Altova MapForce 2023 Professional Edition

Manual del usuario y referencia

Altova MapForce 2023 Professional Edition
Manual del usuario y referencia

Todos los derechos reservados. Ningún fragmento de esta publicación podrá ser reproducido de manera alguna (ya sea de forma gráfica, electrónica o mecánica, fotocopiado, grabado o reproducido en sistemas de almacenamiento y recuperación de información) sin el consentimiento expreso por escrito de su autor/editor.

Los productos a los que se hace referencia en este documento pueden ser marcas registradas de sus respectivos propietarios. El autor y editor no afirman ser propietarios de dichas marcas registradas.

Durante la elaboración de este documento se tomaron todas las precauciones necesarias para prevenir errores. Sin embargo, el autor y editor no se responsabilizan de los errores u omisiones que pudiese contener el documento ni de los posibles daños o perjuicios derivados del uso del contenido de este documento o de los programas y código fuente que vengan con el documento. Bajo ninguna circunstancia se podrá considerar al autor y editor responsables de la pérdida de beneficios ni de cualquier otro daño y perjuicio derivado directa o indirectamente del uso de este documento.

Fecha de publicación: 2022

© 2016-2022 Altova GmbH

Contenido

1 Introducción 17

1.1 Novedades 18
- 1.1.1 Versión 2023 18
- 1.1.2 Versión 2022 18
- 1.1.3 Versión 2021 19
- 1.1.4 Versión 2020 20
- 1.1.5 Versión 2019 20
- 1.1.6 Versión 2018 21

1.2 ¿Qué es MapForce? 22
- 1.2.1 Asignación: origen y destino 23
- 1.2.2 Lenguajes de transformación 24
- 1.2.3 Tipos de asignaciones 25
- 1.2.4 Integración con productos de Altova 27

1.3 Interfaz del usuario 29
- 1.3.1 Barras de herramientas 30
- 1.3.2 Ventanas 30
- 1.3.3 Ventana Mensajes 34
- 1.3.4 Paneles 35

2 Tutoriales 39

2.1 De esquema a esquema 40
- 2.1.1 Crear y guardar diseños 41
- 2.1.2 Agregar un componente de origen 43
- 2.1.3 Agregar componente de destino 44
- 2.1.4 Conectar origen y destino 45
- 2.1.5 Vista previa del resultado de la asignación 48

2.2 Varios archivos de origen a un solo destino 50
- 2.2.1 Preparar archivos de código fuente 51
- 2.2.2 Agregar componente de destino 52

2.2.3 Comprobar y configurar los archivos de entrada/salida ... 53
2.2.4 Conectar varios orígenes a un destino ... 54
2.3 Asignaciones encadenadas ... 57
2.3.1 Preparar el diseño de la asignación ... 57
2.3.2 Configurar el segundo archivo de destino ... 58
2.3.3 Conectar componentes de destino ... 60
2.3.4 Filtrar datos ... 60
2.3.5 Vista previa y guardar resultados ... 63
2.4 Varios archivos de origen a varios archivos de destino ... 64
2.4.1 Preparar el diseño de la asignación ... 66
2.4.2 Configurar el componente de entrada ... 67
2.4.3 Configurar el resultado, parte 1 ... 67
2.4.4 Configurar el resultado, parte 2 ... 70

3 Trabajar con asignaciones de datos 73
3.1 Trabajar con componentes ... 75
3.1.1 Agregar componentes a la asignación ... 77
3.1.2 Componentes básicos ... 79
3.1.3 Rutas de acceso de archivos ... 82
3.2 Trabajar con conexiones ... 87
3.2.1 Tipos de conexión ... 91
3.2.2 Configuración de las conexiones ... 97
3.2.3 Menú contextual de las conexiones ... 99
3.2.4 Mover conexiones y conexiones secundarias ... 101
3.2.5 Conservar conexiones tras eliminación de componentes ... 102
3.3 Tareas generales ... 105
3.3.1 Validar la asignación de datos ... 105
3.3.2 Generar código XSLT ... 107
3.3.3 Características de la vista Texto ... 108
3.3.4 Búsquedas en la vista Texto ... 111
3.3.5 Cambiar configuración de la asignación ... 115
3.4 Trabajar con proyectos de asignación de datos ... 118
3.4.1 Proyectos nuevos ... 118
3.4.2 Configurar proyectos ... 120

3.4.3 Carpetas de proyecto ... 121

4 Componentes estructurales 123

4.1 XML y esquemas XML ... 124
4.1.1 Configuración de componentes XML ... 125
4.1.2 Tipos XML Schema derivados ... 129
4.1.3 Valores Nil y Nillable ... 131
4.1.4 Comentarios e instrucciones de procesamiento ... 135
4.1.5 Secciones CDATA ... 136
4.1.6 Comodines: xs:any / xs:anyAttribute ... 137
4.1.7 Declarar espacios de nombres personalizados ... 140
4.1.8 Gestor de esquemas ... 143
4.2 Bases de datos ... 159
4.2.1 Conectarse a un origen de datos ... 161
4.2.2 Introducción a las asignaciones de datos de BD ... 246
4.2.3 Asignar datos a bases de datos ... 281
4.2.4 Combinar datos de BD ... 335
4.2.5 Filtrar y ordenar datos de BD (SQL WHERE/ORDER) ... 352
4.2.6 Instrucciones SQL SELECT como tablas virtuales ... 359
4.2.7 Asignaciones entre datos XML y campos de BD ... 369
4.2.8 Examinar y consultar bases de datos ... 380
4.2.9 Procedimientos almacenados ... 395
4.2.10 Gestionar excepciones de BD ... 422
4.2.11 Seguimiento de BD y registro de errores ... 435
4.3 Archivos CSV y archivos de texto ... 445
4.3.1 Ejemplo: asignar archivos CSV a XML ... 445
4.3.2 Ejemplo: recorrer elementos ... 447
4.3.3 Ejemplo: crear jerarquías a partir de archivos CSV y de texto de longitud fija ... 450
4.3.4 Opciones de configuración de componentes CSV ... 454
4.3.5 Ejemplo: asignar archivos de texto de longitud fija a bases de datos ... 458
4.3.6 Opciones de configuración de componentes FLF ... 462
4.4 HL7 versión 3 ... 469
4.5 Archivos binarios ... 471
4.5.1 Ejemplo: leer archivos binarios ... 473

4.5.2 Ejemplo: escribir archivos binarios ... 474

5 Componentes de transformación 476

5.1 Pasar parámetros a la asignación ... 477
5.1.1 Agregar componentes de entrada simples ... 478
5.1.2 Configurar componentes de entrada simples ... 479
5.1.3 Crear un valor de entrada predeterminado ... 481
5.1.4 Ejemplo: usar nombres de archivo como parámetros de asignación ... 482
5.2 Obtener valores de cadena de una asignación ... 488
5.2.1 Agregar componentes de salida simples ... 489
5.2.2 Ejemplo: vista previa de resultados de una función ... 490
5.3 Variables ... 492
5.3.1 Agregar variables ... 494
5.3.2 Cambiar el contexto y ámbito de las variables ... 498
5.3.3 Ejemplo: contar filas de tabla de BD ... 500
5.3.4 Ejemplo: filtrar y numerar nodos ... 501
5.3.5 Ejemplo: crear grupos y subgrupos de registros ... 503
5.4 Ordenar datos ... 505
5.4.1 Ordenar según varias claves ... 507
5.4.2 Ordenar con variables ... 509
5.5 Agrupar datos ... 511
5.5.1 Ejemplo: agrupar registros por clave ... 514
5.6 Combinar datos ... 517
5.6.1 Agregar condiciones de combinación ... 519
5.6.2 Combinar tres o más estructuras ... 522
5.6.3 Ejemplo: combinar estructuras XML ... 523
5.7 Añadir excepciones ... 529
5.7.1 Ejemplo: excepción en la condición Greater than ... 530
5.7.2 Ejemplo: excepción cuando un nodo no existe ... 530
5.8 Usar asignaciones de valores ... 533
5.8.1 Pasar datos sin modificarlos ... 537
5.8.2 Propiedades de las asignaciones de valores ... 539
5.9 Filtros y condiciones ... 543
5.9.1 Ejemplo: filtrar nodos ... 545

5.9.2 Ejemplo: devolver un valor de forma condicional ... 547

6 Funciones 549

6.1 Fundamentos de las funciones ... 550
6.2 Gestionar bibliotecas de funciones ... 553
6.2.1 Bibliotecas locales y globales ... 555
6.2.2 Rutas relativas de acceso a bibliotecas ... 556
6.3 Valores predeterminados y funciones de nodo ... 558
6.3.1 Configurar reglas ... 560
6.3.2 Casos de uso ... 563
6.3.3 Suministrar metadatos del nodo a funciones de nodo ... 572
6.4 Funciones definidas por el usuario ... 575
6.4.1 Crear funciones definidas por el usuario ... 576
6.4.2 Parámetros en funciones definidas por el usuario ... 581
6.4.3 Ejemplo: búsqueda recursiva ... 586
6.4.4 Implementación de la búsqueda ... 589
6.5 Funciones personales ... 593
6.5.1 Importar funciones XSLT 1.0/2.0 personales ... 593
6.5.2 Importar funciones XQuery 1.0 personales ... 600
6.5.3 Importar bibliotecas Java y .NET personales ... 604
6.5.4 Referencias manuales a bibliotecas Java, C# y C++ personales ... 612
6.6 Expresiones regulares ... 627
6.7 Referencia de la biblioteca de funciones ... 631
6.7.1 core | aggregate functions (agregado) ... 633
6.7.2 core | conversion functions (conversión) ... 640
6.7.3 core | file path functions (ruta de archivos) ... 656
6.7.4 core | generator functions (generador) ... 660
6.7.5 core | logical functions (lógica) ... 662
6.7.6 core | math functions (matemáticas) ... 668
6.7.7 core | node functions (nodo) ... 674
6.7.8 core | QName functions (QName) ... 679
6.7.9 core | sequence functions (secuencia) ... 681
6.7.10 core | string functions (cadena) ... 707
6.7.11 db (base de datos) ... 722

6.7.12 lang | datetime functions (fechaHora) 725
6.7.13 lang | file functions 744
6.7.14 lang | generator functions (generador) 746
6.7.15 lang | logical functions (lógica) 747
6.7.16 lang | math functions (matemáticas) 749
6.7.17 lang | QName functions (QName) 757
6.7.18 lang | string functions (cadena) 758
6.7.19 xpath2 | accessors (descriptores de acceso) 775
6.7.20 xpath2 | anyURI functions 777
6.7.21 xpath2 | boolean functions (booleanas) 778
6.7.22 xpath2 | constructors (constructores) 779
6.7.23 xpath2 | context functions (contexto) 780
6.7.24 xpath2 | durations, date, time functions (duración, fecha y hora) 783
6.7.25 xpath2 | node functions (nodo) 799
6.7.26 xpath2 | numeric functions (numéricas) 806
6.7.27 xpath2 | string functions (cadena) 807
6.7.28 xpath3 | external information functions 817
6.7.29 xpath3 | formatting functions 820
6.7.30 xpath3 | math functions 824
6.7.31 xpath3 | URI functions 830
6.7.32 xslt | xpath functions 832
6.7.33 xslt | xslt functions 835

7 Asignaciones avanzadas 839

7.1 Asignaciones en cadena 840
7.1.1 Ejemplo: paso a través activo 842
7.1.2 Ejemplo: paso a través inactivo 846
7.2 Asignar nombres de nodos 850
7.2.1 Obtener acceso al nombre de los nodos 851
7.2.2 Obtener acceso a determinado tipo de nodos 858
7.2.3 Ejemplo: asignar nombres de elemento a valores de atributo 863
7.2.4 Ejemplo: agrupar y filtrar nodos según su nombre 866
7.3 Reglas y estrategias de asignación 871
7.3.1 Secuencias 872

7.3.2 Contexto de la asignación ... 874
7.3.3 Contexto de prioridad ... 883
7.3.4 Multiple target components ... 888
7.4 Procesar varios archivos de entrada o salida simultáneamente ... 892
7.4.1 Asignar varios archivos de entrada a un solo archivo de salida ... 895
7.4.2 Asignar varios archivos de entrada a varios archivos de salida ... 896
7.4.3 Especificar nombres de archivo como parámetros de asignación ... 897
7.4.4 Vista previa de varios archivos de salida ... 898
7.4.5 Ejemplo: dividir un archivo XML en varios archivos ... 898
7.4.6 Ejemplo: dividir una tabla de BD en varios archivos XML ... 900
7.5 Redistribuir y serializar cadenas de texto ... 903
7.5.1 Información sobre el componente de análisis/serialización ... 903
7.5.2 Ejemplo: serializar datos a una cadena (de XML a BD) ... 906
7.6 Paneles de resultados de StyleVision ... 911
7.7 Generar la documentación de la asignación ... 915
7.7.1 Hojas de estilos predefinidas de StyleVision ... 916
7.7.2 Diseño personalizado ... 921

8 Depuración de asignaciones 924

8.1 Preparar la depuración ... 927
8.2 Información sobre el modo depurador ... 928
8.3 Agregar y quitar puntos de interrupción ... 931
8.4 Ventana Valores ... 934
8.5 Ventana Contexto ... 936
8.6 Ventana Puntos de interrupción ... 938
8.7 Vista previa parcial de resultados generados ... 940
8.8 Ver el valor actual de un conector ... 941
8.9 Retroceder en el pasado reciente ... 942
8.10 Ver historial de valores procesados por un conector ... 943
8.11 Establecer el contexto en un valor ... 945

9 Automatizar tareas de MapForce con otros productos de Altova 946

9.1 Automatización con RaptorXML Server 947
9.2 Automatización con MapForce Server 948
9.3 Preparar asignaciones para ejecutarlas en servidores 949
9.4 Compilar asignaciones en archivos de ejecución de MapForce Server 955
9.5 Implementar asignaciones en FlowForce Server 958
9.6 Interfaz de la línea de comandos de MapForce 963

10 Recursos globales de Altova 969

10.1 Crear recursos globales 970
10.2 Global Resource Setup Part 2 972
10.3 Ejemplo: ejecutar asignación con archivos de entrada variables 976
10.4 Ejemplo: generar resultados en carpetas variables 978
10.5 Bases de datos como recursos globales 980
10.6 Resultados de transformaciones de MapForce y StyleVision como recursos globales 982
10.7 Recursos globales en entornos de ejecución 986
10.7.1 Recursos globales en en el código generado 986
10.7.2 Recursos globales en MapForce Server 987
10.7.3 Recursos globales en FlowForce Server 987

11 Catálogos en MapForce 990

11.1 Funcionamiento de los catálogos 991
11.2 Estructura de los catálogos de MapForce 993
11.3 Personalizar catálogos 995
11.4 Variables de entorno 997

12 Complemento de MapForce para Visual Studio 999

13 Complemento de MapForce para Eclipse 1002

13.1 Instalación 1003
13.2 Perspectiva MapForce 1005
13.3 Acceso a menús y funciones frecuentes 1008

13.4 Trabajar con asignaciones y proyectos ... 1012
13.4.1 Crear un proyecto MapForce/Eclipse ... 1012
13.4.2 Crear asignaciones nuevas ... 1014
13.4.3 Importar asignaciones a un proyecto de Eclipse ... 1016
13.4.4 Configurar la generación automática de código MapForce ... 1019
13.5 Ampliar el complemento de MapForce para Eclipse ... 1022

14 Comandos de menú 1024

14.1 Archivo ... 1025
14.2 Edición ... 1028
14.3 Insertar ... 1029
14.4 Proyecto ... 1032
14.5 Componente ... 1034
14.6 Conexión ... 1036
14.7 Función ... 1037
14.8 Resultados ... 1038
14.9 Depuración ... 1040
14.10 Vista ... 1042
14.11 Herramientas ... 1044
14.11.1 Personalizar menús ... 1045
14.11.2 Personalizar teclas de acceso rápido ... 1047
14.11.3 Opciones ... 1049
14.12 Ventanas ... 1061
14.13 Ayuda ... 1062

15 Generador de código 1067

15.1 Novedades ... 1069
15.2 Generar código C++ ... 1071
15.2.1 Generar código a partir de una asignación de datos ... 1072
15.2.2 Generar código a partir de un proyecto de asignación ... 1073
15.2.3 Compilar el proyecto ... 1074
15.2.4 Ejecutar la aplicación ... 1074
15.3 Generar código C# ... 1076

15.3.1 Seleccionar la plataforma de destino ... 1077
15.3.2 Generar código a partir de una asignación de datos ... 1078
15.3.3 Generar código a partir de un proyecto de asignación ... 1079
15.3.4 Compilar el proyecto ... 1080
15.3.5 Compilar proyectos .NET 5 y .NET Core ... 1081
15.4 Generar código Java ... 1083
15.4.1 Generar código a partir de una asignación de datos ... 1084
15.4.2 Generar código a partir de un proyecto de asignación ... 1085
15.4.3 Compilar el proyecto con Ant ... 1086
15.4.4 Control de referencias JDBC ... 1087
15.4.5 Ejemplo: generar y ejecutar código Java ... 1089
15.5 Integrar código generado con MapForce ... 1097
15.5.1 Ejemplo: Java ... 1098
15.5.2 Ejemplo: C# ... 1100
15.5.3 Ejemplo: C++ ... 1102
15.5.4 Cambiar tipo de datos de entrada/salida de la asignación (C# y Java) ... 1103
15.6 Generar código fuente a partir de esquemas XML o DTD ... 1110
15.6.1 Bibliotecas contenedoras de esquemas (C++) ... 1113
15.6.2 Bibliotecas contenedoras de esquemas (C#) ... 1115
15.6.3 Bibliotecas contenedoras de esquemas (Java) ... 1118
15.6.4 Integrar las bibliotecas contenedoras de esquemas ... 1120
15.6.5 Ejemplo: biblioteca ... 1123
15.6.6 Ejemplo: orden de compra ... 1147
15.7 Referencia de las clases generadas (C++) ... 1155
15.7.1 altova::DateTime ... 1155
15.7.2 altova::Duration ... 1159
15.7.3 altova::DayTimeDuration ... 1161
15.7.4 altova::YearMonthDuration ... 1162
15.7.5 altova::meta::Attribute ... 1162
15.7.6 altova::meta::ComplexType ... 1163
15.7.7 altova::meta::Element ... 1164
15.7.8 altova::meta::SimpleType ... 1165
15.7.9 [SuEsquema]::[CDoc] ... 1166
15.7.10 [SuEsquema]::[ElementType] ... 1168

15.7.11 [SuEsquema]::MemberAttribute ... 1170
15.7.12 [SuEsquema]::MemberElement ... 1170
15.8 Referencia de las clases generadas (C#) ... 1172
15.8.1 Altova.Types.DateTime ... 1172
15.8.2 Altova.Types.DateTimeFormat ... 1175
15.8.3 Altova.Types.Duration ... 1177
15.8.4 Altova.Xml.Meta.Attribute ... 1179
15.8.5 Altova.Xml.Meta.ComplexType ... 1180
15.8.6 Altova.Xml.Meta.Element ... 1181
15.8.7 Altova.Xml.Meta.SimpleType ... 1181
15.8.8 [SuEsquema].[Doc] ... 1182
15.8.9 [SuEsquema].[ElementType] ... 1184
15.8.10 [SuTipoEsquema].MemberAttribute ... 1185
15.8.11 [SuTipoEsquema].MemberElement ... 1186
15.9 Referencia de las clases generadas (Java) ... 1188
15.9.1 com.altova.types.DateTime ... 1189
15.9.2 com.altova.types.Duration ... 1194
15.9.3 com.altova.xml.meta.Attribute ... 1197
15.9.4 com.altova.xml.meta.ComplexType ... 1198
15.9.5 com.altova.xml.meta.Element ... 1199
15.9.6 com.altova.xml.meta.SimpleType ... 1199
15.9.7 com.[SuEsquema].[Doc] ... 1200
15.9.8 com.[SuEsquema].[ElementType] ... 1202
15.9.9 com.[SuEsquema].[SuTipoEsquema].MemberAttribute ... 1203
15.9.10 com.[SuEsquema].[SuTipoEsquema].MemberElement ... 1204
15.10 Consejos para la generación de código ... 1206
15.11 Opciones del generador de código ... 1207
15.12 SPL: el lenguaje de programación Spy ... 1209
15.12.1 Estructura básica de SPL ... 1209
15.12.2 Declaraciones ... 1210
15.12.3 Variables ... 1211
15.12.4 Variables predefinidas ... 1212
15.12.5 Crear archivos de salida ... 1214
15.12.6 Operadores ... 1215

15.12.7 Condiciones 1215
15.12.8 Colecciones y foreach 1216
15.12.9 Subrutinas 1218
15.12.10 Tipos integrados 1220

16 La API de MapForce 1225

16.1 Acceder a la API 1226
16.2 El modelo de objetos 1229
16.3 Gestión de errores 1230
16.4 Ejemplo de proyecto en C# 1232
16.5 Ejemplo de proyecto en Java 1236
16.6 Ejemplos en JScript 1240
16.6.1 Iniciar la aplicación 1240
16.6.2 Acceso a documentos simples 1241
16.6.3 Generar código 1242
16.6.4 Ejemplo: generación de código 1244
16.6.5 Ejemplo: ejecución de asignaciones 1246
16.6.6 Ejemplo: ayuda con el proyecto 1249
16.7 Referencia de objetos 1254
16.7.1 Interfaces 1254
16.7.2 Enumeraciones 1412

17 ActiveX Integration 1419

17.1 Prerequisites 1420
17.2 Adding the ActiveX Controls to the Toolbox 1422
17.3 Integration at Application Level 1424
17.4 Integration at Document Level 1427
17.5 ActiveX Integration Examples 1431
17.5.1 C# 1431
17.5.2 Java 1438
17.5.3 VB.NET 1447
17.6 Command Reference 1450
17.6.1 "File" Menu 1450

17.6.2 "Edit" Menu ... 1451
17.6.3 "Insert" Menu ... 1452
17.6.4 "Project" Menu ... 1452
17.6.5 "Component" Menu ... 1453
17.6.6 "Connection" Menu ... 1454
17.6.7 "Function" Menu ... 1455
17.6.8 "Output" Menu ... 1455
17.6.9 "Debug" Menu ... 1456
17.6.10 "View" Menu ... 1456
17.6.11 "Tools" Menu ... 1457
17.6.12 "Window" Menu ... 1458
17.6.13 "Help" Menu ... 1458
17.7 Object Reference ... 1459
17.7.1 MapForceCommand ... 1459
17.7.2 MapForceCommands ... 1461
17.7.3 MapForceControl ... 1462
17.7.4 MapForceControlDocument ... 1470
17.7.5 MapForceControlPlaceHolder ... 1476
17.7.6 Enumerations ... 1478

18 Anexos 1480

18.1 Notas sobre compatibilidad ... 1481
18.1.1 Orígenes y destinos de datos compatibles ... 1481
18.1.2 Funciones compatibles en el código generado ... 1482
18.2 Información sobre motores ... 1485
18.2.1 Información sobre motores XSLT y XQuery ... 1485
18.2.2 Funciones XSTL y XPath/XQuery ... 1490
18.3 Datos técnicos ... 1590
18.3.1 Requisitos de SO y memoria ... 1590
18.3.2 Motores XSLT y XQuery de Altova ... 1590
18.3.3 Compatibilidad con Unicode ... 1591
18.3.4 Uso de Internet ... 1591
18.4 Información sobre licencias ... 1593
18.4.1 Distribución electrónica de software ... 1593

18.4.2 Activación del software y medición de licencias..... 1594
18.4.3 Contrato de licencia para el usuario final..... 1595

Índice 1596

1 Introducción

MapForce® 2023 Professional Edition es una herramienta de asignación de datos visual para crear proyectos avanzados de integración de datos. MapForce® es una aplicación para Windows de 32/64 bits compatible con Windows Server 2008 R2 SP1 con actualización de la plataforma o superior y Windows 7 SP1 con actualización de la plataforma, Windows 8, Windows 10, Windows 11. Las ediciones Enterprise y Professional están disponibles en 32 y 64 bits. MapForce también puede integrarse con Visual Studio y Eclipse, así como con los productos de Microsoft Office (*véase* *Notas sobre compatibilidad*[1481]).

Última actualización: 10 October 2022

1.1 Novedades

En esta sección se describen las novedades de cada versión nueva de MapForce. Para más detalles consulte el apartado de la versión correspondiente.

1.1.1 Versión 2023

Versión 2023

- Hemos añadido compatibilidad con estos temas: *Clásico*, *Ligero* y *Oscuro*. Visite el apartado Ventanas 1061 para obtener más información.
- Mejoras y actualizaciones internas.
- Hemos ampliado la compatibilidad con Eclipse a las versiones: 2022-09, 2022-06, 2022-03, 2021-12 (*ediciones Professional y Enterprise*). Para más detalles Véase Complemento de MapForce para Eclipse 1002.
- Compatibilidad con mensajes ODETTE EDI (*Enterprise Edition*).
- Compatibilidad con la Especificación XII Transformation Registry 5 (*Enterprise Edition*).
- Ahora es posible crear parámetros en funciones definidas por el usuario 583 basados en BD y variables 493 con una estructura de tablas relacionadas (*ediciones Professional y Enterprise*).
- Ahora es posible enviar estructuras de solicitud `application/x-www-form-urlencoded` a un servicio REST (*Enterprise Edition*).
- Compatibilidad con los directorios UN/EDIFACT D.21B y D.22A (*Enterprise Edition*).
- Compatibilidad con SQLite 3.39.2, MariaDB MySQL 10.9.2, PostgreSQL 14.5 (*ediciones Professional y Enterprise*). Para más información sobre la compatibilidad con bases de datos consulte Bases de datos 159.
- Compatibilidad con el Gestor de esquemas XML 143, con el que podrá instalar y administrar esquemas XML de forma centralizada para todas las aplicaciones de Altova compatibles con XBRL.
- Compatibilidad con delimitadores EDI asignables (*Enterprise Edition*). Esta funcionalidad es compatible actualmente con estos estándares EDI: EDIFACT, X12 y NCPDP SCRIPT.

1.1.2 Versión 2022

Versión 2022 Release 2

- Hemos ampliado la compatibilidad con Eclipse 1002 a las versiones: 2021-12, 2021-09; 2021-06; 2021-03 (*ediciones Professional y Enterprise*).
- El complemento de MapForce para Visual Studio 999 y la generación de código 1067 (*ediciones Professional y Enterprise*) son compatibles con Visual Studio 2022.
- Generación de código 1067 compatible con NET 6.0 (*ediciones Professional y Enterprise*).
- También hemos ampliado la compatibilidad con bases de datos a: PostgreSQL 14, SQLite 3.37.2, MariaDB 10.6.5, MySQL 8.0.28, IBM DB2 11.5.7 (*ediciones Professional y Enterprise*). Para más información sobre la compatibilidad con bases de datos consulte Bases de datos SQL 159.
- Ahora se puede generar una vista previa de las imágenes en la ventana **Proyecto**. Para más información consulte Aspectos básicos de un proyecto 120 (*ediciones Professional y Enterprise*).

© 2016-2022 Altova GmbH

Versión 2022

- Hemos ampliado la compatibilidad con Eclipse 1002 a las versiones: 2021-09; 2021-06; 2021-03; 2020-12 (*ediciones Professional y Enterprise*).
- Las conexiones de copia total 95 ahora también son compatibles con JSON. Esta funcionalidad solo está disponible para tipos compatibles con JSON (*solo en la edición Enterprise*).
- En StyleVision existe un panel de resultados nuevo 911 llamado **Texto**. Si adjunta un archivo SPS a un componente puede acceder a una vista previa del formato de salida en texto plano en MapForce (*en las ediciones Professional y Enterprise*).
- Compatibilidad con JSON Schema en variables 492 y parámetros 581 (*solo en la edición Enterprise*).

1.1.3 Versión 2021

Versión 2021 Release 3

- Compatibilidad ampliada para JSON Schema Draft 2019-09 y Draft 2020-12. (*Solo en la edición Enterprise Edition*).

Versión 2021 Release 2

- Ahora se puede usar XSLT 3.0 como lenguaje de asignación, véase Generar código XSLT 107. Además, MapForce ahora incluye funciones integradas nuevas compatibles con XSLT 3.0, véase la referencia de la biblioteca de funciones 631.
- Al generar código en C# puede seleccionar como destino .NET Core 3.1 y .NET 5.0 desde las opciones de generación de código 1207 (además de la compatibilidad que ya existe con proyectos .NET Framework) . Para más detalles consulte Generar código C# 1076.
- Otras mejoras y actualizaciones internas.

Versión 2021

- Las asignaciones de MapForce ahora pueden leer datos BLOB (objeto binario grande) de archivos binarios y escribir archivos binarios en disco. Esto permite, por ejemplo, leer campos BLOB de una BD y guardarlos en disco como archivos de imagen, o leer archivos binarios como PDFs desde el disco y guardarlos como campos `xs:base64Binary` dentro de un archivo XML. Consulte Archivos binarios 471 para más información.
- Hemos ampliado la compatibilidad con a versiones nuevas de BD: MariaDB 10.4, 10.5
- Hemos ampliado la compatibilidad con a versiones nuevas de Eclipse: 2019.09, 2019.12, 2020.03, 2020.06
- Al vincular varias tablas o vistas de BD con componentes de unión SQL en una asignación puede usar tanto el vínculo externo izquierdo o el vínculo interno; consulte Combinaciones en modo SQL 337.
- Otras mejoras y actualizaciones internas.

1.1.4 Versión 2020

Versión 2020 Release 2

- La ventana nueva Gestionar bibliotecas 553 permite ver y gestionar todas las bibliotecas de funciones importadas por la asignación actual, así como gestionarlas a nivel de programa (esto incluye las funciones de MapForce definidas por el usuario y otros tipos de biblioteca). Por ejemplo, puede copiar y pegar fácilmente funciones definidas por el usuario de una asignación a otra (véase Copiar y pegar funciones definidas por el usuario entre asignaciones 580).
- Cuando un archivo de asignación importa bibliotecas, la ruta de los archivos de biblioteca importados es relativa a la asignación por defecto (véase Rutas relativas de acceso a bibliotecas 556).
- Si un archivo de asignación importa bibliotecas XSLT o XQuery, se puede generar código XSLT o XQuery que use una ruta relativa para hacer referencia a los archivos de biblioteca importados. Puede encontrar esta opción en el cuadro de diálogo "Configurar asignación" 115.
- Hemos ampliado la API de MapForce 1225 para que pueda administrar las bibliotecas importadas de forma programática (por ejemplo, añadir o eliminar bibliotecas).
- El código generado para las bibliotecas contenedoras de esquemas XML ahora permite tener un mayor control sobre los espacios de nombre y los prefijos de los elementos. Existen métodos nuevos para declarar o sobrescribir espacios de nombres para un elemento o para anexar un elemento con un espacio de nombre prefijado. Consulte Ejemplo: orden de compra 1147.
- Compatibilidad con las versiones nuevas de las bases de datos PostgreSQL 12.1 e Informix 14.10.
- Otras mejoras y actualizaciones internas

Versión 2020

- El Complemento de MapForce para Visual Studio 999 y el Generador de código 1067 ahora son compatibles con Visual Studio 2019.
- Compatibilidad con Eclipse 4.9 - 4.12 (*véase Complemento de MapForce para Eclipse* 1002).
- Si un paquete Oracle contiene funciones o procedimientos almacenados públicos, estos también estarán disponibles en la asignación (véase Agregar procedimientos almacenados a la asignación 398).
- Puede configurar un componente de BD para que los nombres de los objetos de BD se traten como relativos al esquema predeterminado y no a uno en concreto. Esto ahorra tiempo si en un futuro tiene que cambiar a una BD distinta (*véase Cambiar de BD y esquema* 255).
- Puede implementar en FlowForce Server recursos globales creados en MapForce (*véase Implementar recursos en FlowForce Server* 987).
- Cuando reemplace valores con ayuda de una tabla de consulta puede pegar datos tabulares (pares clave-valor) desde un origen externo, como CSV o Excel, en la asignación. Además, ahora es más fácil manejar los casos en los que no se encuentra un valor en la tabla de consulta predefinida (para procesar esos valores ya no es necesario usar la función `subsitute-missing`. Consulte Usar asignaciones de valores 533.
- Otras mejoras y actualizaciones internas.

1.1.5 Versión 2019

Versión 2019 Release 3

- Se han optimizado partes principales de la interfaz del usuario para pantallas con alta densidad de píxeles (HiDPI)

 © 2016-2022 Altova GmbH

- Compatibilidad con Seguimiento de BD y registro de errores 435
- Compatibilidad mejorada para gestionar transacciones de BD en código de programa generado y la ejecución en servidor. Si ocurre un error cuando una asignación actualiza una BD ahora se puede continuar ejecutando la asignación incluso aunque algunas operaciones fallen (*véase Gestionar excepciones de BD* 422).
- Una asignación puede codificar o descodificar datos en Base 64 con ayuda de las nuevas funciones charset-encode 761 y charset-decode 759.
- Compatibilidad con versiones nuevas de BD: PostgreSQL 11, Db2 para i 7.2 y 7.3
- Compatibilidad con Eclipse 4.7 - 4.10 (*véase MapForce Plug-in for Eclipse* 1002) 1002
- Opción de indicar de forma explícita la ruta de acceso al equipo virtual Java desde MapForce (*véase Java Settings* 1052) 1052
- Otras mejoras y actualizaciones internas

Versión 2019

- Hemos ampliado la compatibilidad a las versiones de BD más recientes: Firebird 3.0, Informix 12.10, MariaDB 10.2, MySQL 8.0, PostgreSQL 10, SQL Server 2017 y SQL Server para Linux (además de ser compatible con versiones anteriores)
- Puede aplicar valores predeterminados de forma selectiva a nodos concretos que cumplan los criterios que defina de forma personalizada. Por ejemplo, puede aplicar una función de nodo solo a nodos de un tipo específico cuyo nombre coincida con una expresión regular (*véase Aplicar funciones de nodo y valores predeterminados de forma condicional* 568)
- Al crear funciones de nodo puede procesar datos de forma condicional basándose en metadatos como el nombre o una anotación del nodo (*véase Suministrar metadatos del nodo a funciones de nodo* 572)
- Las asignaciones que contienen funciones de agrupamiento se ejecutan más rápido en código generado en C#, C++ o Java
- Mejoras y actualizaciones internas

1.1.6 Versión 2018

Versión 2018 Release 2

- Compatibilidad con las siguientes bases de datos: MariaDB 10, Teradata 16
- Las funciones integradas, las definidas por el usuario y las constantes se pueden agregar a las asignaciones haciendo doble clic en un área vacía de la asignación (*véase Funciones básicas* 550)
- Actualizaciones y mejoras internas

Versión 2018

- Opción para generar código de programa para Visual Studio 2013, 2015 y 2017 (*véase Generador de código* 1067).
- Compatibilidad con estas versiones de bases de datos: Sybase ASE 16, PostgreSQL 9.6, MySQL 5.7
- Actualizaciones internas y mejoras.

1.2 ¿Qué es MapForce?

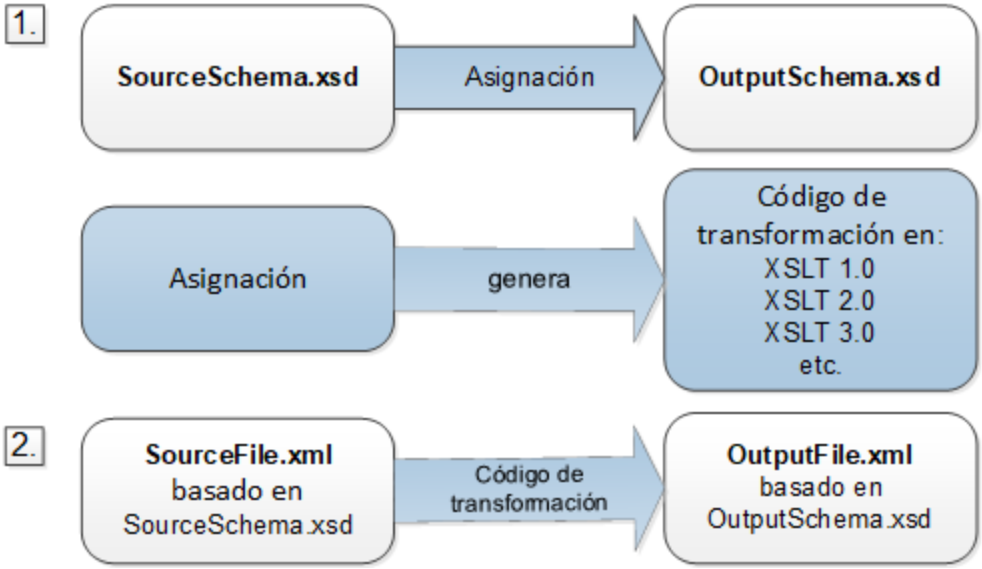
Sitio web de Altova: Herramienta de asignación de datos

MapForce es una potente y flexible herramienta con la que puede transformar de forma gráfica datos de todo tipo de formatos. Consulte Asignación: origen y destino 23 para ver una lista completa de los formatos disponibles. MapForce permite asignar un archivo de origen a uno de destino, un archivo de origen a varios de destino, varios archivos de origen a uno de destino o varios archivo de origen a varios archivos de destino. Para saber más sobre asignaciones consulte Tipos de asignaciones 25 y Tutoriales 39. MapForce también cuenta con numerosas opciones de procesamiento y filtrado de datos, como las funciones 549, las variables 492 o los filtros y las condiciones 543.

Para poder ejecutar una asignación debe indicar una estructura de datos para cada uno de los archivos de origen y de destino. Por ejemplo, un esquema XML define la estructura de un archivo XML. Los datos se asignan (del origen al destino) mediante acciones de arrastrar y colocar en la interfaz gráfica del usuario. No es necesario que escriba código de programa para la asignación, ya que MapForce lo genera automáticamente. Puede usar este código para transformar documentos con la estructura de datos de origen en documentos con la estructura de datos de destino.

Modelo abstracto

El modelo abstracto que ve a continuación ilustra una de las transformaciones de datos básicas que puede realizar con MapForce. El primer paso muestra una estructura, **`SourceSchema.xsd`**, que se asigna a otra estructura abstracta, **`OutputSchema.xsd`**. La asignación genera código de transformación en el lenguajes de transformación seleccionado. El segundo paso muestra que el contenido del archivo XML de origen, que se basa en **`SourceSchema.xsd`**, se asigna al archivo XML de destino, que se basa en **`OutputSchema.xsd`**. La asignación del contenido de origen a destino se realiza con el código de transformación generado en el paso anterior.



Todas las ediciones de MapForce tienen una versión de 32 bits. Además, las ediciones Enterprise y Professional están disponibles en 64 bits.

 © 2016-2022 Altova GmbH

Notas generales

Puede encontrar los archivos de asignación que se usan o a los que se hace referencia en el manual en estas ubicaciones:

- `C:\Usuario\<usuario>\Documentos\Altova\MapForce2023\MapForceExamples`
- `C:\Usuario\<usuario>\Documentos\Altova\MapForce2023\MapForceExamples\Tutorial`
- `C:\Usuario\<usuario>\Documentos\Altova\MapForce2023\MapForceExamples\Tutorial\BasicTutorials`

Apartados de esta sección

Estos son los apartados de esta sección:

- Asignación: Origen y destino 23
- Lenguajes de transformación 24
- Tipos de asignaciones 25
- Integración con productos de Altova 27

1.2.1 Asignación: origen y destino

En MapForce, origen y destino son términos esenciales que hacen referencia a las estructuras de datos desde o hacia las que se asignan los datos. Estas son las tecnologías que puede usar como origen o destino de las asignaciones de datos en MapForce:

MapForce Basic Edition

- XML y XML schema

MapForce Professional Edition

- XML y XML schema
- Archivos planos, incluido el formato CSV (valores separados por comas) y FLF (campo de longitud fija)
- Bases de datos (las principales bases de datos relacionales, incluidas las de Microsoft Access y SQLite)
- Archivos binarios (contenido BLOB sin formato)

MapForce Enterprise Edition

- XML y XML schema
- Archivos planos, incluido el formato CSV (valores separados por comas) y FLF (campo de longitud fija)
- Los datos de archivos de texto heredados pueden asignarse y convertirse a otros formatos con la aplicación MapForce FlexText
- Bases de datos SQL (las principales bases de datos relacionales, incluidas las de Microsoft Access y SQLite)
- Bases de datos NoSQL
- Archivos binarios (contenido BLOB sin formato)
- Familia de formatos EDI (incluidos UN/EDIFACT, ANSI X12, HL7, IATA PADIS, SAP IDoc, TRADACOMS)
- Archivos JSON

- Archivos de Microsoft Excel 2007 y posteriores
- Archivos de instancia y taxonomías XBRL
- Protocol Buffers

1.2.2 Lenguajes de transformación

En MapForce los lenguajes de transformación sirven para generar el código de transformación que lleva a cabo las asignaciones. Puede seleccionar o modificar el lenguaje de transformación en cualquier momento. MapForce permite visualizar el código de transformación en el lenguaje seleccionado. Para más información consulte Generar documentación XSLT 107. También puede generar este código con el comando de menú **Archivo | Generar código** en y usar ese código para transformar cualquier documento de datos que sea válido conforme al esquema del componente de entrada. Según la edición de MapForce puede elegir uno de estos lenguajes para la transformación de datos:

MapForce Basic Edition	Ediciones MapForce Professional y Enterprise
• XSLT 1.0 • XSLT 2.0 • XSLT 3.0	• XSLT 1.0 • XSLT 2.0 • XSLT 3.0 • BUILT-IN • XQuery • Java • C# • C++

Si quiere seleccionar todas las tablas, tiene dos opciones:

- En el menú **Resultados** haga clic en el nombre del lenguaje que quiere usar para la transformación.
- Haga clic en el lenguaje en la barra de herramientas de **selección de lenguajes** (*véase más abajo*).

Si selecciona otro lenguaje de transformación, ciertas características de MapForce pueden ser incompatibles. Para saber más consulte Notas sobre compatibilidad 1481.

Durante el proceso de diseño MapForce valida constantemente la integridad de los esquemas y de las transformaciones. Si hay errores de validación, los muestra en la ventana Mensajes 34. Esto permite revisar y corregir esos errores al instante.

Lenguajesdetransformación en las ediciones MapForce Professional y Enterprise
Cuando elige Java, C# o C++ como lenguaje de transformación, MapForce genera los proyectos y las soluciones necesarios que necesita para que se puedan abrir directamente en Visual Studio o Eclipse. Cuando se trate de un proyecto de integración de datos más complejo, podrá añadir código al programa generado con ayuda de las bibliotecas de Altova y de la API de MapForce.

BUILT-IN
Cuando se selecciona el lenguaje de transformación integrado BUILT IN para la asignación, MapForce utiliza su motor de transformación nativo para ejecutar la asignación de datos. MapForce también usa implícitamente

 © 2016-2022 Altova GmbH

esta opción cada vez que se genera la vista previa de los resultados cuando el lenguaje de transformación elegido es Java, C# o C++.

El motor BUILT-IN ejecuta asignaciones sin necesidad de procesadores externos, por lo que puede ser una buena elección si no dispone de mucha memoria. Si no necesita generar código de programa en un lenguaje concreto, use BUILT-IN como opción predeterminada, ya que es compatible con la mayoría de funciones de MapForce, al contrario que muchos otros lenguajes, como se explica en Notas sobre compatibilidad 1481. Además, si selecciona BUILT-IN como lenguaje de transformación puede ejecutar la asignación en MapForce Server. Para obtener más información consulte Automatizar asignaciones de datos 946.

1.2.3 Tipos de asignaciones

Los tipos de asignaciones pueden variar en función de: (i) origen y destino y (ii) la complejidad de la asignación Se pueden usar distintas estructuras de datos como origen y destino: por ejemplo, XML Schema, un archivo XML al que se le asigna un esquema, bases de datos, etc. Para obtener más información sobre formatos de origen y destino consulte Asignaciones: Origen y destino 23

La complejidad de los diseños de asignación incluye pero no se limita a:

- Asignar un archivo de origen a un archivo de destino. Para más información sobre este tipo de asignación consulte Tutorial 1 40.
- Combinar varios orígenes de datos en un único destino. Visite Tutorial 2 50 para obtener más información.
- Filtrar los datos de forma que solamente un subconjunto de estos se asigne archivo de destino. Consulte Tutorial 3 57.
- Asignar la estructura y el contenido del archivo de origen al archivo de destino. Consulte Tutorial 4 64.

Independientemente de la tecnología que utilice, MapForce determina automáticamente la estructura de los datos o sugiere que se asigne un esquema a esos datos. MapForce también puede generar esquemas a partir de un archivo de instancia de muestra. Por ejemplo, si está trabajando con un archivo de instancia XML pero no tiene una definición de esquema, MapForce puede generar uno. De esta forma MapForce hace que los datos del archivo XML se puedan asignar a otros archivos o formatos. Para obtener más información sobre las características y los términos básicos de MapForce consulte Tareas generales 73 e Interfaz del usuario 29.

Para administrar y acceder a los diseños de las asignaciones de datos de forma más fácil puede organizarlos en proyectos. Esta opción está disponible en las ediciones MapForce Professional y Enterprise. Además de generar código para asignaciones individuales dentro del proyecto, también puede generar código de programa a partir de proyectos enteros.

En MapForce no solo puede personalizar el aspecto del entorno de desarrollo, sino que además puede configurar todos los tipos de componente y de formato: Por ejemplo:

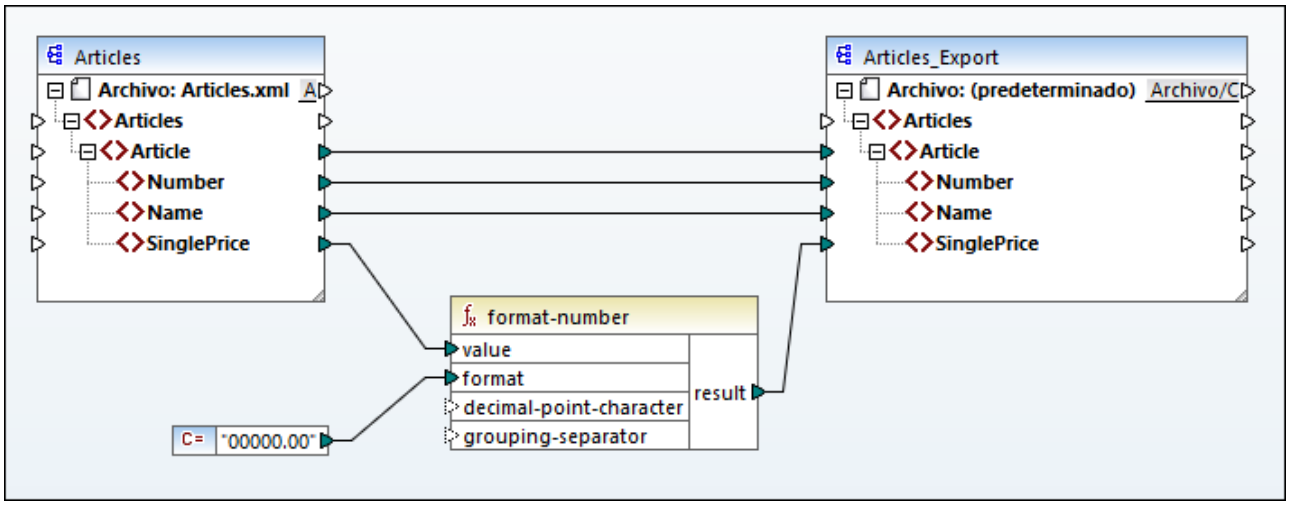
- Si se trata de una asignación de datos XML, puede elegir (i) si se incluye una referencia de esquema o (ii) si la declaración XML se suprime en los archivos XML de salida. También puede elegir la codificación de los archivos generados (p. ej. UTF-8).
- Al asignar bases de datos puede configurar como el tiempo de espera para que se ejecuten instrucciones. También puede definir si MapForce debe usar transacciones de BD o si se debe eliminar el nombre del esquema de BD de los nombres de tabla al generar el código.

- Si se trata de una asignación de datos XBRL, puede seleccionar las vistas que MapForce debe mostrar (p. ej. la vista **Bases de enlaces de presentación y definición**, la vista **Base de enlaces de tabla** o la vista **Todos los conceptos**).

Los ejemplos siguientes ilustran algunos diseños de asignación que usan el mismo tipo de estructuras de origen y destino (*ejemplo 1*) o diferentes (*ejemplo 2*). Los dos ejemplos son simples en cuanto que solo tienen un componente de datos de origen y uno de destino. Para saber más sobre asignaciones, tareas y procedimientos avanzados, consulte Asignaciones avanzadas 839.

Ejemplo 1: Asignación XML

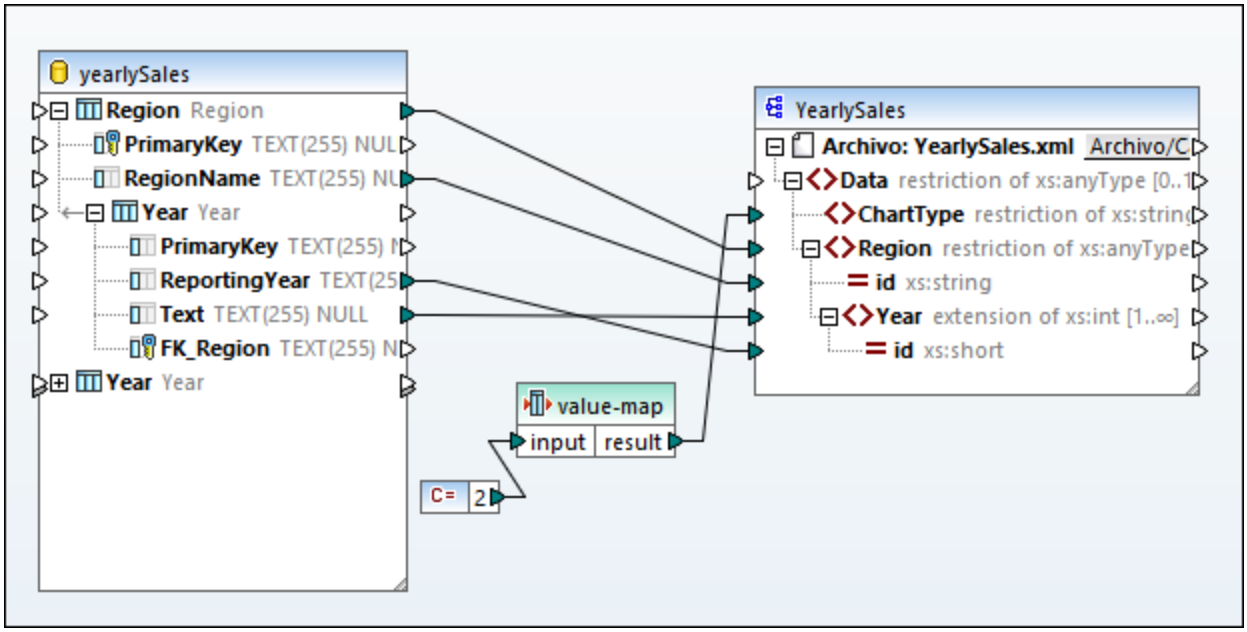
MapForce permite diseñar todas las transformaciones de asignación de forma visual. Por ejemplo, si trabaja con XML puede conectar cualquiera de los elementos y atributos del archivo XML a un elemento o atributo de otro archivo XML. Esto quiere decir que le está indicando a MapForce que lea los datos del nodo de origen y los escriba en el nodo de destino. A continuación puede ver un ejemplo de la transformación de un archivo XML en otro:



Ejemplo 2: Correspondencias con elementos de BD

Si trabaja con bases de datos en las ediciones MapForce Professional o Enterprise puede ver las columnas de BD en el área de asignaciones de datos de MapForce y asignar los datos de forma visual. Recuerde que cuando configure la conexión a la BD, podrá elegir el controlador de BD y el tipo de conexión (ADO, ADO.NET, ODBC o JDBC). Por último, podrá generar consultas SQL de forma gráfica, utilizar procedimientos almacenados o consultar la BD directamente (aunque las opciones disponibles dependen del tipo de BD, la edición y el controlador). A continuación puede ver un ejemplo de una transformación de datos de una BD a un archivo XML:

 © 2016-2022 Altova GmbH



1.2.4 Integración con productos de Altova

Puede ejecutar transformaciones en MapForce con los motores XSLT/XQuery integrados. MapForce también se puede usar en combinación con otros productos de Altova (*véase más abajo*).

XMLSpy
Si tiene XMLSpy instalado en el mismo equipo que MapForce, podrá abrir y editar cualquier tipo de archivo compatible en XMLSpy desde MapForce directamente Por ejemplo, si hace clic con el botón derecho en un componente XML de MapForce, el menú contextual incluye el comando **Componente | Editar la definición del esquema en XMLSpy**.
RaptorXML Server
Puede ejecutar el código XSLT generado directamente en MapForce y obtener una vista previa del resultado de la transformación de datos. Cuando necesite un mayor rendimiento, puede procesar la asignación con RaptorXML Server, un rapidísimo motor de transformación XML.

MapForce Server (ediciones Enterprise y Professional)
Puede automatizar tareas de MapForce con Altova MapForce Server, que se puede instalar en sistemas Windows, Linux y macOS. MapForce Server permite ejecutar las transformaciones especificadas en una asignación mediante llamadas a la API (.NET, COM, Java) además desde la línea de comandos del sistema operativo correspondiente.
FlowForce Server (ediciones Enterprise y Professional)
También automatizar tareas de MapForce con Altova FlowForce Server, que se puede instalar en sistemas Windows, Linux y macOS. FlowForce Server permite ejecutar tareas de MapForce Server de forma planificada.
StyleVision (ediciones Enterprise y Professional)
Con StyleVision podrá usar hojas de estilos de StyleVision o diseñar hojas nuevas para obtener vistas previas del resultado de la transformación de datos en formato HTML, RTF, PDF o Word 2007+.

MapForce como complemento
La edición Enterprise y Professional de MapForce se puede instalar como complemento de Visual Studio y Eclipse. Así podrá diseñar asignaciones y acceder a las funciones de MapForce sin salir de su entorno de desarrollo favorito.

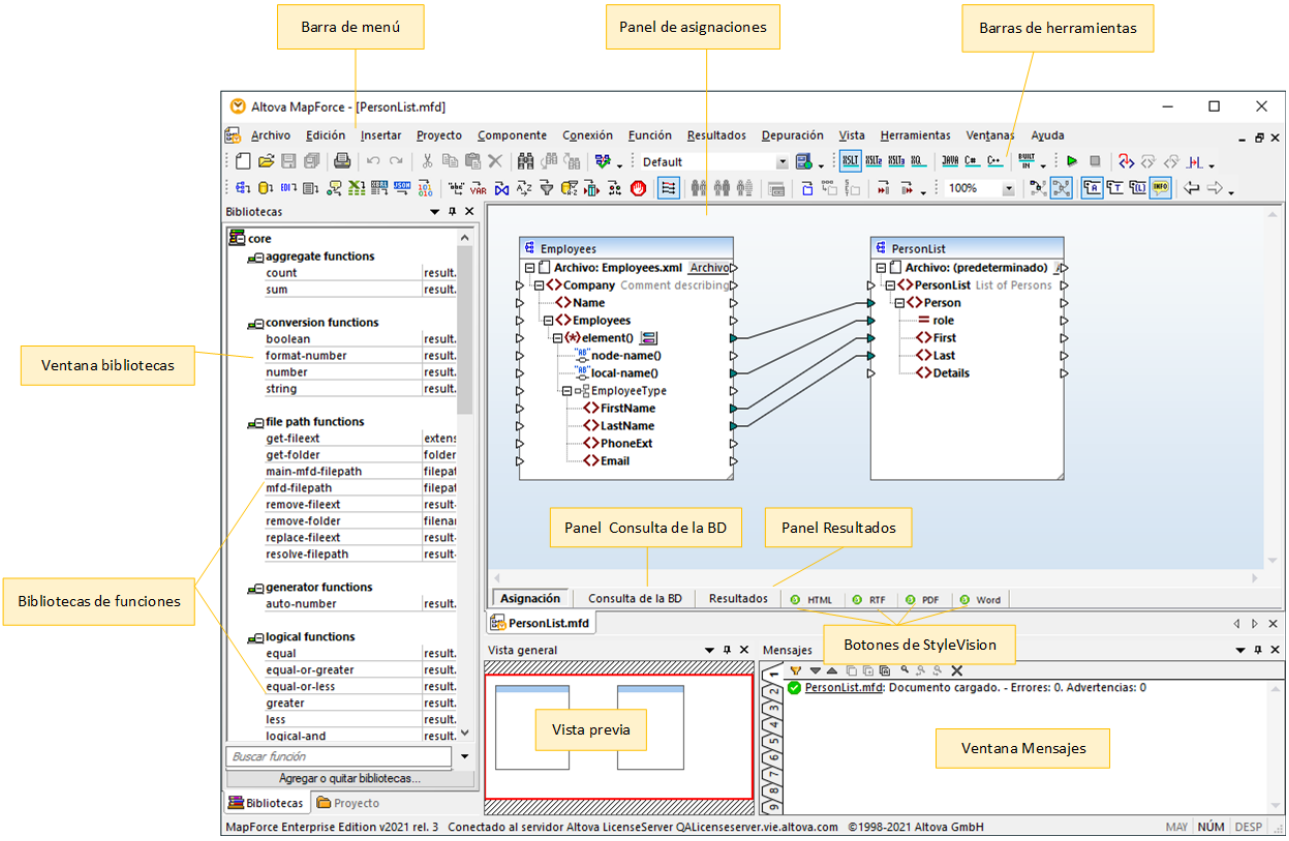
Para obtener más información sobre la automatización de tareas consulte Automatizar asignaciones de datos 946. Para obtener más información sobre MapForce como complemento consulte Complemento de MapForce para Visual Studio 999 y Complemento de MapForce para Eclipse 1002.

© 2016-2022 Altova GmbH

1.3 Interfaz del usuario

La interfaz gráfica del usuario de MapForce está organizada como un entorno de desarrollo integrado. A continuación puede ver los componentes principales de la interfaz. Además, puede cambiar la configuración de la interfaz con el comando de menú **Herramientas | Personalizar**. Todas las ventanas de la interfaz incluyen los botones ▼ 📌 ✕, que sirven para mostrar, ocultar, anclar o acoplar las ventanas. Puede restaurar todas las barras de herramientas a su estado predeterminado con solo ejecutar el comando de menú **Herramientas | Restaurar barras de herramientas y ventanas**.

La imagen siguiente muestra las partes principales de la interfaz gráfica del usuario de MapForce.



Para más información las características y funciones de cada parte consulte el apartado correspondiente.

Apartados de esta sección

Esta sección tiene varias partes:

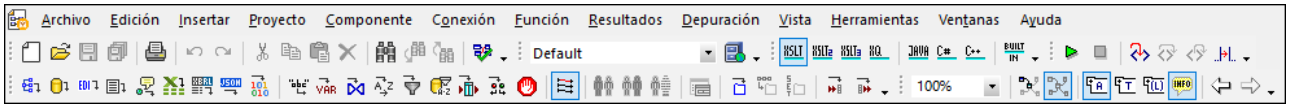
- Barras de herramientas 30
- Ventanas 30
- Ventana Mensajes 34
- Paneles 35

1.3.1 Barras de herramientas

En este apartado explicamos las distintas barras de herramientas.

Barra de menú y barras de herramientas

La **barra de menú** muestra los diferentes menús de la aplicación. Por su parte, las barras de herramientas muestran un grupo de botones que corresponden a comandos de MapForce. Las barras de herramientas se pueden arrastrar a una posición diferente si lo desea. En la imagen siguiente puede ver la **barra de menú** y las barras de herramientas. La interfaz actual depende de la edición de MapForce que esté usando y de cómo configure la aplicación.



Barra de estado de la aplicación

La barra de estado está situada en la parte inferior de la ventana de la interfaz y muestra información general sobre MapForce. Si pasa el puntero del ratón sobre los botones verá aparecer información rápida sobre ellos. Si usa la versión de 64 bits de MapForce, el nombre de la aplicación que aparece en la barra de estado incluye el sufijo x64. La versión de 32 bits no lleva ningún sufijo.

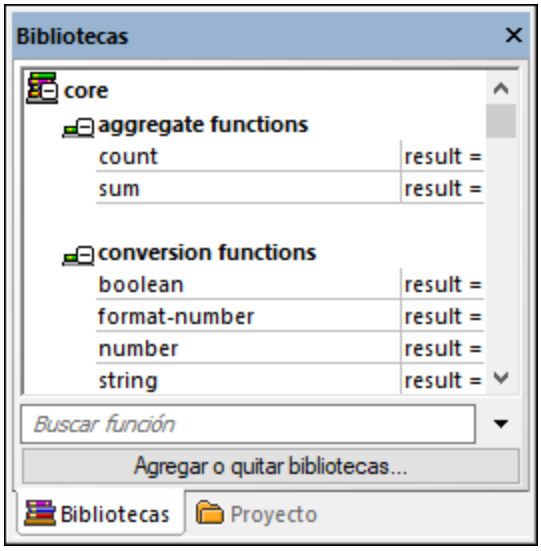
1.3.2 Ventanas

En este apartado explicamos las distintas ventanas.

Ventana Bibliotecas

En la ventana **Bibliotecas** aparecen todas las funciones integradas de MapForce, organizadas por biblioteca. La lista de funciones que aparecen en esta ventana depende del lenguaje de transformación elegido en el menú **Resultados** o en la barra de herramientas de **selección de lenguajes**. Para saber más consulte Seleccionar el lenguaje de transformación 24. Además, en esta ventana aparecen también las funciones definidas por el usuario que se hayan creado y las bibliotecas externas que se hayan importado.

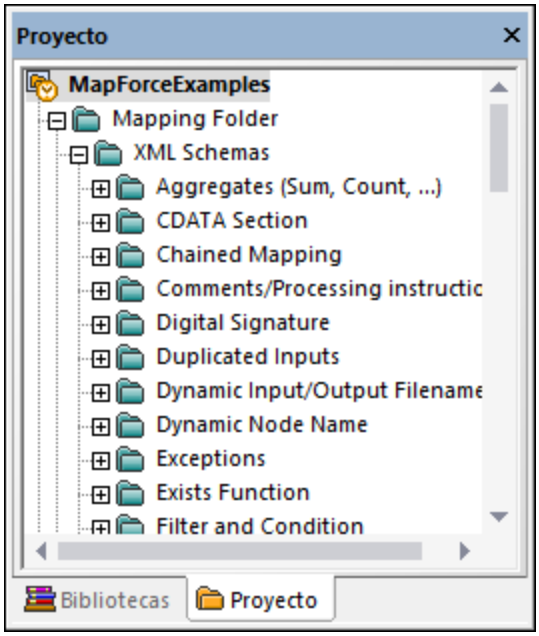
 © 2016-2022 Altova GmbH



Para buscar funciones por nombre o por su descripción, escriba el valor de búsqueda en el cuadro de texto situado en la parte inferior de la ventana **Bibliotecas**. Para buscar todas las instancias de una función dentro de la asignación activa basta con hacer clic con el botón derecho en la función y elegir el comando **Buscar todas las llamadas** en el menú contextual. En la ventana **Bibliotecas** también puede ver el tipo de datos de la función y su descripción. Visite Trabajar con funciones 549 para obtener más información.

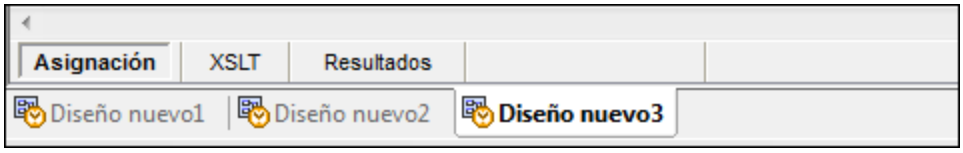
Ventana de proyecto

MapForce ofrece una interfaz compatible con múltiples documentos y permite agrupar las asignaciones en proyectos. La ventana **Proyecto** muestra todos los archivos y todas las carpetas del proyecto. La extensión de los archivos de proyecto de MapForce es **`*.mfp`**. Cuando quiera buscar una asignación dentro de un proyecto, haga clic dentro de la ventana **Proyecto** y pulse **Ctrl+F**. Para obtener más información consulte la sección Trabajar con proyectos de asignación de datos 118.



Ventana(s) de asignación

MapForce usa una interfaz de documentos múltiples. Cada uno de los archivos de asignación que abre en MapForce cuenta con una ventana separada. Esto le permite trabajar con varias ventanas de asignación y colocarlas y cambiar sus tamaños como necesite, todo dentro de la ventana principal de MapForce. También puede colocar todas las ventanas abiertas con las opciones estándar de Windows: En mosaico horizontal, en mosaico vertical y en cascada. Cuando hay varias asignaciones abiertas en MapForce puede cambiar rápidamente de una a otra con las pestañas que hay bajo el panel Asignación (imagen siguiente).



