(xs:dateTime("2014-02-03T09:00:00"), 1) は月曜を表示する 1 を返します。
- **altova:weekday-from-dateTime**(xs:dateTime("2014-02-03T09:00:00"), 4) は月曜を表示する 1 を返します。
- **altova:weekday-from-dateTime**(xs:dateTime("2014-02-03T09:00:00"), 0) は月曜を表示する 2 を返します。

▼ weekday-from-date [altova:]

altova:weekday-from-date(Date *as xs:date*) を **xs:integer** とする XP3.1 XQ3.1

日付を単一の引数として、この日付の曜日を正数として返します。曜日は 日曜=1 から開始して番号を付けます。(月曜=1 とする) ヨーロッパのフォーマットが必要な場合、この関数の他の署名を使用します (*下の次の署名を参照*)。

▣ *サンプル*

- **altova:weekday-from-date**(xs:date("2014-02-03+01:00")) は月曜を表示する 2 を返します。

altova:weekday-from-date(Date *as xs:date*, **Format** *as xs:integer*) を **xs:integer** とする XP3.1 XQ3.1

日付を最初の引数とし、この日付の曜日を正数として返します。曜日は 月曜=1 から開始して番号を付けます。第 2 (フォーマット) 引数が 0, の場合、日曜=1 から開始し、曜日は 1 から 7 で番号付けられます。第 2 引数が整数で 0 以外の場合、月曜=1 です。第 2 の引数がない場合、関数はこの関数の他の署名を持つと読み込まれます (*前の署名を参照*)。

▣ *サンプル*

- **altova:weekday-from-date**(xs:date("2014-02-03"), 1) は月曜を示す 1 を返します。
- **altova:weekday-from-date**(xs:date("2014-02-03"), 4) は月曜を示す 1 を返します。
- **altova:weekday-from-date**(xs:date("2014-02-03"), 0) は月曜を示す 2 を返します。

[トップ]

xs:dateTime または xs:date から週数を返す XP2 XQ1 XP3.1 XQ3.1

これらの関数は週数(整数として)を xs:dateTime から xs:date から返します。週の番号付けは、米国、欧州、イスラムのカレンダーフォーマットで使用することができます。週の始まりが異なるため、週数の番号付けは、カレンダーのフォーマットにより異なります。(米国フォーマットでは、日曜、欧州フォーマットでは月曜、イスラムフォーマットでは土曜が週の開始日です)。

weeknumber-from-date [altova:]

altova:weeknumber-from-date(Date *as xs:date*, **Calendar** *as xs:integer*) を **xs:integer** とする
XP2 XQ1 XP3.1 XQ3.1

正数として提出された Date 引数の週数を返します。第 2 の引数(カレンダー)は続くカレンダーシステムを指定します。
サポートされる カレンダー の値は次のとおりです:

- 0 = US 米国のカレンダー(週の始まりは日曜日)
- 1 = ISO 標準、 欧州のカレンダー(週の始まりは月曜日)
- 2 = イスラムのカレンダー(週の始まりは土曜日)

デフォルトは 0 です。

サンプル

- **altova:weeknumber-from-date**(xs:date("2014-03-23"), 0) は 13 を返します。
- **altova:weeknumber-from-date**(xs:date("2014-03-23"), 1) は 13 を返します。
- **altova:weeknumber-from-date**(xs:date("2014-03-23"), 2) は 13 を返します。
- **altova:weeknumber-from-date**(xs:date("2014-03-23")) は 13 を返します。

上のサンプル(2014-03-23) の date の曜日は日曜日です。米国およびイスラムのカレンダーは欧州カレンダーのこの日付よりも一週間先です。

weeknumber-from-dateTime [altova:]

altova:weeknumber-from-dateTime(DateTime *as xs:dateTime*, **Calendar** *as xs:integer*) を **xs:integer** とする XP2 XQ1 XP3.1 XQ3.1

正数として提出された DateTime 引数の週数を返します。第 2 の引数(カレンダー)は続くカレンダーシステムを指定します。
サポートされる カレンダー の値は次のとおりです:

- 0 = US 米国のカレンダー(週の始まりは日曜日)
- 1 = ISO 標準、 欧州のカレンダー(週の始まりは月曜日)
- 2 = イスラムのカレンダー(週の始まりは土曜日)

Default is 0.

サンプル

- **altova:weeknumber-from-dateTime**(xs:dateTime("2014-03-23T00:00:00"), 0) は 13 を返します。
- **altova:weeknumber-from-dateTime**(xs:dateTime("2014-03-23T00:00:00"), 1) は 13 を返します。
- **altova:weeknumber-from-dateTime**(xs:dateTime("2014-03-23T00:00:00"), 2) は 13 を返します。

- **altova:weeknumber-from-dateTime**(xs:dateTime("2014-03-23T00:00:00")) は13 を返します。

上のサンプル(2014-03-23T00:00:00) のdateTime の曜日は日曜日です。米国およびイスラムのカレンダーは欧州カレンダーのこの日付よりも一週間先です。

[トップ]

各型の構文コンポーネントから日付、時刻、期間の型を構築する XP3.1 XQ3.1

関数は、xs:date, xs:time または xs:duration の構文コンポーネントを入力引数とし、引数を結合させて対応するデータ型を構築します。

build-date [altova:]

altova:build-date(Year *as xs:integer*, **Month** *as xs:integer*, **Date** *as xs:integer***)** を **xs:date** とする XP3.1 XQ3.1
第 1 、第 2 、第3 の引数は、それぞれ、年、月、日を表します。xs:date 型の値を構築するため結合されます。整数の値は、特定の日付の一部の適正な範囲内である必要があります。例えば第 2 の引数 (月の部分) は12 以上であってはなりません。

サンプル

- **altova:build-date**(2014, 2, 03) は2014-02-03 を返します。

build-time [altova:]

altova:build-time(Hours *as xs:integer*, **Minutes** *as xs:integer*, **Seconds** *as xs:integer***)** を**xs:time** とする XP3.1 XQ3.1
第 1 、第 2 、第3 引数はそれぞれ時間数 (0 から23)、分数 (0 から59)、および秒数 (0 から59) の値です。これらの値は、xs:time 型の値を作成するために結合されます。整数の値は、それぞれの部分の正しい範囲内である必要があります。例えば、秒 (分) 引数は、59 以上であってはなりません。値にタイムゾーンを追加するには、この関数の他の署名を使用してください (*次の署名を参照*)。

サンプル

- **altova:build-time**(23, 4, 57) は23:04:57 を返します。

altova:build-time(Hours *as xs:integer*, **Minutes** *as xs:integer*, **Seconds** *as xs:integer*, **TimeZone** *as xs:string***)** を**xs:time** とする XP3.1 XQ3.1
第 1 、第 2 、第 3 引数はそれぞれ時間数 (0 から23)、分数 (0 から59)、および秒数 (0 から59) の値です。第四の引数は、値の一部としてタイムゾーンを与えます。4 つの引数が結合され、xs:time 型の値を構築します。例えば、第 2 の引数 (分数) は59 以上であってはなりません。

サンプル

- **altova:build-time**(23, 4, 57, '+1') は23:04:57+01:00 を返します。

build-duration [altova:]

altova:build-duration(Years *as xs:integer*, **Months** *as xs:integer***)** を **xs:yearMonthDuration** とする XP3.1 XQ3.1
xs:yearMonthDuration 型の値を構築するためには2 つの引数が必要です。最初の引数は期間の 年数 値の部分を与え、第 2 の引数は、月数 値の部分を与えます。第 2 の引数 (月数) が と同じまたはより大きくなると、整数は12 により分割されます。商は、年数 の部分を表す 最初の引数に加算され、(除算の) 残りは の部分は 月数の部分を表すために使用されます。期間の

xs:dayTimeDuration 型を作成するには、次の署名を参照してください。

サンプル

- **altova:build-duration**(2, 10) は P2Y10M を返します。
- **altova:build-duration**(14, 27) は P16Y3M を返します。
- **altova:build-duration**(2, 24) は P4Y を返します。

altova:build-duration(Days *as xs:integer*, **Hours** *as xs:integer*, **Minutes** *as xs:integer*, **Seconds** *as xs:integer*) を **xs:dayTimeDuration** とする **XP3.1** **XQ3.1**

xs:dayTimeDuration 型の値を構築するためには 4 つの引数が必要です。最初の引数は期間の日数 値の部分で、第 2 、第 3 、第 4 引数は、それぞれ期間の時間数、分数、秒数 値のです。これらの 3 つの時間 引数は、次の高い単位の値に加算され、結果は期間の全体の値を計算するために使用されます。例えば、72 秒は、1M+12S (1 分 と12 秒)に変換され、この値が期間全体の値を計算するために使用されます。xs:yearMonthDuration 型の期間を構築するためには、次の署名を参照してください。

サンプル

- **altova:build-duration**(2, 10, 3, 56) は P2DT10H3M56S を返します。
- **altova:build-duration**(1, 0, 100, 0) は P1DT1H40M を返します。
- **altova:build-duration**(1, 0, 0, 3600) は P1DT1H を返します。

[トップ]

文字列入力から 日付、日付時刻 または 時刻 を構築する **XP2** **XQ1** **XP3.1** **XQ3.1**

関数は、文字列を引数として、xs:date、xs:dateTime または xs:time データ型を構築します。文字列は、データ型のコンポーネントを提出されたパターン引数をベースにして分析されます。

parse-date [altova:]

altova:parse-date(Date *as xs:string*, **DatePattern** *as xs:string*) を **xs:date** とする **XP2** **XQ1** **XP3.1** **XQ3.1**

Date 入力文字列を **xs:date** の値として返します。第二の引数である **DatePattern** は、入力文字列のパターン(コンポーネントのシーケンス) を指定します。**DatePattern** は、下にリストされるコンポーネント指定子および任意の文字であるコンポーネントセパレータと共に説明されています。下のサンプルを参照してください。

D 日付
M 月
Y 年

DatePattern のパターンは、**Date** のパターンに一致する必要があります。出力は **xs:date** 型であるため、出力は常に **YYYY-MM-DD** 構文フォーマットになります。

サンプル

- **altova:parse-date**(xs:string("06-03-2014"), "[D]-[M]-[Y]") は 2014-03-06 を返します。
- **altova:parse-date**(xs:string("06-03-2014"), "[M]-[D]-[Y]") は 2014-03-06 を返します。
- **altova:parse-date**("06/03/2014", "[M]/[D]/[Y]") は 2014-03-06 を返します。
- **altova:parse-date**("06 03 2014", "[M] [D] [Y]") は 2014-03-06 を返します。
- **altova:parse-date**("6 3 2014", "[M] [D] [Y]") は 2014-03-06 を返します。

parse-dateTime [altova:]

(C) 2015-2021 Altova GmbH

altova:parse-dateTime(DateTime *as xs:string*, **DateTimePattern** *as xs:string*) を **xs:dateTime** とする XP2 XQ1 XP3.1 XQ3.1

DateTime 入力文字列を xs:dateTime の値として返します。第二の引数である DateTimePattern は、入力文字列のパターン(コンポーネントのシーケンス)を指定します。DateTimePattern は、下にリストされるコンポーネント指定子および任意の文字であるコンポーネント セパレータと共に説明されています。下のサンプルを参照してください。

D	日
M	月
Y	年
H	時間
m	分
s	秒

DateTimePattern のパターンは、DateTime のパターンに一致する必要があります。出力は xs:dateTime 型であるため、出力は常に YYYY-MM-DDTHH:mm:ss 構文フォーマットになります。

サンプル

- `altova:parse-dateTime(xs:string("06-03-2014 13:56:24"), "[D]-[M]-[Y] [H]:[m]:[s]")` は `2014-03-06T13:56:24` を返します。
- `altova:parse-dateTime("time=13:56:24; date=06-03-2014", "time=[H]:[m]:[s]; date=[D]-[M]-[Y]")` は `2014-03-06T13:56:24` を返します。

parse-time [altova:]

altova:parse-time(Time *as xs:string*, **TimePattern** *as xs:string*) を **xs:time** とする XP2 XQ1 XP3.1 XQ3.1

Time 入力文字列を xs:time の値として返します。第二の引数である TimePattern は、入力文字列のパターン(コンポーネントのシーケンス)を指定します。TimePattern は、下にリストされるコンポーネント指定子および任意の文字であるコンポーネント セパレータと共に説明されています。下のサンプルを参照してください。

H	時間
m	分
s	秒

TimePattern のパターンは、Time のパターンに一致する必要があります。出力は xs:time 型であるため、出力は常に YYYY-HH:mm:ss 構文フォーマットになります。

サンプル

- `altova:parse-time(xs:string("13:56:24"), "[H]:[m]:[s]")` は `13:56:24` を返します。
- `altova:parse-time("13-56-24", "[H]-[m]")` は `13:56:00` を返します。
- `altova:parse-time("time=13h56m24s", "time=[H]h[m]m[s]s")` は `13:56:24` を返します。
- `altova:parse-time("time=24s56m13h", "time=[s]s[m]m[H]h")` は `13:56:24` を返します。

[トップ]

年齢に関連した関数 XP3.1 XQ3.1

関数は計算された年齢(i) 入力引数の日付と現在の日付の期間 (ii) 2 つの入力引数の日付の期間 を返します。altova:age 関数は、年齢を年数で返します、altova:age-details は年齢を年数、月数、日数からなる3 つの整数のシーケンスで返します。

▼ age [altova:]

altova:age(StartDate *as xs:date*) を **xs:integer** とする **XP3.1** **XQ3.1**
引数として提出された開始日から(システムクロックから取得された)現在の日付 までの日数を数えて、あるオブジェクトの年齢の年数である正数を返します。入力引数が1 年より先大きい場合、または一年の場合、返される値は負の数です。

▣ *サンプル*

If the current date is **2014-01-15**:

- **altova:age**(xs:date("2013-01-15")) は 1 を返します。
- **altova:age**(xs:date("2013-01-16")) は 0 を返します。
- **altova:age**(xs:date("2015-01-15")) は -1 を返します。
- **altova:age**(xs:date("2015-01-14")) は 0 を返します。

altova:age(StartDate *as xs:date*, **EndDate** *as xs:date*) を **xs:integer** とする **XP3.1** **XQ3.1**
最初の引数として提出された開始日から第 2 の引数の終了日 までの日数を数えて、あるオブジェクトの年齢の年数である正数を返します。最初の引数が1 年または第 2 の引数より先後の日付の場合、返される値は負の数です。

▣ *サンプル*

If the current date is **2014-01-15**:

- **altova:age**(xs:date("2000-01-15"), xs:date("2010-01-15")) は 10 を返します。
- **altova:age**(xs:date("2000-01-15"), current-date()) は、現在の日付が 2014-01-15 の場合 14 を返します。
- **altova:age**(xs:date("2014-01-15"), xs:date("2010-01-15")) は -4 を返します。

▼ age-details [altova:]

altova:age-details(InputDate *as xs:date*) を (**xs:integer**)* とする **XP3.1** **XQ3.1**
引数と(システムクロックから取得された)現在の日付として提出された日付の間の年数、月数、日数の3 つの整数を返します。返された years+months+days の合計は、2 つの日付(入力の日付と現在の日付)の時間差です。 入力の日付は現在の日付より先早いまたはお遅い値をもつことができますが、入力の日付が早いか遅いかは返される値で示されていません。戻される値は常に正の数です。

▣ *サンプル*

現在の日付が**2014-01-15** の場合:

- **altova:age-details**(xs:date("2014-01-16")) は (0 0 1) を返します。
- **altova:age-details**(xs:date("2014-01-14")) は (0 0 1) を返します。
- **altova:age-details**(xs:date("2013-01-16")) は (1 0 1) を返します。
- **altova:age-details**(current-date()) は (0 0 0) を返します。

altova:age-details(Date-1 *as xs:date*, **Date-2** *as xs:date*) を (**xs:integer**)* とする **XP3.1** **XQ3.1**
二つの引数間の年数、月数、日数の3 つの整数を返します。 返された years+months+days の合計は、2 つの入力の日付の時間差です。最初の引数として提出される二つの日付はどちらが速くても、または遅くてもかまいません。返される値は入力の日付が現在の日付より早いかまたは遅いかを示しません。戻される値は常に正の数です。

▣ *サンプル*

- **altova:age-details**(xs:date("2014-01-16"), xs:date("2014-01-15")) は (0 0 1) を返します。
- **altova:age-details**(xs:date("2014-01-15"), xs:date("2014-01-16")) は (0 0 1) を返しま

す。

[トップ]

19.1.2.1.3 XPath/XQuery 関数: 位置情報

以下の位置情報 XPath/XQuery 拡張関数は、MapForce の現在のバージョンによりサポートされています。また、次で使用することができます: (i) XSLT コンテキスト内の XPath 式、または (ii) XQuery ドキュメント内の XQuery 式。

関数の名前指定と言語の適用性に関するメモ

Altova 拡張関数は XPath/XQuery 式で使用することができ、XPath、XQuery、および XSLT 関数の標準ライブラリで使用可能な機能に更なる機能性を与えます。Altova 拡張関数は **Altova 拡張関数名前空間**、**http://www.altova.com/xslt-extensions** に収められており、**altova:** プレフィックスが、このセクションでは使用されます。製品の今後のバージョンが拡張機能への継続的サポート、または個別の関数の振る舞いは変更する可能性があることにご注意してください。Altova 拡張機能へのサポートに関しては、今後のリリースのドキュメントを参照してください。

XPath 関数 (XSLT 内の XPath 式で使用):	XP1 XP2 XP3.1
XSLT 関数 (XSLT 内の XPath 式で使用):	XSLT1 XSLT2 XSLT3
XQuery 関数 (XQuery 内の XQuery 式で使用):	XQ1 XQ3.1

▼ format-geolocation [altova:]

altova:format-geolocation(Latitude *as xs:decimal*, **Longitude** *as xs:decimal*, **GeolocationOutputStringFormat** *as xs:integer*) を **xs:string** とする XP3.1 XQ3.1
緯度と経度を最初の 2 つの引数とし、位置情報を文字列として出力します。第 3 の引数 **GeolocationOutputStringFormat** は、位置情報出力文字列のフォーマットです。出力文字列のフォーマットを識別するために、正数の値 1 から 4 を使用します。(*下の「位置情報出力文字列フォーマット」を参照してください*)。緯度の値の範囲は +90 から -90 (北 から 南) です。経度の値の範囲は +180 から -180 (東 から 西) です。

メモ image-exif-data 関数と Exif メタデータの @Geolocation 属性を位置情報入力文字列を提供する際に使用することができます。

▣ *サンプル*

- **altova:format-geolocation**(33.33, -22.22, 4) は xs:string "33.33 -22.22" を返します。
- **altova:format-geolocation**(33.33, -22.22, 2) は xs:string "33.33N 22.22W" を返します。
- **altova:format-geolocation**(-33.33, 22.22, 2) は xs:string "33.33S 22.22E" を返します。
- **altova:format-geolocation**(33.33, -22.22, 1) は xs:string "33°19'48.00"S 22° 13'12.00"E" を返します。

▣ *位置情報出力文字列フォーマット:*

与えられた緯度と経度は下にリストされる出力フォーマットによりフォーマットされます。希望するフォーマットは整数 ID (1 から

4) により識別されます。緯度の値の範囲は +90 から -90 (北 から 南) です。経度の値の範囲は +180 から -180 (東 から 西) です。

1
度、分、10 進の秒、方角のサフィックス付き (N/S, E/W) D°M'S.SS"N/S D°M'S.SS"E/W *サンプル:* 33°55'11.11"N 22°44'66.66"W

2
10 進の度、方角のサフィックス付き (N/S, E/W) D.DDN/S D.DDE/W *サンプル:* 33.33N 22.22W

3
度、分、10 進の秒、プレフィックスサイン付き (+/-); (N/E) のためのプラスサインは任意です。 +/-D°M'S.SS" +/-D°M'S.SS" *サンプル:* 33°55'11.11" -22°44'66.66"

4
10 進の度、プレフィックスサイン付き (+/-); (N/E) のためのプラスサインは任意です。 +/-D.DD +/-D.DD *サンプル:* 33.33 -22.22

Altova Exif 属性: 位置情報

Altova XPath/XQuery エンジンはカスタム属性 Geolocationを標準 Exif メタデータ タグから生成します。 Geolocation は、4 つのExif タグの連結です: 単位の追加された(*下のテーブル参照*) GPSLatitude、GPSLatitudeRef、GPSLongitude、GPSLongitudeRef 。

GPSLatitude	GPSLatitudeRef	GPSLongitude	GPSLongitudeRef	Geolocation
33 51 21.91	S	151 13 11.73	E	33°51'21.91"S 151° 13'11.73"E

parse-geolocation [altova:]

altova:parse-geolocation(GeolocationInputString *as xs:string*) を**xs:decimal+** とする **XP3.1 XQ3.1**

GeolocationInputString 引数を解析して、位置情報の緯度と経度 (この通りの順番) を 2 つのxs:decimal アイテムのシーケンスとして返します。位置情報入力文字列が提供されることのできるフォーマットは以下のリストの通りです。

メモ image-exif-data 関数とExif メタデータの@Geolocation 属性を位置情報入力文字列を提供する際に使用することができます。

サンプル

(C) 2015-2021 Altova GmbH

- **altova:parse-geolocation**("33.33 -22.22") は2 つのxs:decimals (33.33, 22.22) のシーケンスを返します。
- **altova:parse-geolocation**("48°51'29.6""N 24°17'40.2""") は2 つの xs:decimals (48.8582222222222, 24.2945) のシーケンスを返します。
- **altova:parse-geolocation**('48°51''29.6"N 24°17''40.2"') は2 つの xs:decimals (48.8582222222222, 24.2945) のシーケンスを返します。
- **altova:parse-geolocation**(**image-exif-data**(//MyImages/Image20141130.01)**/@Geolocation**) は2 つの xs:decimals のシーケンスを返します。

位置情報入力文字列フォーマット:

位置情報入力文字列は空白で区別された緯度と経度（この通りの順番）を含む必要があります。緯度と経度は以下のフォーマットをとることができます。組み合わせることも可能です。緯度が1つのフォーマットで、経度が他のフォーマットをとることができます。緯度の値の範囲は+90 から-90 (N から S)。経度の値の範囲は+180 から-180 (E から W)。

メモ 単一およびダブル引用符が入力文字列引数を区切るために使用されていると、使用されている単一およびダブル引用が、それぞれ、分の値と秒の値、不一致をもたらします。この様な場合、分の値と秒の値を表すための使用されている引用符は、ダブルにしてエスケープされる必要があります。このセクションのサンプルでは、入力文字列を区別するために使用されている引用符は黄色い(") でハイライトされており、エスケープした単位インジケーターは青い("") でハイライトされています。

- 度、分、10 進の秒、方角のサフィックス付き（N/S, E/W)
  D°M'S.SS"N/S D°M'S.SS"W/E
  サンプル: **33°55'11.11"N 22°44'55.25"W**

- 度、分、10 進の秒、プレフィックスサイン付き（+/-); (**N/E**) のためのプラスサインは任意です。+/-D°M'S.SS" +/-D°M'S.SS"
  サンプル: **33°55'11.11" -22°44'55.25"**

- 度、分、10 進の分、方角のサフィックス付き (N/S, E/W)
  D°M.MM'N/S D°M.MM'W/E
  サンプル: **33°55.55'N 22°44.44'W**

- 度、分、10 進の分、プレフィックスサイン付き（+/-); (**N/E**) のためのプラスサインは任意です。+/-D°M.MM' +/-D°M.MM'
  サンプル: **+33°55.55' -22°44.44'**

- 10 進の度、方角のサフィックス付き（N/S, E/W)
  D.DDN/S D.DDW/E
  サンプル: **33.33N 22.22W**

- 10 進の度、プレフィックスサイン付き（+/-); (**N/S E/W**)) のためのプラスサインは任意です。+/-D.DD +/-D.DD
  サンプル: **33.33 -22.22**

フォーマットの組み合わせのサンプル:

```
33.33N  -22°44'55.25"
33.33  22°44'55.25"W
33.33  22.45
```

Altova Exif 属性: 位置情報

Altova XPath/XQuery エンジンはカスタム属性 **Geolocation**を標準 Exif メタデータ タグから生成します。

Geolocation は、4 つの Exif タグの連結です: 単位の追加された(*下のテーブル参照*) GPSLatitude、GPSLatitudeRef、GPSLongitude、GPSLongitudeRef。

GPSLatitude	GPSLatitudeRef	GPSLongitude	GPSLongitudeRef	Geolocation
33 51 21.91	S	151 13 11.73	E	33°51'21.91"S 151° 13'11.73"E

geolocation-distance-km [altova:]

altova:geolocation-distance-km(GeolocationInputString-1 *as xs:string*, **GeolocationInputString-2** *as xs:string*) を **xs:decimal** とする **XP3.1** **XQ3.1**
2 つの位置情報の間の距離をキロメートルで計算します。位置情報入力文字列を提供することのできるフォーマットは下にリストされています。緯度の値の範囲は +90 から -90 (北 から 南) です。経度の値の範囲は +180 から -180 (東 から 西) です。

メモ image-exif-data 関数と Exif メタデータの @Geolocation 属性を位置情報入力文字列を提供する際に使用することができます。

サンプル

- **altova:geolocation-distance-km**("33.33 -22.22", "48°51'29.6""N 24°17'40.2""") は xs:decimal 4183.08132372392 を返します。

位置情報入力文字列フォーマット:

位置情報入力文字列は空白で区別された緯度と経度 (この通りの順番) を含む必要があります。緯度と経度は以下のフォーマットをとることができます。組み合わせることも可能です。緯度が 1 つのフォーマットで、経度が他のフォーマットをとることができます。緯度の値の範囲は +90 から -90 (N から S)。経度の値の範囲は +180 から -180 (E から W)。

メモ 単一およびダブル引用符が入力文字列引数を区切るために使用されていると、使用されている単一およびダブル引用が、それぞれ、分の値と秒の値、不一致をもたらします。この様な場合、分の値と秒の値を表すための使用されている引用符は、ダブルにしてエスケープされる必要があります。このセクションのサンプルでは、入力文字列を区別するために使用されている引用符は黄色い (") でハイライトされており、エスケープした単位インジケーターは青い ("") でハイライトされています。

- 度、分、10 進の秒、方角のサフィックス付き (N/S, E/W)
  D°M'S.SS"N/S D°M'S.SS"W/E
  サンプル: **33°55'11.11"N 22°44'55.25"W**

- 度、分、10 進の秒、プレフィックスサイン付き (+/-); (**N/E**) のためのプラスサインは任意です。+/-D°M'S.SS" +/-D°M'S.SS"
  サンプル: **33°55'11.11" -22°44'55.25"**

- 度、分、10 進の分、方角のサフィックス付き (N/S, E/W)
  D°M.MM'N/S D°M.MM'W/E
  サンプル: **33°55.55'N 22°44.44'W**

- 度、分、10 進の分、プレフィックスサイン付き (+/-); (**N/E**) のためのプラスサインは任意です。+/-D°M.MM' +/-D°M.MM'
  サンプル: **+33°55.55' -22°44.44'**

(C) 2015-2021 Altova GmbH

- 10 進の度、方角のサフィックス付き (N/S, E/W)
  D.DDN/S D.DDW/E
  サンプル: 33.33N 22.22W

- 10 進の度、プレフィックスサイン付き (+/-); (N/S E/W)) のためのプラスサインは任意です。+/-D.DD +/-D.DD
  サンプル: 33.33 -22.22

フォーマットの組み合わせのサンプル:
33.33N -22°44'55.25"
33.33 22°44'55.25"W
33.33 22.45

Altova Exif 属性: 位置情報

Altova XPath/XQuery エンジンはカスタム属性 Geolocationを標準 Exif メタデータ タグから生成します。 Geolocation は、4 つのExif タグの連結です: 単位の追加された(*下のテーブル参照*) GPSLatitude、GPSLatitudeRef、GPSLongitude、GPSLongitudeRef 。

GPSLatitude	GPSLatitudeRef	GPSLongitude	GPSLongitudeRef	Geolocation
33 51 21.91	S	151 13 11.73	E	33°51'21.91"S 151° 13'11.73"E

geolocation-distance-mi [altova:]

altova:geolocation-distance-mi(GeolocationInputString-1 *as xs:string*, **GeolocationInputString-2** *as xs:string*) を **xs:decimal** とする **XP3.1** **XQ3.1**
2 つの位置情報の間の距離をマイルで計算します。位置情報入力文字列を提供することのできるフォーマットは下にリストされています。緯度の値の範囲は +90 から -90 (北 から 南) です。経度の値の範囲は +180 から -180 (東 から 西) です。

メモ image-exif-data 関数とExif メタデータの @Geolocation 属性を位置情報入力文字列を提供する際に使用することができます。

サンプル

- **altova:geolocation-distance-mi**("33.33 -22.22", "48°51'29.6""N 24°17'40.2""")
  は xs:decimal 2599.40652340653 を返します。

位置情報入力文字列フォーマット:

位置情報入力文字列は空白で区別された緯度と経度 (この通りの順番) を含む必要があります。緯度と経度は以下のフォーマットをとることができます。組み合わせることも可能です。緯度が 1 つのフォーマットで、経度が他のフォーマットをとることができます。緯度の値の範囲は +90 から -90 (N から S)。経度の値の範囲は +180 から -180 (E から W)。

メモ 単一およびダブル引用符が入力文字列引数を区切るために使用されていると、使用されている単一およびダブル引用が、それぞれ、分の値と秒の値、不一致をもたらします。この様な場合、分の値と秒の値を表すための使用されている引用符は、ダブルにしてエスケープされる必要があります。このセクションのサンプルでは、入力文字列を区別するために使用されている引用符は黄色い (") でハイライトされており、エスケープした単位インジケーターは青い ("") でハイライトされています。

- 度、分、10 進の秒、方角のサフィックス付き (N/S, E/W)

`D°M'S.SS"N/S D°M'S.SS"W/E`
サンプル: `33°55'11.11"N 22°44'55.25"W`

- 度、分、10 進の秒、プレフィックスサイン付き(`+/-`); (`N/E`) のためのプラスサインは任意です。+/-D°M'S.SS" +/-D°M'S.SS"
  サンプル: `33°55'11.11" -22°44'55.25"`

- 度、分、10 進の分、方角のサフィックス付き (`N/S`, `E/W`)
  `D°M.MM'N/S D°M.MM'W/E`
  サンプル: `33°55.55'N 22°44.44'W`

- 度、分、10 進の分、プレフィックスサイン付き(`+/-`); (`N/E`) のためのプラスサインは任意です。+/-D°M.MM' +/-D°M.MM'
  サンプル: `+33°55.55' -22°44.44'`

- 10 進の度、方角のサフィックス付き(`N/S`, `E/W`)
  `D.DDN/S D.DDW/E`
  サンプル: `33.33N 22.22W`

- 10 進の度、プレフィックスサイン付き(`+/-`); (`N/S E/W`)) のためのプラスサインは任意です。+/-D.DD +/-D.DD
  サンプル: `33.33 -22.22`

フォーマットの組み合わせのサンプル:
```
33.33N  -22°44'55.25"
33.33  22°44'55.25"W
33.33  22.45
```

Altova Exif 属性: 位置情報

Altova XPath/XQuery エンジンはカスタム属性 `Geolocation`を標準 Exif メタデータ タグから生成します。`Geolocation` は、4 つのExif タグの連結です: 単位の追加された(*下のテーブル参照*) `GPSLatitude`、`GPSLatitudeRef`、`GPSLongitude`、GPSLongitudeRef 。

GPSLatitude	GPSLatitudeRef	GPSLongitude	GPSLongitudeRef	Geolocation
33 51 21.91	S	151 13 11.73	E	33°51'21.91"S 151° 13'11.73"E

geolocations-bounding-rectangle [altova:]

`altova:geolocations-bounding-rectangle(Geolocations を xs:sequence として GeolocationOutputStringFormat を xs:integer として )` を`xs:string` として `XP3.1` `XQ3.1`
最初の引数として文字列のシーケンスを取ります。シーケンスの各文字列が位置情報です。関数はそれぞれが、最初の引数に送信されたすべての位置情報を含むために最適にサイズ調整された長方形の左上と右下の位置情報の座標である2 つの文字列のシーケンスを返します。位置情報入力文字列が表示される書式は以下にリストされています(*「位置情報入力文字列書式」を参照してください*)。緯度の値は`+90` から`-90` への(`N` から`S` への) の範囲です。経度の値は`+180` から`-180` への(`E` から`W` への) の範囲です。

関数の2 番目の引数は出力シーケンス内の2 つの位置情報文字列の書式を指定します。引数はそれぞれの値が異なる位置情報文字列書式を識別する`1` から`4` の整数の値を取ります(*下記の「位置情報出力文字列書式」を参照してください*)。

メモ image-exif-data 関数とExif メタデータの属性は入力文字列を提供するために使用することができます。

サンプル

- `altova:geolocations-bounding-rectangle(("48.2143531 16.3707266", "51.50939 -0.11832"), 1)` はシーケンス `("51°30'33.804"N 0°7'5.952"W", "48°12'51.67116"N 16°22'14.61576"E")` を返します。
- `altova:geolocations-bounding-rectangle(("48.2143531 16.3707266", "51.50939 -0.11832", "42.5584577 -70.8893334"), 4)` はシーケンス `("51.50939 -70.8893334", "42.5584577 16.3707266")` を返します

位置情報入力文字列フォーマット:

位置情報入力文字列は空白で区別された緯度と経度（この通りの順番）を含む必要があります。緯度と経度は以下のフォーマットをとることができます。組み合わせることも可能です。緯度が1つのフォーマットで、経度が他のフォーマットをとることができます。緯度の値の範囲は +90 から -90 (N から S)。経度の値の範囲は +180 から -180 (E から W)。

メモ 単一およびダブル引用符が入力文字列引数を区切るために使用されていると、使用されている単一およびダブル引用が、それぞれ、分の値と秒の値、不一致をもたらします。この様な場合、分の値と秒の値を表すための使用されている引用符は、ダブルとしてエスケープされる必要があります。このセクションのサンプルでは、入力文字列を区別するために使用されている引用符は黄色い(") でハイライトされており、エスケープした単位インジケーターは青い("") でハイライトされています。

- 度、分、10 進の秒、方角のサフィックス付き (N/S, E/W)
  D°M'S.SS"N/S D°M'S.SS"W/E
  サンプル: **33°55'11.11"N 22°44'55.25"W**

- 度、分、10 進の秒、プレフィックスサイン付き (+/-); (**N/E**) のためのプラスサインは任意です。+/-D°M'S.SS" +/-D°M'S.SS"
  サンプル: **33°55'11.11" -22°44'55.25"**

- 度、分、10 進の分、方角のサフィックス付き (N/S, E/W)
  D°M.MM'N/S D°M.MM'W/E
  サンプル: **33°55.55'N 22°44.44'W**