Puede acceder a las opciones de configuración de la ventana con el comando de menú **Ventanas | Ventanas**. El cuadro de diálogo "Ventanas" permite llevar a cabo distintas acciones, como activar, guardar, cerrar o minimizar las ventanas de asignación abiertas. Para seleccionar varias ventanas en el cuadro de diálogo "Ventanas" haga clic en las entradas correspondientes mientras mantiene pulsada la tecla **Ctrl**.

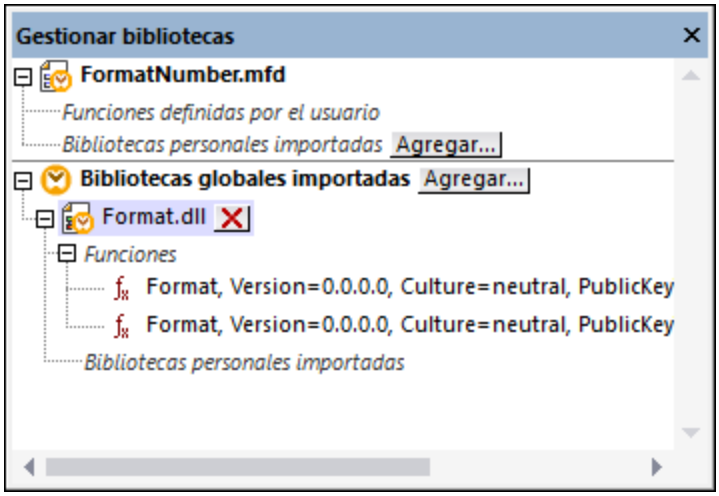
Administrar la ventana Bibliotecas

En esta ventana puede ver y gestionar todas las funciones definidas por el usuario (UDFs), así como las bibliotecas personalizadas que haya importado(incluidos los archivos Java .class compilados y los archivos de ensamblaje .NET DLL) que estén usando las asignaciones abiertas en ese momento.

Por defecto, la ventana Gestionar bibliotecas no está visible. Para que aparezca tiene dos opciones:

- En el menú **Vista** haga clic en **Gestionar bibliotecas**.
- En la parte inferior de la ventana Bibliotecas haga clic en **Agregar o quitar bibliotecas**.

 © 2016-2022 Altova GmbH



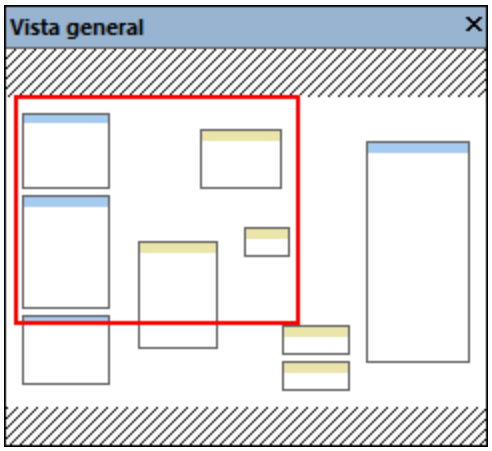
Puede elegir si quiere ver solamente las funciones definidas por el usuario del archivo de asignación activo o si quiere ver las de todos los archivos de asignación abiertos. Para ver las funciones y bibliotecas importadas para todas las asignaciones abiertas actualmente haga clic dentro de la ventana y seleccione **Mostrar documentos abiertos** en el menú contextual.

Si en vez de el nombre del archivo de asignación abierto prefiere ver su ruta de acceso haga clic dentro de la ventana y seleccione **Mostrar todas las rutas de acceso** en el menú contextual.

Para más información consulte Gestionar bibliotecas de funciones 553.

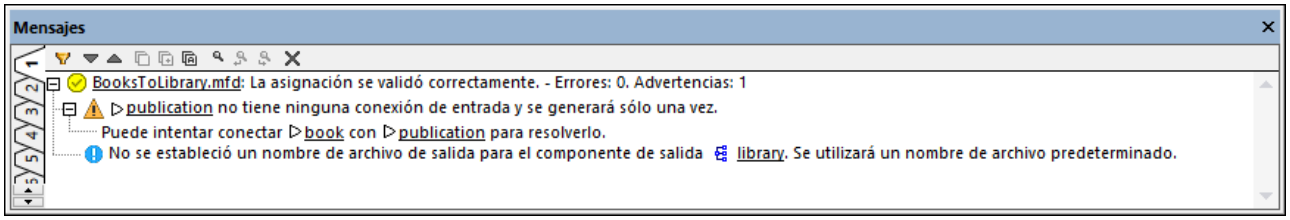
Ventana Vista general

La ventana **Vista general** ofrece una perspectiva general del panel Asignación 35. Desde esta ventana podrá navegar más fácilmente por el área de asignación cuando la asignación sea muy grande. Para navegar por la asignación basta con arrastrar el rectángulo rojo por la ventana.



1.3.3 Ventana Mensajes

La ventana **Mensajes** (*imagen siguiente*) muestra el estado de validación, mensajes, errores y/o advertencias cuando se accede a la vista previa de una asignación o cuando esta se valida [105]. Para ir hasta el componente o estructura que provocó el mensaje, el error o la advertencia haga clic en el texto subrayado en la ventana **Mensajes**.



Iconos de estado de validación

Para validar la asignación MapForce comprueba, entre otras muchas cosas, si faltan conexiones o si hay conexiones incorrectas, si hay tipos de componente incompatibles, etc. El resultado de la validación aparece en la **ventana Mensajes** acompañado de uno de estos iconos de estado:

Icono	Significado
	La validación finalizó correctamente.
	La validación finalizó con advertencias.
	Error de validación.

La ventana **Mensajes** también puede mostrar mensajes de información, advertencia y error:

Icono	Significado
	Mensaje de información. La ejecución de la asignación no se interrumpe.
	Mensaje de advertencia. La ejecución de la asignación no se interrumpe. Pueden aparecer cuando, por ejemplo, no se crean conexiones obligatorias con conectores de entrada. Cuando esto ocurra, se generarán resultados para los componentes que tengan conexiones válidas.
	Mensaje de error. La ejecución de la asignación se interrumpe y no se generan resultados. Tampoco se puede generar una vista previa del código XSLT o XQuery.

Para ir hasta el componente o estructura que provocó el mensaje, el error o la advertencia haga clic en el texto subrayado en la ventana **Mensajes**.

Acciones relacionadas con los mensajes

Estas son las acciones que puede completar desde la ventana **Mensajes**:

 © 2016-2022 Altova GmbH

Icono	Descripción
	Filtra los mensajes según el nivel de gravedad: información, errores y advertencias. Seleccione **Activar todos** para incluir todos los niveles de gravedad (opción predeterminada). Seleccione **Desactivar todos** para quitar todos los niveles de gravedad del filtro. Cuando seleccione esta opción, solamente se mostrarán mensajes relacionados con la ejecución o con el estado de validación.
	Ir a la línea siguiente.
	Ir a la línea anterior.
	Copiar la línea seleccionada en el portapapeles.
	Copiar la línea seleccionada en el portapapeles, incluidas todas sus líneas anidadas.
	Copiar todo el contenido de la ventana Mensajes en el portapapeles.
	Buscar un texto en la ventana Mensajes. Ofrece una opción para buscar **Solo palabras completas** y la opción **Coinc. mayús/min** para refinar la búsqueda.
	Buscar un texto desde la línea seleccionada hasta el final de los mensajes.
	Buscar un texto desde la línea seleccionada hasta el principio de los mensajes.
	Borra el contenido de la ventana Mensajes.

Cuando trabaje con varios archivos de asignación a la vez, puede consultar los mensajes de información, advertencia y error de cada archivo en pestañas diferentes. Para ello, antes de validar la asignación, seleccione la pestaña numerada correspondiente en el lateral izquierdo de la ventana **Mensajes**.

1.3.4 Paneles

En este apartado explicamos los distintos paneles.

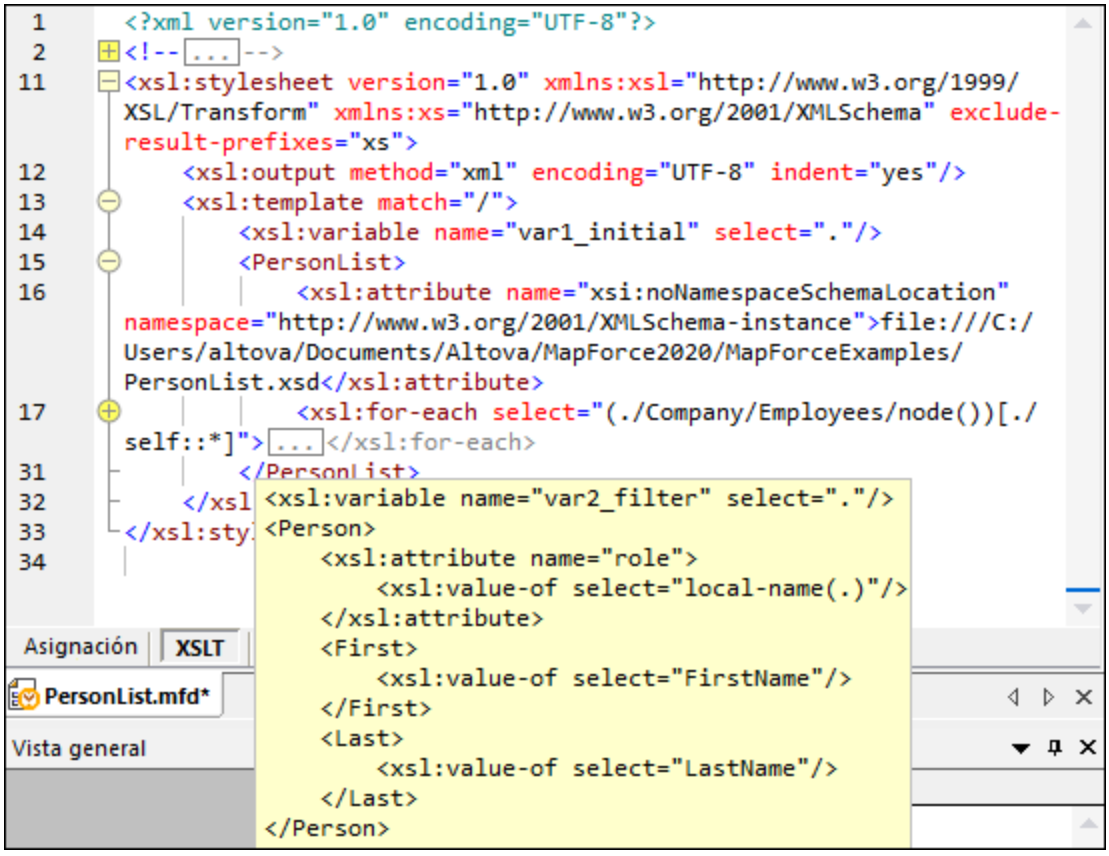
Panel Asignación

El panel **Asignación** es el área de trabajo donde se diseñan las asignaciones 105. En este panel puede añadir componentes de asignación (archivos, esquemas, constantes, variables, etc.) desde el menú **Insertar**. Para más información consulte Agregar componentes a la asignación 77. También puede arrastrar funciones desde la ventana **Bibliotecas** hasta el panel **Asignación**. Para más información consulte Trabajar con funciones 77.

Panel XSLT

El panel **XSLT** muestra el código de transformación XSLT 1.0 que se genera a partir de la asignación. Para cambiar a este panel seleccione el lenguaje de transformación 24 XSLT (XSLT, XSLT 2 o XSLT3) y después haga clic en la pestaña del mismo nombre.

Este panel incluye funciones de numeración de líneas y plegamiento de código. Para expandir y contraer porciones de código basta con hacer clic en lo iconos + y - situados en el lateral izquierdo del panel. Las porciones de código contraído se señalan con puntos suspensivos. Para ver el código contraído debe pasar el cursor del ratón sobre los puntos suspensivos. Al hacerlo se abre un cuadro emergente que muestra una vista previa del código, como se ve en la imagen siguiente. No olvide que si la vista previa no cabe en el cuadro emergente, al final del código aparecen otra vez puntos suspensivos.



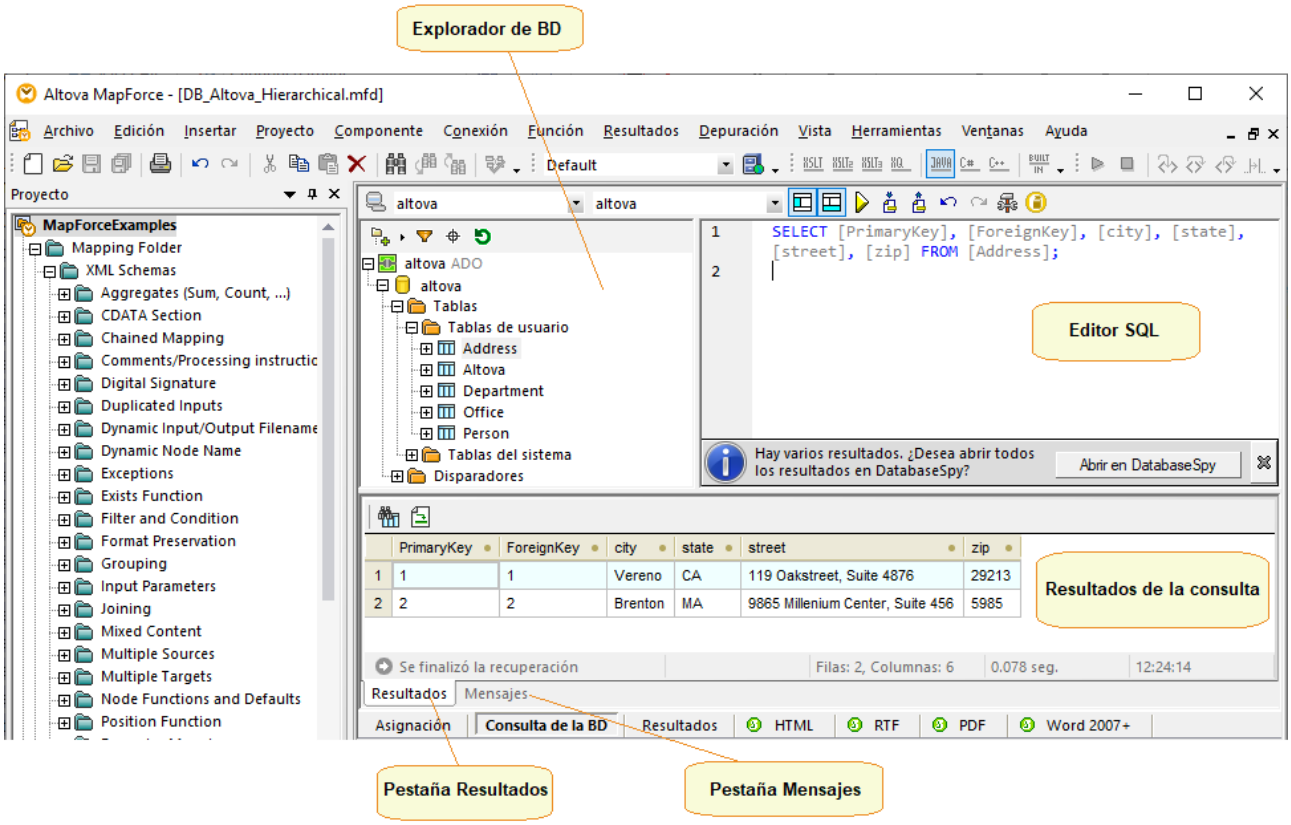
Para configurar las opciones de la vista, como la sangría, los marcadores de final de línea, etc. Haga clic con el botón derecho en el panel y seleccione **Configurar la vista Texto** en el menú contextual. También puede hacer clic en (**Configurar la vista Texto**) en la barra de herramientas.

Panel XQuery (ediciones Enterprise y Professional)

En el panel **XQuery** puede ver el código de transformación generado a partir de la aplicación al hacer clic en el botón **XQuery** . Este panel se habilita si selecciona XQuery como lenguaje de transformación. Este panel también incluye funciones de numeración de líneas y plegamiento de código idénticas a las del panel XSLT (véase más arriba).

Panel Consulta de BD (ediciones Enterprise y Professional)

El **panel de consulta de DB** sirve para consultar las principales bases de datos. En este panel puede trabajar con varias conexiones activas a diferentes bases de datos. Para más información consulte el apartado Examinar y consultar bases de datos 380.



Panel Resultados

El panel **Resultados** muestra el resultado de la transformación de datos. Si la asignación genera varios archivos podrá navegar por ellos de forma secuencial.

```
1   <?xml version="1.0" encoding="UTF-8"?>
2   <PersonList xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="file:///C:/Users/altova/Documents/
    Altova/MapForce2020/MapForceExamples/PersonList.xsd">
3       <Person role="Manager">
4           <First>Vernon</First>
5           <Last>Callaby</Last>
6       </Person>
7       <Person role="Programmer">
8           <First>Frank</First>
9           <Last>Further</Last>
10      </Person>
11      <Person role="Support">
12          <First>Loby</First>
13          <Last>Matise</Last>
14      </Person>
15      <Person role="Support">
16          <First>Susi</First>
17          <Last>Sanna</Last>
18      </Person>
19  </PersonList>
```

Asignación | XSLT | **Resultados**

PersonList.mfd*

Este panel también incluye funciones de numeración de líneas y plegamiento de código idénticas a las del panel XSLT (véase más arriba).

Resultado de StyleVision (ediciones Enterprise y Professional)

Si tiene instalado Altova StyleVision podrá ver los paneles de resultados de StyleVision junto al panel **Resultados**. Estos paneles incluyen botones para obtener una vista previa de los resultados de StyleVision y guardarlos en formato HTML, RTF, PDF y Word 2007+ Esto es posible gracias a los archivos SPS (StyleVision Power Stylesheet) diseñados en StyleVision y asignados a un componente de asignación en MapForce.

© 2016-2022 Altova GmbH

2 Tutoriales

Estos tutoriales le ayudarán a familiarizarse con los procesos de transformación de datos de MapForce. Aprenderá los procesos básicos paso a paso. La complejidad de los tutoriales aumenta gradualmente. Por este motivo recomendamos que los siga en el orden en que los presentamos. Es recomendable tener conocimientos básicos de XML y XML Schema.

Archivos de ejemplo
Los archivos de asignación de datos que aparecen en los tutoriales pueden encontrarse en la carpeta de tutoriales básicos 23. Si no está seguro de cómo podrían afectar ciertos cambios a estos archivos le animamos a que haga copias de seguridad antes de modificarlos.

Tutoriales disponibles

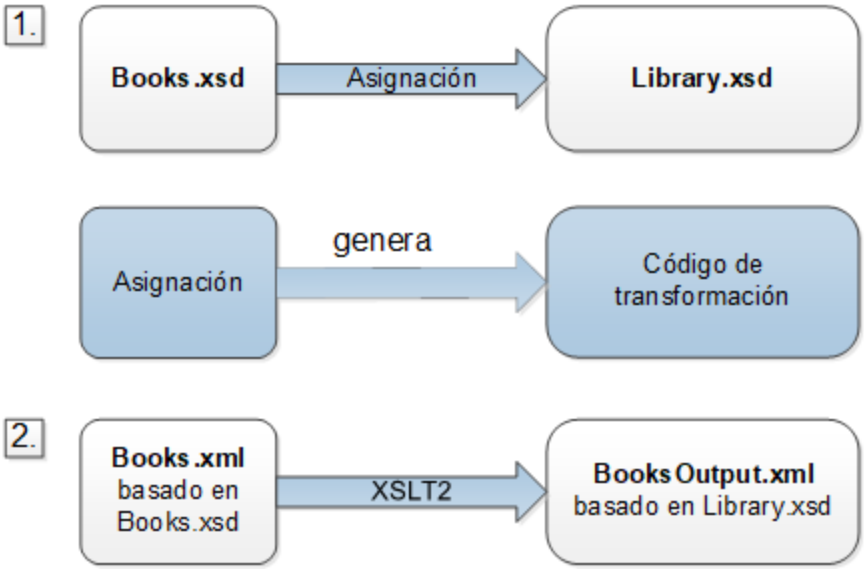
Un archivo de origen a un archivo de destino
En este tutorial 40 aprenderá a usar los mecanismos clave de MapForce para asignar los nodos de un archivo de origen a los de un archivo de destino. El tutorial explica cómo convertir un archivo XML definido por un esquema XML en un archivo XML definido por un esquema XML distinto.
Varios archivos de origen a un solo destino
En este tutorial 50 aprenderá a combinar los datos de distintos archivos XML de origen con los de un único archivo de destino.
Asignaciones encadenadas
En este tutorial 57 crearemos una asignación simple como la del primer tutorial para después filtrar los datos resultantes y pasarlos a un segundo archivo de destino.
Varios archivo de origen a varios archivos de destino
En este tutorial aprenderá a leer datos de distintos archivos de instancia XML de una misma carpeta y escribir esos datos en distintos archivos XML que se generan sobre la marcha.

2.1 De esquema a esquema

En este tutorial se explica cómo crear una asignación básica. El objetivo es tomar los datos de un archivo XML A que tenga asignado un esquema XML A y pasarlos a un archivo XML B que tenga asignado un esquema B. Por tanto, el archivo de destino contendrá los mismos datos que el de origen, pero tendrá la estructura de un esquema distinto. El código de transformación permite ver cómo cambia la estructura del archivo de origen. La asignación se puede ejecutar solo entre las estructuras pero no podremos obtener una vista previa de la transformación en el panel Resultados 37. Por tanto, en este ejemplo usaremos un archivo XML llamado `Books.xml` (*véase el fragmento de código más abajo*). Estos son los pasos que seguiremos a grandes rasgos:

1. Como estamos usando dos estructuras de datos, vamos a crear dos componentes (*origen* y *destino*) en el diseño de la asignación.
2. Después conectaremos los nodos de origen con los nodos de destino correspondientes. Estas conexiones son las que constituyen la asignación y determinan a qué nodo de destino está asignado cada nodo de origen.
3. La transformación de un documento en otro se lleva a cabo usando un lenguaje de transformación, como XSLT, por lo que tendremos que elegir uno.
4. Usamos los motores de transformación integrados de MapForce para transformar el documento XML de origen (`Books.xml`) en un documento de destino. Este documento de destino será un documento XML válido conforme al XSD de destino (`Library.xsd`).
5. Por último, guardamos el archivo XML de salida.

En la imagen siguiente puede ver un modelo abstracto de la transformación de datos de este tutorial:



El modelo abstracto anterior muestra los dos pasos del proceso de asignación. En el primero, la estructura de `Books.xml` se asigna a una estructura nueva llamada `Library.xsd`. Para ello se usa un lenguaje de transformación 24. Los lenguajes que puede usar dependen de la versión de MapForce que use. En este caso hemos escogido XSLT2 como lenguaje de transformación. En el segundo paso, el contenido de `Books.xml`, al que se ha asignado `Books.xsd`, se asigna al archivo de destino (`BooksOutput.xml`) y se basa en un esquema nuevo (`Library.xsd`). El fragmento de código siguiente contiene algunos de los datos de muestra de **`Books.xml`** que se usan como datos de origen.

 © 2016-2022 Altova GmbH

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

Y así es como queremos que queden los datos en el archivo de destino **`BooksOutput.xml`**:

```
<library>
   <last_updated>2015-06-02T16:26:55+02:00</last_updated>
   <publication>
      <id>1</id>
      <author>Mark Twain</author>
      <title>The Adventures of Tom Sawyer</title>
      <genre>Fiction</genre>
      <publish_year>1876</publish_year>
   </publication>
   <publication>
      <id>2</id>
      <author>Franz Kafka</author>
      <title>The Metamorphosis</title>
      <genre>Fiction</genre>
      <publish_year>1912</publish_year>
   </publication>
</library>
```

Algunos nombres de elementos no coinciden en los archivos de origen y de destino. Lo que queremos es rellenar los elementos `<author>`, `<title>`, `<genre>` y `<publish_year>` del archivo de destino con el contenido de los elementos equivalentes del archivo de origen (`<author>`, `<title>`, `<category>`, `<year>`). El atributo `id` del archivo de origen se asigna al elemento `<id>` del archivo de destino. Por último, debemos rellenar el elemento `<last_updated>` del archivo de destino con la fecha y hora en que el archivo se actualizó por última vez.

Para llevar a cabo la transformación de datos siga los pasos que describimos en los apartados siguientes.

2.1.1 Crear y guardar diseños

En este apartado explicamos cómo crear un diseño nuevo, seleccionar un lenguaje de transformación y validar y guardar asignaciones.

Crear un diseño nuevo

Para poder llevar a cabo una transformación primero debe crear un diseño de asignación nuevo, algo que puede hacer de varias maneras:

- En el menú **Archivo** haga clic en el comando **Nuevo**. Después seleccione **Asignación** y haga clic en **Aceptar**.
- En la barra de herramientas haga clic en . Después seleccione **Asignación** y haga clic en **Aceptar**.

Seleccionar el idioma de transformación

Según la edición de MapForce con la que trabaje tendrá disponibles distintos lenguajes de transformación 24. En este tutorial hemos seleccionado XSLT2. Puede seleccionar este lenguaje de transformación de dos maneras diferentes:

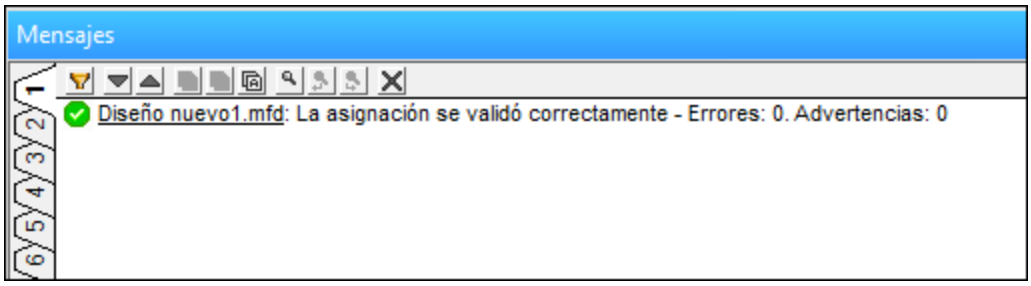
- En la barra de herramientas haga clic en (**XSLT2**).
- En el menú **Resultados** haga clic en **XSLT 2.0**.

Validar y guardar el diseño

La validación es un paso opcional que permite detectar y corregir errores y advertencias potenciales antes de que se ejecute la asignación. Puede validar la asignación en cualquier momento. Para comprobar si una asignación es válidas tiene dos opciones:

- En el menú **Archivo**, haga clic en **Validar asignación**.
- En la barra de herramientas haga clic en (**Validar**).

En la ventana **Mensajes** aparece el resultado de la validación:



Para guardar la validación tiene dos opciones:

- En el menú **Archivo** haga clic en **Guardar**.
- En la barra de herramientas haga clic en (**Guardar**).

Hemos guardado la asignación creada en este tutorial como `Tut1_SchemaToSchema.mfd`.

 © 2016-2022 Altova GmbH

2.1.2 Agregar un componente de origen

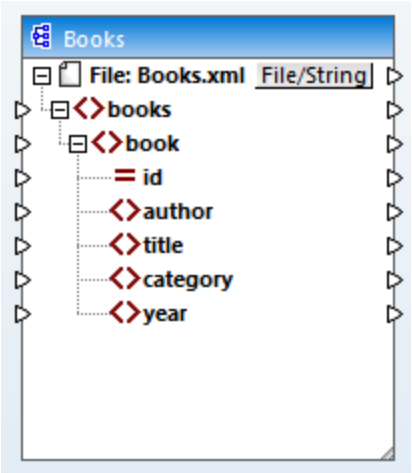
En este punto queremos agregar un archivo XSD, que será la estructura del primer componente, y un archivo XML, que contenga los datos de ese componente. El archivo de origen `Books.xsd` se puede agregar a la asignación de varias maneras:

- En la barra de herramientas haga clic en (Insertar archivo o esquema XML).
- En el menú **Insertar** haga clic en **Arc&hivo o esquema XML**.
- Arrastre `Books.xsd` desde el explorador de Windows hasta el área de asignación.

Al añadir un esquema, MapForce sugiere que se añada un archivo de ejemplo. Haga clic en **Examinar** y navegue hasta `Books.xml`, que se encuentra en la misma carpeta. Ahora el archivo de origen contiene tanto un esquema como contenido. Puede indicar archivos XSD y XML en las propiedades de todos los componentes de origen y de destino. El archivo XSD define la estructura del documento de ese componente. El archivo XML proporciona los datos de ese componente (de origen o de destino) y debe ser válido conforme al esquema se ese componente. Si se crea un componente a partir de un archivo XSD, la aplicación solicita un archivo XML que usar como archivo de datos del componente. Si se crea un componente a partir de un archivo XML, entonces el archivo XSD al que se hace referencia desde el archivo XML se usa para definir la estructura de los datos del componente. Si no existen referencias a ningún archivo XSD, MapForce le preguntará si quiere generar uno para ese componente.

Ver la estructura

Ahora que hemos añadido un archivo de origen al área de asignación, podemos ver su estructura. En MapForce esta estructura se conoce como componente de asignación o simplemente componente 73. Para ampliar elementos en el componente haga clic en el icono. También puede pulsar la tecla **+** del teclado numérico. En la imagen siguiente puede ver el componente de origen:



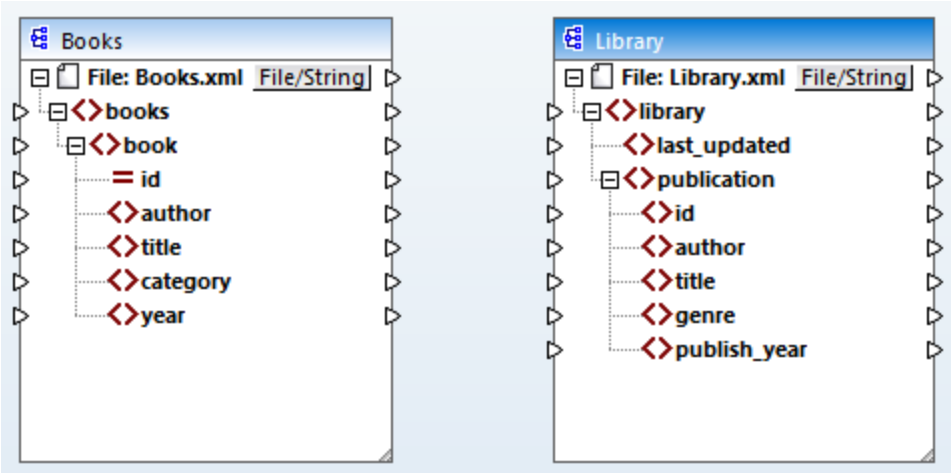
En la barra del título puede verse el nombre del componente, que es `Books`. El nodo de nivel superior representa el nombre del archivo de instancia XML: `Archivo: Books.xml`. Los elementos XML de la estructura están representados por el icono . Los atributos XML están representados por el icono . Los triángulos pequeños que aparecen a ambos lados del componente representan la entrada de datos por la izquierda y la salida de datos por la derecha. En MapForce estos conectores se llaman respectivamente *conectores de entrada* y *conectores de salida*.

Mover y cambiar el tamaño de los componentes

Para mover un componente dentro del panel de asignaciones haga clic en el componente y arrastre el ratón hasta la posición nueva. Haga clic en la esquina inferior derecha de un componente y arrastre el ratón para ajustar su tamaño. También puede hacer doble clic en esa esquina para que MapForce ajuste el tamaño automáticamente.

2.1.3 Agregar componente de destino

El paso siguiente es agregar un componente de destino y configurarlo. Para agregar el archivo de destino Library.xsd a la asignación haga clic en (Insertar archivo o esquema XML). Haga clic en **Omitir** cuando MapForce le pida que indique un archivo de instancia. En este punto la asignación tiene este aspecto:

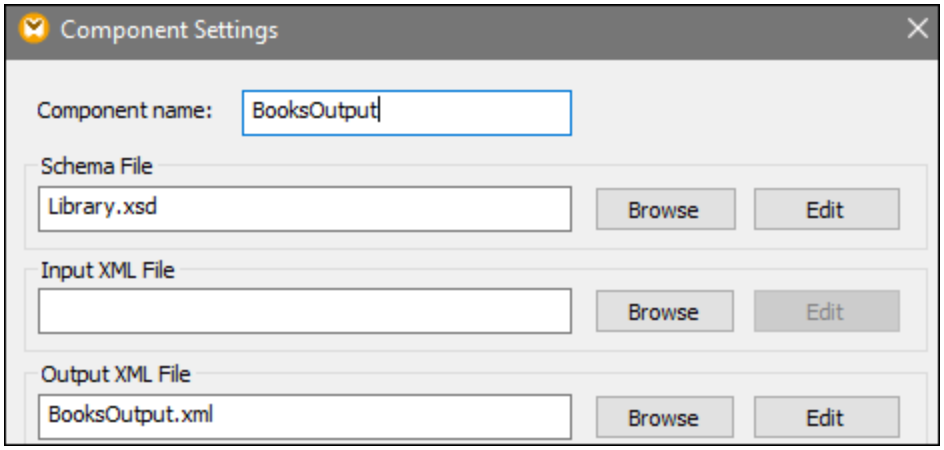


Como ve, al abrir `Library.xsd`, este aparece como archivo XML en el componente. En realidad, lo que hace MapForce es crear una referencia al archivo XML llamada `Library.xml`, pero sin crear ningún archivo XML de verdad. Por tanto, el componente de destino ahora mismo tiene un esquema pero no contenido.

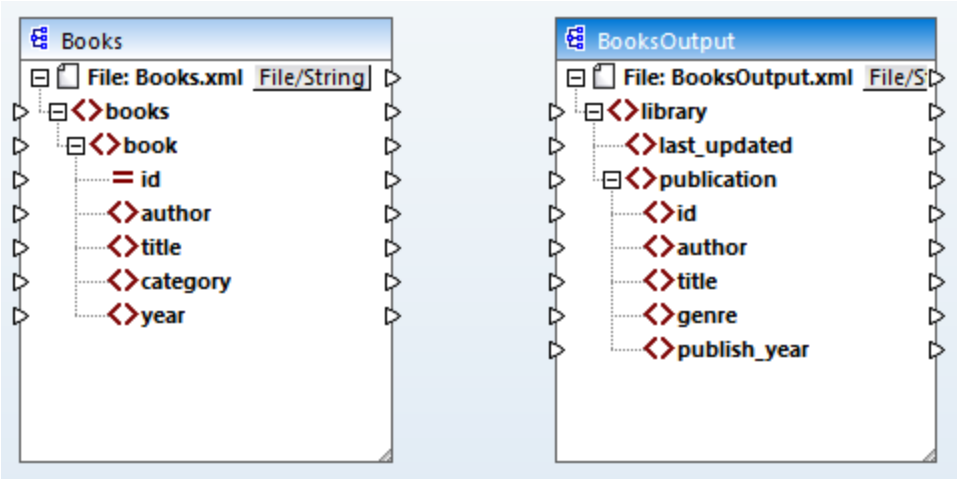
Configurar componentes

Ahora cambiamos el nombre del componente de destino a `BooksOutput.xml`. Así evitamos confundirnos en los tutoriales siguientes, ya que vamos a usar un archivo aparte llamado `Library.xml`, que tiene contenido propio y se basa en el mismo esquema `Library.xsd`. Para cambiarle el nombre al archivo de destino haga doble clic en el encabezado del componente de destino. Se abre el cuadro de diálogo "Configuración del componente" 125 (*imagen siguiente*), donde vamos a cambiar el nombre del archivo de destino:

 © 2016-2022 Altova GmbH



El diseño de la asignación ahora tiene este aspecto:



Nota: el nombre del encabezado solamente hace referencia al nombre del componente y no al nombre del esquema en el que se basa este archivo. Para ver el nombre del esquema abra el cuadro de diálogo "Configuración del componente".

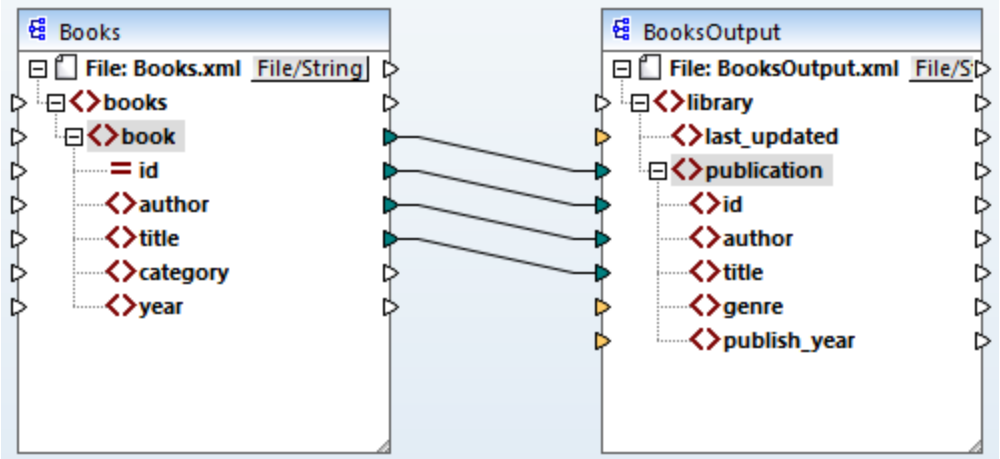
2.1.4 Conectar origen y destino

En este paso vamos a asignar los datos del archivo de origen al archivo de destino. El objetivo es asignar dos tipos de entrada a los nodos de destino: (I) nodos de origen y (ii) datos; ambos se asignan simultáneamente. La mayoría de los datos viene de **`Books.xml`**. En este ejemplo también proporcionaremos información sobre la fecha y hora actuales con la función XPath `current-dateTime` 780.

Conexiones automáticas

Ahora vamos a crear una conexión entre el elemento `<book>` del componente de origen y el elemento `<publication>` del componente de destino. Para ello haga clic en el conector de salida (el pequeño triángulo) a la derecha del elemento `<book>` y arrástrelo hasta el conector de entrada del elemento `<publication>` en el

componente de destino. Con esta acción MapForce puede conectar automáticamente todos los elementos secundarios de `<book>` del archivo de origen a los elementos que tienen el mismo nombre en el archivo de destino. En este ejemplo se han creado cuatro conexiones de forma simultánea (*imagen siguiente*). Esta acción se llama *Conectar automáticamente los secundarios equivalentes* y se puede deshabilitar y también configurar.



Para habilitar o deshabilitar esta función:

- En la barra de herramientas haga clic en (**Conectar automáticamente los secundarios equivalentes**).
- En el menú **Conexión** haga clic en el comando **Conectar automáticamente los secundarios equivalentes**.

Conectar elementos obligatorios

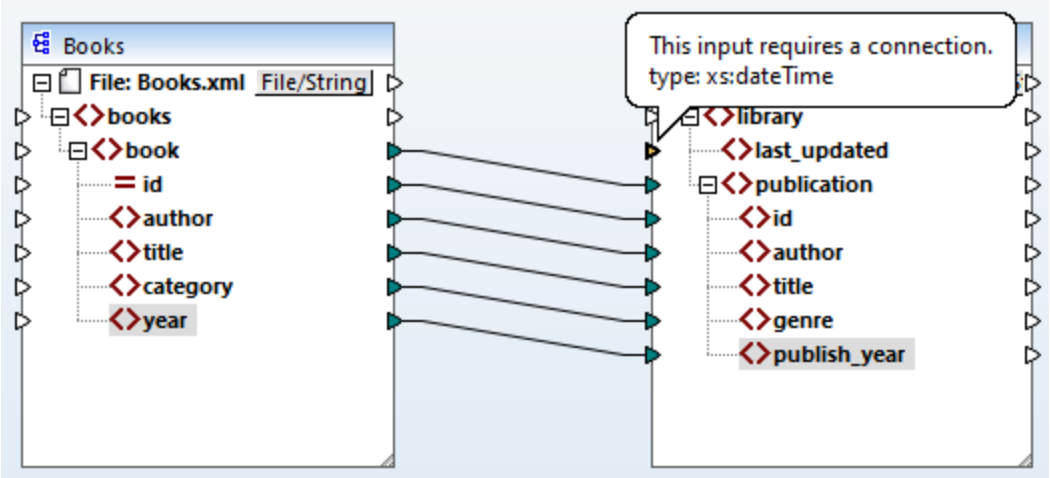
Como puede ver en la imagen anterior, algunos de los conectores aparecen en color naranja, lo que indica que son obligatorios. Estos elementos son obligatorios porque así se han definido en el esquema del archivo. Para asegurarse de que el archivo XML de destino es válido e indicar valores para los elementos obligatorios:

- Conecte el elemento `<category>` del archivo de origen con el elemento `<genre>` del componente de destino.
- Conecte el elemento `<year>` del archivo de origen con el elemento `<publish_year>` del componente de destino.

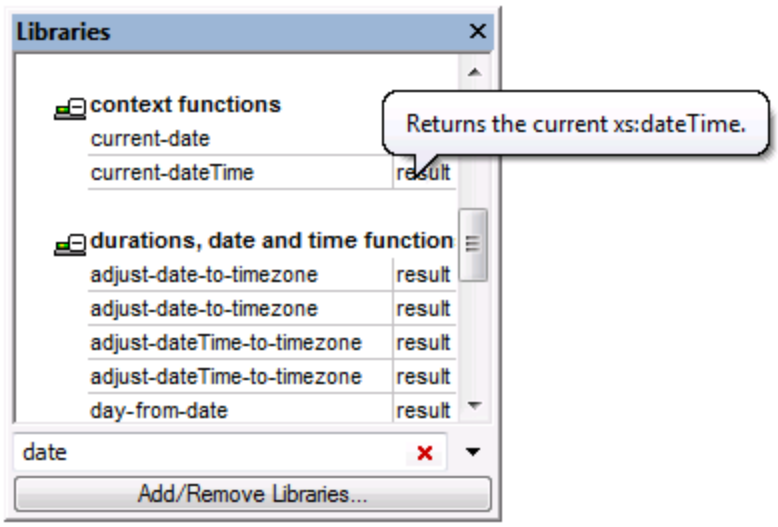
Agregar fecha y hora actuales

Por último debe indicar un valor para el elemento `<last_updated>`. Si pasa el cursor del ratón por encima de este conector de entrada puede ver que es un elemento de tipo `xs:dateTime` (*imagen siguiente*). Para poder ver esa información pulse el botón de la barra de herramientas (**Ver información rápida**). Si pulsa el botón (**Mostrar tipos de datos**) de la barra de herramientas podrá ver el tipo de datos de los elementos en todo momento.

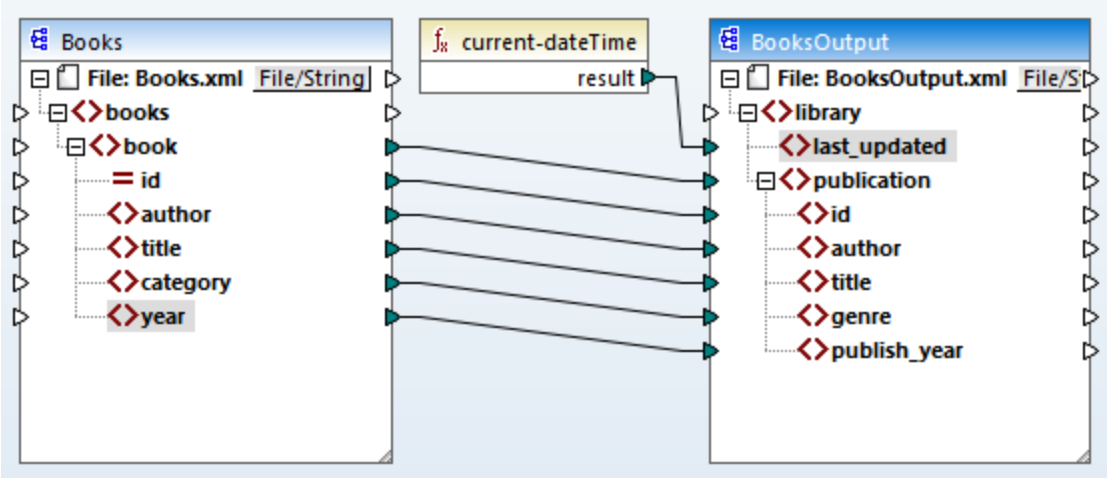
 © 2016-2022 Altova GmbH



Para obtener la fecha y hora actuales puede usar la función XSLT2 **`current-dateTime`**. Para encontrarla empiece a teclearla en la caja de texto que hay en la parte inferior de la ventana Bibliotecas [30] (*imagen siguiente*). También puede hacer doble clic en un área vacía dentro del panel **Asignación** y empezar a teclear `current-date`.



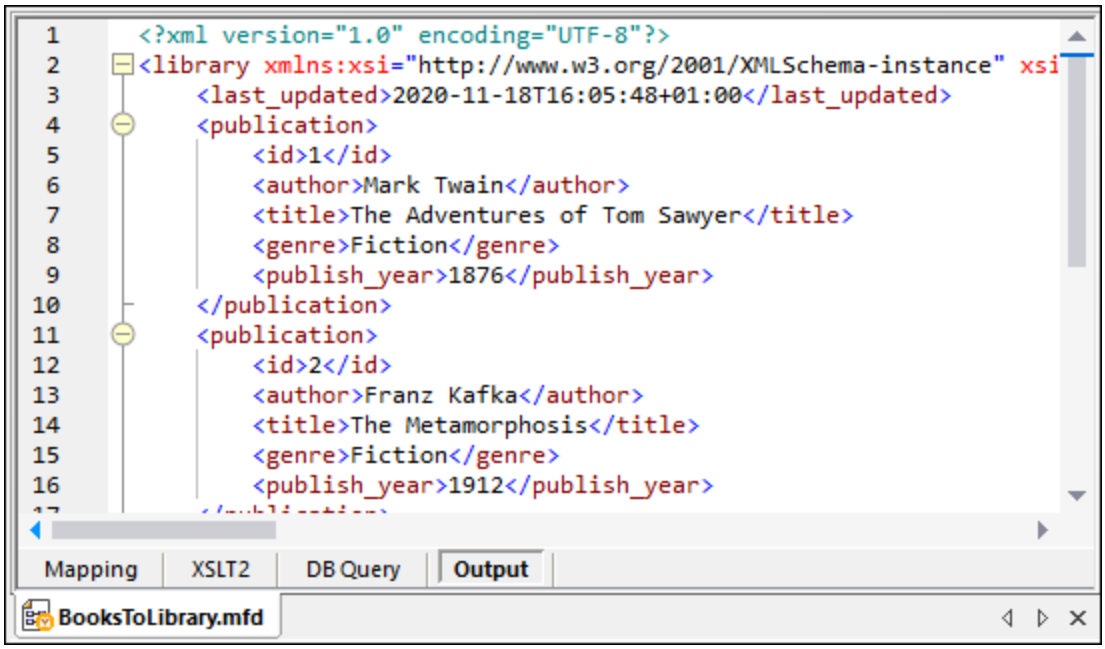
Para agregar la función a la asignación basta con arrastrarla al panel **Asignación** y conectar su conector de salida con el conector de entrada del elemento `<last_updated>` (*imagen siguiente*).



Ahora puede validar y guardar la asignación, como explicamos en Crear y guardar diseños 41.

2.1.5 Vista previa del resultado de la asignación

MapForce usa sus motores integrados para generar el resultado de la asignación y permite visualizar una vista previa del mismo en el panel **Resultados** (*imagen siguiente*).



Por defecto los archivos que aparecen en el panel **Resultados** no se guardan en disco. En su lugar, MapForce crea archivos temporales. Para guardar el resultado abra el panel **Resultados** y seleccione el comando de menú **Resultados | Guardar archivo de salida** o haga clic en (Guardar resultado generado) en la barra de herramientas.

 © 2016-2022 Altova GmbH

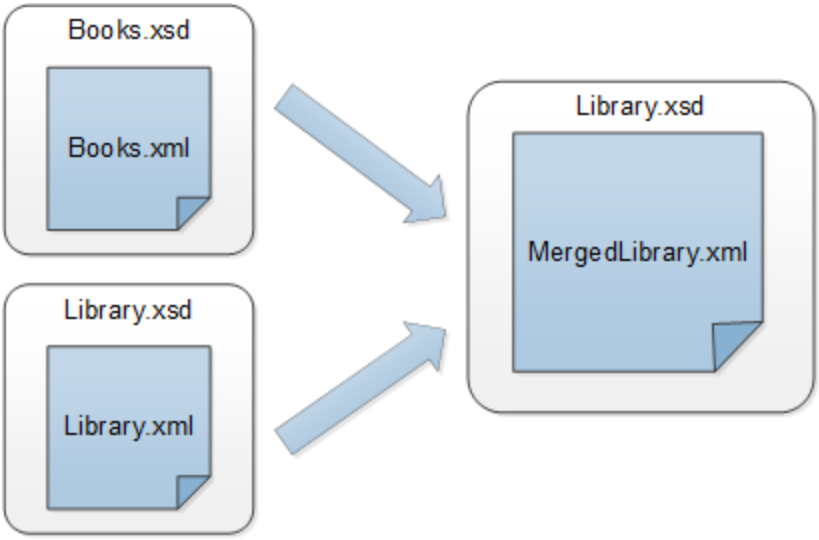
Para que MapForce escriba el resultado directamente en archivos finales en vez de temporales vaya a **Herramientas | Opciones | Generales** y marque la casilla de verificación **Escribir directamente en archivos de salida finales**. No recomendamos activar esta opción si está siguiendo el tutorial porque puede sobrescribir los archivos originales del tutorial sin querer.

También puede acceder a la vista previa del código XSLT generado que lleva a cabo la transformación. Para ello haga clic en el botón **XSLT2** que hay en la parte inferior del panel **Asignación**. Para generar el código XSLT2 y guardarlo en un archivo seleccione el elemento de menú **Archivo | Generar código en | XSLT 2.0**. Cuando la aplicación lo pida, seleccione la carpeta en la que quiere guardar el código generado. Una vez haya generado el código la carpeta de destino contendrá estos dos archivos:

1. Un archivo XSLT de transformación con el mismo nombre que el esquema de destino. Este es el formato del archivo de transformación: `MappingMapTo<NombreArchivoDestino>.xslt`.
2. Un archivo `DoTransform.bat`, que permite ejecutar la transformación XSLT con Altova RaptorXML Server desde la línea de comandos. Para ejecutar este comando necesita tener instalado RaptorXML.

2.2 Varios archivos de origen a un solo destino

En este tutorial aprenderá a combinar los datos de un archivo nuevo llamado `Library.xml` con los datos de `Books.xml`. El resultado será un archivo de destino llamado `MergedLibrary.xml` que contendrá los datos de los dos archivos de destino. El archivo de destino se basa en el esquema `Library.xsd`. Observe que los archivo de origen tienen distintos esquemas. Si los archivos de origen tuvieran el mismo esquema, podría combinar sus datos usando otro método, como explicamos en Varios archivos de origen a varios archivos de destino 64. En la imagen siguiente puede ver un modelo abstracto de la transformación de datos de este tutorial.



El fragmento de código siguiente contiene algunos de los datos de **`Books.xml`**, el primer archivo que usamos como origen de datos.

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

El fragmento de código siguiente muestra un extracto de **`Library.xml`**, , el archivo que usaremos como segundo origen de datos:

```
<library>
   <publication>
      <id>5</id>
      <author>Alexandre Dumas</author>
      <title>The Three Musketeers</title>
      <genre>Fiction</genre>
      <publish_year>1844</publish_year>
   </publication>
</library>
```

 © 2016-2022 Altova GmbH

Este es el aspecto que queremos que tengan los datos combinados en el archivo de destino **MergedLibrary.xml**:

```
<library>
   <publication>
      <id>1</id>
      <author>Mark Twain</author>
      <title>The Adventures of Tom Sawyer</title>
      <genre>Fiction</genre>
      <publish_year>1876</publish_year>
   </publication>
   <publication>
      <id>5</id>
      <author>Alexandre Dumas</author>
      <title>The Three Musketeers</title>
      <genre>Fiction</genre>
      <publish_year>1844</publish_year>
   </publication>
</library>
```

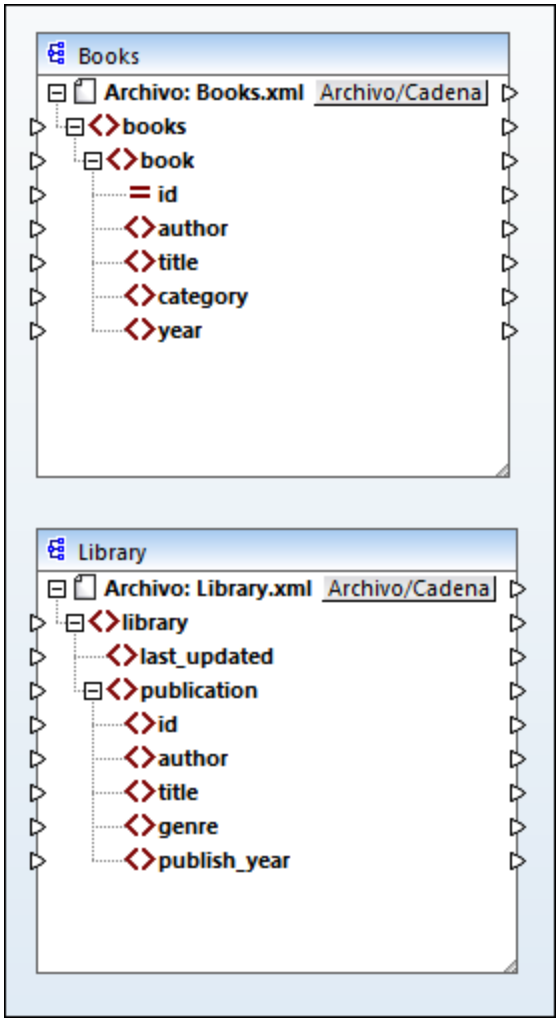
Para llevar a cabo la transformación siga los pasos que describimos en los apartados siguientes.

2.2.1 Preparar archivos de código fuente

En este tutorial empezamos con dos archivos de origen que tienen cada uno un esquema (**Books.xsd** y **Library.xsd**) y datos (**Books.xml** y **Library.xml**). Para preparar los archivos de origen para la asignación:

1. Abra **Books.xsd**.
2. Cuando MapForce le pida que añada un archivo XML de ejemplo, haga clic en **Examinar** y abra **Books.xml**.
3. Abra **Library.xsd**.
4. Cuando MapForce le pida que añada un archivo XML de ejemplo, haga clic en **Examinar** y abra **Library.xml**.
5. Recomendamos que coloque los archivos de origen uno sobre el otro (*imagen siguiente*).

En este punto la asignación tiene este aspecto:

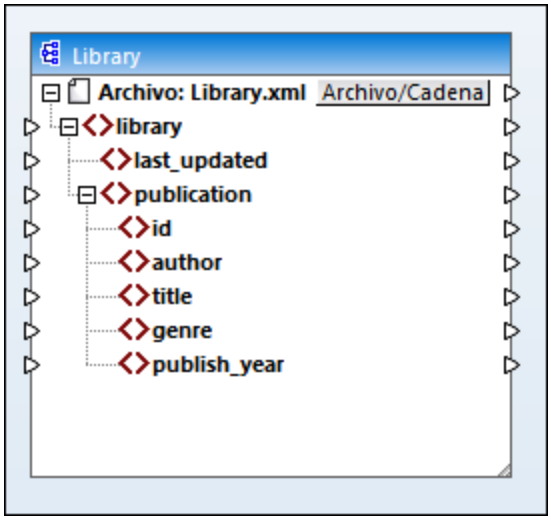


La asignación nueva hace referencia a varios archivos de una misma carpeta, por lo que debe guardarla en la carpeta **`BasicTutorials`**.

2.2.2 Agregar componente de destino

El paso siguiente es agregar un archivo de destino llamado `Library.xsd`. Para ello haga clic en (Insertar archivo o esquema XML). Haga clic en **Omitir** cuando MapForce le pida que indique un archivo de instancia. Ahora haga clic en el encabezado del componente nuevo y arrástrelo a la derecha de los dos componentes de origen. Puede mover los componentes en cualquier momento y dirección. Sin embargo, si coloca los componentes de origen a la izquierda de los de destino la asignación será más fácil de leer y de entender. Esta es la convención que usan en todas las asignaciones de esta documentación, así como los archivos de asignación de ejemplo que se instalan con MapForce.

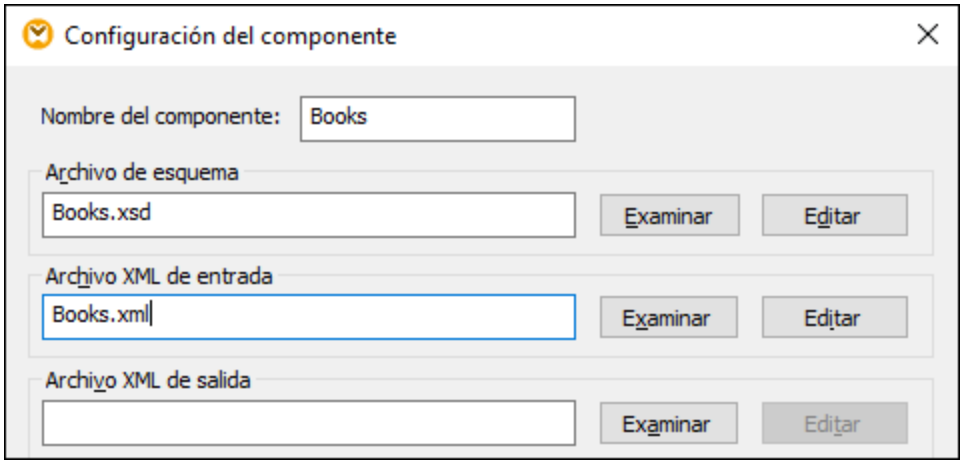
 © 2016-2022 Altova GmbH

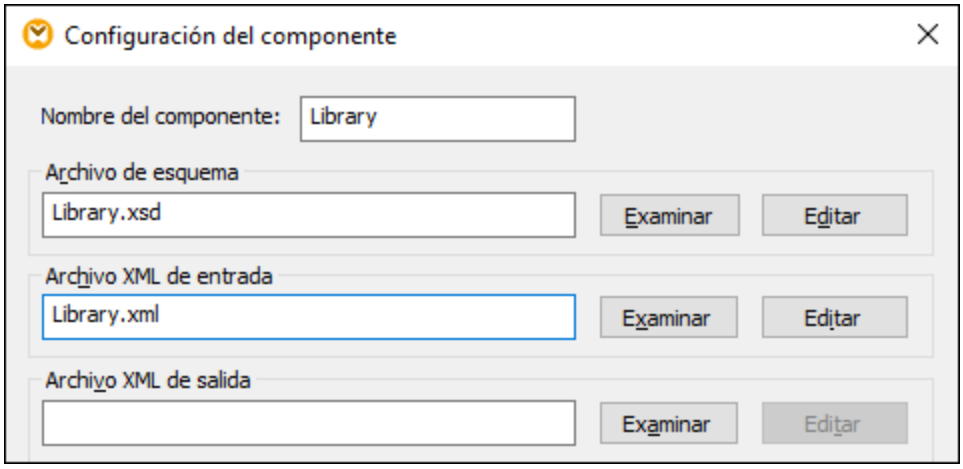


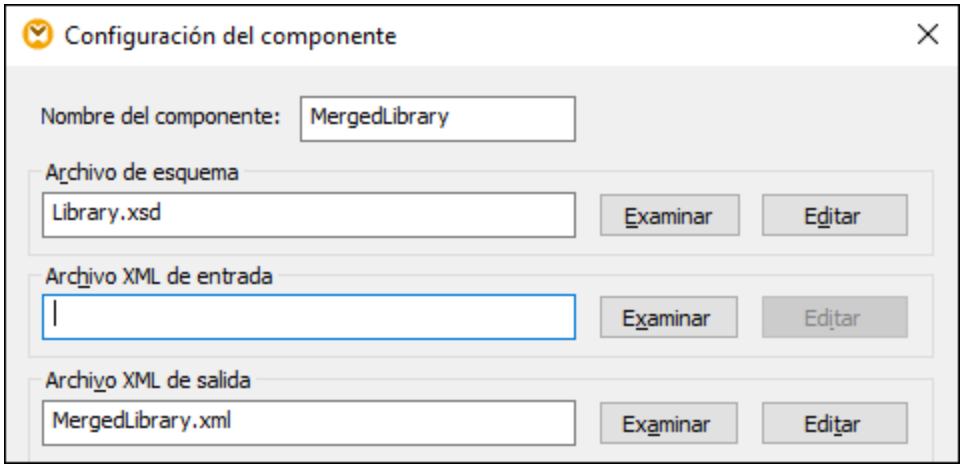
En este punto el archivo de destino tiene estructura pero no datos. Los datos se combinan más adelante de los dos archivos de origen y se asignan al archivo de destino.

2.2.3 Comprobar y configurar los archivos de entrada/salida

En este punto la asignación tiene dos componente de origen (`Books` y `Library`) y un componente de destino (`Library`). Es decir, tenemos dos componentes con el mismo nombre, `Library`. Para evitar malentendidos debemos cambiar el nombre en el cuadro de diálogo "Configuración de componentes XML" 125. Haga doble clic en el título de los componentes. Compruebe y cambie el nombre y los archivos de entrada/salida como se muestra a continuación.







Como puede ver, el primer componente de origen lee datos de `Books.xml`. El segundo lee los datos de `Library.xml`. Por último, el componente de destino escribe los datos del resultado en un archivo llamado `MergedLibrary.xml`.

2.2.4 Conectar varios orígenes a un destino

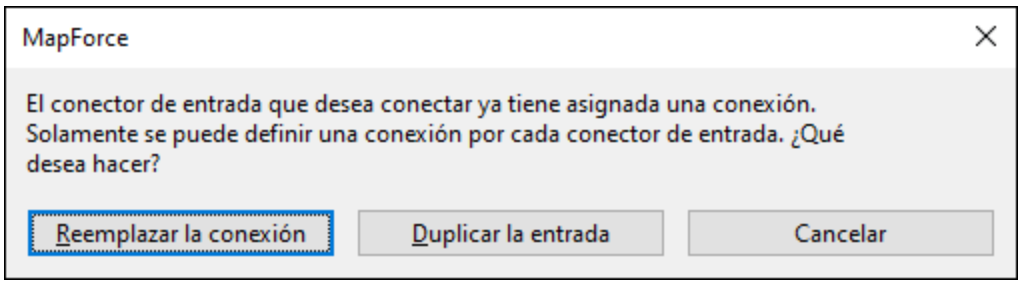
Ahora queremos asignar los datos de dos archivos de origen (**`Books.xml`** y **`Library.xml`**) al archivo de destino (**`MergedLibrary.xml`**). Para ello siga estos pasos:

- Conecte el elemento `<book>` del primer componente de origen con el elemento `<publication>` de **`MergedLibrary.xml`**. Con esta acción MapForce puede conectar automáticamente todos los elementos secundarios de `<book>` del archivo de origen a los elementos que tienen el mismo nombre en el archivo de destino. En este ejemplo las cuatro conexiones se han creado de forma simultánea. Para saber más sobre las conexiones automáticas de elementos secundarios consulte el primer tutorial 45.
- Al conectar el elemento `<book>` con `<publication>`, observe que algunos de los conectores de entrada del componente de destino aparecen en naranja. Eso indica que se trata de conectores

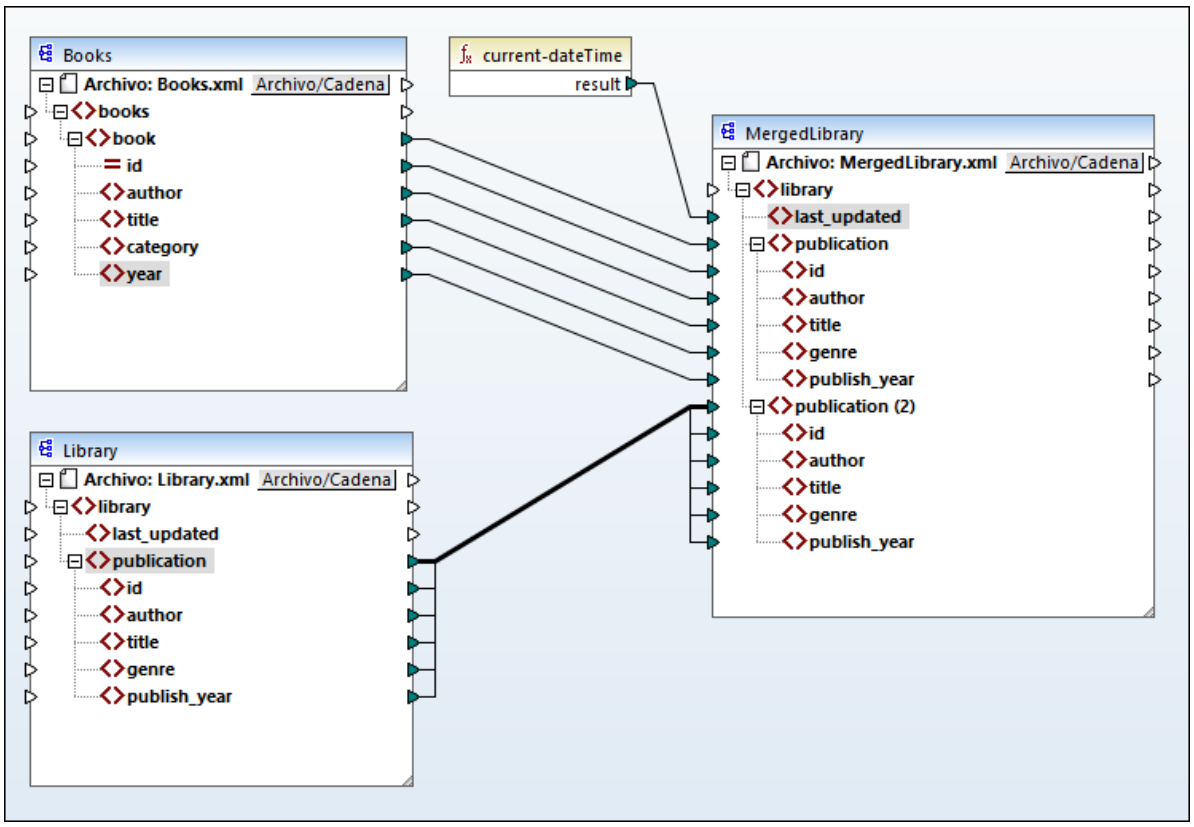
 © 2016-2022 Altova GmbH

obligatorios. Para asegurarse de que el archivo XML de destino es válido, conecte el elemento `<category>` con el elemento `<genre>` y el elemento `<year>` con `<publish_year>`.

- Para indicar el valor del elemento `<last_updated>` debe buscar la función **`current-dateTime`** en la biblioteca XSLT2. Arrastre la función al área de asignación y conecte `result` con el elemento `<last_updated>` de **`MergedLibrary.xml`**.
- Para indicar a MapForce que escriba datos del segundo elemento de entrada, mantenga pulsado en el conector de salida del elemento `<publication>` de **`Library.xml`** y arrástrelo hasta el conector de entrada del elemento **`<publication>`** de la biblioteca del componente de destino. El conector de entrada de destino ya tiene una conexión, por lo que aparece este mensaje:



En este tutorial el objetivo es asignar los datos de dos componentes de origen a uno de destino. Por tanto debemos hacer clic en **Agregar un duplicado de entrada**. Con esta acción se configura el componente de destino de forma que acepte también los datos del componente de origen nuevo. La asignación ahora tiene este aspecto:



En la imagen anterior se observa que el elemento `publications` del componente de destino se ha duplicado. El nodo nuevo `publication(2)` acepta los datos del componente de origen `Library`. Hay que recalcar que

aunque el nombre de este nodo aparezca como `publication(2)` en la asignación, su nombre en el archivo XML de destino será `publication`, que es lo que queremos en este caso.

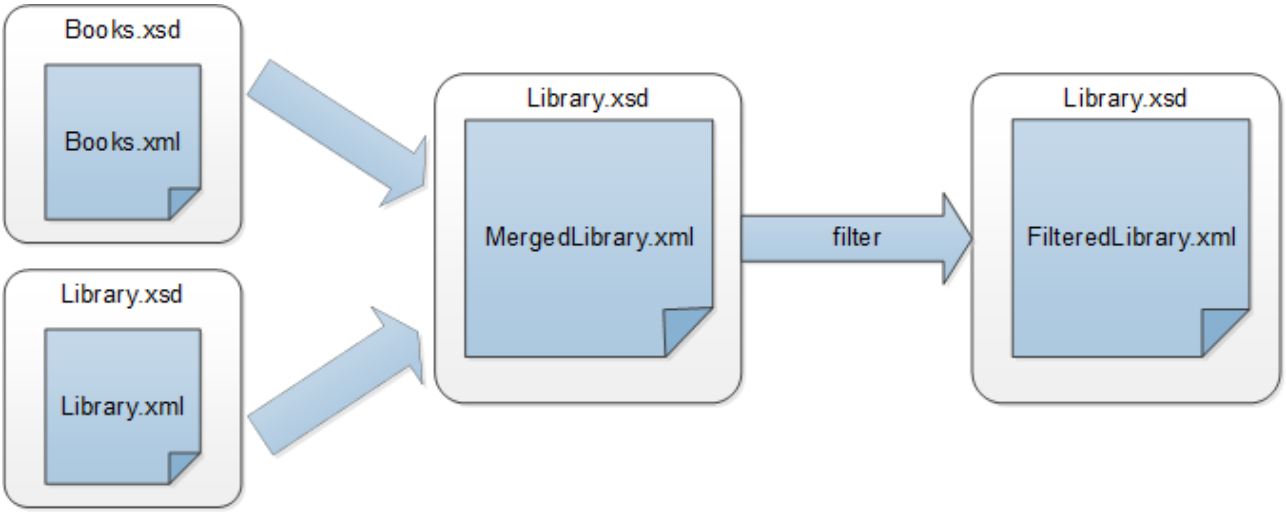
Haga clic en el botón **Resultado** de la parte inferior del panel **Asignación** para ver los resultados de la asignación. Observará que los datos de **`Books.xml`** y de **`Library.xml`** ahora se han combinado en el archivo nuevo **`MergedLibrary.xml`**. Para guardar el resultado abra el panel **Resultados** y seleccione el comando de menú **Resultados | Guardar archivo de salida** o haga clic en (**Guardar resultado generado**) en la barra de herramientas.

Hemos guardado el diseño de asignación de este tutorial como **`Tut2_MultipleToOne.mfd`**. Esta es la asignación que usaremos como punto de partida del tutorial siguiente 57.

 © 2016-2022 Altova GmbH

2.3 Asignaciones encadenadas

El objetivo de este tutorial es filtrar los datos de `MergedLibrary.xml`, que creamos en el tutorial anterior 50 y configuramos para que en el resultado solo aparezca una parte de los datos. Más concretamente, solo aparecen los libros publicados a partir de 1900. En la imagen siguiente puede ver un modelo abstracto de la transformación de datos de este tutorial.

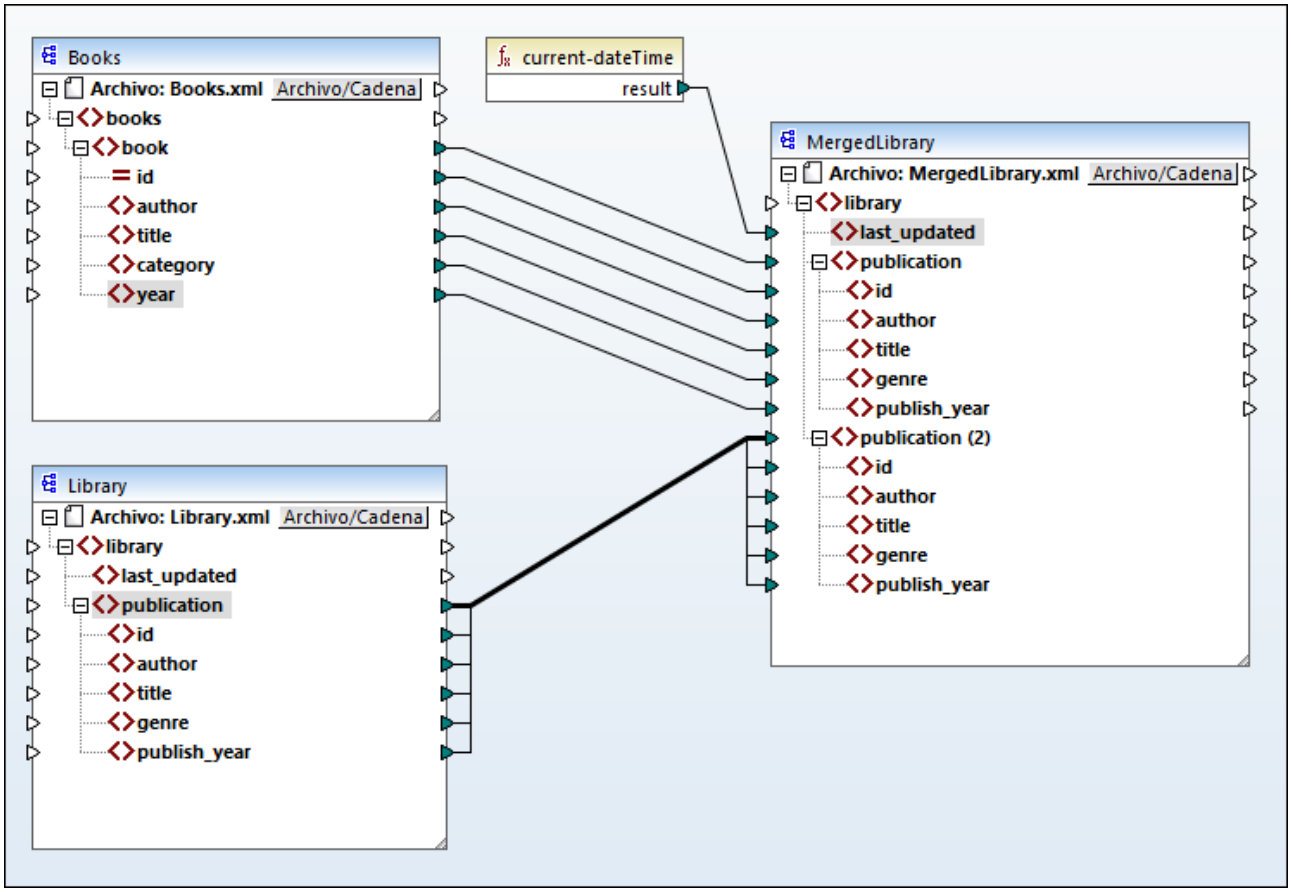


En este diagrama puede ver que primero se combinan los datos de los dos archivos de origen (`Books.xml` y `Library.xml`) en un único archivo de destino, `MergedLibrary.xml`. Después los datos se transforman con una función de filtro y se pasan al siguiente componente, que es `FilteredLibrary.xml`. `FilteredLibrary.xml` se basa en el esquema `Library.xsd`. El componente intermedio actúa como entrada de datos y también como salida. En MapForce esta técnica se conoce como *asignaciones encadenadas* 840.

Para llevar a cabo la asignación de datos siga los pasos que describimos en los apartados siguientes.

2.3.1 Preparar el diseño de la asignación

El punto de partida de este tutorial es la asignación **`Tut2_MultipleToOne.mfd`** (*imagen siguiente*). Esta es la asignación que diseñamos en el tutorial anterior 50.



La asignación nueva hace referencia a varios archivos de una misma carpeta, por lo que debe guardarla en la carpeta **`BasicTutorials`**.

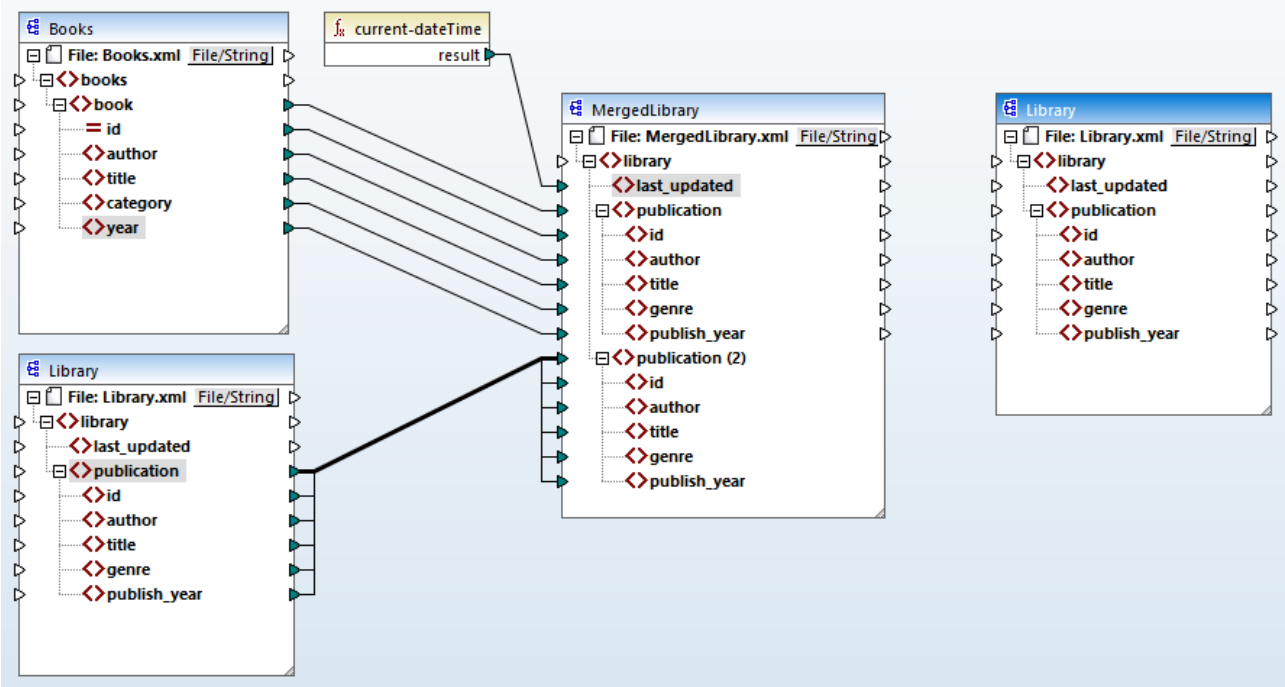
2.3.2 Configurar el segundo archivo de destino

Ahora tenemos que añadir y configurar el segundo archivo de destino (`FilteredLibrary.xml`), que contiene solamente un subconjunto de elementos `<book>` provenientes de `MergedLibrary.xml`.

Agregar el segundo componente de destino

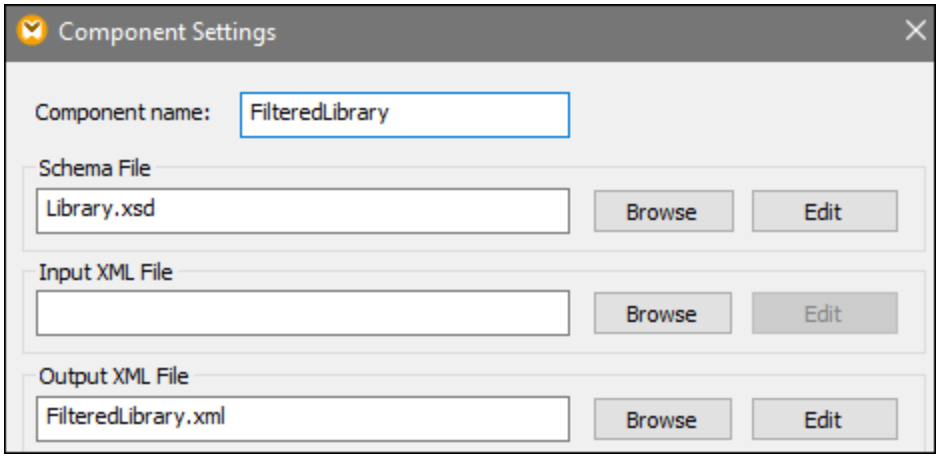
Para agregar el segundo componente de destino haga clic en el botón (**Insertar archivo o esquema XML**) de la barra de herramientas y abra `Library.xsd`. Haga clic en **Omitir** cuando MapForce le pida que indique un archivo de instancia de ejemplo. El segundo componente de destino tiene estructura pero no contenido. Más adelante asignaremos los datos filtrados a este archivo de destino. El diseño de la asignación ahora tiene este aspecto:

 © 2016-2022 Altova GmbH



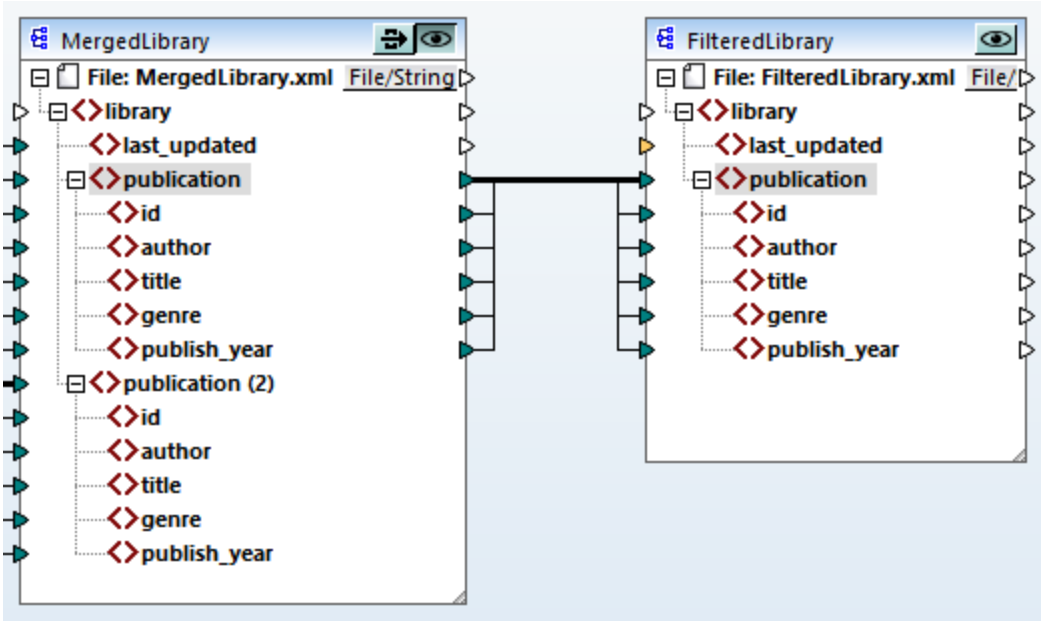
Configurar el segundo componente de destino

Como se ve más arriba, la asignación ahora tiene dos componente de origen (`Books` and `Library`) y dos componente de destino (`MergedLibrary` y `Library`). Para evitar malentendidos vamos a cambiar el nombre del componente que acabamos de añadir, `FilteredLibrary`. Para ello haga doble clic en el encabezado del componente que hay más a la derecha y edite su configuración 125 como sigue:



2.3.3 Conectar componentes de destino

El paso siguiente es asignar el elemento `<publication>` de `MergedLibrary` al elemento `<publication>` de `FilteredLibrary`. Al conectar el conector de salida de `MergedLibrary` con el conector de entrada de `FilteredLibrary` aparece un mensaje como este: "Ha creado varios componentes de destino en el proyecto de asignación. Para acceder a la vista previa de un componente de destino en concreto haga clic en el botón Vista previa de la barra de título de ese componente y después en la pestaña XSLT o Resultados." Haga clic en **Aceptar**. Observe que ahora hay botones nuevos disponibles en la esquina superior derecha de los dos componente de destino: (**Preview**) y (**Pass-through**). Usaremos y explicaremos estos botones en los pasos siguientes.



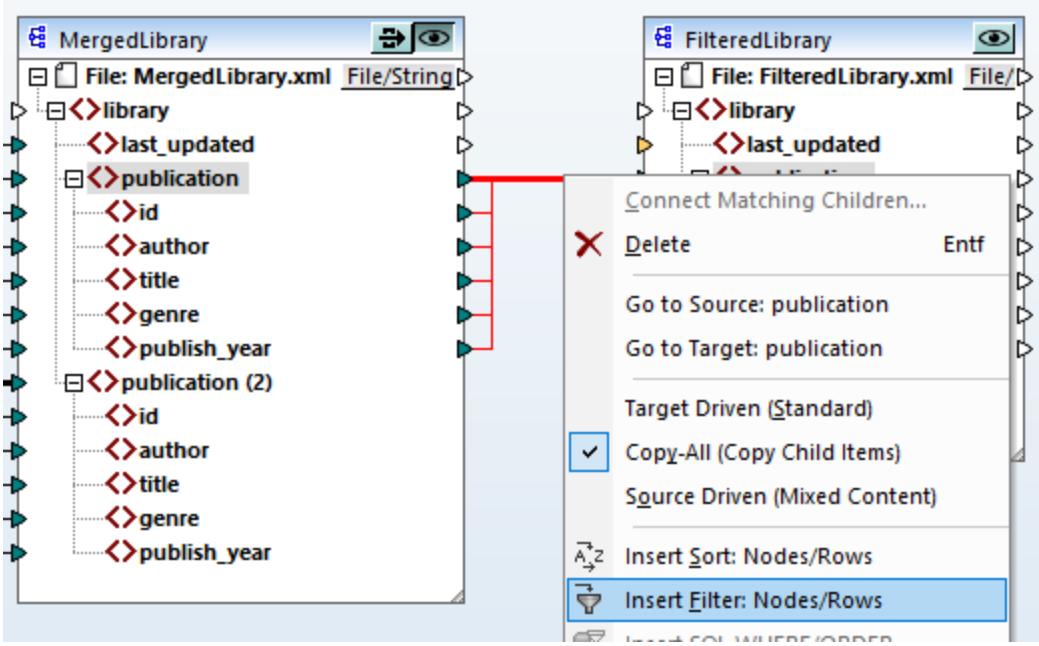
2.3.4 Filtrar datos

En este paso vamos a filtrar los datos de `MergedLibrary` para que solamente pasen al componente `FilteredLibrary` los libros publicados a partir de 1900. Para ello usaremos un componente **Filtro**.

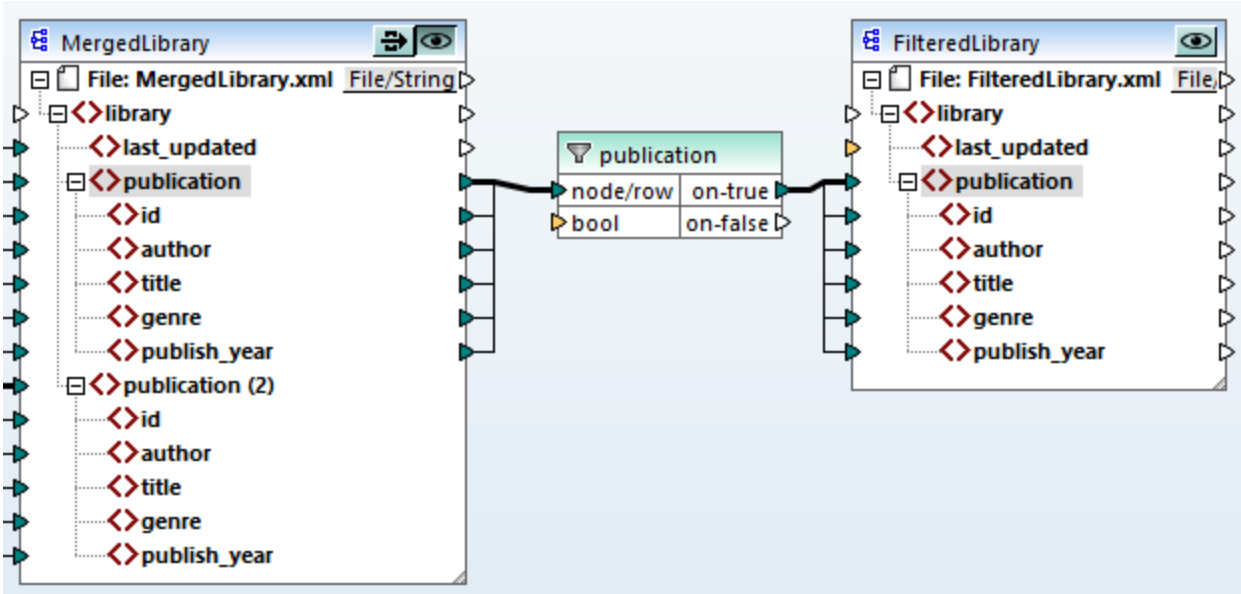
Agregar un filtro

Para agregar un filtro haga clic con el botón derecho en la conexión entre `MergedLibrary` y `FilteredLibrary` y seleccione **Insertar filtro**: **nodos/filas** en el menú contextual.

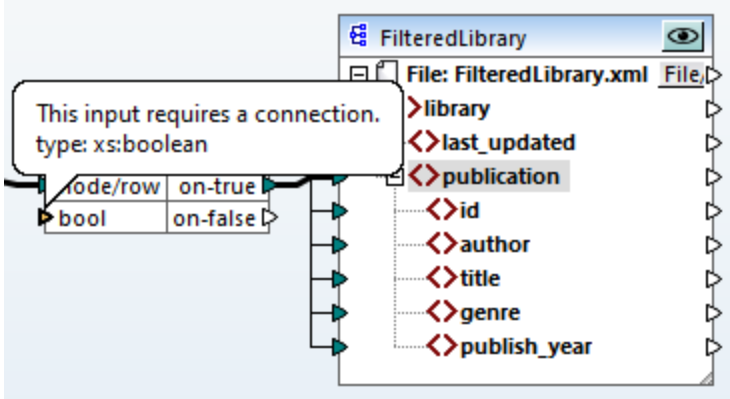
© 2016-2022 Altova GmbH



Ahora hay un componente de filtro en la asignación (*imagen siguiente*).



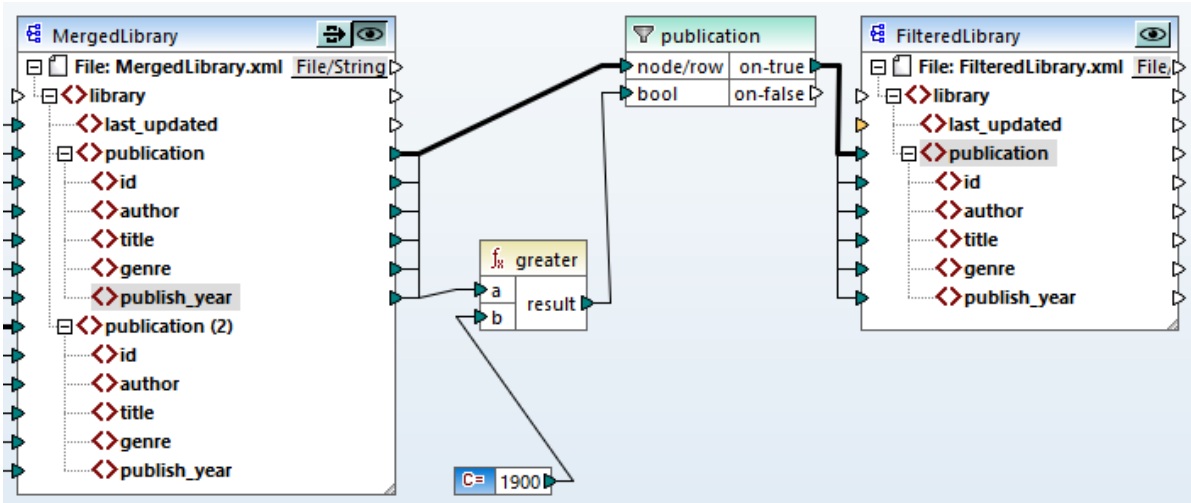
Como se ve más arriba, el conector de entrada `bool` aparece en naranja, lo que significa que es obligatorio. Si pasa el cursor del ratón por encima del conector verá que es un elemento de entrada de tipo `xs:boolean` (*imagen siguiente*). Para ver la información rápida habilítela primero haciendo clic en en la barra de herramientas (**Mostrar información rápida**).



Solo los libros publicados a partir de 1900

El componente de filtro necesita una condición que devuelva `true` o `false`. Si la condición booleana devuelve `true`, los datos de la secuencia `publication` actual se copian en el destino. Si la condición devuelve `false` no se copian datos. En este tutorial la condición filtra todos los libros que se publicaron a partir de 1900. Siga estos pasos para crear la condición:

1. En el menú **Constante** haga clic en **Insertar**. Agregue una constante con el valor *1900*. Seleccione **Número** como tipo.
2. En la ventana **Bibliotecas** busque la función `greater` y arrástrela hasta el panel de asignación.
3. Trace las conexiones de asignación a y desde la función `greater` como se ve en la imagen siguiente. Esto indica a MapForce que copie el elemento de origen actual `<publication>` en el elemento `<publication>` del componente de destino si `publish_year` es mayor que 1900.



 © 2016-2022 Altova GmbH

2.3.5 Vista previa y guardar resultados

Ahora estamos listos para acceder a la vista previa y guardar el resultado de los componentes de destino. Si existen varios componentes de destino en una misma asignación puede elegir cuál quiere conservar haciendo clic en el botón . Si el botón **Vista previa** está activado indica que ese componente en concreto se puede visualizar en la vista previa y el resultado generado aparecerá en el panel **Resultados**. La vista previa no se puede habilitar para más de un componente al mismo tiempo.

Esto quiere decir que si quiere visualizar y guardar el resultado del componente intermedio `MergedLibrary` debe:

1. Hacer clic en en el componente `MergedLibrary`.
2. Haga clic en el botón **Resultado** de la parte inferior del panel **Asignación**.
3. En el menú **Resultados** haga clic en **Guardar.archivo de salida** si quiere guardar los resultados en un archivo.

Si quiere visualizar y guardar el resultado del componente `FilteredLibrary`:

1. Haga clic en en el componente `MergedLibrary`.
2. Haga clic en en el componente `FilteredLibrary`.
3. Haga clic en el botón **Resultado** de la parte inferior del panel **Asignación**.
4. En el menú **Resultados** haga clic en **Guardar.archivo de salida** si quiere guardar los resultados en un archivo.

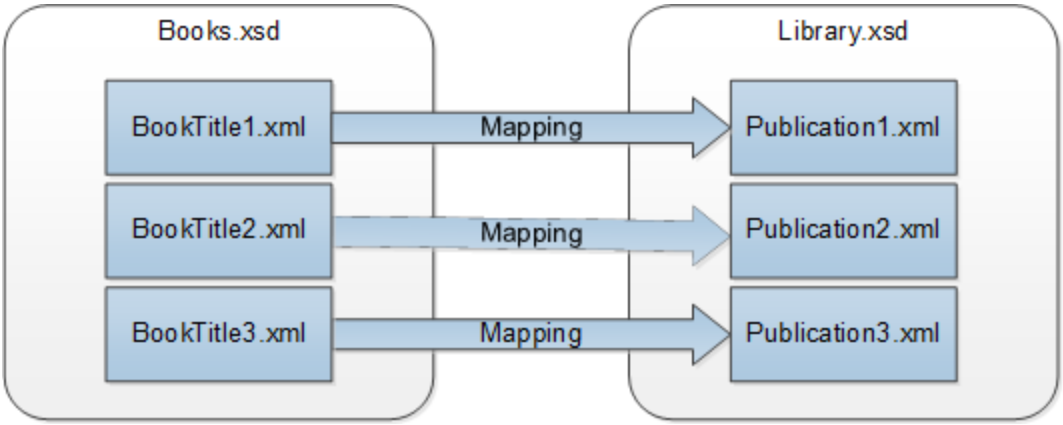
Con este paso terminamos de diseñar la asignación con dos componentes de destino. Para más información sobre componentes de paso consulte Asignaciones encadenadas 840. Hemos guardado el diseño de asignación de este tutorial como **`Tut3_ChainedMapping.mfd`**.

2.4 Varios archivos de origen a varios archivos de destino

En este tutorial aprenderá a asignar los datos de varios archivos de origen a varios archivos de destino en una sola transformación. Para que vea cómo funciona este proceso vamos a crear una asignación con estos objetivos:

1. Leer datos de varios archivos XML de un mismo directorio.
2. Asignar el esquema de cada archivo a un esquema nuevo.
3. Generar por cada archivo XML de origen un archivo XML de destino con el esquema nuevo.
4. Eliminar las declaraciones XML y de espacio de nombres de los archivos generados.

En la imagen siguiente puede ver un modelo abstracto de la transformación de datos de este tutorial:



Punto de partida

Vamos a usar tres archivos XML de origen como ejemplo. Estos archivos son `BookTitle1.xml`, `BookTitle2.xml` y `BookTitle3.xml`. Cada uno de los tres archivos se basa en `Books.xsd` y guarda un solo libro (*véase más abajo*).

`BookTitle1.xml`

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

`BookTitle2.xml`

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

 © 2016-2022 Altova GmbH

```
   <book id="2">
      <author>Franz Kafka</author>
      <title>The Metamorphosis</title>
      <category>Fiction</category>
      <year>1912</year>
   </book>
</books>
```

BookTitle3.xml

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

Pasos siguientes

Ahora seguiremos estos pasos:

1. Antes que nada tenemos que asignar el esquema de origen (`Books.xsd`) al esquema de destino (`Library.xsd`).
2. Después de la transformación la asignación genera tres archivos según el esquema nuevo (*véanse los fragmentos de código siguientes*).
3. También vamos a configurar la asignación para que los nombres de los archivos generados sean `Publication1.xml`, `Publication2.xml` y `Publication3.xml`. Tenga en cuenta que debe eliminar las declaraciones XML y de espacio de nombres.

   Publication1.xml

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

   Publication2.xml

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

```
   </publication>
</library>
```

Publication3.xml

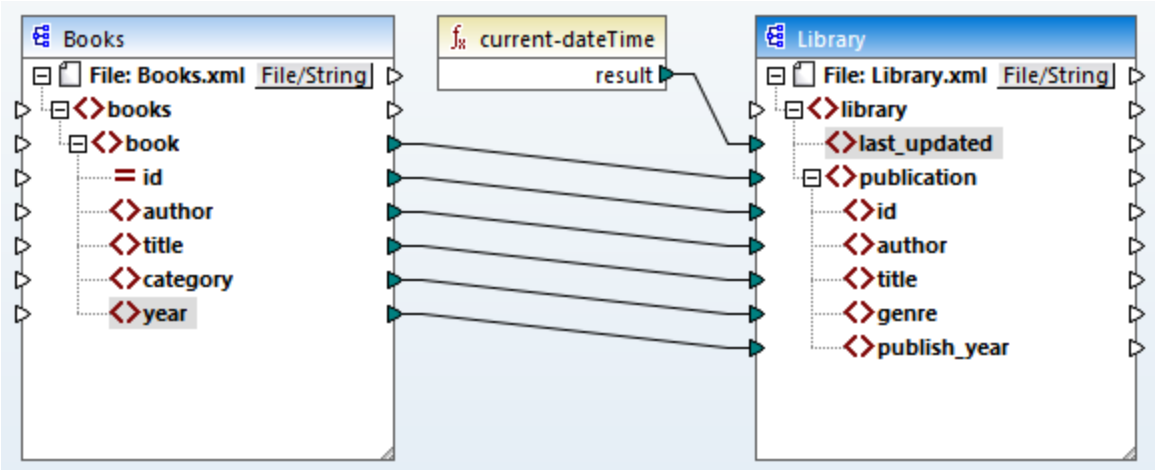
```
<library>
   <publication>
      <id>3</id>
      <author>Herman Melville</author>
      <title>Moby Dick</title>
      <genre>Fiction</genre>
      <publish_year>1851</publish_year>
   </publication>
</library>
```

Para llevar a cabo la transformación de datos siga los pasos que describimos en los apartados siguientes.

2.4.1 Preparar el diseño de la asignación

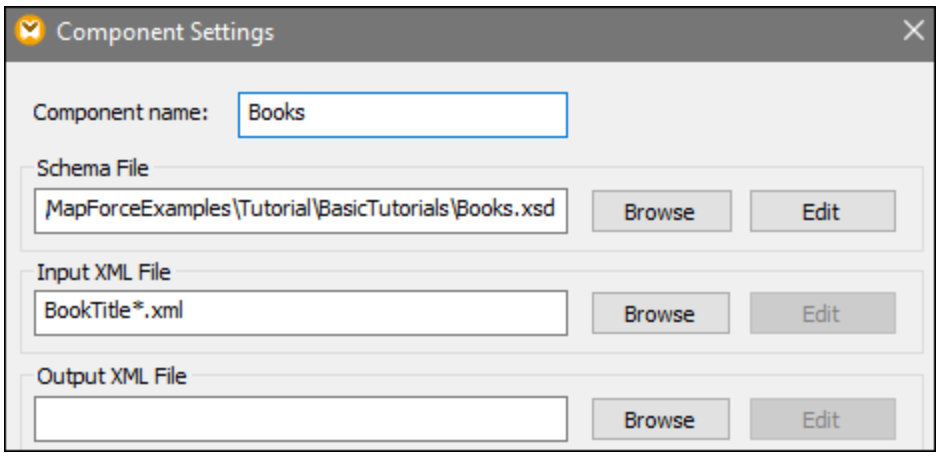
El punto de partida de este tutorial es parecido al de la asignación `BooksToLibrary.mfd` del primer tutorial [40], pero en este caso aún no se ha indicado ningún archivo XML. Nuestro objetivo es asignar una estructura abstracta (`Books.xsd`) a otra (`Library.xsd`). Para reproducir el caso del ejemplo siga estos pasos:

1. Abra `Books.xsd`.
2. Cuando MapForce le pida que indique un archivo XML, haga clic en **Omitir**.
3. Abra `Library.xsd`.
4. Cuando MapForce le pida que indique un archivo XML, haga clic en **Omitir**.
5. Conecte los nodos correspondientes como se ve en la imagen siguiente.
6. Busque la función `current-dateTime` en la ventana **Bibliotecas**.
7. Arrastre la función al área de asignación y conecte `result` con el elemento `last_updated`.



2.4.2 Configurar el componente de entrada

Ahora lo que queremos son tres archivos XML que contengan cada uno un libro y se basen en un mismo esquema, que sería **`Books.xsd`**. Estos archivos XML, **`BookTitle1.xml`**, **`BookTitle2.xml`** y **`BookTitle3.xml`**, son los que usaremos como archivos de entrada. Antes de cambiar la configuración de los componentes guarde la asignación como archivo **`.mfd`** en la carpeta de tutoriales básicos. Para indicar a MapForce que debe procesar varios archivos de instancia XML haga doble clic en el encabezado del componente de origen. Introduzca **`BookTitle*.xml`** como archivo de entrada en el cuadro de diálogo "Configuración del componente" 125. El asterisco (`*`) del nombre del archivo indica a MapForce que use todos los archivos que tengan el prefijo `BookTitle` como archivos de entrada de la asignación. La ruta es relativa, por lo que MapForce buscará todos los archivos `BookTitle` en el mismo directorio en el que está el archivo de asignación, pero también puede usar una ruta absoluta si quiere.

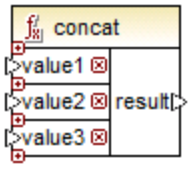


2.4.3 Configurar el resultado, parte 1

El paso siguiente es crear el nombre de archivo de los archivos de salida. Para ello vamos a usar la función **`concat`**, que une todos los valores suministrados como argumentos. Al unir estos valores se crea un nombre de archivo de salida (p. ej. **`Publication1.xml`**). Para generar los nombres de archivo con la función **`concat`** siga los pasos que describimos a continuación.

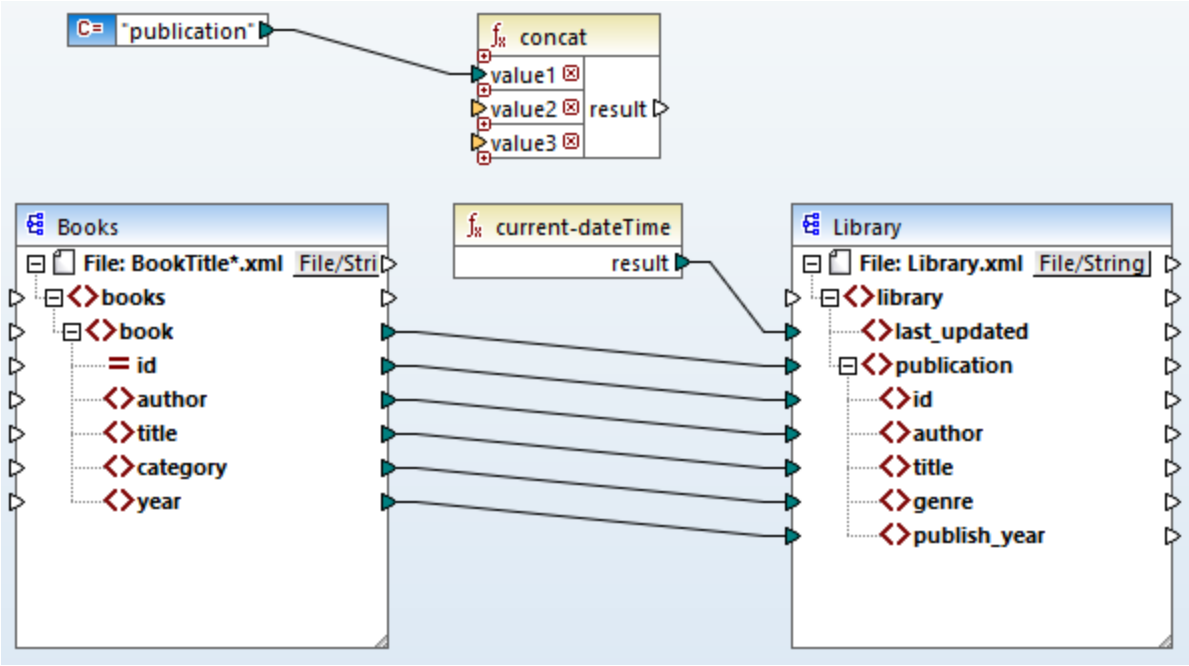
Agregar la función concat

Busque la función **`concat`** en la ventana **Bibliotecas** y arrástrela hasta el área de asignación (*imagen siguiente*). Por defecto esta función tiene dos parámetros cuando se añade a la asignación. En este ejemplo necesitamos tres parámetros. Haga clic en ⊡ (**Agregar parámetro**) dentro del componente de la función y añada un tercer parámetro. Para borrar parámetros use ⊠ (**Eliminar parámetro**).



Insertar una constante

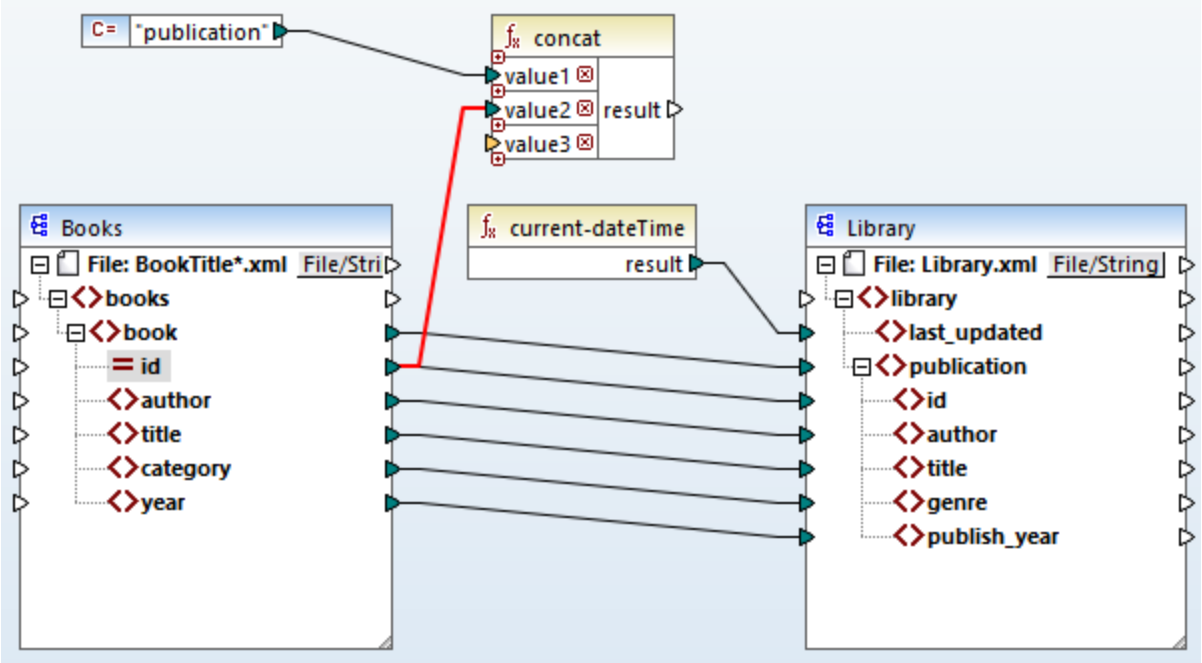
Para agregar una constante haga clic en **Constante** en el menú **Insertar**. Cuando tenga que indicar un valor introduzca `publication` y deje la opción **String** como está. La constante `publication` suministra el valor de cadena de constante `publication`. Conecte la constante con `value1` de la función `concat`, como se ve en la imagen siguiente:



Suministrar el id

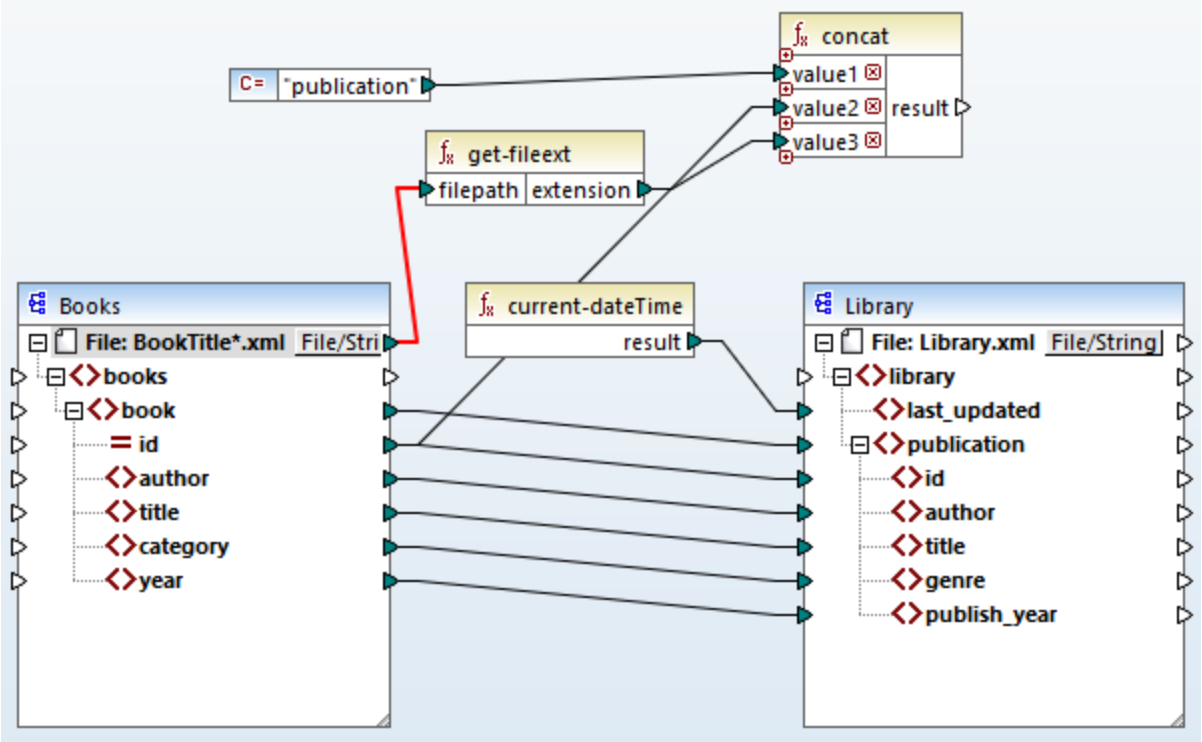
Conecte el atributo `id` del componente de origen con `value2` de la función `concat`. El atributo `id` del archivo XML de origen proporciona un valor de identificador único para cada uno de los archivos. Esta acción evita que todos los archivos se generen con el mismo nombre. La conexión se vuelve roja al hacer clic en ella.

© 2016-2022 Altova GmbH



Extraer la extensión de archivo

Busque la función **`get-fileext`** en la ventana **Bibliotecas** y arrástrela hasta el área de asignación. Cree una conexión desde del nodo superior del componente de origen (**`Archivo: BookTitle.xml`**) hasta el parámetro `filepath` de esta función. Ahora conecte el parámetro `extension` de la función **`get-fileext`** a `value3` de la función **`concat`**. Así extrae solamente la parte de la extensión (en este caso, **`.xml`**) del nombre del archivo de origen y lo pasa al nombre del archivo de salida.



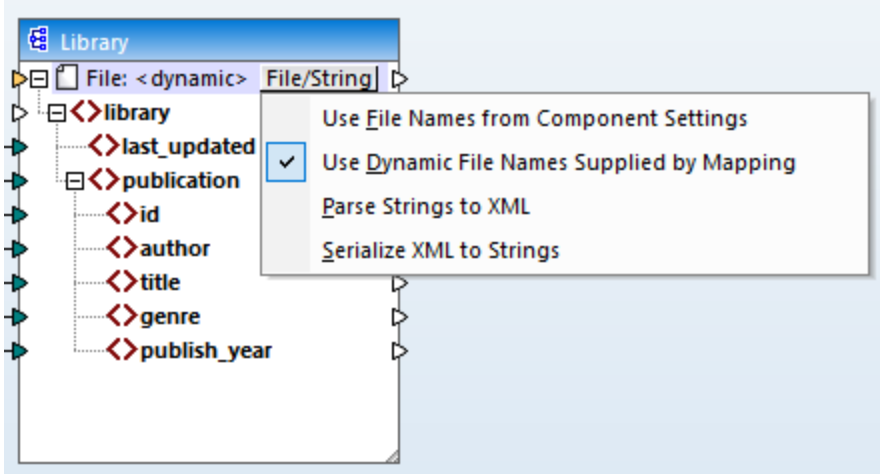
2.4.4 Configurar el resultado, parte 2

Ahora podemos indicarle a MapForce que cree los nombres de archivo al ejecutar la asignación. Para ello usaremos nombres de archivo dinámicos (*véase los puntos siguientes*).

Nombres de archivo dinámicos

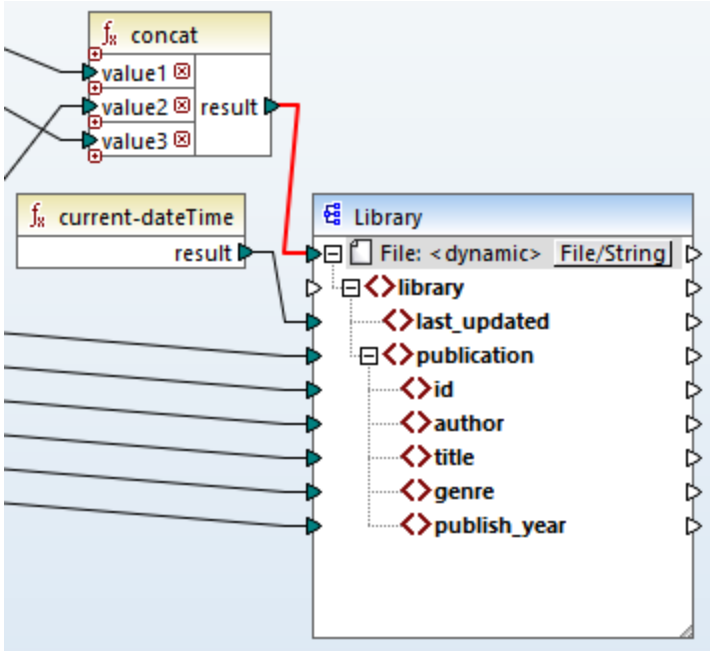
Ahora tenemos que indicar a MapForce que genere los nombres de archivo de forma dinámica, lo que significa que los archivos de salida reciben sus respectivos nombres en base a los argumentos suministrados a la función `concat`. Para ello haga clic en Archivo o Archivo/Cadena en el componente de destino y seleccione **Usar nombres de archivo dinámicos dados por la asignación**.

 © 2016-2022 Altova GmbH



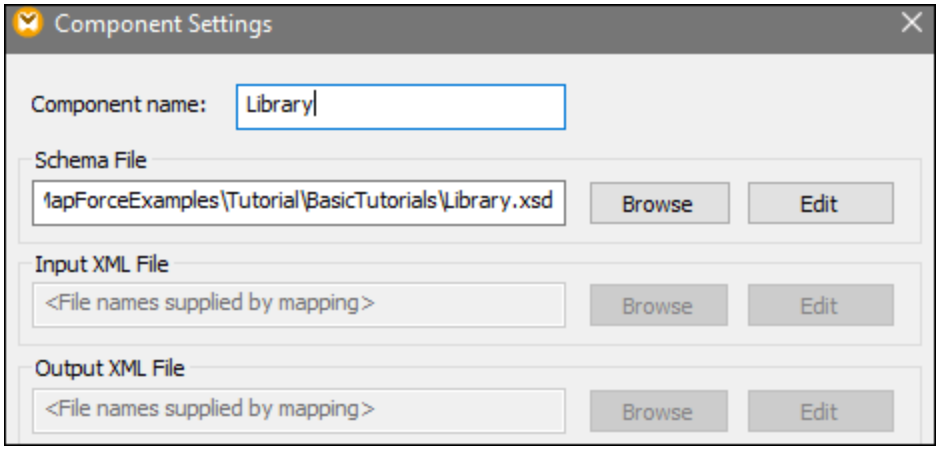
Conectar la función al nodo dinámico

El paso siguiente es conectar el resultado de la función `concat` con el nodo `Archivo: <dynamic>` del componente de destino.

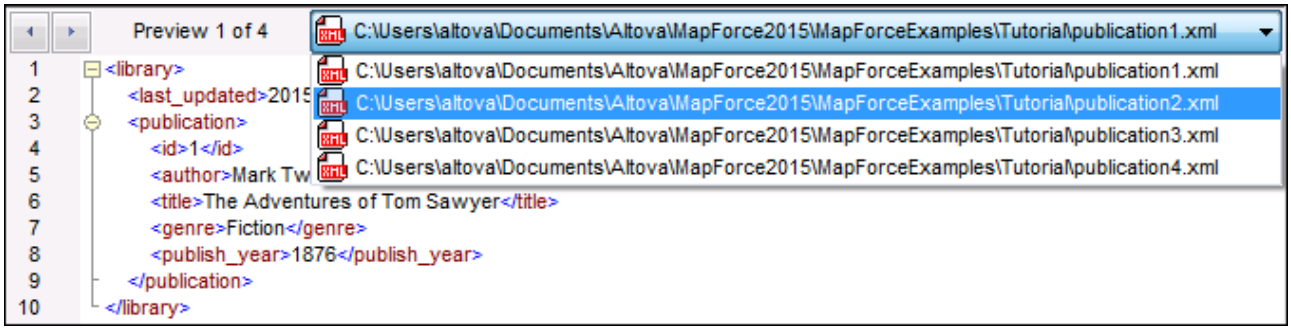


Comprobar la configuración del componente

Si hace doble clic en el encabezado del componente de destino ahora notará que las casillas **Archivo XML de entrada** y **Archivo XML de salida** están deshabilitadas y en lugar de valores aparece <*Nombre de archivo dado por la aplicación*> (*imagen siguiente*). Esto indica que los nombres de archivo de instancia se han obtenido de forma dinámica de la asignación. Por tanto, ya no es necesario definirlos en la configuración del componente.



Ahora puede ejecutar la asignación y ver el resultado, así como los nombres de los archivos generados. Esta asignación genera varios archivos de salida. Puede pasar de un archivo de salida a otro con los botones de izquierda y derecha que hay en la esquina superior izquierda del panel **Resultados** o escogiendo un archivo en la lista desplegable adyacente (*imagen siguiente*).



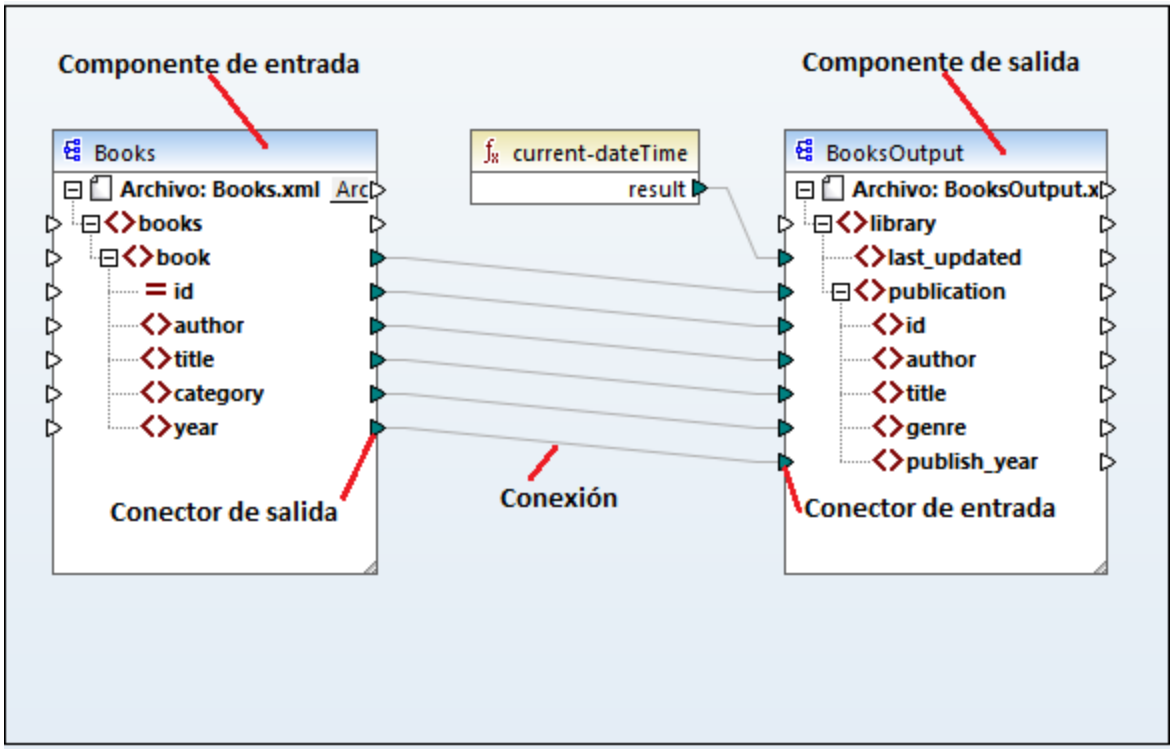
Hemos guardado el diseño de asignación de este tutorial como **`Tut4_MultipleToMultiple.mfd`**.

 © 2016-2022 Altova GmbH

3 Trabajar con asignaciones de datos

Un diseño de asignación de datos (o simplemente asignación de datos) es la representación visual de cómo se deben transformar datos de un formato en otro. Una asignación consiste en *componentes* que se añaden al área de asignación para crear transformaciones de datos. Una asignación válida está formada por uno o varios *componentes de origen* que están conectados a uno o varios *componentes de destino*. La asignación se puede ejecutar y MapForce ofrece una vista previa del resultado. También puede generar código desde MapForce y ejecutar la asignación en una aplicación externa. Por último, puede compilar la asignación en un archivo de ejecución de MapForce y automatizar la ejecución con MapForce Server o FlowForce Server. MapForce guarda las asignaciones en archivos `.mfd`.

En la imagen siguiente puede ver la estructura básica de una asignación de datos:



Asignación nueva

Para crear una asignación nueva haga clic en el botón (**Nuevo**) de la barra de herramientas. También puede hacer clic en el menú **Archivo** y después en **Nuevo**. Después seleccione Asignación y haga clic en **Aceptar**. El paso siguiente es agregar componentes 77 a la asignación y trazar las conexiones 87 correspondientes.

Partes de una asignación

En los apartados siguientes describimos las partes principales de una asignación.

Componente

En MapForce un *componente* es el elemento gráfico que representa visualmente la estructura de los datos o el elemento que determina cómo se deben transformar los datos. Los componentes son piezas fundamentales necesarias para construir una asignación y aparecen en el área de asignación de MapForce en forma de

rectángulo. Los componentes pueden dividirse en dos grandes grupos:

- Componentes de origen y de destino;
- Componentes estructurales 123 y de transformación 476.

Estos dos grupos no se excluyen mutuamente. El primer grupo refleja las relaciones entre componentes, es decir, un componente puede ser de origen para un componente y de destino para otro. MapForce lee los datos de los componentes de origen y escribe daos en los componentes de destino. Al ejecutarse la asignación, el componente de destino da a MapForce instrucciones para generar archivos de salida o generar el resultado como valor de cadena para poder procesarlo en una aplicación externa. A continuación describimos los tipos de componentes del primer grupo:

- El componente de *origen* se encuentra a la izquierda del componente de destino. En este componente es donde MapForce lee los datos.
- El componente de *destino* está a la derecha del de origen. Aquí es donde MapForce escribe los datos.
- Los componentes de *paso a través* son un subtipo de los componentes de origen y destino y actúan tanto como componentes de origen como de destino. Para más información consulte Asignaciones encadenadas 840. Solamente pueden ser componentes de paso a través los componentes estructurales.

En el segundo grupo se puede ver si un componente tiene una estructura de datos o si se usa para transformar los datos asignados desde otro componente.

Para saber más sobre componentes y acciones relacionadas con los componentes consulte Componentes 75
.

Conector
Un conector es un pequeño triángulo situado a la izquierda o derecha de un componente. Los conectores situados a la izquierda del componente ofrecen puntos de entrada para ese componente. Los situados a la derecha ofrecen puntos de salida desde ese componente.

Conexión
Una conexión es la línea que se puede dibujar para unir dos conectores. Al dibujar la línea estamos dando a MapForce instrucciones de transformar los datos de una forma determinada (p. ej. leer datos de un documento XML y escribirlos en otro documento XML).

Apartados de esta sección

En esta sección explicamos las tareas y los conceptos más comunes de MapForce. Hemos dividido la sección en varios apartados:

- Componentes 75
- Conexiones 87
- Tareas generales 105
- Proyectos 118

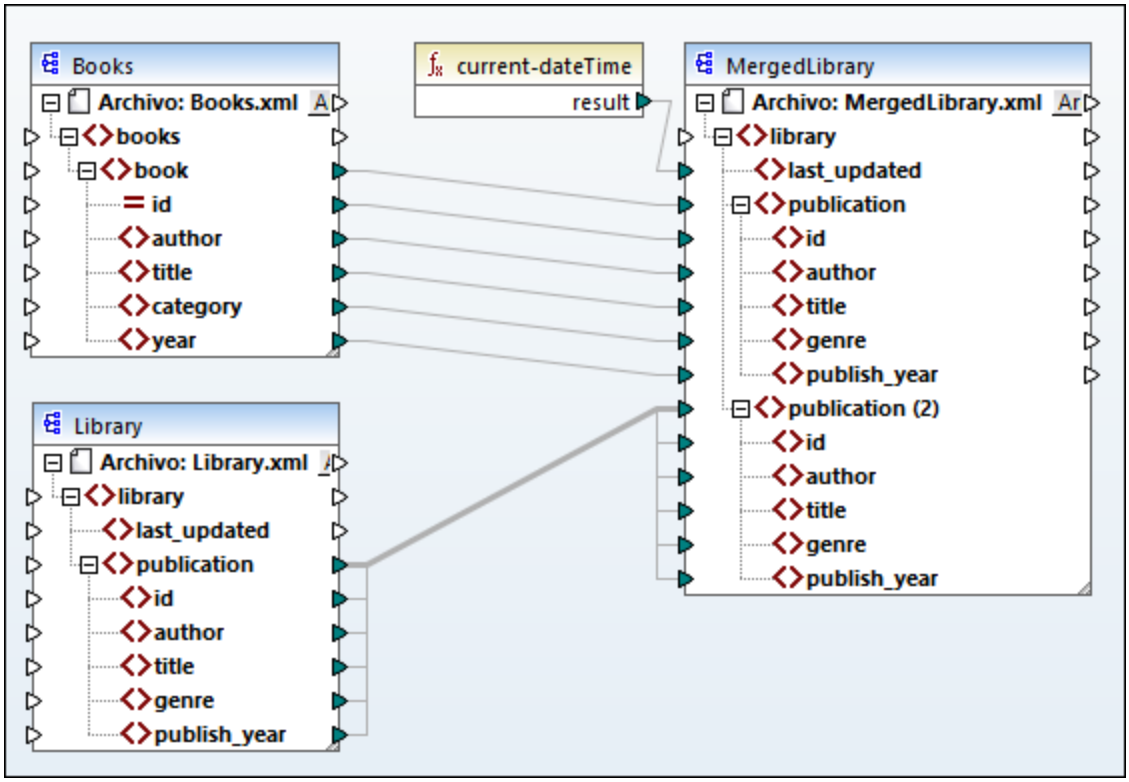
 © 2016-2022 Altova GmbH

3.1 Trabajar con componentes

Los componentes son los elementos centrales de los diseños de asignación de datos en MapForce. En el área de asignación de MapForce los componentes aparecen en forma de rectángulo. En este apartado encontrará un resumen de los componentes estructurales y de transformación (*véase el ejemplo siguiente*). La distinción se basa en si un componente tiene una estructura de datos o si se usa para transformar datos. Consulte la descripción de estos tipos en las subsecciones siguientes. Consulte también Tareas generales 73
.

Ejemplo de los componentes

En la asignación siguiente se pueden ver dos componentes de datos de origen (`Books` y `Library`), un componente de datos de destino (`MergedLibrary`) y un componente de transformación (la función **`current-dateTime`**).