- 度、分、10 進の分、プレフィックスサイン付き (+/-); (**N/E**) のためのプラスサインは任意です。+/-D°M.MM' +/-D°M.MM'
  サンプル: **+33°55.55' -22°44.44'**

- 10 進の度、方角のサフィックス付き (N/S, E/W)
  D.DDN/S D.DDW/E
  サンプル: **33.33N 22.22W**

- 10 進の度、プレフィックスサイン付き (+/-); (**N/S E/W**)) のためのプラスサインは任意です。+/-D.DD +/-D.DD
  サンプル: **33.33 -22.22**

フォーマットの組み合わせのサンプル:

```
33.33N  -22°44'55.25"
33.33  22°44'55.25"W
33.33  22.45
```

位置情報出力文字列フォーマット:

与えられた緯度と経度は下にリストされる出力フォーマットによりフォーマットされます。希望するフォーマットは整数 ID（1 から 4）により識別されます。緯度の値の範囲は +90 から -90（北 から 南）です。経度の値の範囲は +180 から -180（東 から 西）です。

1
度、分、10 進の秒、方角のサフィックス付き (N/S, E/W) D°M'S.SS"N/S D°M'S.SS"E/W *サンプル:* 33°55'11.11"N 22°44'66.66"W

2
10 進の度、方角のサフィックス付き (N/S, E/W) D.DDN/S D.DDE/W *サンプル:* 33.33N 22.22W

3
度、分、10 進の秒、プレフィックスサイン付き (+/-); (N/E) のためのプラスサインは任意です。 +/-D°M'S.SS" +/-D°M'S.SS" *サンプル:* 33°55'11.11" -22°44'66.66"

4
10 進の度、プレフィックスサイン付き (+/-); (N/E) のためのプラスサインは任意です。 +/-D.DD +/-D.DD *サンプル:* 33.33 -22.22

Altova Exif 属性: 位置情報

Altova XPath/XQuery エンジンはカスタム属性 Geolocation を標準 Exif メタデータ タグから生成します。Geolocation は、4 つの Exif タグの連結です: 単位の追加された（*下のテーブル参照*) GPSLatitude、GPSLatitudeRef、GPSLongitude、GPSLongitudeRef。

GPSLatitude	GPSLatitudeRef	GPSLongitude	GPSLongitudeRef	Geolocation
33 51 21.91	S	151 13 11.73	E	33°51'21.91"S 151° 13'11.73"E

geolocation-within-polygon [altova:]

altova:geolocation-within-polygon(Geolocation *as xs:string*, **((PolygonPoint** *as xs:string*)+)) を **xs:boolean** とする **XP3.1 XQ3.1**

PolygonPoint 引数により規明されている **Geolocation**（最初の引数）が多角形のエリア内に存在するかを決定します。もし、PolygonPoint 引数が最初と最後のポイントが同じ場合に作成される閉じられたフィギュアを作成しない場合、フィギュアを閉じるために、最初のポイントが明示的に最後のポイントとして追加されます。全ての引数（Geolocation および PolygonPoint+）は、（*下にリストされるフォーマットの*）位置情報入力文字列により提供されます。Geolocation 引数が多角形エリア内にある場合、関数は true(); を返します。その他の場合は false() を返します。緯度の値の範囲は +90 から -

90 (北 から 南) です。経度の値の範囲は+180 から-180 (東 から 西)です。

メモ image-exif-data 関数とExif メタデータの@Geolocation 属性を位置情報入力文字列を提供する際に使用することができます。

サンプル

- **altova:geolocation-within-polygon("33 -22", ("58 -32", "-78 -55", "48 24", "58 -32"))** は true() を返します。
- **altova:geolocation-within-polygon("33 -22", ("58 -32", "-78 -55", "48 24"))** は true() を返します。
- **altova:geolocation-within-polygon("33 -22", ("58 -32", "-78 -55", "48° 51'29.6""N 24°17'40.2""))** は true() を返します。

位置情報入力文字列フォーマット:

位置情報入力文字列は空白で区別された緯度と経度 (この通りの順番) を含む必要があります。緯度と経度は以下のフォーマットをとることができます。組み合わせることも可能です。緯度が1つのフォーマットで、経度が他のフォーマットをとることができます。緯度の値の範囲は+90 から-90 (N から S)。経度の値の範囲は+180 から-180 (E から W)。

メモ 単一およびダブル引用符が入力文字列引数を区切るために使用されていると、使用されている単一およびダブル引用が、それぞれ、分の値と秒の値、不一致をもたらします。この様な場合、分の値と秒の値を表すための使用されている引用符は、ダブルにしてエスケープされる必要があります。このセクションのサンプルでは、入力文字列を区別するために使用されている引用符は黄色い(") でハイライトされており、エスケープした単位インジケーターは青い("") でハイライトされています。

- 度、分、10 進の秒、方角のサフィックス付き (N/S, E/W)
  D°M'S.SS"N/S D°M'S.SS"W/E
  サンプル: **33°55'11.11"N 22°44'55.25"W**

- 度、分、10 進の秒、プレフィックスサイン付き (+/-); (**N/E**) のためのプラスサインは任意です。+/-D°M'S.SS" +/-D°M'S.SS"
  サンプル: **33°55'11.11" -22°44'55.25"**

- 度、分、10 進の分、方角のサフィックス付き (N/S, E/W)
  D°M.MM'N/S D°M.MM'W/E
  サンプル: **33°55.55'N 22°44.44'W**

- 度、分、10 進の分、プレフィックスサイン付き (+/-); (**N/E**) のためのプラスサインは任意です。+/-D°M.MM' +/-D°M.MM'
  サンプル: **+33°55.55' -22°44.44'**

- 10 進の度、方角のサフィックス付き (N/S, E/W)
  D.DDN/S D.DDW/E
  サンプル: **33.33N 22.22W**

- 10 進の度、プレフィックスサイン付き (+/-); (**N/S E/W**)) のためのプラスサインは任意です。+/-D.DD +/-D.DD
  サンプル: **33.33 -22.22**

フォーマットの組み合わせのサンプル:

```
33.33N  -22°44'55.25"
33.33  22°44'55.25"W
33.33  22.45
```

Altova Exif 属性: 位置情報

Altova XPath/XQuery エンジンはカスタム属性 Geolocationを標準 Exif メタデータ タグから生成します。 Geolocation は、4 つのExif タグの連結です: 単位の追加された(下のテーブル参照) GPSLatitude、 GPSLatitudeRef、GPSLongitude、GPSLongi tudeRef 。

GPSLatitude	GPSLatitudeRef	GPSLongitude	GPSLongitudeRef	Geolocation
33 51 21.91	S	151 13 11.73	E	33°51'21.91"S 151° 13'11.73"E

geolocation-within-rectangle [altova:]

altova:geolocation-within-rectangle(Geolocation *as xs:string*, **RectCorner-1** *as xs:string*, **RectCorner-2** *as xs:string*) を **xs:boolean** とする **XP3.1** **XQ3.1**
長方形の対格の角を指定する第 2 及び第 3 引数(RectCorner-1、RectCorner-2)により説明されている Geolocation (最初の引数) が長方形のエリア内に存在するかを決定します。全ての引数(Geolocation、RectCorner-1 および RectCorner-2) は、(下にリストされるフォーマットの) 位置情報入力文字列により提供されます。Geolocation 引数が長方形エリア内にある場合、関数は true(); を返します。その他の場合は false()を返します。緯度の値の範囲は +90 から -90 (北 から 南) です。経度の値の範囲は +180 から −180 (東 から 西)です。

メモ image-exif-data 関数とExif メタデータの @Geolocation 属性を位置情報入力文字列を提供する際に使用することができます。

サンプル

- **altova:geolocation-within-rectangle("33 -22", "58 -32", "-48 24")** は true() を返します。
- **altova:geolocation-within-rectangle("33 -22", "58 -32", "48 24")** は false() を返します。
- **altova:geolocation-within-rectangle("33 -22", "58 -32", "48°51'29.6""S 24° 17'40.2""")** は true() を返します。

位置情報入力文字列フォーマット:

位置情報入力文字列は空白で区別された緯度と経度 (この通りの順番) を含む必要があります。緯度と経度は以下のフォーマットをとることができます。組み合わせることも可能です。緯度が 1 つのフォーマットで、経度が他のフォーマットをとることができます。緯度の値の範囲は +90 から -90 (N から S)。経度の値の範囲は +180 から −180 (E から W)。

メモ 単一およびダブル引用符が入力文字列引数を区切るために使用されていると、使用されている単一およびダブル引用が、それぞれ、分の値と秒の値、不一致をもたらします。この様な場合、分の値と秒の値を表すための使用されている引用符は、ダブルにしてエスケープされる必要があります。このセクションのサンプルでは、入力文字列を区別するために使用されている引用符は黄色い(") でハイライトされており、エスケープした単位インジケーターは青い("") でハイライトされています。

- 度、分、10 進の秒、方角のサフィックス付き(N/S, E/W)
  D°M'S.SS"N/S D°M'S.SS"W/E
  サンプル: **33°55'11.11"N 22°44'55.25"W**

- 度、分、10 進の秒、プレフィックスサイン付き(+/-); (**N/E**) のためのプラスサインは任意です。+/-D°M'S.SS" +/-D°M'S.SS"

サンプル: `33°55'11.11"  -22°44'55.25"`

- 度、分、10 進の分、方角のサフィックス付き (`N/S`, `E/W`)
  `D°M.MM'N/S  D°M.MM'W/E`
  サンプル: `33°55.55'N  22°44.44'W`

- 度、分、10 進の分、プレフィックスサイン付き (`+/-`); (`N/E`) のためのプラスサインは任意です。+/-D°M.MM' +/-D°M.MM'
  サンプル: `+33°55.55'  -22°44.44'`

- 10 進の度、方角のサフィックス付き (`N/S`, `E/W`)
  `D.DDN/S  D.DDW/E`
  サンプル: `33.33N  22.22W`

- 10 進の度、プレフィックスサイン付き (`+/-`); (`N/S E/W`)) のためのプラスサインは任意です。+/-D.DD +/-D.DD
  サンプル: `33.33  -22.22`

フォーマットの組み合わせのサンプル:

```
33.33N  -22°44'55.25"
33.33  22°44'55.25"W
33.33  22.45
```

Altova Exif 属性: 位置情報

Altova XPath/XQuery エンジンはカスタム属性 `Geolocation`を標準 Exif メタデータ タグから生成します。`Geolocation` は、4 つの Exif タグの連結です: 単位の追加された (*下のテーブル参照*) `GPSLatitude`、`GPSLatitudeRef`、`GPSLongitude`、`GPSLongitudeRef`。

GPSLatitude	GPSLatitudeRef	GPSLongitude	GPSLongitudeRef	Geolocation
33 51 21.91	S	151 13 11.73	E	33°51'21.91"S 151° 13'11.73"E

[トップ]

19.1.2.1.4 XPath/XQuery 関数: イメージに関連

以下のイメージに関連した XPath/XQuery 拡張関数は、MapForce の現在のバージョンによりサポートされています。また、次で使用することができます: (i) XSLT コンテキスト内の XPath 式、または (ii) XQuery ドキュメント内の XQuery 式。

関数の名前指定と言語の適用性に関するメモ

Altova 拡張関数は XPath/XQuery 式で使用することができ、XPath、XQuery、および XSLT 関数の標準ライブラリで使用可能な機能に更なる機能性を与えます。Altova 拡張関数は Altova 拡張関数名前空間、`http://www.altova.com/xslt-extensions` に収められており、`altova:` プレフィックスが、このセクションでは使用されます。製品の今後のバージョンが拡張機能への継続的サポート、または個別の関数の振る舞いは変更する可能性があることにご注意してください。Altova 拡張機能へのサポートに関しては、今後のリリースのドキュメントを参照してください。

XPath 関数 (XSLT 内の XPath 式で使用):	XP1 XP2 XP3.1
XSLT 関数 (XSLT 内の XPath 式で使用):	XSLT1 XSLT2 XSLT3
XQuery 関数 (XQuery 内の XQuery 式で使用):	XQ1 XQ3.1

▼ suggested-image-file-extension [altova:]

altova:suggested-image-file-extension(Base64String *as string*) を **string?** とする XP3.1 XQ3.1
イメージファイルのBase64 エンコード を引数として、イメージのファイル拡張子を、イメージのBase64エンコード内の記録として返します。返された値は、エンコード内で使用することのできるイメージ型情報を基にしたヒントです。この情報が使用できない場合、空の文字列が返されます。この関数は、Base64 イメージをファイルとして保存し、適切なファイル拡張子を動的取得する際に役に立ちます。

サンプル

- **altova:suggested-image-file-extension**(/MyImages/MobilePhone/Image20141130.01) は 'jpg' を返します。
- **altova:suggested-image-file-extension**($XML1/Staff/Person/@photo) は '' を返します。

上のサンプルでは、関数の引数として与えられたノードはBase64 エンコードイメージを含むと仮定します。最初のサンプルは jpg をファイルの型および拡張子として取得します。二番目のサンプルでは、与えられたBase64 エンコードは使用できる拡張子の情報を提供しません。

▼ mt-transform-image [altova:]

altova:mt-transform-image(Base64Image *as Base64BinaryString*, **Size** *as item()+*, **Rotation** *as xs:integer*, **Quality** *as xs:integer*) を **Base64BinaryString** とする XP3.1 XQ3.1
Base64-エンコード イメージ を最初の引数として、変換されたBase64-エンコード イメージを返します。第 2、第 3、第 4 引数は変換された以下のイメージ パラメーターです: サイズ、回転、およびクオリティ。

- サイズ引数には3 つのサイズ変更のオプションがあります。

(X,Y)	絶対ピクセルの値。アスペクト 率は保持されません。高さと幅は自動的にイメージの長いおよび短い辺に逢わされるため、高さと幅の指示は関係ありません。2 つの整数アイテムのシーケンスとして値が入力されます。かっこが必要です。
X	X をピクセルの新しい長い辺として、イメージの縦横比のサイズ変更が行われます。アスペクト 率は保持されます。値は整数で引用符なしで入力されます。
'X%'	元のディメンションの与えられたパーセンテージにイメージのサイズを変更します。値は文字列として引用符を付けて入力される必要があります。

- 回転は以下の値を持つことができます: 90、180、270、-90、-180、-270。これらの値は回転の度数です。正の値はイメージを時計回りに回転させます。負の値はイメージを反時計回りに回転させます。 Altova Exif 属性 **OrientationDegree** を使用して、イメージの現在の回転の度数(0, 90, 180, 270) をイメージのExif **Orientation** タグから取得することができます。ですが、**OrientationDegree** 属性はデータの**Orientation** タグから取得されるため、Exif データ内に**Orientation** タグが存在する場合のみ使用することができます。(*下の OrientationDegree 説明を参照してください*)。
- クオリティは、0 から100 の値で、JPEG 圧縮のIJG クオリティスケールの値を参照していますが、クオリティのパーセンテージのインジケーターではありません。サイズとクオリティのどちらかを優先すると、もう一方の優先度が下がります。フルカラーソースのイメージでは、75 が通常最高値と見なされています。もし、75 が満足のいく結果をもたらさない場合は、値を上げてく

(C) 2015-2021 Altova GmbH

ださい。

メモ Exif データが元のイメージに存在する場合、変換時に削除され、変換されたイメージにはExif データは存在しません。

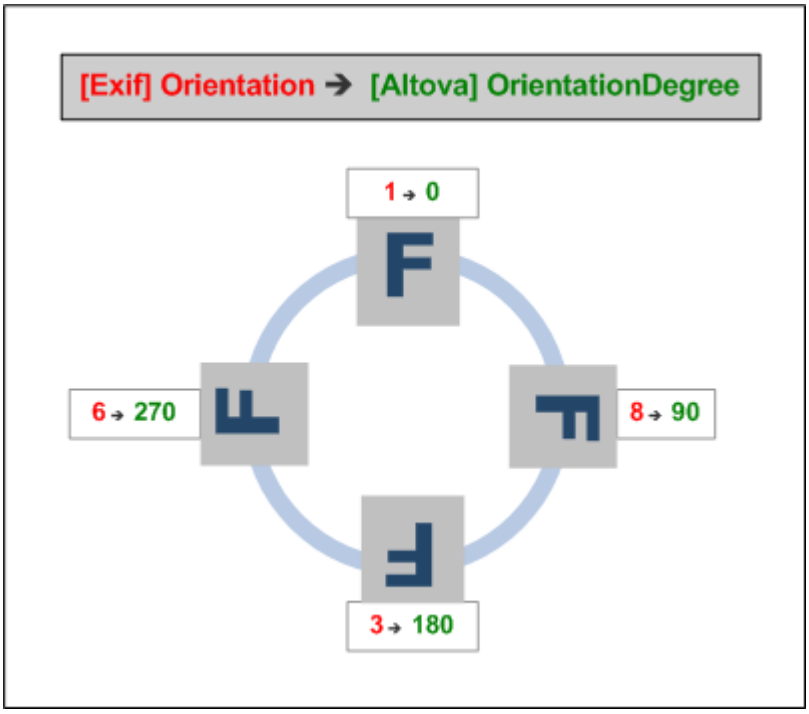
サンプル

- **mt-transform-image**(Images/Image[@id='43'], '50%', 90, 75)
  関数は、43 の@id 値を持つイメージ子孫ノード内でBase64-エンコード 文字列として保管されているイメージを入力とします。関数は変換されたイメージを返します。変換されたイメージは50% まで、サイズ変更され、時計回りに90 度回転され、75 のクオリティレベルを与えられます。
- **mt-transform-image**(Images/Image[@id='43'], 400, 90, 75)
  関数は前のサンプルと同じ結果を出しますが、長い辺は400 ピクセルの特定の値に設定されています。元のイメージのアスペクト率は保持されます。
- **mt-transform-image**(Images/Image[@id='43'], (400, 280), image-exif-data($XML1/$XML1/Images/ReferenceImage)/@OrientationDegree, 75)
  このサンプルは、前のサンプルと同じイメージを選択し、同じクオリティ値(75) を設定します。イメージのサイズは400x280 ピクセルに設定されています。回転値は、ノード内のBase64-エンコード イメージの**@OrientationDegree** 属性から得られます。

Altova Exif 属性: OrientationDegree

Altova XPath/XQuery エンジンはカスタム属性 **OrientationDegree** をExif メタデータタグ **Orientation** から生成します。

OrientationDegree は標準 Exif タグ Orientation を正数の値(1、8、3 または6) からそれぞれ対応する度数の値(0, 90, 180, 270) へ下の図に示されているように変換します。2、4、5、7 のOrientation 値の変換はないことに注意してください。(これらの向きはイメージ1 を垂直方向中央軸で反転して、2 の値を持つイメージを取得します。そして、このイメージを90-度ごとに時計回りにジャンプさせそれぞれ7、4、および5 の値を取得します。)。



標準 Exif メタタグ

- ImageWidth
- ImageLength
- BitsPerSample
- Compression
- PhotometricInterpretation
- Orientation
- SamplesPerPixel
- PlanarConfiguration
- YCbCrSubSampling
- YCbCrPositioning
- XResolution
- YResolution
- ResolutionUnit
- StripOffsets
- RowsPerStrip
- StripByteCounts
- JPEGInterchangeFormat
- JPEGInterchangeFormatLength
- TransferFunction
- WhitePoint
- PrimaryChromaticities
- YCbCrCoefficients
- ReferenceBlackWhite
- DateTime
- ImageDescription

(C) 2015-2021 Altova GmbH

- Make
- Model
- Software
- Artist
- Copyright

- ExifVersion
- FlashpixVersion
- ColorSpace
- ComponentsConfiguration
- CompressedBitsPerPixel
- PixelXDimension
- PixelYDimension
- MakerNote
- UserComment
- RelatedSoundFile
- DateTimeOriginal
- DateTimeDigitized
- SubSecTime
- SubSecTimeOriginal
- SubSecTimeDigitized
- ExposureTime
- FNumber
- ExposureProgram
- SpectralSensitivity
- ISOSpeedRatings
- OECF
- ShutterSpeedValue
- ApertureValue
- BrightnessValue
- ExposureBiasValue
- MaxApertureValue
- SubjectDistance
- MeteringMode
- LightSource
- Flash
- FocalLength
- SubjectArea
- FlashEnergy
- SpatialFrequencyResponse
- FocalPlaneXResolution
- FocalPlaneYResolution
- FocalPlaneResolutionUnit
- SubjectLocation
- ExposureIndex
- SensingMethod
- FileSource
- SceneType
- CFAPattern
- CustomRendered
- ExposureMode
- WhiteBalance
- DigitalZoomRatio

- FocalLengthIn35mmFilm
- SceneCaptureType
- GainControl
- Contrast
- Saturation
- Sharpness
- DeviceSettingDescription
- SubjectDistanceRange
- ImageUniqueID

- GPSVersionID
- GPSLatitudeRef
- GPSLatitude
- GPSLongitudeRef
- GPSLongitude
- GPSAltitudeRef
- GPSAltitude
- GPSTimeStamp
- GPSSatellites
- GPSStatus
- GPSMeasureMode
- GPSDOP
- GPSSpeedRef
- GPSSpeed
- GPSTrackRef
- GPSTrack
- GPSImgDirectionRef
- GPSImgDirection
- GPSMapDatum
- GPSDestLatitudeRef
- GPSDestLatitude
- GPSDestLongitudeRef
- GPSDestLongitude
- GPSDestBearingRef
- GPSDestBearing
- GPSDestDistanceRef
- GPSDestDistance
- GPSProcessingMethod
- GPSAreaInformation
- GPSDateStamp
- GPSDifferential

▼ image-exif-data [altova:]

altova:image-exif-data(Base64BinaryString *as string*) を **element?** とする **XP3.1** **XQ3.1**
Base64 エンコード JPEG イメージを引数として、イメージの Exif メタデータを含む **Exif** という名の要素を返します。Exif メタデータは `Exif` 要素の属性の値ペアとして作成されます。属性名は、Base64 エンコード内で検出された Exif データタグです。Exif 仕様タグのリストは以下の通りです。ベンダー特有のタグが Exif データ内に存在する場合、タグとその値も属性値のペアとして返されます。標準 Exif メタデータタグに追加して（*下のリスト参照*）Altova 特有の属性値のペアもまた生成されます。これらの Altova Exif 属性は以下のとおりです。

(C) 2015-2021 Altova GmbH

サンプル

- 属性にアクセスするには、以下の関数を使用します:
  ```
  image-exif-data(//MyImages/Image20141130.01)/@GPSLatitude
  image-exif-data(//MyImages/Image20141130.01)/@Geolocation
  ```
- 全ての属性にアクセスするには、以下の関数を使用します:
  ```
  image-exif-data(//MyImages/Image20141130.01)/@*
  ```
- 全ての属性の名前にアクセスするには、以下の式を使用します:
  ```
  for $i in image-exif-data(//MyImages/Image20141130.01)/@* return name($i)
  ```
  関数により返される属性の名前を検出するために役に立ちます。

Altova Exif 属性: 位置情報

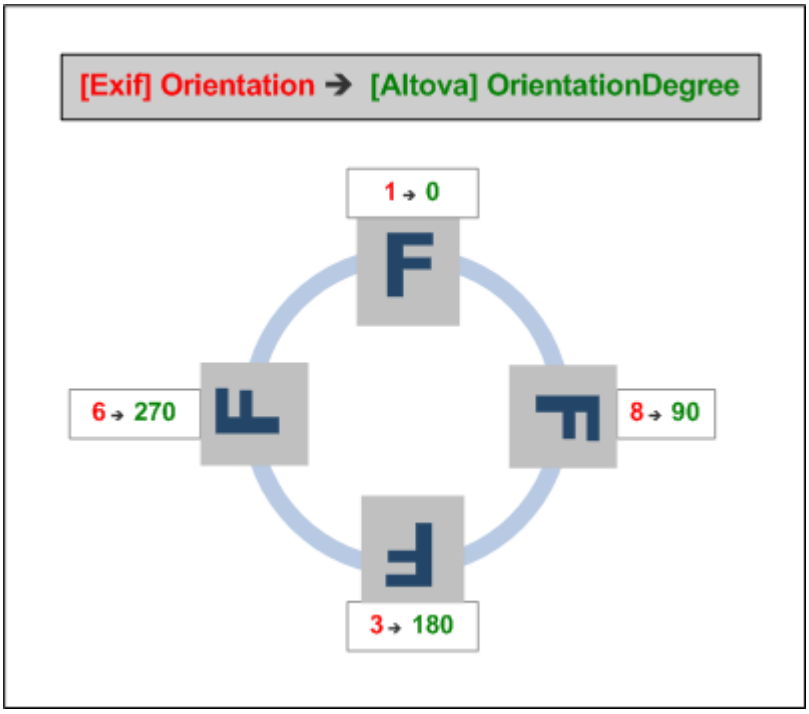
Altova XPath/XQuery エンジンはカスタム属性 **Geolocation**を標準 Exif メタデータ タグから生成します。**Geolocation** は、4 つのExif タグの連結です: 単位の追加された(*下のテーブル参照*) GPSLatitude、GPSLatitudeRef、GPSLongitude、GPSLongitudeRef 。

GPSLatitude	GPSLatitudeRef	GPSLongitude	GPSLongitudeRef	Geolocation
33 51 21.91	S	151 13 11.73	E	33°51'21.91"S 151° 13'11.73"E

Altova Exif 属性: OrientationDegree

Altova XPath/XQuery エンジンはカスタム属性 **OrientationDegree** をExif メタデータタグ **Orientation** から生成します。

OrientationDegree は標準 Exif タグ Orientation を 正数の値 (1、8、3 または 6) からそれぞれ対応する度数の値 (0, 90, 180, 270) へ下の図に示されているように変換します。2、4、5、7 のOrientation 値の変換はないことに注意してください。(これらの向きはイメージ 1 を垂直方向中央軸で反転して、2 の値を持つイメージを取得します。そして、このイメージを 90-度ごとに時計回りにジャンプさせそれぞれ 7、4、および 5 の値を取得します。)。



標準 Exif メタデータ タグのリスト

- ImageWidth
- ImageLength
- BitsPerSample
- Compression
- PhotometricInterpretation
- Orientation
- SamplesPerPixel
- PlanarConfiguration
- YCbCrSubSampling
- YCbCrPositioning
- XResolution
- YResolution
- ResolutionUnit
- StripOffsets
- RowsPerStrip
- StripByteCounts
- JPEGInterchangeFormat
- JPEGInterchangeFormatLength
- TransferFunction
- WhitePoint
- PrimaryChromaticities
- YCbCrCoefficients
- ReferenceBlackWhite
- DateTime
- ImageDescription

(C) 2015-2021 Altova GmbH

- Make
- Model
- Software
- Artist
- Copyright

- ExifVersion
- FlashpixVersion
- ColorSpace
- ComponentsConfiguration
- CompressedBitsPerPixel
- PixelXDimension
- PixelYDimension
- MakerNote
- UserComment
- RelatedSoundFile
- DateTimeOriginal
- DateTimeDigitized
- SubSecTime
- SubSecTimeOriginal
- SubSecTimeDigitized
- ExposureTime
- FNumber
- ExposureProgram
- SpectralSensitivity
- ISOSpeedRatings
- OECF
- ShutterSpeedValue
- ApertureValue
- BrightnessValue
- ExposureBiasValue
- MaxApertureValue
- SubjectDistance
- MeteringMode
- LightSource
- Flash
- FocalLength
- SubjectArea
- FlashEnergy
- SpatialFrequencyResponse
- FocalPlaneXResolution
- FocalPlaneYResolution
- FocalPlaneResolutionUnit
- SubjectLocation
- ExposureIndex
- SensingMethod
- FileSource
- SceneType
- CFAPattern
- CustomRendered
- ExposureMode
- WhiteBalance
- DigitalZoomRatio

- FocalLengthIn35mmFilm
- SceneCaptureType
- GainControl
- Contrast
- Saturation
- Sharpness
- DeviceSettingDescription
- SubjectDistanceRange
- ImageUniqueID

- GPSVersionID
- GPSLatitudeRef
- GPSLatitude
- GPSLongitudeRef
- GPSLongitude
- GPSAltitudeRef
- GPSAltitude
- GPSTimeStamp
- GPSSatellites
- GPSStatus
- GPSMeasureMode
- GPSDOP
- GPSSpeedRef
- GPSSpeed
- GPSTrackRef
- GPSTrack
- GPSImgDirectionRef
- GPSImgDirection
- GPSMapDatum
- GPSDestLatitudeRef
- GPSDestLatitude
- GPSDestLongitudeRef
- GPSDestLongitude
- GPSDestBearingRef
- GPSDestBearing
- GPSDestDistanceRef
- GPSDestDistance
- GPSProcessingMethod
- GPSAreaInformation
- GPSDateStamp
- GPSDifferential

[トップ]

19.1.2.1.5 XPath/XQuery 関数: 数値

Altova の数値拡張関数はXPath とXQuery 内で使用することができ、データを更に処理するための追加機能を提供します。このセクションの関数はAltova の**XPath 3.0** と**XQuery 3.0** エンジンと使用することができます。XPath/XQuery コンテキストで使用することができます。

(C) 2015-2021 Altova GmbH

関数の名前指定と言語の適用性に関するメモ

Altova 拡張関数はXPath/XQuery 式で使用することができ、XPath、XQuery、およびXSLT 関数の標準ライブラリで使用可能な機能に更なる機能性を与えます。Altova 拡張関数はAltova 拡張関数名前空間、**http://www.altova.com/xslt-extensions** に収められており、**altova:** プレフィックスが、このセクションでは使用されます。製品の今後のバージョンが拡張機能への継続的サポート、または個別の関数の振る舞いは変更する可能性があることご注意してください。Altova 拡張機能へのサポートに関しては、今後のリリースのドキュメントを参照してください。

XPath 関数 (XSLT 内のXPath 式で使用):	XP1 XP2 XP3.1
XSLT 関数 (XSLT 内のXPath 式で使用):	XSLT1 XSLT2 XSLT3
XQuery 関数 (XQuery 内のXQuery 式で使用):	XQ1 XQ3.1

自動付番関数

▼ generate-auto-number [altova:]

altova:generate-auto-number(ID *as xs:string*, **StartsWith** *as xs:double*, **Increment** *as xs:double*, **ResetOnChange** *as xs:string*) を xs:integer とする XP1 XP2 XQ1 XP3.1 XQ3.1
関数が呼び出される度に番号を生成します。関数が初めて呼び出される際に生成される最初の番号はStartsWith 引数により指定されます。その後の関数の呼び出しは新しい番号を生成します。この番号は前に生成された番号を、Increment 引数で指定された値ごとにインクリメントします。実質的には、altova:generate-auto-number 関数は、ID 引数により指定されたカウンターを作成し、このカウンターが関数が呼び出されるごとにインクリメントされます。ResetOnChange 引数の値が、前の関数呼び出しから変更されると、生成される番号の値がStartsWith 値にリセットされます。自動付番はaltova:reset-auto-number 関数を使用してリセットすることができます。

▣ *サンプル*

- **altova:generate-auto-number**("ChapterNumber", 1, 1, "SomeString") は、関数が呼び出される度に番号を1 つ返します。1 から始まり、関数が呼び出される度に1 ずつインクリメントします。その後の呼び出しの4 番目の"SomeString"引数がである限り、インクリメントは継続されます。4 番目の引数の値が変更されると、(ChapterNumber と呼ばれる) カウンターは1 にリセットされます。ChapterNumber の値もaltova:reset-auto-number 関数の呼び出しによりのaltova:reset-auto-number("ChapterNumber")ようにリセットされます。

▼ reset-auto-number [altova:]

altova:reset-auto-number(ID *as xs:string*) XP1 XP2 XQ1 XP3.1 XQ3.1
この関数は、ID 引数内で名づけられた自動付番カウンターの番号をリセットします。 引数内のカウンター名を作成した関数の引数により指定された数値にリセットされます。altova:generate-auto-number

▣ *サンプル*

- **altova:reset-auto-number**("ChapterNumber") は、altova:generate-auto-number 関数により作成されたChapterNumber という名の自動付番カウンターをリセットします。
- ChapterNumber を作成したaltova:generate-auto-number 関数のStartsWith 引数の値に数値をリセットします。

[[トップ]]

数値関数

▼ hex-string-to-integer [altova:]

altova:hex-string-to-integer(HexString *as xs:string*) を**xs:integer** とする **XP3.1** **XQ3.1**
10 進のシステム(10進)内の正数の16 進の同値の文字列引数を必要とし、10 進の整数を返します。

サンプル

- **altova:hex-string-to-integer**('1') は1 を返します。
- **altova:hex-string-to-integer**('9') は9 を返します。
- **altova:hex-string-to-integer**('A') は10 を返します。
- **altova:hex-string-to-integer**('B') は11 を返します。
- **altova:hex-string-to-integer**('F') は15 を返します。
- **altova:hex-string-to-integer**('G') はエラーを返します。
- **altova:hex-string-to-integer**('10') は16 を返します。
- **altova:hex-string-to-integer**('01') は1 を返します。
- **altova:hex-string-to-integer**('20') は32 を返します。
- **altova:hex-string-to-integer**('21') は33 を返します。
- **altova:hex-string-to-integer**('5A') は90 を返します。
- **altova:hex-string-to-integer**('USA') はエラーを返します。

integer-to-hex-string [altova:]

altova:integer-to-hex-string(Integer *as xs:integer*) を**xs:string** とする **XP3.1** **XQ3.1**
正数を引数として必要とし、文字列として自身のベース16 の同値を返します。

サンプル

- **altova:integer-to-hex-string**(1) は'1' を返します。
- **altova:integer-to-hex-string**(9) は'9' を返します。
- **altova:integer-to-hex-string**(10) は'A' を返します。
- **altova:integer-to-hex-string**(11) は'B' を返します。
- **altova:integer-to-hex-string**(15) は'F' を返します。
- **altova:integer-to-hex-string**(16) は'10' を返します。
- **altova:integer-to-hex-string**(32) は'20' を返します。
- **altova:integer-to-hex-string**(33) は'21' を返します。
- **altova:integer-to-hex-string**(90) は'5A' を返します。

[[トップ]]

数値フォーマット関数

[[トップ]]

19.1.2.1.6 XPath/XQuery 関数: スキーマ

下にリストされるAltova 拡張関数はスキーマの情報を返します。以下は関数の詳細 (i) サンプル(ii) スキーマコンポーネントのリストとパースペクティブのプロパティです。 Altova の**XPath 3.0** と**XQuery 3.0** エンジンと共に使用することができXPath/XQuery コンテキスト内で見つけることができます。