Componentes estructurales

Los componentes estructurales representan la estructura abstracta de unos datos (por ejemplo, un archivo XML). Puede encontrar la lista de componentes estructurales que se pueden usar como orígenes y destinos de datos en Componentes estructurales 123. Los componentes estructurales pueden leer datos de archivos u otras fuentes, escribir datos en archivos u otras fuentes o almacenar datos en alguna fase intermedia del proceso de asignación (por ejemplo, para previsualizar datos). En la tabla siguiente aparece una vista general de los componentes estructurales y los botones de la barra de herramientas correspondientes.

Icono	Descripción
	Componente XML
	Componente de texto (*ediciones Professional y Enterprise*)
	Componente de base de datos (*ediciones Professional y Enterprise* para bases de datos SQL; *ediciones Professional y Enterprise* para bases de datos NoSQL)
	Componente JSON (*edición Enterprise*)
	Componente de Microsoft Excel (*edición Enterprise*)
	Componente WSDL (*edición Enterprise*)
	Componente EDI (*edición Enterprise*)
	Componente XBRL (*edición Enterprise*)
	Protocol Buffers (*edición Enterprise*)

Componentes de transformación

Con los componentes de transformación puede transformar datos 549, almacenar resultados intermedios de una asignación 492 para procesarlos más adelante, reemplazar un valor por otro 533, ordenar 505, agrupar 511, combinar 517 y filtrar 543 datos. También puede agregar excepciones 529, lo que detiene el proceso de asignación y muestra un error si se da una de las condiciones del filtro. En la tabla siguiente aparece una vista general de los componentes de transformación y los botones de la barra de herramientas correspondientes.

Icono	Descripción
	Entrada simple
	Salida simple
	Componente de filtro
	Ordenar datos
	Función integrada
	Función definida por el usuario
	Componente WHERE/ORDER de SQL/NoSQL (*ediciones Professional y Enterprise*)
	Componente de asignación de valores
	Variable
	Función de servicio web (*edición Enterprise*)

 © 2016-2022 Altova GmbH

Icono	Descripción
	Excepción (*ediciones Professional y Enterprise*)
	Constante
	Condición If-Else
	Componente de combinación (*ediciones Professional y Enterprise*)

Apartados de esta sección

Esta sección sobre componentes se divide en distintos apartados:

- Agregar componentes 77
- Componentes básicos 79
- Rutas de acceso de archivos 82

3.1.1 Agregar componentes a la asignación

En este apartado explicamos cómo agregar componentes a una asignación. Para agregar un componente primero debe crear un diseño de asignación nuevo 73 y después seguir estos pasos:

- En el menú **Insertar**, elija un tipo de componente (p. ej. **Esquema o archivo XML**).
- Arrastre un archivo desde el explorador de Windows hasta el área de asignación. Tenga en cuenta que esta operación solo es válida para componentes basados en archivos compatibles.
- Haga clic en el botón correspondiente de la barra de herramientas **Insertar componente** (*imagen siguiente*).

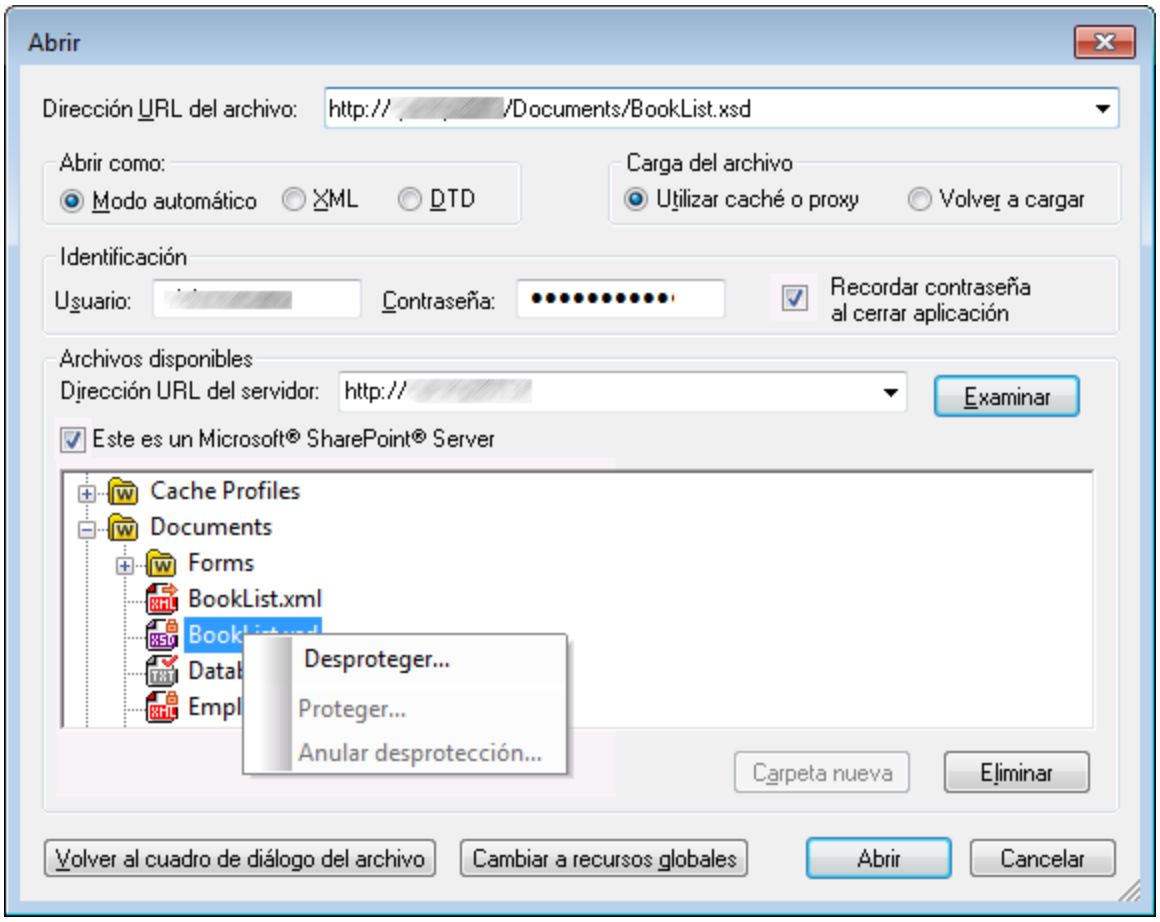
Cada tipo de componente tiene un objetivo y comportamiento concretos Para ver un resumen de los componentes consulte Componentes 75. Si quiere saber más acerca de las estructuras de datos que se pueden usar como componentes de origen and de destino en consulte Orígenes y destinos de datos 123. Para más información sobre los componentes integrados de MapForce que se usan para almacenar datos temporalmente o para transformarlos consulte Diseño de asignaciones 476.

Para ver una lista completa de los componentes consulte la referencia de iconos de componentes que incluimos a continuación.

Agregar componentes desde una URL

Además de añadir archivos locales como componentes de asignación, también puede añadir archivos desde una URL. Tenga en cuenta que esta operación se admite si añade archivos como componentes de origen 73. Los protocolos compatibles son HTTP, HTTPS y FTP. Siga estos pasos para agregar un componente desde una URL:

1. En el menú **Insertar** seleccione componente que quiere agregar (p. ej. **Archivo o esquema XML**).
2. En el cuadro de diálogo **Abrir** (*imagen siguiente*) haga clic en *Cambiar a URL*.
3. Introduzca la URL en la caja de texto **Dirección URL del archivo** y haga clic en **Abrir**. Asegúrese de que el tipo de archivo de la caja de texto **Dirección URL del archivo** es el mismo que el que indicó en el paso 1.



En la lista siguiente se describen las opciones del cuadro de diálogo **Opciones**.

- *Recordar contraseña al cerrar la aplicación:* Si el servidor requiere autenticación con contraseña le pedirá que indique un nombre de usuario y una contraseña. Si quiere que MapForce recuerde sus datos de acceso la próxima vez que lo inicie, introduzca su nombre de usuario y contraseña en el cuadro de diálogo **Abrir** y marque la casilla **Recordar contraseña al cerrar aplicación**.

- *Abrir como:* Define la gramática del analizador. La opción predeterminada y recomendada es **Automático**.

- *Utilizar caché o proxy:* Si sabe que no es probable que el archivo que va a cargar cambie, seleccione **Utilizar caché o proxy** para guardar los datos en la memoria caché y acelerar el proceso de carga. De lo contrario, seleccione **Volver a cargar** para que el archivo se cargue de nuevo cada vez que abra la asignación.

- *Dirección URL del servidor:* En el caso de los servidores compatibles con WebDAV (Autoría y versionado distribuidos por Web) puede explorar archivos una vez haya introducido la URL del servidor en la caja de texto **URL del servidor** y hecho clic en **Examinar**. Aunque en la vista previa aparecen

 © 2016-2022 Altova GmbH

todos los tipos de archivos, debe asegurarse de que abre el mismo tipo de archivo que el que en el paso 1 más arriba. De lo contrario ocurrirán errores.

- *Desproteger/Proteger:* Si está usando un servidor Microsoft SharePoint Server, marque la casilla que así lo indica. Así podrá acceder a una vista previa del estado del archivo. Si quiere asegurarse de que nadie más puede editar el archivo en el servidor mientras lo está usando, haga clic con el botón derecho en el archivo y seleccione **Proteger** (*imagen anterior*). Para desproteger archivos protegidos haga clic con el botón derecho en el archivo y seleccione **Desproteger**.

3.1.2 Componentes básicos

En este apartado explicamos cómo configurar, buscar y manipular componentes estructurales 75. Para más información siga leyendo.

Cambiar la configuración de los componentes

Tras añadir un componente al área de asignación podrá configurarlo desde el cuadro de diálogo **Configuración del componente**. Puede abrir este cuadro de diálogo de una de las siguientes maneras:

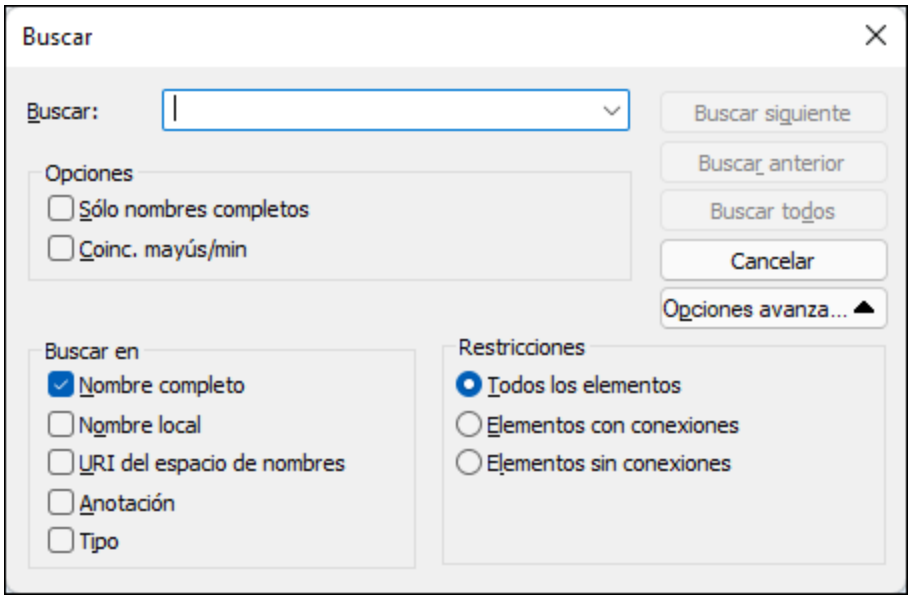
- Haga doble clic en el encabezado del componente.
- Seleccione el componente y haga clic en **Propiedades** en el menú **Componente**.
- Haga clic con el botón derecho en el encabezado del componente y luego en **Propiedades**.

En los componentes basados en archivos (p. ej. un archivo XML) los botones Archivo (*Basic Edition*) o Archivo/Cadena (*ediciones Professional y Enterprise*) aparecen junto al nodo raíz. Con este botón puede acceder a las opciones avanzadas, con las que puede procesar o generar varios archivos en una sola asignación. Para más información consulte la sección Procesar varios archivos de entrada o salida 892. Este botón también permite configurar las opciones para analizar cadenas de texto y serializar datos en cadenas de texto 903.

Buscar dentro de los componentes

Para buscar un nodo en concreto dentro de un componente siga estos pasos:

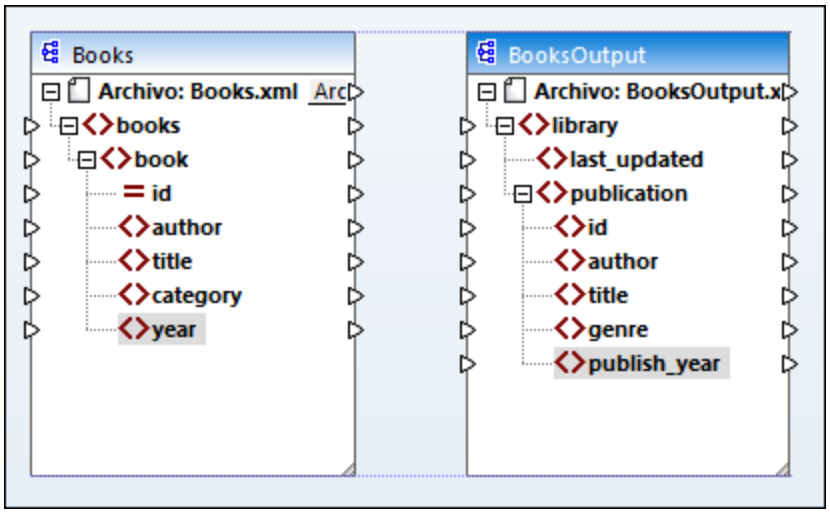
1. Haga clic en el componente donde quiere buscar y pulse **Ctrl+F**.
2. Introduzca un término de búsqueda y haga clic en Buscar siguiente/anterior/todos (*imagen siguiente*).



Use las opciones **Avanzadas** para definir qué elementos (nodos) quiere buscar. También puede restringir las opciones de búsqueda en función de conexiones específicas.

Alinear componentes

Al mover componentes en el área de asignación, MapForce muestra unas líneas guía (líneas de puntos) que ayudan a alinear componentes (*imagen siguiente*).



Para habilitar esta opción, siga estos pasos:

1. En el menú **Herramientas** haga clic en el comando **Opciones**.
2. En la pestaña de opciones **Edición** active/desactive la casilla **Alinear componentes al arrastrarlos con el ratón**.

 © 2016-2022 Altova GmbH

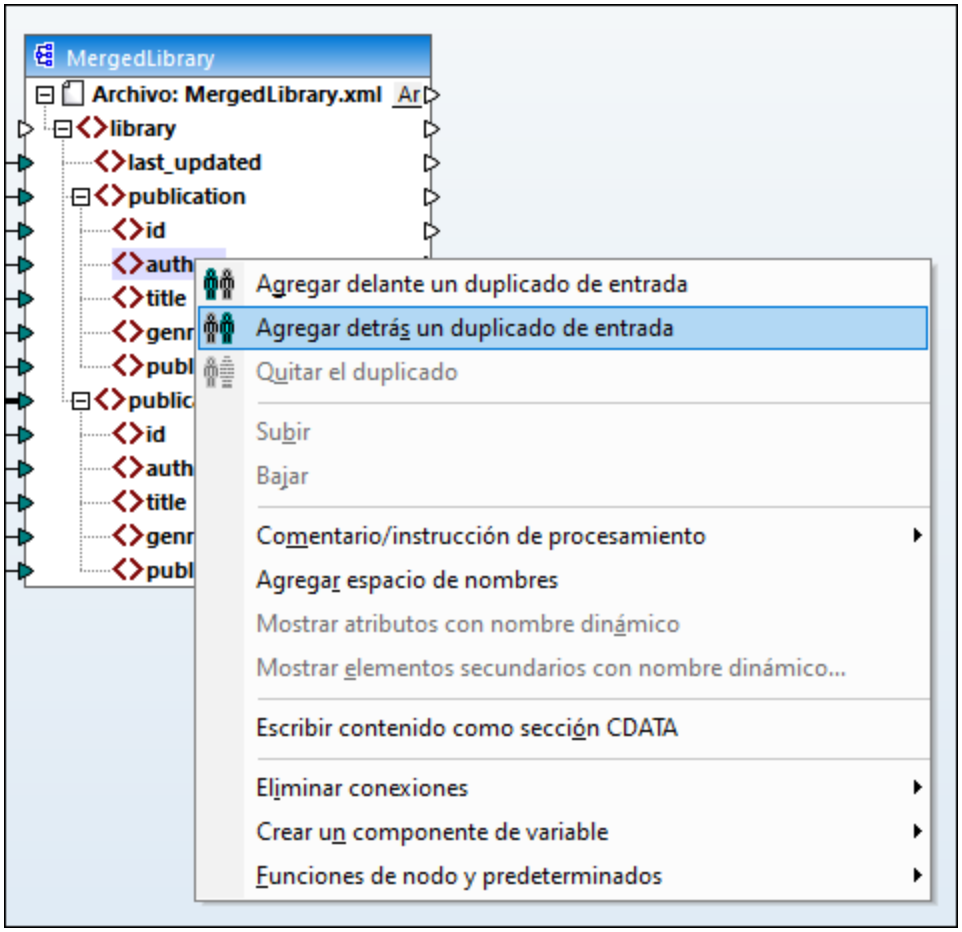
Duplicar elementos de entrada

A veces es necesario configurar un componente para que acepte datos de varios orígenes de datos a la vez. Si quiere que el esquema de destino acepte datos de más de un esquema de origen, puede duplicar los nodos de entrada en el componente de destino. Si lo hace, debe tener en cuenta que: Los nodos duplicados aceptan datos, pero no se pueden asignar datos de nodos duplicados. Puede duplicar tantos nodos como quiera.

Hay dos maneras de duplicar entradas: (i) seleccionar **Agregar delante/detrás un duplicado de entrada** en el menú contextual y (ii) conectar un nodo de origen con un nodo de destino que ya esté conectado a un nodo distinto. A continuación describimos la primera opción. Puede encontrar información sobre la segunda en el segundo tutorial 50.

Agregar duplicados de entrada

Para crear un duplicado de un elemento de entrada, haga clic con el botón derecho en el elemento y seleccione **Agregar delante/detrás un duplicado de entrada** en el menú contextual (*imagen siguiente*). En la imagen siguiente se va a duplicar el nodo `author` para poder asignar los datos al nodo duplicado desde otro nodo de origen.



Nota: no está permitido crear duplicados de atributos XML porque daría lugar a una instancia XML no válida.

3.1.3 Rutas de acceso de archivos

Un archivo de diseño de asignación de datos (**`*.mfd`**) puede tener referencias a varios archivos de instancia o de esquema. Los archivos de esquema le sirven a MapForce para determinar la estructura de los datos que se deben asignar y validarlos. En las ediciones MapForce Professional y Enterprise las asignaciones también pueden incluir referencias a archivos StyleVision Power Stylesheet (**`*.sps`**), que se usan para dar formato a los datos de salida como PDF, HTML o Word. Las asignaciones también pueden contener referencias a bases de datos basadas en archivos, como Microsoft Access o SQLite.

Cuando añadimos un componente a la asignación, MapForce crea referencias a los archivos. No obstante, estas referencias se pueden cambiar y configurar a mano en cualquier momento.

En esta sección explicamos cómo configurar y modificar las rutas de acceso de los diferentes tipos de archivo a los que se hace referencia en las asignaciones de datos. Estos son los apartados de esta sección:

- Usar rutas de acceso relativas en un componente 82
- Rutas de acceso según el entorno de ejecución 84

3.1.3.1 Usar rutas de acceso relativas en un componente

En este apartado explicamos cómo usar las rutas de acceso absolutas y relativas de los archivos a los que hace referencia un componente. Una ruta absoluta muestra la ubicación completa de un archivo y empieza por el directorio raíz. En la imagen siguiente puede ver las rutas absolutas dentro del recuadro rojo. En una ruta relativa se puede ver la ubicación del archivo de manera relativa al directorio de trabajo actual: p. ej. **`Books.xml`**.

En el cuadro de diálogo **Configuración del componente** (*ejemplo siguiente*) puede indicar rutas absolutas o relativas para los distintos archivos a los que hace referencia el componente. A continuación puede ver una lista de esos archivos:

- Archivos de entrada de los que MapForce lee datos;
- Archivos de salida en los que MapForce escribe datos;
- Archivos de esquema, que se pueden aplicar a componentes que tienen un esquema;
- Archivos de estructura, que se usan como parámetros de entrada o salida para funciones definidas por el usuario y variables;
- Archivos StyleVision Power Stylesheet (**`*.sps`**), que se usan para dar formato a los datos para formatos de salida como PDF, HTML y Word;
- Archivos de BD en el caso de los componentes de BD (*en las ediciones Professional y Enterprise*).

Rutas relativas y las acciones cortar y pegar
Si copia un componente de una asignación y lo pega en otra, MapForce comprueba si las rutas relativas de los archivos de esquema se pueden resolver con respecto a la carpeta de la asignación de destino. Si esas rutas no se pueden resolver, la aplicación le pedirá que las convierta en absolutas.

Rutas de acceso rotas
Cuando añada o modifique una referencia de archivo en una asignación y no se pueda resolver la ruta de acceso, MapForce emite un mensaje de advertencia. No obstante, pueden darse referencias de ruta de acceso rotas en estos casos:

 © 2016-2022 Altova GmbH

- Cuando use rutas de acceso relativas y después mueva el archivo de asignación a un directorio nuevo sin mover también los archivos de esquema y de instancia.
- Cuando use rutas de acceso absolutas de archivos que están en el mismo directorio que el archivo de asignación y después mueva el directorio a otra ubicación.

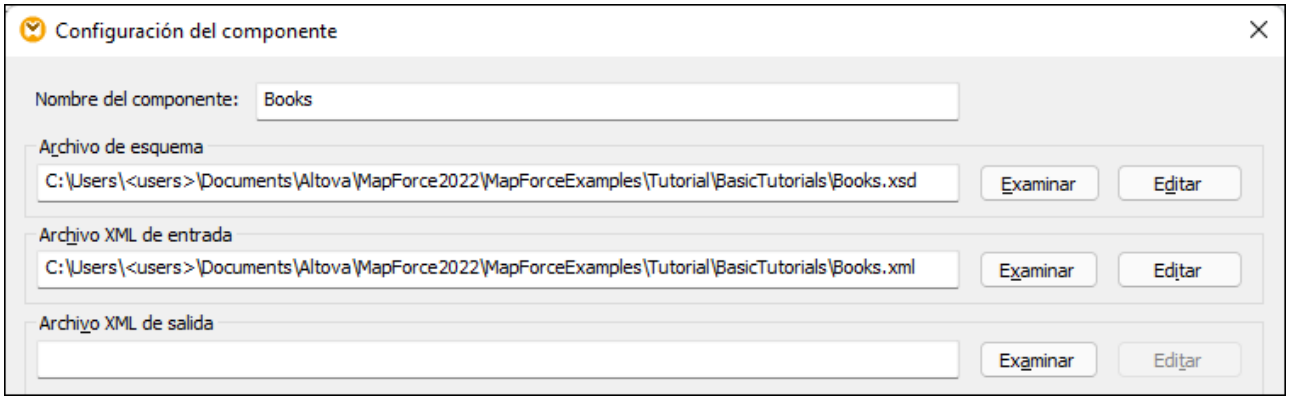
Cuando esto ocurra, MapForce resaltará el componente en color rojo. En casos así, la solución consiste en hacer doble clic en el título del componente y actualizar las referencias de ruta de acceso rotas en el cuadro de diálogo **Configuración del componente**. Consulte también Cambiar configuración de los componentes 79.

Ejemplo: Componente XML

En el ejemplo siguiente se ve cómo se pueden usar las rutas de acceso en un componente XML. Si quiere guardar todos los archivos de la asignación como relativos al archivo de asignación (`.mfd`) debe marcar la casilla **Guardar todas las rutas de acceso de archivos como relativas al archivo MFD** de la parte inferior del cuadro de diálogo **Configuración del componente**. Esta es la opción predeterminada y recomendada, y afecta a todos los archivos a los que hace referencia el componente (*en el recuadro rojo en la imagen siguiente*). Si todavía no ha guardado la asignación, en el cuadro de diálogo **Configuración del componente** verá que las rutas de acceso al esquema y a los archivos de instancia son absolutas. Para convertirlas en relativas:

1. Cree una asignación nueva 73 y agregue un componente estructural 77: p. ej. un archivo XML que tenga asignado un esquema XML.
2. Haga doble clic el encabezado del componente para abrir el cuadro de diálogo **Configuración del componente**.
3. Marque la casilla **Guardar todas las rutas de acceso de archivos como relativas al archivo MFD** de la parte inferior del cuadro de diálogo **Configuración del componente**.
4. Guarde la asignación.
5. Ahora puede volver a abrir el cuadro de diálogo **Configuración del componente**, que ahora contendrá rutas relativas en los campos de texto correspondientes.

Nota: las rutas que hacen referencia a unidades no locales o usan una URL no se pueden convertir en relativas.



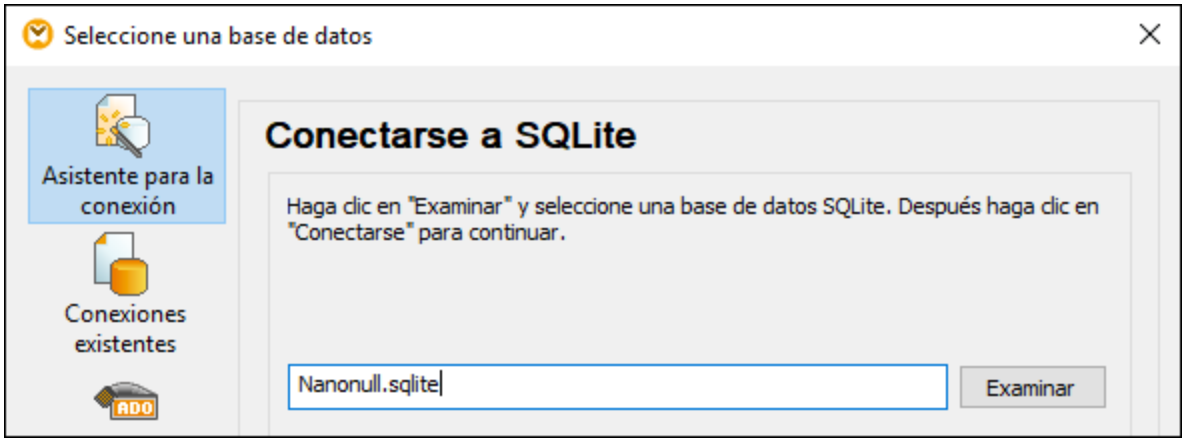
Si se marca la casilla **Guardar todas las rutas de acceso de archivos como relativas al archivo MFD** MapForce hará un seguimiento de los archivos a los que hace referencia el componente incluso aunque guarde la asignación en una carpeta distinta. Si todos los archivos están en el mismo directorio que la asignación, las referencias de las rutas no se rompen aunque mueva todo el directorio a otra ubicación en disco.

La opción **Guardar todas las rutas de acceso de archivos** como relativas al archivo MFD afecta a estos archivos:

- Archivos de estructura, que usan parámetros complejos de entrada o salida para funciones definidas por el usuario, y variables de tipo complejo;
- Archivos planos de entrada o salida (*en las ediciones Professional y Enterprise*);
- Archivos de esquema a los que hacen referencia los componentes de BD que admiten campos XML (*en las ediciones Professional y Enterprise*);
- Archivos de BD (*en las ediciones Professional y Enterprise*);
- Archivos XBRL, FlexText, EDI, Excel 2007+ o JSON de entrada o salida (*en la edición Enterprise*);

Ejemplo: Componente de BD (Ediciones Professional y Enterprise)

Si agrega a la asignación un archivo de BD 163, p. ej. Microsoft Access o SQLite, puede introducir una ruta relativa en vez de una absoluta en el cuadro de diálogo **Seleccionar una base de datos** (*imagen siguiente*). Antes de introducir las rutas de acceso asegúrese de que ha guardado primer el archivo **`.mfd`**. Si quiere cambiar la ruta de un componente de BD que ya está en la asignación, haga clic en **Cambiar** en el cuadro de diálogo **Configuración del componente**.



Nota: cuando genere el código de programa, compile archivos de ejecución de MapForce Server (**`.mfx`**) o implemente la asignación en FlowForce Server, las rutas relativas se convierten en absolutas si marca la casilla **Convertir las rutas de acceso en absolutas en el código generado** en las opciones de la asignación 115. Para más información consulte Rutas de acceso según el entorno de ejecución 84.

3.1.3.2 Rutas de acceso según el entorno de ejecución

Si quiere generar código a partir de asignaciones, compilar asignaciones en archivos de ejecución de MapForce Server (**`.mfx`**) o implementar asignaciones en FlowForce Server, los archivos que se generan los ejecuta el entorno de destino que haya elegido: por ejemplo, RaptorXML Server, MapForce Server o una aplicación C#. Para que la asignación se pueda ejecutar correctamente las rutas relativas deben corresponderse con el formato del entorno de ejecución. Estas son las rutas base para los distintos lenguajes de destino:

 © 2016-2022 Altova GmbH

Lenguaje de destino	Ruta base
XSLT, XSLT2, XSLT3	Ruta del archivo XSLT.
XQuery*	Ruta del archivo XQuery.
C++, C#, Java*	Directorio de trabajo de la aplicación generada.
Motor integrado* (para la vista previa de la asignación en con MapForce)	Ruta del archivo de asignación (**.mfd**).
Motor integrado* (para ejecutar la asignación con MapForce Server)	El directorio de trabajo actual.
Motor integrado* (para ejecutar la asignación con MapForce Windows Server bajo el Server Archivos)	El directorio de trabajo del trabajo o el de FlowForce Server.

** Lenguajes disponibles en las ediciones MapForce Professional y Enterprise.*

De ruta relativa a ruta absoluta
Cuando genere el código de programa, compile archivos de ejecución de MapForce Server (**.mfx**) o implemente la asignación en FlowForce Server, las rutas relativas se convierten en absolutas si marca la casilla **Convertir las rutas de acceso en absolutas en el código generado** en las opciones de la asignación 115.

Si genera código y esa casilla está marcada, MapForce resuelve las rutas relativas basadas en el directorio del archivo **.mfd** y las convierte en absolutas en el código generado. Esta opción afecta a las rutas de los siguientes archivos:

- Archivos de instancia de entrada y de salida de todos los componentes basados en archivos;
- Archivos de BD Access y SQLite que se usen como componentes de asignación (*en las ediciones Professional y Enterprise*).

Rutas de acceso a bibliotecas en el código generado

Los archivos de asignación pueden contener referencias de rutas de acceso a distintas bibliotecas, Por ejemplo,, puede importar funciones definidas por el usuario desde otro archivo de asignación, desde bibliotecas personales XSLT, XQuery*, C#* o Java* o desde archivos **.mff*** (MapForce Funcion). Para más información consulte Gestionar bibliotecas de funciones 553.

** Características disponibles en las ediciones MapForce Professional y Enterprise.*

La opción **Convertir las rutas de acceso en absolutas** en el código generado solamente afecta a los componentes de asignación, pero no a las rutas de bibliotecas externas. En el caso de las bibliotecas que no sean XSLT ni XQuery, la ruta de acceso se convierte en absoluta en el código generado. Por ejemplo, si un archivo de asignación contiene referencias de biblioteca como archivos .NET **.dll** o archivos Java **.class**, y si quiere ejecutar el código generado en algún otro entorno, las bibliotecas a las que se hace referencia deben existir en la misma ruta que el entorno de destino.

Si quiere generar un archivo XSLT o XQuery a partir de una asignación, configure la ruta para que sea relativa al archivo XSLT o XQuery:

1. Abra la configuración de la asignación 115:

2. Marque la casilla **Bibliotecas de referencia con rutas relativas a los archivos &XSLT / XQuery generados**. Asegúrese de que el archivo de biblioteca XSLT/XQuery existe en esa ruta.

© 2016-2022 Altova GmbH

3.2 Trabajar con conexiones

Una conexión es una línea que conecta un elemento de origen con uno de destino. Las conexiones son una representación visual de cómo están asignados los datos de unos nodos a otros. Los apartados siguientes describen las distintas acciones relacionadas con conexiones que se pueden ejecutar.

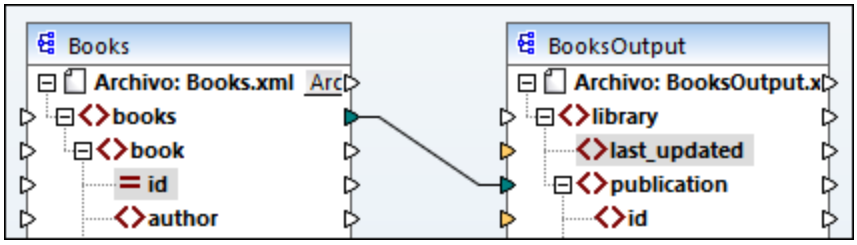
Crear, copiar y eliminar conexiones

Para crear una conexión entre dos elementos pulse y mantenga pulsado el conector de salida 74 de un nodo de origen y arrástrelo hasta el nodo de destino. Un conector de entrada solo acepta *una conexión*. Si intenta agregar una segunda conexión a un elemento de entrada, MapForce le pedirá que reemplace una conexión por otra o que duplique el elemento de entrada 81. Un elemento de salida sí que puede tener varias conexiones a distintos elementos de entrada.

Para copiar una conexión a otro elemento pulse y mantenga pulsada la parte gruesa que hay al final de la conexión (*véase la imagen de Mover conexiones*) y arrástrela hasta su destino mientras mantiene pulsada la tecla **Ctrl**.

Para eliminar una conexión haga clic en ella y pulsar la tecla **Supr**. También puede hacer clic con el botón derecho en la conexión y seleccionar **Eliminar** en el menú contextual.

Entradas obligatorias
Para facilitar la creación de asignaciones, MapForce resalta en color naranja las entradas obligatorias de los componentes de destino. En el ejemplo siguiente puede ver cómo al conectar el elemento `book` del componente `Books` al elemento `publication` del componente `BookOutput` se resaltan los conectores de los nodos obligatorios del componente `BookOutput`. Si no conecta las entradas obligatorias, los nodos correspondientes no se asignan a los elementos de destino y la asignación no será válida.



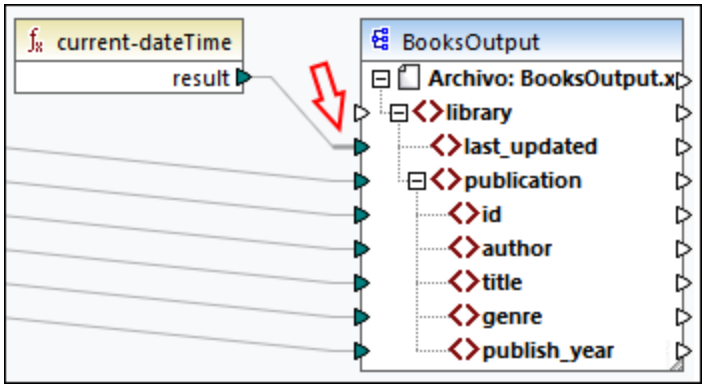
Conexiones antecesoras ausentes
Cuando se crean conexiones entre nodos de origen y destino de forma manual, MapForce analiza los posibles resultados de la asignación. Si se crea una asignación entre dos nodos secundarios, MapForce puede emitir una notificación para sugerir una conexión entre el elemento primario del nodo de origen y el elemento primario del nodo de destino. Esta notificación le ayudará a evitar que en la ventana de vista previa de resultados aparezca un solo nodo secundario.

Si prefiere deshabilitar este tipo de notificaciones:

1. En el menú **Herramientas** haga clic en el comando **Opciones**.
2. Haga clic en el grupo **Mensajes** para abrirlo.
3. Desactive la casilla **Al crear una conexión nueva, sugerir conexión de elementos antecesores**.

Mover conexiones

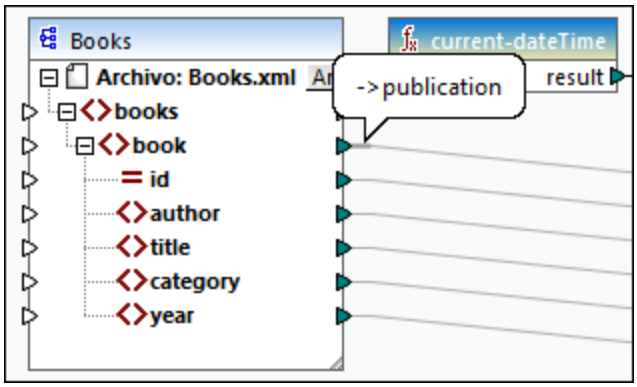
Para mover una conexión a otro nodo pulse y mantenga pulsada la parte gruesa que hay al final de la conexión (*véase la imagen de más abajo*) y arrástrela hasta su destino.



Ver la información rápida de las conexiones

La información rápida de las conexiones permite ver los nombres de los (i) nodos a los que se asignan datos o (ii) nodos desde los que se asignan datos. Para poder ver esa información pulse el botón de la barra de herramientas (**Ver información rápida**). Para ver los nombres de los nodos a los que se asignan los datos haga que el cursor apunte a la parte gruesa de la conexión que se encuentra junto al conector de salida (*imagen siguiente*). Para ver el nombre de un nodo desde el que se asignan datos haga que el cursor apunte a la parte gruesa de la conexión que se encuentra junto al conector de entrada. Si se han definido varias conexiones desde un mismo elemento de salida, el número máximo de nombres que pueden aparecer en la información rápida es diez.

En el ejemplo siguiente el elemento de destino al que se asignan los datos del elemento `book` es `publication`.



Cambiar la configuración de las conexiones

Si quiere cambiar la configuración de las conexiones tiene varias opciones:

 © 2016-2022 Altova GmbH

- Seleccionar una conexión Después haga clic en el comando **Propiedades** del menú **Conexión** .
- Haga doble clic en la conexión.
- Haga clic con el botón derecho en la conexión y después elija **Propiedades**.

Para más información consulte Configuración de las conexiones 97.

Resaltar conexiones de forma selectiva

MapForce permite resaltar conexiones en una asignación. Esta característica puede ser útil si una asignación tiene muchos componentes con muchas conexiones. Al resaltar conexiones de forma selectiva es más fácil comprobar si los nodos del componente seleccionado están asignados correctamente. Observe que el término *conector* de los botones de la barra de herramientas se refiere a una conexión en el sentido de una línea que une nodos de componente. A continuación puede ver las opciones disponibles.

	Mostrar los conectores del componente seleccionado (solo conexiones directas)
	Mostrar los conectores desde su origen a su destino (conexiones directas e indirectas)

Solo conexiones directas

Si deja *sin pulsar* botón para **conexiones directas** puede ver todas las conexiones en negro. Si pulsa el botón para **conexiones directas** solo aparecen en negro las conexiones relacionadas con el componente seleccionado en ese momento. El resto de conexiones aparecen de color gris pálido.

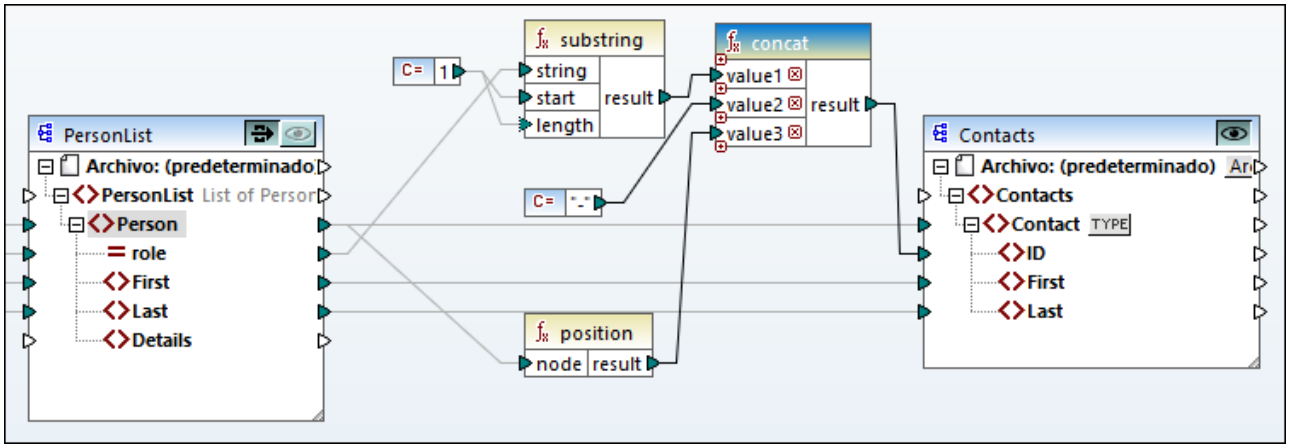
Conexiones directas e indirectas

El botón para **conexiones de origen a destino** se habilita solo si pulsa el botón para **conexiones directas**. Si pulsa el botón para **conexiones de origen a destino** puede hacer un seguimiento de las conexiones del componente seleccionado en ese momento, incluidas las conexiones directas y las de sus componentes conectados, hasta los archivos de origen y destino.

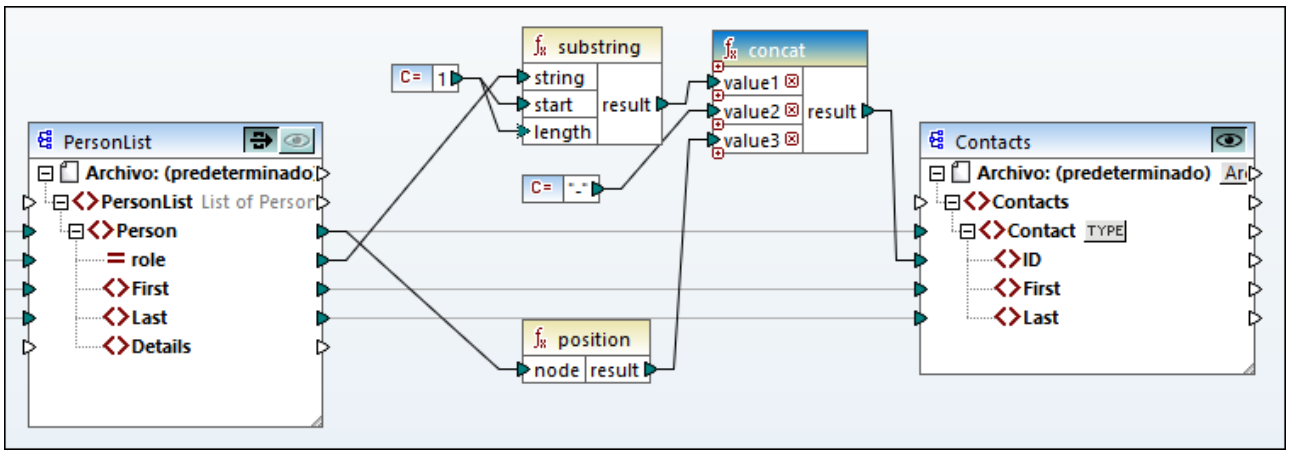
A continuación puede ver un ejemplo que explica cómo funcionan estas dos opciones.

Ejemplo

En la imagen siguiente puede ver una parte de la asignación **`ChainedPersonList.mfd`**, que puede encontrar en la carpeta **`MapForceExamples`**. En la asignación siguiente hemos pulsado el botón para **conexiones directas**, hemos hecho clic en el encabezado del componente **`concat`** pero todavía no hemos pulsado el botón Mostrar los conectores desde su origen a su destino. Esto quiere decir que ahora solo las conexiones directas que van de la función **`concat`** a la constante **`"-"`**, los componentes `substring`, `position` y `Contacts`, aparecen en negro. El resto de conexiones de la asignación aparecen de color gris pálido.



El paso siguiente es pulsar el botón **Mostrar los conectores desde su origen a su destino**. En la imagen siguiente se pueden ver los cambios:



Al pulsar el botón **Mostrar los conectores desde su origen a su destino** hay más conexiones que aparecen en negro: (i) las conexiones que van de la función **`substring`** a la constante **`1`** y al componente `PersonList`, y (ii) la conexión que va de la función **`position`** al componente `PersonList`. El resto de conexiones entre el componente `PersonList` y su componente anterior siguen siendo de color gris pálido. Es decir, si pulsa el botón **Mostrar los conectores desde su origen a su destino** puede hacer un seguimiento de las conexiones directas de ese componente. Si el componente seleccionado está conectado a algún componente de transformación 76 (p. ej. funciones, constantes, filtros, componentes de ordenación, componentes SQL-NoSQL-WHERE/ORDER, condiciones if-else, asignaciones de valores), también podrá ver sus conexiones, lo que incluye los componentes estructurales 75 (como `PersonList`, más arriba) las variables, los componente de combinación o las funciones de servicios web a las que están conectados estos componentes de transformación.

Apartados de esta sección

Esta sección sobre conexiones se divide en distintos apartados:

- Tipos de conexión 91
- Configuración de las conexiones 97

 © 2016-2022 Altova GmbH

- Menú contextual de las conexiones 99
- Conexiones defectuosas 101
- Conservar conexiones tras eliminación de componentes 102

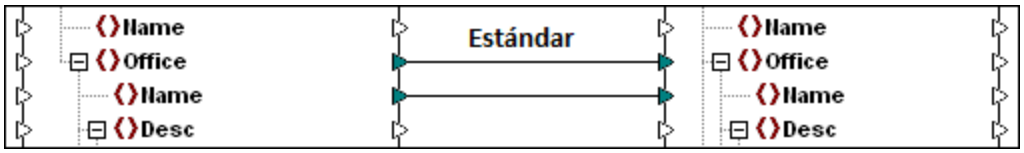
3.2.1 Tipos de conexión

Estos son los tipos de conexión que se pueden generar en MapForce:

- Conexiones basadas en el destino 91 (estándar)
- Conexiones basadas en el origen 91 (contenido mixto)
- Conectar los secundarios equivalentes 94
- Conexiones de copia total 95 (copia los elementos secundarios)

Conexiones basadas en el destino vs. conexiones basadas en el destino

Las conexiones basadas en el destino y conexiones basadas en el destino se excluyen mutuamente. La elección de uno u otro tipo depende del orden en que se deban asignar los nodos. En las conexiones basadas en el destino, el orden de los nodos en el resultado viene dado por el esquema de *destino*. Este tipo de conexión se puede usar en la mayoría de asignaciones y es el tipo de conexión predeterminado en MapForce. Las conexiones basadas en destino aparecen como líneas sólidas (*imagen siguiente*).



Las conexiones basadas en origen pueden no ser adecuadas si quiere asignar nodos XML con contenido mixto (nodos secundarios y texto). En este caso recomendamos usar una conexión basada en origen 91: El orden de los nodos en el resultado aquí viene dado por el esquema de *origen*.

Secundarios equivalentes y conexiones de copia total

Los secundarios equivalentes y conexiones de copia total pertenecen a un tipo distinto de conexiones. Estas conexiones asignan datos entre nodos con nodos secundarios que son parecidos o idénticos en los componentes de origen y destino. Las conexiones de copia total se parecen a las de secundarios equivalentes pero tienen solamente una conexión sólida en lugar de varias para no abarrotar la vista.

En esta sección explicamos los distintos tipo de conexión y en qué casos es útil cada uno.

3.2.1.1 Conexiones basadas en el origen

Una conexión basada en origen permite asignar el contenido mixto (texto y nodos secundarios) automáticamente en el mismo orden que el definido en el archivo XML de *origen*. Estas conexiones aparecen como líneas de puntos al nivel del nodo principal (*véase el elemento <para> de la asignación*). En este apartado explicamos cómo asignar contenido mixto. También verá el efecto que tiene usar conexiones estándar (basadas en destino) con contenido mixto.

Nota: las conexiones basadas en origen también se pueden usar en campos de BD con contenido mixto (*en las ediciones Professional y Enterprise*).

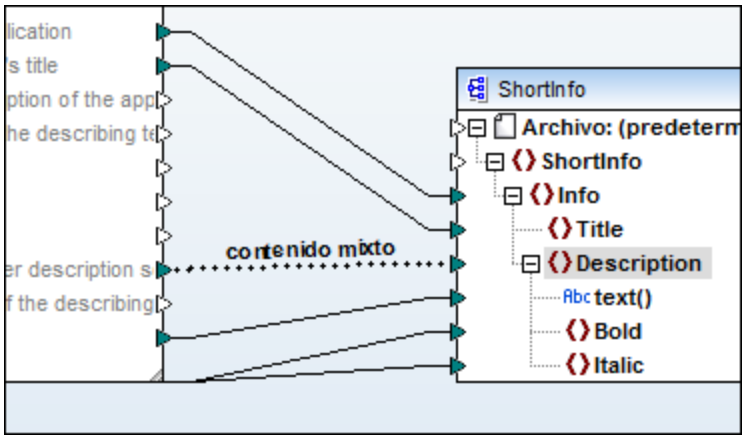
Nota: para que acepten contenido mixto los componente de destino deben tener nodos de contenido mixto.

Asignar contenido mixto

En este apartado explicamos cómo asignar contenido mixto usando una conexión basada en origen. Para ello necesita estos archivos: **`Tut-OrgChart.xml`**, **`Tut-Orgchart.mfd`** y **`Tut-OrgChart.xsd`**., que encontrará en la carpeta Tutorial 23.

Instancia XML de origen

A continuación puede ver un extracto de **`Tut-OrgChart.xml`**. En este ejemplo nos centraremos en el elemento de contenido mixto `<para>` y en sus nodos secundarios `<bold>` e `<italic>`. El elemento `<para>` también contiene una instrucción de procesamiento (`<?sort alpha-ascending?>`) y un comentario (`<!--Company details... -->`), que se pueden asignar también, como verá más abajo. Fíjese en la secuencia de los nodos `text` y `bold/italic` en el archivo de instancia XML.

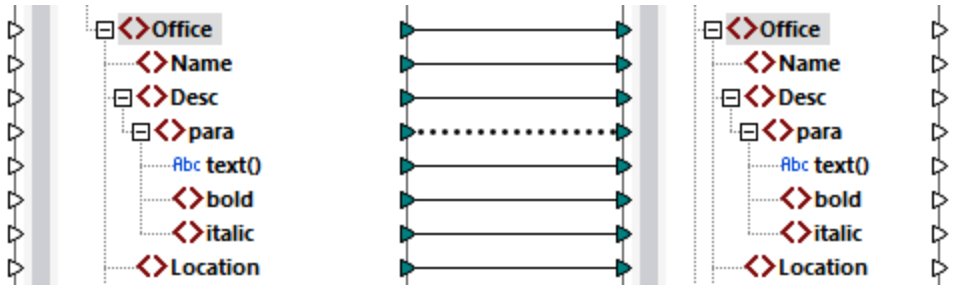


Asignar el elemento <para>

En la imagen siguiente se puede ver una parte de **`Tut-Orgchart.mfd`**. En el ejemplo siguiente las líneas de puntos indican que el elemento `<para>` tiene contenido mixto. Para crear una conexión de contenido mixto siga estos pasos:

1. Seleccione el comando de menú **Conexión | Conectar los secundarios equivalentes**, que conectar lo secundarios equivalentes 94 automáticamente. También puede asignar manualmente el nodo `<para>` a sus nodos secundarios.
2. Conecte el elemento `< para>` del componente de origen con el elemento `< para>` del componente de destino. Aparece un cuadro de diálogo que le pide que defina las conexiones como basadas en el origen.
3. Haga clic en **Sí** para crear una conexión de contenido mixto.
4. Haga clic en el panel **Resultados** para ver el resultado de la asignación. Haga clic en el botón (**Ajuste automático de línea**) de la barra de herramientas del panel **Resultados** para ver todo el código. Ahora el contenido mixto del nodo `<para>` está asignado en el mismo orden en que aparece en el archivo XML de origen.

 © 2016-2022 Altova GmbH



Instrucciones de procesamiento y comentarios

Si una asignación tiene instrucciones de procesamiento y/o comentarios y los quiere asignar también, siga estos pasos:

1. Haga clic con el botón derecho en la conexión de contenido mixto (línea de puntos) y seleccione **Propiedades**.
2. En el punto **Basada en origen (contenido mixto)** marque las casillas **Asignar instrucciones de procesamiento** y **Asignar comentarios**.

Conexiones basadas en destino con contenido mixto

Usar conexiones basadas en destino para contenido mixto puede tener consecuencias no deseadas. Para ver cómo afectan estas conexiones al orden de los nodos de contenido mixto siga estas instrucciones:

1. Abra la asignación **`Tut-OrgChart.mfd`**, que OrgChart.mfd la carpeta **`Tutorial`**.
2. Pulse el botón de la barra de herramientas (Conectar automáticamente los secundarios equivalentes 94). Desmarque la casilla **Crear conexiones de copia total** en la configuración de las conexiones de secundarios equivalentes 94. Esto evita que MapForce cree conexiones de copia total 95 automáticamente.
3. Cree una conexión entre el nodo `para` de origen y el nodo `para` de destino. Aparece un de diálogo que le pide que defina las conexiones como basadas en el origen. Haga clic en **No** para crear una conexión basada en destino.
4. Haga clic en el panel **Resultados** para ver el resultado de la asignación (*imagen siguiente*).

```
6          <Desc>
7              <para>The company was established in  in 1995. Nanonull develops nanoelectronic
technologies for  February 1999 saw the unveiling of the first prototype  The company hopes
to expand its operations  to drive down operational costs.
8                  <bold>Vereno</bold><bold>Nano-grid.</bold><italic>multi-core processors.</
italic><italic>offshore</italic></para>
9              <para>White papers and further information will be made available in the near
future.
10             </para>
11         </Desc>
```

En la imagen anterior puede ver que el contenido del elemento `text()` de origen se ha asignado al destino. Sin embargo, el orden de los nodos secundarios(`bold` y `italic`) del resultado sigue el orden que tienen esos nodos en el esquema XML de destino. Esto quiere decir que los elementos `bold` e `italic` no están integrados en el texto sino que se asignan por separado.

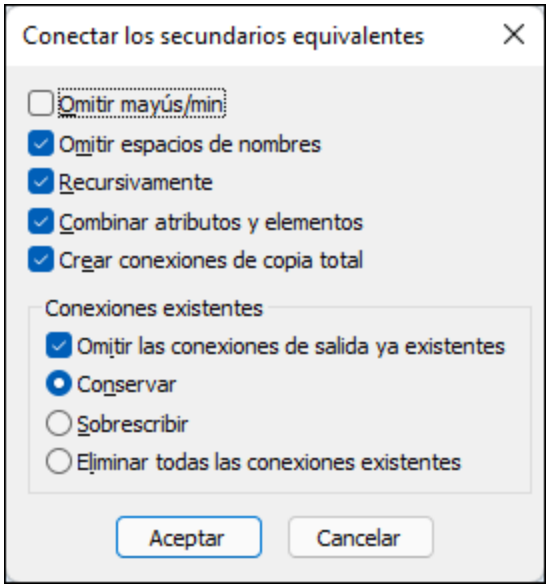
3.2.1.2 Conectar los secundarios equivalentes

Las conexiones de secundarios equivalentes conectan automáticamente todos los secundarios equivalentes que tengan el mismo nombre en los archivos de origen y de destino. Hay varias maneras de habilitar esta opción:

- Haga clic en el botón de la barra de herramientas (**Conectar los secundarios equivalentes automáticamente**).
- Vaya al menú Conexión y haga clic en **Conectar los secundarios equivalentes automáticamente**.

Configurar las conexiones de secundarios equivalentes

Para cambiar la configuración de los secundarios equivalentes haga clic con el botón derecho del ratón en cualquier conexión y seleccione la opción **Conectar los secundarios equivalentes** en el menú contextual (*imagen siguiente*). También puede ir al menú **Conexión** y hacer clic en **Configurar la conexión de secundarios equivalentes**.



En la lista siguiente se describen las opciones disponibles en el cuadro de diálogo **Configurar la conexión de secundarios equivalentes**. Estas opciones solamente se aplican si está activado el botón de la barra de herramientas (**Alternar la conexión automática de secundarios**).

Configuración general

- *Omitir mayús/min* Los secundarios equivalentes se conectan independientemente de si sus nombres son iguales pero no coinciden en el uso de mayúsculas y minúsculas.
- *Omitir espacios de nombres* Los secundarios equivalentes se conectan independientemente de los espacios de nombres de los nodos secundarios.
- *Recursivamente* Esta opción crea conexiones nuevas entre secundarios equivalentes si estos tienen el mismo nombre. En este caso no se tiene en cuenta la profundidad de anidación de los nodos.
- *Combinar atributos y elementos* Esta opción permite crear conexiones entre atributos y elementos que tienen el mismo nombre. Por ejemplo, si existen dos nodos `Name` se crea una conexión aunque uno sea un elemento y el otro un atributo.

 © 2016-2022 Altova GmbH

- *Crear conexiones de copia total* Esta casilla está activada por defecto. Es una opción que crea (si es posible) conexiones de copia total 95 que crean asignaciones de datos entre nodos que tienen nodos secundarios que son muy parecidos o idénticos.

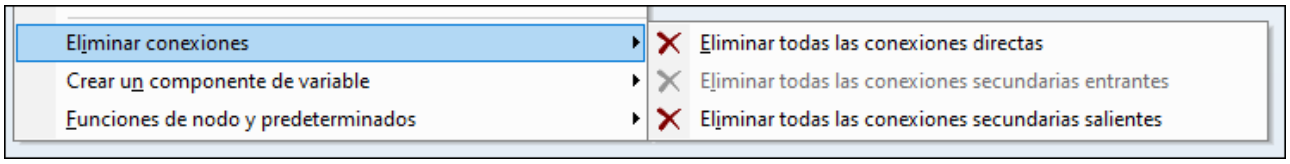
Conexiones existentes

- *Omitir las conexiones de salida ya existentes* Crea más conexiones para todos los secundarios equivalentes, aunque ya tengan conexiones de salida.
- *Conservar* Esta opción mantiene las conexiones que ya existen.
- *Sobrescribir* Esta opción sobrescribe las conexiones que ya existen.
- *Eliminar todas las conexiones existentes* Esta opción elimina todas las conexiones existentes antes de crear conexiones nuevas.

Eliminar conexiones en grupo

Si quiere eliminar conexiones en grupo siga estos pasos:

1. Haga clic con el botón derecho en un nombre de nodo dentro del componente.
2. Seleccione **Eliminar conexiones | Eliminar todas las conexiones <...>** en el menú contextual (*imagen siguiente*).



- *Eliminar todas las conexiones directas:* esta opción elimina todas las conexiones que están asignadas directamente a o desde el nodo seleccionado.
- *Eliminar todas las conexiones secundarias entrantes:* esta opción se activa solamente si hace clic con el botón derecho del ratón en un nodo principal del componente de destino. Esta opción elimina todas las conexiones secundarias entrantes del nodo principal seleccionado.
- *Eliminar todas las conexiones secundarias salientes:* esta opción se activa solamente si hace clic con el botón derecho del ratón en un nodo principal del componente de origen. Esta opción elimina todas las conexiones secundarias salientes del nodo principal seleccionado.

3.2.1.3 Conexiones de copia total

Las conexiones de copia total crean asignaciones de datos entre nodos que tienen nodos secundarios que son muy parecidos o idénticos. Solo se pueden establecer este tipo de conexiones si los dos formatos son idénticos (p. ej. JSON a JSON o XML a XML). Este principio también se aplica a todos los componentes de texto: archivos planos, FlexText y archivos EDI. Todos estos formatos son archivos de texto, por lo que puede combinar cualquiera de ellos y crear una conexión de copia total entre los archivos EDI y FlexText, por ejemplo. Consulte Asignación: origen y destino 23 para más detalles sobre los formatos que puede usar como origen y destino.

La ventaja principal de las conexiones de copia total es que simplifican visualmente el área de trabajo: se crea una conexión, representada por una línea gruesa, en lugar de muchas conexiones (*imagen siguiente*). En los apartados siguientes explicamos cómo crear conexiones de copia total automáticamente y manualmente.

Crear conexiones de copia total automáticamente

Para crear una conexión de copia total automática:

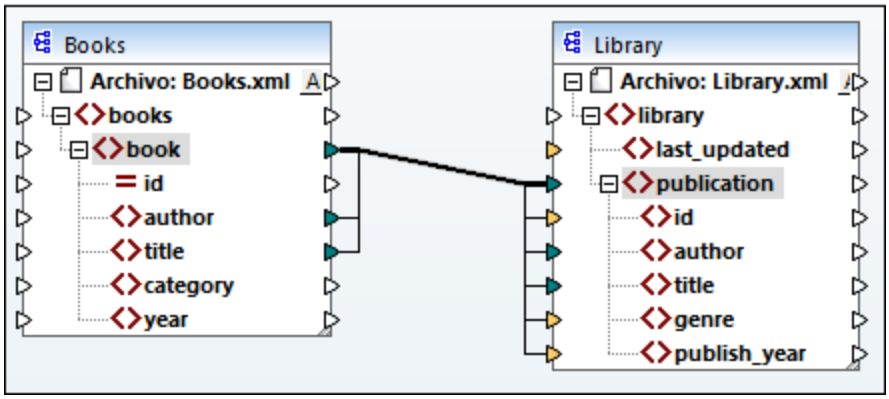
1. Vaya al menú **Conexión**.
2. Haga clic en **Configurar la conexión de secundarios equivalentes**.
3. Marque la casilla **Crear conexiones de copia total** y haga clic en **Aceptar**.
4. Pulse el botón **Alternar la conexión automática de secundarios** de la barra de herramientas. También puede ir al menú **Conexión** y hacer clic en el comando **Conectar automáticamente los secundarios equivalentes**.

Si el tipo y/o los nombres de los nodos secundarios son distintos, la conexión de copia total no se crea automáticamente, sino que tiene que crearla manualmente.

Crear conexiones de copia total manualmente

Para crear una conexión de copia total manualmente:

1. Agregue un archivo de origen: En el menú **Insertar** haga clic en **Archivo o esquema XML** y navegue hasta **`Books.xml`**,que está en lacarpeta de tutoriales básicos 23.
2. Agregue un archivo de destino: En el menú **Insertar** haga clic en **Archivo o esquema XML** y navegue hasta **`Library.xsd`**, que está en la misma carpeta que **`Books.xml`**. Haga clic en **Omitir** cuando MapForce le pida que indique un archivo XML de muestra.
3. Asigne el nodo `<book>` del componente **`Books`** al nodo `<publication>` del componente **`Library`**. Como las estructuras de `<book>` y `<publication>` no coinciden, no se crea la conexión de copia total. Lo que sí se activa es la función **Conectar automáticamente los secundarios equivalentes**, que conecta automáticamente todos los nodos secundarios que tienen el mismo nombre, como explicamos en el tutorial nº 1 45.
4. Para cambiar la conexión automática a una conexión de copia total haga clic con el botón derecho entre `<book>` y `<publication>` y seleccione **Copia total (copia los elementos secundarios)** en el menú contextual.
5. Aparece una ventana de texto que sugiere que reemplace las conexiones existentes con una conexión de copia total. Haga clic en **Aceptar**. Ahora hay una conexión de copia total entre el origen y el destino (*imagen siguiente*).



En la asignación anterior solo dos nodos secundarios son idénticos en las dos estructuras: `<author>` y `<title>`. Esto quiere decir que existe una conexión de copia total entre esos nodos. Los nodos secundarios que no son idénticos no se pueden conectar. En la imagen se puede ver que el elemento `id` no se incluye en la conexión de copia total porque no es del mismo tipo en el origen y el destino: `id` es un atributo en el origen

 © 2016-2022 Altova GmbH

y un elemento en el destino. Si intenta crear una conexión entre nodos que no son iguales, como `<category>` y `<genre>`, MapForce le pide que reemplace o duplique la entrada.