(C) 2015-2021 Altova GmbH

スキーマドキュメントからのスキーマ情報
関数 **altova:schema** には以下の2つの引数が存在します: 1つのゼロ引数と他の2つの引数。ゼロ引数関数はスキーマ全体を返します。この開始点からスキーマ内をナビゲートし必要とするスキーマコンポーネントをロケートします。2-引数関数は自身のQName により識別される特定のコンポーネントの型を返します。 両方の場合、戻り値は関数です。返されたコンポーネント内をナビゲートするには、特定のコンポーネントのプロパティを選択する必要があります。プロパティが非動的アイテム(すなわちコンポーネントの場合) このコンポーネントのプロパティをさらに選択してナビゲートすることが可能です。選択されたプロパティが動的なアイテムの場合、アイテムの値が返され、ナビゲートを更に行うことはできません。

メモ Xquery 式ではスキーマは明示的にインポートされる必要があります。XPath 式では、スキーマは処理環境にインポートされる必要があります。例えば、XSLT には **xslt:import** 命令を使用してインポートします。

XML ノードからのスキーマ情報
関数 **altova:type** はXMLドキュメントのノードを送信しノードの型情報をPSVI から返します。

関数の名前指定と言語の適用性に関するメモ

Altova 拡張関数はXPath/XQuery 式で使用することができ、XPath、XQuery、およびXSLT 関数の標準ライブラリで使用可能な機能に更なる機能性を与えます。Altova 拡張関数はAltova 拡張関数名前空間、**http://www.altova.com/xslt-extensions** に収められており、**altova:** プレフィックスが、このセクションでは使用されます。製品の今後のバージョンが拡張機能への継続的サポート、または個別の関数の振る舞いは変更する可能性があることご注意してください。Altova 拡張機能へのサポートに関しては、今後のリリースのドキュメントを参照してください。

XPath 関数 (XSLT 内のXPath 式で使用):	XP1 XP2 XP3.1
XSLT 関数 (XSLT 内のXPath 式で使用):	XSLT1 XSLT2 XSLT3
XQuery 関数 (XQuery 内のXQuery 式で使用):	XQ1 XQ3.1

▼ Schema (引数無し)

altova:schema() as (function(xs:string) as item()*)? XP3.1 XQ3.1
schema コンポーネント全体を返します。**schema** コンポーネントを **schema** コンポーネントのプロパティの一つを選択して更にナビゲートすることができます。

- このプロパティがコンポーネントの場合、このコンポーネントのプロパティの1つを選択して更に深く他のステップをナビゲートすることができます。このステップは更にスキーマをナビゲートするために繰り返すことができます。
- コンポーネントが動的な値の場合、動的な値が返され更にナビゲートすることはできません。

schema コンポーネントのプロパティ:

```
"type definitions"
"attribute declarations"
"element declarations"
"attribute group definitions"
"model group definitions"
"notation declarations"
"identity-constraint definitions"
```

(**schema** 以外の) 他のすべてのコンポーネント型のプロパティが下にリストされています。

メモ Xquery 式ではスキーマは明示的にインポートされる必要があります。XPath 式では、スキーマは処理環境にインポートされる必要があります。例えば、XSLT には **xslt:import** 命令を使用してインポートします。

▣ *サンプル*

- `import schema "" at "C:\Test\ExpReport.xsd"; for $typedef in altova:schema() ("type definitions") return $typedef ("name")` はスキーマ内のすべての単純型または複合型の名前を返します
- `import schema "" at "C:\Test\ExpReport.xsd"; altova:schema() ("type definitions")[1]("name")` はスキーマ内の単純型または複合型の最初の名前を返します

コンポーネントとそのプロパティ

Assertion

プロパティ名	プロパティ型	プロパティ値
kind	文字列	"Assertion"
test	XPath プロパティレコード	

Attribute Declaration

プロパティ名	プロパティ型	プロパティ値
kind	文字列	"Attribute Declaration"
name	文字列	属性のローカル名
target namespace	文字列	属性の名前空間 URI
type definition	Simple Type または Complex Type	
scope	プロパティを持つ関数 ("class":"Scope", "variety": "global" or "local", "parent": the containing Complex Type or Attribute Group)	
value constraint	存在する場合、プロパティを持つ関数 ("class": "Value Constraint"、"variety": "fixed" または "default", "value": atomic value, "lexical form": string。"value" プロパティは namespace-sensitive 型のために使用することはできないことにご注意してください。	
inheritable	ブール値	

Attribute Group Declaration

プロパティ名	プロパティ型	プロパティ値
kind	文字列	"Attribute Group Definition"
name	文字列	属性グループのローカル名
target namespace	文字列	属性グループの名前空間 URI
attribute uses	(Attribute Use) のシーケンス	
attribute wildcard	任意の属性のワイルドカード	

Attribute Use

(C) 2015-2021 Altova GmbH

プロパティ名	プロパティ型	プロパティ値
kind	文字列	"Attribute Use"
required	ブール値	属性が必要な場合はtrue 、任意の場合はfalse
value constraint	Attribute Declaration を参照	
inheritable	ブール値	

Attribute Wildcard

プロパティ名	プロパティ型	プロパティ値
kind	文字列	"Wildcard"
namespace constraint	プロパティを持つ関数 (″class″: ″Namespace Constraint", "variety": "any"|"enumeration"|"not", "namespaces": sequence of xs:anyURI, "disallowed names": list containing QNames and/or the strings "defined" and "definedSiblings"	
process contents	文字列 (″strict″|″lax″|″skip″)	

Complex Type

プロパティ名	プロパティ型	プロパティ値
kind	文字列	"Complex Type"
name	文字列	(匿名の場合は空)型のローカル名
target namespace	文字列	(匿名の場合は空)型の名前空間 URI
base type definition	Complex Type Definition	
final	文字列のシーケンス ("restriction"|"extension")	
context	空のシーケンス(not implemented)	
derivation method	文字列 (″restriction″|″extension″)	
abstract	ブール値	
attribute uses	Attribute Use のシーケンス	
attribute wildcard	任意の属性のワイルドカード	
content type	プロパティを持つ関数: (″class″:″Content Type", "variety":string ("element-only"|"empty"|"mixed"|"simple"), particle: optional Particle, "open content": function with properties ("class":"Open Content", "mode": string ("interleave"|"suffix"), "wildcard": Wildcard), "simple type definition": Simple Type)	
prohibited	文字列のシーケンス	

substitutions	("restriction"|"extension")	
assertions	アサーションのシーケンス	

Element Declaration

プロパティ名	プロパティ型	プロパティ値
kind	文字列	"Complex Type"
name	文字列	(匿名の場合は空)型のローカル名
target namespace	文字列	(匿名の場合は空)型の名前空間 URI
type definition	Simple Type または Complex Type	
type table	プロパティを持つ関数 (″class″:″Type Table″, "alternatives": sequence of Type Alternative, "default type definition": Simple Type or Complex Type)	
scope	プロパティを持つ関数 (″class″:″Scope″, "variety": ("global"|"local"), "parent": optional Complex Type)	
value constraint	Attribute Declaration を参照	
nillable	ブール値	
identity-constraint definitions	Identity Constraint のシーケンス	
substitution group affiliations	Element Declaration のシーケンス	
substitution group exclusions	文字列のシーケンス ("restriction"|"extension")	
disallowed substitutions	文字列のシーケンス ("restriction"|"extension"|"substitution")	
abstract	ブール値	

Element Wildcard

プロパティ名	プロパティ型	プロパティ値
kind	文字列	"Wildcard"
namespace constraint	プロパティを持つ関数 (″class″: ″Namespace Constraint", "variety": "any"|"enumeration"|"not", "namespaces": sequence of xs:anyURI, "disallowed names": list containing QNames and/or the strings "defined" and "definedSiblings"	
process contents	文字列 (″strict″|″lax″|″skip″)	

Facet

プロパティ名	プロパティ型	プロパティ値

(C) 2015-2021 Altova GmbH

kind	文字列	ファセットの名前、例えば、"minLength" または "enumeration"
value	ファセットによる	ファセットの値
fixed	ブール値	
typed-value	ファセットの列挙のみ、Array(xs:anyAtomicType	列挙値を含む配列。それぞれがアトミック値のシーケンスである場合があります。(メモ 列挙ファセットに関しては、実際の型にかかわらず "value" 値プロパティは文字列のシーケンスです)

Identity Constraint

プロパティ名	プロパティ型	プロパティ値
kind	文字列	"Identity-Constraint Definition"
name	文字列	制約のローカル名
target namespace	文字列	制約の名前空間 URI
identity-constraint category	文字列 ("key"|"unique"|"keyRef")	
selector	XPath プロパティレコード	
fields	XPath プロパティレコード のシーケンス	
referenced key	(keyRef のためのみ): Identity Constraint	対応するキー制約

Model Group

プロパティ名	プロパティ型	プロパティ値
kind	文字列	"Model Group"
compositor	文字列 ("sequence"|"choice"|"all")	
particles	Particle のシーケンス	

Model Group Definition

プロパティ名	プロパティ型	プロパティ値
kind	文字列	"Model Group Definition"
name	文字列	モデルグループのローカル名
target namespace	文字列	モデルグループの名前空間 URI
model group	Model Group	

Notation

プロパティ名	プロパティ型	プロパティ値
kind	文字列	"Notation Declaration"
name	文字列	表記のローカル名

target namespace	文字列	表記の名前空間 URI
system identifier	anyURI	
public identifier	文字列	

Particle

プロパティ名	プロパティ型	プロパティ値
kind	文字列	"Particle"
min occurs	整数	
max occurs	整数または文字列("unbounded")	
term	Element Declaration、Element Wildcard、または ModelGroup	

Simple Type

プロパティ名	プロパティ型	プロパティ値
kind	文字列	"Simple Type Definition"
name	文字列	(匿名の場合は空)型のローカル名
target namespace	文字列	(匿名の場合は空)型の名前空間 URI
final	文字列のシーケンス ("restriction"|"extension"|"list"|"union")	
context	含まれるコンポーネント	
base type definition	Simple Type	
facets	Sequence of Facet	
fundamental facets	(実装されていない)空のシーケンス	
variety	文字列 ("atomic"|"list"|"union")	
primitive type definition	Simple Type	
item type definition	(リスト型のみのための) Simple Type	
member type definitions	(ユニオン型のための) Simple Type のシーケンス	

Type Alternative

プロパティ名	プロパティ型	プロパティ値
kind	文字列	"Type Alternative"
test	XPath プロパティレコード	
type definition	Simple Type または Complex Type	

XPath Property Record

プロパティ名	プロパティ型	プロパティ値

(C) 2015-2021 Altova GmbH

namespace bindings	プロパティを持つ関数のシーケンス("prefix": string, "namespace": anyURI)	
default namespace	anyURI	
base URI	anyURI	XPath 式の静的ベースURI
expression	文字列	文字列としてのXPath 式

▼ Schema (2つの引数)

altova:schema(ComponentKind *as xs:string*, **Name** *as xs:QName*) **as (function(xs:string) as item()*)?** **XP3.1** **XQ3.1**

2番目の引数内で与えられている名前と同じ名前を持つ最初の引数内で指定されているコンポーネントの型を返します。コンポーネントのプロパティの一つを選択して更にナビゲートすることができます。

- このプロパティがコンポーネントの場合、このコンポーネントのプロパティの1つを選択して更に深く他のステップをナビゲートすることができます。このステップは更にスキーマをナビゲートするために繰り返すことができます。
- コンポーネントが動的な値の場合、動的な値が返され更にナビゲートすることはできません。

メモ Xquery 式ではスキーマは明示的にインポートされる必要があります。XPath 式では、スキーマは処理環境にインポートされる必要があります。例えば、XSLT には**xslt:import** 命令を使用してインポートします。

▣ *サンプル*

- ```
  import schema "" at "C:\Test\ExpReport.xsd";
  altova:schema("element declaration", xs:QName("OrgChart"))("type definition")
  ("content type")("particles")[3]!.("term")("kind")
  ```
  は3番目の`particles` コンポーネントの `kind` プロパティを返します。 この`particles` コンポーネントは`OrgChart` の `Qname` を持つ要素宣言の子孫です。

- ```
  import schema "" at "C:\Test\ExpReport.xsd";
  let $typedef := altova:schema("type definition", xs:QName("emailType"))
  for $facet in $typedef ("facets")
  return [$facet ("kind"), $facet("value")]
  ```
  は各 **emailType** コンポーネントの各 **facet** にそのファセットの型と値を含む列挙を返します。

コンポーネントとそのプロパティ

▣ Assertion

プロパティ名	プロパティ型	プロパティ値
kind	文字列	"Assertion"
test	XPath プロパティレコード	

▣ Attribute Declaration

プロパティ名	プロパティ型	プロパティ値
kind	文字列	"Attribute Declaration"
name	文字列	属性のローカル名

target namespace	文字列	属性の名前空間 URI
type definition	Simple Type または Complex Type	
scope	プロパティを持つ関数 (″class″:″Scope″, "variety": "global" or "local", "parent": the containing Complex Type or Attribute Group)	
value constraint	存在する場合、プロパティを持つ関数 (″class″: ″Value Constraint″、″variety″: ″fixed″ または ″default″, ″value″: atomic value, ″lexical form″: string。″value″ プロパティは namespace-sensitive 型のために使用することはできないことにご注意してください。	
inheritable	ブール値	

Attribute Group Declaration

プロパティ名	プロパティ型	プロパティ値
kind	文字列	"Attribute Group Definition"
name	文字列	属性グループのローカル名
target namespace	文字列	属性グループの名前空間 URI
attribute uses	(Attribute Use) のシーケンス	
attribute wildcard	任意の属性のワイルドカード	

Attribute Use

プロパティ名	プロパティ型	プロパティ値
kind	文字列	"Attribute Use"
required	ブール値	属性が必要な場合は true 、任意の場合は false
value constraint	Attribute Declaration を参照	
inheritable	ブール値	

Attribute Wildcard

プロパティ名	プロパティ型	プロパティ値
kind	文字列	"Wildcard"
namespace constraint	プロパティを持つ関数 (″class″: ″Namespace Constraint", "variety": "any"|"enumeration"|"not", "namespaces": sequence of xs:anyURI, "disallowed names": list containing QNames and/or the strings "defined" and "definedSiblings"	
process contents	文字列 (″strict″|″lax″|″skip″)	

(C) 2015-2021 Altova GmbH

- Complex Type

プロパティ名	プロパティ型	プロパティ値
kind	文字列	"Complex Type"
name	文字列	(匿名の場合は空)型のローカル名
target namespace	文字列	(匿名の場合は空)型の名前空間 URI
base type definition	Complex Type Definition	
final	文字列のシーケンス ("restriction"|"extension")	
context	空のシーケンス(not implemented)	
derivation method	文字列 ("restriction"|"extension")	
abstract	ブール値	
attribute uses	Attribute Use のシーケンス	
attribute wildcard	任意の属性のワイルドカード	
content type	プロパティを持つ関数: ("class":"Content Type", "variety":string ("element-only"|"empty"|"mixed"|"simple"), particle: optional Particle, "open content": function with properties ("class":"Open Content", "mode": string ("interleave"|"suffix"), "wildcard": Wildcard), "simple type definition": Simple Type)	
prohibited substitutions	文字列のシーケンス ("restriction"|"extension")	
assertions	アサーションのシーケンス	

- Element Declaration

プロパティ名	プロパティ型	プロパティ値
kind	文字列	"Complex Type"
name	文字列	(匿名の場合は空)型のローカル名
target namespace	文字列	(匿名の場合は空)型の名前空間 URI
type definition	Simple Type または Complex Type	
type table	プロパティを持つ関数 ("class":"Type Table", "alternatives": sequence of Type Alternative, "default type definition": Simple Type or Complex Type)	
scope	プロパティを持つ関数 ("class":"Scope", "variety": ("global"|"local"), "parent": optional Complex Type)	
value constraint	Attribute Declaration を参照	
nillable	ブール値	
identity-constraint	Identity Constraint のシーケンス	

definitions		
substitution group affiliations	Element Declaration のシーケンス	
substitution group exclusions	文字列のシーケンス ("restriction"|"extension")	
disallowed substitutions	文字列のシーケンス ("restriction"|"extension"|"substitution")	
abstract	ブール値	

Element Wildcard

プロパティ名	プロパティ型	プロパティ値
kind	文字列	"Wildcard"
namespace constraint	プロパティを持つ関数 ("class": "Namespace Constraint", "variety": "any"|"enumeration"|"not", "namespaces": sequence of xs:anyURI, "disallowed names": list containing QNames and/or the strings "defined" and "definedSiblings"	
process contents	文字列 ("strict"|"lax"|"skip")	

Facet

プロパティ名	プロパティ型	プロパティ値
kind	文字列	ファセットの名前、例えば、"minLength" または"enumeration"
value	ファセットによる	ファセットの値
fixed	ブール値	
typed-value	ファセットの列挙のみ、Array(xs:anyAtomicType	列挙値を含む配列。それぞれがアトミック値のシーケンスである場合があります。(メモ 列挙ファセットに関しては、実際の型にかかわらず "value" 値プロパティは文字列のシーケンスです)

Identity Constraint

プロパティ名	プロパティ型	プロパティ値
kind	文字列	"Identity-Constraint Definition"
name	文字列	制約のローカル名
target namespace	文字列	制約の名前空間 URI
identity-constraint category	文字列 ("key"|"unique"|"keyRef")	
selector	XPath プロパティレコード	
fields	XPath プロパティレコード のシーケンス	

(C) 2015-2021 Altova GmbH

referenced key	(keyRef のためのみ): Identity Constraint	対応するキー制約

Model Group

プロパティ名	プロパティ型	プロパティ値
kind	文字列	"Model Group"
compositor	文字列 ("sequence"|"choice"|"all")	
particles	Particle のシーケンス	

Model Group Definition

プロパティ名	プロパティ型	プロパティ値
kind	文字列	"Model Group Definition"
name	文字列	モデルグループのローカル名
target namespace	文字列	モデルグループの名前空間 URI
model group	Model Group	

Notation

プロパティ名	プロパティ型	プロパティ値
kind	文字列	"Notation Declaration"
name	文字列	表記のローカル名
target namespace	文字列	表記の名前空間 URI
system identifier	anyURI	
public identifier	文字列	

Particle

プロパティ名	プロパティ型	プロパティ値
kind	文字列	"Particle"
min occurs	整数	
max occurs	整数または文字列("unbounded")	
term	Element Declaration、Element Wildcard、または ModelGroup	

Simple Type

プロパティ名	プロパティ型	プロパティ値
kind	文字列	"Simple Type Definition"
name	文字列	(匿名の場合は空) 型のローカル名
target namespace	文字列	(匿名の場合は空) 型の名前空間 URI
final	文字列のシーケンス	

	("restriction"|"extension"|"list"|"union")	
context	含まれるコンポーネント	
base type definition	Simple Type	
facets	Sequence of Facet	
fundamental facets	(実装されていない)空のシーケンス	
variety	文字列 ("atomic"|"list"|"union")	
primitive type definition	Simple Type	
item type definition	(リスト型のみのための) Simple Type	
member type definitions	(ユニオン型のための) Simple Type のシーケンス	

Type Alternative

プロパティ名	プロパティ型	プロパティ値
kind	文字列	"Type Alternative"
test	XPath プロパティレコード	
type definition	Simple Type または Complex Type	

XPath Property Record

プロパティ名	プロパティ型	プロパティ値
namespace bindings	プロパティを持つ関数のシーケンス("prefix": string, "namespace": anyURI)	
default namespace	anyURI	
base URI	anyURI	XPath 式の静的ベースURI
expression	文字列	文字列としてのXPath 式

型

altova:type(Node *as item?***) as (function(xs:string) as item()*)? XP3.1 XQ3.1**
関数 **altova:type** はXMLドキュメントの要素または属性ノードを送信しノードの型情報をPSVI から返します。

メモ スキーマが参照可能になるためにXMLドキュメントはスキーマ宣言を持つ必要があります。

サンプル

- ```
  for $element in //Email
  let $type := altova:type($element)
  return $type
  ```
  はノードの型情報を含む関数を返します。

- ```
  for $element in //Email
  let $type := altova:type($element)
  return $type ("kind")
  ```


(C) 2015-2021 Altova GmbH

はノードの型のコンポーネント（単純型または複合型）を取りコンポーネントの`kind` property の値を返します。

コンポーネントとそのプロパティ

- Assertion

プロパティ名	プロパティ型	プロパティ値
kind	文字列	"Assertion"
test	XPath プロパティレコード	

- Attribute Declaration

プロパティ名	プロパティ型	プロパティ値
kind	文字列	"Attribute Declaration"
name	文字列	属性のローカル名
target namespace	文字列	属性の名前空間 URI
type definition	Simple Type またはComplex Type	
scope	プロパティを持つ関数（"class":"Scope", "variety": "global" or "local", "parent": the containing Complex Type or Attribute Group)	
value constraint	存在する場合、プロパティを持つ関数（"class": "Value Constraint"、"variety": "fixed" または"default", "value": atomic value, "lexical form": string。"value" プロパティは namespace-sensitive 型のために使用することはできないことにご注意してください。	
inheritable	ブール値	

- Attribute Group Declaration

プロパティ名	プロパティ型	プロパティ値
kind	文字列	"Attribute Group Definition"
name	文字列	属性グループのローカル名
target namespace	文字列	属性グループの名前空間 URI
attribute uses	（Attribute Use）のシーケンス	
attribute wildcard	任意の属性のワイルドカード	

- Attribute Use

プロパティ名	プロパティ型	プロパティ値
kind	文字列	"Attribute Use"
required	ブール値	属性が必要な場合はtrue 、任意の場合はfalse

value constraint	Attribute Declaration を参照	
inheritable	ブール値	

- Attribute Wildcard

プロパティ名	プロパティ型	プロパティ値
kind	文字列	"Wildcard"
namespace constraint	プロパティを持つ関数 ("class": "Namespace Constraint", "variety": "any"|"enumeration"|"not", "namespaces": sequence of xs:anyURI, "disallowed names": list containing QNames and/or the strings "defined" and "definedSiblings"	
process contents	文字列 ("strict"|"lax"|"skip")	

- Complex Type

プロパティ名	プロパティ型	プロパティ値
kind	文字列	"Complex Type"
name	文字列	(匿名の場合は空)型のローカル名
target namespace	文字列	(匿名の場合は空)型の名前空間 URI
base type definition	Complex Type Definition	
final	文字列のシーケンス ("restriction"|"extension")	
context	空のシーケンス(not implemented)	
derivation method	文字列 ("restriction"|"extension")	
abstract	ブール値	
attribute uses	Attribute Use のシーケンス	
attribute wildcard	任意の属性のワイルドカード	
content type	プロパティを持つ関数: ("class":"Content Type", "variety":string ("element-only"|"empty"|"mixed"|"simple"), particle: optional Particle, "open content": function with properties ("class":"Open Content", "mode": string ("interleave"|"suffix"), "wildcard": Wildcard), "simple type definition": Simple Type)	
prohibited substitutions	文字列のシーケンス ("restriction"|"extension")	
assertions	アサーションのシーケンス	

- Element Declaration

(C) 2015-2021 Altova GmbH

プロパティ名	プロパティ型	プロパティ値
kind	文字列	"Complex Type"
name	文字列	(匿名の場合は空)型のローカル名
target namespace	文字列	(匿名の場合は空)型の名前空間 URI
type definition	Simple Type または Complex Type	
type table	プロパティを持つ関数 ("class":"Type Table", "alternatives": sequence of Type Alternative, "default type definition": Simple Type or Complex Type)	
scope	プロパティを持つ関数 ("class":"Scope", "variety": ("global"|"local"), "parent": optional Complex Type)	
value constraint	Attribute Declaration を参照	
nillable	ブール値	
identity-constraint definitions	Identity Constraint のシーケンス	
substitution group affiliations	Element Declaration のシーケンス	
substitution group exclusions	文字列のシーケンス ("restriction"|"extension")	
disallowed substitutions	文字列のシーケンス ("restriction"|"extension"|"substitution")	
abstract	ブール値	

- Element Wildcard

プロパティ名	プロパティ型	プロパティ値
kind	文字列	"Wildcard"
namespace constraint	プロパティを持つ関数 ("class": "Namespace Constraint", "variety": "any"|"enumeration"|"not", "namespaces": sequence of xs:anyURI, "disallowed names": list containing QNames and/or the strings "defined" and "definedSiblings"	
process contents	文字列 ("strict"|"lax"|"skip")	

- Facet

プロパティ名	プロパティ型	プロパティ値
kind	文字列	ファセットの名前、例えば、"minLength" または"enumeration"
value	ファセットによる	ファセットの値
fixed	ブール値	

typed-value	ファセットの列挙のみ、Array(xs:anyAtomicType	列挙値を含む配列。それぞれがアトミック値のシーケンスである場合があります。(メモ 列挙ファセットに関しては、実際の型にかかわらず "value" 値プロパティは文字列のシーケンスです)

Identity Constraint

プロパティ名	プロパティ型	プロパティ値
kind	文字列	"Identity-Constraint Definition"
name	文字列	制約のローカル名
target namespace	文字列	制約の名前空間 URI
identity-constraint category	文字列 ("key"|"unique"|"keyRef")	
selector	XPath プロパティレコード	
fields	XPath プロパティレコード のシーケンス	
referenced key	(keyRef のためのみ): Identity Constraint	対応するキー制約

Model Group

プロパティ名	プロパティ型	プロパティ値
kind	文字列	"Model Group"
compositor	文字列 ("sequence"|"choice"|"all")	
particles	Particle のシーケンス	

Model Group Definition

プロパティ名	プロパティ型	プロパティ値
kind	文字列	"Model Group Definition"
name	文字列	モデルグループのローカル名
target namespace	文字列	モデルグループの名前空間 URI
model group	Model Group	

Notation

プロパティ名	プロパティ型	プロパティ値
kind	文字列	"Notation Declaration"
name	文字列	表記のローカル名
target namespace	文字列	表記の名前空間 URI
system identifier	anyURI	
public identifier	文字列	

Particle

(C) 2015-2021 Altova GmbH

プロパティ名	プロパティ型	プロパティ値
kind	文字列	"Particle"
min occurs	整数	
max occurs	整数または文字列("unbounded")	
term	Element Declaration、Element Wildcard、またはModelGroup	

Simple Type

プロパティ名	プロパティ型	プロパティ値
kind	文字列	"Simple Type Definition"
name	文字列	(匿名の場合は空)型のローカル名
target namespace	文字列	(匿名の場合は空)型の名前空間 URI
final	文字列のシーケンス ("restriction"|"extension"|"list"|"union")	
context	含まれるコンポーネント	
base type definition	Simple Type	
facets	Sequence of Facet	
fundamental facets	(実装されていない)空のシーケンス	
variety	文字列 ("atomic"|"list"|"union")	
primitive type definition	Simple Type	
item type definition	(リスト型のみのための) Simple Type	
member type definitions	(ユニオン型のための) Simple Type のシーケンス	

Type Alternative

プロパティ名	プロパティ型	プロパティ値
kind	文字列	"Type Alternative"
test	XPath プロパティレコード	
type definition	Simple Type またはComplex Type	

XPath Property Record

プロパティ名	プロパティ型	プロパティ値
namespace bindings	プロパティを持つ関数のシーケンス("prefix": string, "namespace": anyURI)	
default namespace	anyURI	
base URI	anyURI	XPath 式の静的ベースURI
expression	文字列	文字列としてのXPath 式

19.1.2.1.7 XPath/XQuery 関数: シーケンス

Altova *のシーケンス*拡張関数はXPath とXQuery 式で使用することができ、XML スキーマの異なる日付および時刻データ型で保存されているデータを処理するための追加機能を提供します。*このセクション*の関数は、Altova の**XPath 3.0** および**XQuery 3.0** エンジンで使用することができます。これらの関数は、XPath/XQuery コンテキストで使用することができます。

関数の名前指定と言語の適用性に関するメモ

Altova 拡張関数はXPath/XQuery 式で使用することができ、XPath、XQuery、およびXSLT 関数の標準ライブラリで使用可能な機能に更なる機能性を与えます。Altova 拡張関数はAltova 拡張関数名前空間、**http://www.altova.com/xslt-extensions** に収められており、**altova:** *プレフィックス*が、このセクションでは使用されます。製品の今後のバージョンが拡張機能への継続的サポート、または個別の関数の振る舞いは変更する可能性があることにご注意してください。Altova 拡張機能へのサポートに関しては、今後のリリースのドキュメントを参照してください。

XPath 関数 (XSLT 内のXPath 式で使用):	**XP1** **XP2** **XP3.1**
XSLT 関数 (XSLT 内のXPath 式で使用):	**XSLT1** **XSLT2** **XSLT3**
XQuery 関数 (XQuery 内のXQuery 式で使用):	**XQ1** **XQ3.1**

▼ attributes [altova:]

altova:attributes(AttributeName *as xs:string***)** を**attribute()*** とする **XP3.1** **XQ3.1**
入力引数 AttributeName 内で与えられた名前と同じローカル名を持つすべての属性を返します。検索は大文字と小文字を区別し、attribute:: 軸に対して行われます。これは、コンテキストノードが親要素ノードである必要があることを意味します。

サンプル

- **altova:attributes**("MyAttribute") は MyAttribute()* を返します。

altova:attributes(AttributeName *as xs:string*, **SearchOptions** *as xs:string***)**
asattribute()* **XP3.1** **XQ3.1**
入力引数 AttributeName 内で与えられた名前と同じローカル名を持つすべての属性を返します。検索は大文字と小文字を区別し、attribute:: 軸に対して行われます。コンテキストノードが親要素ノードである必要があります。第 2 引数はオプションのフラグを含みます。使用することのできるフラグは以下の通りです:
r = 正規表現検索を切り替えます; AttributeName は正規表現検索文字列である必要があります;
f = このオプションが指定されている場合、AttributeName は完全一致を提供します。それ以外の場合は、AttributeName は属性名に部分的に一致するとその属性を返します。例えば **f** が指定されていない場合、MyAtt は MyAttribute を返します。
i = 大文字と小文字を一致させる検索に切り替えます。
p = 検索に名前空間*プレフィックス*を含みます。AttributeName は、名前空間*プレフィックス*を含みます。例えば altova:MyAttribute。
フラグは順序に関わりなく書き込むことができます。無効なフラグはエラーを生成します。1 つまたは複数のフラグを省略することができます。空の文字列は許可されていますが、引数を1 つしか持たない関数と同じ効果を持ちます(*前の署名*)、ですが、空のシーケンスは第 2 の引数としては許可されていません。

サンプル

- **altova:attributes**("MyAttribute", "rfip") は MyAttribute()* を返します。

- **altova:attributes**("MyAttribute", "pri") は MyAttribute()* を返します。
- **altova:attributes**("MyAtt", "rip") は MyAttribute()* を返します。
- **altova:attributes**("MyAttributes", "rfip") は不一致を返します。
- **altova:attributes**("MyAttribute", "") は MyAttribute()* を返します。
- **altova:attributes**("MyAttribute", "Rip") 認識されないフラグエラーを返します。
- **altova:attributes**("MyAttribute",) 見つからない第 2 引数を返します。

elements [altova:]

altova:elements(ElementName *as xs:string*) を **element()*** とする **XP3.1** **XQ3.1**
入力引数 ElementName で与えられた名前と同じローカル名を持つすべての要素を返します。検索は大文字と小文字を区別して child:: 軸に対して実行されます。コンテキストノードは、検索される要素の親ノードである必要があります。

サンプル

- **altova:elements**("MyElement") は MyElement()* を返します。

altova:elements(ElementName *as xs:string*, **SearchOptions** *as xs:string*) **aselement()*** **XP3.1** **XQ3.1**
入力引数 ElementName 内で与えられた名前と同じローカル名を持つすべての属性を返します。検索は大文字と小文字を区別し、child:: 軸に対して行われます。コンテキストノードが親要素ノードである必要があります。第 2 引数はオプションのフラグを含みます。使用することのできるフラグは以下の通りです:
r = 正規表現検索を切り替えます; ElementName は正規表現検索文字列である必要があります;
f = このオプションが指定されている場合、ElementName は完全一致を提供します。それ以外の場合は、ElementName は属性名に部分的に一致するとその属性を返します。例えば **f** が指定されていない場合、MyElem は MyElement を返します。
i = 大文字と小文字を一致させる検索に切り替えます。
p = 検索に名前空間プレフィックスを含みます。ElementName は、名前空間プレフィックスを含みます。例えば altova:MyElement。
フラグは順序に関わりなく書き込むことができます。無効なフラグはエラーを生成します。1 つまたは複数のフラグを省略することができます。空の文字列は許可されていますが、引数を 1 つしか持たない関数と同じ効果を持ちます(*前の署名*)、ですが、空のシーケンスは第 2 の引数としては許可されていません。

サンプル

- **altova:elements**("MyElement", "rip") は MyElement()* を返します。
- **altova:elements**("MyElement", "pri") は MyElement()* を返します。
- **altova:elements**("MyElement", "") は MyElement()* を返します。
- **altova:attributes**("MyElem", "rip") は MyElement()* を返します。
- **altova:attributes**("MyElements", "rfip") 不一致を課wします。
- **altova:elements**("MyElement", "Rip") 認識されないフラグエラーを返します。
- **altova:elements**("MyElement",) 見つからない第 2 引数を返します。

find-first [altova:]

altova:find-first((Sequence *as item()**), **(Condition(Sequence-Item** *as xs:boolean*))) を **item()?** とする **XP3.1** **XQ3.1**
この関数は2 つの引数を必要とします。最初の引数は1 つ、または1 つ以上のデータ型のアイテムのシーケンスです。第 2 の引数 Condition は、(1 のアリティを持つ) 1 つの引数 を必要とし、boolean を返す XPath 関数に対する参照です。Condition で参照された関数の代わりに、**Sequence** の各アイテムが提出されます。(*注意:* この関数は1 つの引数のみを必要とします。) **Condition** 内の関数に **true()** と評価させる最初の **Sequence** アイテムは、**altova:find-first**、反復の終了の結果として返されます。

サンプル

- **altova:find-first**(5 to 10, function($a) {$a mod 2 = 0}) は xs:integer 6 を返します。**Condition** 引数は、**$a** という名のインライン関数を宣言し、定義します。XPath 3.0 インライン関数 **function()** を参照します。**altova: Sequence** 引数内の各アイテムでは、**find-first** が パスされ、代わりに、**$a** を入力値とします。入力値は関数定義($a mod 2 = 0)内の条件に対してテストされます。この条件を満たす最初の入力値が **altova:find-first**（この場合は 6）の結果として返されます。

- **altova:find-first**((1 to 10), (function($a) {$a+3=7})) は xs:integer 4 を返します。

更なるサンプル
ファイル **C:\Temp\Customers.xml** が存在する場合:

- **altova:find-first**(("C:\Temp\Customers.xml", "http://www.altova.com/index.html"), (doc-available#1)) は xs:string C:\Temp\Customers.xml を返します。

ファイル **C:\Temp\Customers.xml** が存在せず、**http://www.altova.com/index.html** が存在する場合:

- **altova:find-first**(("C:\Temp\Customers.xml", "http://www.altova.com/index.html"), (doc-available#1)) は xs:string http://www.altova.com/index.html を返します。

ファイル **C:\Temp\Customers.xml** が存在せず、**http://www.altova.com/index.html** も存在しない場合:

- **altova:find-first**(("C:\Temp\Customers.xml", "http://www.altova.com/index.html"), (doc-available#1)) 結果無しを返します。

上のサンプルについての注意点

- XPath 3.0 関数 doc-available は URI として使用され、ドキュメントノードが提出された URI で検出される場合 true を返する単一の引数を必要とします。（ですから、提出された URI でのドキュメントは XML ドキュメントである必要があります。）
- doc-available 関数は、altova:find-first の第 2 引数である、**Condition** で使用することができます。これは、1 つの引数（アリティ=1）のみを必要とするからであり、item() を入力(URI として使用される文字列) として、boolean の値を返すからです。
- doc-available 関数は、参照されているだけで、呼び出されていない点に注意してください。アタッチされている #1 サフィックスは関数が 1 つのアリティであることを表示するためです。doc-available#1 の意味は以下のとおりです: アリティ=1 を持つ doc-availabe() 関数を使用し、最初のシーケンスの各アイテムの代わりに、単一引数として パスします。この結果、 2 つの文字列の各自つは、文字列をURIとして使用し、URIにドキュメントノードが存在するかテストする **doc-available()** に パスされます。1 つが各当する場合、**doc-available()** 関数は true() を評価し、シーケンス内のその文字列のインデックス ポジションは、 **altova:find-first** 関数の結果として返されます。doc-available() 関数に関しての注意点: 相対パスは、デフォルトで関数がロードされる XML ドキュメントの現在のベース URI に対して相対的に解決されます。

find-first-combination [altova:]

altova:find-first-combination((Seq-01 *as item()**), **(Seq-02** *as item()**), **(Condition(Seq-01-Item, Seq-02-Item** *as xs:boolean*)) を **item()*** とする **XP3.1** **XQ3.1**
この関数は 3 つの引数を必要とします:

- 最初の2 つの引数 Seq-01 とSeq-02, は、1 つまたは1 つ以上のデータ型のアイテムです。
- 第 3 の引数 Condition は、(2 のアリティを持つ) 2 つの引数を必要とし、boolean を返す XPath 関数に対しての参照です。

Seq-01 とSeq-02 のアイテムは指定された組み合わせ(各シーケンスからの1 つずつにアイテムで構成されるペア)で、Condition 内の関数の引数としてパスされました。組み合わせは以下のように指定されています。

```
If   Seq-01 = X1, X2, X3 ... Xn
And  Seq-02 = Y1, Y2, Y3 ... Yn
Then (X1 Y1), (X1 Y2), (X1 Y3) ... (X1 Yn), (X2 Y1), (X2 Y2) ... (Xn Yn)
```

Condition 関数にtrue() と評価するように指示する最初のペアはaltova:find-first-combination の結果として返されます。以下の点に注意してください: (i) Condition 関数が提出された引数ペア内で繰り返され、true() を評価しない場合、altova:find-first-combination は、結果を返しません; (ii) altova:find-first-combination の結果が常に(データ型の) アイテムのペアである場合またはアイテムでない場合。

サンプル

- altova:find-first-combination(11 to 20, 21 to 30, function($a, $b) {$a+$b = 32}) はxs:integers (11, 21) のシーケンスを返します。
- altova:find-first-combination(11 to 20, 21 to 30, function($a, $b) {$a+$b = 33}) はxs:integers (11, 22) のシーケンを返します。
- altova:find-first-combination(11 to 20, 21 to 30, function($a, $b) {$a+$b = 34}) は xs:integers (11, 23) のシーケンを返します。

find-first-pair [altova:]

altova:find-first-pair((Seq-01 *as item()**), (Seq-02 *as item()**), (Condition(Seq-01-Item, Seq-02-Item *as xs:boolean*)) をitem()* とする XP3.1 XQ3.1
この関数は3 つの引数を必要とします:

- 最初の2 つの引数 Seq-01 とSeq-02, は、1 つまたは1 つ以上のデータ型のアイテムです。
- 第 3 の引数 Condition は、(2 のアリティを持つ) 2 つの引数を必要とし、boolean を返す XPath 関数に対しての参照です。

Seq-01 とSeq-02 のアイテムは指定された組み合わせで、Condition 内の関数の引数としてパスされました。組み合わせは以下のように指定されています。

```
If   Seq-01 = X1, X2, X3 ... Xn
And  Seq-02 = Y1, Y2, Y3 ... Yn
Then (X1 Y1), (X2 Y2), (X3 Y3) ... (Xn Yn)
```

Condition 関数にtrue() と評価するように指示する最初のペアはaltova:find-first-pair の結果として返されます。以下の点に注意してください: (i) Condition 関数が提出された引数ペア内で繰り返され、true() を評価しない場合、altova:find-first-pair は、結果を返しません; (ii) altova:find-first-combination の結果が常に(データ型の) アイテムのペアである場合またはアイテムでない場合。

サンプル

- altova:find-first-pair(11 to 20, 21 to 30, function($a, $b) {$a+$b = 32}) は xs:integers (11, 21) のシーケンスを返します。
- altova:find-first-pair(11 to 20, 21 to 30, function($a, $b) {$a+$b = 33}) は*結果無し*を返します。

上の2 つのサンプルに表示される通り、ペアの順序は以下の通りです: (11, 21) (12, 22) (13, 23)...(20,

30) 。(33 を返す指示されたペアがないため) この理由で第 2 のサンプルは *結果無し*を返します。

find-first-pair-pos [altova:]

altova:find-first-pair-pos((Seq-01 *as item()), (Seq-02 *as item()**), (Condition(Seq-01-Item, Seq-02-Item *as xs:boolean*)) を xs:integer とする XP3.1 XQ3.1**
この関数は3 つの引数を必要とします:

- 最初の2 つの引数 **Seq-01** と**Seq-02**, は、1 つまたは1 つ以上のデータ型のアイテムです。
- 第 3 の引数 **Condition** は、(2 のアリティを持つ) 2 つの引数を必要とし、boolean を返す XPath 関数に対しての参照です。

Seq-01 と**Seq-02** のアイテムは指定された組み合わせで、**Condition** 内の関数の引数としてパスされました。組み合わせは以下のように指定されています。

```
If   Seq-01 = X1, X2, X3 ... Xn
And  Seq-02 = Y1, Y2, Y3 ... Yn
Then (X1 Y1), (X2 Y2), (X3 Y3) ... (Xn Yn)
```

Condition 関数に**true()**を評価させる、最初に指示されたペアのインデックス ポジションは、は**altova:find-first-pair-pos** の結果として返されます。関数が提出された引数ペア内で繰り返され、**true()**を一度も評価しない場合、**altova:find-first-pair-pos** *結果無し*が返します。

サンプル

- **altova:find-first-pair-pos**(11 to 20, 21 to 30, function($a, $b) {$a+$b = 32}) は 1 を返します
- **altova:find-first-pair-pos**(11 to 20, 21 to 30, function($a, $b) {$a+$b = 33}) は*結果無し*を返します。

上の2 つのサンプルに表示される通り、ペアの順序は以下の通りです: (11, 21) (12, 22) (13, 23)...(20, 30)。最初のサンプルでは、最初のペアは**Condition** 関数に**true()**を評価させ、シーケンス内のインデックス ポジションには**1** が返されます。第 2 のサンプルでは、 33 を返すペアがないため、*結果無し*が返します。

find-first-pos [altova:]

altova:find-first-pos((Sequence *as item()), (Condition(Sequence-Item *as xs:boolean*)) を xs:integer とする XP3.1 XQ3.1**
この関数は2 つの引数を必要とします。最初の引数は1 つ、または1 つ以上のデータ型のアイテムのシーケンスです。第 2 の引数 Condition は、(1 のアリティを持つ) 1 つの引数 を必要とし、boolean を返す XPath 関数に対する参照です。Condition で参照された関数の代わりに、**Sequence** の各アイテムが提出されます。(*注意:* この関数は1 つの引数のみを必要とします。) **Condition** 内の関数に**true()**と評価させる最初の **Sequence** アイテムは、**altova:find-first-pos** の結果として返された**Sequence** 内インデックス ポジションを持ちます。

サンプル

- **altova:find-first-pos**(5 to 10, function($a) {$a mod 2 = 0}) はxs:integer 2 を返します。

Condition 引数は、**$a** という名のインライン関数を宣言し、定義します。XPath 3.0 インライン関数 **function()**を参照します。 **Sequence** 引数内の各アイテムでは、**find-first-pos** がパスされ、代わりに、**$a** を入力値とします。入力値は関数定義($a mod 2 = 0)内の条件に対してテストされます。この条件を満たす最初の入力値が **altova:find-first-pos** ((シーケンス内で条件を満たす最初の値である6 がシーケンスのインデクス位置 2 にあるためこの場合は2、) の結果として返されます。

(C) 2015-2021 Altova GmbH

- **altova:find-first-pos**((2 to 10), (function($a) {$a+3=7})) は xs:integer 3 を返します。

更なるサンプル
ファイル **C:\Temp\Customers.xml** が存在する場合:

- **altova:find-first-pos**(("C:\Temp\Customers.xml", "http://www.altova.com/index.html"), (doc-available#1)) は 1 を返します。

ファイル **C:\Temp\Customers.xml** が存在せず、**http://www.altova.com/index.html** が存在する場合:

- **altova:find-first-pos**(("C:\Temp\Customers.xml", "http://www.altova.com/index.html"), (doc-available#1)) は 2 を返します。

ファイル **C:\Temp\Customers.xml** が存在せず、**http://www.altova.com/index.html** も存在しない場合:

- **altova:find-first-pos**(("C:\Temp\Customers.xml", "http://www.altova.com/index.html"), (doc-available#1)) 結果無しを返します。

上のサンプルについての注意点

- XPath 3.0 関数 doc-available は URI として使用され、ドキュメントノードが提出された URI で検出される場合 true を返する単一の引数を必要とします。(ですから、提出された URI でのドキュメントは XML ドキュメントである必要があります。)
- doc-available 関数は、altova:find-first-pos の第 2 引数である、**Condition** で使用することができます。これは、1 つの引数 (アリティ=1) のみを必要とするからであり、item() を入力(URI として使用される文字列) として、boolean の値を返すからです。
- doc-available 関数は、参照されているだけで、呼び出されていない点に注意してください。アタッチされている #1 サフィックスは関数が 1 つのアリティであることを表示するためです。doc-available#1 の意味は以下のとおりです: アリティ=1 を持つ doc-availabe() 関数を使用し、最初のシーケンスの各アイテムの代わりに単一引数として パスします。 この結果、 2 つの文字列の各自つは、文字列をURIとして使用し、URIにドキュメントノードが存在するかテストする **doc-available()** に パスされます。1 つが各当する場合、**doc-available()** 関数は true() を評価し、シーケンス内のその文字列のインデックス ポジションは、 **altova:find-first-pos** 関数の結果として返されます。 doc-available() 関数に関しての注意点: 相対パスは、デフォルトで関数がロードされる XML ドキュメントの現在のベース URI に対して相対的に解決されます。

for-each-attribute-pair [altova:]

altova:for-each-attribute-pair(Seq1 *as element()?*, **Seq2** *as element()?*, **Function** *as function()*) **asitem()*** **XP3.1** **XQ3.1**
ペアの一つの属性が最初の要素を取得し、もう一つの属性が2番目の要素から取得された箇所で、最初の2 つの引数は、2 つの要素、および、属性ペアを作成するために使用された属性を識別します。要素ペアは同じ名前を持つことをベースに選択され、ペアは名前別に アルファベット順にセットに並べ替えられます。 属性のために他の要素上に対応する属性が存在しない場合、ペアのジョイントは解除されます。これは、一つのメンバーにより飲み構成されることを意味します。関数アイテム(3番目の引数 Function) はアイテムのシーケンスである出力であるペアのシーケンス内のペア(ジョイント、ジョイントの解除) に個別に適用されます。

サンプル

- **altova:for-each-attribute-pair**(/Example/Test-A, /Example/Test-B, function($a, $b){$a+b}) は以下 を返します。...

```
(2, 4, 6) if
<Test-A att1="1" att2="2" att3="3" />
<Test-B att1="1" att2="2" att3="3" />

(2, 4, 6) if
<Test-A att2="2" att1="1" att3="3" />
<Test-B att3="3" att2="2" att1="1" />

(2, 6) if
<Test-A att4="4" att1="1" att3="3" />
<Test-B att3="3" att2="2" att1="1" />
```

メモ: 結果 (2, 6) は次のアクションにより取得されます: (1+1, ()+2, 3+3, 4+()) 。演算子の一つか空のシーケンスの場合、アイテム 2 と 4 の場合同様、結果は空のシーケンスになります。

- **altova:for-each-attribute-pair**(/Example/Test-A, /Example/Test-B, concat#2) は以下を返します。

```
(11, 22, 33) if
<Test-A att1="1" att2="2" att3="3" />
<Test-B att1="1" att2="2" att3="3" />

(11, 2, 33, 4) if
<Test-A att4="4" att1="1" att3="3" />
<Test-B att3="3" att2="2" att1="1" />
```

for-each-combination [altova:]

altova:for-each-combination(FirstSequence *as item()**, **SecondSequence** *as item()**, **function($i,$j){$i || $j})** を **item()*** とする **XP3.1** **XQ3.1**
第 3 の引数として与えられた、インライン関数を返します。FirstSequence 内の各 $i を SecondSequence内の各 $j と結合します。出力はこれらのアイテムのシーケンスです。

サンプル

- **altova:for-each-combination**(('a', 'b', 'c'), ('1', '2', '3'), function($i, $j) {$i || $j}) は ('a1', 'a2', 'a3', 'b1', 'b2', 'b3', 'c1', 'c2', 'c3') を返します。

for-each-combination [altova:]

altova:for-each-combination(FirstSequence *as item()**, **SecondSequence** *as item()**, **Function($i,$j){$i || $j}) asitem()*** **XP3.1** **XQ3.1**
最初の2 つの引数内の2 つのシーケンスのアイテムは最初のシーケンスの各アイテムが2 番目のシーケンスになるように結合されるように結合されます。3 番目の引数として与えられる関数は結果のシーケンス内の組み合わせに適用され、アイテムのシーケンスである出力内で出力されます（*サンプル参照*）。

サンプル

- **altova:for-each-combination**(('a', 'b', 'c'), ('1', '2', '3'), function($i, $j) {$i || $j}) は ('a1', 'a2', 'a3', 'b1', 'b2', 'b3', 'c1', 'c2', 'c3') を返します。

(C) 2015-2021 Altova GmbH

▼ substitute-empty [altova:]

altova:substitute-empty(FirstSequence *as item()**, **SecondSequence** *as item()*) を**item()*** とする XP3.1 XQ3.1
FirstSequence が空の場合、SecondSequence を返します。FirstSequence が空でない場合、FirstSequence を返します。

サンプル

- **altova:substitute-empty**((1,2,3), (4,5,6)) は (1,2,3) を返します。
- **altova:substitute-empty**((), (4,5,6)) は (4,5,6) を返します。

19.1.2.1.8 XPath/XQuery 関数: 文字列

Altova の文字列拡張関数は XPath と XQuery 式で使用することができ、XML スキーマの異なる日付および時刻データ型で保存されているデータを処理するための追加機能を提供します。このセクションの関数は、Altova の **XPath 3.0** および **XQuery 3.0** エンジンで使用することができます。これらの関数は、XPath/XQuery コンテキストで使用することができます。

関数の名前指定と言語の適用性に関するメモ

Altova 拡張関数は XPath/XQuery 式で使用することができ、XPath、XQuery、および XSLT 関数の標準ライブラリで使用可能な機能に更なる機能性を与えます。Altova 拡張関数は **Altova 拡張関数名前空間**、**http://www.altova.com/xslt-extensions** に収められており、**altova:** プレフィックスが、このセクションでは使用されます。製品の今後のバージョンが拡張機能への継続的サポート、または個別の関数の振る舞いは変更する可能性があることにご注意してください。Altova 拡張機能へのサポートに関しては、今後のリリースのドキュメントを参照してください。

XPath 関数 (XSLT 内の XPath 式で使用):	XP1 XP2 XP3.1
XSLT 関数 (XSLT 内の XPath 式で使用):	XSLT1 XSLT2 XSLT3
XQuery 関数 (XQuery 内の XQuery 式で使用):	XQ1 XQ3.1

▼ camel-case [altova:]

altova:camel-case(InputString *as xs:string*) を**xs:string** とする XP3.1 XQ3.1
入力文字列を **InputString** をキャメルケースで返します。文字列は空白スペースのショートカットである) 正規表現 **'\s'** を使用して分析されます。空白文字または連続する空白文字のシーケンスの後の最初の非空白スペース文字は大文字です。出力文字列の最初の文字は大文字です。

サンプル

- **altova:camel-case**("max") Max を返します。
- **altova:camel-case**("max max") Max Max を返します。
- **altova:camel-case**("file01.xml") File01.xml を返します。
- **altova:camel-case**("file01.xml file02.xml") File01.xml File02.xml を返します。
- **altova:camel-case**("file01.xml file02.xml") File01.xml File02.xml を返します。
- **altova:camel-case**("file01.xml -file02.xml") File01.xml -file02.xml を返します。

altova:camel-case(InputString *as xs:string*, **SplitChars** *as xs:string*, **IsRegex** *as xs:boolean*) を**xs:string** とする XP3.1 XQ3.1
SplitChars を使用して、次の大文字をトリガーする文字を決定し、入力文字列を **InputString** キャメルケースに変換しま

す。 **SplitChars** は、**IsRegex = true()** の場合、または、**IsRegex = false()** の場合、プレーン文字は正規表現として使用されます。出力文字列の最初の文字は大文字です。

サンプル

- **altova:camel-case**("setname getname", "set|get", true()) setName getName を返します。
- **altova:camel-case**("altova\documents\testcases", "\", false()) Altova\Documents\Testcases を返します。

char [altova:]

altova:char(Position *as xs:integer*) を **xs:string** とする **XP3.1** **XQ3.1**
xs:string.Position に対してのコンテキストアイテムの値を変換することにより得られた文字列内の Position 引数により指定されたポジションにある文字を含む文字列を返します。引数により提出されたインデックスに文字が存在しない場合、結果文字列は空です。

サンプル

コンテキスト アイテムが **1234ABCD** の場合:

- **altova:char**(2) は 2 を返します。
- **altova:char**(5) は A を返します。
- **altova:char**(9) は空の文字列を返します。
- **altova:char**(-2) は空の文字列を返します。

altova:char(InputString *as xs:string*, **Position** *as xs:integer*) を **xs:string** とする **XP3.1** **XQ3.1**
引数として提出された文字列内の Position 引数により指定されたポジションでの文字を含む文字列を返します。Position 引数により提出されたインデックスに文字が存在しない場合、結果文字列は空です。

サンプル

- **altova:char**("2014-01-15", 5) は - を返します。
- **altova:char**("USA", 1) は U を返します。
- **altova:char**("USA", 10) は空の文字列を返します。
- **altova:char**("USA", -2) は空の文字列を返します。

create-hash-from-string[altova:]

altova:create-hash-from-string(InputString *as xs:string*) **asxs:string** **XP2** **XQ1** **XP3.1** **XQ3.1**
altova:create-hash-from-string(InputString *as xs:string*, **HashAlgo** *as xs:string*) **asxs:string** **XP2** **XQ1** **XP3.1** **XQ3.1**
HashAlgo 引数により指定されているハッシュアルゴリズムを使用して InputString からハッシュ文字列を生成します。次のハッシュアルゴリズムは（大文字、または、小文字で指定されている可能性があります: **MD5**, **SHA-224**, **SHA-256**, **SHA-384**, **SHA-512**。（最初の署名を参照してください）2番目の引数が指定されていない場合、**SHA-256** ハッシュアルゴリズムが使用されます。

サンプル

- **altova:create-hash-from-string**('abc') は **SHA-256** ハッシュアルゴリズムを使用して生成されたハッシュ文字列 を返します。
- **altova:create-hash-from-string**('abc', 'md5') は **MD5** ハッシュアルゴリズムを使用して生成されたハッシュ文字列 を返します。
- **altova:create-hash-from-string**('abc', 'MD5') は **MD5** ハッシュアルゴリズムを使用して生成された

(C) 2015-2021 Altova GmbH

ハッシュ文字列 を返します。

▼ first-chars [altova:]

altova:first-chars(X-Number *as xs:integer*) を **xs:string** とする **XP3.1** **XQ3.1**
xs:string に対するコンテキストアイテムの値を変換することにより得られた最初の X-Number 文字を含む文字列を返します。

▣ *サンプル*

コンテキストアイテムが **1234ABCD** の場合:

- **altova:first-chars**(2) は 12 を返します。
- **altova:first-chars**(5) は 1234A を返します。
- **altova:first-chars**(9) は 1234ABCD を返します。

altova:first-chars(InputString *as xs:string*, **X-Number** *as xs:integer*) を **xs:string** とする **XP3.1** **XQ3.1**
InputString 引数として提出された文字列の最初の文字を含む文字列を返します。

▣ *サンプル*

- **altova:first-chars**("2014-01-15", 5) は 2014- を返します。
- **altova:first-chars**("USA", 1) は U を返します。

▼ format-string [altova:]

altova:format-string(InputString *as xs:string*, **FormatSequence** *as item()**) **as xs:string** **XP3.1** **XQ3.1**
入力文字列（最初の引数）には、配置パラメーター（%1, %2, etc）が含まれています。各パラメーターは、（2番目の引数として提出されている）フォーマットシーケンス内の対応するポジションでロケートされる文字列アイテムと置き換えられます。フォーマットシーケンス内の最初のアイテムは、配置パラメーター %1 を置換、二番目のアイテムは %2 を置き換えます。関数は、書式設定された代替を持つ文字列を返します。配置パラメーターに文字列が存在しない場合は、配置パラメーターが返されます。これは、配置パラメーターのインデックスが書式シーケンス内のアイテムの数より大きい場合発生します。

▣ *サンプル*

- **altova:format-string**('Hello %1, %2, %3', ('Jane','John','Joe')) は "Hello Jane, John, Joe" を返します。
- **altova:format-string**('Hello %1, %2, %3', ('Jane','John','Joe', 'Tom')) は"Hello Jane, John, Joe" を返します。
- **altova:format-string**('Hello %1, %2, %4', ('Jane','John','Joe', 'Tom')) は "Hello Jane, John, Tom" を返します。
- **altova:format-string**('Hello %1, %2, %4', ('Jane','John','Joe')) は "Hello Jane, John, %4" を返します。

▼ last-chars [altova:]

altova:last-chars(X-Number *as xs:integer*) を **xs:string** とする **XP3.1** **XQ3.1**
xs:string に対してのコンテキストアイテムの値の変換より取得された文字列の最後の X-Number 文字を含んでいる文字列を返します。

▣ *サンプル*

コンテキストアイテムが **1234ABCD** の場合:

- **altova:last-chars**(2) は CD を返します。
- **altova:last-chars**(5) は 4ABCD を返します。
- **altova:last-chars**(9) は 1234ABCD を返します。

altova:last-chars(InputString *as xs:string*, **X-Number** *as xs:integer*) **as xs:string** とする
XP3.1 **XQ3.1**
引数として提出された文字列の最後の X-Number 文字を含んでいる文字列を返します。

サンプル

- **altova:last-chars**("2014-01-15", 5) は 01-15 を返します。
- **altova:last-chars**("USA", 10) は USA を返します。

pad-string-left [altova:]

altova:pad-string-left(StringToPad *as xs:string*, **StringLength** *as xs:integer*, **PadCharacter** *as xs:string*) を **xs:string** とする **XP3.1** **XQ3.1**
PadCharacter 引数は1文字です。文字列の左側にパドされ、この数が、StringLength 引数の整数の値と等しくなるように StringToPad の文字の数を増やします。StringLength 引数は任意の整数の値（正数または負数）を持つことができますが、パディングは、StringLength の値が StringToPad 内の文字数よりも多い場合の場合のみ発生します。もし、StringToPad が StringLength の値よりも多くの文字数も持つ場合、StringToPad は変更されません。

サンプル

- **altova:pad-string-left**('AP', 1, 'Z') は 'AP' を返します。
- **altova:pad-string-left**('AP', 2, 'Z') は 'AP' を返します。
- **altova:pad-string-left**('AP', 3, 'Z') は 'ZAP' を返します。
- **altova:pad-string-left**('AP', 4, 'Z') は 'ZZAP' を返します。
- **altova:pad-string-left**('AP', -3, 'Z') は 'AP' を返します。
- **altova:pad-string-left**('AP', 3, 'YZ') は[パド文字が長すぎます] エラーを返します。

pad-string-right [altova:]

altova:pad-string-right(StringToPad *as xs:string*, **StringLength** *as xs:integer*, **PadCharacter** *as xs:string*) を **xs:string** とする **XP3.1** **XQ3.1**
PadCharacter 引数は1文字です。文字列の右側にパドされ、この数が、StringLength 引数の整数の値と等しくなるように StringToPad の文字の数を増やします。StringLength 引数は任意の整数の値（正数または負数）を持つことができますが、パディングは、StringLength の値が StringToPad 内の文字数よりも多い場合の場合のみ発生します。もし、StringToPad が StringLength の値よりも多くの文字数も持つ場合、StringToPad は変更されません。

サンプル

- **altova:pad-string-right**('AP', 1, 'Z') を 'AP' を返します。
- **altova:pad-string-right**('AP', 2, 'Z') を 'AP' を返します。
- **altova:pad-string-right**('AP', 3, 'Z') を 'APZ' を返します。
- **altova:pad-string-right**('AP', 4, 'Z') を 'APZZ' を返します。
- **altova:pad-string-right**('AP', -3, 'Z') を 'AP' を返します。
- **altova:pad-string-right**('AP', 3, 'YZ') は[パド文字が長すぎます] エラーを返します。

repeat-string [altova:]

altova:repeat-string(InputString *as xs:string*, **Repeats** *as xs:integer*) を **xs:string** とする

(C) 2015-2021 Altova GmbH

XP2 XQ1 XP3.1 XQ3.1
最初のInputString引数により構成される文字列、Repeats 回繰り返してを生成します。
サンプル

- altova:repeat-string("Altova #", 3) は"Altova #Altova #Altova #" を返します。

substring-after-last [altova:]

altova:substring-after-last(MainString *as xs:string*, **CheckString** *as xs:string*) を **xs:string** とする XP3.1 XQ3.1
CheckString がMainString 内で検出された場合、MainString 内のCheckString が発生した後のサブ文字列が返されます。MainString 内でCheckString が検出されない場合、空の文字列が返されます。CheckString が空の文字列の場合、MainString 全体が返されます。一度以上発生する場合、CheckString の最後の発生の後の サブ文字列が返されます。
サンプル

- altova:substring-after-last('ABCDEFGH', 'B') は'CDEFGH' を返します。
- altova:substring-after-last('ABCDEFGH', 'BC') は'DEFGH' を返します。
- altova:substring-after-last('ABCDEFGH', 'BD') は'' を返します。
- altova:substring-after-last('ABCDEFGH', 'Z') は'' を返します。
- altova:substring-after-last('ABCDEFGH', '') は'ABCDEFGH' を返します。
- altova:substring-after-last('ABCD-ABCD', 'B') は'CD' を返します。
- altova:substring-after-last('ABCD-ABCD-ABCD', 'BCD') は'' を返します。

substring-before-last [altova:]

altova:substring-before-last(MainString *as xs:string*, **CheckString** *as xs:string*) を **xs:string** とする XP3.1 XQ3.1
CheckString がMainString 内で検出された場合、MainString 内のCheckString が発生する前のサブ文字列が返されます。MainString 内でCheckString が一度以上発生する場合、CheckString の最後の発生の前の サブ文字列が返されます。
サンプル

- altova:substring-before-last('ABCDEFGH', 'B') は'A' を返します。
- altova:substring-before-last('ABCDEFGH', 'BC') は'A' を返します。
- altova:substring-before-last('ABCDEFGH', 'BD') は'' を返します。
- altova:substring-before-last('ABCDEFGH', 'Z') は'' を返します。
- altova:substring-before-last('ABCDEFGH', '') は'' を返します。
- altova:substring-before-last('ABCD-ABCD', 'B') は'ABCD-A' を返します。
- altova:substring-before-last('ABCD-ABCD-ABCD', 'ABCD') は'ABCD-ABCD-' を返します。

substring-pos [altova:]

altova:substring-pos(StringToCheck *as xs:string*, **StringToFind** *as xs:string*) を **xs:integer** とする XP3.1 XQ3.1
StringToCheck 内でのStringToFind の最初の発生の文字位置を整数として返します。StringToCheck の最初の文字は、位置 1 にあります。StringToFind がStringToCheck 内で発生しない場合、整数 0 が返されます。第 2 またはその後のStringToCheck, 発生を確認するには、この関数の次の署名を確認してください。
サンプル

- altova:substring-pos('Altova', 'to') は 3 を返します。

- **altova:substring-pos**('Altova', 'tov') は 3 を返します。
- **altova:substring-pos**('Altova', 'tv') は 0 を返します。
- **altova:substring-pos**('AltovaAltova', 'to') は 3 を返します。

altova:substring-pos(StringToCheck *as xs:string*, **StringToFind** *as xs:string*, **Integer** *as xs:integer*) を **xs:integer** とする **XP3.1** **XQ3.1**
StringToCheck 内での StringToFind の最初の発生の文字位置を返します。Integer 引数により与えられた、文字位置から StringToFind の検索が開始されます。この位置の前の文字サブ文字列は検索されません。しかし、返された整数は、全体文字列 StringToCheck の検索された文字列の位置です。この署名は、StringToCheck 内で複数回発生する、第 2 または後の発生を検索する際に役に立ちます。 StringToFind が StringToCheck 内で発生しない場合、整数 0 が返されます。

サンプル

- **altova:substring-pos**('Altova', 'to', 1) は 3 を返します。
- **altova:substring-pos**('Altova', 'to', 3) は 3 を返します。
- **altova:substring-pos**('Altova', 'to', 4) は 0 を返します。
- **altova:substring-pos**('Altova-Altova', 'to', 0) は 3 を返します。
- **altova:substring-pos**('Altova-Altova', 'to', 4) は 10 を返します。

trim-string [altova:]

altova:trim-string(InputString *as xs:string*) を **xs:string** とする **XP3.1** **XQ3.1**
この関数は xs:string 引数を必要とし、先頭および後続の空白を削除し、トリミングされた xs:string を返します。

サンプル

- **altova:trim-string**(" Hello World ")) は "Hello World" を返します。
- **altova:trim-string**("Hello World ")) は "Hello World" を返します。
- **altova:trim-string**(" Hello World")) は "Hello World" を返します。
- **altova:trim-string**("Hello World")) は "Hello World" を返します。
- **altova:trim-string**("Hello World")) は "Hello World" を返します。

trim-string-left [altova:]

altova:trim-string-left(InputString *as xs:string*) を **xs:string** とする **XP3.1** **XQ3.1**
この関数は xs:string 引数を必要とし、先頭および後続の空白を削除し、トリミングされた xs:string を返します。

サンプル

- **altova:trim-string-left**(" Hello World ")) は "Hello World " を返します。
- **altova:trim-string-left**("Hello World ")) は "Hello World " を返します。
- **altova:trim-string-left**(" Hello World")) は "Hello World" を返します。
- **altova:trim-string-left**("Hello World")) は "Hello World" を返します。
- **altova:trim-string-left**("Hello World")) は "Hello World" を返します。

trim-string-right [altova:]

altova:trim-string-right(InputString *as xs:string*) を **xs:string** とする **XP3.1** **XQ3.1**
この関数は xs:string 引数を必要とし、先頭および後続の空白を削除し、トリミングされた xs:string を返します。

サンプル

(C) 2015-2021 Altova GmbH

- **altova:trim-string-right**(" Hello World ")は" Hello World" を返します。
- **altova:trim-string-right**("Hello World ")は"Hello World" を返します。
- **altova:trim-string-right**(" Hello World")は" Hello World" を返します。
- **altova:trim-string-right**("Hello World")は"Hello World" を返します。
- **altova:trim-string-right**("Hello World")は"Hello World" を返します。

19.1.2.1.9 XPath/XQuery 関数: その他

XPath/XQuery 拡張関数の以下の一般的な目的は現在のMapForce バージョンでサポートされており(i) XSLT コンテキスト内のXPath 式 または(ii) Xquery ドキュメント内のXQuery 式内で使用することができます。。

関数の名前指定と言語の適用性に関するメモ

Altova 拡張関数はXPath/XQuery 式で使用することができ、XPath、XQuery、およびXSLT 関数の標準ライブラリで使用可能な機能に更なる機能性を与えます。Altova 拡張関数はAltova 拡張関数名前空間、**http://www.altova.com/xslt-extensions** に収められており、**altova:** プレフィックスが、このセクションでは使用されます。製品の今後のバージョンが拡張機能への継続的サポート、または個別の関数の振る舞いは変更する可能性があることご注意してください。Altova 拡張機能へのサポートに関しては、今後のリリースのドキュメントを参照してください。

XPath 関数 (XSLT 内のXPath 式で使用):	**XP1** **XP2** **XP3.1**
XSLT 関数 (XSLT 内のXPath 式で使用):	**XSLT1** **XSLT2** **XSLT3**
XQuery 関数 (XQuery 内のXQuery 式で使用):	**XQ1** **XQ3.1**

▼ decode-string [altova:]

altova:decode-string(Input *as xs:base64Binary*) **as xs:string** **XP3.1** **XQ3.1**
altova:decode-string(Input *as xs:base64Binary*, **Encoding** *as xs:string*) **as xs:string** **XP3.1** **XQ3.1**
指定されたエンコードを使用して送信されたbase64Binary 入力を文字列にデコードします。エンコードが指定されていない場合UTF-8 エンコードが使用されます。以下のエンコードがサポートされます: US-ASCII, ISO-8859-1, UTF-16, UTF-16LE, UTF-16BE, ISO-10646-UCS2, UTF-32, UTF-32LE, UTF-32BE, ISO-10646-UCS4

▣ *サンプル*

- **altova:decode-string**($XML1/MailData/Meta/b64B)はbase64Binary 入力をUTF-8 エンコード済み文字列として返します。
- **altova:decode-string**($XML1/MailData/Meta/b64B, "UTF-8")はbase64Binary 入力をUTF-8-エンコード済み文字列として返します。
- **altova:decode-string**($XML1/MailData/Meta/b64B, "ISO-8859-1")はbase64Binary 入力をISO-8859-1 エンコード済み文字列として返します。

▼ encode-string [altova:]

altova:encode-string(InputString *as xs:string*) **as xs:base64Binaryinteger** **XP3.1** **XQ3.1**
altova:encode-string(InputString *as xs:string*, **Encoding** *as xs:string*) **as**

xs:base64Binaryinteger XP3.1 XQ3.1
与えらえている場合指定されているエンコードを使用して送信された文字列をエンコードします。エンコードが与えられていない場合 UTF-8 エンコードが使用されます。エンコードされた文字列はbase64Binary 文字に変換され、エンコードされたbase64Binary 値が返されます。最初にUTF-8 エンコードがサポートされ、サポートは以下のエンコードに拡張されます: `US-ASCII, ISO-8859-1, UTF-16, UTF-16LE, UTF-16BE, ISO-10646-UCS2, UTF-32, UTF-32LE, UTF-32BE, ISO-10646-UCS4`

サンプル

- **altova:encode-string**("Altova") はUTF-8 エンコード済み文字列 "Altova" の等価である base64Binary を返します。
- **altova:encode-string**("Altova", "UTF-8") はUTF-8 エンコード済み文字列 "Altova" の等価であるbase64Binary のを返します。

get-temp-folder [altova:]

altova:get-temp-folder() を xs:string とする XP2 XQ1 XP3.1 XQ3.1
この関数は引数を必要としません。この関数は現在のユーザーの一時的なフォルダーへのパスを返します。

サンプル

- **altova:get-temp-folder**() は、マシン上で、xs:string として C:\Users\<UserName>\AppData\Local\Temp\ と類似したパスを返します。

generate-guid [altova:]

altova:generate-guid() asxs:string XP2 XQ1 XP3.1 XQ3.1
ユニークな文字列 GUID 文字列を生成します。

サンプル

- **altova:generate-guid**() は例えば 85F971DA-17F3-4E4E-994E-99137873ACCD を返します。

high-res-timer [altova:]

altova:high-res-timer() asxs:double XP3.1 XQ3.1
秒数でシステムハイれぞリューションタイマーの値を返します。ハイれぞリューションタイマーがシステム内で存在すると、必要とされる場合、高度に正確な時間の計算を有効化します（例えば、アニメーションと正確なコード実行の時間の決定などが例として挙げられます）。この関数は、システムのタイマーに精度を与えます。

サンプル

- **altova:high-res-timer**() は '1.16766146154566E6' などを返します。

parse-html [altova:]

altova:parse-html(HTMLText *as xs:string*) **asnode()** XP3.1 XQ3.1
HTMLText 引数は、HTML ドキュメントのテキストを含む文字列です。関数は、文字列からHTML ツリーを作成します。提供された文字列は、HTML 要素を含む、または、含まない場合があります。いずれの場合でも、ツリーのルート要素は、**HTML** と名付けられます。提出された文字列内のHTML コードが有効な HTMLであることを確認することが奨励されます。.