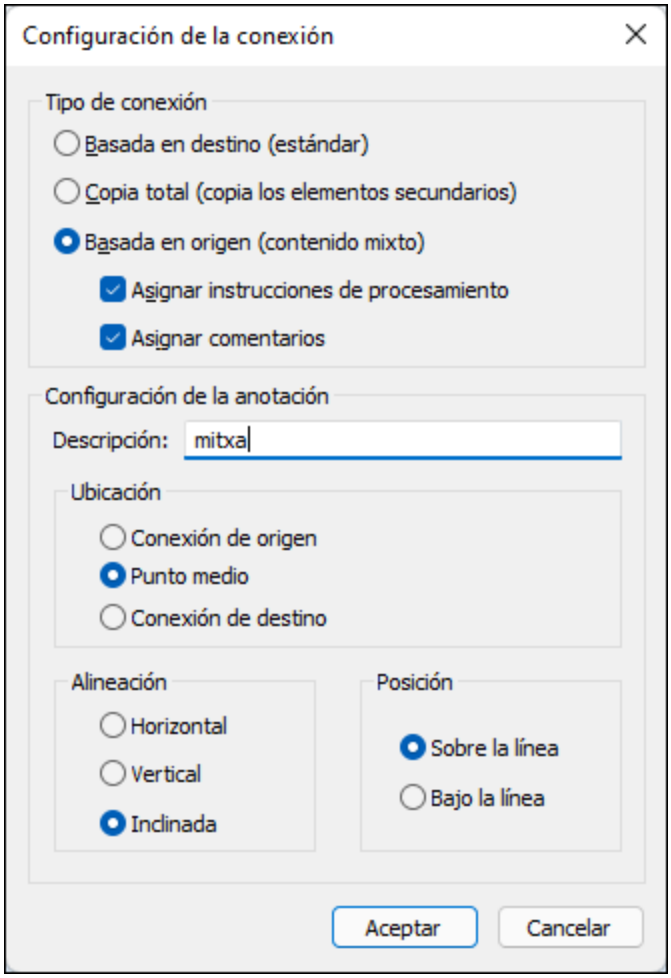
Duplicar la entrada 81 solo es útil si quiere que el destino acepte datos de más de una entrada, algo que no es necesario en este caso. Si elige reemplazar la conexión de copia total aparece un mensaje que le pide que resuelva o elimine esta conexión. Haga clic en **Resolver la conexión de copia total** si quiere reemplazarla con conexiones basadas en el destino 91 . Si prefiere eliminar la conexión de copia total por completo haga clic en **Eliminar conexiones secundarias**.

Nota:

Cuando se crea una conexión de copia total entre un esquema y un parámetro de una función definidas por el usuario 575, los dos componentes deben basarse en el mismo esquema. Sin embargo, no es necesario que los dos tengan el mismo elemento raíz.

3.2.2 Configuración de las conexiones

Este cuadro de diálogo sirve para configurar conexiones. Para abrirlo haga doble clic en una conexión. También puede hacer clic con el botón derecho en la conexión y seleccionar **Propiedades** en el menú contextual. Los ajustes se dividen en dos partes: tipos de conexión y configuración de la anotación. Para más información siga leyendo.

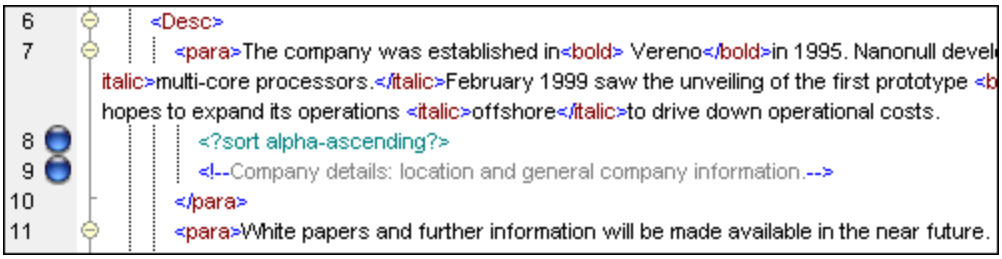


Tipos de conexión

Puede elegir uno de los tipos de conexión que describimos a continuación:

- Basada en destino (estándar) 91: estas conexiones se pueden usar en casi todas las asignaciones.
- Copia total (copia lo elementos secundarios) 95: si un componente de origen y uno de destino tienen los mismos nodos o nodos parecidos con secundarios equivalentes se crea una conexión de copia total automáticamente entre los nodos equivalentes.
- Basada en origen (contenido mixto) 91: estas conexiones asignan el contenido mixto (texto y nodos secundarios) en el mismo orden que el definido en el archivo XML de origen. Si selecciona **Asignar instrucciones de procesamiento** y/o **Asignar comentarios** podrá incluir estos grupos de datos en el archivo de salida (*imagen siguiente*).

 © 2016-2022 Altova GmbH



Configuración de la anotación

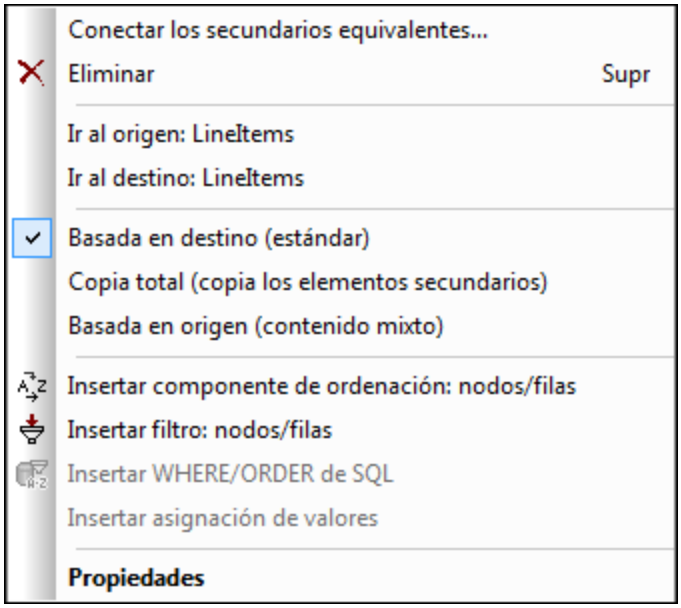
En esta sección puede aplicar etiquetas a las conexiones. Puede aplicar esta opción a todos los tipos de conexión. Para anotar una conexión siga estas instrucciones:

1. Haga clic con el botón derecho en la conexión y seleccione **Propiedades** del menú contextual. O, si lo prefiere, haga doble clic en la conexión.
2. Introduzca el nombre de la conexión seleccionada en el campo **Descripción**. Esto habilita todas las conexiones de la sección **Configuración de la anotación**.
3. Use el resto de grupos para definir las opciones **Ubicación**, **Alineación** y **Posición** de la etiqueta.
4. Pulse el botón de la barra de herramientas (**Ver anotaciones**). Si no ve este botón en la barra de herramientas puede activarlo en la barra de herramientas **Opciones de vista**.

Nota: si el botón **Ver anotaciones de la barra de herramientas** no está activo sigue pudiendo ver la anotación si pasa el cursor del ratón por encima de la conexión. Si el botón (Ver información rápida) de la barra de herramientas **Opciones de vista** está habilitado, la anotación aparece como información rápida.

3.2.3 Menú contextual de las conexiones

En este apartado explicamos los comandos que tiene disponibles en el menú contextual de las conexiones. Cuando se hace clic con el botón derecho en una conexión aparece este menú contextual:



Para más información siga leyendo.
Configuración general

- *Conectar los secundarios equivalentes:* Abre el cuadro de diálogo Conectar los secundarios equivalentes 94. Este comando se habilita si la conexión puede tener secundarios equivalentes.
- *Eliminar:* Elimina la conexión seleccionada.
- *Ir al origen: <nombre del elemento>:* Resalta el conector de salida 74 de la conexión seleccionada.
- *Ir al destino: <nombre del elemento>:* Resalta el conector de entrada 74 de la conexión seleccionada.

Tipos de conexión
Consulte los detalles sobre los tipos de conexión en Tipos de conexión 91 y Configuración de las conexiones 97.

Comandos de inserción

- *Insertar componente de ordenación: nodos/filas:* Agrega un componente de ordenación 505 entre un nodo de origen y uno de destino.
- *Insertar filtro: nodos/filas:* Agrega un componente de filtrado 543 entre un nodo de origen y uno de destino.
- *Insertar componente WHERE/ORDER de SQL/NoSQL:* Agrega un componente WHERE/ORDER de SQL/NoSQL entre un nodo de origen y uno de destino (*en las ediciones Professional y Enterprise*). Para más detalles consulte Filtrar y ordenar datos de BD (SQL WHERE/ORDER) 352.
- *Insertar asignación de valores:* Agrega una asignación de valores 533 entre un nodo de origen y uno de destino.

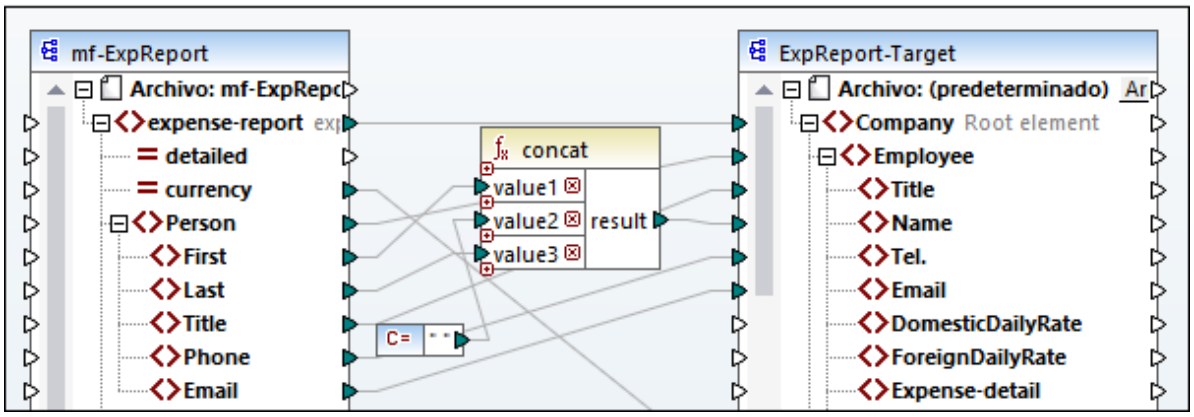
Propiedades
Abre el cuadro de diálogo Configuración de las conexiones 97.

 © 2016-2022 Altova GmbH

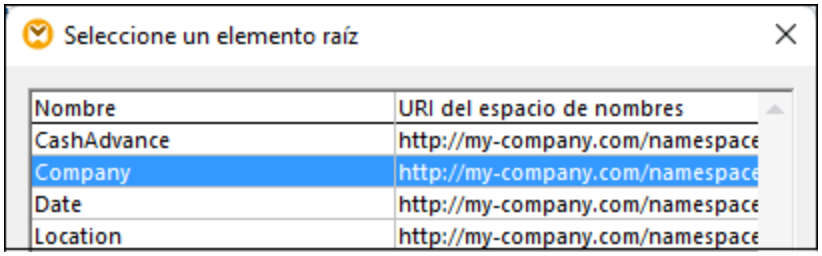
3.2.4 Mover conexiones y conexiones secundarias

En algunas situaciones puede que quiera cambiar el esquema de un componente de origen o de destino. Cambiar esquemas puede afectar a la validez de la asignación y resultar en conexiones defectuosas. En este apartado explicamos cómo reparar esas conexiones después de cambiar el archivo del esquema. Siga las instrucciones del ejemplo siguiente para entender mejor cómo proceder si encuentra conexiones defectuosas.

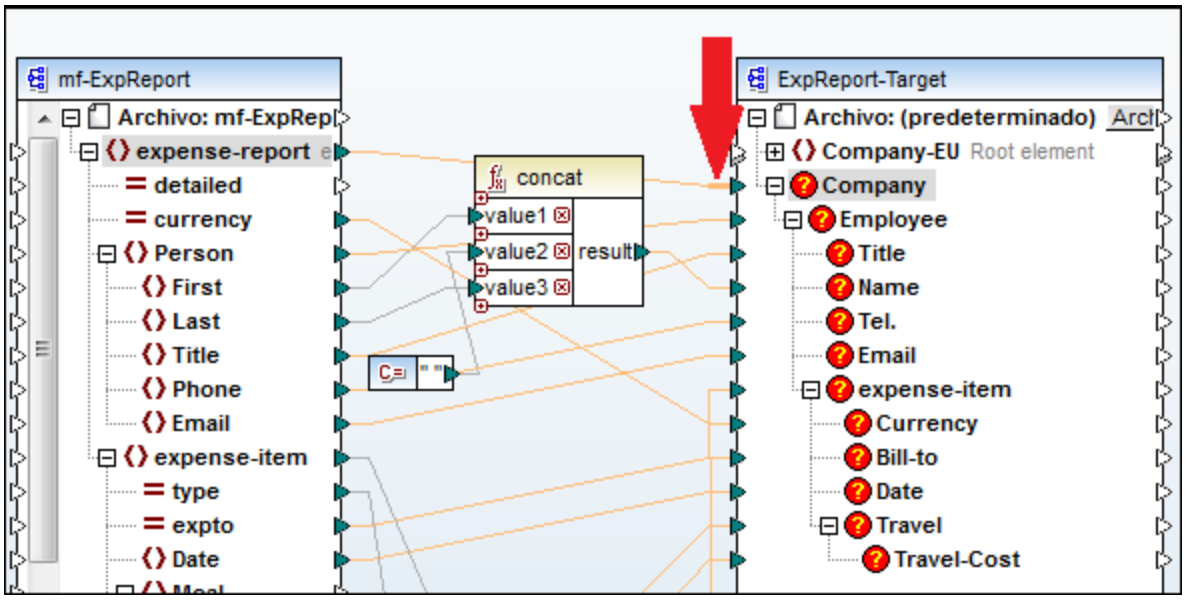
1. Abra la asignación **`Tut-ExpReport.mfd`**, que encontrará en la carpeta Tutorials 23. A continuación puede ver parte de esta.



2. Abra **`ExpReport-Target.xsd`** en un editor (p. ej. Altova XMLSpy) y cambie el elemento raíz `Company` en el esquema de destino a `Company-EU`. No es necesario que cierre MapForce.
3. Una vez haya editado el elemento raíz del esquema de destino MapForce le informará de que se han cambiado archivos. Haga clic en el botón **Volver a cargar**. Al haber cambiado el elemento raíz, en el componente hay varias conexiones defectuosas.
4. Haga clic en **Seleccionar un elemento raíz nuevo** en la parte superior del componente (*imagen siguiente*). También puede cambiar el elemento raíz haciendo clic con el botón derecho en el encabezado del componente y seleccionando **Cambiar elemento raíz** en el menú contextual.



5. Seleccione `Company-EU` como nuevo elemento raíz y haga clic en **Aceptar**. El elemento raíz `Company-EU` ahora se puede ver en la parte superior del componente.
6. Ahora tiene que mover la conexión del nodo `Company` al nuevo elemento raíz. Pulse y mantenga pulsada la parte gruesa de la conexión `Company` (*véase la flecha roja más abajo*). Después arrastre la conexión hasta el elemento raíz `Company-EU`.



Un cuadro de diálogo le preguntará si quiere mover todos los nodos secundarios equivalentes. Puede elegir entre mover solamente la conexión seleccionada o mover la conexión seleccionada y aquellos de sus nodos secundarios que tengan equivalentes en el elemento raíz nuevo. En nuestro ejemplo hemos elegido la opción **Incluir conexiones de descendientes**. En cuanto haga clic en este botón los nodos defectuosos desaparecerán del componente.

Nota:si el nodo al que está asignando los datos tiene el mismo nombre que el nodo de origen pero un espacio de nombres distinto, el cuadro de diálogo de la notificación incluirá el botón **Incluir descendientes y asignar espacio de nombres**. Si hace clic en este botón se mueven las conexiones secundarias del mismo espacio de nombres que el nodo primario de origen a los mismos nodos secundarios del espacio de nombres distinto.

Una solución alternativa

Una solución alternativa al problema que presentamos sería eliminar los nodos defectuosos que ya no necesita en la asignación. Por ejemplo, al eliminar la conexión entre la función **`concat`** y `Name`, el nodo `Name` desaparece del componente `ExpReport-Target`.

Conexiones defectuosas en bases de datos (en las ediciones Professional y Enterprise)

Si un componente de BD tiene conexiones defectuosas deberá cambiar la configuración de los componentes 79. Haga clic en el botón **Cambiar** del cuadro de diálogo **Configuración del componente** para seleccionar una BD distinta o cambiar las tablas del componente de BD. Todas las conexiones y datos de BD válidos/correctos se mantienen si selecciona una BD que tenga la misma estructura.

3.2.5 Conservar conexiones tras eliminación de componentes

MapForce permite conservar conexiones (secundarias) aunque se hayan eliminado varios componentes de transformación 75, como variables, filtros o componentes de ordenación, asignaciones de valores, componentes de entrada simples o componentes SQL/NoSQL-WHERE/ORDER. Las conexiones pueden ser

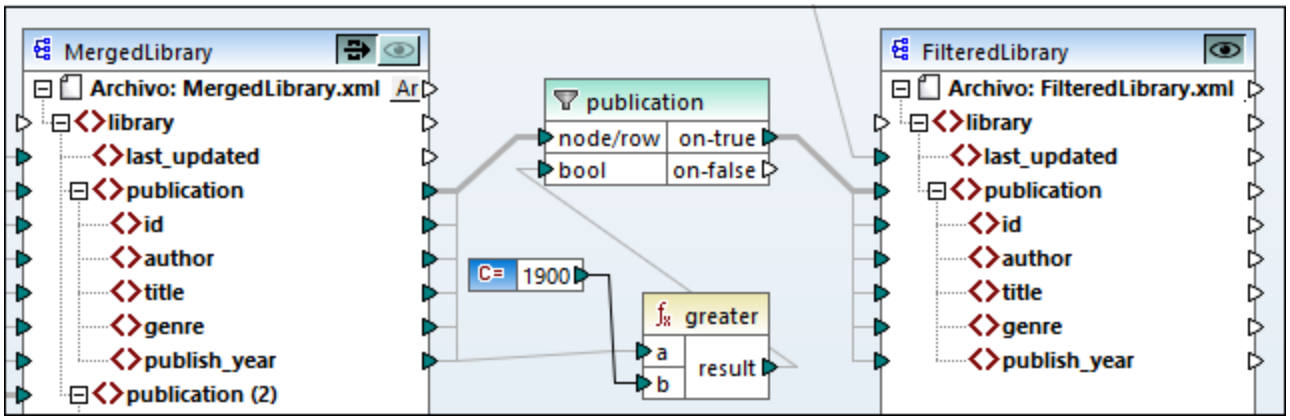
individuales o múltiples. Mantener conexiones puede ser especialmente útil si tiene varias conexiones secundarias, ya que no tiene que restaurar cada una de ellas manualmente después de eliminar el componente de transformación en cuestión. Para habilitar esta característica vaya a **Herramientas | Opciones | Edición** y seleccione **Eliminación inteligente de componentes** (conservar conexiones útiles). Esta opción está deshabilitada por defecto, es decir, si no la activa y elimina un componente de transformación también eliminará todas sus conexiones directas.

Ejemplo

Para ilustrar la eliminación inteligente de componentes hemos usado el archivo de muestra **`Tut3-ChainedMapping`**. Puede encontrarlo en la carpeta de tutoriales básicos 23:

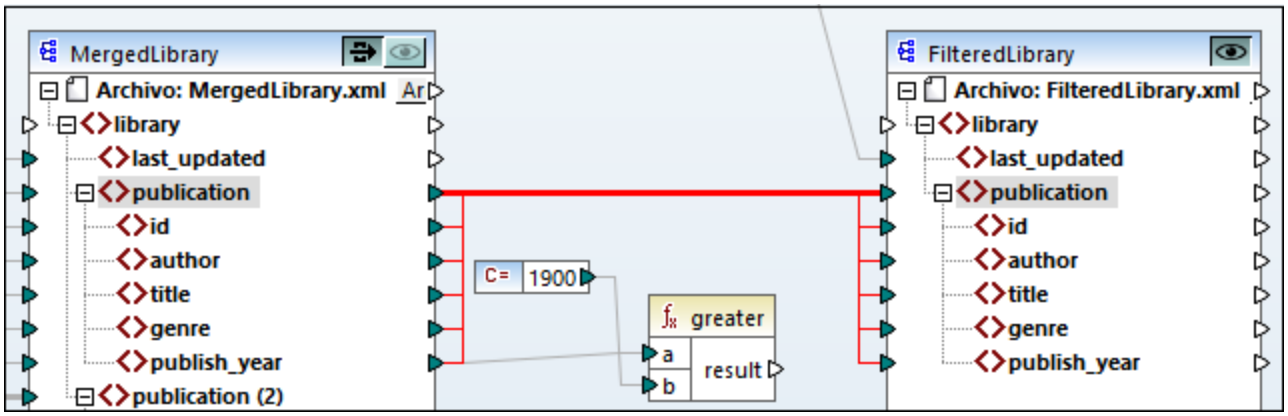
Antes de eliminar el componente

En la imagen siguiente se ve que existen conexiones de copia total 95 entre el componente `MergedLibrary` y el filtro `publication`, así como entre el filtro `publication` y el componente `FilteredLibrary`. Lo que queremos es eliminar el filtro `publication` pero mantener las conexiones de copia total. Para ello marque la casilla **Eliminación inteligente de componentes** del cuadro de diálogo **Opciones** (*imagen anterior*).



Después de eliminar el componente

Una vez haya eliminado la función `publication` se crea la conexión de copia total entre el nodo `publication` de `MergedLibrary` y el nodo `publication` de `FilteredLibrary` (*imagen siguiente*).



Nota: si un componente de filtrado tiene conectadas las dos salidas (`on-true` y `on-false`), entonces se conservan las conexiones secundarias de ambas salidas.

© 2016-2022 Altova GmbH

3.3 Tareas generales

Además de crear asignaciones, también puede validar asignaciones y sus resultados, generar código, usar la vista texto y configurar asignaciones. En esta sección encontrará:

- Validación 105
- Generación de código 107
- Características de la vista Texto 108
- Búsqueda en la vista Texto 111
- Configuración de la asignación 115

3.3.1 Validar la asignación de datos

En este apartado explicamos cómo validar asignaciones. También como acceder a una vista previa del resultado, guardarlo y validarlo.

Validar asignaciones de datos

MapForce valida las asignaciones automáticamente cuando se hace clic en el panel **Resultados**. También puede validarlas manualmente, lo que ayuda a identificar y corregir errores potenciales y advertencias antes de ejecutar la asignación. Para validar una asignación manualmente, haga clic en el panel **Asignación** y siga estos pasos:

- En el menú **Archivo** haga clic en **Validar asignación**.
- En la barra de herramientas haga clic en (**Validar**).

Para validar la asignación MapForce comprueba, entre otras muchas cosas, si faltan conexiones o si hay conexiones incorrectas, si hay tipos de componente incompatibles, etc. Para ver la referencia de todos los estados de validación consulte el apartado de la ventana Mensajes 34. La ventana Mensajes también permite ejecutar acciones relacionadas con los mensajes 34. Para ver los resultados de cada validación por separado, basta con hacer clic en las diferentes pestañas numeradas situadas en el lateral izquierdo de la ventana **Mensajes**. Esta característica es muy práctica cuando se trabaja con varios archivos de asignación simultáneamente.

Validación de los componentes de transformación
A continuación explicamos cómo se validan los componentes de transformación 76:

- Si algún **conector de entrada** obligatorio está desconectado, se genera un mensaje de error y se interrumpe la transformación.
- Si algún **conector de salida** está desconectado, se genera una advertencia y continúa el proceso de transformación. El componente infractor y sus datos se pasan por alto y los datos no se asignan al de salida.

Vista previa y validación de resultados

En MapForce puede incluso consultar una vista previa de los resultados sin necesidad de ejecutar ni compilar el código generado con un procesador o compilador externo. Por lo general, se recomienda consultar la vista previa de resultados de la transformación en MapForce antes de procesar el código generado en una aplicación

externa. Para generar la vista previa de los resultados MapForce ejecuta la asignación y rellena el panel Resultados 37.

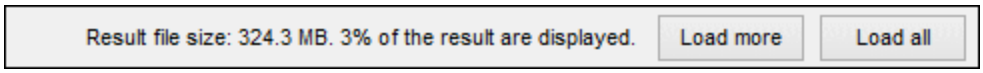
Una vez disponibles en el panel **Resultados**, los datos se pueden validar y guardar. Visite el apartado Validación 105 para obtener más información. También puede usar el comando **Buscar** (**Ctrl+F**) para encontrar un patrón de texto en el archivo de salida. Para más información consulte Búsqueda en la vista Texto 111. Los mensajes de error, advertencia e información relacionados con la ejecución de la asignación aparecen en la ventana Mensajes 34.

Si selecciona C++, C# o Java (*ediciones Professional y Enterprise*) como lenguajes de transformación 24, MapForce ejecuta la asignación con su motor de transformación integrado y muestra el resultado en el panel **Resultados**.

Para guardar el resultado de la asignación haga clic en el panel **Resultados** y siga estos pasos:

- Haciendo clic en el comando de menú **Resultados | Guardar el archivo de salida**.
- Haciendo clic en el botón (**Guardar resultado generado**) de la barra de herramientas.

Cargar opciones
A la hora de generar una previa de archivos de gran tamaño, MapForce limita la cantidad de datos que se presentan en el panel **Resultados**. En concreto se presenta solamente parte del archivo y aparece el botón **Cargar más** en la parte inferior del panel (*imagen siguiente*). Al hacer clic en **Cargar más**, se añaden más datos. La vista previa también se puede configurar en la pestaña **Generales** del cuadro de diálogo **Opciones**. Para más información consulte Cambiar opciones de MapForce 1049.

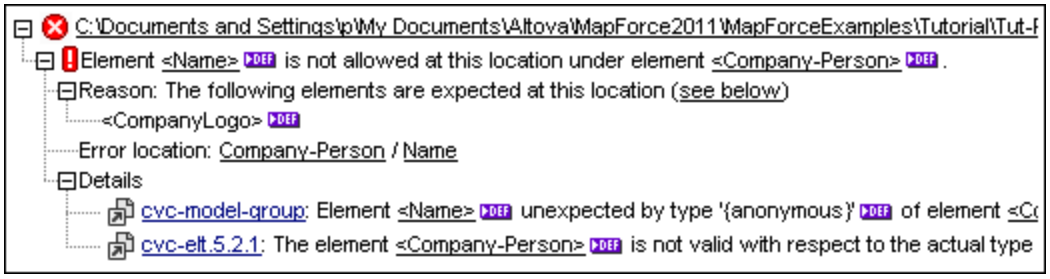


Validar resultados de la asignación
Al hacer clic en el panel **Resultados**, aparece una vista previa del resultado de la asignación, que puede validar usando el esquema que tenga asociado. El botón **Validar archivo de salida** y su correspondiente comando de menú (**Resultados | Validar el archivo de salida**) solamente se habilitan si el archivo de salida admite la validación con un esquema. Los resultados de la validación aparecen en la ventana **Mensajes**. Para validar el resultado tiene dos opciones:

- Abrir el panel **Resultados** y hacer clic en el botón (Validar archivo de salida) de la barra de herramientas.
- Abrir el panel **Resultados** y hacer clic en **Resultados | Validar el archivo de salida**.

La imagen siguiente ilustra una validación fallida. La ventana **Mensajes** contiene información detallada sobre los errores- Por ejemplo, si hace clic en el enlace `<Name>`, MapForce resaltará este elemento en el panel **Resultados**.

 © 2016-2022 Altova GmbH



3.3.2 Generar código XSLT

Puede generar código a partir de una asignación de datos en el lenguaje que haya seleccionado como lenguaje de transformación [24]. Tiene varias opciones:

- XSLT 1.0/XSLT 2.0/XSLT 3.0
- XQuery (*ediciones Professional y Enterprise*)
- Java (*ediciones Professional y Enterprise*)
- C# (*ediciones Professional y Enterprise*)
- C++ (*ediciones Professional y Enterprise*)

Ejemplo
Siga estos pasos para generar código XSLT:

1. Seleccione el comando de menú **Archivo | Generar código en | XSLT)**.
2. Seleccione la carpeta donde desea guardar el archivo XSLT generado y haga clic en **Aceptar**. MapForce genera el código y muestra el resultado de la operación en la ventana Mensajes. MapForce genera el código y muestra el resultado de la operación en la ventana Mensajes [34].

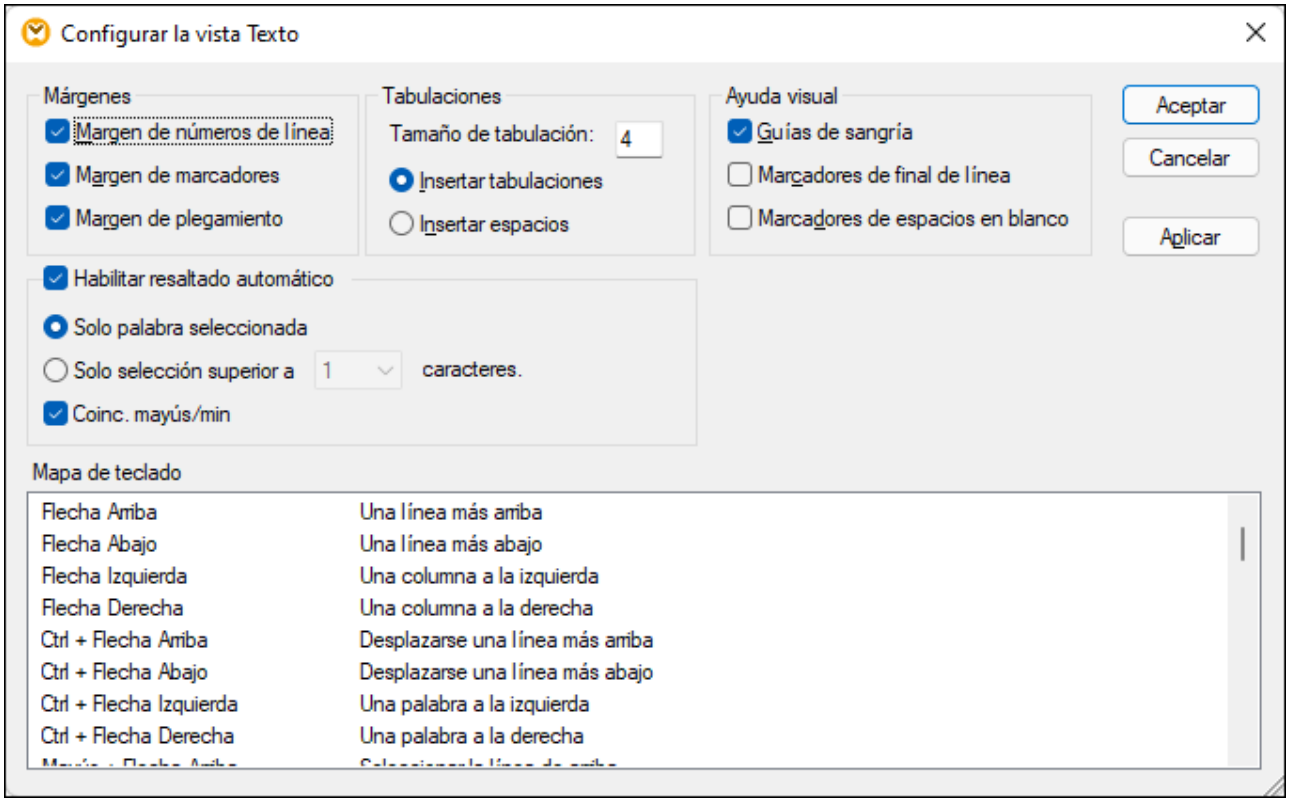
Para acceder a la vista previa del código XSLT generado haga clic en el botón **XSLT** que hay en la parte inferior del panel **Asignación**. Para el resto de lenguajes los pasos que debe seguir son idénticos.

Una vez haya generado el código la carpeta de destino contendrá estos dos archivos:

1. Un archivo XSLT de transformación con el mismo nombre que el esquema de destino. Este es el formato del archivo de transformación: **`<Mapping>MapTo<TargetFileName>.xslt`**. `<Mapping>` es el valor del campo **ApplicationName** en la configuración de la asignación [115]. `<TargetFileName>` es el nombre del componente de destino de la asignación. Para cambiar este valor abra la configuración del componente y edite el valor del campo **Nombre del componente**. Para más información consulte Cambiar configuración de los componentes [79] y Rutas de acceso según el entorno de ejecución [85].
2. Un archivo **`DoTransform.bat`**, que permite ejecutar la transformación XSLT con Altova RaptorXML Server desde la línea de comandos. Para ejecutar este comando necesita tener instalado RaptorXML.

3.3.3 Características de la vista Texto

Los paneles Resultados 37, XQuery 36 y XSLT ofrecen varias funciones visuales para facilitar la consulta y edición del texto. Puede personalizar estas funciones en el cuadro de diálogo Configurar la vista Texto (*imagen siguiente*). La configuración elegida en este cuadro de diálogo se aplica a toda la aplicación y no solo al documento activo.



Hay varias maneras de abrir el cuadro de diálogo **Configurar la vista Texto**:

- Con el comando de menú **Resultados | Configurar la vista Texto**.
- Con el botón (**Configurar la vista Texto**) de la barra de herramientas.
- Haciendo clic con el botón derecho en el panel **Resultados** y seleccionando **Configurar la vista Texto** en el menú contextual.

Algunas características visuales de la **vista Texto** también se pueden activar/desactivar desde la barra de herramientas Vista Texto, desde el menú de la aplicación o con teclas de acceso rápido. Para más información sobre las teclas de acceso rápido consulte el panel **Mapa de teclado** situado en la parte inferior del cuadro de diálogo **Configurar la vista Texto**.

A continuación puede consultar la lista de opciones disponibles.

⊟ Márgenes

margen de números de línea

Los números de línea aparecen en el margen de números de línea (imagen anterior), que se puede activar o desactivar en el cuadro de diálogo **Configurar la vista Texto**. Cuando se contrae una sección de texto, los números de línea del texto contraído también se ocultan.

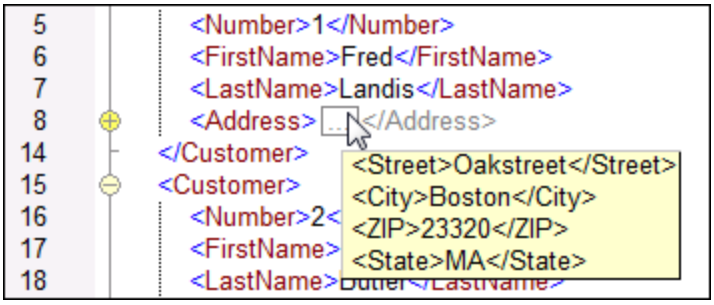
Margen de marcadores (cuadro de diálogo)
Las líneas del documento se pueden marcar para realizar consultas rápidas más adelante. Si el **margen de marcadores** está activado en en el cuadro de diálogo **Configurar la vista Texto**, los marcadores aparecen en el margen de marcadores (*imagen siguiente*). De lo contrario, las líneas marcadas se resaltan en color cian.

```
6      <LastName>Little</LastName>
7      <Address>
8        <Street>Long Way</Street>
9        <City>Los-Angeles</City>
10       <ZIP>34424</ZIP>
11       <State>CA</State>
12     </Address>
13   </Customer>
14   <LineItems>
15     <LineItem> ... </LineItem>
24     <LineItem> ... </LineItem>
33   </LineItems>
```

Puede editar marcadores y navegar por ellos con los comandos de la tabla que ve a continuación. Estos comandos también están disponibles en el menú **Resultados** y en el menú contextual que aparece cuando se hace clic con el botón derecho en los paneles **Resultados**, **XSLT** y **XQuery**.

	Insertar o quitar marcador (Ctrl + F2)
	Ir al siguiente marcador (F2)
	Ir al marcador anterior (Mayús + F2
	Eliminar todos los marcadores (Ctrl + Mayús + F2

Marcadores de plegamiento de código
La característica plegamiento de código permite expandir y contraer nodos. Los nodos que se pueden expandir/contraer se marcan en el margen de plegamiento. El margen puede activarse o desactivarse en el cuadro de diálogo **Configurar la vista Texto**. Para expandir y contraer porciones de código basta con hacer clic en los iconos + y - situados en el lateral izquierdo del panel. Las porciones de código contraído se señalan con puntos suspensivos (*imagen siguiente*). Para ver el código contraído debe pasar el cursor del ratón sobre los puntos suspensivos. Al hacerlo se abre un cuadro emergente que muestra una vista vista previa, como se ve en la imagen siguiente. No olvide que si la vista previa no cabe en el cuadro emergente, al final del código aparecen puntos suspensivos.



Habilitar el resaltado automático

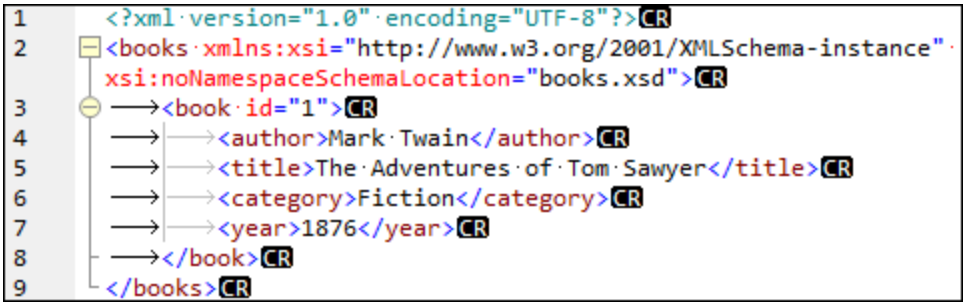
La opción **Habilitar el resaltado automático** permite visualizar todas las coincidencias del texto seleccionado. La selección se resalta en azul pálido y las coincidencias se resaltan en naranja pálido. La selección y sus coincidencias también se indican con cuadrados grises en la barra de desplazamiento, que también indica la posición actual del cursor por medio de un marcador azul. Puede definir como selección una palabra entera o un número fijo de caracteres. También puede especificar si se deben tener en cuenta o no las mayúsculas y minúsculas.

En el caso de la selección de caracteres, podrá especificar el número mínimo de caracteres que deben coincidir con la selección, empezando por el primer carácter de la selección. Por ejemplo, puede elegir que coincidan dos caracteres o más. Cuando se trate de búsquedas de palabras, la aplicación considera que estos elementos son palabras independientes: nombres de elementos (sin paréntesis angulares), paréntesis angulares de etiquetas de elementos, nombres de atributos y valores de atributos sin comillas.

Ayuda visual

Guías de sangría
Las guías de sangría son líneas verticales que indican la longitud de la sangría de una línea. Las guías de sangría se pueden activar o desactivar en el cuadro de diálogo **Configurar la vista Texto**. Las opciones **Insertar tabulaciones** e **Insertar espacios** tienen efecto cuando se ejecuta el comando **Resultados | Texto XML pretty-print**.

Marcadores de final de línea y marcadores de espacios en blanco
Los marcadores de final de línea y los marcadores de espacios en blanco (*imagen siguiente*) se pueden activar o desactivar en el cuadro de diálogo **Configurar la vista Texto**. Cada flecha representa un carácter de tabulación, *CR* representa un retorno de carro y cada punto representa un espacio en blanco.



Más opciones de configuración de la vista Texto

 © 2016-2022 Altova GmbH

Color de sintaxis
El color de sintaxis es otra ayuda visual que permite leer código más cómodamente. El color de sintaxis se aplica en función del valor semántico del texto. Por ejemplo, en los documentos XML, dependiendo de si el nodo XML es un elemento, un atributo, contenido, una sección CDATA, un comentario o una instrucción de procesamiento, el nombre del nodo (y en ocasiones el contenido del nodo) tendrá un color diferente.

Alejarse y acercarse con el zoom
Puede acercarse y alejarse en la vista Texto si mueve la rueda del ratón mientras pulsa la tecla **Ctrl**. También puede usar las teclas - o + mientras pulsa la tecla **Ctrl**.

Formato pretty-print
El comando **Texto XML pretty-print** ajusta el formato del documento XML que está activo en la **vista Texto** para representarlo de forma estructurada. Por defecto, cada nodo secundario se presenta alejado de su elemento primario por cuatro caracteres de espacio. Las guías de sangría se pueden activar o desactivar en el cuadro de diálogo **Configurar la vista Texto**. Para aplicar formato pretty-print a un documento XML seleccione el comando de menú **Resultados | Texto XML pretty-print** o haga clic en el botón (**Pretty-print**) de la barra de herramientas.

Ajuste automático de línea
El ajuste automático de línea ayuda a visualizar fragmentos de código dentro del margen del área o la trabajo. Si se deshabilita, hay partes del texto que no son visibles en el área de trabajo. Para activar el ajuste automático de línea en el documento activo use el comando Resultados | Ajuste automático de línea haga clic en el botón (**Ajuste automático de línea**) de la barra de herramientas.

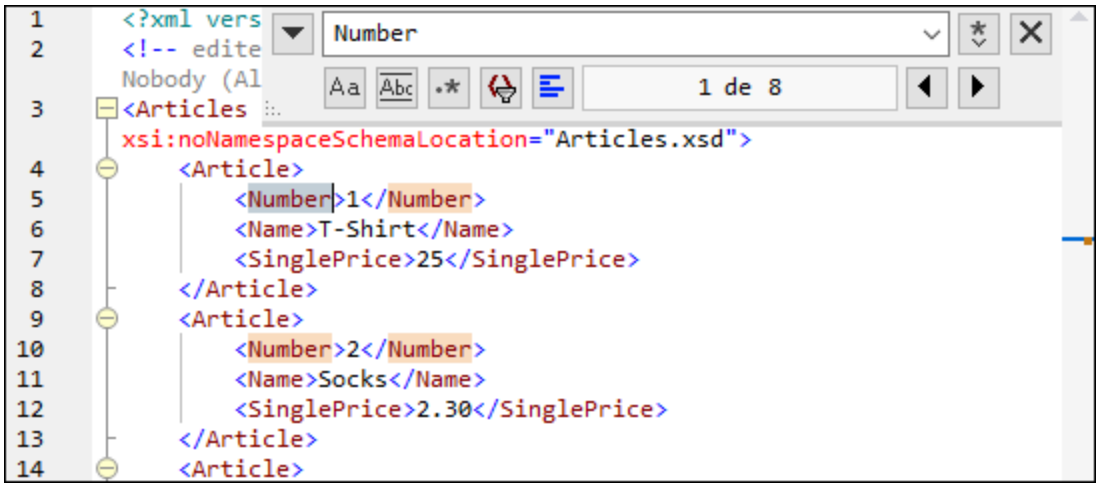
3.3.4 Búsquedas en la vista Texto

Puede realizar búsquedas en el texto de los paneles **Resultados**, XQuery y XSLT con varias opciones y ayudas visuales.

Para iniciar una búsqueda pulse **Ctrl+F** (o seleccione el comando de menú **Edición | Buscar**). La búsqueda puede realizarse en todo el documento o dentro de una selección de texto.

Puede introducir una cadena de texto o seleccionar una de las últimas diez búsquedas en el cuadro combinado. Cuando introduzca o seleccione la cadena de búsqueda, todas las coincidencias se resaltarán y las posiciones de los resultados se indican en la barra de desplazamiento por medio de marcadores color beige. El resultado que esté seleccionado se resalta en gris.

El resultado que esté seleccionado se resalta en un color distinto al de los demás resultados y su posición se indica en la barra de desplazamiento por medio de un marcador de cursor azul oscuro. El número total de resultados aparece debajo del término de búsqueda, además de la posición de índice del resultado que esté seleccionado. Puede recorrer los resultados en ambos sentidos con los botones **Anterior** **(Mayús+F3)** y **Siguiente** **(F3)** situados en la esquina inferior derecha.



Para realizar búsquedas dentro de una selección: (i) marque la selección, (ii) active la opción **Buscar en la selección** para bloquear la selección e (iii) introduzca el término de búsqueda. Para buscar dentro de otra selección, desbloquee la selección actual desactivando la opción **Buscar en la selección**, marque una nueva selección y active otra vez la opción **Buscar en la selección**.

Opciones de búsqueda

Los criterios de búsqueda pueden configurarse con los botones situados debajo del campo del término de búsqueda. Si una opción está activada, su botón aparece en color azul. Estas son las opciones de búsqueda disponibles:

Opción	Icono	Descripción
Coinc. mayús/min	Aa	La búsqueda tiene en cuenta las mayúsculas y minúsculas a la hora de realizar la búsqueda (`Address` no es lo mismo que `address`).
Sólo palabras completas	Abc	Solo se consideran coincidencias las palabras completas.
Expresión regular	.*	Si activa esta opción, el término de búsqueda se lee como expresión regular (*véase más abajo*).
Buscar delimitador	I	Cuando se introduce un término de búsqueda, los resultados de la búsqueda se resaltan y uno de ellos se marcará como selección actual. Con la opción **Buscar delimitador** puede definir si esta primera selección se hace en relación a la posición actual del cursor o no. Es decir, si la opción **Buscar delimitador** está activada, entonces el primer resultado seleccionado será el siguiente resultado a partir de la posición actual del cursor. Por el contrario, si la opción **Buscar delimitador** está desactivada, el primer resultado seleccionado será el primer resultado del documento, empezando desde el principio.
Buscar en la selección		Si activa esta opción, la selección actual se bloquea y la búsqueda se ejecuta en la selección solamente. De lo contrario, la búsqueda se ejecuta en todo el documento. Antes de realizar una selección nueva deberá

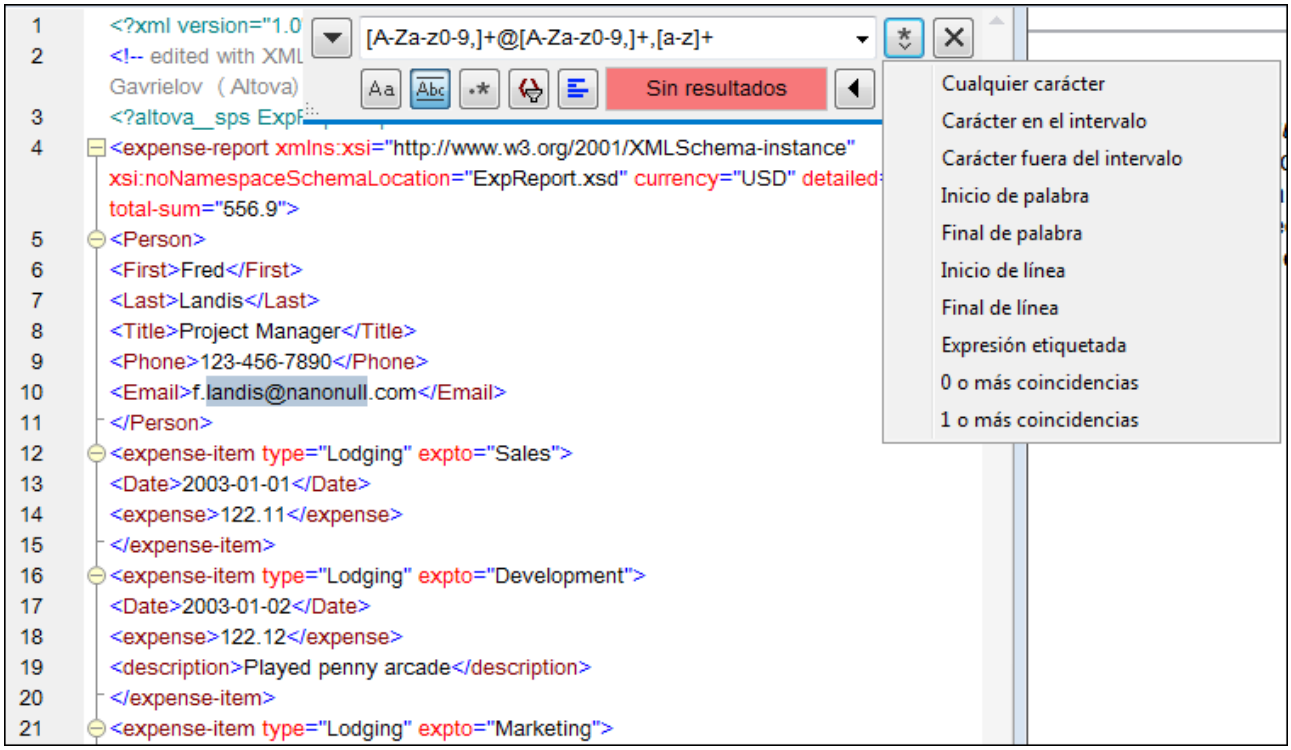
 © 2016-2022 Altova GmbH

		desbloquear la selección actual desactivando el botón de la opción **Buscar en la selección**.

Uso de expresiones regulares

Puede usar expresiones regulares (regex) para buscar cadenas de texto en el documento. Para ello lo primero es activar la opción **Expresión regular** .*. Al activar esta opción estamos especificando que el texto del campo del término de búsqueda debe evaluarse como expresión regular. El segundo paso consiste en introducir la expresión regular en el campo de búsqueda. Si necesita ayuda para construir su expresión regular, haga clic en el botón **Generador de expresiones regulares** (situado a la derecha del campo de búsqueda). Seleccione un elemento de la lista desplegable para introducir los caracteres correspondientes en el campo de búsqueda.

A continuación puede ver un ejemplo de expresión regular que se utiliza para buscar direcciones de correo electrónico.



Expresiones regulares personalizadas

A continuación puede ver una lista de metacaracteres de expresión regular compatibles con la función de búsqueda y reemplazo.

.	Cualquier carácter. Es un comodín para un solo carácter.
(abc)	Los metacaracteres (y) marcan el inicio y el final de una expresión regular. Las expresiones regulares pueden serle de utilidad a la hora de etiquetar (es decir, recordar)

	una región concreta del resultado de la búsqueda y poder hacerle referencia más adelante (referencia inversa). Puede etiquetar (y hacer referencia inversa a) un máximo de 9 subexpresiones. Por ejemplo, **`(the) \1`** encuentra la cadena `the the`. Esta expresión significa literalmente: buscar la cadena "the" (y recordarla como región etiquetada), seguida de un espacio, seguida de una referencia inversa a la región etiquetada encontrada previamente.
`\n`	Siendo `n` un número del 1 al 9, `n` hace referencia a la correspondiente región etiquetada (*ver fila anterior*).
`\x`	Permite usar caracteres que de lo contrario tendrían un significado propio. Por ejemplo, `\[` se interpretaría como `[` y no como el inicio de un conjunto de caracteres.
`\<`	Inicio de palabra.
`\>`	Final de palabra.
`\`	Inserta un carácter de escape al carácter que aparece después de la barra diagonal inversa. En otras palabras, la expresión `\x` permite usar el carácter `\` de forma literal. Por ejemplo, `\[` se interpretaría como `[` y no como el principio de un conjunto de caracteres.
`[...]`	Encuentra cualquiera de los caracteres del conjunto. Por ejemplo, **`[abc]`**]encuentra los caracteres `a`, `b` o `c`. También puede usar intervalos como **`[a-z]`** para buscar cualquier carácter en minúsculas.
`[^...]`	Encuentra cualquier carácter que no esté en este conjunto. Por ejemplo, **`[^A-Za-z]`** encuentra cualquier carácter excepto caracteres alfabéticos en mayúsculas o minúsculas.
`^`	Encuentra el inicio de línea (a no ser que se use dentro de un conjunto de caracteres, ver fila anterior).
`$`	Encuentra el final de línea. Por ejemplo, **`A+$`** encuentra una A o más de una A que estén al final de una línea.
`*`	Encuentra cero o más instancias de la expresión precedente. Por ejemplo, **`Sa*m`** encuentra `Sm` , `Sam` , `Saam` , `Saaam` , etc.
`+`	Encuentra una o más instancias de la expresión precedente. Por ejemplo **`Sa+m`** encuentra `Sam` , `Saam` , `Saaam` , etc

Buscar caracteres especiales

Puede buscar cualquiera de los caracteres especiales siguientes dentro del texto si habilita la opción **Usar expresiones regulares**:

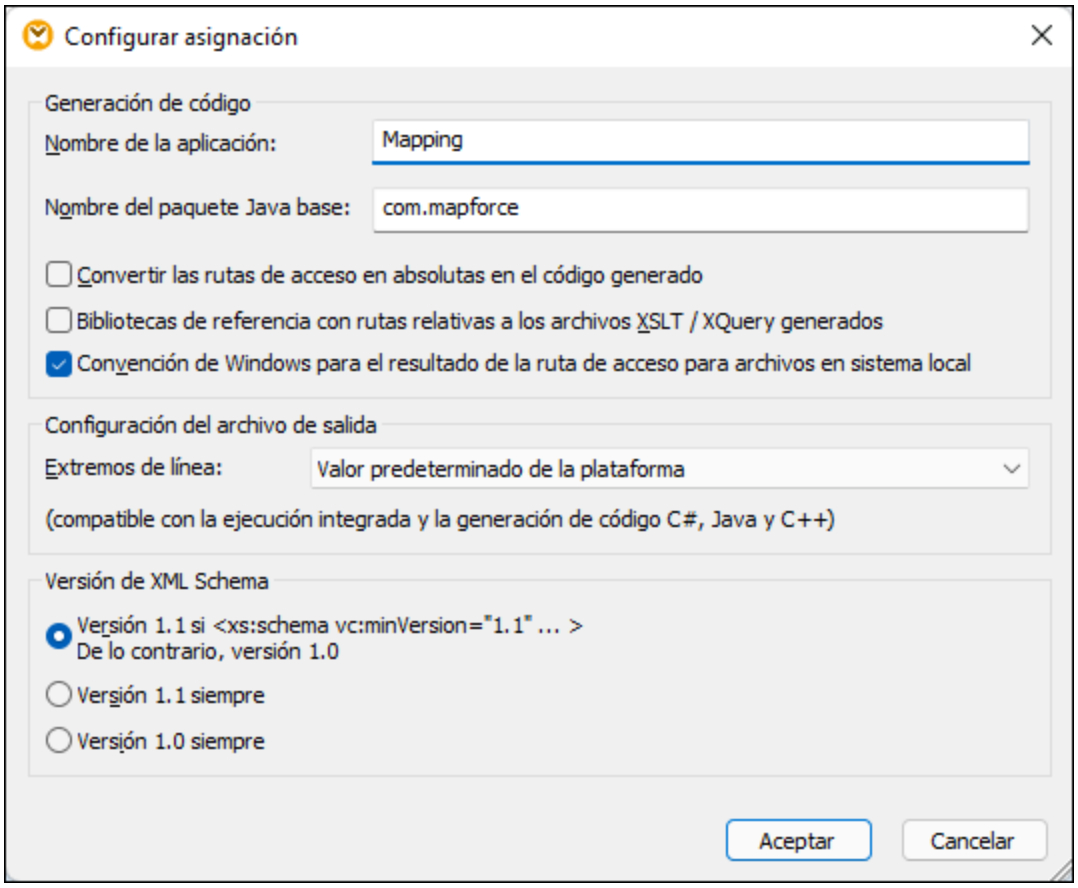
- `\t` (tabulador)
- `\r` (retorno de carro)
- `\n` (línea nueva)
- `\\` (barra inversa invertida)

Por ejemplo, para encontrar un caracter de tabulador pulse **Ctrl + F**, seleccione la opción **Usar expresiones regulares** e introduzca **`\t`** en la caja de diálogo **Buscar**.

 © 2016-2022 Altova GmbH

3.3.5 Cambiar configuración de la asignación

El cuadro de diálogo **Configuración de la asignación** (*imagen siguiente*) permite definir las opciones de los documentos. Para abrirlo, vaya al menú **Archivo** y haga clic en **Configurar asignación**. También puede hacer clic con el botón derecho en un área vacía del panel de asignación y seleccionar **Configurar asignación** en el menú contextual.



Estas son las opciones que se pueden configurar en este cuadro de diálogo:

- Generación de código
  - *Nombre de la aplicación:* define el prefijo de nombre del archivo XSLT o el nombre de aplicación Java, C# o C++ para los archivos de transformación generados (*solo disponible en las ediciones MapForce Professional y MapForce Enterprise*).

- *Nombre del paquete Java base (solo disponible en las ediciones MapForce Professional y MapForce Enterprise):* esta opción es relevante si escoge Java como lenguajes de transformación. Define el nombre del paquete base para los resultados Java.

- Convertir las rutas de acceso en absolutas en el código generado: esta casilla afecta a todas las rutas de los componentes de asignación, excepto a las rutas de acceso a los archivos de las bibliotecas externas (como las bibliotecas XSLT). Define si las rutas de acceso de los archivos deben ser relativas o absolutas en el código de programa generado y en los archivos de ejecución de MapForce Server (`.mfx`) y funciones de asignación implementadas en FlowForce Server. Consulte Rutas de acceso según el entorno de ejecución 84 para obtener más información.

- *Hacer referencia a bibliotecas con rutas relativas a los archivos XSLT/XQuery generados:* esta casilla se puede usar si el lenguaje de la asignación es XQuery (*solo disponible en las ediciones MapForce Professional y MapForce Enterprise*) o XSLT. Esta opción es útil si una asignación hace referencia a una biblioteca XSLT o XQuery y quiere generar archivos XSLT o XQuery a partir de esa asignación. Marque esta casilla si quiere que las rutas de la biblioteca sean relativas al directorio del código XSLT o XQuery generado. Si no la marca, las rutas de las bibliotecas serán generadas cuando genere el código. Consulte también Rutas de las bibliotecas en el código generado 85.

- *Convención de Windows para el resultado de la ruta de acceso para archivos en sistema local:* esta casilla se puede usar si el lenguaje de la asignación es XQuery (*solo disponible en las ediciones MapForce Professional y MapForce Enterprise*), XSLT 2.0 o XSLT 3.0. Marque esta casilla para que MapForce siga la convención de Windows para rutas de acceso. Cuando genere código XSLT 2.0, XSLT 3.0 o XQuery, el nombre del archivo procesado se recupera internamente usando la función **`document-uri`** que devuelve una ruta con el formato `archivo:// URI` para lo archivos locales. Si marca esta casilla, la especificación `archivo:// ruta URI` se convierte automáticamente a una ruta de archivo Windows completa (p. ej. `C:\...`) y así poder seguir con el procesamiento.

Configuración de los archivos de salida (solo disponible en MapForce Professional y MapForce Enterprise Edition)

En el cuadro combinado **Extremos de línea** puede especificar el extremo de las líneas de los archivos de salida. La opción *Valor predeterminado de la plataforma* se refiere a la opción predeterminada para el sistema operativo de destino (p. ej. CR+LF para Windows, LF para Mac OS X o LF para Linux). También puede seleccionar el extremo de línea que prefiera. Las opciones seleccionadas aquí son de una importancia fundamental para la implementación de archivos de ejecución de MapForce Server (`.mfx`) en servidores FlowForce Server que se ejecuten en sistemas operativos distintos.

Versión de XML Schema

En este grupo de opciones puede indicar qué versión de XML Schema se usa en el archivo de asignación. Sin embargo, tenga en cuenta que no todas las características propias de la versión 1.1 son compatibles con MapForce. Si la declaración `xs:schema vc:minVersion="1.1"` está presente en el esquema, se usa la versión 1.1. De lo contrario se usa la versión 1.0.

Si el documento XSD no tiene el atributo `vc:minVersion` o el valor de `vc:minVersion` no es `1.0` ni `1.1`, entonces el modo predeterminado será XSD 1.0. No se debe confundir el atributo `vc:minVersion` con el atributo `xsd:version`. El primero almacena el número de versión XSD, mientras que el segundo almacena el número de versión del documento. Cuando se cambia esta opción de configuración en una

 © 2016-2022 Altova GmbH

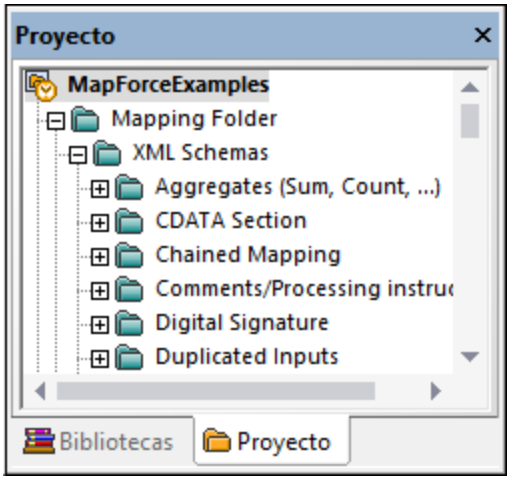
asignación, todos los esquemas que tengan la versión de esquema XML seleccionada se volverán a cargar y puede que la validez de la asignación se vea afectada.

- Configuración de la operación de servicio web (solo disponible en la edición MapForce Enterprise)

  Los campos **Definiciones WSDL**, **Servicio**, **Extremo** y **Operación** se rellenan automáticamente si el documento de asignación forma parte de una implementación de servicio web.

3.4 Trabajar con proyectos de asignación de datos

En MapForce también puede crear proyectos de asignación de datos formados por varios diseños de asignación. Todas las asignaciones de datos que añada a un proyecto estarán disponibles en la ventana **Proyecto** de la aplicación (*imagen siguiente*).



La principal ventaja de trabajar con un proyecto es que podrá definir una configuración de generación de código común (como el lenguaje de destino y el directorio de salida) para todas las asignaciones que formen parte del proyecto. También se pueden crear carpetas de proyecto y definir una configuración de código especial para cada carpeta del proyecto. Para más información sobre la función de generación de código de programa de MapForce (en C++, C# y Java), consulte la sección Generador de código 1067. Los archivos de proyecto de MapForce se guardan con la extensión **.mfp**.

En MapForce Enterprise Edition también puede crear proyectos de servicio web. Con este tipo de proyectos podrá generar código de programa Java o C# que implemente servicios web SOAP basados en archivos WSDL.

3.4.1 Proyectos nuevos

A continuación explicamos cómo empezar a trabajar con proyectos. Los procedimientos relacionados con proyectos se dividen a grandes rasgos en: (i) crear el proyecto, (ii) organizarlo y (iii) ejecutar acciones diversas. Consulte la información sobre estos procedimientos en las subsecciones siguientes.

Proyecto nuevo

Para crear un proyecto nuevo siga estos pasos:

1. Haga clic en el botón de la barra de herramientas. Si lo prefiere, vaya al menú **Archivo** haga clic en el comando **Nuevo**.
2. Seleccione **Archivo de proyecto** y haga clic en **Aceptar**.
3. En el cuadro de diálogo **Guardar proyecto como** introduzca el nombre del proyecto y haga clic en el botón **Guardar**. El proyecto nuevo aparece ahora en la ventana **Proyecto**.

 © 2016-2022 Altova GmbH

Para cerrar un proyecto vaya al menú **Proyecto** y haga clic en **Cerrar proyecto**.

Organización del proyecto

Agregar asignaciones a un proyecto
Si quiere agregar una asignación abierta a un proyecto, tiene dos opciones:

- En el menú **Proyecto** haga clic en el comando **Agregar archivo activo al proyecto**.
- Haga clic con el botón derecho en el nombre del proyecto en la ventana **Proyecto** y eligiendo el comando de menú **Agregar archivo activo al proyecto** del menú contextual.

También hay dos maneras de agregar archivos de asignación a un proyecto:

- En el menú **Proyecto** haga clic en el comando **Agregar archivos al proyecto**.
- Haga clic con el botón derecho en el nombre del proyecto en la ventana **Proyecto** y seleccione **Agregar archivos al proyecto**.

Consejo: si quiere agregar varios archivos mantenga pulsada la tecla **Ctrl** mientras selecciona los archivos que quiere agregar en el cuadro de diálogo **Abrir**.

Eliminar un archivo del proyecto
Para quitar un archivo o una carpeta de un proyecto siga estos pasos:

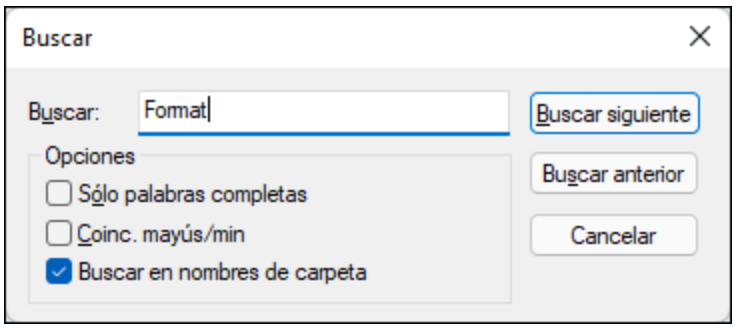
- Elija el archivo que quiere eliminar en la ventana **Proyecto**. Haga clic con el botón derecho en el archivo y elija Eliminar en el menú contextual.
- Seleccione el archivo en cuestión en la ventana **Proyecto** y pulse **Eliminar**.

La extensión de los archivos de proyecto de MapForce es **`*.mfp`**. Puede abrir proyectos de MapForce igual que lo hace con las asignaciones: En el menú Archivo haga clic en el comando Abrir. Por defecto, al ejecutar MapForce por primera vez verá el proyecto **`MapForceExamples.mfp`** en la ventana **Proyecto**.

Acciones relacionadas con el proyecto

Buscar un proyecto
Para buscar archivos en un proyecto siga estos pasos:

1. En la ventana **Proyecto** haga clic en el proyecto o la carpeta donde quiere realizar la búsqueda.
2. Pulse **Ctrl+F**. Se abre el cuadro de diálogo **Buscar**, donde puede seleccionar distintas opciones. Por ejemplo, si quiere incluir los nombres de las carpetas en la búsqueda seleccione la opción **Buscar en nombres de carpeta** (*imagen siguiente*).

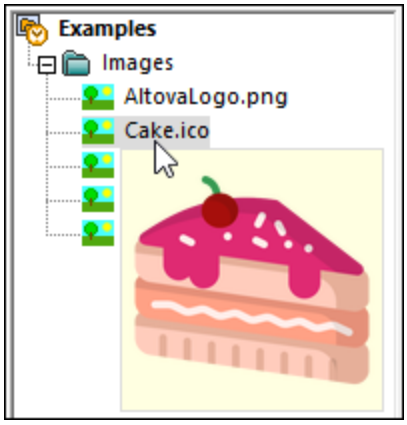


Generar código para el proyecto
En los proyectos se puede generar código para (i) asignaciones individuales, (ii) una carpeta en concreto o (iii) todo el proyecto.