サンプル

- **altova:parse-html**("<html><head/><body><h1>Header</h1></body></html>") は提供された文字列からHTML ツリーを作成します。

(C) 2015-2021 Altova GmbH

▼ sleep[altova:]

altova:sleep(Millisecs *as xs:integer*) **asempty-sequence()** XP2 XQ1 XP3.1 XQ3.1

Millisecs により与えられるミリ秒で示される期間のための実行を延期します。

サンプル

- **altova:sleep**(1000) は1000 ミリ秒のための実行を延期します。

[[トップ]]

19.1.2.1.10 チャート関数

以下に示されるチャート関数により、チャートをイメージとして作成、生成、保存することができます。これらの機能は、お使いの Altova 製品で、以下に記される方法によりサポートされます。しかしながら、将来使用されるバージョンの製品において、以下にある関数のサポートが打ち切られたり、個々の振る舞いが変化する可能性があることに留意してください。将来のリリースにおける Altova 拡張関数のサポートについては、そのバージョンのドキュメンテーションを参照ください。

(XSLT 関数ではない) チャートに関する XPath 関数は、2 つのグループに分けられています::

- チャートの生成と保存を行う関数
- チャートの作成を行う関数

メモ: チャート関数は、**Enterprise** ならびに**サーバー エディション**の Altova 製品でしかサポートされていないことにご注意してください。

メモ: サーバーエディションのチャートでサポートされるイメージのフォーマットは jpg、png、および bmp です。無損失圧縮のため、推奨されるオプションは png です。Enterprise エディションでサポートされるフォーマットは jpg. png、bmp、および gif です。

チャートの生成と保存を行う関数

これらの関数は、(チャート作成関数により得られた) チャートオブジェクトを受け取り、イメージの生成を行うか、イメージをファイルへ保存します。

altova:generate-chart-image ($chart, $width, $height, $encoding) as atomic

以下の説明を参照ください

- $chart は altova:create-chart 関数により得られたチャート拡張アイテムです。
- $width ならびに $height により大きさを指定する必要があります。
- $encoding は x-binarytobase64 または x-binarytobase16 であることができます。

関数からは、指定されたエンコーディングならびにイメージフォーマットにてチャートのイメージが返されます。

altova:generate-chart-image ($chart, $width, $height, $encoding, $imagetype) as atomic

以下の説明を参照ください

- $chart は altova:create-chart 関数により得られたチャート拡張アイテムです。

- $width ならびに$height により大きさを指定する必要があります。
- $encoding はx-binarytobase64 またはx-binarytobase16 であることができます。
- $imagetype は次のフォーマットから選択することができます: png、gif、bmp、jpg、jpeg。gif はサーバー製品でサポートされていないことにご注意してください。*ページの上のメモを参照してください*。

関数からは、指定されたエンコーディングならびにイメージフォーマットにてチャートのイメージが返されます。

altova:save-chart-image ($chart, $filename, $width, $height) as empty() *(Windows のみ)*

以下の説明を参照ください

- $chart はaltova:create-chart 関数により得られたチャート拡張アイテムです。
- $filename は保存されるチャートイメージのファイルパスならびにファイル名です。
- $width ならびに$height により大きさを指定する必要があります。

関数により$filename で指定されたファイルにチャートイメージが保存されます。

altova:save-chart-image ($chart, $filename, $width, $height, $imagetype) as empty() *(Windows のみ)*

以下の説明を参照ください

- $chart はaltova:create-chart 関数により得られたチャート拡張アイテムです。
- $filename は保存されるチャートイメージのファイルパスならびにファイル名です。
- $width ならびに$height により大きさを指定する必要があります。
- $imagetype は次のフォーマットから選択することができます: png、gif、bmp、jpg、jpeg。gif はサーバー製品でサポートされていないことにご注意してください。*ページの上のメモを参照してください*。

指定されたイメージフォーマットで$filename により指定されたファイルにチャートイメージが保存されます。

チャートの作成を行う関数

以下の関数によりチャートの作成を行うことができます

altova:create-chart($chart-config, $chart-data-series*) as chart extension item

以下の説明を参照ください

- $chart-config はaltova:create-chart-config 関数またはaltova:create-chart-config-from-xml 関数により取得されたchart-config 拡張アイテムとなります。
- $chart-data-series はaltova:create-chart-data-series 関数またはaltova:create-chart-data-series-from-rows 関数により取得されたchart-data-series 拡張アイテムとなります。

関数により、引数により与えられたデータから作成されたチャート拡張アイテムが返されます。

altova:chart($chart-config, $chart-data-series*) as chart extension item

以下の説明を参照ください

- $chart-config はチャート構成拡張アイテムです。順序の決まっていない4つのキーのシリーズです: 4つのキーが"width"、"height"、"title"、および"kind" である値ペアです。width とheight の値は整数で、チャートの幅と高さを

(C) 2015-2021 Altova GmbH

ピクセルで指定します。kind の値は以下のとおりです: Pie、 Pie3d、 BarChart、 BarChart3d、 BarChart3dGrouped、 LineChart、 ValueLineChart、 RoundGauge、 BarGauge。

- $chart-data-series は各配列がチャートデータ系列を定義するサイズ 3 の配列です。各配列は以下により構成されています: (i) データ系列の名前 (ii) X-軸の値 (iii) Y-軸の値。複数のデータ系列は提供される場合があります。このサンプルでは、例えば、2 つのデータ配列はそれぞれ月々の最高および最低気温を提供します。

巻数は引数を介して提供されるデータから作成されたチャート拡張アイテムを返します。

サンプル

```
altova:chart( map{'width':800, 'height':600, "kind":"LineChart", "title":"Monthly Temperatures"}, (['Min', $temps/Month, $temps/Month/@min], ['Max', $temps/Month, $temps/Month/@max]) )
```

altova:create-chart-config($type-name, $title) as chart-config extension item

以下の説明を参照ください

- $type-name により、以下から作成される種類のチャートが選択されます: Pie、Pie3d、 BarChart、BarChart3d、BarChart3dGrouped、 LineChart、ValueLineChart、RoundGauge、BarGauge
- $title によりチャートの名前が指定されます。

関数により、チャートの構成情報が含まれる chart-config 拡張アイテムが返されます。

altova:create-chart-config-from-xml($xml-struct) as chart-config extension item

以下の説明を参照ください

- $xml-struct はチャートの構成情報が含まれる XML 構造です。

関数により、チャートの構成情報が含まれる chart-config 拡張アイテムが返されます。この情報は XML データフラグメントから与えられます。

altova:create-chart-data-series($series-name?, $x-values*, $y-values*) as chart-data-series extension item

以下の説明を参照ください

- $series-name により、系列の名前が指定されます。
- $x-values により、X-軸値のリストが与えられます。
- $y-values により、Y-軸値のリストが与えられます。

関数により、チャートの作成に必要なデータの情報(系列の名前と軸のデータ)が含まれる chart-data-series 拡張アイテムが返されます。

altova:create-chart-data-row(x, y1, y2, y3, ...) as chart-data-x-*N*y-row extension item

以下の説明を参照ください

- x はチャートデータ行の X-軸カラムにおける値です。
- yN は Y-軸カラムの値です。

関数により、1 つの系列の X-軸カラムならびに Y-軸カラムに対するデータが含まれる chart-data-x-*N*y-row 拡張アイテムが返されます。

altova:create-chart-data-series-from-rows($series-names as xs:string*, $row*) as chart-data-series extension item

以下の説明を参照ください

- $series-name は作成される系列の名前です。
- $row は系列として作成される chart-data-x-*N*y-row 拡張アイテムです。

関数により、系列のX-軸ならびにY-軸に関するデータが含まれる chart-data-series 拡張アイテムが返されます。

altova:create-chart-layer($chart-config, $chart-data-series*) as chart-layer extension item

以下の説明を参照ください

- $chart-config は altova:create-chart-config 関数または altova:create-chart-config-from-xml 関数により取得された $chart-config 拡張アイテムです。
- $chart-data-series は altova:create-chart-data-series 関数または altova:create-chart-data-series-from-rows 関数により取得された chart-data-series 拡張アイテムです。

関数により、チャートのレイヤーデータが含まれる **chart-layer** 拡張アイテムが返されます。

altova:create-multi-layer-chart($chart-config, $chart-data-series*, $chart-layer*)

以下の説明を参照ください

- $chart-config は altova:create-chart-config 関数または altova:create-chart-config-from-xml 関数により取得された chart-config 拡張アイテムです。
- $chart-data-series は altova:create-chart-data-series 関数または altova:create-chart-data-series-from-rows 関数により取得された chart-data-series 拡張アイテムです。
- $chart-layer は altova:create-chart-layer 関数により取得された **chart-layer** 拡張アイテムです。

関数により、複数レイヤーのチャートアイテムが返されます。

altova:create-multi-layer-chart($chart-config, $chart-data-series*, $chart-layer*, xs:boolean $mergecategoryvalues)

以下の説明を参照ください

- $chart-config は altova:create-chart-config 関数または altova:create-chart-config-from-xml 関数により取得された chart-config 拡張アイテムです。
- $chart-data-series は altova:create-chart-data-series 関数または altova:create-chart-data-series-from-rows 関数により取得された chart-data-series 拡張アイテムです。
- $chart-layer は altova:create-chart-layer 関数により取得された **chart-layer** 拡張アイテムです。
- $mergecategoryvalues は true の場合、複数のデータシリーズをマージします。false の場合はマージしません。

関数により、複数レイヤーのチャートアイテムが返されます。

(C) 2015-2021 Altova GmbH

19.1.2.1.10.1 チャートデータの XML 構造

以下にチャートデータのXML 構造、ならびにそれがAltova のチャート拡張関数においてどのように表示されるかを示します。チャートの種類によっては、ここでの記述内容により外観が変化します。全ての要素が全種類のチャートに対して使用されるわけではなく、例えば<Pie>要素は棒グラフに対して無視されます。

メモ チャート関数は、**Enterprise** ならびに**サーバー エディション**のAltova 製品でしかサポートされていないことご注意してください。

```
<chart-config>
    <General
    SettingsVersion="1" must be provided
    ChartKind="BarChart" Pie, Pie3d, BarChart, StackedBarChart, BarChart3d,
BarChart3dGrouped, LineChart, ValueLineChart, AreaChart, StackedAreaChart, RoundGauge, BarGauge,
CandleStick
    BKColor="#ffffff" Color
    BKColorGradientEnd="#ffffff" Color. In case of a gradient, BKColor and BKColorGradientEnd define
the gradient's colors
    BKMode="#ffffff" Solid, HorzGradient, VertGradient
    BKFile="Path+Filename" String. If file exists, its content is drawn over the background.
    BKFileMode="Stretch" Stretch, ZoomToFit, Center, Tile
    ShowBorder="1" Bool
    PlotBorderColor="#000000" Color
    PlotBKColor="#ffffff" Color
    Title="" String
    ShowLegend="1" Bool
    OutsideMargin="3.%" PercentOrPixel
    TitleToPlotMargin="3.%" PercentOrPixel
    LegendToPlotMargin="3.%" PercentOrPixel
    Orientation="vert" Enumeration: possible values are: vert, horz
    >
    <TitleFont
        Color="#000000" Color
        Name="Tahoma" String
        Bold="1" Bool
        Italic="0" Bool
        Underline="0" Bool
        MinFontHeight="10.pt" FontSize (only pt values)
        Size="8.%" FontSize />
    <LegendFont
        Color="#000000"
        Name="Tahoma"
        Bold="0"
        Italic="0"
        Underline="0"
        MinFontHeight="10.pt"
        Size="3.5%" />
    <AxisLabelFont
        Color="#000000"
        Name="Tahoma"
        Bold="1"
        Italic="0"
        Underline="0"
```

```
      MinFontHeight="10.pt"
      Size="5.%" />
</General>

<Line
      ConnectionShapeSize="1.%"   PercentOrPixel
      DrawFilledConnectionShapes="1"   Bool
      DrawOutlineConnectionShapes="0"   Bool
      DrawSlashConnectionShapes="0"   Bool
      DrawBackslashConnectionShapes="0"   Bool
/>

<Bar
      ShowShadow="1"   Bool
      ShadowColor="#a0a0a0"   Color
      OutlineColor="#000000"   Color
      ShowOutline="1"   Bool
/>

<Area
      Transparency="0"   UINT ( 0-255 ) 255 is fully transparent, 0 is opaque
      OutlineColor="#000000"   Color
      ShowOutline="1"   Bool
/>

<CandleStick
      FillHighClose="0"   Bool. If 0, the body is left empty. If 1, FillColorHighClose is used for
the candle body
      FillColorHighClose="#ffffff"   Color. For the candle body when close > open
      FillHighOpenWithSeriesColor="1"   Bool. If true, the series color is used to fill the
candlebody when open > close
      FillColorHighOpen="#000000"   Color. For the candle body when open > close and
FillHighOpenWithSeriesColor is false
/>

<Colors   User-defined color scheme: By default this element is empty except for the style and
has no Color attributes
      UseSubsequentColors ="1"   Boolean. If 0, then color in overlay is used. If 1, then
subsequent colors from previous chart layer is used
      Style="User"   Possible values are: "Default", "Grayscale", "Colorful", "Pastel", "User"
      Colors="#52aca0"   Color: only added for user defined color set
      Colors1="#d3c15d"   Color: only added for user defined color set
      Colors2="#8971d8"   Color: only added for user defined color set
      ...
      ColorsN=""   Up to ten colors are allowed in a set: from Colors to Colors9
</Colors>

<Pie
      ShowLabels="1"   Bool
      OutlineColor="#404040"   Color
      ShowOutline="1"   Bool
      StartAngle="0."   Double
      Clockwise="1"   Bool
      Draw2dHighlights="1"   Bool
      Transparency="0"   Int (0 to 255: 0 is opaque, 255 is fully transparent)
```

```
            DropShadowColor="#c0c0c0"   Color
            DropShadowSize="5.%"   PercentOrPixel
            PieHeight="10.%"   PercentOrPixel. Pixel values might be different in the result because
of 3d tilting
            Tilt="40.0"   Double (10 to 90: The 3d tilt in degrees of a 3d pie)
            ShowDropShadow="1"   Bool
            ChartToLabelMargin="10.%"   PercentOrPixel
            AddValueToLabel="0"   Bool
            AddPercentToLabel="0"   Bool
            AddPercentToLabels_DecimalDigits="0"  UINT ( 0 – 2 )
            >
            <LabelFont
                  Color="#000000"
                  Name="Arial"
                  Bold="0"
                  Italic="0"
                  Underline="0"
                  MinFontHeight="10.pt"
                  Size="4.%"    />
      </Pie>

      <XY>
            <XAxis   Axis
                  AutoRange="1"   Bool
                  AutoRangeIncludesZero="1"   Bool
                  RangeFrom="0."   Double: manual range
                  RangeTill="1."   Double : manual range
                  LabelToAxisMargin="3.%"   PercentOrPixel
                  AxisLabel=""   String
                  AxisColor="#000000"   Color
                  AxisGridColor="#e6e6e6"   Color
                  ShowGrid="1"   Bool
                  UseAutoTick="1"   Bool
                  ManualTickInterval="1."   Double
                  AxisToChartMargin="0.px"   PercentOrPixel
                  TickSize="3.px"   PercentOrPixel
                  ShowTicks="1"   Bool
                  ShowValues="1"   Bool
                  AxisPosition="LeftOrBottom"   Enums: "LeftOrBottom", "RightOrTop", "AtValue"
                  AxisPositionAtValue = "0"   Double
                  >
                  <ValueFont
                        Color="#000000"
                        Name="Tahoma"
                        Bold="0"
                        Italic="0"
                        Underline="0"
                        MinFontHeight="10.pt"
                        Size="3.%"    />
            </XAxis>
            <YAxis   Axis (same as for XAxis)
                  AutoRange="1"
                  AutoRangeIncludesZero="1"
                  RangeFrom="0."
                  RangeTill="1."
                  LabelToAxisMargin="3.%"
                  AxisLabel=""
```

```
                AxisColor="#000000"
                AxisGridColor="#e6e6e6"
                ShowGrid="1"
                UseAutoTick="1"
                ManualTickInterval="1."
                AxisToChartMargin="0.px"
                TickSize="3.px"
                ShowTicks="1"   Bool
                ShowValues="1"   Bool
                AxisPosition="LeftOrBottom"   Enums: "LeftOrBottom", "RightOrTop", "AtValue"
                AxisPositionAtValue = "0"   Double
                >
                <ValueFont
                      Color="#000000"
                      Name="Tahoma"
                      Bold="0"
                      Italic="0"
                      Underline="0"
                      MinFontHeight="10.pt"
                      Size="3.%"/>
          </YAxis>
    </XY>

    <XY3d
          AxisAutoSize="1"   Bool: If false, XSize and YSize define the aspect ration of x and y
axis. If true, aspect ratio is equal to chart window
          XSize="100.%"   PercentOrPixel. Pixel values might be different in the result because of
3d tilting and zooming to fit chart
          YSize="100.%"   PercentOrPixel. Pixel values might be different in the result because of
3d tilting and zooming to fit chart
          SeriesMargin="30.%"   PercentOrPixel. Pixel values might be different in the result
because of 3d tilting and zooming to fit chart
          Tilt="20."   Double. -90 to +90 degrees
          Rot="20."   Double. -359 to +359 degrees
          FoV="50.">   Double. Field of view: 1-120 degree
          >
          <ZAxis
                AutoRange="1"
                AutoRangeIncludesZero="1"
                RangeFrom="0."
                RangeTill="1."
                LabelToAxisMargin="3.%"
                AxisLabel=""
                AxisColor="#000000"
                AxisGridColor="#e6e6e6"
                ShowGrid="1"
                UseAutoTick="1"
                ManualTickInterval="1."
                AxisToChartMargin="0.px"
                TickSize="3.px" >
                <ValueFont
                      Color="#000000"
                      Name="Tahoma"
                      Bold="0"
                      Italic="0"
                      Underline="0"
                      MinFontHeight="10.pt"
                      Size="3.%"/>
          </ZAxis>
```

```
        </XY3d>

        <Gauge
                MinVal="0."   Double
                MaxVal="100."   Double
                MinAngle="225"   UINT: -359-359
                SweepAngle="270"   UINT: 1-359
                BorderToTick="1.%"   PercentOrPixel
                MajorTickWidth="3.px"   PercentOrPixel
                MajorTickLength="4.%"   PercentOrPixel
                MinorTickWidth="1.px"   PercentOrPixel
                MinorTickLength="3.%"   PercentOrPixel
                BorderColor="#a0a0a0"   Color
                FillColor="#303535"   Color
                MajorTickColor="#a0c0b0"   Color
                MinorTickColor="#a0c0b0"   Color
                BorderWidth="2.%"   PercentOrPixel
                NeedleBaseWidth="1.5%"   PercentOrPixel
                NeedleBaseRadius="5.%"   PercentOrPixel
                NeedleColor="#f00000"   Color
                NeedleBaseColor="#141414"   Color
                TickToTickValueMargin="5.%"   PercentOrPixel
                MajorTickStep="10."   Double
                MinorTickStep="5."   Double
                RoundGaugeBorderToColorRange="0.%"   PercentOrPixel
                RoundGaugeColorRangeWidth ="6.%"   PercentOrPixel
                BarGaugeRadius="5.%"   PercentOrPixel
                BarGaugeMaxHeight="20.%"   PercentOrPixel
                RoundGaugeNeedleLength="45.%"   PercentOrPixel
                BarGaugeNeedleLength="3.%"   PercentOrPixel
                >
                <TicksFont
                        Color="#a0c0b0"
                        Name="Tahoma"
                        Bold="0"
                        Italic="0"
                        Underline="0"
                        MinFontHeight="10.pt"
                        Size="4.%"
                />
                <ColorRanges>   User-defined color ranges. By default empty with no child element entries
                        <Entry
                                From="50. "   Double
                                FillWithColor="1"   Bool
                                Color="#00ff00"   Color
                        />
                        <Entry
                                From="50.0"
                                FillWithColor="1"
                                Color="#ff0000"
                        />
                        ...
                </ColorRanges>
        </Gauge>
</chart-config>
```

19.1.2.1.10.2 チャート関数

以下のサンプルXSLTドキュメントでは、Altova のチャート拡張関数の使用方法が示されます。更に下には、XMLドキュメントと、Altova XSLT 2.0 エンジンとXSLTドキュメントによりXMLドキュメントが処理された結果生成される出力イメージのスクリーンショットが示されます。

メモ: チャート関数は、**Enterprise** ならびに**サーバー エディション**のAltova 製品でしかサポートされていないことにご注意してください。

メモ: チャートデータテーブルの作成に関する詳しい情報については、Altova XMLSpy ならびにStyleVision 製品のドキュメンテーションを参照ください。

XSLT ドキュメント

以下にあるXSLTドキュメントでは、Altova チャート拡張関数を使用することで円グラフを生成します。更に下に記されているXMLドキュメントを処理するために使用することができます。

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="2.0"
    xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    xmlns:altovaext="http://www.altova.com/xslt-extensions"
    exclude-result-prefixes="#all">
    <xsl:output version="4.0" method="html" indent="yes" encoding="UTF-8"/>
    <xsl:template match="/">
        <html>
            <head>
                <title>
                    <xsl:text>HTML Page with Embedded Chart</xsl:text>
                </title>
            </head>
            <body>
                <xsl:for-each select="/Data/Region[1]">
                    <xsl:variable name="extChartConfig" as="item()*">
                        <xsl:variable name="ext-chart-settings" as="item()*">
                            <chart-config>
                                <General
                                    SettingsVersion="1"
                                    ChartKind="Pie3d"
                                    BKColor="#ffffff"
                                    ShowBorder="1"
                                    PlotBorderColor="#000000"
                                    PlotBKColor="#ffffff"
                                    Title="{@id}"
                                    ShowLegend="1"
                                    OutsideMargin="3.2%"
                                    TitleToPlotMargin="3.%"
                                    LegendToPlotMargin="6.%"
                                    >
                                    <TitleFont
                                        Color="#023d7d"
                                        Name="Tahoma"
                                        Bold="1"
                                        Italic="0"
                                        Underline="0"
                                        MinFontHeight="10.pt"
                                        Size="8.%" />
```

```
                                </General>
                            </chart-config>
                        </xsl:variable>
                        <xsl:sequence select="altovaext:create-chart-config-from-xml( $ext-chart-settings )"/>
                    </xsl:variable>
                    <xsl:variable name="chartDataSeries" as="item()*">
                        <xsl:variable name="chartDataRows" as="item()*">
                            <xsl:for-each select="(Year)">
                                <xsl:sequence select="altovaext:create-chart-data-row( (@id), ( .) )"/>
                            </xsl:for-each>
                        </xsl:variable>
                        <xsl:variable name="chartDataSeriesNames" as="xs:string*" select=" ( (&quot;Series 1&quot;),
&apos;&apos; )[1]"/>
                        <xsl:sequence
                            select="altovaext:create-chart-data-series-from-rows( $chartDataSeriesNames,
$chartDataRows)"/>
                    </xsl:variable>
                    <xsl:variable name="ChartObj" select="altovaext:create-chart( $extChartConfig, ( $chartDataSeries),
false() )"/>
                    <xsl:variable name="sChartFileName" select="'mychart1.png'"/>
                    <img src="{$sChartFileName, altovaext:save-chart-image( $ChartObj, $sChartFileName, 400,
400 ) }"/>
                </xsl:for-each>
            </body>
        </html>
    </xsl:template>
</xsl:stylesheet>
```

XML ドキュメント

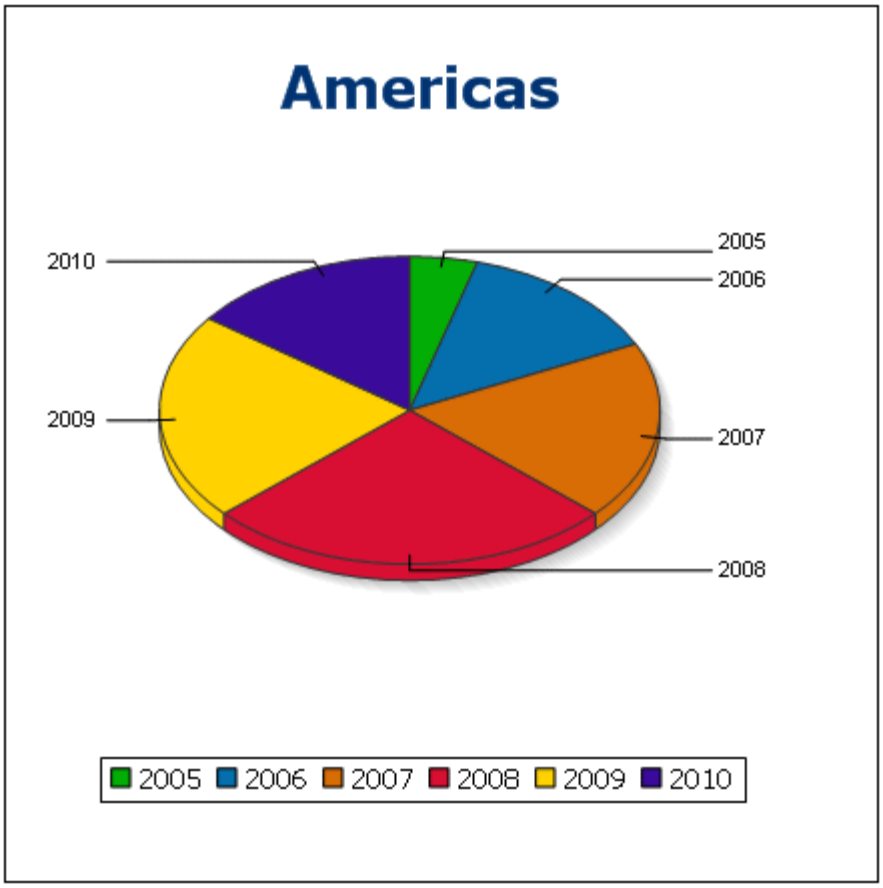
上にある XSLT ドキュメントにより以下の XML ドキュメントを処理することができます。XML ドキュメント内にあるデータを使用することで、下に示される円グラフを生成することができます。

```
<?xml version="1.0" encoding="UTF-8"?>
<Data xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="YearlySales.xsd">
    <ChartType>Pie Chart 2D</ChartType>
    <Region id="Americas">
        <Year id="2005">30000</Year>
        <Year id="2006">90000</Year>
        <Year id="2007">120000</Year>
        <Year id="2008">180000</Year>
        <Year id="2009">140000</Year>
        <Year id="2010">100000</Year>
    </Region>
    <Region id="Europe">
        <Year id="2005">50000</Year>
        <Year id="2006">60000</Year>
        <Year id="2007">80000</Year>
        <Year id="2008">100000</Year>
        <Year id="2009">95000</Year>
        <Year id="2010">80000</Year>
    </Region>
    <Region id="Asia">
        <Year id="2005">10000</Year>
        <Year id="2006">25000</Year>
        <Year id="2007">70000</Year>
        <Year id="2008">110000</Year>
        <Year id="2009">125000</Year>
        <Year id="2010">150000</Year>
    </Region>
```

</Data>

出力イメージ

上にあるXSLTドキュメントを使ってXMLドキュメントを処理することで、以下にある円グラフが生成されます



19.1.2.2 その他の拡張関数

Java やC# などのプログラミング言語には、XPath 2.0 / XQuery 関数、またはXSLT 2.0 関数として利用できない関数がいくつかあります。そのような関数の良い例として、Java で利用することのできる `sin()` や `cos()` といった数学関数があります。XSLT スタイルシートやXQuery のクエリにてこれらの関数が利用できるのであれば、スタイルシートやクエリの適用範囲を大幅に拡張することができ、スタイルシート作成タスクの負担が大幅に軽減されます。Altova 製品で使用されているAltova エンジン(XSLT 1.0、XSLT 2.0、XQuery 1.0)では、Java や.NET およびMSXSL scripts for XSLTにおける拡張関数の使用がサポートされます。 このセクションでは、拡張機能およびXSLT スタイルシート内でMSXSL スクリプト。およびXQuery ドキュメントを使用する方法について記述します。使用できる拡張関数は以下のように構成されます:

- Java 拡張関数
- .NET 拡張関数
- XSLT に対するMSXSL スクリプト

記述の中では特に、(i) 関連するライブラリ内の関数がどのように呼ばれるか、(ii) 関数呼び出しを行う際に入力として使用される引数を変換するのにどのようなルールが適用され、return により値が返される際にどのような変換ルールが適用されるのか(XSLT/XQuery データオブジェクトに対する関数の結果)について説明されます。

必要条件

拡張関数のサポートを有効にするには、XSLT 変換や XQuery の実行を行うコンピューターに Java Runtime Environment (Java 関数にアクセスする場合)ならびに .NET Framework 2.0 以上(.NET 関数にアクセスする場合)がインストールされている、またはアクセスできる環境が整っている必要があります。

19.1.2.2.1 Java 拡張関数

Java 拡張関数は、XPath または XQuery 条件式にて使用することができるほか、Java のコンストラクターを呼び出したり、Java の(静的またはインスタンス)メソッドを呼び出すことができます。

Java クラスのフィールドは、引数を持たないメソッドとして扱われます。フィールドは静的またはインスタンスとして存在することができます。フィールドへのアクセス方法については、静的とインスタンスの両方について、以下のサブセクションにて記述されます。

このセクションは以下のサブセクションにより構成されます:

- Java : コンストラクター
- Java : 静的メソッドと静的フィールド
- Java : インスタンスメソッドとインスタンスフィールド
- データ型 : XPath/XQuery から Java へ
- データ型 : Java から XPath/XQuery へ

以下の点に注意してください:

- Altova デスクトップ製品を使用している場合、Altova アプリケーションは、Java 仮想マシンへのパスを(以下の順序で)読み取ることにより自動的に検知しようとします: (i) Windows レジストリ (ii) **JAVA_HOME** 環境変数。アプリケーションのオプションダイアログ内にカスタムパスを追加することもできます。このエントリは自動的に検知された他の Java VM パス以上の優先順位を有します。
- Altova サーバー製品が Windows マシン上で作動している場合、Java 仮想マシンへのパスは Windows レジストリから最初に読み取られます。成功しない場合、**JAVA_HOME** 環境変数が使用されます。
- Altova サーバー製品を作動する場合、および、Linux または macOS マシン上でサーバー製品を作動する場合、Java 仮想マシンへのパスが **JAVA_HOME** 環境変数内に保管されていることを確認してください。**\bin\server** または **\bin\client** ディレクトリ内の **jvm.dll** ファイルを指している必要があります。

拡張関数のフォーム

XPath/XQuery 条件式における拡張関数では、`prefix:fname()` の形式を取る必要があります。

- `prefix:` 部により拡張関数が Java 関数として認識されます。`java:` から始まる URI のスコープ内の名前空間宣言に拡張関数を関連付けることで Java 関数であるという認識が行われます。名前空間の宣言により、例えば `xmlns:myns="java:java.lang.Math"` という Java クラスが特定されます。名前空間の宣言は、(コロン無しの) `xmlns:myns="java"` という形式で、Java クラスの識別子を拡張関数にある `fname()` 部の左型に配置することでも行うことができます。

- `fname()` 部により、呼び出されている Java メソッドが識別され、メソッドの引数が提供されます（以下の例を参照ください）。
  `prefix:` 部にて識別された名前空間 URI が Java クラスを識別できなかった場合、Java クラスの識別はクラスの前にくる `fname()` 部にて行うことになり、ピリオドによりクラスから分離されることになります（以下にある2番目のXSLT サンプルを参照）。

メモ 呼び出されるクラスはコンピューターのクラス パス上にある必要があります。

XSLT サンプル

以下に静的メソッドを呼び出す2つのサンプルを示します。最初のサンプルでは、クラス名（`java.lang.Math`）が名前空間 URI に加えられており、`fname()` へ加えることはできません。2 番目のサンプルでは、`prefix:` 部に`java:` が与えられており、`fname()` 部にてクラスとメソッドが識別されます。