Para generar código para una asignación o carpeta de un proyecto, haga clic con el botón derecho en ella y seleccione **Generar código** o **Generar código en**. Si selecciona **Generar código**, el código se genera en el lenguaje indicado en la configuración del proyecto 120. También puede generar código en uno de los lenguajes disponibles en su edición de MapForce. Para más información consulte Generación de código 107.

Para generar código para todo el proyecto, vaya al menú **Proyecto** y seleccione **Generar código para todo el proyecto**. También puede hacer clic con el botón derecho en la ventana **Proyecto** y seleccionar **Generar código**. El código se genera en el lenguaje indicado en la configuración del proyecto 120. También puede seleccionar el lenguaje en el que quiere generar código en el menú **Proyecto** o en el menú contextual del proyecto. Los lenguajes que puede usar dependen de la versión de MapForce que use. Para más información consulte Generación de código 107.

Vista previa de imágenes
En la ventana **Proyecto** puede generar una vista previa de las imágenes en varios formatos: **`.png`**, **`.jpeg`**, **`.gif`**, **`.bmp`**, **`.tiff`** y **`.ico`** (*imagen siguiente*). Haga doble clic en un archivo de imagen para abrirlo en una aplicación externa, que dependerá de la asociación de archivos de Windows.

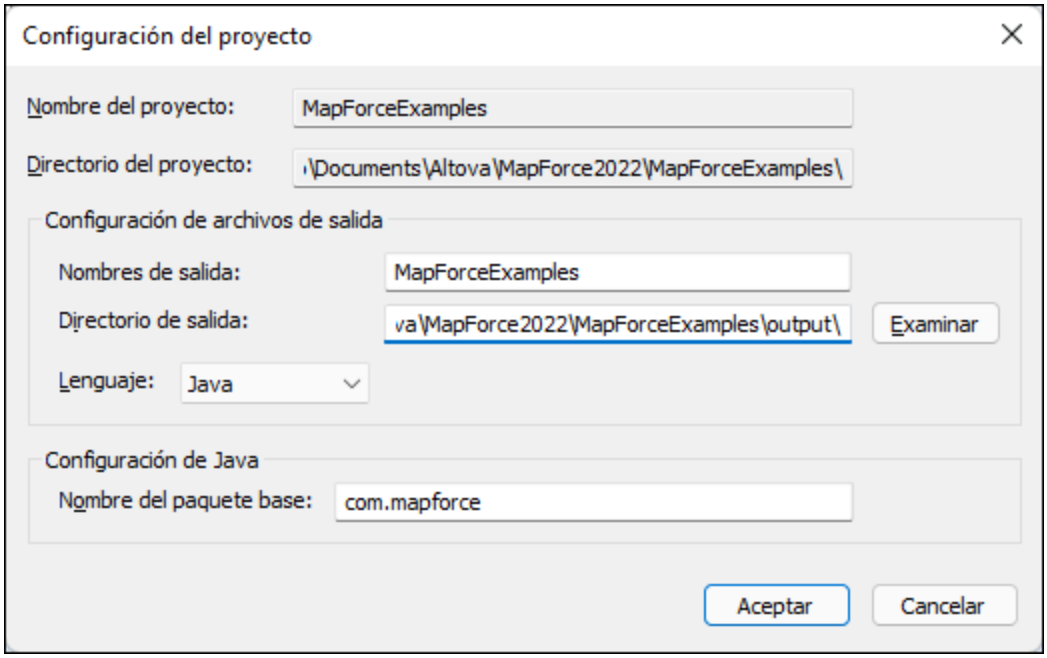


3.4.2 Configurar proyectos

Puede elegir una configuración de generación de código para cada proyecto y dicha configuración se aplicará a todas las asignaciones de datos del proyecto. Hay dos maneras de abrir el cuadro de diálogo **Configuración del proyecto** (*imagen siguiente*):

- Haciendo clic con el botón derecho en el nombre del proyecto en la ventana **Proyecto** y eligiendo el comando de menú **Propiedades** del menú contextual.
- Haciendo clic en el comando **Propiedades** del menú **Proyecto**.

 © 2016-2022 Altova GmbH



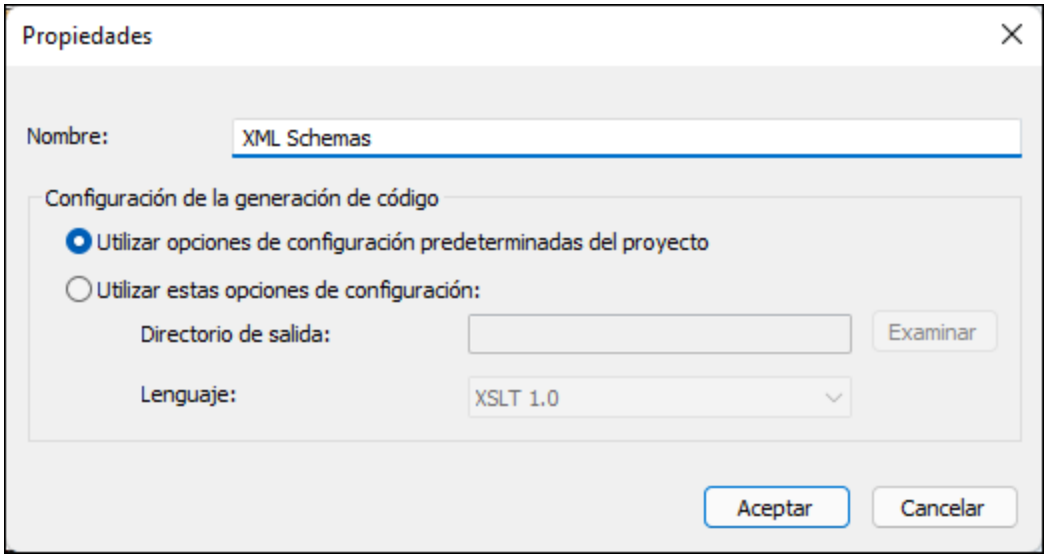
A continuación mostramos las opciones disponibles. (recuerde que el nombre y el directorio del proyecto no se pueden cambiar después de crear el proyecto):

- *Nombre de salida:* Nombre del proyecto o solución que se genera. También será el nombre de los objetos que se creen en el código generado.
- *Directorio de salida:* Carpeta de Windows donde se guardará el código generado a partir de todas las asignaciones del proyecto. El directorio de salida predeterminado es el directorio **`MapForceExamples\output\`**.
- *Lenguaje:* Define el lenguaje de generación de código que se debe utilizar para todos los archivos de asignación que están en el proyecto.
- *Nombre del paquete base:* Esta opción solo es relevante si seleccionó el lenguaje Java. Define el nombre del paquete base en el proyecto de Java generado.

3.4.3 Carpetas de proyecto

Con MapForce puede organizar las asignaciones de un proyecto por carpetas. Puede crear tantas carpetas como necesite y guardas en ellas las asignaciones. Estas carpetas de proyecto son en realidad carpetas virtuales, relevantes solamente dentro del proyecto de MapForce. Es decir, no se corresponden con ninguna carpeta del sistema operativo. Una de las ventajas de crear carpetas de proyecto es que podrá definir una configuración de generación de código común para todos los archivos de asignación de cada carpeta. Para crear una carpeta dentro de un proyectos de MapForce siga estos pasos:

1. En el menú **Proyecto** haga clic en el comando **Crear carpeta**. También puede hacer clic con el botón derecho en la ventana **Proyecto** y seleccionar **Crear carpeta**.
2. En el cuadro de diálogo **Propiedades** (*imagen siguiente*), defina la configuración de generación de código y haga clic en **Aceptar**.



A continuación puede ver las opciones de configuración que ofrece el cuadro de diálogo **Propiedades** de una carpeta de proyecto.

- *Nombre:* Nombre de la carpeta de proyecto.
- *Utilizar opciones de configuración predeterminadas del proyecto:* Esta es la opción predeterminada y significa que la configuración de generación de código de la carpeta actual es idéntica a la del proyecto entero. Por tanto, cuando genere código a partir del proyecto, MapForce usará la configuración de generación de código 120 definida a nivel de proyecto y no la de la carpeta. Si necesita una configuración de generación de código especial para la carpeta, seleccione la opción **Utilizar estas opciones de configuración** y especifique el directorio y el lenguaje de salida.
- *Directorio de salida:* Esta es la carpeta donde se guardará el código generado (a partir de todas las asignaciones que están en la carpeta de proyecto).
- *Idioma:* Define el lenguaje de generación de código que se debe utilizar para todos los archivos de El asignación que están en la carpeta de proyecto.

 © 2016-2022 Altova GmbH

4 Componentes estructurales

En esta sección explicamos los distintos formatos de datos con los que puede crear asignaciones:

- XML y XML Schema 124
- Bases de datos 159
- CSV y archivos de texto 445
- Archivos binarios 471

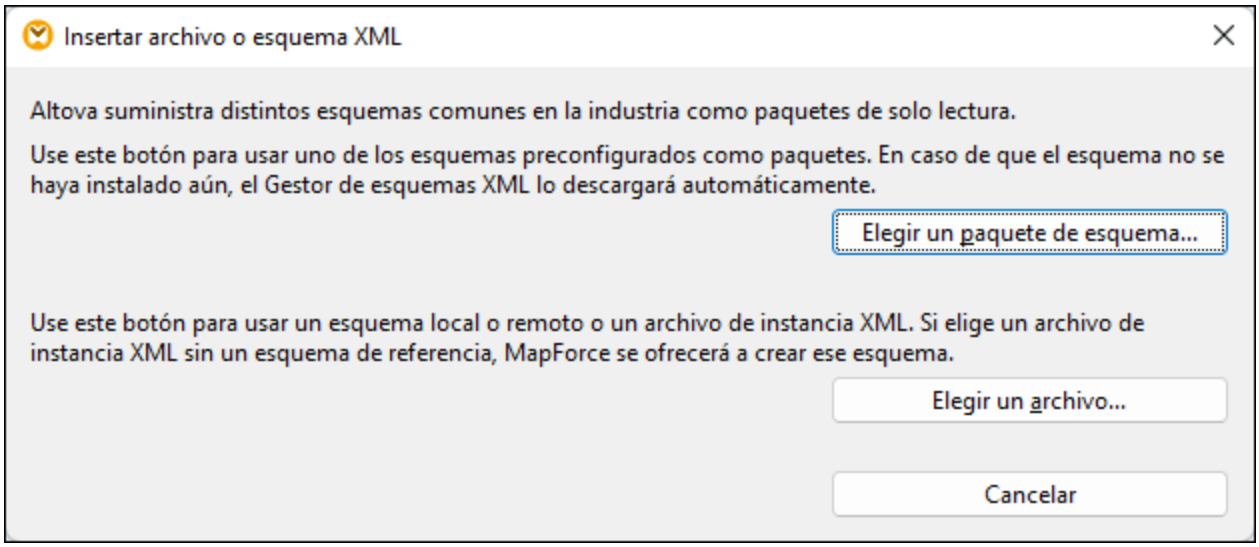
4.1 XML y esquemas XML

Sitio web de Altova: Asignación XML

En este apartado se explican los distintos aspectos de los componentes XML. Para más información sobre casos básicos de transformación de datos consulte los Tutoriales 39.

Insertar un archivo o esquema XML

Para insertar un archivo o esquema XML seleccione el comando de menú **Insertar | Archivo o esquema XML** o use el botón de la barra de herramientas. El cuadro de diálogo (*imagen siguiente*) le pedirá que elija entre un paquete de esquema estándar en el sector y un archivo de esquema o instancia remoto o local. Si elige el paquete de esquema deberá indicar un punto de entrada. Si el esquema que quiere usar todavía no está instalado, el Gestor de esquemas XML 143 lo descargará automáticamente.



Generar un esquema XML

Al agregar un archivo XML local o remoto sin referencia a un esquema, MapForce sugiere generar uno por usted. Si quiere que lo genere deberá seleccionar el directorio en el que quiere guardar el esquema generado.

Cuando MapForce genera un esquema a partir de un archivo XML, los tipos de datos para elementos y atributos deben inferirse del documento de instancia XML y puede que el resultado no sea exactamente lo que se esperaba. Por eso, recomendamos que terminada la operación compruebe si el esquema generado representa correctamente los datos de instancia.

Si existen elementos o atributos en más de un espacio de nombres, MapForce genera un esquema XML distinto por cada espacio de nombres (es decir, puede que se creen varios archivos en el disco).

DTD como estructura de un documento

A partir de la versión 2006 SP2, MapForce es compatible con el uso de documentos DTD preparados para espacios de nombres como componentes de origen y destino. Para ello, los URI de espacio de nombres se extraen de las declaraciones de atributo `xmlns` de la DTD. Sin embargo, algunas DTD contienen declaraciones de atributo `xmlns` sin URI de espacio de nombres (p. ej. las DTD que utiliza StyleVision). Estas DTD se deben

 © 2016-2022 Altova GmbH

ampliar para poder utilizarse en MapForce. Concretamente los atributos `xmlns` deben definirse con el URI de espacio de nombres tal y como aparece en este ejemplo:

```
<!ATTLIST fo:root
   xmlns:fo CDATA #FIXED 'http://www.w3.org/1999/XSL/Format'
   ...
>
```

En esta sección

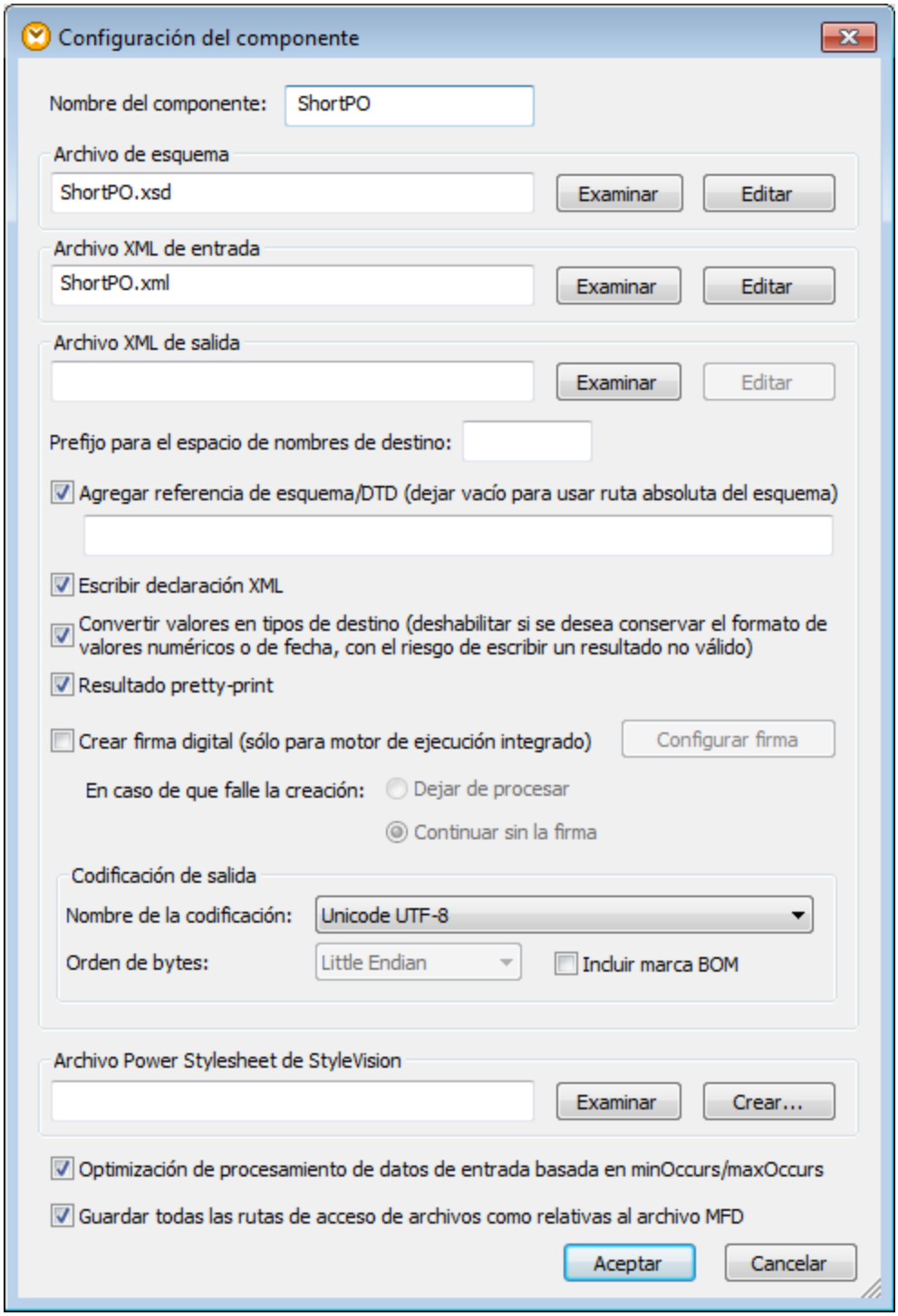
Estos son los apartados de esta sección:

- Configuración de componentes XML 125
- Tipos derivados 129
- Valores NULL 131
- Comentarios e instrucciones de procesamiento 135
- Secciones CDATA 136
- Comodines: xs:any / xs:anyAttribute 137
- Declarar espacios de nombres personalizados 140
- Gestor de esquemas XML 143

4.1.1 Configuración de componentes XML

Tras añadir un componente XML al área de asignación, podrá configurarlo desde el cuadro de diálogo "Configuración del componente". Este cuadro de diálogo se puede abrir de tres maneras diferentes:

- Seleccionando el componente en el área de asignación y haciendo clic en el comando de menú **Componente | Propiedades**.
- Haciendo doble clic en el título del componente.
- Haciendo clic con el botón derecho en el título del componente y eligiendo **Propiedades** en el menú contextual.



A continuación explicamos las opciones disponibles.

Configuración general

- Nombre del componente

© 2016-2022 Altova GmbH

El nombre del componente se genera automáticamente al crearlo, pero se puede volver a cambiar cuando quiera. El nombre del componente puede contener espacios y puntos, pero no barra, barra inversa, dos puntos, comillas dobles, ni espacios antes ni después del nombre.

Si cambia el nombre del componente, tenga en cuenta que:

- Si quiere implementar la asignación en FlowForce Server, el nombre del componente debe ser único.
- Recomendamos que use únicamente caracteres que se puedan usar también en la línea de comandos. Los caracteres especiales propios de cada país pueden estar codificados de forma diferente en Windows y la línea de comandos.

Archivo de esquema

Especifica el nombre o la ruta de acceso del archivo de esquema XML que MapForce usará para validar y asignar datos. Para cambiar de archivo de esquema haga clic en **Examinar** y seleccione el nuevo archivo. Para editarlo en XMLSpy haga clic en **Editar**..

Archivo XML de entrada

Especifica el archivo XML de instancia donde MapForce leerá datos. Este campo es relevante para los componentes de origen y se rellena automáticamente cuando se crea el componente por primera vez y se le asigna un archivo XML de instancia. En un componente de origen el nombre del archivo de instancia también se usa para detectar el elemento XML raíz y el esquema al que hace referencia, además de para validar los datos con el esquema seleccionado. Para cambiar la ubicación del archivo haga clic en **Examinar** y seleccione el nuevo archivo. Para editarlo en XMLSpy haga clic en **Editar**.

Archivo XML de salida

Especifica el archivo XML de instancia donde MapForce escribirá los datos. Este campo es relevante para los componentes de destino. Para cambiar la ubicación del archivo haga clic en **Examinar** y seleccione el nuevo archivo. Para editarlo en XMLSpy haga clic en **Editar**.

Prefijo para el espacio de nombres de destino

AllowsPermite introducir un prefijo para el espacio de nombres de destino. Antes de asignar el prefijo, asegúrese de que en el esquema de destino está definido el espacio de nombres de destino.

Agregar referencia de esquema/DTD

Agrega la ruta de acceso del esquema XML referenciado al elemento raíz del archivo XML de salida. La ruta de acceso del esquema introducido en este campo se escribe en los archivos de instancia de destino que se generan (en el atributo `xsi:schemaLocation` o en la declaración `DOCTYPE` si se usa una DTD.

En las ediciones Professional and Enterprise: si genera código en XQuery o C++, la opción de agregar una referencia de DTD no estará disponible.

Al introducir una ruta de acceso en este campo podrá definir dónde está ubicado el archivo de esquema al que hace referencia el archivo XML de instancia. Esto garantiza que la instancia de salida se pueda validar en el destino de la asignación cuando se ejecute la asignación. En este campo puede introducir tanto una dirección **http://** como una ruta de acceso absoluta o relativa.

Si desactiva esta opción, podrá desvincular la instancia XML del esquema XML o DTD de referencia (por ejemplo, si desea enviar el documento XML de salida a alguien que no tiene acceso al esquema XML subyacente).

- Escribir declaración XML

ByEsta opción permite suprimir la declaración XML del documento de salida que se genera. Esta opción está habilitada por defecto, es decir, la declaración XML se describe por defecto en los resultados. Esta característica no es compatible con todos los lenguajes de destino y motores de ejecución de MapForce, como puede verse en la tabla siguiente.

Lenguaje de destino / motor de ejecución	**Si el resultado es un archivo**	**Si el resultado es una cadena de texto**
Integrado (ediciones *Professional y Enterprise*)	Yes	Yes
MapForce Server (ediciones *Professional y Enterprise*)	Yes	Yes
XSLT, XQuery	Yes	No
Generador de código (C++, C#, Java) (ediciones *Professional y Enterprise*)	Yes	Yes

- Convertir valores en tipos de destino

Permite definir si los tipos del esquema XML de destino deben utilizarse durante la asignación o si todos los datos asignados al componente de destino deben considerarse valores de **cadena**. Esta opción está habilitada por defecto. Si desactiva esta opción podrá conservar el formato preciso de los valores. Por ejemplo, si se desea satisfacer una faceta de patrón de un esquema que requiere un número concreto de dígitos decimales en un valor numérico. Puede usar funciones de asignación de datos para dar el formato de cadena necesario al número y después asignar esta cadena al destino.

Recuerde que si deshabilita esta opción, también se deshabilita la detección de valores no válidos (p. ej. si se escriben letras en campos numéricos).

- Resultado pretty-print

Ajusta el formato del documento XML de salida para que tenga un aspecto estructurado. Cada nodo secundario se desplaza una tabulación con respecto a su primario.

- Create digital signature (*Crear firma digital (solo en la edición Enterprise Edition)*)

Permite agregar una firma digital al archivo XML de instancia de salida. Esto solo es compatible con el lenguaje de transformación integrado.

Codificación de salida

Permite especificar estas opciones de configuración para el archivo de instancia de salida:

- Nombre de la codificación

 © 2016-2022 Altova GmbH

- Orden de bytes
- Si se incluye o no la marca BOM

Por defecto, todos los componentes nuevos tienen la codificación definida en la opción *Codificación predeterminada para componentes nuevos*. Esta opción se puede modificar en la pestaña *Generales* de **Herramientas | Opciones**.

Si la asignación de datos genera código XSLT 1.0/2.0, el estado de la casilla *Incluir marca BOM* no tiene ningún efecto porque estos lenguajes no son compatibles con marcas BOM.

Archivo Power Stylesheet de StyleVision

Esta opción permite seleccionar o crear un archivo de hoja de estilos de Altova StyleVision. Estos archivos permiten presentar los datos de salida de un archivo XML de instancia en varios formatos, como HTML, RTF, etc. Consulte también el apartado Usar rutas de acceso relativas en un componente 82.

Otras opciones

- Optimización de procesamiento de datos de entrada basada en minOccurs/maxOccurs

  Esta opción permite tratar de forma especial las secuencias que contienen un elemento exactamente, como por ejemplo los atributos obligatorios o los elementos secundarios con atributos `minOccurs` y `maxOccurs="1"`. En este caso, MapForce extrae el primer elemento de la secuencia y después procesa el elemento directamente como valor atómico (y no como secuencia).

  Si los datos de entrada **no son válidos** según el esquema, puede que la asignación incluya una secuencia vacía que detenga la ejecución con un mensaje de error. Para permitir el procesamiento de dicha **entrada no válida** basta con desactivar esta casilla.

- Guardar todas las rutas de acceso de archivos como relativas al archivo MFD

  Cuando se habilita esta opción, MapForce guarda las rutas de acceso de archivos que aparecen en el cuadro de diálogo "Configuración del componente" como relativas a la ubicación del archivo de diseño de MapForce (.mfd). Consulte también el apartado Usar rutas de acceso relativas en un componente 82.

4.1.2 Tipos XML Schema derivados

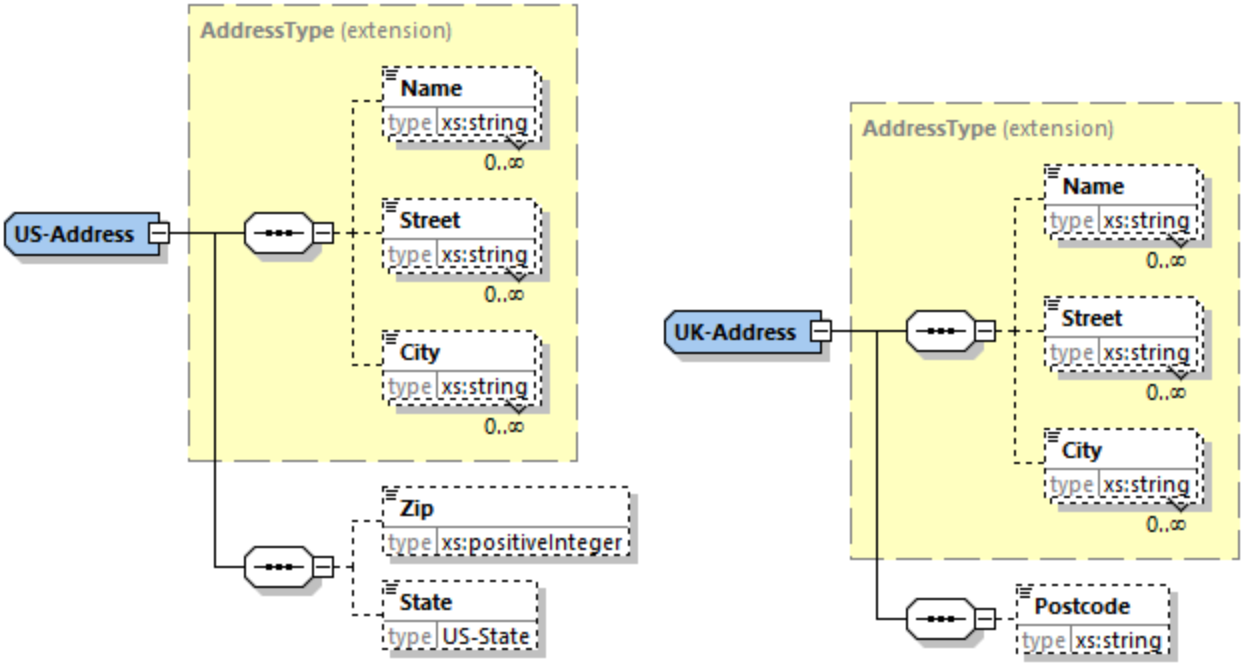
MapForce permite realizar asignaciones de datos entre tipos derivados de un tipo complejo. Los tipos derivados son tipos complejos de un esquema XML que usan el atributo `xsi:type` para identificar los tipos derivados especificados (p.e., `<Address xsi:type="UK-Address">`). Puede ver las definiciones de estos tipos en la especificación de esquema XML de W3C (sección 2.5.2).

Caso posible

En esta subsección explicamos un posible caso de uso de un tipo derivado. Por ejemplo, imagine que hay una empresa con dos filiales: una en Reino Unido y la otra en Estados Unidos. Lo que queremos es tener dos listas (`UKCustomers` y `USCustomers`), cada una de las cuales debe incluir la información sobre la dirección de la filial correspondiente y todos los clientes asociados a ella.

Defiición de tipos derivados

En las imágenes siguientes puede ver la definición de los tipos derivados `US-Address` y `UK-Address` (*en la vista Esquema de XMLSpy*). El tipo base (o tipo complejo originario) es `AddressType` e incluye los elementos `Name`, `Street` y `City`. En el elemento `US-Address` se añadieron también dos elementos más para crear los tipos derivados `US-Address`: `Zip` y `State`, mientras que el elemento `UK-Address` incluye el tipo base y el elemento `Postcode`. En este ejemplo solo vamos a asignar al archivo de destino el elemento `UK-Address`.

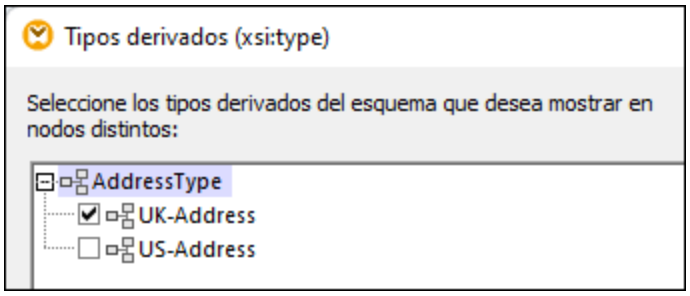


Tipos derivados en una asignación

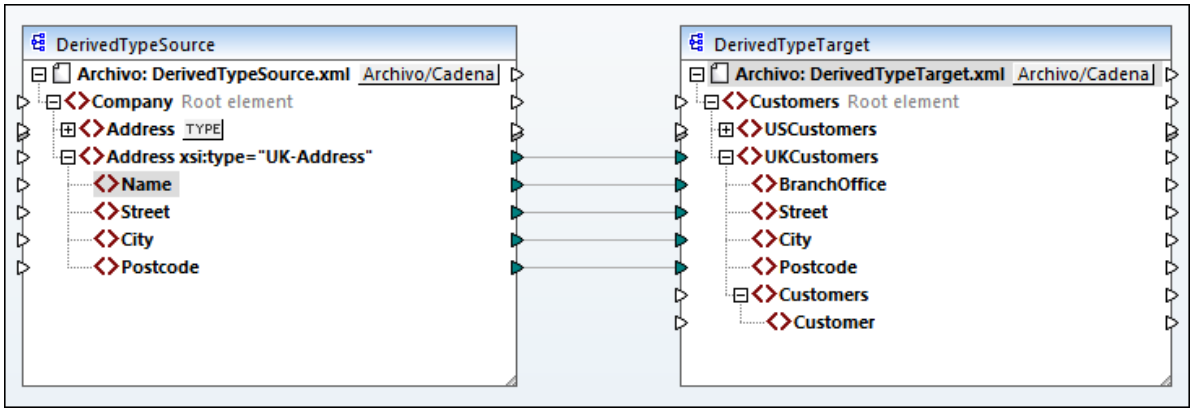
A continuación explicamos cómo realizar asignaciones de datos entre tipos derivados de esquema XML. Lo que buscamos es asignar información sobre la oficina de Reino Unido al elemento `UKCustomers`. Puede encontrar los archivos de ejemplo en la carpeta **`Tutorial`**.

1. En el menú **Insertar** haga clic en el comando **Archivo o esquema XML** y abra **`DerivedTypeSource.xml`**. Este archivo XML se basa en **`DerivedTypeSource.xsd`**..
2. Inserte el archivo de destino **`DerivedTypeTarget.xsd`**. Observe que el esquema de destino no necesita incluir el atributo `xsi:type`.
3. Ahora haga clic en el botón TYPE situado junto al elemento `Address`. Este botón indica que para este elemento existen tipos derivados en el esquema.
4. EL cuadro de diálogo Tipos derivados (*imagen siguiente*) permite seleccionar los tipos derivados disponibles para ese elemento en concreto. En esta asignación de ejemplo solamente queremos asignar el elemento `UK-Address`.

 © 2016-2022 Altova GmbH



5. Al marcar la casilla del tipo derivado `UK-Address` verá que en el componente aparece un elemento nuevo llamado `Address xsi:type="UK-Address"`.
6. Conecte los nodos como se ver en la imagen siguiente.



Resultado
Al hacer clic en el panel Resultados verá este código:

```
<UKCustomers>
   <BranchOffice>Sleuth Corp. UK</BranchOffice>
   <Street>222 Baker St</Street>
   <City>London</City>
   <Postcode>NW1 6XE</Postcode>
</UKCustomers>
```

Esta asignación de ejemplo está guardada como **`Tutorial\DerivedType.mfd`**. También puede agregar otro archivo XML de origen que incluya información sobre los clientes de Reuno Unido y asignar esos datos al nodo `Customers` del componente de destino. De esta manera el elemento `UKCustomers` incluye información sobre la dirección de Reino Unido y todos los clientes asociados a esa filial.

4.1.3 Valores Nil y Nillable

En esta sección explicamos cómo se ocupa MapForce de los valores NULL en los componente de origen y de destino. Para poder usar el atributo `xsi:nil="true"` en un archivo XML debe especificar el atributo `nillable="true»` en los elementos relevantes del archivo de esquema correspondiente. Para saber más sobre los atributos `nillable` y `xsi:nil` consulte la especificación del W3C. Observe que el atributo `xsi:nil` no es visible en la estructura de los componentes del panel **Asignación**.

A continuación explicamos algunos casos de uso de los valores NULL en asignaciones.

Valores NULL en componentes XML

En esta subsección vamos a ver algunos ejemplos de casos de uso de elementos de asignación que tienen un atributo `xsi:nil="true"`.

Solo el elemento de origen tiene xsi:nil="true"/Tanto el elemento de origen como el de destino tienen xsi:nil="true"
En este caso las condiciones son:

- La conexión está basada en destino 91.
- El elemento de origen es un atributo `xsi:nil="true"`. El elemento de destino correspondiente no tiene este atributo.
- También es posible que los dos elementos, el de origen y el de destino, tengan atributos `xsi:nil="true"`.
- Los atributos `nillable="true"` se deben definir en los esquemas de origen y de destino,
- Los elementos de origen y de destino son de tipo simple.

En este caso, el elemento de destino tendrá el atributo `xsi:nil="true"` en el archivo de salida, como se ve en el archivo de resultados de más abajo (*resaltado en amarillo*).

```
<book id="7">
       <author>Edgar Allan Poe</author>
       <title>The Murders in the Rue Morgue</title>
       <category xsi:nil="true"/>
       <year>1841</year>
       <OrderID id="213"/>
</book>
```

Nota: si el atributo `nillable="true"` no está definido en el esquema de destino, el elemento de destino correspondiente estará vacío en el resultado.

Solo el elemento de destino tiene xsi:nil="true»
En este caso las condiciones son:

- La conexión está basada en destino 91.
- El elemento de origen no tiene un atributo `xsi:nil="true"`.
- El elemento de destino correspondiente tiene un atributo `xsi:nil="true"`.
- Los elementos de origen y de destino pueden ser de tipo simple o complejo.

En este caso el elemento de origen sobrescribirá el elemento de destino que contiene el atributo `xsi:nil="true"`. En el ejemplo siguiente puede ver un archivo de resultados de ejemplo. El elemento `<genre>` incluye el atributo `xsi:nil="true»` en el elemento de destino. Sin embargo, este elemento se sobrescribe en tiempo de ejecución, por lo que el elemento `<genre>` (*resaltado en amarillo*) tiene `Fiction` en el resultado.

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

 © 2016-2022 Altova GmbH

```
        <OrderID id="124"/>
</publication>
```

El elemento de tipo complejo de origen/los dos elementos de tipo complejo tienen xsi:nil="true"
En este caso las condiciones son:

- La conexión está basada en destino 91.
- El elemento de origen es de tipo complejo. En este ejemplo el elemento de origen tiene un atributo `id="213"` y un atributo `xsi:nil="true»`. El elemento de destino correspondiente también es de tipo complejo y tiene un atributo `id="124"`, pero no un atributo `xsi:nil="true»`.
- También puede ser que los dos elementos, el de origen y el de destino, que son de tipo complejo, tengan atributos `xsi:nil="true»`.

En este caso el elemento de origen sobrescribirá el elemento de destino (*resaltado en amarillo*). Sin embargo, el atributo `xsi:nil="true"` no se escribe automáticamente en el archivo de salida. Para ver el atributo `xsi:nil="true"` en el elemento de destio del archivo de resultados use una conexión de copia total 95.

```
<book id="7">
        <author>Edgar Allan Poe</author>
        <title>The Murders in the Rue Morgue</title>
        <year>1841</year>
        <OrderID id="213"/>
</book>
```

Funciones útiles

Estas funciones sirven para comprobar, reemplazar y asignar valores NULL:

- **`is-xsi-nil`** 675: Ayuda a comprobar de forma explícita si un elemento de origen tiene el atributo `xsi:nil` definido como `true`.
- **`substitute-missing`** 707: Sustituye un valor NULL en el elemento de origen con algo específico.
- **`set-xsi-nil`** 677: Asigna el elemento `xsi:nil="true"` a un elemento de destino. Esto funciona con elementos de destino de tipo simple y complejo.
- **`substitute-missing-with-xsi-nil`** 678: Si hay contenido, este se escribe en el elemento de destino; si falta algún valor, esta función produce un elemento de destino con el atributo `xsi:nil="true»` en el resultado.
- Si conecta la función **`exists`** 683 a un elemento de origen con un valor NULL devuelve `true` aunque el elemento no tenga contenido.

Observe que las funciones que generan `xsi:nil` no pueden pasar a través de funciones o componentes que solo operan en valores (como la función **`if-else`**).

Valores NULL en componentes de BD

En las subsecciones siguientes explicamos cómo se trata a los valores NULL en los componentes de BD.

Asignar campos de BD NULL a elementos NULL
Los elementos de destino que reciben valores NULL de campos de BD no se crean en el resultado automáticamente. Para ver esos elementos en el resultado tiene que (i) añadir atributos `nillable="true»` a los elementos de destino relevantes en el archivo de esquema y (ii) usar la función **`substitute-missing-with-xsi-nil`** 678 en la asignación. En el ejemplo siguiente puede ver cómo gestionas los valores NULL en asignaciones que tienen un componente de BD de origen.

Tabla de aplicaciones en el panel Consulta de la BD
Puede encontrar la asignación de ejemplo en: **`Tutorial\DBNullToXML.mfd`**. Para este ejemplo hemos elegido una sola tabla (`Application`) de la BD `Accounts` (*imagen siguiente*).

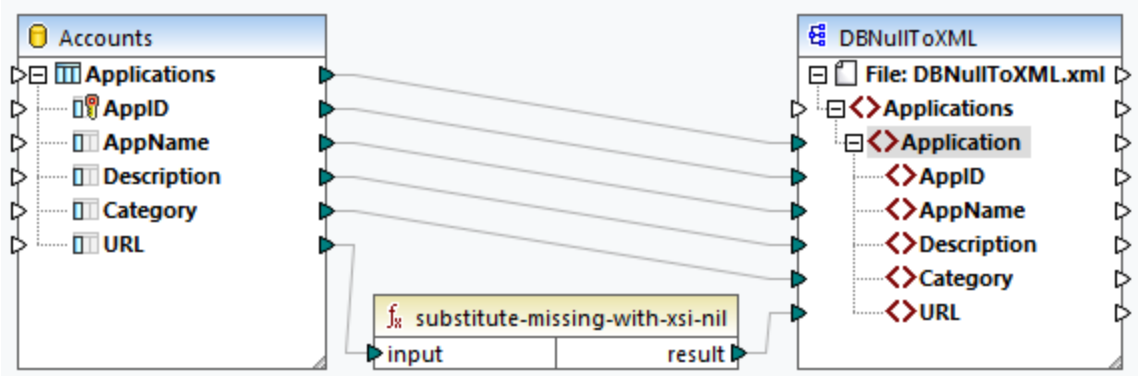
	AppID	AppName	Description	Category	URL
1	1	Altova MapForce	Best data mapping tool!	IDE	https://www.altova.com/mapforce
2	2	Notepad	*[NULL]*	*[NULL]*	*[NULL]*

Para ver la tabla `Application` siga estos pasos:

- Abra el panel **Consulta de BD**.
- Seleccione la base de datos `Accounts` para ver su estructura en el Explorador de BD.
- Haga clic con el botón derecho en la tabla `Application` y haga clic en **Mostrar en Editor SQL | SELECT**.
- Haga clic en el botón (**Ejecutar consulta**). Aparece la tabla `Application` en la pestaña **Resultados**.

Para saber más sobre consultas de BD consulte consulte Examinar y consultar bases de datos 380.

Asignación
En la tabla `Application` de más arriba puede ver que el segundo registro tienen valores NULL en los campos `Description`, `Category` y `URL` Para este ejemplo vamos a asignar casi todas las columnas directamente con los elementos de destino correspondientes. Para la columna `URL` vamos a usar la función **`substitute-missing-with-xsi-nil`** para que el valor NULL del elemento de destino tenga un atributo `xsi:nil="true"` (*imagen siguiente*).



Resultados
En el archivo de resultados siguiente puede ver que el primer registro de la tabla se ha escrito por completo en el resultado, mientras que el segundo se ha escrito solo parcialmente. Los valores NULL de BD están ausentes del resultado, salvo por el elemento `URL`. El elemento `URL` tiene el atributo `nillable="true»` en el archivo de esquema y estamos usando la función **`substitute-missing-with-xsi-nil`**, por lo que el elemento `URL` ahora tiene el atributo `xsi:nil="true"` en el resultado (*resaltado en amarillo*).

```
<Application>
        <AppID>1</AppID>
        <AppName>Altova MapForce</AppName>
```

```
        <Description>Best data mapping tool!</Description>
        <Category>IDE</Category>
        <URL>https://www.altova.com/mapforce</URL>
</Application>
<Application>
        <AppID>2</AppID>
        <AppName>Notepad</AppName>
        <URL xsi:nil="true"/>
</Application>
```

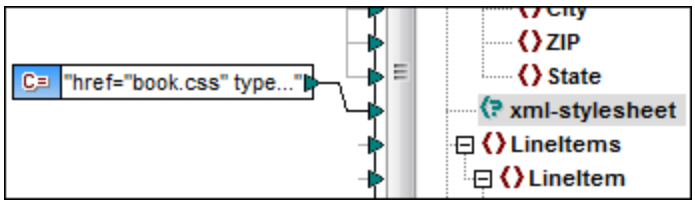
Asignar elementos NULL a campos de BD NULL
Si asigna un elemento XML NULL a una columna de BD MapForce escribe el valor NULL en la columna de BD correspondiente. También puede usar la función **set-null** 723 si quiere definir un campo de BD como NULL. Para saber más sobre las funciones relacionadas con bases de datos consulte la Biblioteca de bases de datos 722.

4.1.4 Comentarios e instrucciones de procesamiento

En los componentes XML de destino se pueden insertar comentarios e instrucciones de procesamiento (consulte la especificación de W3C). Estas últimas sirven para pasar información a las aplicaciones que continúan con el procesamiento de los documentos XML. Recuerde que no se pueden definir comentarios ni instrucciones de procesamiento en nodos que formen parte de un grupo de asignación de copia total 95.

Para insertar una instrucción de procesamiento
Para insertar una instrucción de procesamiento siga estos pasos:

1. Haga clic con el botón derecho en un elemento del componente de destino y elija **Comentario/instrucción de procesamiento** en el menú contextual y después elija **Agregar delante una instrucción de procesamiento** o bien **Agregar detrás una instrucción de procesamiento**. Introduzca el nombre (de destino) de la instrucción de procesamiento y haga clic en **Aceptar** para confirmar. En este ejemplo hemos insertado la instrucción `xml-stylesheet` e la estructura del componente de destino, tras el elemento `State`.



2. Ahora puede usar un componente de constante para aportar el valor del atributo de la instrucción de procesamiento (*imagen anterior*).

Nota: puede añadir varias instrucciones de procesamiento delante o detrás de cualquier elemento del componente de destino.

Nota: solo se puede añadir un comentario antes y después de un nodo de destino. Para crear varios comentarios use la función de duplicar componentes de entrada 81.

Para eliminar un comentario o una instrucción de procesamiento
Haga clic con el botón derecho en el correspondiente nodo, elija **Comentario/instrucción de procesamiento** en el menú contextual y después elija **Eliminar**.

4.1.5 Secciones CDATA

Las secciones CDATA sirven para añadir caracteres de escape a bloques de texto que contienen caracteres que se podrían interpretar como marcado. Para más información consulte la especificación de W3C. Los nodos de destino pueden escribir los datos de entrada que reciben como secciones CDATA. Los componentes de destino pueden ser: datos XML, datos XML incrustados en campos de BD o elementos secundarios XML de dimensiones con tipo de un componente XBRL de destino. También se pueden definir secciones CDATA en nodos duplicados y en nodos `xsi:type`.

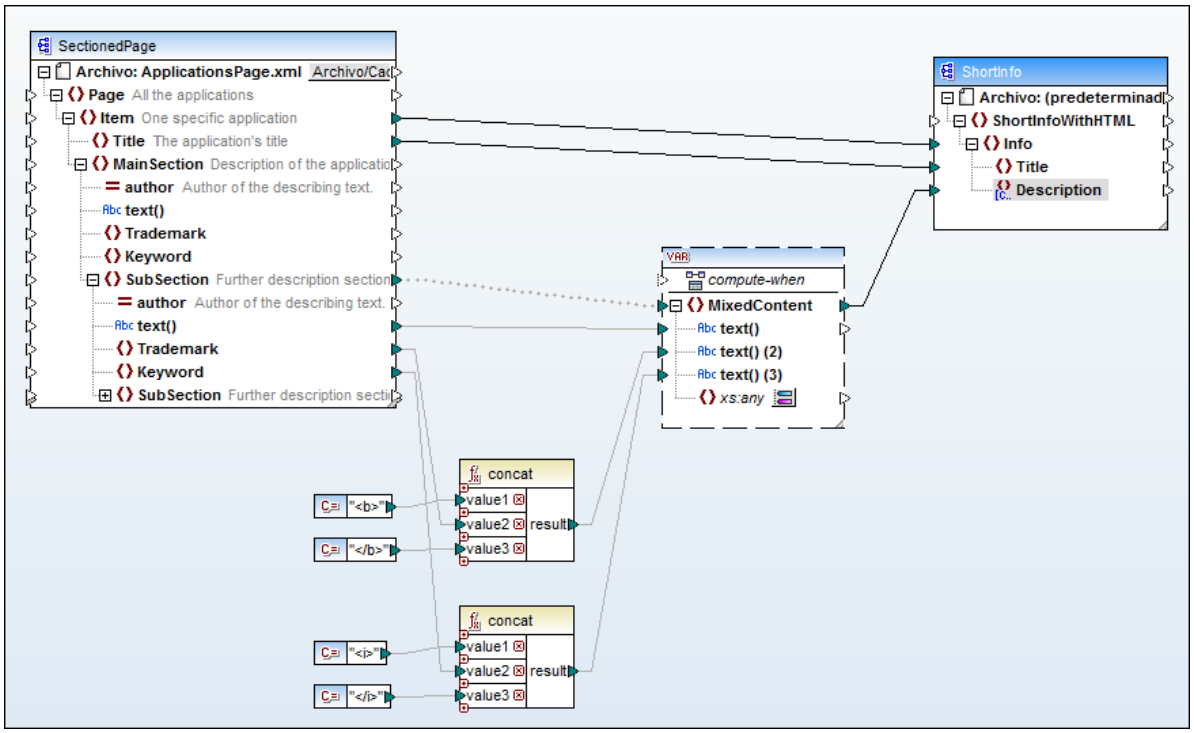
Para crear una sección CDATA haga clic con el botón derecho en el nodo de destino que desea definir como sección CDATA y seleccione **Escribir contenido como sección CDATA**. Aparece entonces una advertencia señalando que los datos de entrada no deben contener el delimitador de cierre de sección CDATA **`]]>`**. Haga clic en **Aceptar** para cerrar el aviso. El icono **`[C..`** que aparece debajo de la etiqueta del elemento indica que este nodo está definido como sección CDATA.

Ejemplo

El diseño de asignación **`MapForceExamples\HTMLinCDATA.mfd`** muestra lo útil que pueden ser las secciones CDATA (*imagen siguiente*). En este ejemplo:

- El elemento `SubSection` se definió como nodo basado en el origen 91 (de contenido mixto).
- Al contenido del elemento de origen `Trademark` se añaden las etiquetas de apertura y cierre `<b></b>`.
- Al contenido del elemento de origen `Keyword` se añaden las etiquetas de apertura y cierre `<i></i>`
- Los datos resultantes se pasan a los nodos `text()` duplicados en el orden en que aparecen en el documento de origen porque el conector del elemento `SubSection` se definió como nodo basado en el origen 91 (de contenido mixto).
- El resultado del nodo `MixedContent` se pasa después al nodo `Description` del componente de destino `ShortInfo`, que se definió como sección CDATA.

 © 2016-2022 Altova GmbH



Resultado

Si abrimos el panel *Resultados* podremos ver la sección CDATA que contiene el texto con marcado.

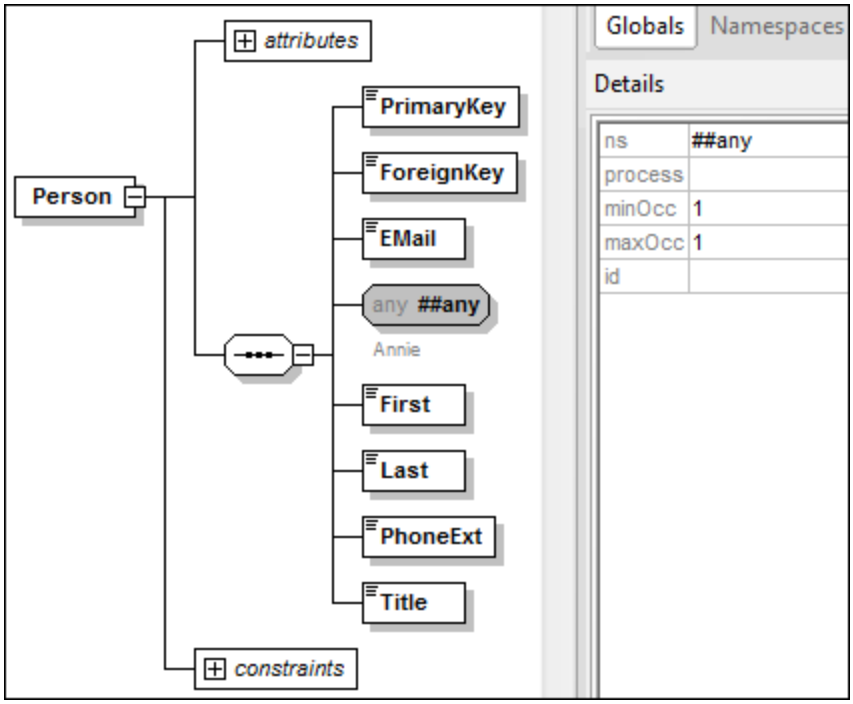
```
7    <Info>
8      <Title>MapForce</Title>
9      <Description><![CDATA[Altova <b>MapForce</b> 2014 Enterprise Edition is the premier <i>XML</i>
/ <i>database</i> / <i>flat file</i> / <i>EDI</i> data mapping tool that auto-generates mapping code in
<i>XSLT</i> 1.0/2.0, <i>XQuery</i> , <i>Java</i> , <i>C++</i> and <i>C#</i> . It is the definitive tool for
data integration and information leverage.]]></Description>
10   </Info>
```

4.1.6 Comodines: xs:any / xs:anyAttribute

En este apartado explicamos cómo trabajar con comodines en asignaciones de datos. Los comodines `xs:any` (y `xs:anyAttribute`) permiten usar cualquier elemento o atributo de los esquemas. más información consulte la especificación de W3C.

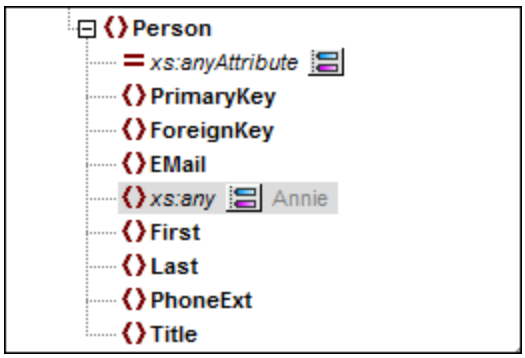
Comodines en la definición del esquema

En la imagen siguiente puede ver el elemento `xs:any` en la vista Esquema de Altova XMLSpy. Este elemento se ha definido como elemento secundario del elemento `Person`.



Comodines en MapForce

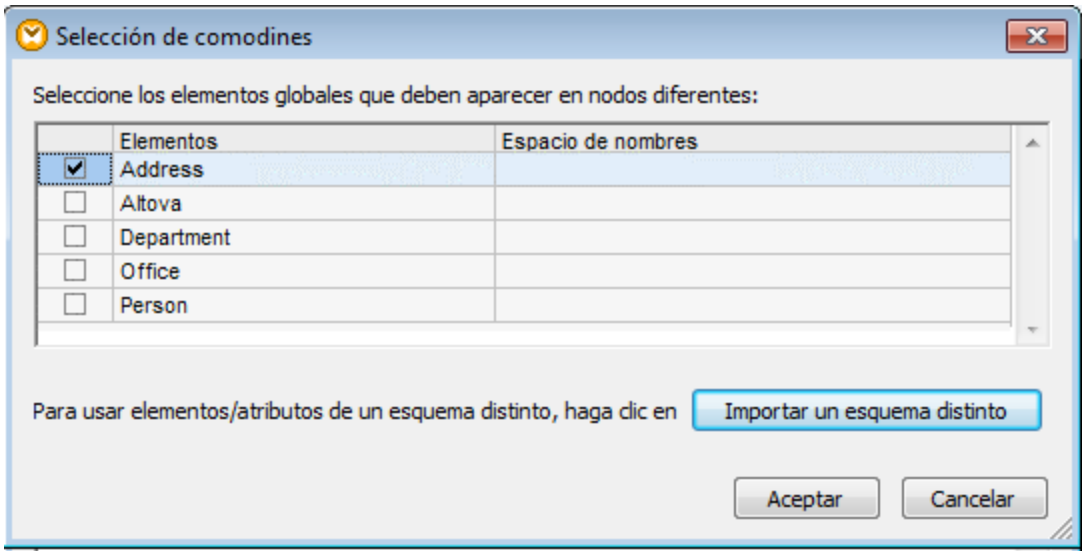
Cuando se define un comodín para un elemento y/o atributo, el nodo comodín aparece con el icono (**Cambiar selección**) junto a él (*imagen siguiente*).



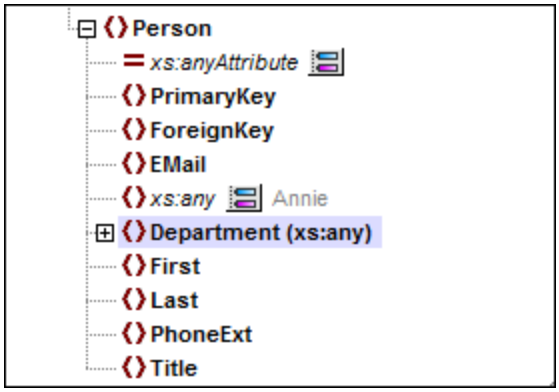
Selección de comodines

Cuando se hace clic en el botón **Cambiar selección**, aparece el cuadro de diálogo "Selección de comodines" (*imagen siguiente*). En este cuadro de diálogo se enumeran los elementos y atributos globales declarados en el esquema actual.

 © 2016-2022 Altova GmbH



En este ejemplo hemos seleccionado `Department`. Observe que los elementos y atributos comodines se insertan tras el nodo al que acompaña el icono . Ahora el componente tiene este aspecto:



Ahora puede crear asignaciones entre estos nodos y cualquier otro elemento. En un componente puede reconocer los elementos y atributos comodines por el texto `(xs:any) y (xs:anyAttribute)` que aparece anexado (*imagen anterior*).

Eliminar comodines

Para eliminar un elemento comodín haga clic en el botón **Cambiar selección** y desactive su casilla en el cuadro de diálogo "Selección de comodines".

Agregar elementos de un esquema distinto como comodines

El cuadro de diálogo "Selección de comodines" (*véase más arriba*) permite usar como comodines los elementos de un esquema que no sea el asignado. Para poder ver estos elementos en el componente haga clic en el botón **Importar un esquema distinto**, donde podrá elegir entre (i) importar un archivo de esquema o (ii) generar un esquema contenedor (*véase más abajo*).

Importar un esquema

La opción **Importar esquema** importa el esquema externo en el esquema actual que está asignado al componente. Tenga en cuenta que esta opción sobrescribe el esquema actual del componente en el disco. Si el esquema actual es un esquema remoto que se abrió desde una dirección URL (véase Agregar componentes desde una URL 77) y no desde el disco, entonces no se podrá modificar. En este caso debe utilizarse la opción **Generar esquema contenedor**.

Generar un esquema contenedor

La opción **Generar esquema contenedor** crea un archivo de esquema nuevo que se denomina esquema *contenedor*. La ventaja de usar esta opción reside en que el esquema actual del componente no se modifica. Por el contrario, se creará un esquema nuevo (es decir, un esquema contenedor) que incluirá tanto el esquema actual como el esquema que se desea importar. Cuando haga clic en esta opción, MapForce preguntará dónde se debe guardar el esquema contenedor. El nombre predeterminado del esquema contenedor será **`somefile-wrapper.xsd`**.

Una vez guardado, el esquema contenedor se asigna automáticamente y por defecto al componente y aparece un aviso que le pregunta si quiere cambiar la ubicación del esquema para poder hacer referencia al esquema principal anterior. Haga clic en **Sí** para volver al esquema anterior y en **No** para que el esquema contenedor recién creado siga asignado al componente.

Comodines y nombres de nodo dinámicos

En algunos casos los elementos o atributos que aparecen en la instancia son demasiados. Por ejemplo, en esta instancia:

```
<?xml version="1.0" encoding="UTF-8"?>
<message>
   <line1>1</line1>
   <line2>2</line2>
   <line3>3</line3>
   ................
   <line999></line999>
</message>
```

Para casos como este, en lugar de comodines, se recomienda un acceso dinámico a los nombres de nodo (*véase Asignar nombres de nodos 850*).

4.1.7 Declarar espacios de nombres personalizados

Cuando una asignación produce un resultado XML, MapForce deriva automáticamente el espacio de nombres (o conjunto de espacios de nombres) de cada elemento y atributo a partir del esquema asociado al componente de destino 73. Este es el comportamiento predeterminado de MapForce y el más apropiado en

las asignaciones de datos que implican la generación de resultados XML. Sin embargo, en otros casos puede ser preferible tener un mayor control sobre el espacio de nombres de los elementos en el código XML resultante. Por ejemplo, en algunos casos puede ser necesario declarar a mano el espacio de nombres de un elemento desde la asignación directamente.

La declaración de espacios de nombres personalizados (y el comando **Agregar espacio de nombres**) solamente es relevante para componentes de destino XML y solamente para elementos. El comando **Agregar espacio de nombres** no está disponible en el caso de atributos ni nodos comodín. Tampoco está disponible en caso de nodos que reciben datos por medio de conexiones de copia total 95.

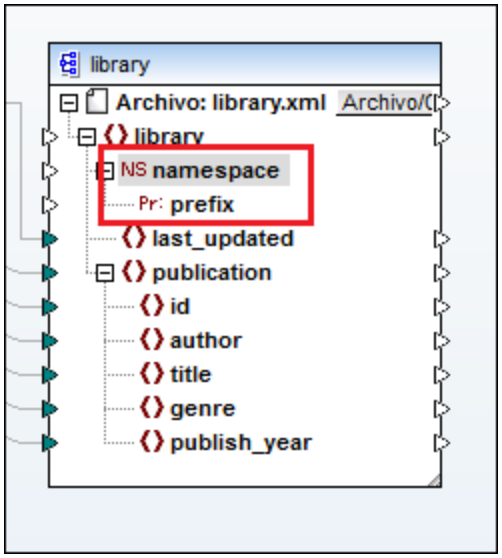
Para comprender mejor este funcionamiento siga las instrucciones de las subsecciones que siguen.

Declarar un espacio de nombres manualmente

Para este ejemplo necesita la asignación **`BasicTutorials\Tut1-SchemaToSchema.mfd`**.

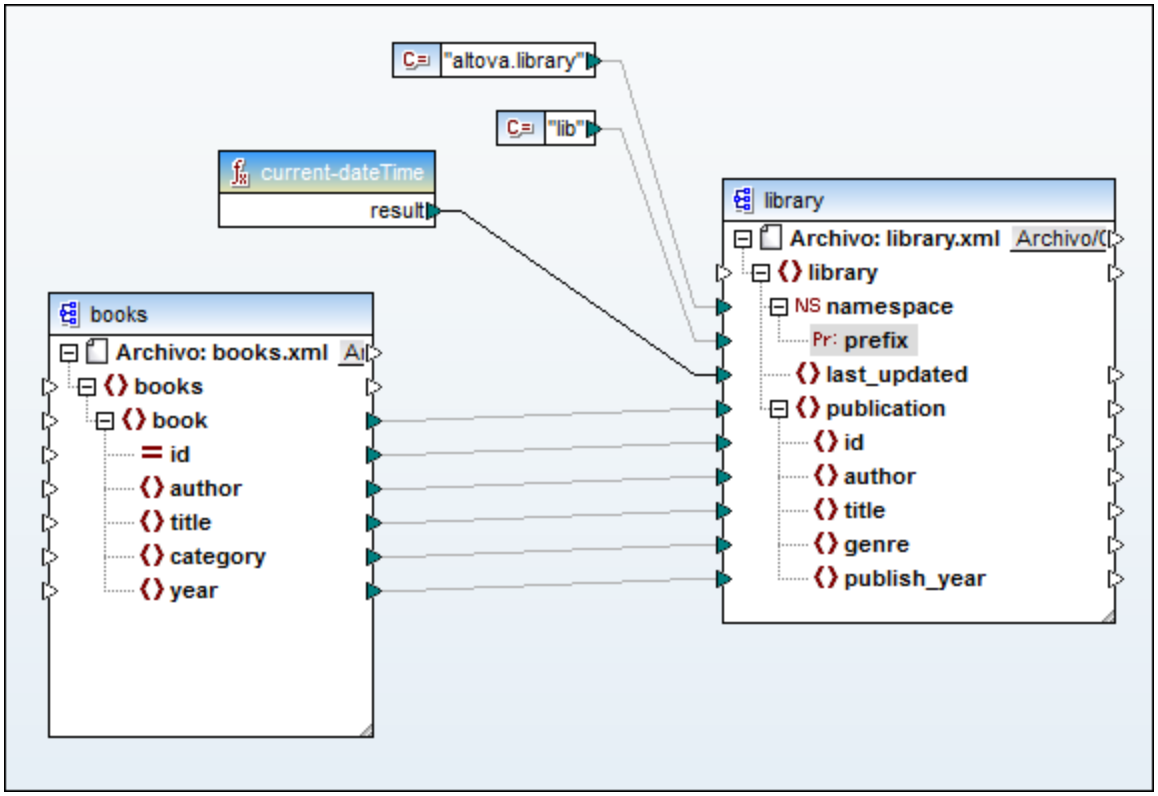
Agregar un espacio de nombres

Abra la asignación, haga clic con el botón derecho en el nodo `library` del componente `BooksOutput` y seleccione **Agregar espacio de nombres** en el menú contextual. Observe que ahora aparecen dos nodos nuevos bajo el nodo `library`: el nodo `namespace` y el nodo `prefix` (*imagen siguiente*).



Indicar los valores del espacio de nombres

Ahora puede crear asignaciones entre estos nodos y valores de cadena. Por ejemplo, en la imagen siguiente puede ver que se definieron dos constantes que aportan el espacio de nombres `altova.library` y el prefijo `lib` (*imagen siguiente*).



Nota: debe asignar los conectores de entrada `namespace` y `prefix` aunque les asigne valores vacíos.

Resultado
En el resultado podrá ver que el atributo `xmlns:<prefix>="<namespace>"` se añadió al elemento y que `<prefix>` y `<namespace>` son valores procedentes de la asignación. El resultado será (*fíjese en la parte resaltada*):

```
<?xml version="1.0" encoding="UTF-8"?>
<library xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:lib="altova.library" xsi:noNamespaceSchemaLocation="Library.xsd">
...
```

También puede declarar varios espacios de nombres para un mismo elemento. Para ello vuelva a hacer clic con el botón derecho en el nodo y seleccione **Agregar espacio de nombres** en el menú contextual. Bajo el nodo aparecerán dos nodos más para el espacio de nombres y para el prefijo. Ahora podrá conectar dos nuevos valores a estos dos nodos.