```
<xsl:value-of xmlns:jMath="java:java.lang.Math"
              select="jMath:cos(3.14)" />

<xsl:value-of xmlns:jmath="java"
              select="jmath:java.lang.Math.cos(3.14)" />
```

拡張関数内にあるメソッド名（上の例では`cos()` ）は、名前付き Java クラス（上の例では`java.lang.Math` ）のpublic な静的メソッドの名前に一致する必要があります。

XQuery サンプル

以下にXSLT のサンプルに似たXQuery のサンプルを示します:

```
<cosine xmlns:jMath="java:java.lang.Math">
   {jMath:cos(3.14)}
</cosine>
```

ユーザー定義されたJava クラス

独自のJava クラスやメソッドを作成した場合、(i) JAR ファイル（または class ファイル）を介してこれらクラスファイルへアクセスしているか、(ii) これら（JAR または class）ファイルが、カレントディレクトリ（XSLT やXQuery ドキュメントが存在するディレクトリ）に配置されているかにより、これらクラスの呼び出し方法が変わってきます。これらファイルの特定方法については、ユーザー定義クラスファイルならびにユーザー定義 JAR ファイルを参照ください。カレントディレクトリには無いクラスファイルやJAR ファイルへのパスは指定しなければならないことにご注意してください。

19.1.2.2.1.1 ユーザー定義のクラスファイル

アクセスがクラスファイルを介したものである場合、4つのケースが考えられます:

- クラスファイルはパッケージである。XSLT またはXQuery ファイルがJava パッケージと同じ場所に収められている。（下のサンプルを参照）
- クラスファイルはパッケージでは無い。XSLT またはXQuery ファイルがJava パッケージと同じ場所に収められている。（下のサンプルを参照）
- クラスファイルはパッケージである。XSLT またはXQuery ファイルがランダムな場所に収められている。（下のサンプルを参照）
- クラスファイルはパッケージである。XSLT またはXQuery ファイルがランダムな場所に収められている。（下のサンプルを参照）

クラスファイルがパッケージではなく、XSLT または XQuery ドキュメントと同じ場所に収められているケースを考えてみましょう。この場合、フォルダー内の全クラスを発見することができるため、ファイルの場所を指定する必要はありません。クラスの識別を行う構文は以下のようになります:

```
java:classname
```

ここで、

`java:` によりユーザー定義の Java 関数が呼ばれていることが示されます(デフォルトでカレントディレクトリにある Java クラスがロードされます)。
`classname` は目的となるメソッドのクラスが含まれているクラスの名前です。

クラスが名前空間 URI にて識別され、名前空間がメソッド呼び出しにて使用されます。

クラスファイルがパッケージで、XSLT/XQuery ファイルが Java パッケージと同じ場所に収められている

以下の例では、`com.altova.extfunc` パッケージにある `Car` クラスの `getVehicleType()` メソッドが呼び出されています。`com.altova.extfunc` パッケージは `JavaProject` という名前のフォルダーに置かれており、XSLT ファイルも同じフォルダーに配置されています。

```
<xsl:stylesheet version="2.0"
       xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
       xmlns:xs="http://www.w3.org/2001/XMLSchema"
       xmlns:fn="http://www.w3.org/2005/xpath-functions"
       xmlns:car="java:com.altova.extfunc.Car" >
<xsl:output exclude-result-prefixes="fn car xsl fo xs"/>

<xsl:template match="/">
   <a>
   <xsl:value-of select="car:getVehicleType()"/>
   </a>
</xsl:template>

</xsl:stylesheet>
```

クラスファイルが参照され、XSLT/XQuery ファイルがクラスファイルと同じフォルダーに収められている

下のサンプルでは `Car` クラスの `getVehicleType()` メソッドが呼び出されます。以下を述べることができます: (i) `Car` クラスは次のフォルダー内にあります: `JavaProject/com/altova/extfunc` 。(ii) このフォルダーは下のサンプルの現在のフォルダー内に存在します。XSLT ファイルもフォルダー `JavaProject/com/altova/extfunc` 内にあります。

```
<xsl:stylesheet version="2.0"
       xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
       xmlns:xs="http://www.w3.org/2001/XMLSchema"
       xmlns:fn="http://www.w3.org/2005/xpath-functions"
       xmlns:car="java:Car" >
<xsl:output exclude-result-prefixes="fn car xsl fo xs"/>

<xsl:template match="/">
   <a>
   <xsl:value-of select="car:getVehicleType()"/>
   </a>
</xsl:template>

</xsl:stylesheet>
```

クラスファイルがパッケージされXSLT/XQuery ファイルがランダムな場所に収められている

以下の例では、com.altova.extfunc パッケージにある Car クラスの getVehicleColor() メソッドが呼び出されています。com.altova.extfunc パッケージは JavaProject という名前のフォルダーに置かれており、XSLT ファイルが任意の場所に配置されています。この場合、以下のような構文でパッケージの場所をクエリ文字列として URI 内にて指定する必要があります:

```
java:classname[?path=uri-of-package]
```

ここで、

java: によりユーザー定義の Java 関数が呼ばれていることを表します。
uri-of-package は Java パッケージの URI です。
classname は目的のメソッドが含まれているクラス名です。

クラスは名前空間 URI により特定され、名前空間がメソッド呼び出しのプレフィックスで使用されます。以下の例ではカレントディレクトリ以外にあるクラスファイルへのアクセスを行うことができます。

```
<xsl:stylesheet version="2.0"
      xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
      xmlns:xs="http://www.w3.org/2001/XMLSchema"
      xmlns:fn="http://www.w3.org/2005/xpath-functions"
      xmlns:car="java:com.altova.extfunc.Car?path=file:///C:/JavaProject/" >

<xsl:output exclude-result-prefixes="fn car xsl xs"/>

<xsl:template match="/">
   <xsl:variable name="myCar" select="car:new('red')" />
   <a><xsl:value-of select="car:getCarColor($myCar)"/></a>
</xsl:template>

</xsl:stylesheet>
```

クラスファイルがパッケージではなく、XSLT/XQuery ファイルがランダムな場所に収められている

以下の例では、com.altova.extfunc パッケージにある Car クラスの getCarColor() メソッドが呼び出されています。com.altova.extfunc パッケージは JavaProject という名前のフォルダーに置かれており、XSLT ファイルが任意の場所に配置されています。以下のような構文で、クラスファイルの場所をクエリ文字列として URI 内にて指定する必要があります:

```
java:classname[?path=uri-of-classfile]
```

ここで

java: によりユーザー定義の Java 関数が呼ばれていることを表します。
uri-of-classfile は Java パッケージの URI です。
classname は目的のメソッドが含まれているクラス名です。

クラスは名前空間 URI により特定され、名前空間はメソッド呼び出しのプレフィックスで使用されます。以下の例ではカレントディレクトリ以外にあるクラスファイルへのアクセスを行うことができます。

```
<xsl:stylesheet version="2.0"
      xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
      xmlns:xs="http://www.w3.org/2001/XMLSchema"
      xmlns:fn="http://www.w3.org/2005/xpath-functions"
      xmlns:car="java:Car?path=file:///C:/JavaProject/com/altova/extfunc/" >
```


(C) 2015-2021 Altova GmbH

```
<xsl:output exclude-result-prefixes="fn car xsl xs"/>

<xsl:template match="/">
   <xsl:variable name="myCar" select="car:new('red')" />
   <a><xsl:value-of select="car:getCarColor($myCar)"/></a>
</xsl:template>

</xsl:stylesheet>
```

メモ パスが外部関数により与えられている場合、ClassLoader により パスが追加されます。

19.1.2.2.1.2 ユーザー定義の JAR ファイル

JAR ファイル経由でアクセスが行われた場合、以下の構文により JAR ファイルの URI を指定する必要があります:

```
xmlns:classNS="java:classname?path=jar:uri-of-jarfile!/"
```

クラスの識別を行う名前空間 URI のプレフィックスを使用してメソッドが呼び出されます: `classNS:method()`

上の例に対する説明は以下のとおりです:

`java:` 関数が呼び出されていることを表します。
`classname` がユーザー定義されたクラスの名前になります。
`?` はクラス名と パスを分離するために使用されます。
`path=jar:` により、JAR ファイルへの パスが与えられていることを示します。
`uri-of-jarfile` は JAR ファイルの URI となります。
`!/` は、終了を表すデリミタとなります。
`classNS:method()` により、メソッドの呼び出しが行われます。

その他にも、メソッド名とともにクラス名を与えることができます。構文の例を以下に示します:

```
xmlns:ns1="java:docx.layout.pages?
path=jar:file:///c:/projects/docs/docx.jar!/"
ns1:main()

xmlns:ns2="java?path=jar:file:///c:/projects/docs/docx.jar!/"
ns2:docx.layout.pages.main()
```

以下に JAR ファイルを使った Java 拡張関数を呼び出す XSLT サンプルを記します:

```
<xsl:stylesheet version="2.0"
      xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
      xmlns:xs="http://www.w3.org/2001/XMLSchema"
      xmlns:fn="http://www.w3.org/2005/xpath-functions"
      xmlns:car="java?path=jar:file:///C:/test/Car1.jar!/" >
<xsl:output exclude-result-prefixes="fn car xsl xs"/>

<xsl:template match="/">
      <xsl:variable name="myCar" select="car:Car1.new('red')" />
      <a><xsl:value-of select="car:Car1.getCarColor($myCar)"/></a>
</xsl:template>

<xsl:template match="car"/>
```

```
</xsl:stylesheet>
```

メモ: 拡張関数によりパスが与えられている場合、ClassLoader にパスが追加されます。

19.1.2.2.1.3 Java：コンストラクター

拡張関数を使用することで Java コンストラクターを呼び出すことができます。`new()` により全てのコンストラクターを呼び出すことができます。

Java コンストラクターの呼び出し結果を、黙示的に XPath/XQuery データ型へ変換できる場合、Java 拡張関数により XPath/XQuery データ型のシーケンスが返されます。Java コンストラクターの呼び出し結果が XPath/XQuery データ型へ変換できない場合、値を返すクラス名でラップした Java オブジェクトがコンストラクターにより作成されます。例えば、`java.util.Date` クラスに対するコンストラクターが呼び出された場合（`java.util.Date.new()`）、`java.util.Date` を持つオブジェクトが返されます。返されたオブジェクトのレキシカルフォーマットは、XPath データ型のレキシカルフォーマットにマッチしない場合もあり、目的の XPath データ型に対するレキシカルフォーマットへ値の変換を行い、その後目的の XPath データ型へ変換を行う必要があります。

コンストラクターにより作成された Java オブジェクトにより2つのことが行えます:

- 変数への割り当てを行うことができます:
  ```
  <xsl:variable name="currentdate" select="date:new()"
  xmlns:date="java:java.util.Date" />
  ```
- 拡張関数への受け渡しを行うことができます（*インスタンスメソッドならびにインスタンスフィールド* を参照ください）:
  ```
  <xsl:value-of select="date:toString(date:new())"
  xmlns:date="java:java.util.Date" />
  ```

19.1.2.2.1.4 Java：静的メソッドと静的フィールド

静的メソッドは、Java 名ならびにメソッドの引数により直接呼び出すことができます。`E` や `PI` といった定数の静的フィールド（引数を持たないメソッド）は、引数を指定することなくアクセスすることができます。

XSLT の例

静的メソッドならびにフィールドを呼び出す例を以下に示します:

```
<xsl:value-of xmlns:jMath="java:java.lang.Math"
              select="jMath:cos(3.14)" />

<xsl:value-of xmlns:jMath="java:java.lang.Math"
              select="jMath:cos( jMath:PI() )" />

<xsl:value-of xmlns:jMath="java:java.lang.Math"
              select="jMath:E() * jMath:cos(3.14)" />
```

上の拡張関数は `prefix:fname()` という形式を使用していることに注意してください。3つの例にあるプレフィックスは全て jMath: となっており、このプレフィックスは、`java:java.lang.Math` という名前空間 URI に関連付けられています。名前空間 URI は java: で開始しなければなりません。上の例では、クラス名（`java.lang.Math`）を含むように拡張されています。拡張関数の fname() 部は（`java.lang.Math` のような）public クラスにマッチする必要があり、その後 public な静的メソッドが引数とともに続くか（例: `cos(3.14)`）、public な静的フィールドが続きます（例: PI()）。

上の例では、クラス名が名前空間 URI に含まれています。クラス名が名前空間 URI に含まれていない場合、以下の例にあるように、拡張関数の `fname()` 部にて追加する必要があります:

```
<xsl:value-of xmlns:java="java:"
              select="java:java.lang.Math.cos(3.14)" />
```

XQuery の例

XQuery における似たようなサンプルを以下に示します:

```
<cosine xmlns:jMath="java:java.lang.Math">
   {jMath:cos(3.14)}
</cosine>
```

19.1.2.2.1.5 Java：インスタンスメソッドとインスタンスフィールド

メソッド呼び出しの第一引数としてパースされる Java オブジェクトが、インスタンスメソッドには与えられています。このような Java オブジェクトは通常、拡張関数を使うことで作成される(例：コンストラクターの呼び出し)か、スタイルシートパラメーター/変数により作成されます。以下に XSLT サンプルを示します:

```
<xsl:stylesheet version="1.0" exclude-result-prefixes="date"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
  xmlns:date="java:java.util.Date"
  xmlns:jlang="java:java.lang">
  <xsl:param name="CurrentDate" select="date:new()"/>
  <xsl:template match="/">
      <enrollment institution-id="Altova School"
                  date="{date:toString($CurrentDate)}"
                  type="{jlang:Object.toString(jlang:Object.getClass( date:new() ))}">
      </enrollment>
  </xsl:template>
</xsl:stylesheet>
```

上の例では、ノード `enrollment/@type` の値が以下のように作成されます:

1. `java.util.Date` クラスに対するオブジェクトがコンストラクター(`date:new()` コンストラクター)とともに作成されます。
2. `jlang.Object.getClass` メソッドの引数として Java オブジェクトがパースされます。
3. `getClass` メソッドにより得られたオブジェクトが `jlang.Object.toString` メソッドの引数としてパースされます。

結果(`@type` の値)は `java.util.Date` を持った文字列となります。

インスタンスフィールドは、引数としてインスタンスフィールドへ渡される Java オブジェクトではないという点で、理論的にはインスタンスメソッドと異なります。パラメーターや変数がその代わりに引数として渡されますが、パラメーター/変数そのものに Java オブジェクトから返された値が含まれている場合もあります。例えば、`CurrentDate` パラメーターには `java.util.Date` クラスのコンストラクターから返された値が含まれます。この値は引数として、`date:toString` インスタンスメソッドへ渡され、`/enrollment/@date` の値として使用されます。

19.1.2.2.1.6 データ型：XPath/XQuery から Java へ

XPath/XQuery 条件式内部から Java 関数が呼び出された場合、複数ある同名の Java クラスのうち、どのクラスが呼び出されたのか決定するのに、関数へ渡される引数のデータ型が重要になります。

Java では、以下のルールが適用されます:

- 同名の Java メソッドが2つ以上あり、それぞれが違う数の引数を受け取る場合、呼び出しに使用されている引数の数に一番マッチするメソッドが選択されます。
- XPath/XQuery の文字列、数値、boolean データ型は、黙示的に対応する Java データ型へ変換されます（以下のリストを参照）。与えられた XPath/XQuery 型が2つ以上の Java 型へ変換できる場合（例: `xs:integer`）、選択されたメソッドにて宣言されている Java 型が使用されます。例えば、呼び出された Java メソッドが `fx(decimal)` で、与えられた XPath/XQuery データ型が `xs:integer` の場合、`xs:integer` が Java の `decimal` データ型へ変換されます。

以下のテーブルに、XPath/XQuery の文字列、数値、boolean 型から Java データ型への黙示的な変換リストを示します。

`xs:string`	`java.lang.String`
`xs:boolean`	`boolean`（プリミティブ型）, `java.lang.Boolean`
`xs:integer`	`int`, `long`, `short`, `byte`, `float`, `double`, ならびに、`java.lang.Integer` のようなこれらのラッパークラス
`xs:float`	`float`（プリミティブ型）, `java.lang.Float`, `double`（プリミティブ型）
`xs:double`	`double`（プリミティブ型）, `java.lang.Double`
`xs:decimal`	`float`（プリミティブ型）, `java.lang.Float`, `double`（プリミティブ型）, `java.lang.Double`

上のリストにある XML スキーマデータ型（ならびに XPath や XQuery で使用されているデータ型）のサブタイプも、対応する祖先のサブタイプとして Java のデータ型へ変換されます。

場合によっては、与えられた情報から正しい Java メソッドを選択することができない場合もあります。例えば、以下のような場合を考えてみましょう:

- 与えられた引数が10という値を持った xs:untypedAtomic 型で、`mymethod(float)` メソッドへ渡されるのを意図している。
- しかし、そのクラスには別のデータ型を取る `mymethod(double)` というメソッドも存在する。
- メソッド名が同じで、与えられた型（`xs:untypedAtomic`）も `float` と `double` の両方に変換することができるため、`xs:untypedAtomic` が `float` ではなく `double` に変換される可能性もある。
- 結果として、意図したメソッドは選択されず、予期しない動作結果が招く可能性がある。この問題を回避するには、意図したメソッドを使用するユーザー定義のメソッドを別の名前で新たに作成する必要があります。

上のリストでカバーされていない型（例: `xs:date`）は変換されず、エラーとなります。しかし場合によっては、Java コンストラクターを使用して、目的の Java データ型を作成することが可能だということも留意してください。

19.1.2.2.1.7 データ型：Java から XPath/XQuery へ

Java メソッドにより値が返され、値のデータ型が文字列、数値、または boolean 型の場合、対応する XPath/XQuery 型への変換が行われます。例えば、Java の `java.lang.Boolean` や `boolean` データ型は `xsd:boolean` へ変換されます。

関数から返された一次元配列は、シーケンス（sequence）に展開されます。2 次元以上の配列は変換されることが無いため、ラップして使用するべきでしょう。

Java オブジェクトや文字列、数値、boolean 以外のデータ型がラップされて返された場合、最初に（例えば `toString` という）Java メソッドを使用して Java オブジェクトを文字列へ変換することで、目的の XPath/XQuery 型への変換を行います。XPath/XQuery では、文字列を目的となる型のレキシカルフォーマットへ変換し、目的の型への変換を（例えば `cast as` 式を使用することで）行うことができます。

19.1.2.2.2 .NET 拡張関数

.NET プラットフォームにて作業を行なっている場合、.NET 言語（例えば C#）で記述された拡張関数を使用することができます。.NET 拡張関数は XPath や XQuery 条件式内部から使用することができ、.NET クラス内部にあるコンストラクターや（static または インスタンス変数）プロパティを呼び出すことができます。

`get_PropertyName()` 構文を使用することにより .NET クラスのプロパティを呼び出すことができます。

このセクションは、以下のサブセクションにより構成されています:

- .NET コンストラクター
- .NET：静的メソッドならびに静的フィールド
- .NET：インスタンスメソッドとインスタンスフィールド
- データ型：XPath/XQuery から .NET へ
- データ型：.NET から XPath/XQuery へ

拡張関数のフォーム

XPath/XQuery 条件式にある拡張関数は、`prefix:fname()` の形式を取る必要があります。

- `prefix:` 部は、呼び出されている .NET クラスを特定する URI となります。
- `fname()` 部により、.NET クラス内にあるコンストラクター、プロパティ、または（静的または インスタンス）メソッドが特定され、必要な場合は引数が与えられます。
- URI は `clitype:` で開始する必要があり、これにより関数が .NET 拡張関数であることが認識されます。
- 拡張関数の `prefix:fname()` 形式は、システムクラスならびにロードされたアセンブリとともに使用することもできます。しかしクラスをロードする必要がある場合、必要な情報が含まれる パラメーターが必要になります。

パラメーター

アセンブリをロードするには以下の パラメーターを使用してください:

`asm`	ロードするアセンブリの名前。
`ver`	バージョン番号（ピリオドにより分離された最大 4 桁の整数）。

sn	アセンブリ厳密名のキートークン(16新数の数値)。
from	ロードするアセンブリ(DLL) の場所を特定するURI。URI が相対パスの場合、XSLT やXQuery ドキュメントに対して相対的になります。このパラメーターが指定された場合、その他のパラメーターが無視されます。
partialname	アセンブリ名の一部。Assembly.LoadWith.PartialName() へ渡され、アセンブリへのロードが試みられます。 partialname が指定された場合、その他のパラメーターが無視されます。
loc	例えばen-US というロケール。デフォルトはneutral です。

アセンブリがDLL からロードされる場合、from パラメータが使用して、sn パラメーターは使用しないでください。アセンブリがグローバルアセンブリキャッシュ(GAC) からロードされる場合、sn パラメータを使用して from パラメーターは使用しないでください。

最初のパラメーターの前に疑問符 (?) を挿入し、パラメーター同士はセミコロンで分離する必要があります。パラメーター名へ値を受け渡すには、統合符号 (=) を使用します(以下の例を参照ください)。

名前空間宣言の例

XSLT において、システムクラスSystem.Environment を特定する名前空間宣言の例を以下に示します:

```
xmlns:myns="clitype:System.Environment"
```

XSLT において、ロードするクラスをTrade.Forward.Scrip として特定する名前空間宣言の例を以下に示します:

```
xmlns:myns="clitype:Trade.Forward.Scrip?asm=forward;version=10.6.2.1"
```

XQuery において、システムクラス MyManagedDLL.testClass を特定する名前空間宣言の例を以下に示します。2つのケースが考えられます:

1. アセンブリがGAC からロードされた場合:
```
declare namespace cs="clitype:MyManagedDLL.testClass?asm=MyManagedDLL;
       ver=1.2.3.4;loc=neutral;sn=b9f091b72dccfba8";
```
2. アセンブリがDLL からロードされた場合(完全参照と一部の参照):
```
       declare namespace cs="clitype:MyManagedDLL.testClass?from=file:///C:/Altova
       Projects/extFunctions/MyManagedDLL.dll;

   declare namespace cs="clitype:MyManagedDLL.testClass?from=MyManagedDLL.dll;
```

XSLT の例

システムクラスSystem.Math 内の関数を呼び出すための完全なXSLT の例を以下に示します:

```
<xsl:stylesheet version="2.0"
   xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
   xmlns:xs="http://www.w3.org/2001/XMLSchema"
   xmlns:fn="http://www.w3.org/2005/xpath-functions">
   <xsl:output method="xml" omit-xml-declaration="yes" />
   <xsl:template match="/">
      <math xmlns:math="clitype:System.Math">
         <sqrt><xsl:value-of select="math:Sqrt(9)"/></sqrt>
         <pi><xsl:value-of select="math:PI()"/></pi>
         <e><xsl:value-of select="math:E()"/></e>
```


(C) 2015-2021 Altova GmbH

```
        <pow><xsl:value-of select="math:Pow(math:PI(), math:E())"/></pow>
      </math>
   </xsl:template>
</xsl:stylesheet>
```

math 要素にある名前空間宣言により、math: プレフィックスと clitype:System.Math URI が関連付けられます。URI の最初にある clitype: により、それ以降の記述がシステムクラスまたはロードされたクラスを特定するものであることが示されます。XPath 条件式にある math: プレフィックスにより、拡張関数が URI（そしてクラス） System.Math に関連付けられます。拡張関数により、System.Math クラス内のメソッドが特定され、必要な場所に引数が与えられます。

XQuery の例

上の XSLT に対する例と同様の XQuery 例を以下に示します:

```
<math xmlns:math="clitype:System.Math">
   {math:Sqrt(9)}
</math>
```

上の XSLT と同様に、名前空間宣言により .NET クラス（この場合はシステムクラス）が特定されます。XQuery 式により呼び出されるメソッドが特定され、引数が与えられます。

19.1.2.2.2.1 .NET コンストラクター

拡張関数を使用することで、.NET コンストラクターを呼び出すことができます。new() により全てのコンストラクターを呼び出すことができます。クラス内に2つ以上のコンストラクターがある場合、与えられた引数の数が最もマッチするコンストラクターが選択されます。与えられた引数に対してマッチするコンストラクターが見つからない場合、"No constructor found'" エラーが返されます。

XPath/XQuery データ型を返すコンストラクター

.NET コンストラクター呼び出しの結果が XPath/XQuery データ型へ黙示的に変換することができる場合、.NET 拡張関数から XPath/XQuery データ型のシーケンスが返されます。

.NET オブジェクトを返すコンストラクター

.NET コンストラクター呼び出しの結果が XPath/XQuery データ型へ適切に変換できない場合、値を返すクラス名でラップした .NET オブジェクトがコンストラクターにより作成されます。例えば、System.DateTime クラスのコンストラクターが（System.DateTime.new() により）呼ばれた場合 System.DateTime 型を持つオブジェクトが返されます。

返されたオブジェクトのレキシカルフォーマットは、目的の XPath データ型と違っている場合があります。その場合、返された値を: (i) 目的の XPath データ型のレキシカルフォーマットへ変換し、(ii) 目的の XPath データ型へキャストする必要があります。

コンストラクターにより作成された .NET オブジェクトに対して3つのことを行うことができます:

- 変数内で使用することができます:
  ```
  <xsl:variable name="currentdate" select="date:new(2008, 4, 29)"
  xmlns:date="clitype:System.DateTime" />
  ```
- 拡張関数へ渡すことができます（*インスタンスメソッドとインスタンスフィールド* を参照ください）:
  ```
  <xsl:value-of select="date:ToString(date:new(2008, 4, 29))"
  xmlns:date="clitype:System.DateTime" />
  ```
- 文字列、数値、または boolean へ変換することができます:

```
<xsl:value-of select="xs:integer(date:get_Month(date:new(2008, 4, 29)))"
  xmlns:date="clitype:System.DateTime" />
```

19.1.2.2.2.2 .NET: 静的メソッドと静的フィールド

メソッド名と引数を与えることで、静的メソッドを直接呼び出すことができます。呼び出しに使用される名前は、クラス内にある public static メソッドと完全に一致する必要があります。関数の呼び出しに使用されたメソッド名と引数の数にマッチするものがクラス内に複数ある場合、与えられた引数が評価され、最もマッチするものが選択されます。マッチする結果が得られない場合、エラーが返されます。

メモ: .NET クラス内にあるフィールドは、引数を持たないメソッドとしてみなされます。プロパティは `get_PropertyName()` 構文により呼び出されます。

例

1つの引数とともにメソッド (`System.Math.Sin(arg)`) を呼び出す XSLT サンプルを以下に示します:

```
<xsl:value-of select="math:Sin(30)" xmlns:math="clitype:System.Math"/>
```

(引数なしのメソッドとしてみなされる) フィールド (`System.Double.MaxValue()`) を呼び出す XSLT サンプルを以下に示します:

```
<xsl:value-of select="double:MaxValue()" xmlns:double="clitype:System.Double"/>
```

(`get_PropertyName()` 構文を使って) プロパティ (`System.String()`) を呼び出す XSLT サンプルを以下に示します:

```
<xsl:value-of select="string:get_Length('my string')"
xmlns:string="clitype:System.String"/>
```

1つの引数とともにメソッド ((`System.Math.Sin(arg)`) を呼び出す XQuery サンプルを以下に示します:

```
<sin xmlns:math="clitype:System.Math">
   { math:Sin(30) }
</sin>
```

19.1.2.2.2.3 .NET: インスタンスメソッドとインスタンスフィールド

インスタンスメソッドには、メソッド呼び出しの第一引数として .NET オブジェクトが渡されます。通常この .NET オブジェクトは、拡張関数 (例えばコンストラクター呼び出し) またはスタイルシートパラメーター/変数により作成されます。以下に XSLT の例を示します:

```
<xsl:stylesheet version="2.0"
   xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
   xmlns:xs="http://www.w3.org/2001/XMLSchema"
   xmlns:fn="http://www.w3.org/2005/xpath-functions">
   <xsl:output method="xml" omit-xml-declaration="yes"/>
   <xsl:template match="/">
      <xsl:variable name="releasedate"
         select="date:new(2008, 4, 29)"
         xmlns:date="clitype:System.DateTime"/>
```

```
      <doc>
         <date>
            <xsl:value-of select="date:ToString(date:new(2008, 4, 29))"
               xmlns:date="clitype:System.DateTime"/>
         </date>
         <date>
            <xsl:value-of select="date:ToString($releasedate)"
               xmlns:date="clitype:System.DateTime"/>
         </date>
      </doc>
   </xsl:template>
</xsl:stylesheet>
```

上の例では、System.DateTime コンストラクター(`new(2008, 4, 29)`) が、System.DateTime 型の.NET オブジェクトの作成に使用されます。このオブジェクトは、最初に`releasedate` 変数の値として、次に`System.DateTime.ToString()` メソッドの引数として作成されます。`System.DateTime.ToString()` インスタンスメソッドは、`System.DateTime` コンストラクターの (new(2008, 4, 29)) における引数として2度呼び出されます。これらインスタンスにおいて `releasedate` 変数が.NET オブジェクトを取得するのに使用されます。

インスタンスメソッドとインスタンスフィールド

インスタンスメソッドとインスタンスフィールドの違いは理論的なものです。インスタンスメソッドでは.NET オブジェクトが直接引数に渡され、インスタンスフィールドではパラメーターや変数が(.NET オブジェクトそのものを含めることはできるものの) 代わりに渡されます。例えば上の例では、`releasedate` 変数に.NET オブジェクトが含まれており、この変数が2番目のdate 要素コンストラクターにて `ToString()` の引数として渡されます。そのため、最初のdate 要素にある `ToString()` インスタンスがインスタンスメソッドであるのに対し、2番目はインスタンスフィールドとしてみなされます。両方のインスタンスで求められる結果は等価です。

19.1.2.2.2.4 データ型:XPath/XQuery から .NET へ

.NET 拡張関数がXPath/XQuery 条件式内部で使用された場合、複数ある.NET メソッドのうち、どれが呼び出されたのか決定するのに関数の引数に使用されるデータ型が重要になります。

.NET では、以下のルールが適用されます:

- クラス内に同名のメソッドが2つ以上ある場合、呼び出しに使用された引数の数がマッチするメソッドだけが、呼び出される関数の候補に狭められます。
- XPath/XQuery の文字列、数値、boolean データ型は黙示的に対応する.NET データ型へ変換されます(以下のリストを参照)。与えられたXPath/XQuery 型が2つ以上の.NET 型へ変換できる場合(例: `xs:integer`)、選択されたメソッドにて宣言されている.NET 型が使用されます。例えば、呼び出された.NET メソッドがfx(fouble) で、与えられたXPath/XQuery データ型がxs:integer の場合、`xs:integer` が.NET の`double` データ型へ変換されます。

以下のテーブルに、XPath/XQuery の文字列、数値、boolean 型から.NET データ型へ行われる黙示的な変換リストを示します。

`xs:string`	`StringValue, string`
`xs:boolean`	`BooleanValue, bool`
`xs:integer`	`IntegerValue, decimal, long, integer, short, byte, double, float`
`xs:float`	`FloatValue, float, double`
`xs:double`	`DoubleValue, double`

xs:decimal	DecimalValue, decimal, double, float

上のリストにあるXML スキーマ型(ならびにXPath やXQuery で使用されているデータ型)のサブタイプも、対応する祖先のサブタイプとして.NET のデータ型へ変換されます。

場合によっては、与えられた情報から正しい.NET メソッドを選択することができない場合もあります。例えば、以下のような場合を考えてみましょう:

- 与えられた引数が0ある xs:untypedAtomic 値で、mymethod(float) メソッドへ渡されるのを意図している。
- しかし、そのクラスには別のデータ型をとる mymethod(double) というメソッドも存在する。
- メソッド名が同じで、与えられた型(xs:untypedAtomic)も float と double の両方に変換することができるため、xs:untypedAtomic が float ではなく double に変換される可能性もある。
- 結果として、意図したメソッドは選択されず、予期しない動作結果が招く可能性がある。この問題を回避するには、意図したメソッドを使用するユーザー定義のメソッドを別の名前で新たに作成する必要があります。

上のリストでカバーされていない型(例: xs:date)は変換されずエラーとなります。

19.1.2.2.2.5 データ型:.NET から XPath/XQuery へ

.NET メソッドにより値が返される際に値のデータ型が文字列、数値、またはboolean 型の場合、対応するXPath/XQuery 型への変換が行われます。例えば、.NET のdecimalデータ型はxsd:decimal へ変換されます。

.NET オブジェクトや文字列、数値、boolean 以外のデータ型が返された場合、最初に(例えばSystem.DateTime.ToString()といった).NET メソッドを使用して.NET オブジェクトを文字列へ変換します。XPath/XQuery では、文字列を目的となる型のレキシカルフォーマットへ変換し、目的の型への変換を(例えばcast as 式を使用することで)行うことができます。

19.1.2.2.3 XSLT に対する MSXSL スクリプト

<msxsl:script> 要素にはユーザー定義の関数や変数が含まれており、XSLT スタイルシート内のXPath 条件式内部から呼び出しを行うことができます。<msxsl:script> はトップレベル要素で、<xsl:stylesheet> または<xsl:transform> の子要素である必要があります。

<msxsl:script> 要素はurn:schemas-microsoft-com:xslt 名前空間内に存在する必要があります(以下を参照ください)。

スクリプト言語と名前空間

ブロック内で使用されるスクリプト言語は<msxsl:script> 要素のlanguage 属性にて指定され、XPath 条件式における関数の呼び出しに対して使用される名前空間はimplements-prefix 属性により特定されます(以下を参照)。

```
<msxsl:script language="scripting-language" implements-prefix="user-namespace-prefix">

   function-1 or variable-1
   ...
   function-n or variable-n

</msxsl:script>
```

<msxsl:script> 要素はWindows Scripting Runtime を使ったやりとりを行うため、お使いのコンピューターにインストールされた言語だけしか<msxsl:script> 要素では使用することができません。MSXSL スクリプトを使用するには.NET Framework 2.0 以上のプラットフォームをインストールする必要があります。結果として<msxsl:script> 言語から.NET スクリプト言語を使用することができます。

HTML の<script> 要素における language 属性と同じ値が language 属性では受理されます。language 属性が指定されていない場合、Microsoft JScript がデフォルトとして想定されます。

implements-prefix 属性には名前空間スコープ内で宣言されたプレフィックスが与えられます。通常この名前空間は関数ライブラリのために予約されたユーザーの名前空間となります。<msxsl:script>要素内で定義された全ての関数ならびに変数は、implements-prefix 属性にて指定されたプレフィックスで特定される名前空間に収められます。XPath 条件式内部から関数が呼ばれる場合、完全修飾関数名が同じ名前空間内に関数として定義されていなければなりません。

サンプル

<msxsl:script> 要素内で定義された関数を使用するXSLT スタイルシートの例を以下に示します:

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="2.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
   xmlns:xs="http://www.w3.org/2001/XMLSchema"
   xmlns:fn="http://www.w3.org/2005/xpath-functions"
   xmlns:msxsl="urn:schemas-microsoft-com:xslt"
   xmlns:user="http://mycompany.com/mynamespace">

  <msxsl:script language="VBScript" implements-prefix="user">
    <![CDATA[
    ' Input:A currency value: the wholesale price
    ' Returns:The retail price: the input value plus 20% margin,
    ' rounded to the nearest cent
    dim a as integer  = 13
    Function AddMargin(WholesalePrice) as integer
      AddMargin = WholesalePrice * 1.2 + a
    End Function
  ]]>
  </msxsl:script>

  <xsl:template match="/">
    <html>
      <body>
        <p>
          <b>Total Retail Price =
            $<xsl:value-of select="user:AddMargin(50)"/>
          </b>
          <br/>
          <b>Total Wholesale Price =
            $<xsl:value-of select="50"/>
          </b>
        </p>
      </body>
    </html>
  </xsl:template>
</xsl:stylesheet>
```

データ型

スクリプトブロックとのやりとりに使用されるパラメーターの値はXPath データ型に限定されます。スクリプトブロック内にある関数にてやりとりされるデータや変数に、この制限はありません。

アセンブリ

msxsl:assembly 要素を使用することで、アセンブリをスクリプト内部へインポートすることができます。アセンブリは名前やURI により特定されます。アセンブリのインポートは、コンパイル時に行われます。以下に msxsl:assembly 要素の簡単な使用例を示します:

```
<msxsl:script>
       <msxsl:assembly name="myAssembly.assemblyName" />
       <msxsl:assembly href="pathToAssembly" />

       ...

</msxsl:script>
```

アセンブリ名は、以下のような完全な名前でも::

```
"system.Math, Version=3.1.4500.1 Culture=neutral PublicKeyToken=a46b3f648229c514"
```

`"myAssembly.Draw"` のような短い名前でも指定できます。

名前空間

msxsl:using 要素により名前空間の宣言を行うことができます。これにより、スクリプト内において名前空間無しでアセンブリクラスを使用することができ、タイピングの手間を軽減することができます。以下に `msxsl:using` 要素の簡単な使用例を示します:

```
<msxsl:script>
       <msxsl:using namespace="myAssemblyNS.NamespaceName" />

       ...

</msxsl:script>
```

`namespace` 属性の値は名前空間の名前となります。

(C) 2015-2021 Altova GmbH

19.2 技術データ

このセクションは、ソフトウェアの技術面に関する役に立つ背景情報を含んでいます。以下のように整理されています:
OS とメモリ要件Altova XML バリデーターAltova XSLT とXQuery エンジンUnicode のサポートインターネットの使用

19.2.1 OS とメモリ要件

オペレーティングシステム

Altova ソフトウェア アプリケーションは、以下のプラットフォームでご使用いただけます:
プラットフォーム更新済みのW indows 7 SP1、W indows 8、W indows 10プラットフォーム更新済みのW indows Server 2008 R2 SP1 または以降 メモリ
ソフトウェアがC++ で書かれているため、Java Runtime Environment をダウンロードする必要はなく、Java ベースのアプリケーションに比べ、通常少ないメモリを必要とします。しかしながら、各ドキュメントは完全に解析するため、また、ビューと編集の速度を向上するためにメモリにダウンロードされます。メモリの要件は、ドキュメントのサイズを増やします。

メモリ要件は、制限のない「元に戻す」履歴により影響を受けます。大きなドキュメントの大きなセクションの切り取り、貼り付け操作を繰り返し行うと、使用できるメモリがすぐに消費されます。

19.2.2 Altova XML バリデーター

XML ドキュメントを開くと、アプリケーションは、内蔵のXML バリデーターを使用して、指定されている場合、スキーマに対して整形式をチェックし、ツリーとインフォセットを作成します。XML バリデーターは、ドキュメントを編集する際にインテリジェントな編集ヘルプを提供し、発生する検証エラーを表示するために使用されます。

内蔵のXML バリデーターは、W 3C のXML スキーマ1.0 と1.1 仕様の最終勧告を実装しています。. New developments recommended by the W 3C XML スキーマ作業グループにより勧告される新しい項目は、XML バリデーターに継続的に組み込まれるため、Altova 製品は最高水準の開発環境を届けることができます。

19.2.3 Altova XSLT と XQuery エンジン

Altova 製品 は、Altova XSLT 1.0、2.0、および3.0 エンジンとAltova XQuery 1.0 と3.1 エンジンを使用しています。各エンジンのためのドキュメントと実装に固有の振る舞いに関しては、製品で使用されるエンジンの各ドキュメントの付属書(エンジン情報) で確認することができます。

メモ Altova MapForce は、XSLT 1.0、2.0 およびXQuery 1.0 エンジンを使用したコードを生成します。

19.2.4 Unicode のサポート

Altova XML 製品は、Unicode を完全にサポートします。XML ドキュメントを編集するには、ドキュメント内で使用されているUnicode 文字をサポートするフォントが必要です。

フォントの多くは、Unicode 範囲全体の特定のサブセットを含む場合があり、このため、通常は対応する表記システムをターゲットとします。テキストの一部が、文字化けして表示された場合、理由としては、選択されたフォントが必要とする字形を含まない場合があげられます。です

から、特に、異なる言語、または、異なる言語システムのXMLドキュメントを編集する場合、範囲全体をカバーするフォントを使用することが役にたちます。典型的なUnicode フォントは、Windows PC のArial Unicode MS で確認することができます。

アプリケーションフォルダーの/Examples フォルダー内で、異なる言語システムで表記された次の文章を含むUnicodeUTF-8.html というXHTML ファイルを確認してください:
When the world wants to talk, it speaks Unicode Wenn die Welt miteinander spricht, spricht sie Unicode
世界的に話すなら、Unicode です。) XHTML ファイルを開くと、Unicode の可能性を確認することができ、使用中のPC の使用することのできるフォントによりサポートされている表記システムが表示されます。

19.2.5 インターネットの使用

Altova アプリケーションは、次の状況でインターネット 接続を開始します:

- 登録ダイアログ(「**ヘルプ | ソフトウェアのライセンス認証**」)内の「評価キーコードをリクエスト」をクリックした場合、登録ダイアログボックス内の3つのフィールドが通常のhttp (ポート 80)接続を使用し、サーバーに転送され、無料の評価キーが顧客に通常のSMTP 電子メールを使用して送り返されます。
- Altova 製品の一部では、インターネットからファイルを開くことができます(「**ファイル | 開く | URL に切り替える**」)。この場合、ドキュメントは、次のプロトコルメソッドと接続の1つを使用して取得されます: HTTP (通常、ポート 80)、FTP (通常、ポート 20/21)、HTTPS (通常、ポート 443)。HTTP サーバーをポート 8080 で作動することもできます(URL ダイアログ内で、サーバー名とコロンの後にポートを指定します)。
- XML スキーマ、または、DTD を参照するXMLドキュメント、と、URL により指定されているドキュメントを開くと、参照されているスキーマドキュメントは、HTTP 接続(ポート 80) またはURL により指定されている他のプロトコル(上のポイント 2 参照)により抽出されます。XML ファイルが検証されている場合、スキーマドキュメントも抽出されます(オプションダイアログのファイルタブ内の(「**ツール | オプション**」))。アプリケーションに命令している場合、ドキュメントが開かれると検証が自動的に行われる場合もあります。
- WSDL とSOAP を使用するAltova アプリケーションでは、Web サービスを使用する接続は、WSDLドキュメントにより定義されています。
- XMLSpy 内で、「**電子メールで送信**」コマンドを使用する場合、(「**ファイル | 電子メールで送信**」) 現在選択されている範囲、または、ファイルは、ユーザーのマシンにインストールされているMAPI コンプライアント電子メールプログラムにより送信されます。
- ソフトウェアの荒いセンス認証 とLiveUpdate の一部として、Altova ソフトウェア使用許諾書内で更に詳しい説明を確認することができます。

19.3 ライセンス情報

このセクションには以下の内容が含まれています:

- ソフトウェアの配布に関する情報
- ソフトウェアのアクティベーションとライセンスの計測
- ソフトウェアの使用に関する使用許諾契約書

本製品を使用する前に、上記の情報をよくお読みください。ソフトウェアのインストール時に上記のすべての条件に同意したとみなされ、お客様は上記の条件に拘束されることを同意したとみなされます。

Altova ライセンスの内容を確認するには、Altova Web サイト の Altova 法的な情報のページ に移動してください。

19.3.1 電子的なソフトウェアの配布

この製品は電子的なソフトウェアの配布により利用することが可能で、この配布方法により、以下のユニークなメリットがあります:

- 購入を決定する前に、無料でソフトウェアを試用することができます。*(Note: Altova MobileTogether Designer に対してライセンスを無料で割り当てることができます)。*
- Once ソフトウェアの購入を決定した際には、Altova Web サイト にて注文を行います。すぐにライセンス登録された製品の使用を開始することができます。
- オンラインにて注文を行うと、常に最新のソフトウェアをご利用いただけます。
- 製品パッケージには包括的なヘルプシステムが画面上に表示されます。最新バージョンのユーザーマニュアルは https://www.altova.com/ja/ 上にあり、(i) HTML フォーマットによる閲覧、ならびに(ii) PDF フォーマットのダウンロードと印刷に対応しております。

30 日間の評価期間

この製品をダウンロードした後は、最大で30 日の間無料で製品の評価を行うことができます。20 日間を超えた頃から、製品がライセンス登録されていないことがソフトウェアにより表示されます。このメッセージはアプリケーションが起動されるたびに表示され、30 日間を超えてプログラムを使用するには、キーコードを含むライセンスファイルから提供される製品のライセンスを購入します。ライセンスファイルを製品のソフトウェアアクティベーションダイアログにアップロードして、製品をアンロックします。

https://shop.altova.com/ でライセンスを購入することができます

組織内でソフトウェアの評価を行う

評価版のソフトウェアを組織内のネットワークにて配布したい場合、またはインターネットに接続されていないコンピューターにてソフトウェアを使用する場合、どのような状態でも改変されていないことを条件に、セットアッププログラムおよびの配布を行うことが可能です。ソフトウェアインストーラーへアクセスした人は、例外なく30 日間の評価ライセンスキーコードをリクエストして、試用期間が経過した後は、製品を使い続けるためにライセンスの購入を行う必要があります。

19.3.2 ソフトウェアのアクティベーションとライセンスの計測

Altova のソフトウェアアクティベーションの一部として、ソフトウェアにより内部ネットワークまたはインターネットへの接続を行い、インストール時、登録時、Altova により使用されるライセンスサーバーの更新やライセンスの正当性を検証することで、ソフトウェアの不正な使用を防ぎ、顧客サービスを向上するため、ライセンスに関する情報を送信することがあります。アクティベーションにより、オペレーティングシステムや IP アドレス、

日付/時刻、ソフトウェアのバージョン、コンピュータの名前などのライセンスに関する情報が、お使いのコンピューターとAltova ライセンスサーバー間にてやり取りされます。

お使いのAltova 製品にはライセンス計測モジュールが内蔵されており、エンドユーザー使用許諾契約書の意図しない違反を防ぎます。お使いの製品はシングルユーザーまたはマルチユーザーとしてインストールされており、ライセンス計測モジュールにより、ライセンスされている数を超えたユーザーが同時に製品を使用することが無いことが保証されます。

このライセンス計測技術により、ローカルエリア接続（LAN）において、別々のコンピューター間で動作しているアプリケーションインスタンス間の通信が行われます。

シングルライセンス

ライセンス計測プロセスの一部としてアプリケーションが起動すると、ソフトウェアにより短いデータグラムがブロードキャストにより送信され、同一のネットワークセグメントにある他のコンピューターにてプログラムが動作していないかのチェックが行われます。応答が無い場合は、アプリケーションの他インスタンスから送信される信号に応えるため、ポートが開かれます。

マルチユーザーライセンス

同一のLAN 内にて2つ以上のアプリケーションインスタンスが使用された場合、スタートアップ時に、これらインスタンス間において通信が行われます。これらのインスタンス間にてキーコードのやりとりが行われ、購入された数のライセンスを超えてインスタンスが起動しないように保証することができます。このようなライセンス計測システムはUNIX やデータベース開発ツールにて広く使用されているもので、Altova ユーザーはリーズナブルな価格にて同時使用マルチユーザーライセンスを購入することができます。

弊社はアプリケーションのデザインも行っており、少数の小さなネットワークパケットを送信することで、ネットワークに対する負荷を最小限に抑えております。Altova により使用される2799番 TCP/IP ポートはIANA により公式登録されており（詳細は（ IANA W eb サイト (http://www.iana.org/) を参照ください）、弊社のライセンス計測モジュールも既にテストされたものです。

ファイヤーウォールを使用している場合、2799番ポートにてAltova 製品が動作しているコンピューター同士が通信しているのに気づかれるかも知れません。その他の手段によりライセンス使用許諾書の内容が守られることを保証できる限り、組織間の異なるグループにおいてこのようなトラフィックをブロックすることは勿論可能です。

証明書に関するメモ

Altova アプリケーションはHTTPS を介してAltova ライセンスサーバー（`link.altova.com`）に通信します。この通信のためにAltova は登録済みのSSL 証明書を使用します。（例えば、社内IT 部署または外部エージェンシーにより）この証明書が置き換えられている場合、使用中のAltova アプリケーションは接続が安全でないことを警告します。Altova アプリケーションを開始するために代替の証明書を使用することができますが、自己責任で行ってください。*安全ではない接続* の警告メッセージが表示されると、証明書の発行元を確認して（ Altova 証明書の代替証明書の使用の継続または停止を決定することができる）社内IT チームと相談してください。

（例えば、クライアントマシンへの、または、クライアントマシンへの通信を監視するために）自身の証明書の使用が必要な場合 Altova の無料管理ソフトウェアである Altova LicenseServer を使用中のネットワークにインストールすることが奨励されます。このセットアップでは、Altova LicenseServer はAltova との通信のためにAltova 証明書の使用を許可しつつクライアントマシンが所属機関の証明書の使用を継続することができます。

19.3.3　エンドユーザー使用許諾契約書

- Altova エンドユーザー使用許諾契約書: http://www.altova.com/ja/legal/eula
- Altova プライバシーポリシー: http://www.altova.com/ja/privacy

20 用語

このセクションにはMapForce およびMapForce に関連する製品に固有のキーワードのリストが含まれています。

コンポーネント

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

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

接続

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

コネクタ

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

資格情報

(ユーザー名、パスワード、およびOAuth 認証の詳細などの) 認証データを異なるマッピング実行環境で安全なポータブル化する方法を資格情報オブジェクトは提供します。資格情報は基本的な HTTP 認証 、またはOAuth 2.0 認証を必要とするマッピング内で役に立ちます。MapForce とFlowForce Server 内で資格情報を定義することができます。MapForce 内で資格情報が定義されている場合、マッピングデプロイされるようにFlowForce Server にデプロイすることができます。

固定長 ファイル(FLF)

データが習慣的に固定値を持つフィールドに分割される共通のテキストフォーマット。(例: 行の最初の5 文字がトランザクションID を表し、次の20 文字がトランザクションの詳細を表すなど)。

FlexText

FlexText とはMapForce によりサポートされる他のフォーマットに変換するには複雑な非標準またはレガシーテキストファイルからのデータを変換するMapForce Enterprise Edition 内のモジュールです。

グローバルリソース

Altova グローバルリソースはファイル、フォルダー、またはデータベースへのポータブルなレファレンスです。グローバルリソースとして保管されると、パスとデータベース接続の詳細は再利用できるようになりAltova アプリケーション全体で使用可能になります。例えば、複数のAltova デスクトップアプリケーション内で同じファイルを頻繁に開く必要がある場合、グローバルリソースとして定義すると便利な場合があります。このようにすると、「ファイルを開く」ダイアログボックスから対応するグローバルリソースを開くことができるため、ファイルパスを覚えている必要がありません。この方法はファイルパスが変更されると、パスの一部を変更するのみという利点があります。

グローバルリソースの一般的な使用法は、データベース接続を一度定義し、グローバルリソースをサポートする全てのAltova アプリケーションで再利用することです。例えば、MapForce マッピングがデザインされるマシンでデータベース接続を作成し、MapForce Server がマッピングを実行するマシン上で同じ接続を再利用することができます（これは、一部の場合、両方のマシンに同じデータベースクライアントソフトウェアがインストールされている場合があります）。

任意で、（「構成」として既知の）同じグローバルリソースの複数のバリエーションを作成することができます。これによりファイル、またはフォルダーパス（またはデータベース）を必要に応じて変更することができます。例えば、「開発」と「生産」の構成を持つデータベースリソースを作成することができます。MapForce を例に挙げると、マッピングをプレビューする前に、希望する構成をリストから選択して、開発、または生産データベースからデータを簡単に抽出することができます。

次のAltova デスクトップアプリケーションからグローバルリソースを作成することができます: Altova Authentic、DatabaseSpy、MobileTogether Designer、MapForce、StyleVision、およびXMLSpy。サーバー製品では、次のAltova サーバーアプリケーションでグローバルリソースを作成することができます: FlowForce Server、MapForce Server、RaptorXML Server、RaptorXML+XBRL Server。

入力コンポーネント

入力コンポーネントは、単純型の値をマッピングにパスすることができるMapForce コンポーネントです。入力コンポーネントは通常ファイル名またはその他の文字列の値をランタイムにマッピングにパスするために使用されます。入力コンポーネントとソースコンポーネントを混同しないように注意してください。

ジョインコンポーネント

ジョインコンポーネントは、カスタムに定義付けられた条件をベースにして2つの構造をジョインすることを有効化するMapForce コンポーネントです。条件を満たすアイテムのアソシエーション（ジョインされたセット）を返します。ジョインは、（ID などの）共通のフィールドを共有する2つの構造からのデータを結合する際にとても役に立ちます。

MapForce

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

マッピング

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

(C) 2015-2021 Altova GmbH

MFF

MapForce 関数ファイルのファイル名拡張子

MFD

MapForce デザインドキュメント(マッピング)のファイル名拡張子

MFP

MapForce プロジェクトファイルのファイル名拡張子

MFT

MapForce FlextText テンプレートドキュメントのファイル名拡張子

出力コンポーネント

出力コンポーネント（または「単純型出力」）とは、マッピングから文字列の値を返すことを有効化するMapForce コンポーネントです。出力 コンポーネントは、1つのターゲットコンポーネント型のみをあらわしますが、後者と混同しないようご注意してください。

parent-context

`min、max、avg、count` などのMapForce 一部のコア集計関数 **parent-context** 任意の引数です。複数の階層的シーケンスを持つソースコンポーネントでは、親コンテキストは、ノードのセットがどの関数上で操作されるかを決定します。

ソースコンポーネント

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

ターゲットコンポーネント

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

インデックス

.

.NET 拡張関数,
XSLT と XQuery, 1993
インスタンスメソッド、インスタンスフィールド, 1996
コンストラクター, 1995
データ型変換、.NET から XPath/XQuery へ, 1998
データ型変換、XPath/XQuery から .NET へ, 1997
概要, 1993
静的メソッド、静的フィールド, 1996
.NET 内のXSLT と XQuery のための拡張機能,
.NET 拡張関数を参照する, 1993

1

1回分割する,
「固定長」モードの使用, 698
区切られた (フロート) モードの使用, 699
区切られた (行ベース) モードの使用, 700, 701

1

1回分割する,
FlexText 関数として, 697

A

A から Z,
並べ替えコンポーネント, 203
aar ファイル,
のための Axis2axis アーカイブ, 1293
abs,
MapForce 関数として (lang | math 関数), 1183
MapForce 関数として(xpath2 | numeric 関数), 1244
acos,
MapForce 関数として (lang | math 関数), 1183
ActiveX,
アプリケーションレベルの統合, 1833
ドキュメントレベルの統合, 1836
統合の必要条件, 1829
ActiveX コントロール,
Visual Studio ツールボックスへの追加, 1831
add,
MapForce 関数として (core | math 関数), 1102
ADO,
データ接続インターフェイスとして, 380
接続のセットアップ, 386
ADO.NET,
接続のセットアップ, 392
age,
MapForce 関数として (lang | datetime 関数), 1159
Altova XML パーサー,
について, 2001
Altova エンジン,
Altova 製品内で, 2001
Altova 拡張関数,
チャート関数 (チャート関数を参照), 1901
ANSI X12,
メッセージ構造のカスタム化, 815
Ant,
Build Java project with, 1524
building Java code with, 1527
Java Web サービスのコンパイル, 1293
setting the environment variables for,, 1634
Any,
xs:any, 362
API,
documentation, 1654
Application object, 1658
AS2,
Altova 製品との統合, 1403
ASC X12, 728, 743
asin,
MapForce 関数として (lang | math 関数), 1184
atan,
MapForce 関数として (lang | math 関数), 1184
ATTLIST,
DTD 名前空間 URI, 353
auto-format,
MapForce 関数として (edifact 関数), 1155
auto-number,
MapForce 関数として (コア | generator 関数), 1094
avg,
MapForce 関数として (core | aggregate 関数), 1070
Axis2,

(C) 2015-2021 Altova GmbH

Axis2,
SOAP Web サービスのための構成, 1308

B

base-uri,
MapForce 関数として(xpath2 | accessors ライブラリ), 1216
boolean,
MapForce 関数として (core | conversion 関数), 1076
build.xml,
enabling the zip64mode in,, 1634
Built-in エンジン,
使用, 93
定義, 93

C

C#,
code, 1506
code generation settings, 1635
error handling, 1659
generating program code, 1515
integrate generated code, 1539
MapForce の統合, 1839
reference to generated classes, 1604
Web サービスの生成, 1295
C++,
code, 1506
code generation settings, 1635
error handling, 1659
generating program code, 1510
integrate generated code, 1541
reference to generated classes, 1589
capitalize,
MapForce 関数として (lang | string 関数), 1192
CDATA, 360
ceiling,
MapForce 関数として (core | math 関数), 1103
char-from-code,
MapForce 関数として (core | string 関数), 1139
charset-decode,
MapForce 関数として (lang | string 関数), 1193
charset-encode,
MapForce 関数として (lang | string 関数), 1195
Class ID,
MapForce 統合, 1846
Code,
built in types, 1649
integrating MapForce code, 1536
SPL, 1637
Code generation,
sample, 1672
Code Generator, 1506
code-from-char,
MapForce 関数として (core | string 関数), 1139
columnname-to-index,
MapForce 関数として, 1212
COM API,
documentation, 1654
concat,
(関数として) サンプルの使用方法, 659
MapForce 関数として (core | string 関数), 1140
convert-to-utc,
MapForce 関数として (lang | datetime 関数), 1159
cos,
MapForce 関数として (lang | math 関数), 1184
count,
MapForce 関数として (core | aggregate 関数), 1071
count-substring,
MapForce 関数として (lang | string 関数), 1196
create-guid,
MapForce 関数 (lang | generator 関数), 1180
CSV,
マッピングソースとして, 647
階層の作成 - キー, 652
空のフィールドの置換, 981
作成 複数の 行, 650
CSV として保管(区切られた),
FlexText 関数として, 709
CSV ファイル,
ソース コンポーネントとして, 655
ターゲットコンポーネントとして, 655
のエンコードを設定, 655
プレビュー データ, 655
追加または削除フィールド, 655
current-date,
MapForce 関数として(xpath2 | context 関数), 1221
current-dateTime,
MapForce 関数として(xpath2 | context 関数), 1221
current-time,
MapForce 関数として(xpath2 | context 関数), 1221
currrent,

currrent,
MapForce 関数として(xslt | xslt 関数ライブラリ), 1273

D

Database connection,
setup examples, 409
date-from-datetime,
MapForce 関数として (lang | datetime 関数), 1160
datetime-add,
MapForce 関数として (lang | datetime 関数), 1161
datetime-diff,
MapForce 関数として (lang | datetime 関数), 1162
datetime-from-date-and-time,
MapForce 関数として (lang | datetime 関数), 1163
datetime-from-parts,
MapForce 関数として (lang | datetime 関数), 1163
datetime-to-xlsx,
MapForce 関数として, 1214
date-to-xlsx,
MapForce 関数として, 1213
day-from-datetime,
MapForce 関数として (lang | datetime 関数), 1166
day-from-duration,
MapForce 関数として (lang | datetime 関数), 1166
DB,
ORDER BY, 561
default-collation,
MapForce 関数として(xpath2 | context 関数), 1221
degrees,
MapForce 関数として (lang | math 関数), 1185
distinct-values,
MapForce 関数 (core | sequence 関数)として, 1115
divide,
MapForce 関数として (core | math 関数), 1103
divide-integer,
MapForce 関数として (lang | math 関数), 1185
document,
MapForce 関数として(xslt | xslt 関数ライブラリ), 1273
DoTransform.bat,
RaptorXML Server を使用して実行, 1389
DTD,
ソースとターゲット, 353
duration-add,
MapForce 関数として (lang | datetime 関数), 1167
duration-from-parts,
MapForce 関数として (lang | datetime 関数), 1167
duration-subtract,
MapForce 関数として (lang | datetime 関数), 1168

E

Eclipse,
compiling Java code with, 1527
generating mapping code for, 1522
Eclipse のための MapForce プラグイン,
creating a MapForce/Eclipse プロジェクト, 1469
Eclipse プロジェクトにマッピングをインポートする, 1473
MapForce パースペクティブに切り替える, 1463
インストール, 1459
について, 1458, 1463, 1469
マッピングとプロジェクトとの作業, 1469
拡張機能, 1479
拡張点, 1479
関数共通のメニューと関数の評価, 1466
自動コードの生成の設定, 1476
新規マッピングの作成, 1471
EDI,
ASC X12, 743
IATA PADIS, 753
NCPDP SCRIPT, 757
X12 HIPAA, 747
XML から生成する, 740, 742
XML へのクイック変換, 740, 742
カスタマイズ, 793
セパレーター, 730
検証のカスタマイズ, 821
EDI ファイル,
マッピングコンポーネントとして, 731
EDIFACT, 728
XML にマッピング, 785
マージされたエントリ, 802
用語, 730
element-available,
MapForce 関数として(xslt | xslt 関数ライブラリ), 1274
empty,
MapForce 関数として (lang | string 関数), 1197
Environment variables,
ANT_OPS, 1634
equal,
MapForce 関数として (core | logical 関数), 1097
equal-or-greater,

(C) 2015-2021 Altova GmbH

equal-or-greater,
MapForce 関数として (core | logical 関数), 1097
equal-or-less,
MapForce 関数として (core | logical 関数), 1098
Error handling,
general description, 1659
Excel 2007+,
RSS フィードからのマッピング, 1349
XML にマッピングする, 866
XML ファイルから生成する, 868
セルの範囲からデータのマッピングを読み込む, 856
セルの範囲にデータのマッピングを書き込む, 856
へマッピング データ, 247
マッピング from データベース to, 872
マッピング コンポーネントからセルの範囲を選択する, 857
マッピング コンポーネントとして, 851
マッピング コンポーネントとして追加, 853
マッピング ソース またはターゲットとして, 849
既存の Excel ワークシートの更新, 874
Excel 2007+ コンポーネント 設定,
変更, 863
Excel 2007+ ワークシート,
からのデータの読み込み, 854
へのデータの書き込み, 854
exists,
MapForce 関数 (core | sequence 関数)として, 1117
exp,
MapForce 関数として (lang | math 関数), 1186

F

false,
MapForce 関数として(xpath2 | boolean 関数), 1219
find-substring,
MapForce 関数として (lang | string 関数), 1197
Firebird,
ODBC を使用しての接続, 411
first-items,
MapForce 関数 (core | sequence 関数)として, 1118
FlexText,
インターフェイス, 672
コンテナー, 672
コンポーネント設置の変更, 687
ターゲットコンポーネントとして, 689
チュートリアル, 675
概要の, 672
FlexText 関数,
1回分割する, 697
CSV として保管(区切られた), 709
FLF として保管 (固定長), 717
ノード, 707
繰り返される分割, 690
切り替え, 702
値の保管, 719
無視する, 708
FLF として保管 (固定長),
FlexText 関数として, 717
floor,
MapForce 関数として (core | math 関数), 1104
FlowForce Server,
OAuth 2.0 認証, 285
グローバルリソース, 1422
グローバルリソースのデプロイ, 1399, 1423
マッピングのデプロイ, 1399
資格情報のデプロイ, 286
資格情報, 285
format-date,
MapForce 関数として (core | conversion 関数), 1076
format-dateTime,
MapForce 関数として (core | conversion 関数), 1077
format-guid-string,
MapForce 関数として (lang | string 関数), 1198
format-number,
MapForce 関数として (core | conversion 関数), 1080
format-time,
MapForce 関数として (core | conversion 関数), 1083
function-available,
MapForce 関数として(xslt | xslt 関数ライブラリ), 1274
Functions,
finding in the Libraries window, 959
finding occurences in active mapping, 959

G

Generate,
code from schema, 1506
generate-id,
MapForce 関数として(xslt | xslt 関数ライブラリ), 1274
generate-sequence,
MapForce 関数 (core | sequence 関数)として, 1119
get-fileext,
MapForce 関数として (core | file path 関数), 1090

get-folder,
MapForce 関数として (core | file path 関数), 1091
Global objects,
in SPL, 1641
greater,
MapForce 関数として (core | logical 関数), 1098
group-adjacent,
MapForce 関数 (core | sequence 関数)として, 1120
group-by,
MapForce 関数 (core | sequence 関数)として, 1122
group-ending-with,
MapForce 関数 (core | sequence 関数)として, 1124
group-into-blocks,
MapForce 関数 (core | sequence 関数)として, 1125
group-starting-with,
MapForce 関数 (core | sequence 関数)として, 1127

H

HIPAA,
X12, 747
X12 999 Implementation Acknowledgement の生成, 746
HIPAA X12, 728
HL7, 728
HL7 2.6 から 3.x,
サンプル, 932
hour-from-datetime,
MapForce 関数として (lang | datetime 関数), 1169
hour-from-duration,
MapForce 関数として (lang | datetime 関数), 1170
HRESULT,
and error handling, 1659
HTML,
MapForce の統合, 1848
マッピングドキュメンテーションの生成, 296
マッピングの出力のプレビュー, 292
HTML サンプル,
MapForceControl 統合, 1846, 1847, 1848
of MapForceControl 統合, 1846
HTTPS,
Web サービスの呼び出し, 1370

I

IATA PADIS, 728, 753
IBM DB2,
JDBC を介した接続, 413
ODBC を使用しての接続, 415
XML 型フィールドから読み込む, 575
XML 型フィールドへ書き込む, 575
IBM DB2 for i,
ODBC を使用しての接続, 423
IBM Informix,
connecting through JDBC, 422
JDBC を使用しての接続, 426
If-Else 条件,
マッピングに追加, 214
IIS,
SOAP Web サービスのための構成, 1295
implicit-timezone,
MapForce 関数として(xpath2 | context 関数), 1222
index-to-columnname,
MapForce 関数として, 1214
INNER JOIN,
ジョインコンポーネント, 546
Integrate,
into C#, 1539
into C++, 1541
into Java, 1537
Integrate MapForce code, 1536
is-not-null,
MapForce 関数として (db 関数), 1153
is-null,
MapForce 関数として (db 関数), 1153
ISO 20022,
サポートノート, 769
へのデータのマッピング、からのデータのマッピング, 769
is-xsi-nil,
MapForce 関数として (core | node 関数), 1108
item-at,
MapForce 関数 (core | sequence 関数)として, 1128
items-from-till,
MapForce 関数 (core | sequence 関数)として, 1130

(C) 2015-2021 Altova GmbH

J

Java, 1852
  avoiding exceptions in generated code, 1634
  code, 1506
  generating program code, 1522, 1527
  integrate generated code, 1537
  reference to generated classes, 1619
  Web サービスの作成, 1293
  Web サービスの生成, 1308
Java 拡張関数,
  XSLT と XQuery, 1985
  インスタンスメソッド、インスタンスフィールド, 1991
  コンストラクター, 1990
  データ型変換、Java から XPath/XQuery へ, 1993
  データ型変換、XPath/XQuery から Java へ, 1992
  ユーザー定義の JAR ファイル, 1989
  ユーザー定義のクラスファイル, 1986
  概要, 1985
  静的メソッド、静的フィールド, 1990
JavaScript,
  error handling, 1659
JDBC,
  handling references in generated code, 1526
  Teradata への接続, 458
  データ接続インターフェイスとして, 380
  接続の設定 (Windows), 402
JScript,
  code-generation sample, 1672
JSON,
  Web サービスからのマッピング, 1344
  マッピングサンプル, 840
  追加プロパティからのマッピング, 840
JSON ファイル,
  ソーまたはターゲット コンポーネントとして, 831

L

last,
  MapForce 関数として(xpath2 | context 関数), 1222
last-items,
  MapForce 関数 (core | sequence 関数)として, 1131
leapyear,
  MapForce 関数として (lang | datetime 関数), 1170
left,
  MapForce 関数として (lang | string 関数), 1199
LEFT OUTER JOIN,
  ジョインコンポーネント, 546
left-trim,
  MapForce 関数として (lang | string 関数), 1199
less,
  MapForce 関数として (core | logical 関数), 1099
Libraries window,
  finding functions in, 959
Library, 1650
Linux,
  HTTPS を介して Web
サービスを使用してマッピングを実行する, 1384
  クライアント証明書の転送, 1384
  信頼できるサーバー証明書, 1372
local-name-from-QName,
  MapForce 関数として (lang | QName 関数), 1112
log,
  MapForce 関数として (lang | math 関数), 1186
log10,
  MapForce 関数として (lang | math 関数), 1187
logical-and,
  MapForce 関数として (core | logical 関数), 1099
logical-not,
  MapForce 関数として (core | logical 関数), 1100
logical-or,
  MapForce 関数として (core | logical 関数), 1100
logical-xor,
  MapForce 関数として (lang | logical 関数), 1181
Look-up テーブル,
  マッピング上での使用, 232
lowercase,
  MapForce 関数として (lang | string 関数), 1200

M

Mac,
  HTTPS を介して Web
サービスを使用してマッピングを実行する, 1385
  クライアント証明書の転送, 1385
  信頼できるサーバー証明書, 1373
main-mfd-filepath,
  MapForce 関数として (core | file path 関数), 1091
MapForce,
  API, 1654