Declarar un espacio de nombres predeterminado
Si desea declarar un espacio de nombres predeterminado, basta con asignar un valor de cadena vacío al nodo `prefix`. El resultado será (*fíjese en la parte resaltada*):

```
<?xml version="1.0" encoding="UTF-8"?>
<library xmlns="altova.library" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:noNamespaceSchemaLocation="Library.xsd">
...
```

 © 2016-2022 Altova GmbH

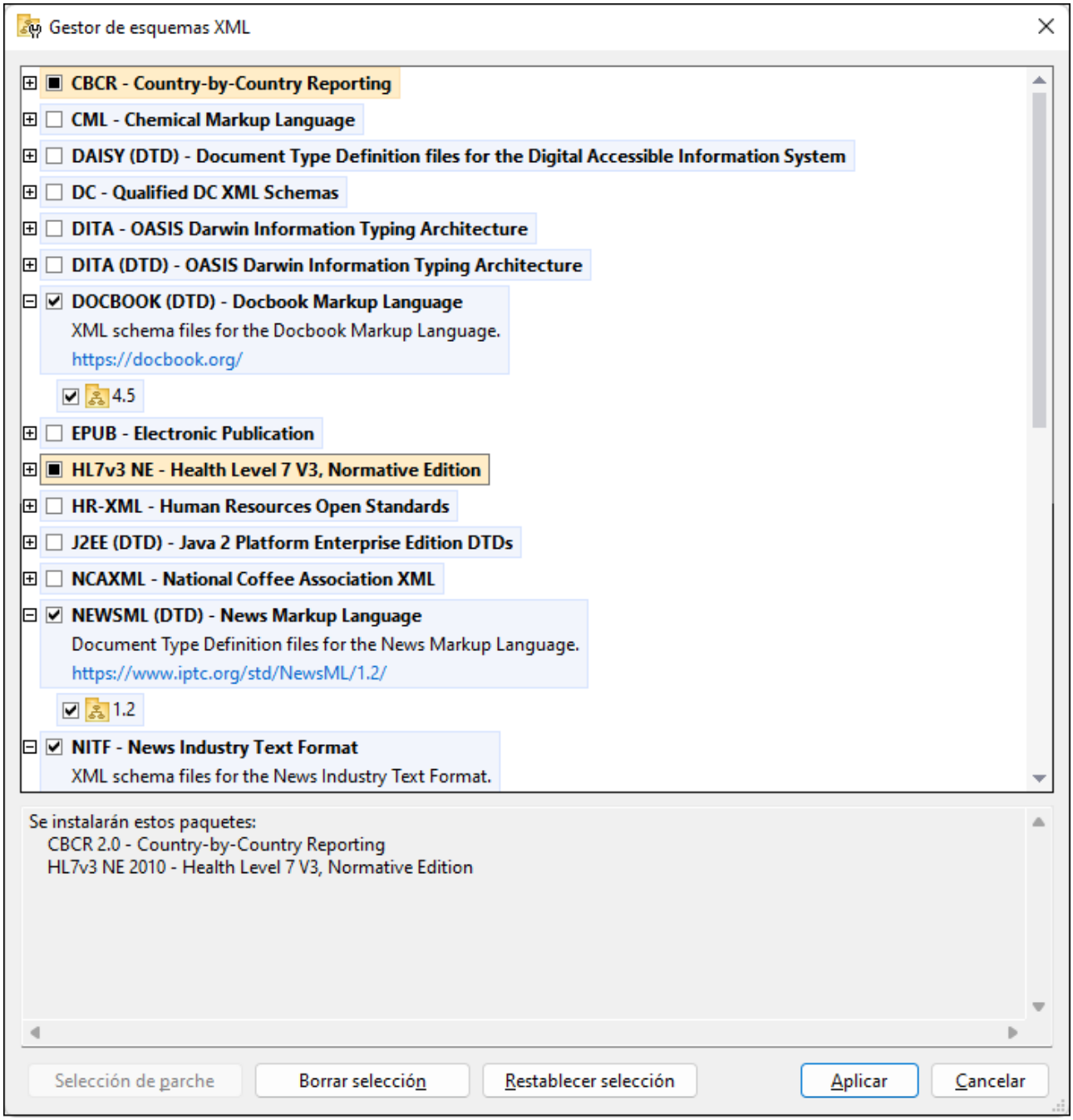
Si necesita crear prefijos para nombres de atributo (p. ej. `<number prod:id="prod557">557</number>`) puede habilitar el acceso dinámico a los atributos del nodo (*véase* *Asignar nombres de nodos* 850) o editar el esquema para que contenga un atributo `prod:id` para `<number>`.

Eliminar espacios de nombre
Para eliminar una declaración de espacio de nombres haga clic con el botón derecho en el nodo `ns:namespace` y seleccione **Quitar espacio de nombres** en el menú contextual.

4.1.8 Gestor de esquemas

El Gestor de esquemas XML es una herramienta que ofrece una forma centralizada de instalar y administrar esquemas XML (DTDs para XML y esquemas XML) para usarlos en todas las aplicaciones de Altova compatibles con XML, incluido MapForce.

- En Windows, el gestor tiene una interfaz gráfica del usuario (*imagen siguiente*) a la que también puede acceder desde la línea de comandos. (Las aplicaciones de escritorio de Altova solo están disponibles para Windows; *consulte la lista siguiente*.)
- En Linux y macOS el Gestor de esquemas solo está disponible en la línea de comandos. (Las aplicaciones de escritorio de Altova están disponibles para Windows, Linux y macOS; *consulte la lista siguiente*.)



Aplicaciones de Altova que funcionan con el Gestor de esquemas

Aplicaciones de escritorio (solo para Windows)	Aplicaciones de servidor (Windows, Linux, macOS)
XMLSpy (todas las ediciones)	RaptorXML Server, RaptorXML+XBRL Server
MapForce (todas las ediciones)	StyleVision Server

 © 2016-2022 Altova GmbH

StyleVision (todas las ediciones)	

Instalación y desinstalación del Gestor de esquemas

El Gestor de esquemas se instala automáticamente al instalar cualquiera de las aplicaciones de Altova compatibles con XML o el Altova Mission Kit Enterprise Edition (*véase la tabla de más arriba*).

También se elimina automáticamente si desinstala todas las aplicaciones de Altova compatibles con XML o el Altova Mission Kit Enterprise Edition del equipo.

Características de Gestor de esquemas

El Gestor de esquemas permite:

- Ver los esquemas XML que hay instaladas en su equipo y comprobar si hay versiones nuevas para descargar.
- Descargar las versiones más recientes de los esquemas XML independientemente del ciclo de versiones de Altova. Altova guarda todos los esquemas en un sistema de almacenamiento en línea al que tiene acceso el Gestor de esquemas y desde donde puede descargarlas tan pronto como estén disponibles.
- Instalar o desinstalas cualquiera de las múltiples versiones de un esquema en concreto (o todas ellas, si las necesita).
- Un solo esquema XML representa un "paquete", pero puede tener dependencias en otros esquemas. Al instalar o desinstalar un esquema, se detectan e instalan o desinstalan también automáticamente todas sus dependencias. La interfaz gráfica del usuario (o la línea de comandos, en su caso) le informa cuando se añaden o eliminan esquemas.
- Los esquemas XML administradas con el Gestor de esquemas pueden usar el catálogo XML, que permite resolver referencias a URI en documentos de instancia o esquema desde archivos locales, en vez de a través de Internet.
- Todos los esquemas principales están incluidos en Gestor de esquemas y se actualizan de forma periódica a la versión más reciente. De esta forma puede administrar todos los esquemas desde un punto común y tenerlos siempre listos para las aplicaciones de Altova que las usan.
- Los cambios que se realizan en el Gestor de esquemas afectan a todos los productos de Altova que estén instalados en ese equipo.

Funcionamiento

Altova mantiene un almacenamiento en línea donde guarda todos los esquemas XML de los productos de Altova. Este almacenamiento se actualiza de forma periódica, por ejemplo, poco después de que las organizaciones correspondientes publiquen las versiones nuevas de los esquemas respectivos. Al ejecutar Gestor de esquemas desde la interfaz gráfica del usuario aparece información sobre los esquemas más recientes disponibles en un cuadro de diálogo en el que puede visualizarlos, instalarlos, actualizarlos o desinstalarlos.

También puede instalar los esquemas de otra manera. En el sitio web de Altova (https://www.altova.com/schema-manager) puede seleccionar el esquema y los esquemas dependientes de este que quiere instalar. El sitio web prepara un archivo de tipo `.altova_xmlschemas` que puede descargar y que contiene la información sobre los esquemas seleccionados. Al hacer doble clic en este archivo o pasarlo a Gestor de esquemas desde la línea de comandos como argumento del comando `install` 154, Gestor de esquemas instala los esquemas que contiene.

Memoria caché local: seguimiento de esquemas
Independientemente de cómo se instalen los esquemas, toda la información sobre los esquemas instalados se almacena en una ubicación centralizada de su equipo, el directorio caché. El directorio caché local está en:

Windows	`C:\ProgramData\Altova\pkgs\.cache`
Linux	`/var/opt/Altova/pkgs\.cache`
macOS	`/var/Altova/pkgs`

El directorio caché local se actualiza automáticamente de vez en cuando para que el estado más actual del equipo corresponda con el del almacenamiento en línea. Más concretamente, el caché se actualiza:

- al ejecutar el Gestor de esquemas.
- al ejecutar MapForce por primera vez en un mismo día natural.
- si MapForce ya se está ejecutando, el directorio caché se actualiza cada 24 horas.
- también puede actualizar el caché local desde el almacenamiento en línea manualmente ejecutando el comando de actualización **update** 157 desde la línea de comandos.

Si instala o desinstala esquemas, el directorio caché local se actualiza automáticamente con información sobre los esquemas disponibles e instalados, además de con los propios archivos de esquema.

No modifique la memoria caché manualmente

El directorio caché local se mantiene automáticamente en base a los esquemas que instale o desinstale; no debe modificarlo ni eliminarlo manualmente. Si necesita restaurar el Gestor de esquemas a su estado original, ejecute el comando **reset** 155 desde la línea de comandos y después ejecute el comando **initialize** 153.

4.1.8.1 Ejecutar el gestor de esquemas

Interfaz gráfica del usuario

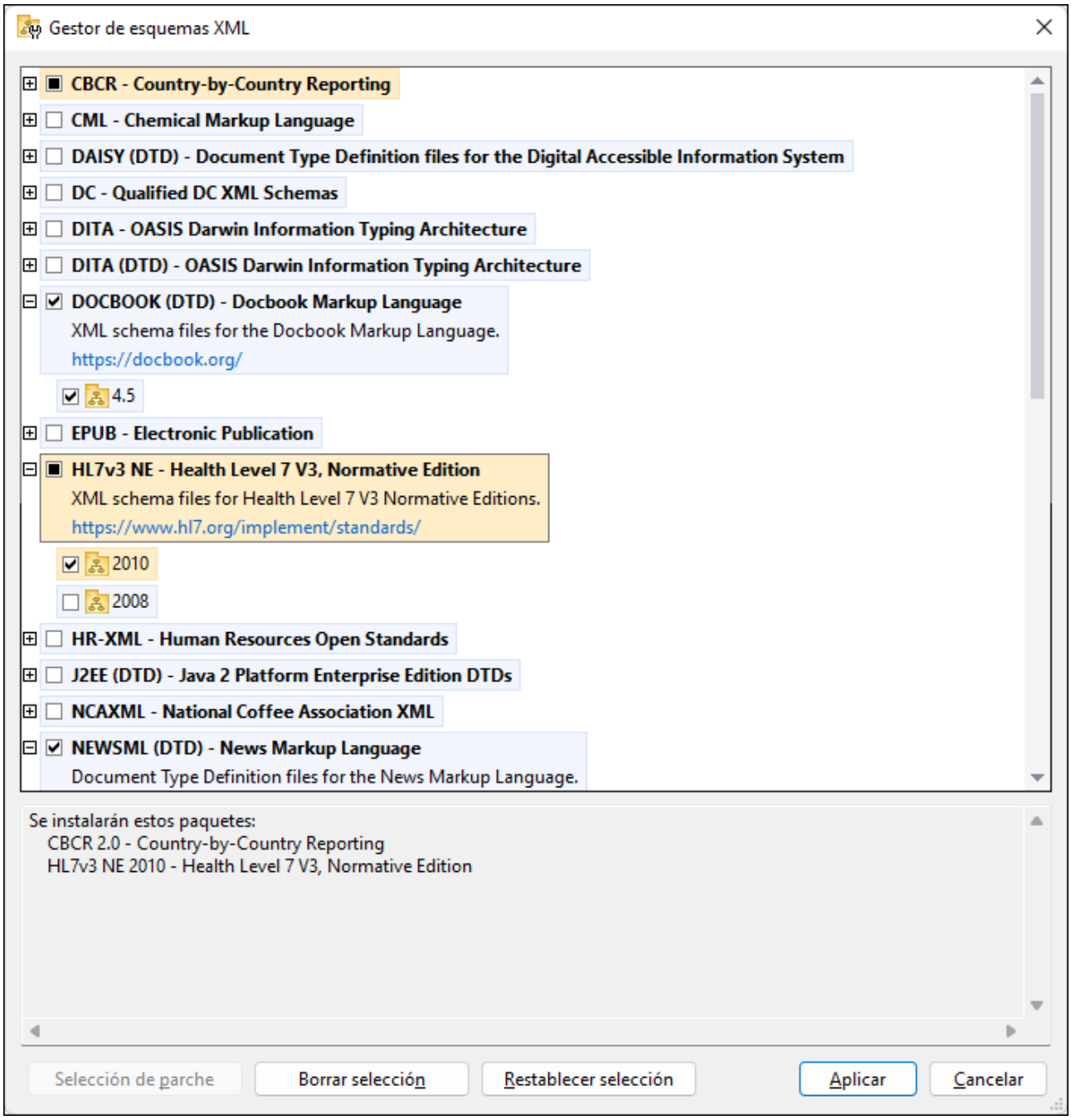
Hay varias formas de acceder a la IGU del Gestor de esquemas:

- *Durante la instalación de MapForce:* al final del proceso de instalación, seleccione la casilla *Invocar al Gestor de esquemas* para acceder directamente a la IGU del gestor de esquemas XML. Con él puede instalar esquemas durante el proceso de instalación de su aplicación de Altova.
- *Después de la instalación de MapForce:* una vez haya instalado la aplicación puede acceder al Gestor de esquemas en cualquier momento desde el comando de menú Herramientas | **Gestor de esquemas XML**.
- Mediante el archivo **`.altova_schemas`** que descargó del *sitio web de Altova:* haga doble clic en el archivo para ejecutar Gestor de esquemas, que instalará los esquemas que haya seleccionado.

Cuando se abra la IGU del Gestor de esquemas (*imagen siguiente*) podrá ver en ella los esquemas que ya se han instalado. Si quiere instalar más solo tiene que seleccionarlas, y al contrario si quiere desinstalar alguna. Una vez haya terminado, puede aplicar los cambios. los esquemas que se vayan a instalar o desinstalar

 © 2016-2022 Altova GmbH

aparecerán resaltados y un mensaje le avisará de los cambios que está a punto de hacer en la ventana *Mensajes*, en la parte inferior de la ventana de Gestor de esquemas (*véase imagen*).



Interfaz de la línea de comandos

Para ejecutar el Gestor de esquemas desde una interfaz de la línea de comandos debe usar su archivo ejecutable, **`xmlschemamanager.exe`**.

Puede encontrar este archivo:

- *en Windows:* `C:\ProgramData\Altova\SharedBetweenVersions`
- *en Linux o macOS (solo para aplicaciones de servidor):* `%INSTALLDIR%/bin`, donde `%INSTALLDIR%` es el directorio de instalación del programa.

Puede usar cualquiera de los comandos de la referencia de la línea de comandos 152, a continuación.

Para mostrar al ayuda de los comandos ejecute:

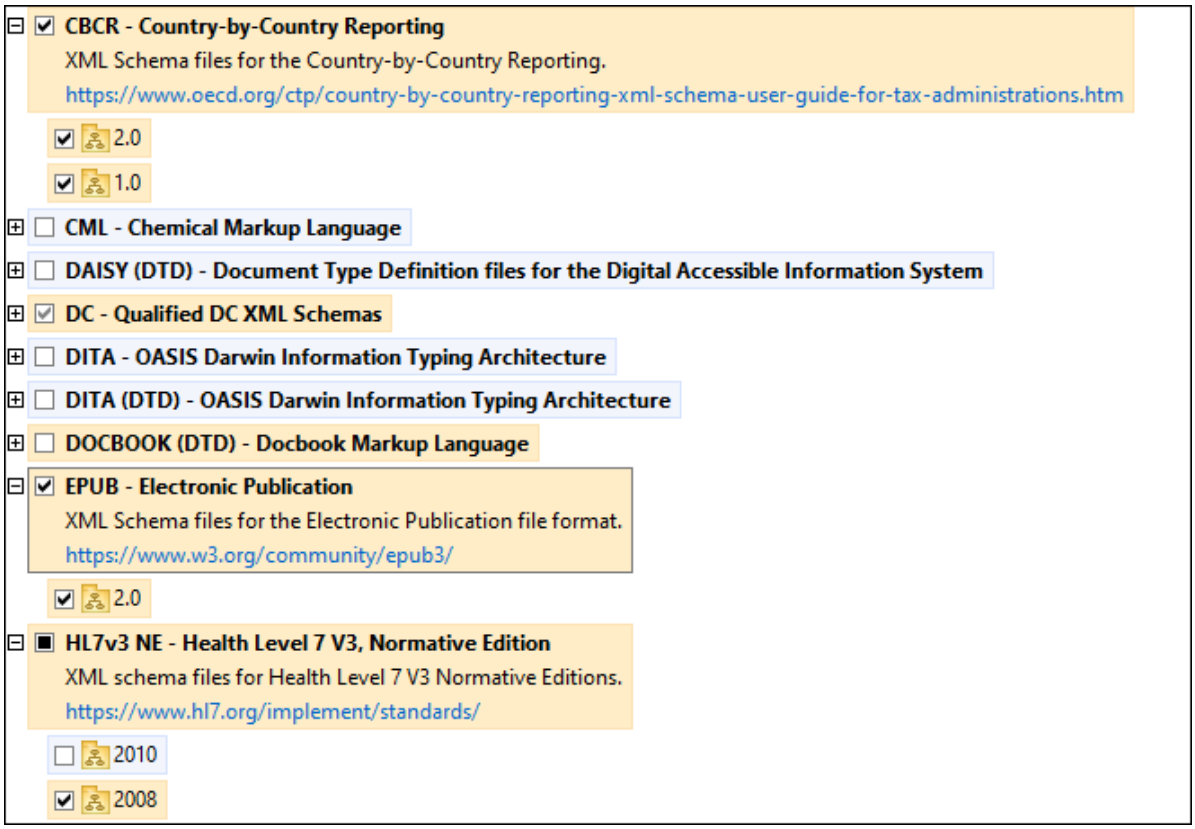
- *en Windows:* `xmlschemamanager.exe --help`
- *en Linux o macOS (solo para aplicaciones de servidor):* `sudo ./xmlschemamanager --help`

4.1.8.2 Categorías de estado

Gestor de esquemas diferencia los esquemas que administra entre:

- *Esquemas instalados:* estos aparecen en la IGU con sus casillas marcadas (*en la imagen siguiente las versiones marcadas de los esquemas EPUB y HL7v3 NE son las que están instaladas*). Si se seleccionan todas las versiones de un esquema, en la casilla del esquema aparece una marca de verificación. Si hay al menos un esquema sin seleccionar, en la en la casilla del esquema aparece un cuadrado negro. Para **desinstalar** un esquema debe desmarcar la casilla correspondiente (*en la imagen siguiente, el DTD DocBook está instalado y su casilla se ha desmarcado, es decir, se va a desinstalar*).
- *Esquemas disponibles no instalados:* estos aparecen en la IGU con las casillas correspondientes sin seleccionar. Para **instalar** esquemas, marque la casilla correspondiente.

 © 2016-2022 Altova GmbH



- *Esquemas que pueden actualizarse:* son los que han sido revisados por sus emisores. Aparecen indicados en la IGU con el icono (*imagen anterior*). Puede aplicar **parches** al esquema seleccionado con la revisión que esté disponible.

Puntos importantes

- En la imagen anterior se han marcado los esquemas CBCR. Las que tienen un fondo azul ya están instaladas. Las que tienen el fondo amarillo no están instaladas pero se han seleccionado para instalarlas. Observe que el esquema HL7v3 NE 2010 no está instalada ni se ha seleccionado para instalarlo.
- Al ejecutar el Gestor de esquemas desde la línea de comandos puede usar el comando `list` con distintas opciones para ver distintas categorías de esquemas:

`xmlschemamanager.exe list`	Muestra todas los esquemas instalados y disponibles; también indica qué esquemas se pueden actualizar
`xmlschemamanager.exe list -i`	Muestra solo los esquemas instalados; también indica qué esquemas se pueden actualizar
`xmlschemamanager.exe list -u`	Muestra qué esquemas se pueden actualizar

Nota: en Linux y macOS `use sudo ./xmlschemamanager list`

4.1.8.3 Aplicar parches o instalar un esquema

Aplicar un parche a un esquema instalado

A veces los emisores de los esquemas XML generan parches. Cuando el Gestor de esquemas XML detecta que hay parches disponibles, estos aparecen en las listas de esquemas, desde donde puede instalarlos.

En la IGU

Los parches se indican con el icono . (*Consulte también el apartado anterior sobre categorías de esquemas* 148.) Si hay parches disponibles se habilita el botón **Seleccionar parches**. Haga clic en él para seleccionar y preparar los parches. En la IGU, el icono de los esquemas correspondientes cambia de a y el cuadro de diálogo le informa de qué parches se van a aplicar. Las listas del panel principal y las del panel Mensajes están ordenadas alfabéticamente, por lo que puede ver y revisar los esquemas antes de aplicar los parches. Una vez esté listo para instalar los parches seleccionados, haga clic en **Aplicar**.

En la línea de comandos

Para aplicar un parche desde lal ínea de comandos:

1. Ejecute el comando `list -u` 155. Aparece una lista con los esquemas para las que hay parches disponibles.
1. Ejecute el comando `upgrade` 158 para instalar todos los parches.

Instalar un esquema disponible

Para instalar esquemas puede usar la IGU del Gestor de esquemas o enviar las instrucciones al Gestor de esquemas desde la línea de comandos.

Nota: si el esquema actual tiene dependencias en otros esquemas, también se instalan (o desinstalan, según el caso) los esquemas dependientes.

En la IGU

Para instalar esquemas con la IGU del Gestor de esquemas, seleccione los esquemas que quiere instalar y haga clic en **Aplicar**.

También puede seleccionar los esquemas que quiere instalar en el sitio web de Altova y generar desde allí un archivo **`.altova_schemas`**. Al hacer doble clic en este archivo se abre el Gestor de esquemas con los esquemas que indicó preseleccionados. Solo tiene que hacer clic en **Aplicar**.

En la línea de comandos

Para instalar esquemas desde la línea de comandos ejecute el comando **`install`**:

```
xmlschemamanager.exe install [opciones] Schema+
```

donde **`FILTER`** es el esquema (o los esquemas) que quiere instalar o un archivo **`.altova_schemas`**. Para hacer referencia a un esquema se usa un identificados con el formato **`<nombre>-<versión>`** que aparece junto a cada esquema que muestra el comando `list` 155. Puede introducir tantos esquemas como quiera. Para más detalles consulte la descripción del comando `install` 154.

Nota: en Linux o macOS, use el comando **`sudo ./xmlschemamanager`**.

 © 2016-2022 Altova GmbH

Instalar un esquema requerido
Si ejecuta un comando en MapForce y MapForce descubre que uno de los esquemas que necesita para ejecutar el comando falta o está incompleta, el Gestor de esquemas incluirá información sobre ese componente de esquema que falta. Entonces puede aplicar el parche indicado y/o instalar el esquema que falta.

Siempre puede ver todos los esquemas instalados previamente ejecutando el Gestor de esquemas desde **Herramientas | Gestor de esquemas**.

4.1.8.4 Desinstalar o restaurar esquemas

Desinstalar un esquema

Para desinstalar esquemas puede usar la IGU del Gestor de esquemas o enviar las instrucciones al Gestor de esquemas desde la línea de comandos.

Nota: si el esquema actual tiene dependencias en otros esquemas, también se instalan (o desinstalan, según el caso) los esquemas dependientes.

En la IGU
Para desinstalar esquemas con la IGU del Gestor de esquemas, seleccione los esquemas que quiere desinstalar y haga clic en **Aplicar**. Los esquemas seleccionadas y sus dependencias se desinstalarán.

Para desinstalar todos los esquemas haga clic en **Deseleccionar todas** y haga clic en **Aplicar**.

En la línea de comandos
Para desinstalar esquemas desde la línea de comandos ejecute el comando `uninstall` 156:

```
xmlschemamanager.exe uninstall [options] FILTER+
```

donde **FILTER** es el esquema (o los esquemas) que quiere desinstalar o un archivo **.altova_schemas**. Para hacer referencia a un esquema se usa un identificador con el formato **<nombre>-<versión>** que aparece junto a cada esquema que muestra el comando `list` 155. Puede introducir tantos esquemas como quiera. Para más detalles consulte la descripción del comando `uninstall` 156.

Nota: en Linux o macOS, use el comando **sudo ./xmlschemamanager**.

Restaurar el Gestor de esquemas

Puede restaurar el Gestor de esquemas, es decir, eliminar todos los esquemas instaladas, así como el directorio caché.

- En la IGU, haga clic en **Restaurar selección**.
- En la línea de comandos, use el comando `reset` 155.

Una vez haya ejecutado este comando, asegúrese de que ejecuta también el comando `initialize` 153 para recrear el directorio caché. También puede ejecutar el comando `reset` 155 con la opción `-i`.

Recuerde que `reset -i` 155 restaura la instalación original del producto, por lo que es recomendable ejecutar el comando `update` 157 después de restaurar el gestor. Puede ejecutar el comando `reset` 155 con las opciones `-i` o `-u`.

4.1.8.5 Interfaz de la línea de comandos (ILC)

Para llamar a Gestor de esquemas desde la línea de comandos necesita saber la ruta del ejecutable. Por defecto, el ejecutable del Gestor de esquemas se encuentra en:

```
C:\ProgramData\Altova\SharedBetweenVersions\XMLSchemaManager.exe
```

Nota: en los sistemas Linux y macOS una vez haya cambiado el directorio al que contiene el ejecutable, puede llamar al ejecutable con **`sudo ./xmlschemamanager`**.El prefijo **`./`** indica que el ejecutable está en el directorio actual. El prefijo **`sudo`** indica que el comando se debe ejecutar con derechos de administrador.

Sintaxis de la línea de comandos

La sintaxis general para usar la línea de comandos es:

```
<exec> -h | --help | --version | <command> [opciones] [argumentos]
```

En el código anterior la barra vertical **`|`** separa elementos que se excluyen mutuamente. Los corchetes **`[]`** indican elementos opcionales. Básicamente, puede teclear la ruta del ejecutable seguida por las opciones `--h`, `--help` o `--version`, o por un comando. Cada comando puede tener opciones y argumentos. Los comandos se describen en los apartados siguientes.

4.1.8.5.1 help

Este comando ofrece ayuda contextual sobre los comandos del ejecutable del Gestor de esquemas.

Sintaxis

```
<exec> help [command]
```

Donde `[command]` es un argumento opcional que indica cualquier nombre válido de comando.

Tenga en cuenta que:

- Puede invocar la ayuda tecleando un comando seguido por `--h` or `--help`, por ejemplo: **`<exec> list -h`**
- Puede invocar la ayuda general tecleando `--h` o `--help` directamente después del ejecutable, por ejemplo:

 © 2016-2022 Altova GmbH

Ejemplo

Este comando muestra la ayuda del comando `list`:

```
xmlschemamanager help list
```

4.1.8.5.2 info

Este comando muestra información detallada sobre cada uno de los esquemas dados como argumento. Esta información incluye el título, la versión, la descripción, el editor y las referencias de las dependencias.

Sintaxis

```
<exec> info [options] Schema+
```

- El argumento **`Schema`** es el nombre de un esquema o parte del nombre de un esquema. (Para ver el ID de un paquete de esquemas y la información relativa a su estado de instalación use el comando `list` 155.)
- Use **`<exec> info -h`** para ver la ayuda sobre este comando en la línea de comandos..

Ejemplo

Este comando muestra información detallada sobre los esquemas **`DocBook-DTD`** y **`NITF`**:

```
xmlschemamanager info doc nitf
```

4.1.8.5.3 initialize

Este comando inicializa el entorno del Gestor de esquemas y crea un directorio caché donde se guardan todos los esquemas localmente. El Gestor de esquemas se inicializa automáticamente la primera vez que instale una aplicación de Altova compatible con él, por lo que normalmente no es necesario ejecutar este comando. Por lo general solo es necesario ejecutarlo después de haber ejecutado el comando `reset`.

Sintaxis

```
<exec> initialize | init [opciones]
```

Opciones

Estas son las opciones del comando **`initialize`**:

`--help, --h`	Muestra la ayuda sobre este comando en la línea de comandos.
`--silent, --s`	Muestra solamente los mensajes de error. El valor predeterminado es **`false`**.

`--verbose, --v`	Muestra información suplementaria durante la ejecución. El valor predeterminado es **`false`**.

Ejemplo

Este comando inicializa el Gestor de esquemas:

```
xmlschemamanager initialize
```

4.1.8.5.4 install

Este comando instala una o más esquemas.

Sintaxis

```
<exec> install [options] Schema+
```

Para indicar varios esquemas, repita el argumento `Schema` tantas veces como sea necesario.

El argumento de `Schema` puede ser:

1. Un identificador de esquema en el formato **`<name>-<version>`**, por ejemplo: **`cbcr-2.10`**). Para ver todos los identificadores de esquemas y sus versiones ejecute el comando `list` 155. También puede usar el nombre de el esquema abreviado, si este es único, por ejemplo **`docbook`**. Si usa una abreviación del nombre se desinstalan todos los esquemas que contengan esa abreviación.
2. La ruta de acceso a un archivo **`.altova_schemas`** descargado desde el sitio web de Altova. Para más información sobre estos archivos consulte la *Introducción al Gestor de esquemas: funcionamiento* 143.

Opciones

Estas son las opciones del comando **`install`**:

`--help, --h`	Muestra la ayuda sobre este comando en la línea de comandos.
`--silent, --s`	Muestra solamente los mensajes de error. El valor predeterminado es **`false`**.
`--verbose, --v`	Muestra información suplementaria durante la ejecución. El valor predeterminado es **`false`**.

Ejemplo

Este comando instala el esquema CBCR 2.0 (Country-By-Country Reporting) y el DTD DocBook más reciente:

```
xmlschemamanager install cbcr-2.0 docbook
```

 © 2016-2022 Altova GmbH

4.1.8.5.5 list

Use este comando para ver los esquemas del Gestor de esquemas; tiene varias opciones:

- lista de todos los esquemas disponibles
- lista de esquemas específicos
- lista de los esquemas instalados
- lista de los esquemas que se pueden actualizar.

Sintaxis

```
<exec> list | ls [options] Schema?
```

Si no se indica ningún argumento **Schema** la lista incluye todos los esquemas. De lo contrario la lista incluye los esquemas indicados en las opciones (*véase el ejemplo de más abajo*). Recuerde que puede usar el argumento **Schema** tantas veces como quiera.

Opciones
Estas son las opciones del comando **list**:

`--help, --h`	Muestra la ayuda sobre este comando en la línea de comandos.
`--installed, --i`	Muestra solamente los esquemas instaladas. El valor predeterminado es **false**.
`--upgradeable, --u`	Muestra solamente los esquemas para las que hay disponible una versión más reciente (parches). El valor predeterminado es **false**.

Ejemplos

- Para ver todos los esquemas disponibles ejecute: **xmlschemamanager list**
- Para ver solamente los esquemas instaladas ejecute: **xmlschemamanager list -i**
- Para ver todos los esquemas cuyos nombres contienen "doc" o "nitf" ejecute: **xmlschemamanager list doc nitf**

4.1.8.5.6 reset

Este comando elimina todos los esquemas instalados, así como el directorio de caché. Este comando elimina todos los esquemas instalados y su información. Una vez haya ejecutado este comando, asegúrese de que ejecuta el comando `initialize` 153 para volver a crear el directorio de caché. También puede ejecutar el comando `reset` con la opción `-i`. Tenga en cuenta que `reset -i` restaura la instalación original del producto, por lo que se recomienda ejecutar también el comando `update` 157 después de una restauración. También puede ejecutar el comando `reset` con las opciones `-i` y `-u`.

Sintaxis

```
<exec> reset [opciones]
```

Opciones
Estas son las opciones del comando **reset**:

`--help, --h`	Muestra la ayuda sobre este comando en la línea de comandos.
`--init, --i`	Inicializa el entorno del Gestor de esquemas XML después de una restauración. El valor predeterminado es **false**.
`--silent, --s`	Muestra solamente los mensajes de error. El valor predeterminado es **false**.
`--update, --u`	Inicializa y actualiza el entorno del Gestor de esquemas XML después de una restauración. El valor predeterminado es **false**.
`--verbose, --v`	Muestra información suplementaria durante la ejecución. El valor predeterminado es **false**.

Ejemplos

- Para restaurar el Gestor de esquemas, ejecute: **xmlschemamanager reset**
- Para restaurar el Gestor de esquemas e inicializarlo, ejecute: **xmlschemamanager reset -i**
- Para restaurar el Gestor de esquemas, inicializarlo y actualizar la lista de esquemas, ejecute: **xmlschemamanager reset -i -u**

4.1.8.5.7 uninstall

Este comando desinstala una o más esquemas. Por defecto, cualquier esquema a la que haga referencia el esquema actual también se desinstala. Para desinstalar solamente el esquema actual y mantener aquellas a las que se hace referencia, use la opción `--k`.

Sintaxis

```
<exec> uninstall [opciones] Schema+
```

Para indicar varios esquemas, repita `FILTER` tantas veces como sea necesario.

El argumento de **Schema** puede ser:

- Un identificador de esquema en el formato **<name>-<version>**, por ejemplo: **eba-2.10**). Para ver todos los identificadores de esquemas y sus versiones ejecute el comando `list -i` 155. También puede usar el nombre del esquema abreviado, si este es único, por ejemplo **eba**. Si usa una abreviación del nombre se desinstalan todos los esquemas que contengan esa abreviación.
- La ruta de acceso a un archivo **.altova_taxonomies** descargado desde el sitio web de Altova. Para más información sobre estos archivos consulte la *Introducción al Gestor de esquemas: funcionamiento* 143.

 © 2016-2022 Altova GmbH

Opciones
Estas son las opciones del comando **`uninstall`**:

`--help, --h`	Muestra la ayuda sobre este comando en la línea de comandos.
`--keep-references, --k`	Si usa esta opción, los esquemas referenciados no se desinstalan. El valor predeterminado es **`false`**.
`--silent, --s`	Muestra solamente los mensajes de error. El valor predeterminado es **`false`**.
`--verbose, --v`	Muestra información suplementaria durante la ejecución. El valor predeterminado es **`false`**.

Ejemplo

Este comando desinstala los esquemas CBCR 2.0 y EPUB 2.0:

```
xmlschemamanager uninstall cbcr-2.0 epub-2.0
```

Este comando desinstala el esquema **`eba-2.10`** pero no los esquemas a los que hace referencia:

```
xmlschemamanager uninstall --k cbcr-2.0
```

4.1.8.5.8 update

Este comando consulta la lista de esquemas disponibles en el almacenamiento en línea y actualiza el directorio de caché local. Esta información se actualiza de forma implícita, por lo que no es necesario ejecutar este comando a no ser que haya ejecutado `reset` 155 e `initialize` 153.

Sintaxis

```
<exec> update [opciones]
```

Opciones
Estas son las opciones del comando `update`:

`--help, --h`	Muestra la ayuda sobre este comando en la línea de comandos.
`--silent, --s`	Muestra solamente los mensajes de error. El valor predeterminado es **`false`**.
`--verbose, --v`	Muestra información suplementaria durante la ejecución. El valor predeterminado es **`false`**.

Ejemplo

Este comando actualiza la lista de esquemas:

```
xmlschemamanager update
```

4.1.8.5.9 upgrade

Este comando actualiza todos los esquemas aptos para la versión *parche* más reciente disponible. Puede identificar cuáles lo son con el comando `list -u`.

Nota: el comando `upgrade` eliminaría un esquema obsoleto si no hay ninguna versión disponible.

Sintaxis

```
<exec> upgrade [opciones]
```

Opciones
Estas son las opciones del comando `upgrade`:

`--help, --h`	Muestra la ayuda sobre este comando en la línea de comandos.
`--silent, --s`	Muestra solamente los mensajes de error. El valor predeterminado es **`false`**.
`--verbose, --v`	Muestra información suplementaria durante la ejecución. El valor predeterminado es **`false`**.

© 2016-2022 Altova GmbH

4.2 Bases de datos

Sitio web de Altova: Asignación de bases de datos

MapForce 2023 es totalmente compatible con la asignación de datos de bases de datos con archivos XML, archivos planos y otros formatos de base de datos. Además, con la edición Enterprise Edition podrá crear asignaciones de datos entre bases de datos y formatos EDI, Excel 2007+, JSON, XBRL y servicios web.

Las bases de datos compatibles y sus objetos raíz aparecen a continuación. Altova procura ofrecer compatibilidad con otras bases de datos, pero sólo se garantiza una conexión y un procesamiento correctos con las bases de datos de la lista. Si usa la versión de 64 bits de MapForce, compruebe que tiene acceso a los controladores de BD de 64 bits de la BD a la que quiere conectarse.

Base de datos	Observaciones
Firebird 2.x, 3.x	
IBM DB2 8.x, 9.x, 10.x, 11.x	
IBM Db2 for i 6.x, 7.4	Los archivos lógicos son compatibles y se muestran en vistas.
IBM Informix 11.70 y superior	Informix es compatible con conexiones vía ADO, JDBC y ODBC. La implementación no es compatible con tipos de datos de objetos de gran dimensión en cualquiera de los idiomas de generación del código. MapForce generará un mensaje de error (durante el proceso de generación del código) si se usa cualquiera de este tipo de datos.
MariaDB 10 y superior	
Microsoft Access 2003 y superior	En el momento de escribir esta documentación (principios de septiembre de 2019) no hay ningún Microsoft Access Runtime disponible para Access 2019. Solo puede conectarse a la BD de Access 2019 con productos de Altova si tiene instalado Microsoft Access 2016 Runtime y solamente si la BD no usa el tipo de datos "Large Number" (número grande).
Microsoft Azure SQL Database	SQL Server 2016 codebase
Microsoft SQL Server 2005 y superior Microsoft SQL Server para Linux	
MySQL 5 y superior	
Oracle 9i y superior	
PostgreSQL 8 y superior	Son compatibles todas las conexiones PostgreSQL, tanto nativas como basadas en controladores, a través de interfaces como

	ODBC o JDBC. Las conexiones nativas no necesitan controladores.
Progress OpenEdge 11.6	
SQLite 3.x	Las conexiones SQLite son conexiones nativas y directas compatibles con el archivo de base de datos de SQLite. No se precisan controladores separados.
Sybase ASE 15, 16	
Teradata 16	Son compatibles las conexiones vía ADO.NET, JDBC y ODBC. Cuando una asignación inserta datos en una tabla de BD, los campos de identidad generados por la base de datos no son compatibles.

Asignaciones de BD en distintos entornos de ejecución

Cuando genera código de programa a partir de una asignación, cuando compila una asignación en archivos de ejecución de MapForce Server o cuando implementa una asignación en FlowForce Server, los detalles de la conexión de BD que se guardan con los archivos generados están adaptados a los controladores necesarios o compatibles para el entorno de destino elegido, como se aprecia en la siguiente tabla. Por ejemplo, si el lenguaje de transformación es Java, las conexiones ADO se convierten en JDBC al generar código a partir de la asignación.

Si ejecuta la asignación en un entorno distinto a MapForce deberá asegurarse de que los detalles de la conexión de BD son los correctos para el equipo en el que se va a ejecutar la asignación (por ejemplo, si está instalado el controlador de BD correspondiente, si la ruta de acceso a la BD es correcta, si se puede acceder al servidor de BD, etc.).

Algunos tipos de conexión de BD no se admiten en ciertos entornos de destino, como mostramos en la siguiente tabla.

Tipo de conexión/Entorno de ejecución	**C#**	**C++**	**Java**	**MapForce Server en Windows**	**MapForce Server en Linux/Mac**
ADO	puente ADO	Como está	Transformada en JDBC	Como está	Transformada en JDBC
ADO.NET	Como está	Definida por el usuario	Transformada en JDBC	Como está	Transformada en JDBC
JDBC	Definida por el usuario	Definida por el usuario	Como está	Como está	Como está
ODBC	puente ODBC	puente ODBC	Transformada en JDBC	Como está	Transformada en JDBC

 © 2016-2022 Altova GmbH

Native PostgreSQL	No es compatible	No es compatible	No es compatible	Como está	Como está
Native SQLite	No es compatible	No es compatible	No es compatible	Como está	Como está

Leyenda de la tabla:

- "Como está" significa que el tipo de conexión de la BD (por ejemplo, JDBC, ODBC) es la que defina MapForce.
- "Transrofmada en JDBC" significa que la conexión de BD se transformará en una URL de conexión de BD similar a la JDBC.
- "Puente ADO" o "puente ODBC" significa que la cadena de la conexión permanece como la haya definido MapForce, pero el código generado usará una clase adecuada que actúe como puente ADO (u ODBC), por ejemplo **System.Data.OleDb.OleDbConnection** o **System.Data.Odbc.OdbcConnection**.
- "Definida por el usuario" significa que, para que la conexión funcione en el código generado, necesitará introducir manualmente los detalles de la conexión en el cuadro de diálogo Configurar componentes de BD 278. En función del caso, estos detalles de conexión se deben introducir como **ADO/OLE-DB-specific settings** o como **JDBC-specific settings**.

Consulte también:

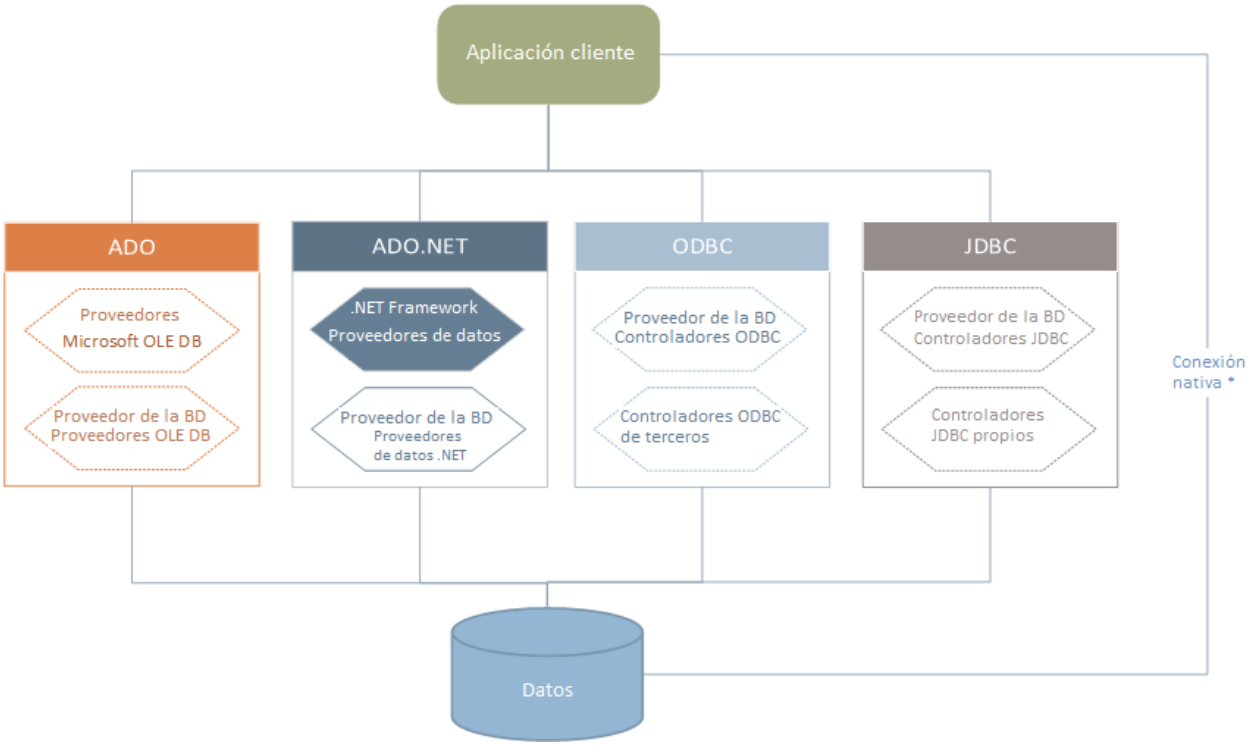
- Preparar asignaciones para ejecutarlas en servidores 949
- Compilar asignaciones en archivos de ejecución de MapForce Server 955
- Implementar asignaciones en FlowForce Server 958
- Generador de código 1067

4.2.1 Conectarse a un origen de datos

En su definición más sencilla, una base de datos es un archivo local como un archivo de base de datos Microsoft Access o SQLite. En casos más complejos, una base de datos puede residir en un servidor de bases de datos remoto o de la red que no tienen por qué usar el mismo sistema operativo que la aplicación que se conecta a la BD y que consume los datos. Por ejemplo, mientras que MapForce se puede ejecutar en sistemas operativos Windows, puede que la base de datos a la que desea acceder (p. ej. una base de datos MySQL) esté en un equipo Linux.

Para interactuar con los diferentes tipos de bases de datos, MapForce se sirve de las interfaces de conexión de datos y los controladores de BD disponibles en su sistema operativo o publicados por los principales proveedores de BD. La tecnología de base de datos evoluciona constantemente y por tanto consideramos que este mecanismo garantiza la mayor compatibilidad y flexibilidad en las principales plataformas.

En el siguiente diagrama puede ver un resumen de las opciones de conectividad de datos que pueden existir entre MapForce (*aplicación cliente*) y un almacén de datos (que puede ser un archivo o un servidor de base de datos).



** Las bases de datos SQLite, PostgreSQL, CouchDB y MongoDB admiten conexiones nativas directas. Para conectarse a este tipo de bases de datos no es necesario tener instalado controladores específicos.*

Por tanto, tal y como muestra el diagrama, desde MapForce puede acceder a los principales tipos de BD con estas tecnologías de acceso de datos:

- ADO (Microsoft® ActiveX® Data Objects), que a su vez utiliza OLE DB
- ADO.NET (un conjunto de bibliotecas disponibles en Microsoft .NET Framework que permiten la interacción con datos)
- JDBC (Java Database Connectivity)
- ODBC (Open Database Connectivity)

Nota: algunos proveedores de ADO.NET no son compatibles o tienen una compatibilidad limitada. Consulte el apartado *Notas sobre compatibilidad con ADO.NET* 180 para obtener más información.

Tecnologías de acceso a datos

La interfaz de conexión de datos que se debe utilizar dependerá de la infraestructura de software con la que cuente. Lo normal es elegir una tecnología de acceso a datos y un controlador de base de datos que se integre bien con el sistema de base de datos al que desea conectarse. Por ejemplo, para conectarse con una base de datos Microsoft Access 2013, puede generar una cadena de conexión ADO que utilice un proveedor nativo como **Microsoft Office Access Database Engine OLE DB Provider**. Para conectarse a Oracle, por su parte, lo mejor será descargar e instalar las interfaces JDBC, ODBC o ADO .NET más recientes del sitio web de Oracle.

 © 2016-2022 Altova GmbH

Lo más probable es que los controladores de los productos Windows (como Microsoft Access o SQL Server) ya estén disponibles en su sistema, pero puede que necesite descargar controladores para otros tipos de bases de datos. Los principales proveedores de BD publican software cliente y controladores con frecuencia. Además, puede encontrar otros controladores de otras organizaciones para las tecnologías de acceso de datos mencionadas. En la mayoría de los casos hay varias maneras de conectarse a una base de datos. Las características y el rendimiento de la base de datos dependerán de la tecnología de acceso o de los controladores utilizados

4.2.1.1 Iniciar el asistente para la conexión de base de datos

MapForce cuenta con un asistente que le guiará durante la conexión a un origen de datos. Antes de empezar a seguir las instrucciones del asistente, debe tener en cuenta que algunos tipos de BD requieren instalar y configurar algunos elementos, como controladores o software cliente. Normalmente este tipo de productos se obtienen del proveedor de BD correspondiente, que los acompañan de la documentación pertinente específica para su versión de Windows. Para ver una lista de controladores de base de datos agrupados por tipo de BD consulte el apartado Resumen de controladores de base de datos 165.

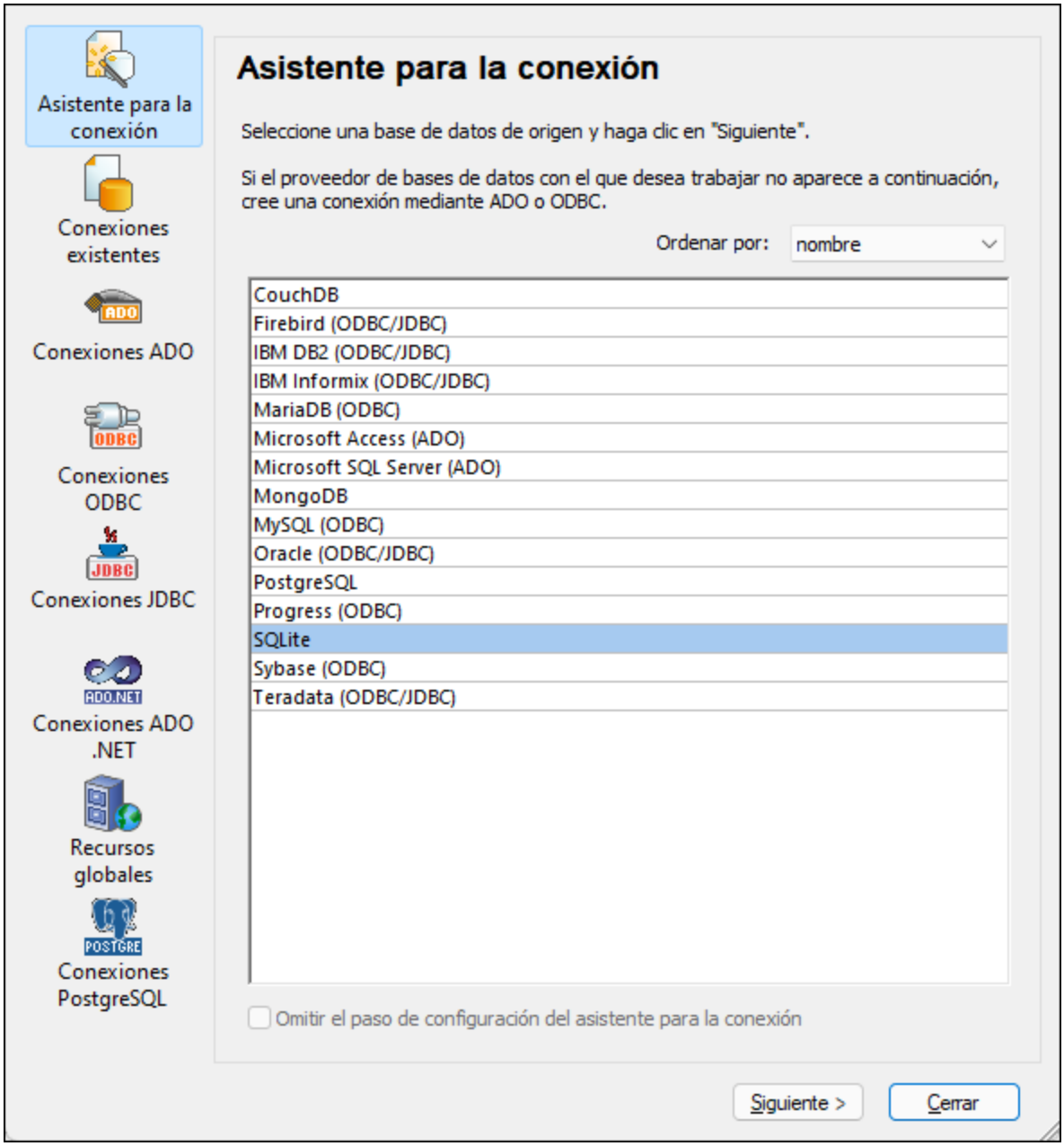
Para iniciar el asistente para la conexión de BD (*imagen siguiente*) siga estos pasos:

- En el menú **Insertar** haga clic en el comando **Base de datos**.

Con estos pasos ha iniciado correctamente el asistente para la conexión a bases de datos (*imagen siguiente*). En la parte izquierda de la ventana puede seleccionar la forma de conectarse a la BD que prefiera:

- Con el asistente para la conexión, que le pide que elija un tipo de base de datos y después le indica qué pasos seguir para conectarse a ese tipo de BD
- Con una conexión que ya existe
- Con uno de estos tipos de tecnología de acceso de datos: ADO, ADO.NET, ODBC o JDBC
- Con un recurso global de Altova en el que se haya guardado una conexión de BD
- Con una conexión PostgreSQL nativa

En el panel del asistente para la conexión (*imagen siguiente*) las bases de datos se pueden ordenar alfabéticamente por tipo de BD o por uso reciente. Seleccione la opción que prefiera en el cuadro combinado *Ordenar*. Una vez haya seleccionado el tipo de BD haga clic en **Siguiente**.



Siga las instrucciones que aparecen en pantalla, que dependerán del tipo de BD, de la tecnología de acceso (ADO, ADO.NET, ODBC, JDBC) y del controlador utilizados. Para ver ejemplos de conexión de cada tipo de BD consulte el apartado Ejemplos de conexión a bases de datos 191.

También puede consultar estos apartados:

- Conexiones ADO 168
- Conexiones ADO.NET 173
- Conexiones ODBC 180
- Conexiones JDBC 183

4.2.1.2 Resumen de controladores de base de datos

En la tabla que aparece más abajo puede ver una lista de controladores de BD que puede utilizar para conectarse a una base de datos a través de las diferentes tecnologías de acceso de datos. Esta lista no contiene todos los controladores disponibles en la actualidad y, por tanto, puede usar otros controladores que no aparecen en la lista.

Aunque puede que Windows venga con algunos controladores de BD, es posible que necesite descargar más controladores. Por lo general, se recomienda utilizar el controlador más reciente publicado por el proveedor de la BD.

Los proveedores de bases de datos pueden ofrecer controladores para descargar en paquetes o junto con el software cliente de base de datos. En este último caso, el cliente de base de datos suele incluir todos los controladores necesarios u ofrecer una opción durante la instalación para seleccionar los controladores y componentes que el usuario desee instalar. El software cliente de base de datos suele incluir funciones de administración y configuración que permiten simplificar la administración de la BD y documentación que explica cómo instalar y configurar el cliente y sus componentes.

Es muy importante configurar bien el cliente de BD para establecer correctamente la conexión con la BD. Antes de instalar y usar el software cliente de BD, recomendamos que lea detenidamente las instrucciones de instalación y configuración porque pueden variar según la versión de la BD y según la versión de Windows.

Si desea conocer las características y limitaciones de cada tecnología de acceso de datos con respecto a cada tipo de BD, consulte la documentación de la BD correspondiente y pruebe la conexión. Debe tener en cuenta estos puntos para evitar problemas de conexión:

- Algunos proveedores de ADO.NET no son compatibles o tienen una compatibilidad limitada (*véase* *Notas sobre compatibilidad con ADO.NET* 180).
- Cuando instale un controlador de BD, recomendamos que tenga la misma plataforma que la aplicación de Altova (32 o 64 bits). Por ejemplo, si usa una aplicación de Altova de 32 bits en un sistema operativo de 64 bits, instale el controlador de 32 bits y configure la conexión de base de datos con ayuda del controlador de 32 bits (*véase* *Ver los controladores ODBC disponibles* 182).
- Cuando configure un origen de datos ODBC, recomendamos crear el nombre de origen de datos (DSN) como DSN del sistema en lugar de como DSN de usuario (*véase* *Conexiones ODBC* 180).
- Si la base de datos de destino es MySQL con conexión por ODBC, deberá habilitar la opción *Return matched rows instead of affected rows* (Devolver filas coincidentes en lugar de filas afectadas) de la pestaña **Cursor/Results** del conector MySQL ODBC. Si lo prefiere e introduce la cadena de conexión a mano en el asistente de conexión para bases de datos, añada `Option=2` a la cadena de conexión (p. ej. `Dsn=mydsn;Option=2;`)
- Cuando configure un origen de datos JDBC, compruebe que tiene instalado JRE (Java Runtime Environment) o Java Development Kit (JDK) y que la variable de entorno CLASSPATH del sistema operativo está configurado (*véase* *Conexiones JDBC* 183).
- Si necesita consultar las instrucciones de instalación e información sobre compatibilidad de los controladores o clientes de BD, consulte la documentación que viene con el paquete de instalación.

Base de datos	**Interfaz**	Controladores
Firebird	ADO.NET	Proveedor de datos ADO.NET (https://www.firebirdsql.org/en/additional-downloads/)

Base de datos	Interfaz	Controladores
	JDBC	Controlador Firebird JDBC (https://www.firebirdsql.org/en/jdbc-driver/)
	ODBC	Controlador Firebird ODBC (https://www.firebirdsql.org/en/odbc-driver/)
IBM DB2	ADO	Proveedor IBM OLE DB para DB2
	ADO.NET	Proveedor IBM Data Server para .NET
	JDBC	Controlador IBM Data Server para JDBC y SQLJ
	ODBC	Controlador IBM DB2 ODBC
IBM DB2 for i	ADO	• Proveedor IBM DB2 para i5/OS IBMDA400 OLE DB • Proveedor IBM DB2 para i5/OS IBMDARLA OLE DB • Proveedor IBM DB2 para i5/OS IBMDASQL OLE DB
	ADO.NET	Proveedor de datos .NET Framework Data Provider para IBM i
	JDBC	Controlador IBM Toolbox para Java JDBC
	ODBC	Controlador iSeries Access ODBC
IBM Informix	ADO	Controlador IBM Informix OLE DB
	JDBC	Controlador IBM Informix JDBC
	ODBC	Controlador IBM Informix ODBC
Microsoft Access	ADO	• Proveedor de Microsoft Jet OLE DB • Proveedor de Microsoft Access Database Engine OLE DB
	ADO.NET	Proveedor de datos de .NET Framework para OLE DB
	ODBC	• Controlador de Microsoft Access
MariaDB	ADO.NET	Si falta el conector .NET especial para MariaDB, use **Connector/NET** pararMySQL (https://dev.mysql.com/downloads/connector/net/).
	JDBC	MariaDB Connector/JDBC (https://downloads.mariadb.org/)
	ODBC	MariaDB Connector/ODBC (https://downloads.mariadb.org/)
Microsoft SQL Server	ADO	• Controlador Microsoft OLE DB para SQL Server (MSOLEDBSQL) • Proveedor de Microsoft OLE DB para SQL Server (SQLOLEDB) • Cliente nativo de SQL Server (SQLNCLI)
	ADO.NET	• Proveedor de datos .NET Framework para SQL Server • Proveedor de datos .NET Framework para OLE DB
	JDBC	• Controlador JDBC de Microsoft para SQL Server (https://msdn.microsoft.com/library/mt484311.aspx)
	ODBC	• Cliente nativo de SQL Server

 © 2016-2022 Altova GmbH

Base de datos	Interfaz	Controladores
MySQL	ADO.NET	Conector/NET (https://dev.mysql.com/downloads/connector/net/)
	JDBC	Conector/JDBC (https://dev.mysql.com/downloads/connector/j/)
	ODBC	Conector/ODBC (https://dev.mysql.com/downloads/connector/odbc/)
Oracle	ADO	• Proveedor de datos Oracle para OLE DB • Proveedor de Microsoft OLE DB para Oracle
	ADO.NET	Proveedor de datos Oracle para .NET (https://www.oracle.com/technetwork/topics/dotnet/index-085163.html)
	JDBC	• Controlador JDBC Thin • Controlador JDBC Oracle Call Interface (OCI) Estos controladores se suelen instalar durante la instalación del cliente de base de datos Oracle. Conéctese con el controlador OCI (no con Thin) si usa el componente Oracle XML DB.
	ODBC	• Microsoft ODBC para Oracle • Controlador Oracle ODBC (por lo general se instala durante la instalación del cliente de base de datos Oracle)
PostgreSQL	JDBC	Controlador PostgreSQL JDBC (https://jdbc.postgresql.org/download.html)
	ODBC	psqlODBC (https://odbc.postgresql.org/)
	Conexión nativa	Disponible. No necesita instalar ningún controladores si usa una conexión nativa.
Progress OpenEdge	JDBC	Conector JDBC (https://www.progress.com/jdbc/openedge)
	ODBC	Conector ODBC (https://www.progress.com/odbc/openedge)
SQLite	Conexión nativa	Disponible. No necesita instalar ningún controladores si usa una conexión nativa.
Sybase	ADO	Proveedor de Sybase ASE OLE DB
	JDBC	jConnect™ para JDBC
	ODBC	Controlador Sybase ASE ODBC
Teradata	ADO.NET	Proveedor de datos .NET para Teradata (https://downloads.teradata.com/download/connectivity/net-data-provider-for-teradata)
	JDBC	Controlador JDBC de Teradata (https://downloads.teradata.com/download/connectivity/jdbc-driver)
	ODBC	Controlador ODBC de Teradata para Windows (https://downloads.teradata.com/download/connectivity/odbc-driver/windows)

4.2.1.3 Conexiones ADO

Microsoft ActiveX Data Objects (ADO) es una tecnología de acceso de datos que permite conectarse a gran variedad de orígenes de datos con OLE DB. OLE DB es una interfaz alternativa a ODBC y JDBC. Ofrece acceso uniforme a los datos en un entorno COM (Component Object Model). ADO es el precursor del nuevo ADO.NET[173] y suele utilizarse para conectarse a bases de datos Microsoft nativas como Microsoft Access o SQL Server, aunque también puede usar ADO para otros orígenes de datos.

Es importante saber que puede escoger entre varios proveedores ADO y que en el caso de algunos es necesario que los descargue e instale en su equipo antes de poder usarlos. Por ejemplo, para conectarse a SQL Server puede usar estos proveedores ADO:

- *Controlador* de Microsoft OLE DB para SQL Server (MSOLEDBSQL)
- *Proveedor* de Microsoft OLE DB para SQL Server (SQLOLEDB)
- Cliente nativo de SQL Server (SQLNCLI)

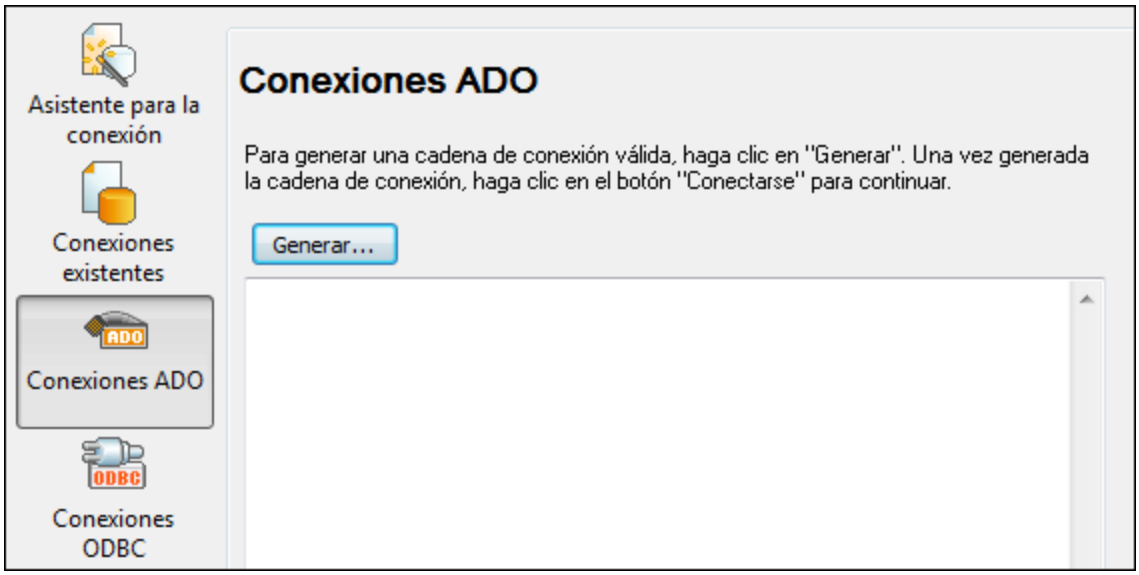
De entre estos tres proveedores recomendamos que use MSOLEDBSQL, que puede descargar desde https://docs.microsoft.com/en-us/sql/connect/oledb/download-oledb-driver-for-sql-server?view=sql-server-ver15. Tenga en cuenta que la versión que descargue debe ser coincidir con la de la plataforma de MapForce (32 bits o 64 bits). Los proveedores SQLOLEDB y SQLNCLI se consideran obsoletos, por lo que no se recomiendan.