(C) 2015-2021 Altova GmbH

MapForce,
概要, 36
基本概念, 41
統合, 1828
MapForce API, 1654
MapForce Server,
OAuth 2.0 認証, 283
から例外をスロー, 241
グローバルリソース, 1421
のためにマッピングをコンパイルする, 1396
マッピングの自動化, 1390
資格情報, 283
MapForce サンプル,
ディスク上の場所, 52
MapForce 統合,
のサンプル, 1846, 1847, 1848
MapForceControl, 1876
C# を使用した統合, 1839
HTML を使用下統合, 1848
Visual Basic を使用した統合, 1861
アプリケーションレベルでのサンプル 統合, 1846, 1847, 1848
オブジェクトレファレンス, 1873
ドキュメント, 1828
ドキュメントレベルでの統合のサンプルサンプル, 1839
MapForceControl ドキュメント, 1883
MapForceControl プレースホルダ, 1890
MapForceコマンド,
MapForceControl, 1873
MapForceControl 内, 1875
MariaDB,
ODBC を使用した接続, 428
match-pattern,
MapForce 関数として (lang | string 関数), 1200
max,
MapForce 関数として (core | aggregate 関数), 1072
MapForce 関数として (lang | math 関数), 1187
maxLength,
ノード関数内での使用, 978
max-string,
MapForce 関数として (core | aggregate 関数), 1072
MERGE,
MapForce-により生成された SQL ステートメントとして, 514
MapForce-により生成された SQL 内のステートメントとして, 507
Method names in generating code,
reserving, 1634
mfd-filepath,
MapForce 関数として (core | file path 関数), 1092
mff,
mff.xsd ファイル, 1029
ライブラリファイル, 1029
mff ファイル,
構成, 1029
Microsoft Access,
ADO を介しての接続, 386, 430
Microsoft Excel,
"Excel 2007+"を参照, 849
Microsoft SharePoint Server,
ファイルをコンポーネントとして追加, 89
Microsoft SQL Server,
ADO を介しての接続, 433
ODBC を使用しての接続, 435
millisecond-from-datetime,
MapForce 関数として (lang | datetime 関数), 1171
millisecond-from-duration,
MapForce 関数として (lang | datetime 関数), 1171
min,
MapForce 関数として (core | aggregate 関数), 1073
MapForce 関数として (lang | math 関数), 1188
minLength,
ノード関数内での使用, 978
min-string,
MapForce 関数として (core | aggregate 関数), 1073
minute-from-datetime,
MapForce 関数として (lang | datetime 関数), 1172
minute-from-duration,
MapForce 関数として (lang | datetime 関数), 1172
modulus,
MapForce 関数として (core | math 関数), 1104
month-from-datetime,
MapForce 関数として (lang | datetime 関数), 1173
month-from-duration,
MapForce 関数として (lang | datetime 関数), 1173
MSXML,
generating code for, 1635
MSXSL スクリプト内の拡張関数, 1998
msxsl:script, 1998
multiply,
MapForce 関数として (core | math 関数), 1105
MySQL,
ODBC を使用しての接続, 441

(C) 2015-2021 Altova GmbH

N

namespace-uri-form-QName,
MapForce 関数として (lang | QName 関数), 1114
NCPDP SCRIPT,
マッピングコンポーネントとして, 757
negative,
MapForce 関数として (lang | logical 関数), 1181
nillable,
XML スキーマ内の属性として, 357
node-name,
MapForce 関数として (core | node 関数), 1110
MapForce 関数として(xpath2 | accessors ライブラリ), 1217
node-name 関数, 257
代替の使用方法, 257
normalize-space,
MapForce 関数として (core | string 関数), 1142
not-equal,
MapForce 関数として (core | logical 関数), 1101
not-exists,
MapForce 関数 (core | sequence 関数)として, 1132
now,
MapForce 関数として (lang | datetime 関数), 1174
NULL,
データベースコンポーネントの処理, 490
データベースマッピング内での扱い, 535
NULL 値,
複数の発生を置き換える, 962
number,
MapForce 関数として (core | conversion 関数), 1084
numeric,
MapForce 関数として (lang | logical 関数), 1182

O

OAuth 2.0,
FlowForce Server 内の, 285
MapForce Server 内の, 283
アクセストークンの取得, 282
資格情報, 278, 282
Object model,
overview, 1658
ODBC,
MariaDB への接続, 428
Teradata への接続, 460
データ接続インターフェイスとして, 380
接続のセットアップ, 399
ODBC ドライバー,
使用可能か確認する, 399
OLE DB,
データ接続インターフェイスとして, 380
OOXML,
デフォルト Excel 2007 フォーマットとして, 849
Oracle,
XML 型フィールドから読み込む, 575
XML 型フィールドへ書き込む, 575
Oracle データベース,
JDBC を使用しての接続, 443
ODBC を使用しての接続, 444
ORDER BY,
SQL where コンポーネント, 561
OS,
Altova 製品のための, 2001
Out of memory exceptions,
resolving, 1634

P

pad-string-left,
MapForce 関数として (lang | string 関数), 1201
pad-string-right,
MapForce 関数として (lang | string 関数), 1202
parse-date,
MapForce 関数として (core | conversion 関数), 1084
parse-dateTime,
MapForce 関数として (core | conversion 関数), 1086
parse-number,
MapForce 関数として (core | conversion 関数), 1088
parse-time,
MapForce 関数として (core | conversion 関数), 1089
PDF,
マッピングドキュメンテーションの生成, 296
マッピングの出力のプレビュー, 292
pi,
MapForce 関数として (lang | math 関数), 1188
position,
MapForce 関数 (core | sequence 関数)として, 1132
positive,
MapForce 関数として (lang | logical 関数), 1182

(C) 2015-2021 Altova GmbH

PostgreSQL,
ODBC を使用しての接続, 450
直接 (ネイティブ)接続, 404
直接接続する(ネイティブ), 406
pow,
MapForce 関数として (lang | math 関数), 1188
Progress OpenEdge データベース,
JDBC を使用しての接続, 453
ODBC を使用しての接続, 454

Q

QName,
MapForce 関数として (lang | QName 関数), 1112
QName サポート, 356
QName-as-string,
MapForce 関数として (lang | QName 関数), 1191

R

radians,
MapForce 関数として (lang | math 関数), 1189
random,
MapForce 関数として (lang | math 関数), 1189
RaptorXML Server,
変換の実行, 1389
read-binary-file,
MapForce 関数として (lang | file 関数), 1179
remove-fileext,
MapForce 関数として (core | file path 関数), 1092
remove-folder,
MapForce 関数として (core | file path 関数), 1092
remove-timezone,
MapForce 関数として (lang | datetime 関数), 1174
repeat-string,
MapForce 関数として (lang | string 関数), 1202
replace,
MapForce 関数として (lang | string 関数), 1203
replace-fileext,
MapForce 関数として (core | file path 関数), 1093
replicate-item,
MapForce 関数 (core | sequence 関数)として, 1134
replicate-sequence,
MapForce 関数 (core | sequence 関数)として, 1136
resolve-filepath,
MapForce 関数として (core | file path 関数), 1094
resolve-uri,
MapForce 関数として(in xpath2 | anyURI 関数), 1218
reversefind-substring,
MapForce 関数として (lang | string 関数), 1203
right,
MapForce 関数として (lang | string 関数), 1204
right-trim,
MapForce 関数として (lang | string 関数), 1205
round,
MapForce 関数として (core | math 関数), 1105
round-half-to-even,
MapForce 関数として(xpath2 | numeric 関数), 1244
round-precision,
MapForce 関数として (core | math 関数), 1106
RSS フィード,
からのマッピング, 1349
RTF,
マッピングドキュメンテーションの生成, 296
マッピングの出力のプレビュー, 292

S

SAP IDoc,
XML へのマッピング, 759
データのマッピング, 759
マッピング コンポーネントとして追加, 759
SAP IDocs, 728
Schema,
code generator, 1506
schemanativetype, 1638
Search,
functions in the Libraries window, 959
second-from-datetime,
MapForce 関数として (lang | datetime 関数), 1175
second-from-duration,
MapForce 関数として (lang | datetime 関数), 1175
set-empty,
MapForce 関数 (core | sequence 関数)として, 1137
set-null,
MapForce 関数として (db 関数), 1153
set-xsi-nil,
MapForce 関数として (core | node 関数), 1110
sin,
MapForce 関数として (lang | math 関数), 1190

(C) 2015-2021 Altova GmbH

skip-first-items,
MapForce 関数 (core | sequence 関数)として, 1137
SOAP,
Web サービスを MapForce を作成する, 1282
サポートメモ, 1280
フォールトの定義, 1291
リクエストの作成, 1282
SOAP メッセージ,
TTL (有効時間)の設定, 1341
ユーザー名トークンの追加, 1341
SPL, 1637
code blocks, 1637
conditions, 1644
foreach, 1645
global objects, 1641
subroutines, 1646
using files, 1642
variables, 1640
SQL, 589
SQL スクリプトのインポート, 590
SQL スクリプトをとして文字列ステートメントをエクスポートする, 590
スクリプトからロード, 588
ステートメントの作成, 589
ステートメントの実行, 588, 590
ステートメントの生成, 588, 589
データのジョイン, 546
SQL Server,
ADO.NET を使用して接続する, 392
XML 型フィールドから読み込む, 575, 576
XML 型フィールドへ書き込む, 575, 576
SQL WHERE,
ORDER BY, 561
コンポーネント - 挿入, 561
SQL WHERE/ORDER,
MapForce コンポーネントとして, 564
SQL エディター,
コメントの挿入, 590
テキストのコメントアウト, 590
ブックマークの挿入, 591
ブックマーク余白, 591
作成 領域, 592
削除 コメント, 590
削除 ブックマーク, 591
削除 領域, 592
使用 ブックマーク, 591
使用 領域, 592
設定 - 一般, 600
領域の挿入, 592
SQL サーバー,
ADO を介しての接続, 386
SQLite,
XML ファイルを書き込む, 579
データのマッピング, 541
へのデータのマッピング, 252
以下からデータをマッピングする, 247
生成されたコード内でデータベースパスを絶対パスにする, 150
相対と絶対パスの使用, 147
sqrt,
MapForce 関数として (lang | math 関数), 1190
starts-with,
MapForce 関数として (core | string 関数), 1142
static-node-annotation,
MapForce 関数として (core | node 関数), 1110
static-node-name,
MapForce 関数として (core | node 関数), 1111
string,
MapForce 関数として (core | conversion 関数), 1090
MapForce 関数として(xpath2 | accessors ライブラリ), 1217
string-as-QName,
MapForce 関数として (lang | QName 関数), 1192
string-compare-ignore-case,
MapForce 関数として (lang | string 関数), 1205, 1206
string-join,
MapForce 関数として (core | aggregate 関数), 1074
string-length,
MapForce 関数として (core | string 関数), 1143
Stylevision,
スタイルシートの作成, 302
マッピングドキュメンテーションの生成, 297
substitute-missing,
MapForce 関数 (core | sequence 関数)として, 1138
substitute-missing-with-xsi-nil,
MapForce 関数として (core | node 関数), 1111
substitute-null,
MapForce 関数として (db 関数), 1154
substring,
MapForce 関数として (core | string 関数), 1143
substring-after,
MapForce 関数として (core | string 関数), 1144
substring-before,
MapForce 関数として (core | string 関数), 1144
subtract,
MapForce 関数として (core | math 関数), 1106
sum,

sum,
MapForce 関数として (core | aggregate 関数), 1075
SWIFT MT, 728
サポートノート, 762
へのデータのマッピング、からのデータのマッピング, 762
マッピングサンプル, 765
SWIFT MX,
サポートノート, 769
へのデータのマッピング、からのデータのマッピング, 769
マッピングサンプル, 765
Sybase,
JDBC を使用しての接続, 457
system-property,
MapForce 関数として(xslt | xslt 関数ライブラリ), 1275

T

tan,
MapForce 関数として (lang | math 関数), 1190
Teradata,
JDBC を使用した接続, 458, 460
time-from-datetime,
MapForce 関数として (lang | datetime 関数), 1176
time-to-xlsx,
MapForce 関数として, 1214
timezone,
MapForce 関数として (lang | datetime 関数), 1176
to-date,
MapForce 関数として (edifact 関数), 1156
to-datetime,
MapForce 関数として (edifact 関数), 1156
to-duration,
MapForce 関数として (edifact 関数), 1158
tokenize,
MapForce 関数として (core | string 関数), 1145
tokenize-by-length,
MapForce 関数として (core | string 関数), 1147
tokenize-regexp,
MapForce 関数として (core | string 関数), 1150
Tomcat,
SOAP Web サービスのための構成, 1308
to-time,
MapForce 関数として (edifact 関数), 1158
TRADACOMS, 728
XML に変換する, 779
マッピング コンポーネント, 769
マッピング コンポーネントとして, 774
検証のルール, 775
構成 ファイル, 778
translate (core | string 関数),
MapForce 関数として, 1151
true,
MapForce 関数として(xpath2 | boolean 関数), 1219
Types,
built in, 1649
派生した型 - xsi:type, 353

U

UDF,
マッピングコンテキスト, 309
UN/EDIFACT, 728
XML にマッピング, 785
メッセージ構造のカスタム化, 809
用語, 730
unary-minus,
MapForce 関数として (lang | math 関数), 1191
Unicode,
コードポイント 照合, 203
特別文字の置き換え, 489
Unicode のサポート,
Altova 製品内で, 2001
unparsed-entity-uri,
MapForce 関数として(xslt | xslt 関数ライブラリ), 1275
uppercase,
MapForce 関数として (lang | string 関数), 1206
URI,
DTD 内, 353
と QNames, 356
URL,
ファイルをコンポーネントとして追加, 89

V

Value-Map,
サンプル, 235, 237
マッピングコンポーネントとして, 232
Variables,
in SPL, 1640
Visual Basic,

Visual Basic,
error handling, 1659
MapForce の統合, 1861
Visual Studio,
generating code for, 1635
generating mapping code for, 1510, 1515
MapForce ActiveX コントロールをツールボックスへ追加, 1831
Visual Studio のための MapForce プラグイン,
について, 1452
マッピングとプロジェクトとの作業, 1455
共通のメニューと関数にアクセスする, 1456
有効化, 1453
Visual Studio プラグイン,
として MapForce を実行する, 1452

W

WADL, 1335
Web サービス,
HTTPS デジタル証明書を使用して署名, 1339
HTTPS を使用した呼び出し, 1370
Java 内で作成, 1293
JSON へのマッピング, 1344
MapForce からの呼び出し, 1315
MapForce コンポーネントとして, 1315
REST-スタイル呼び出しの追加, 1344
RSS フィードからのマッピング, 1349
WS-Security の有効化, 1341
サンプル (WSDL-スタイル), 1354
タイムアウトの期間の構成, 1335
デプロイ, 1293
プリエンプティブ 資格の有効化, 1339
マッピングコンテキスト, 307
ユーザー名トークンの追加, 1341
基本 HTTP 認証資格を提供, 1339
例 (REST-スタイル), 1344
Web サービス (REST),
マッピングに追加する, 1317
Web サービス (SOAP),
MapForce を使用して作成, 1277
MapForceを使用して生成, 1295, 1308
実装, 1277
必要条件, 1277
Web サービス (WSDL),
マッピングに追加, 1334
WebDAV Server,
ファイルをコンポーネントとして追加, 89
weekday,
MapForce 関数として (lang | datetime 関数), 1177
weeknumber,
MapForce 関数として (lang | datetime 関数), 1177
WHERE,
SQL WHERE コンポーネント, 561
Windows,
Altova 製品のためのサポート, 2001
HTTPS を介して Web サービスを使用してマッピングを実行する, 1386
信頼できるサーバー証明書, 1374
Word,
マッピングドキュメンテーションの生成, 296
Word 2007+,
マッピングの出力のプレビュー, 292
Wrapper classes,
in generated code, 1635
write-binary-file,
MapForce 関数として (lang | file 関数), 1179
WSDL,
SOAP Web サービスの作成, 1282
Web サービスをマッピングから呼び出す, 1354
サポートメモ, 1280
検証, 1282

X

X12,
HIPAA, 747
X12 997 Functional Acknowledgement の生成, 745
X12 999 Implementation Acknowledgement の生成, 746
マージされたエントリ, 802
XBRL タクソノミパッケージ,
追加, 901
xbrl-measure-currency,
MapForce 関数として, 1211
xbrl-measure-pure,
MapForce 関数として, 1212
xbrl-measure-shares,
MapForce 関数として, 1212
Xerces,
generating code for, 1635
xlsx-to-date,
MapForce 関数として, 1215
xlsx-to-datetime,
MapForce 関数として, 1215

(C) 2015-2021 Altova GmbH

xlsx-to-time,
MapForce 関数として, 1216
XML,
CSV からのデータマッピング, 647
Excel 2007+ からのマッピング, 866
TRADACOMS から変換する, 779
UN/EDIFACTXML からマッピング, 785
データベース フィールドへ書き込む, 579
マッピングターゲットとして, 647
署名 設定, 374
XML カタログ,
しくみ, 1448
構成, 1448
XML から XML へ, 348
XML データ,
データベース フィールドから読み込む, 575
データベース フィールドへ書き込む, 575
XML パーサー,
について, 2001
XML ファイル,
Excel から生成, 868
データベースレコードから生成する, 177
単一の XML ソースから生成する, 175
XML 出力,
インスタンス ファイル名の変更, 349
エンコード設定の変更, 349
スキーマの変更, 349
作成 デジタル署名, 349
XML 宣言,
出力からの抑制, 349
XMLSpy,
SOAP クライアントとして, 1282
XMLSpy コマンドテーブル, 1864
XQuery,
カスタム関数の追加, 1017, 1018
モジュールのインポート, 1018
拡張関数, 1984
生成されたコードのプレビュー, 106
XQuery プロセッサー,
Altova 製品内で, 2001
xs: any (xs:anyAttribute), 362
xsi:nil,
XML インスタンス内の属性として, 357
xsi:type,
マッピングから派生した型, 353
XSLT,
カスタム関数の削除, 1011
カスタム関数の追加, 1011
テンプレート名前空間, 1011
拡張関数, 1984
生成されたコードのプレビュー, 105
XSLT と XQuery のためのJava 拡張関数,
Java 拡張関数を参照する, 1985
XSLT と XQuery のための拡張関数, 1984
XSLT プロセッサー,
Altova 製品内で, 2001
XSLT/XQuery 内のスクリプト,
拡張関数で参照する, 1984

Y

year-from-datetime,
MapForce 関数として (lang | datetime 関数), 1178
year-from-duration,
MapForce 関数として (lang | datetime 関数), 1178

Z

Z to A,
並べ替えコンポーネント, 203
zip64mode,
enabling in the build.xml file, 1634
アイコン,
データベース クエリのメッセージウィンドウ, 597
データベース クエリの結果ウィンドウ, 597
アイテム,
不足している, 133
インスタンス,
ファイル参照をに変更する, 145
インターネットの使用,
Altova 製品内で, 2002
インライン XBRL,
からデータをマッピングする, 880
エンコード,
CSV ファイル内の変更, 655
フラットテキストファイルの変更, 666
エンコード設定,
XML 出力, 349
エンドユーザー使用許諾契約書, 2003, 2004
エンベロープ,
デジタル署名, 374
オプション,

オプション,
テキスト フォント - データベース クエリ, 602
結果表示 - データベース クエリ, 601
カスタムライブラリ,
追加, 1029
キー, 652
テキストファイルからのマッピング, 652
テキストファイル内のフィールドテキスト, 652
並べ替えキー, 203
キーと値のペア,
マッピング上での使用, 232
グローバルリソース,
FlowForce Server にデプロイする, 1399, 1423
FlowForce Server 内, 1421
FlowForce Server 内の, 1422
はじめに, 1413
作成, 1414
使用例, 1425, 1427, 1429, 1433
コードポイント,
照合, 203
コード生成設定,
プロジェクト全体のためにグローバルに定義する, 140
プロジェクト内のフォルダーのために定義する, 141
コネクタ,
処理された値の履歴の確認, 335
全てコピー, 160
コメント,
ターゲット ファイルに追加, 359
コンテキストウィンドウ,
について, 329, 337
コンポーネント,
アプリケーションメニューとして, 1490
データの並べ替え, 203
マッピングに追加, 88
概要, 111
検索, 112
削除されたアイテム, 133
整列, 113
設定の変更, 114
シーケンス, 305
スキーマ,
XML ファイルのために生成する, 348
と XML マッピング, 348
ファイル参照をに変更する, 145
スケール,
ノード関数内での使用, 978
ストアドプロシージャ,
サポートメモ, 603
マッピングに追加, 605
マッピングの呼び出し, 603, 608, 611, 615, 619, 624
セクション,
CDATA, 360
セパレーター,
EDI, 730
ソース優先,
- 混在コンテンツマッピング, 152
ソース優先接続,
通常 (ターゲット優先) 接続に対して, 159
ソフトウェア製品ライセンス, 2004
ターゲット,
FlexText コンポーネント, 689
ターゲットコンポーネント,
処理順序の変更, 321
ターゲット優先 マッピング, 152
ターゲット優先接続,
ソース優先接続に対して, 159
チャート関数,
サンプル, 1982
チャートデータ構造, 1977
リスト, 1973
ツール,
アプリケーションメニューとして, 1499
データオーバーレイ,
について, 329
データストリーミング,
定義, 92
データのオーダー,
並べ替えコンポーネント, 203
データの統合,
XML ファイルのマージ, 367
データベース,
XML スキーマをフィールドに割り当てる, 576
XML ファイルを書き込む, 579
クエリ, 586
グローバルリソースとして, 1415
シーケンスになった一意の値を生成, 490
データを複数にリンクされているテーブルに挿入する, 500
テーブルデータの更新, 504
テーブルデータの削除, 521
テーブルにデータを挿入する, 496
テーブルの更新と挿入, 507
にデータをマージする, 507
に対してマッピングを実行する, 487
バルク挿入の使用, 532
マッピングコンテキスト, 308
マッピングに追加, 465

データベース,
複数の XML ファイルから生成する, 177
データベース クエリ,
SQL の生成, 592, 597
コメントアウトテキスト, 590
テキスト フォント オプション, 602
フィルタリング テーブル, 594
ブックマーク, 591
結果表示 オプション, 601
領域, 592
データベース リレーションシップ,
マッピング内で定義, 480
マッピング内に保持, 473
データベースオブジェクト,
フィルタリング, 467
マッピングから削除, 467
マッピングへ追加, 467
データベースドライバー,
概要, 383
データベーストランザクション,
ストアドプロシージャのために有効化, 615
マッピング内で使用, 528, 628
ロールバック, 631
データベースへのアクセス,
IF 条件をベースにした更新, 659
データベース接続,
ウィザードの開始, 381
グローバルリソースからの再使用, 408
設定, 380
データベース例外,
ログ, 642
処理, 631
テーブルデータ,
並べ替え, 203
テーブル名からスキーマ名を削除する,
コード生成オプションとして, 470
テキスト,
CSV
ならびに固定長テキストファイルから階層構造を作成, 652
マッピング テキストファイル, 647
テキストファイル,
エンコードの設定, 666
からのデータのプレビュー, 666
ソース コンポーネントとして, 666
ターゲットコンポーネントとして, 666
データのマッピング, 659
フィールドの追加または削除, 666
固定されたフィールドサイズの設定, 659
充填文字の設定, 659
デジタル署名,
XML 出力の作成, 349
アクティブ化, 371
設定, 374
デタッチ済み,
デジタル署名, 374
デバッガーポジション,
の現在の値野確認, 335
デバッグ,
ステップバイステップ, 324
について, 329
のための準備, 327
ブレークポイントと, 324
開始, 328
制限, 324
設定, 346
停止, 328
デフォルトのスキーマに対して相対的にオブジェクト名を使用する,
オプションとして, 470
デフォルトの値,
複数のアイテムに適用, 962
ドキュメントレベル,
XMLSpy の統合のサンプル, 1839
ナビゲート,
ブックマーク, 591
ノード,
FlexText 関数として, 707
ノード関数,
削除, 966
編集, 966
ノード名,
データをマッピング, 257
パーサー,
Altova 製品に内蔵の, 2001
バイナリファイル,
へのデータの書き込み, 953
マッピングコンポーネントとして, 934
マッピングに追加する, 950
以下からのデータの読み込み, 951
概要, 950
パスワード,
デジタル署名, 374
バッチ,
自動化された 処理, 1390
パラメーター,
マッピングに提供する, 179, 184

パラメーター,
ユーザー定義関数内, 991
バリデーター,
Altova 製品内で, 2001
ファイル,
アプリケーションメニューとして, 1481
コンポーネントとしてのボタン, 114
コンポーネントのボタンとして, 170
ファイル - キーフィールドの定義,
フィールドテキストファイル内のキー, 652
ファイル DSN,
セットアップ, 399
ファイル/文字列,
コンポーネントとしてのボタン, 114
コンポーネントのボタンとして, 170
ファイル: (デフォルト),
ルートノードの名前として, 170
ファイル: <dynamic>,
ルートノードの名前として, 170
ファイルパス,
ファイルベースのデータベースの, 147
生成されたコード, 150
相対と絶対, 145, 150
破損した参照の修正, 149
ファイル名,
マッピング入力パラメーターとして提供する, 174
フィルター,
XML ファイルのマージ, 367
オンラインブラウザー, 594
データベース オブジェクト, 594
データベース内, 564
マッピングに追加, 214
全てコピー コネクタ, 160
フィルタリング,
コンポーネントからのデータ, 214
データベーステーブル, 214
マッピング上のデータベースオブジェクト, 467
フォルダー,
レイアウト - データベース クエリ, 592
ブックマーク,
ナビゲート, 591
ブックマーク余白, 591
削除, 591
挿入, 591
ブラウザー,
データベース クエリ, 592
フィルターの適用, 594
フィルタリング, 594
ブラウザービュー,
SQL ステートメントの生成, 595
コンテキストメニュー オプション, 595
フラット' ファイル,
マッピング, 647
プラットフォーム,
Altova 製品のための, 2001
ブレークポイント,
について, 329
削除, 332
追加, 332
ブレークポイントウィンドウ,
について, 329, 339
プロジェクト,
アプリケーションメニューとして, 1488
開く, 138
検索, 138
作成, 138
閉じる, 138
ブロックコメント, 590
プロトコルバッファー,
コンポーネント設定, 938
データの書き込み, 944
マッピングコンポーネントとして, 934
マッピングへの追加, 935
以下からのデータの読み込み, 939
ヘルスヘベル 7,
サンプル, 932
ヘルプ,
アプリケーションメニューとして, 1501
マークされたアイテム,
不足しているアイテム, 133
マージ, 367
XML ファイル, 367
マージされたエントリ, 802
マッピング,
ソース優先 - 混在コンテンツ, 152
デバッグ, 324
フラットファイルフォーマット, 647
型優先, 160
検証, 94
作成, 88
自動化された 処理, 1390
マッピング 出力,
として複数のファイルを生成する, 170, 174
マッピングコンテキスト, 306
マッピングドキュメンテーション,
生成, 296

マッピングメソッド,
ターゲット優先, 152
標準, 152
標準 / 混在 / 全てコピー, 152
マッピング入力,
としてカスタムファイル名を提供する, 174
として複数のファイルを提供する, 170, 172, 174
メッセージ,
ウィンドウ - データベース クエリ, 597
データベース クエリ内のアイコン, 597
メモリ不足エラー,
トラブルシューティング, 92
メモリ要件, 2001
ユーザー DSN,
セットアップ, 399
ユーザー定義関数,
インラインと正規, 996
コピーと張り付け, 1002
サンプル, 986, 1002, 1006
ナビゲート, 998
パラメーターの追加, 991
マッピングにインポート, 1000
概要, 986
呼び出し, 1000
再帰的な呼び出し, 1006
作成, 988
削除, 999
資格情報を持つ, 281
編集, 998
ライセンス, 2004
情報, 2003
ライセンス計測,
Altova 製品にて, 2003
ライブラリ,
カスタムの追加, 1029
自動ロード, 1029
ライブラリファイル,
mff, 1029
ラインコメント, 590
レイアウト,
ブラウザー, 592
レファレンス, 1480
ローカルロケーション,
とストアドプロシージャ, 618, 619, 624
ローカル照合, 203
ログ,
データベース例外, 642
ワークシート,
XML 出力ファイル, 868
ワイルドカード,
xs:any - xs:anyAtrribute, 362
引用文字,
CSV ファイル内, 655
階層構造,
テキストファイルから, 652
外部キー,
データベースマッピング内, 473
データベースマッピング内の, 480
関数,
アプリケーションメニューとして, 1493
パラメーターの削除, 961
パラメーターの追加, 961
マッピングコンポーネントとして追加, 956
引数データ型の確認, 960
関数の説明の確認, 960
複数のアイテムに条件付きで適用する, 973
複数のアイテムに適用, 962
含む,
MapForce 関数として (core | string 関数), 1141
機密な情報,
並べ替え, 278
技術データ, 2001
疑問符,
不足しているアイテム, 133
区切り文字,
CSV ファイル内の変更, 655
フラットテキストファイルの変更, 666
空のフィールド,
CSV ファイル内, 655
フラットテキストファイル内, 666
繰り返される分割,
FlexText 関数として, 690
区切られた (フロート) モードの使用, 692
区切られた (行の開始) モードの使用, 695
区切られた (行ベース) モードの使用, 694
固定長モードの使用, 691
型優先,
接続, 160
結果,
ウィンドウ - データベース クエリ, 597
データベース クエリ内のアイコン, 597
検索,
iマッピング コンポーネント内のアイテム, 112
プロジェクトウィンドウ内のファイル, 138
検証する,
デザインのマッピング, 94

(C) 2015-2021 Altova GmbH

検証する,
マッピング 出力, 95
固定された,
固定長ファイル - マッピング, 647
構成,
mff ファイル, 1029
SQL エディター 設定, 600
混在, 152
コンテンツマッピング, 152
コンテンツマッピング サンプル, 157
コンテンツマッピングのメソッド, 152
ソース優先 マッピング, 152
作成,
領域, 592
削除, 160
コメント, 590
ブックマーク, 591
ブロックコメント, 590
ラインコメント, 590
削除 - 不足しているアイテム, 133
全てコピー 接続, 160
領域, 592
試用期間,
Altova ソフトウェア製品の試用, 2003
資格j情報,
FlowForce Server にデプロイ, 286
管理, 281
作成, 278
資格情報,
FlowForce Server 内の, 285
MapForce Server 内の, 283
自動,
ライブラリのロード, 1029
自動化された,
処理, 1390
主キー,
データベース コンポーネントのために生成, 490
データベースマッピング内, 473
データベースマッピング内の, 480
データベース挿入アクション中に生成する, 496
出力,
アプリケーションメニューとして, 1494
プレビュー, 96
検証, 95
保存, 96
出力パラメーター,
ユーザー定義関数内, 991
処理,
マッピングの自動化, 1390
処理命令,
ターゲット ファイルに追加, 359
処理命令とコメント,
マッピング, 153
署名,
デジタル署名, 371
設定, 374
照合,
Unicode コードポイント, 203
ロケール照合, 203
並べ替えコンポーネント, 203
証明書,
デジタル署名, 374
親コンテキスト,
サンプル, 312
正規表現,
FlexText トコンポーネントの分割, 721
FlexText 内での条件の変更の使用, 723
MapForce FlexText 内の, 721
ノード関数, 973
マッピング内での使用, 1048
生成,
デジタル署名, 371
生成されたコード,
から例外をスロー, 241
生成されたコード内のパス,
絶対パスに設定する, 107
精度,
ノード関数内での使用, 978
切り替え,
FlexText 関数として, 702
接続,
アプリケーションメニューとして, 1492
ルート要素の変更を保存する, 128
異なるコンポーネントへ移動する, 128
型優先, 160
折りたたみ,
領域, 592
設定,
テキスト フォント - データベース クエリ, 602
デジタル署名, 374
結果表示 - データベース クエリ, 601
先頭の 0,
変換中に保持する, 777
選択,
テーブル データ - データベース クエリ, 597
全てコピー,

全てコピー,
コネクタ, 160
とフィルター, 160
マッピングメソッド, 152
解決 / 削除 コネクタ, 160
挿入,
SQL WHERE, 561
アプリケーションメニューとして, 1485
コメント, 590
ブックマーク, 591
ブロックコメント, 590
ラインコメント, 590
領域, 592
単一のターゲット,
複数のソース, 367
単純型,
並べ替え, 203
値ウィンドウ,
コンテキストタブ, 335
について, 329, 335
関連したタブ, 335
履歴タブ, 335
値の保管,
FlexText 関数として, 719
著作権に関する情報, 2003
追加,
カスタムライブラリ, 1029
定数,
マッピングに追加, 958
展開,
領域, 592
電子証明書,
Linux への移行, 1384
Linux 上で信頼できる, 1372
Mac に転送する, 1385
Mac 上で信頼する, 1373
MapForce マッピング内, 1370
Windows からエクスポートする, 1378
Windows 上での管理, 1375
Windows 上で信頼できる, 1374
統合,
アプリケーション内の MapForce, 1828
動的な認証,
FlowForce Server 内の, 289
MapForce Server 内の, 289
構成, 288
入力の複製, 64
追加, 1490
入力パラメーター, 988
ユーザー定義関数内, 988, 991
認証,
OAuth 2.0, 278
基本的な HTTP, 278
派生した型,
マッピング, 353
背景情報, 2001
配布,
Altova ソフトウェア製品, 2003
Altova ソフトウェア製品の配布, 2003
標準,
マッピングメソッド, 152
標準的な XML,
デジタル署名, 374
表示,
アプリケーションメニューとして, 1497
評価機関,
Altova ソフトウェア製品, 2003
不足しているアイテム, 133
複合コンテンツ,
マッピング, 159
複合型,
並べ替え, 203
複数のソース,
単一のターゲットへ, 367
複数の連続した要素,
Edifact - X12, 802
文字の重点,
フラットテキストファイル内, 666
文字列,
からデータを解析する, 245
データのシリアル化, 245
データの解析, 245
データを以下から解析する, 247
へのデータのシリアル化, 245, 252
並べ替え,
データベース内, 564
テーブル データベース クエリ, 592
結果ウィンドウ内のデータ, 597
結果ウィンドウ内の列アイコン, 597
並べ替えコンポーネント, 203
並べ替えの順序,
変更, 203
並べ替キー,
並べ替えコンポーネント, 203
変換,
RaptorXML Server, 1389

変換言語,
選択, 93
変数,
のスコープの変更, 197
マッピングに追加, 193
使用例, 199, 200, 201
始めに, 192
編集,
アプリケーションメニューとして, 1484
法的な情報, 2003
無視する,
FlexText 関数として, 708
名前空間,
カスタムの宣言, 368
とワイルドカード (xs:any), 362
名前空間 URI,
DTD, 353
と QNames, 356
優先コンテキスト, 316
サンプル, 318
用語,
EDIFACT, 730
領域,
作成, 592
削除, 592
折りたたみ, 592
挿入, 592
展開, 592
例外,
追加, 241
列挙,
MapForceControl, 1892

(C) 2015-2021 Altova GmbH