Es un problema conocido que el proveedor de BD para SQL Server **Microsoft OLE DB Provider for SQL Server (SQLOLEDB)** tiene dificultades para enlazar parámetros de consultas complejas como las expresiones comunes de tabla (CTE) e instrucciones SELECT anidadas.

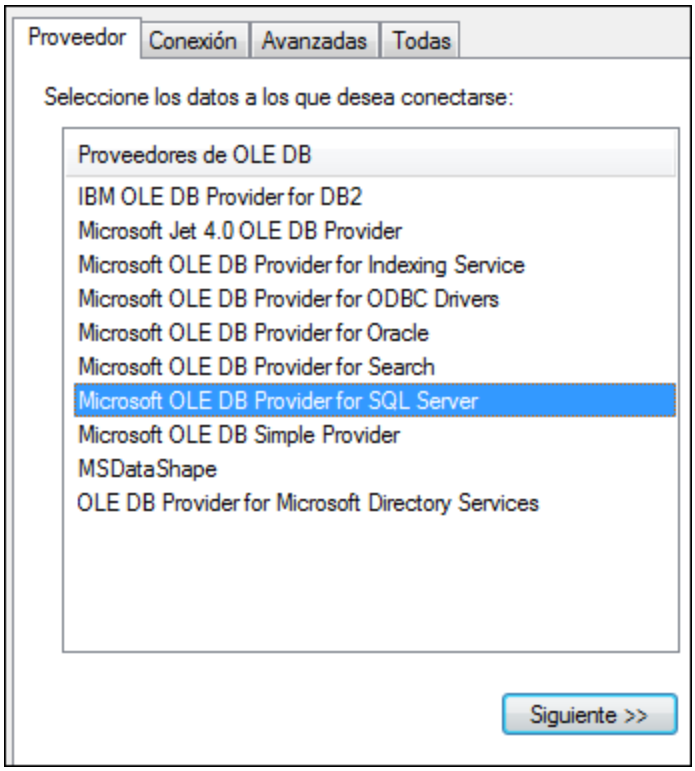
Para configurar una conexión ADO:

1. Inicie el asistente para la conexión de base de datos[163].
2. Haga clic en **Conexiones ADO**.

 © 2016-2022 Altova GmbH



3. Haga clic en **Generar**.



4. Seleccione el proveedor de datos que desea utilizar para establecer la conexión. La tabla que aparece a continuación enumera las combinaciones más comunes.

Para conectarse a esta BD...	Utilice este proveedor...
Microsoft Access	• **Proveedor de Microsoft Office Access Database Engine OLE DB** (recomendado) • **Proveedor de Microsoft Jet OLE DB** Si el proveedor **Microsoft Office Access Database Engine OLE DB Provider** no está disponible en la lista, asegúrese de que tiene instalados o bien Microsoft Access o el componente redistribuible del motor de base de datos de Microsoft Access (https://www.microsoft.com/en-us/download/details.aspx?id=54920) en su equipo.
SQL Server	• **Cliente nativo SQL Server** • **Proveedor para SQL Server de Microsoft OLE DB**
Otras bases de datos	Seleccione el proveedor correspondiente. Si no hay un proveedor OLE DB para su BD, instale el controlador necesario que ofrece el proveedor de la BD (consulte el Resumen de controladores de base de datos 165). Otra opción es configurar una conexión ADO.NET, ODBC o JDBC. Si el sistema operativo tiene un controlador ODBC para la BD, también puede usar **Microsoft OLE DB Provider for ODBC Drivers** o usar una conexión ODBC 180.

5. Haga clic en **Siguiente**.

Las instrucciones de las siguientes pantallas dependen del proveedor elegido. Para SQL Server deberá dar o seleccionar el nombre de host del servidor de BD, el método de autenticación y el nombre de usuario y la contraseña de la base de datos. Para ver un ejemplo, consulte Connecting to Microsoft SQL Server (ADO) 214. En el caso de Microsoft Access deberá buscar o introducir la ruta de acceso al archivo de BD. Para ver un ejemplo consulte Connecting to Microsoft Access (ADO) 212.

La lista de propiedades de inicialización (parámetros de conexión) aparece en la pestaña *Todas* del cuadro de diálogo de conexión. Estas propiedades dependen del proveedor elegido. Consulte estos apartados para aprender a configurar las propiedades básicas de inicialización para bases de datos Microsoft Access y SQL Server:

- Configurar las propiedades de vínculo de datos de SQL Server 171
- Configurar las propiedades de vínculo de datos de Microsoft Access 172

4.2.1.3.1 Conectarse a una base de datos Microsoft Access

Siga estas instrucciones si desea conectarse a una base de datos Microsoft Access que no requiere contraseña. Si la base de datos está protegida con contraseña, entonces consulte el apartado Conectarse a Microsoft Access (ADO) 212.

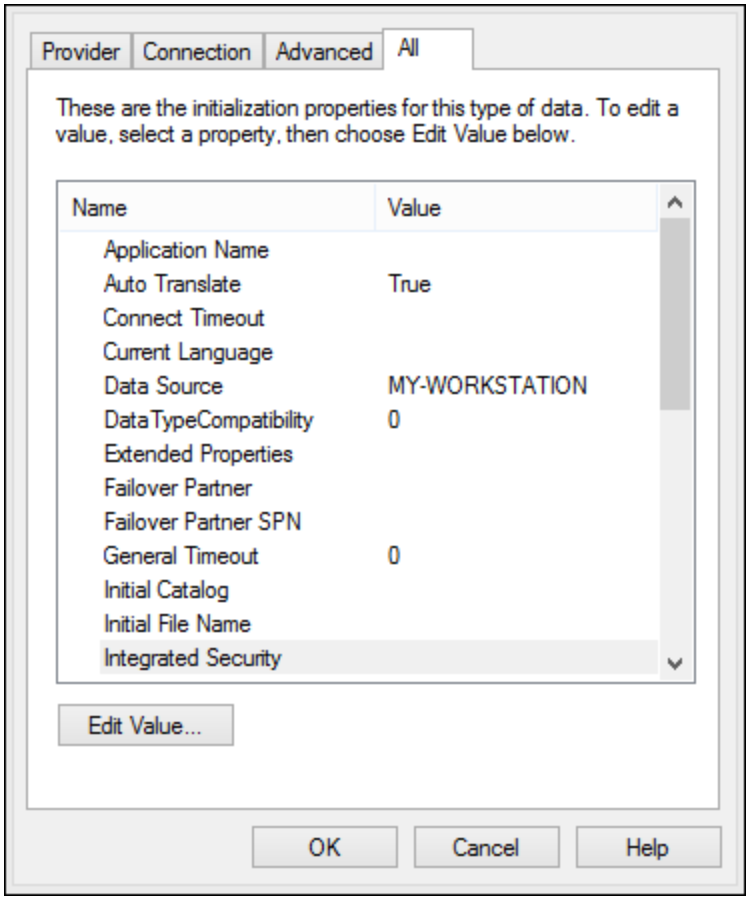
 © 2016-2022 Altova GmbH

Para conectarse a una BD Microsoft Access:

1. Inicie el asistente para la conexión de base de datos 163.
2. Seleccione el botón de opción *Microsoft Access (ADO)* y haga clic en **Siguiente**.
3. Busque el archivo de BD o introduzca su ruta de acceso (relativa o absoluta).
4. Haga clic en **Conectarse**.

4.2.1.3.2 Configurar las propiedades de vínculo de datos de SQL Server

Cuando se conecte a una base de datos Microsoft SQL Server por ADO (consulte Conexiones ADO 168), asegúrese de configurar correctamente estas propiedades de vínculo de datos en la pestaña *Todas* del cuadro de diálogo "Propiedades de vínculo de datos".



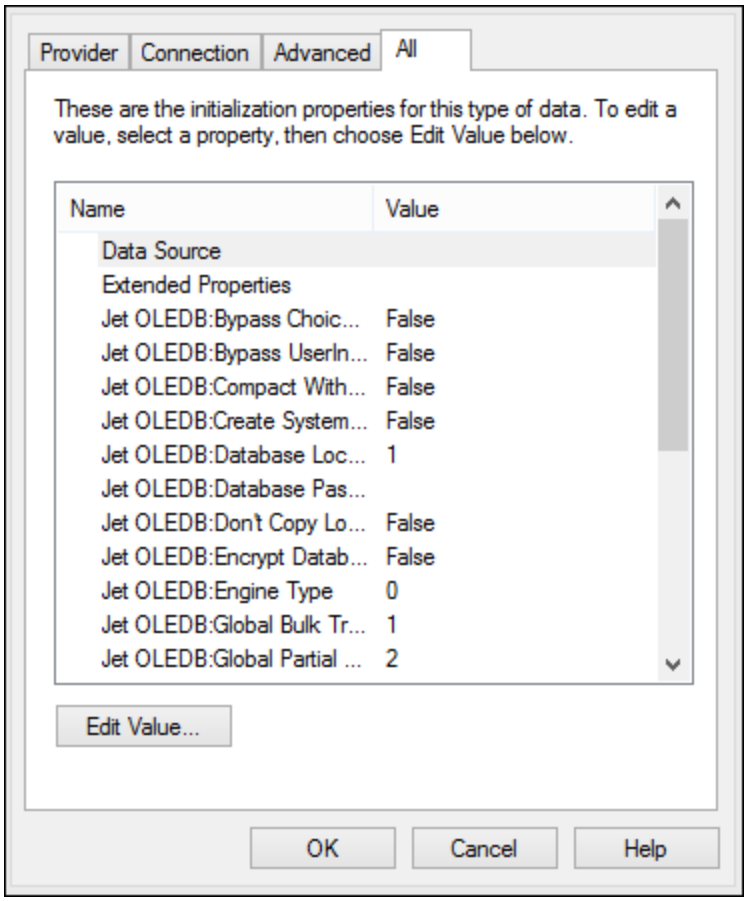
Data Link Properties dialog box

Propiedad	Notas
Seguridad integrada	Si seleccionó el proveedor de datos **SQL Server Native Client** en la pestaña *Proveedor*, esta propiedad debe ser un carácter de espacio en

Propiedad	Notas
	blanco.
Almacenar información de seguridad	Esta propiedad debe tener el valor **True**.

4.2.1.3.3 Configurar las propiedades de vínculo de datos de Microsoft Access

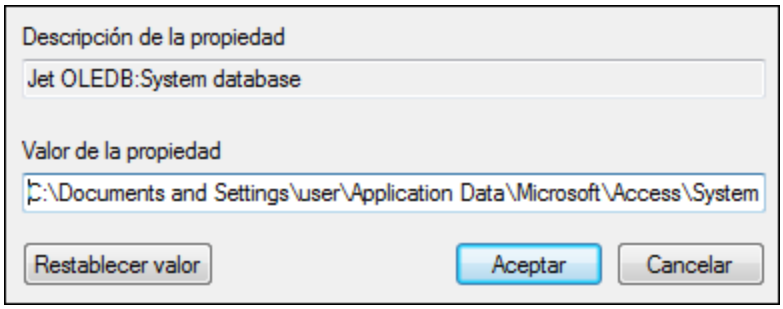
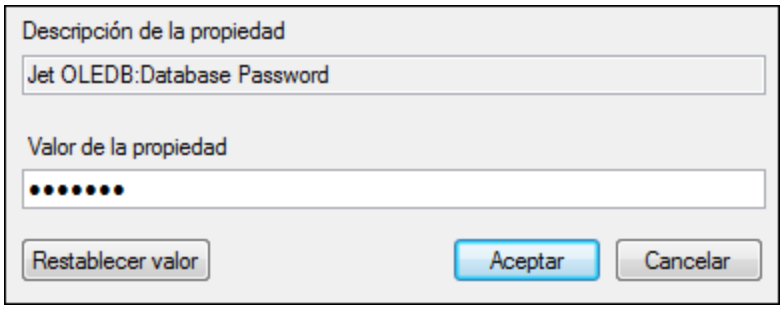
Cuando se conecte a una base de datos Microsoft Access por ADO (consulte Conexiones ADO 168), puede que deba configurar estas propiedades de vínculo de datos en la pestaña *Todas* del cuadro de diálogo "Propiedades de vínculo de datos".



Data Link Properties dialog box

Propiedad	Notas
Origen de datos	Esta propiedad almacena la ruta de acceso del archivo de BD Microsoft Access. Para evitar problemas de conexión, recomendamos usar el formato de ruta de acceso UNC. Por ejemplo:

 © 2016-2022 Altova GmbH

Propiedad	Notas
	`\\servidor\compartir$\rutaArchivo`
Base de datos OLEDB:System Database	Esta propiedad almacena la ruta de acceso del archivo de información de grupo de trabajo. Quizás sea necesario configurar el valor de esta propiedad para poder establecer la conexión con la BD Microsoft Access. Si se produce un error relacionado con el archivo de información de grupo de trabajo, busque el archivo de información de grupo de trabajo (**System.MDW**) que corresponda a su perfil de usuario e introduzca la ruta de acceso del archivo **System.MDW** como valor de esta propiedad. 
Contraseña Jet OLEDB:Database	Si la base de datos está protegida con contraseña, el valor de esta propiedad debe ser la contraseña de la base de datos. 

4.2.1.4 Conexiones ADO.NET

ADO.NET es un conjunto de bibliotecas de Microsoft .NET Framework diseñado para interactuar con datos, incluidos datos de bases de datos. Para conectarse a una base de datos desde MapForce por ADO.NET es necesario tener instalado Microsoft .NET Framework 4 o superior. Como puede ver más abajo, la conexión a la base de datos a través de ADO.NET se hace seleccionando un proveedor .NET y aportando una cadena de conexión.

Un proveedor de datos .NET es una colección de clases que permite conectarse a un tipo concreto de origen de datos (p. ej. un servidor SQL Server o una base de datos Oracle), ejecutar comandos en él y recuperar sus

datos. En otras palabras, con ADO .NET las aplicaciones como MapForce interactúan con una base de datos a través de un proveedor de datos. Cada proveedor de datos está optimizado para poder trabajar con el tipo concreto de origen de datos para el que está diseñado. Hay dos tipos de proveedores .NET:

1. El proveedor que viene por defecto con Microsoft .NET Framework.
2. Proveedores que ofrecen los principales proveedores de bases de datos como extensión para .NET Framework. Este tipo de proveedores ADO.NET deben instalarse por separado y por lo general se pueden descargar del sitio web del correspondiente proveedor de base de datos.

Nota: ciertos proveedores ADO.NET son incompatibles o tienen una compatibilidad limitada. Consulte el apartado Notas sobre compatibilidad con ADO.NET 180.

Para configurar una conexión ADO.NET:

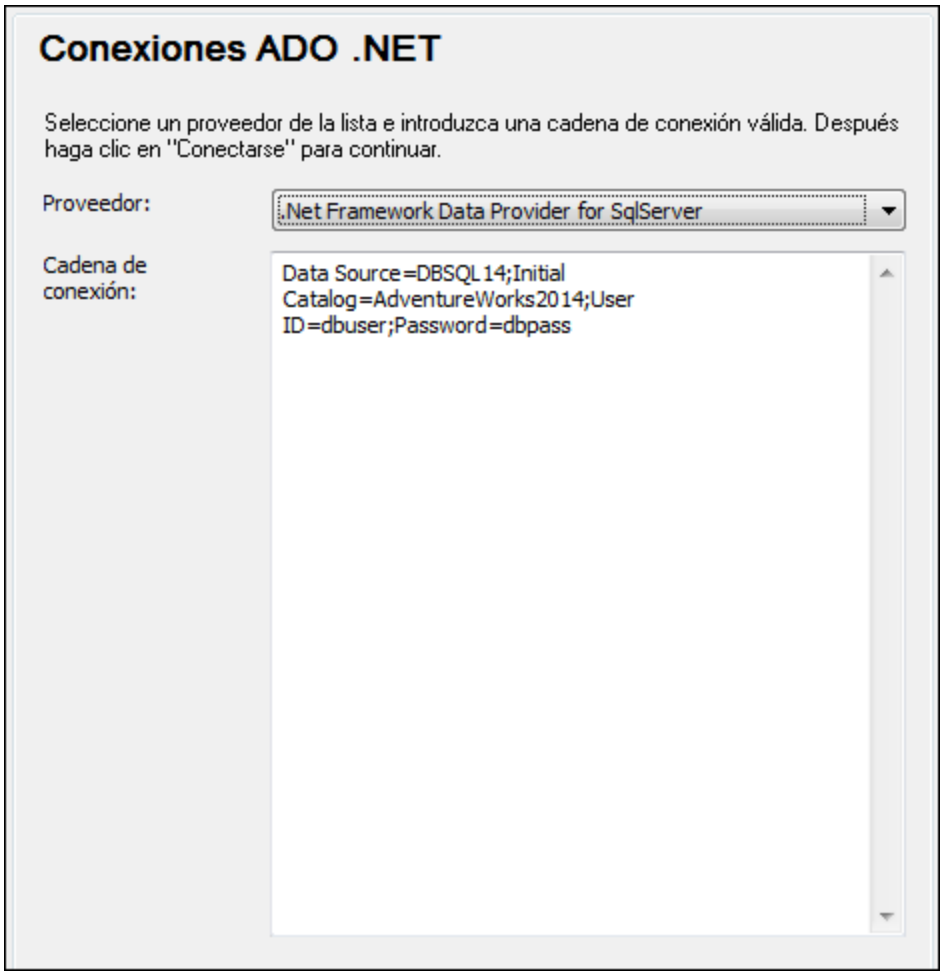
1. Inicie el asistente para la conexión de base de datos 163.
2. Haga clic en **Conexiones ADO.NET**.
3. Seleccione un proveedor de datos .NET de la lista.

> En la lista *Proveedores* aparecen todos los proveedores que vienen por defecto con .NET Framework. Los proveedores de datos .NET propios del proveedor de la base de datos solo aparecen en la lista si ya están instalados en el sistema. Deben instalarse en el caché global de ensamblados (GAC) con ayuda del archivo .msi o .exe que ofrece el proveedor de la base de datos.

4. Introduzca la cadena de conexión con la base de datos. Una cadena de conexión defina la información de conexión con la base de datos y está formada por pares clave/valor de parámetros de conexión delimitados por caracteres de punto y coma. Por ejemplo, la cadena de conexión **`Data Source=DBSQLSERV;Initial Catalog=ProductsDB;User ID=dbuser;Password=dbpass`** permite conectarse a la base de datos SQL Server `ProductsDB` del servidor `DBSQLSERV`, con el nombre de usuario `dbuser` y la contraseña `dbpass`. Puede crear la cadena de conexión tecleando los pares clave/valor directamente en el cuadro de texto *Cadena de conexión*, pero también puede crearla con Visual Studio (*véase* *Crear una cadena de conexión en Visual Studio* 175).

> La sintaxis de la cadena de conexión depende del proveedor que se seleccione en la lista *Proveedores*. Para ver un ejemplo consulte el apartado Ejemplo: cadenas de conexión ADO.NET 178.

 © 2016-2022 Altova GmbH



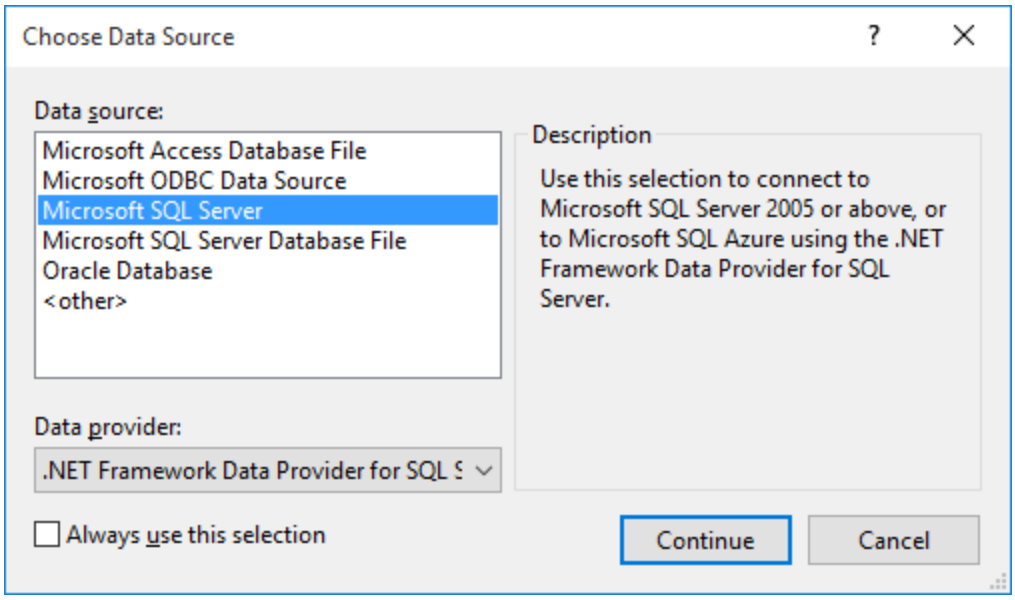
5. Haga clic en **Conectarse** para terminar.

4.2.1.4.1 Crear una cadena de conexión en Visual Studio

Para conectarse a un origen de datos usando ADO.NET se necesita una cadena de conexión válida. A continuación explicamos cómo crear una cadena de conexión desde Visual Studio.

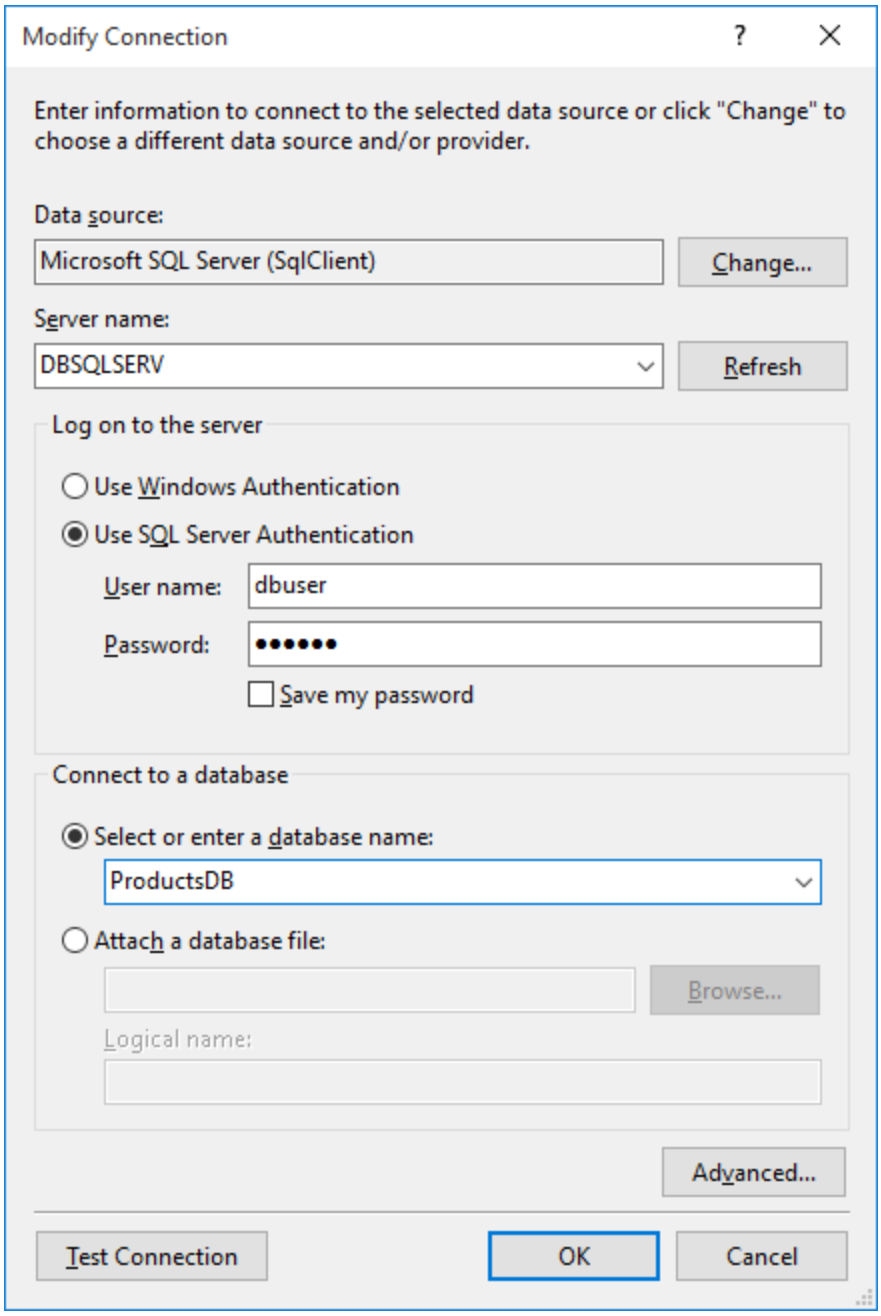
Para crear una cadena de conexión en Visual Studio:

1. En el menú **Herramientas** haga clic en **Conectar a base de datos**.
2. Seleccione un origen de datos de la lista (en este ejemplo Microsoft SQL Server). El proveedor de datos se rellena automáticamente en función de la opción elegida.



3. Haga clic en **Continuar**.

© 2016-2022 Altova GmbH



4. Introduzca el nombre de host del servidor, el nombre de usuario y la contraseña de la base de datos. En este ejemplo nos conectamos a la base de datos `ProductsDB` en el servidor `DBSQLSERV`, usando SQL Server para la autenticación.
5. Para terminar haga clic en **Aceptar**.

Si la conexión se establece correctamente, aparecerá en la ventana Explorador de servidores. Para abrir esta ventana puede usar el comando **Vista | Explorador de servidores**. Para obtener la cadena de conexión con la base de datos, haga clic con el botón derecho en la conexión en la ventana Explorador de servidores y seleccione el comando **Propiedades**. Ahora aparece la cadena de conexión en la ventana Propiedades de

Visual Studio. Recuerde que debe reemplazar los asteriscos con la contraseña antes de pegar la cadena en el cuadro de texto *Cadena de conexión* de MapForce.

4.2.1.4.2 Ejemplo: cadenas de conexión ADO.NET

Para configurar una conexión ADO.NET deberá seleccionar un proveedor ADO.NET en el cuadro de diálogo de conexión a la base de datos y deberá introducir una cadena de conexión (*véase Conexiones ADO.NET* 173). A continuación ofrecemos ejemplos de cadenas de conexión ADO.NET para varias bases de datos.

Proveedor de datos .NET para Teradata

Este proveedor se puede descargar del sitio web de Teradata (https://downloads.teradata.com/download/connectivity/net-data-provider-for-teradata). Un ejemplo de cadena de conexión sería:

```
Data Source=DirecciónServidor;User Id=usuario;Password=contraseña;
```

Proveedor de datos .NET Framework Data Provider para IBM i

Este proveedor se instala con *IBM i Access Client Solutions - Windows Application Package*. Un ejemplo de cadena de conexión sería:

```
DataSource=DirecciónServidor;UserID=usuario;Password=contraseña;DataCompression=True;
```

Para más información consulte el archivo de ayuda llamado ".NET Provider Technical Reference" que viene con el paquete de instalación.

Proveedor de datos .NET Framework Data Provider para MySQL

Este proveedor se puede descargar del sitio web de MySQL (https://dev.mysql.com/downloads/connector/net/). Un ejemplo de cadena de conexión sería:

```
Server=127.0.0.1;Uid=root;Pwd=12345;Database=test;
```

Véase también https://dev.mysql.com/doc/connector-net/en/connector-net-programming-connecting-connection-string.html

 © 2016-2022 Altova GmbH

Proveedor de datos .NET Framework Data Provider para SQL Server

Un ejemplo de cadena de conexión sería:

```
Data Source=DBSQLSERV;Initial Catalog=ProductsDB;User ID=dbuser;Password=dbpass
```

Véase también https://msdn.microsoft.com/en-us/library/ms254500(v=vs.110).aspx

Proveedor de datos IBM DB2 Data Provider 10.1.2 para .NET Framework 4.0

```
Database=PRODUCTS;UID=usuario;Password=contraseña;Server=localhost:50000;
```

Nota: por lo general este proveedor se instala con el paquete de IBM DB2 Data Server Client. Si después de instalar el paquete de IBM DB2 Data Server Client el proveedor no aparece en la lista de proveedores ADO.NET, consulte esta nota técnica: https://www-01.ibm.com/support/docview.wss?uid=swg21429586.

Véase también https://www.ibm.com/support/knowledgecenter/en/SSEPGG_10.1.0/com.ibm.swg.im.dbclient.adonet.ref.doc/doc/DB2ConnectionClassConnectionStringProperty.html

Proveedor de datos Oracle para .NET (ODP.NET)

El paquete de instalación que incluye el proveedor ODP.NET se puede descargar del sitio web de Oracle (http://www.oracle.com/technetwork/topics/dotnet/downloads/index.html). Un ejemplo de cadena de conexión sería:

```
Data Source=DSORCL;User Id=usuario;Password=contraseña;
```

En esta cadena de conexión, `DSORCL` es el nombre del origen de datos que apunta a un nombre de servicio Oracle que está definido en el archivo **tnsnames.ora** (tal y como se describe en el apartado Conectarse a Oracle (ODBC) 227).

Si prefiere establecer la conexión sin configurar un nombre de servicio en el archivo **tnsnames.ora**, entonces use una cadena de conexión parecida a esta:

```
Data Source=(DESCRIPTION=(ADDRESS_LIST=(ADDRESS=(PROTOCOL=TCP)(HOST=host)(PORT=puerto)))
(CONNECT_DATA=(SERVER=DEDICATED)(SERVICE_NAME=MyOracleSID)));User
Id=usuario;Password=contraseña;
```

Véase también https://docs.oracle.com/cd/B28359_01/win.111/b28375/featConnecting.htm

4.2.1.4.3 Notas sobre compatibilidad con ADO.NET

En esta tabla aparecen controladores de base de datos ADO.NET conocidos que por ahora no son compatibles con MapForce o que tienen una compatibilidad limitada.

Base de datos	Controlador	Notas
Todas las bases de datos	**Proveedor de datos .Net Framework para ODBC**	Compatibilidad limitada. Existen problemas conocidos con las conexiones Microsoft Access. Es mejor utilizar conexiones directas ODBC.
	Proveedor de datos .Net Framework para OleDb	Compatibilidad limitada. Existen problemas conocidos con las conexiones Microsoft Access. Es mejor utilizar conexiones directas ADO.
Firebird	**Proveedor de datos Firebird ADO.NET**	Compatibilidad limitada. Es mejor utilizar ODBC o JDBC.
Informix	**Proveedor de datos IBM Informix para .NET Framework 4.0**	Incompatible. Utilice el proveedor **DB2 Data Server Provider**.
IBM DB2 for i (iSeries)	**Proveedor de datos .Net Framework para i5/OS**	Incompatible. utilice el **proveedor de datos .Net Framework para IBM i**, que se instala con el paquete *IBM i Access Client Solutions - Windows Application Package*.
Oracle	**Proveedor de datos .Net Framework para Oracle**	Compatibilidad limitada. Aunque este controlador viene con .NET Framework, Microsoft no recomienda su uso porque es obsoleto.
PostgreSQL	**-**	Para este proveedor no hay controladores ADO.NET compatibles. Es mejor usar una conexión nativa.
Sybase	**-**	Para este proveedor no hay controladores ADO.NET compatibles.

4.2.1.5 Conexiones ODBC

ODBC (Open Database Connectivity) es una tecnología de acceso de datos muy popular con la que se puede conectar a bases de datos desde MapForce. Puede utilizarse como método principal de conexión con bases de datos o como alternativa a las conexiones OLE DB o JDBC.

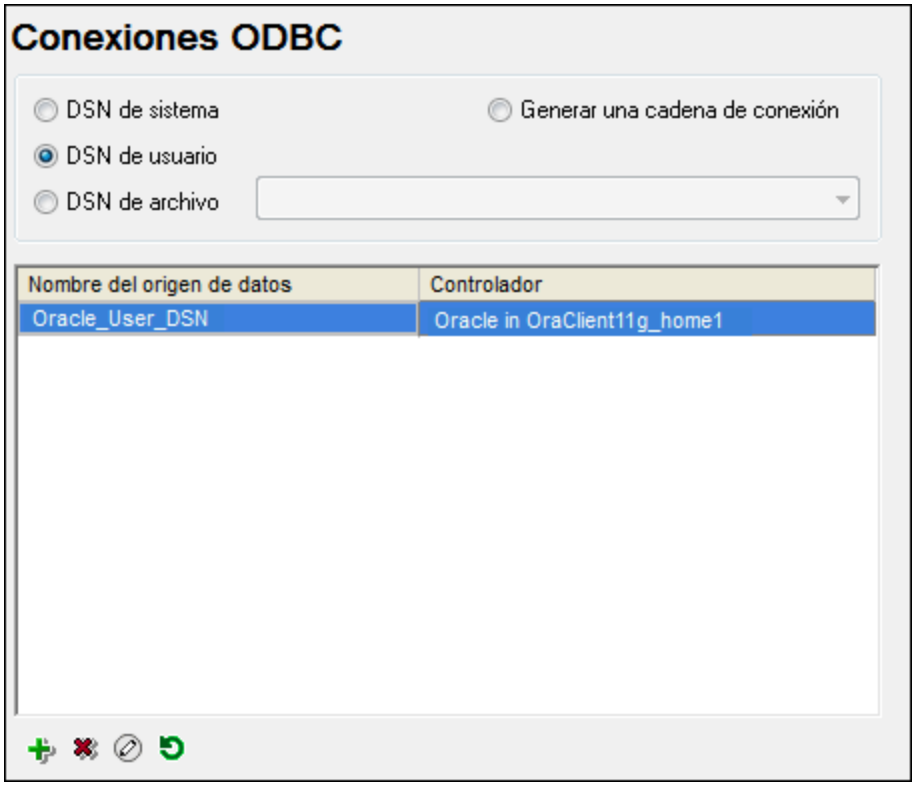
Para conectarse a una base de datos por ODBC primero es necesario contar con un DSN de ODBC en el sistema operativo. El DSN describe de manera uniforme la conexión de BD a todas las aplicaciones cliente compatibles con ODBC que estén en el sistema, incluido MapForce. Los DSN pueden ser de varios tipos:

 © 2016-2022 Altova GmbH

- DSN de sistema
- DSN de usuario
- DSN de archivo

A un origen de datos de sistema pueden acceder todos los usuarios que tengan privilegios en el sistema operativo. A un origen de datos de usuario solo puede acceder el usuario que lo creó. Y, por último, si crea un DSN de archivo, el origen de datos se creará como archivo con extensión `.dsn` que podrá compartir con otros usuarios (siempre que tengan instalados los controladores que utiliza el origen de datos).

Los DSN que estén en el equipo aparecen en el cuadro de diálogo de conexión de base de datos al hacer clic en **Conexiones ODBC**.



Cuadro de diálogo Conexiones ODBC

Si el DSN de la base de datos no existe, el asistente para la conexión de base de datos de MapForce le ayudará a crearlo. Si lo prefiere puede crearlo en Windows directamente. En ambos casos, antes de continuar, compruebe que el controlador ODBC para la base de datos está en la lista de controladores disponibles del sistema operativo (consulte el apartado Ver los controladores disponibles 182).

Para conectarse usando un DSN nuevo:

1. Inicie el asistente para la conexión de base de datos 163.
2. Haga clic en el botón **Conexiones ODBC**.
3. Seleccione un tipo de origen de datos (DSN de usuario, de sistema o de archivo).

> **Nota:** para crear un DSN de sistema necesitará derechos de administrador en el sistema y UModel debe ejecutarse como administrador.

4. Haga clic en **Agregar** .
5. Seleccione un controlador y haga clic en **DSN de usuario** o **DSN de sistema**. Si el controlador correspondiente a la BD no aparece en la lista, descárguelo e instálelo.
6. En el cuadro de diálogo que aparece debe rellenar la información de conexión para terminar de configurar la conexión.

Para que la conexión se establezca correctamente deberá dar el nombre de host (o dirección IP) del servidor de la BD, así como el nombre de usuario y la contraseña. Quizás sean necesarios otros parámetros de conexión. Para más información consulte la documentación que ofrece el proveedor de la base de datos. Una vez creado, el DSN estará en la lista de nombres de orígenes de datos. Así podrá volver a usar los datos de la conexión cada vez que desee conectarse a la BD.

Para conectarse por medio de un DSN:

1. Inicie el asistente para la conexión de base de datos 163.
2. Haga clic en el botón **Conexiones ODBC**.
3. Elija el tipo de origen de datos (DSN de usuario, de sistema o de archivo).
4. Haga clic en el DSN y después en **Conectarse**.

Para generar una cadena de conexión basada en un archivo .dsn:

1. Inicie el asistente para la conexión de base de datos 163.
2. Haga clic en el botón **Conexiones ODBC**.
3. Seleccione *Generar una cadena de conexión* y después haga clic en **Generar**.
4. Si quiere generar la conexión a partir de un DSN de archivo, haga clic en la pestaña *Origen de datos de archivo.* Si no es así, haga clic en la pestaña *Origen de datos de equipo* (los DSN de sistema y de usuario se denominan orígenes de datos de equipo)
5. Seleccione el archivo `.dsn` correspondiente y haga clic en **Aceptar**.

Para conectarse a través de una cadena de conexión preparada previamente:

1. Inicie el asistente para la conexión de base de datos 163.
2. Haga clic en el botón **Conexiones ODBC**.
3. Seleccione *Generar una cadena de conexión*.
4. Pegue la cadena de conexión en el cuadro de texto y haga clic en **Conectarse**.

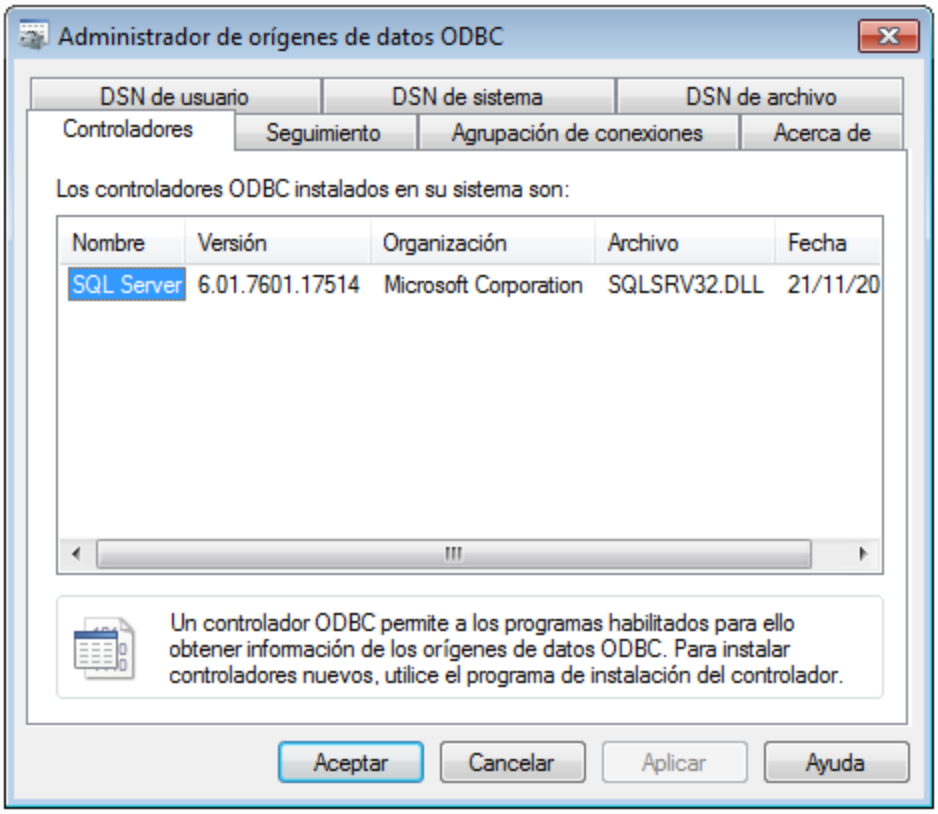
4.2.1.5.1 Controladores ODBC disponibles

En el administrador de orígenes de datos ODBC puede ver qué controladores ODBC están disponibles en su sistema operativo. El administrador (**Odbcad32.exe**) se puede abrir desde el panel de control de Windows (desde **Herramientas administrativas**). En sistemas operativos de 64 bits encontrará dos versiones de este ejecutable:

 © 2016-2022 Altova GmbH

- La versión de 32 bits está en el directorio **C:\Windows\SysWoW64** (siempre y cuando **C:** sea su unidad de sistema).
- La versión de 64 bits está en el directorio **C:\Windows\System32**.

Los controladores de BD de 43 bits aparecerán en la versión de 32 bits del administrador de orígenes de datos ODBC, mientras que los controladores de 64 bits aparecerán en la versión de 64 bits. Esto debe tenerse en cuenta a la hora de consultar los controladores en el administrador.



Si el controlador de la base de datos de destino no está en la lista del administrador o si desea agregar algún controlador, deberá descargar el controlador (Resumen de controladores de base de datos 165). Cuando el controlador esté disponible en el sistema podrá crear conexiones ODBC con él.

4.2.1.6 Conexiones JDBC

JDBC (Java Database Connectivity) es una interfaz de acceso a base de datos que forma parte de la plataforma de software Java de Oracle. Las conexiones JDBC suelen consumir más recursos que las conexiones ODBC pero pueden ofrecer más características.

Requisitos

- Tener instalado Java Runtime Environment (JRE) o Java Development Kit (JDK). Este último puede ser el JDK de Oracle o uno de código abierto, como Oracle OpenJDK. MapForce identifica la ruta de

acceso al equipo virtual Java (JVM) a partir de estas ubicaciones, en este orden: a) la ruta personal de acceso al JVM que puede indicar en en las **Opciones** (véase el apartado Java[1052]); b) la ruta de acceso al JVM que se encuentra en el registro; c) la variable de entorno `JAVA_HOME`.

- Asegúrese de que la plataforma de MapForce (32 o 64 bits) es la misma que la de JRE/JDK.
- Tener instalados los controladores JDBC del proveedor de la BD. Estos pueden ser los controladores JDBC que forman parte de la instalación de un cliente de BD o bibliotecas JDBC (archivos .jar) que haya descargado por separado, siempre que estos estén disponible y la BD sea compatible (*véase también Ejemplos de conexión a bases de datos*[191]).
- La variable del entorno `CLASSPATH` debe incluir la ruta de acceso del controlador JDBC (que puede ser un archivo .jar o varios). Algunos clientes de base de datos configuran esta variable automáticamente durante la instalación. Para más información consulte el apartado Configurar la variable CLASSPATH[186].

Conectarse a un servidor SQL a través de JDBC con credenciales de Windows

Si se conecta a un servidor SQL mediante JDBC con credenciales de Windows (seguridad integrada), tenga en cuenta lo siguiente:

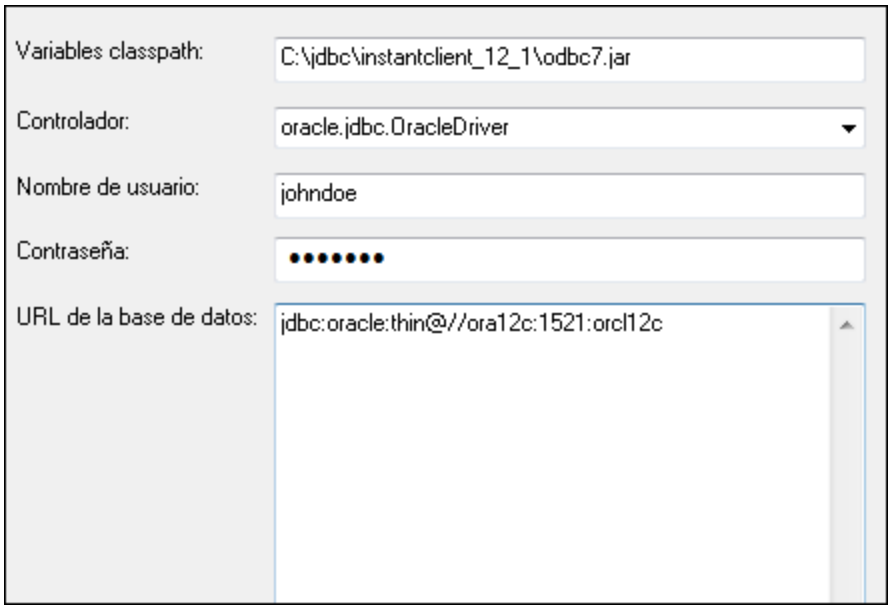
- Debe copiar el archivo **sqljdbc_auth.dll** que viene con el paquete del controlador JDBC a un directorio que esté en la variable de entorno PATH del sistema. Existen dos archivos de este tipo, uno para la plataforma x86 y otro para la plataforma x64. Debe asegurarse de que añade a la ruta PATH el que corresponde con su plataforma de JDK. También debe asegurarse de que reinicia MapForce (o el programa que ejecute la aplicación, si es otro) después de cambiar la variable de entorno.
- La cadena de conexión JDBC debe incluir la propiedad **`integratedSecurity=true`**. Puede añadir esta propiedad desde varios sitios:
  - desde el asistente para la conexión a una base de datos (*véase más abajo*)
  - desde la configuración de componentes de BD[278]
  - si procede, editando la cadena de conexión a la BS en el código Java generado[1083].

Para más información consulte la documentación Controlador JDBC de Microsoft para SQL Server: https://docs.microsoft.com/es-es/sql/connect/jdbc/building-the-connection-url.

Para configurar una conexión JDBC:

1. Inicie el asistente para la conexión de base de datos[163].
2. Haga clic en el botón **Conexiones JDBC**.
3. Si quiere, introduzca una lista de rutas de archivo .jar separadas por caracteres de punto y coma en el cuadro de texto *Variables classpath*. Las bibliotecas .jar que se introduzcan aquí se cargarán en el entorno junto a las que ya estén definidas en la variable de entorno `CLASSPATH`. Cuando termine de editar el cuadro de texto Variables classpath, todos los controladores JDBC encontrados en las bibliotecas .jar de origen se cargan automáticamente a la lista del cuadro combinado *Controlador*.

© 2016-2022 Altova GmbH



4. En el cuadro combinado *Controlador* seleccione un controlador JDBC de la lista o introduzca un nombre de clase Java. Observe que esta lista contiene todos los controladores JDBC configurados a través de la variable de entorno `CLASSPATH` (*véase* *Configurar la variable CLASSPATH* 186), así como los controladores encontrados en el campo *Variables classpath*.

> Las rutas de acceso del controlador JDBC definidas sen la variable CLASSPATH, así como las rutas de acceso de los archivos .jar introducidos directamente en el cuadro de diálogo de conexión a la base de datos se envían a Java Virtual Machine (JVM). JVM decide qué controladores se utilizan para establecer la conexión. Se recomienda realizar un seguimiento de las clases Java que se cargan en JVM para evitar conflictos y resultados inesperados a la hora de conectarse a la base de datos.

5. Introduzca el nombre de usuario y la contraseña de la BD.
6. En el cuadro de texto *URL de la base de datos:* introduzca la URL de la conexión JDBC en el formato propio del tipo de base de datos utilizado. En la siguiente tabla puede ver la sintaxis de las URL de conexión JDBC para los tipos de base de datos más frecuentes.

Base de datos	URL de conexión JDBC
Firebird	`jdbc:firebirdsql://<host>[:<puerto>]/<ruta de acceso o alias de la BD>`
IBM DB2	`jdbc:db2://nombreHost:puerto/nombreBaseDatos`
IBM DB2 for i	`jdbc:as400://[host]`
IBM Informix	`jdbc:informix-sqli://nombreHost:puerto/nombreBaseDatos:INFORMIXSERVER=miservidor`
MariaDB	`jdbc:mariadb://nombreHost:puerto/nombreBaseDatos`

Base de datos	URL de conexión JDBC
Microsoft SQL Server	`jdbc:sqlserver://nombreHost:puerto;nombreBaseDatos=name`
MySQL	`jdbc:mysql://nombreHost:puerto/nombreBaseDatos`
Oracle	`jdbc:oracle:thin:@nombreHost:puerto:SID` `jdbc:oracle:thin:@//nombreHost:puerto:servicio`
Oracle XML DB	`jdbc:oracle:oci:@//nombreHost:puerto:servicio`
PostgreSQL	`jdbc:postgresql://nombreHost:puerto/nombreBaseDatos`
Progress OpenEdge	`jdbc:datadirect:openedge://host:puerto;databaseName=nombre_bd`
Sybase	`jdbc:sybase:Tds:nombreHost:puerto/nombreBaseDatos`
Teradata	`jdbc:teradata://nombreServidorBaseDatos`

Nota: en algunos casos puede modificarse el formato sintáctico (p. ej. la URL de la base de datos puede excluir el puerto o puede incluir el nombre de usuario y la contraseña). Consulte la documentación del proveedor de BD para obtener más información.

7. Haga clic en **Conectarse**.

4.2.1.6.1 Configurar la variable CLASSPATH

La variable de entorno `CLASSPATH` es utilizada por Java Runtime Environment (JRE) o por Java Development Kit (JDK) para encontrar las clases Java y otros archivos de recursos del sistema operativo. Cuando se conecte a una BD con JDBC, esta variable debe incluir la ruta de acceso del controlador JDBC del sistema y, en algunos casos, la ruta de acceso de otros archivos de biblioteca relacionados con el tipo de BD que esté utilizando.

En la tabla que aparece a continuación puede ver ejemplos de variables `CLASSPATH`. Lo más importante que debe tener en cuenta es que quizás deba adaptar esta información dependiendo de la ubicación y del nombre del controlador JDBC y de la versión JRE/JDK que esté en su sistema operativo. Para evitar problemas de conexión recomendamos leer detenidamente las instrucciones de instalación del controlador JDBC que esté instalado en el sistema.

Base de datos	Ejemplo de variable CLASSPATH
Firebird	`C:\Archivos de programa\Firebird\Jaybird-2.2.8-JDK_1.8\jaybird-full-2.2.8.jar`
IBM DB2	`C:\Archivos de programa (x86)\IBM\SQLLIB\java\db2jcc.jar;C:\Archivos de programa (x86)\IBM\SQLLIB\java\db2jcc_license_cu.jar;`
IBM DB2 para i	`C:\jt400\jt400.jar;`

© 2016-2022 Altova GmbH

Base de datos	Ejemplo de variable CLASSPATH
IBM Informix	`C:\Informix_JDBC_Driver\lib\ifxjdbc.jar;`
MariaDB	**`<directorio de instalación>`**`\mariadb-java-client-2.2.0.jar`
Microsoft SQL Server	`C:\Archivos de programa\Microsoft JDBC Driver 4.0 for SQL Server\sqljdbc_4.0\enu\sqljdbc.jar`
MySQL	**`<directorio de instalación>`**`\mysql-connector-java-`*`version`*`-bin.jar;`
Oracle	**`ORACLE_HOME`**`\jdbc\lib\ojdbc6.jar;`
Oracle (with XML DB)	**`ORACLE_HOME`**`\jdbc\lib\ojdbc6.jar;`**`ORACLE_HOME`**`\LIB\xmlparserv2.jar;` **`ORACLE_HOME`**`\RDBMS\jlib\xdb.jar;`
PostgreSQL	**`<directorio de instalación>`**`\postgresql.jar`
Progress OpenEdge	`%DLC%\java\openedge.jar;%DLC%\java\pool.jar;` **Nota:** siempre y cuando Progress OpenEdge SDK esté instalado en el equipo, `%DLC%` es el directorio donde está instalado OpenEdge.
Sybase	`C:\sybase\jConnect-7_0\classes\jconn4.jar`
Teradata	**`<directorio de instalación>`**`\tdgssconfig.jar;`**`<directorio de instalación>`**`\terajdbc4.jar`

- Los cambios en la configuración de la variable CLASSPATH pueden afectar al comportamiento de las aplicaciones Java del equipo. Consulte la documentación de Java antes de continuar.
- Las variables de entorno pueden ser del sistema o del usuario. Para cambiar las variables de entorno del sistema es necesario tener derechos de administrador.
- Tras modificar la variable de entorno, reinicie los programas que estén en ejecución para que los cambios surtan efecto. También puede cerrar sesión o reiniciar el sistema.

Para configurar CLASSPATH en Windows 7:

1. Abra el menú **Inicio** y haga clic con el botón derecho en **Equipo**.
2. Haga clic en **Propiedades**.
3. Haga clic en **Configuración avanzada del sistema**.
4. En la pestaña *Avanzadas*, haga clic en **Variables de entorno**.
5. Busque la variable `CLASSPATH` del sistema o del usuario y haga clic en **Editar**. Si no existe, haga clic en **Nueva** para crearla.

6. Edite el valor de la variable e incluya la ruta de acceso del controlador JDBC. Utilice un punto y coma para separar la ruta de acceso del controlador de las demás partes de la variable.

Para configurar CLASSPATH en Windows 10:

1. Pulse la tecla Windows y teclee "variables de entorno".
2. Haga clic en la sugerencia **Editar las variables de entorno del sistema**.
3. Haga clic en **Variables de entorno**.
4. Busque la variable CLASSPATH del sistema o del usuario y haga clic en **Editar**. Si no existe, haga clic en **Nueva** para crearla.
5. Edite el valor de la variable e incluya la ruta de acceso del controlador JDBC. Utilice un punto y coma para separar la ruta de acceso del controlador de las demás partes de la variable.

4.2.1.7 Conexiones PostgreSQL

Las conexiones a bases de datos PostgreSQL pueden configurarse como conexiones nativas o a través de ODBC, JDBC y otros controladores. La ventaja de configurar una conexión nativa es que no es necesario tener controladores instalados en el sistema.Además, si tiene pensado implementar archivos para ejecutarlos en servidores macOS o Linux, tampoco será necesario instalar controladores en el servidor de destino.

Si prefiere establecer la conexión por medio de un controlador no nativo, consulte estos temas de la documentación:

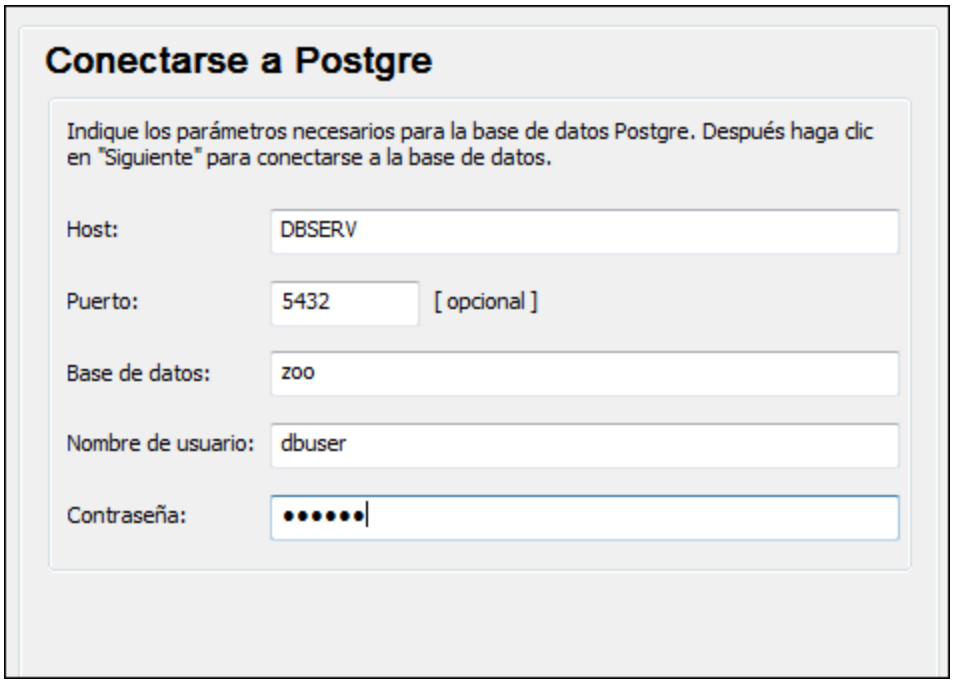
- Configurar una conexión JDBC 183
- Conectarse a PostgreSQL (ODBC) 233

Si lo que quiere es configurar una conexión nativa con PostgreSQL, siga las instrucciones que aparecen más abajo. El único requisito es conocer el nombre de host, el puerto, el nombre de la base de datos, el nombre de usuario y la contraseña.

Para configurar una conexión PostgreSQL nativa:

1. Inicie el asistente para la conexión de base de datos 163.
2. Haga clic en el botón **Conexiones PostgreSQL**.
3. Ahora introduzca el host (localhost si PostgreSQL se ejecuta en el mismo equipo), el puerto (es opcional, pero suele ser 5432), el nombre de la BD, el nombre de usuario y la contraseña.

 © 2016-2022 Altova GmbH



4. Para terminar haga clic en **Conectarse**.

Si el servidor de base de datos PostgreSQL está en otro equipo, debe tener en cuenta lo siguiente:

- El servidor de base de datos PostgreSQL debe estar configurado para aceptar conexiones de clientes. Concretamente, debe configurar el archivo **pg_hba.conf** para que permita conexiones no locales. Además debe configurar el archivo **postgresql.conf** para que escuche determinadas direcciones IP y puertos. Para más información consulte la documentación de PostgreSQL (https://www.postgresql.org/docs/9.5/static/client-authentication-problems.html).
- El equipo servidor debe estar configurado para aceptar conexiones en el puerto correspondiente (suele ser 5432) a través del servidor de seguridad. Por ejemplo, imagine que tiene un servidor de BD que se ejecuta en un equipo Windows. Deberá crear una regla que permita conexiones en el puerto 5432 a través del servidor de seguridad (desde **Panel de control > Firewall de Windows > Configuración avanzada > Reglas de entrada**).

4.2.1.8 Conexiones SQLite

SQLite (https://www.sqlite.org/index.html) es un tipo de base de datos basado en archivos y con almacenamiento. Las bases de datos SQLite son compatibles de forma nativa con MapForce, así que no es necesario instalar ningún controlador para poder conectarse a este tipo de BD.

Notas sobre compatibilidad con bases de datos SQLite

- En Linux no hay un tiempo de espera de ejecución de instrucciones para bases de datos SQLite.
- No son compatibles con búsquedas de texto completo en tablas.
- SQLite admite valores de diferentes tipos de datos en cada fila de una tabla. En MapForce todos los valores procesados deben ser compatibles con el tipo de columna declarado. Por tanto, pueden darse

errores en tiempo de ejecución si la base de datos SQLite tiene valores de fila que no coinciden con el tipo de columna declarado.

- Si la asignación de datos debe escribir datos en una base de datos SQLite y no tiene preparado el archivo de BD de destino, entonces deberá crearlo por separado. Este archivo se puede crear con una herramienta como Altova DatabaseSpy (https://www.altova.com/es/databasespy.html) o puede descargar la shell de comandos de SQLite del sitio web oficial y crear el archivo desde la línea de comandos (*véase Ejemplo: asignar datos XML a SQLite* 330). Para más información consulte la documentación de SQLite.
- Las bases de datos SQLite son compatibles con el lenguaje de transformación integrado de MapForce (tanto para obtener una vista previa de la asignación de datos como para ejecutar archivos de ejecución de MapForce Server).
- Las bases de datos SQLite no se pueden usar en funciones definidas por el usuario.

Importante

Es recomendable que use la palabra clave `STRICT` al crear tablas para asegurarse de que el comportamiento de los datos es más predecible. De lo contrario, es posible que los datos no se lean o escriban correctamente si en una misma columna hay valores de distintos tipos mezclados. Para saber más sobre las tablas `STRICT` consulte la documentation SQLite.

4.2.1.8.1 Conectarse a una base de datos SQLite

Para conectarse a una base de datos SQLite:

1. Inicie el asistente para la conexión de base de datos 163
2. Seleccione el botón de opción *SQLite* y después haga clic en **Siguiente**.
3. Busque el archivo de BD o introduzca su ruta de acceso (relativa o absoluta). El botón **Conectarse** se habilita una vez introducida la ruta de acceso del archivo.
4. Ahora haga clic en **Conectarse**.

4.2.1.9 Usar una conexión a través de recursos globales

Si ya configuró una conexión de base de datos como recurso global, podrá volver a utilizar la conexión siempre que quiera e incluso desde aplicaciones de Altova diferentes.

Crear una base de datos como recurso global

Para crear una BD como recurso global siga estos pasos:

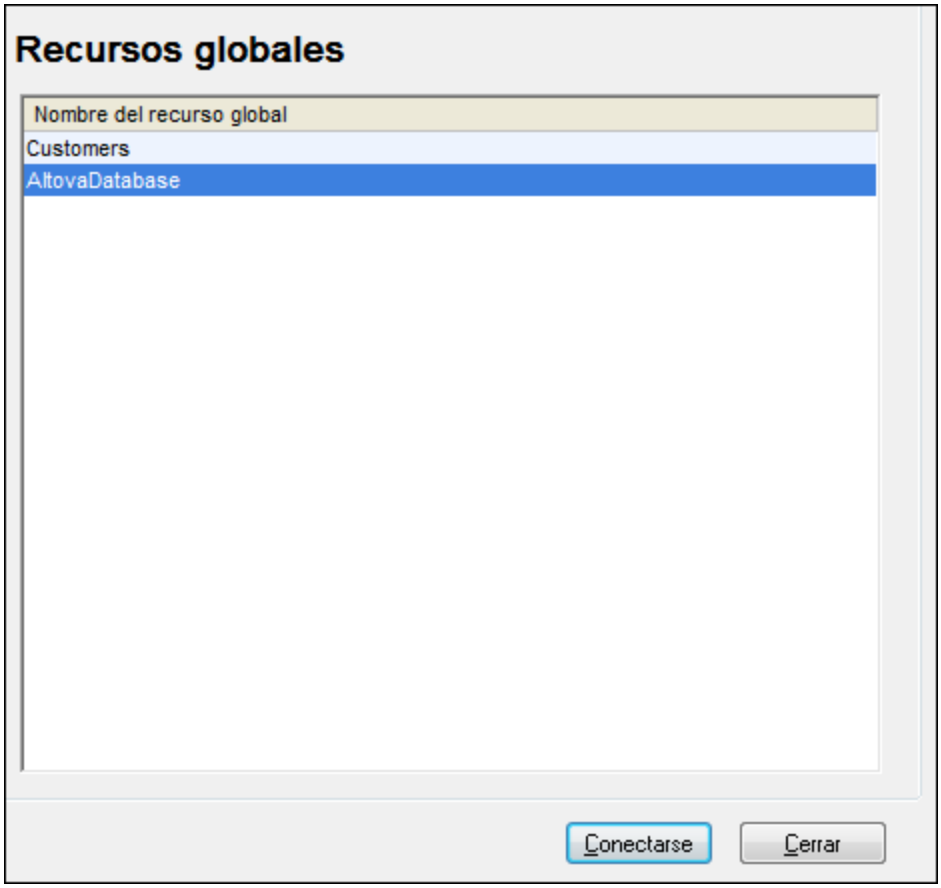
1. En el menú **Herramientas** de MapForce, haga clic en **Recursos globales**.
2. Haga clic en **Agregar** y después en Base de datos.
3. Escriba un nombre para el recurso global en el campo *Alias del recurso*.
4. Haga clic en **Elegir base de datos**. Aparece el asistente para la conexión de base de datos 163.
5. Use el asistente para agregar una conexión de base de datos como hemos explicado.

 © 2016-2022 Altova GmbH

Usar una conexión de base de datos disponible como recurso global

Para usar una conexión de BD disponible como recurso global (*punto anterior*) siga estos pasos:

1. Inicie el asistente para la conexión de base de datos tal y como se explica en el punto anterior.
2. Haga clic en **Recursos globales** en el panel izquierdo. En pantalla aparecen ahora las conexiones de base de datos configuradas previamente como recurso global (*imagen siguiente*).



3. Seleccione la conexión de base de datos correspondiente y haga clic en **Conectarse**. Consejo: para obtener información sobre un recurso global pase el puntero por encima de la entrada.

4.2.1.10 Ejemplos de conexión a bases de datos

Esta sección incluye ejemplos de cómo conectarse a una base de datos desde MapForce con ADO, ODBC o JDBC. Para ver los ejemplos de ADO.NET consulte el apartado Ejemplo: cadenas de conexión ADO.NET 178. Para ver cómo establecer una conexión nativa a PostgreSQL y SQLite (véanse respectivamente Conexiones PostgreSQL 188 y Conexiones SQLite 189).

Tenga en cuenta lo siguiente:

- Puede que las instrucciones sean distintas si su configuración de Windows, entorno de red y cliente de BD o software de servidor no son los mismos que los descritos en el ejemplo correspondiente.

- En la mayoría de los casos se puede establecer la conexión por medio de varias tecnologías de acceso (ADO, ADO.NET, ODBC, JDBC) o controladores. El rendimiento de la conexión de BD, así como sus características y limitaciones, dependerá del controlador, software cliente y parámetros de conexión seleccionados.

4.2.1.10.1 Firebird (JDBC)

Este ejemplo explica cómo conectarse a una base de datos Firebird por JDBC.

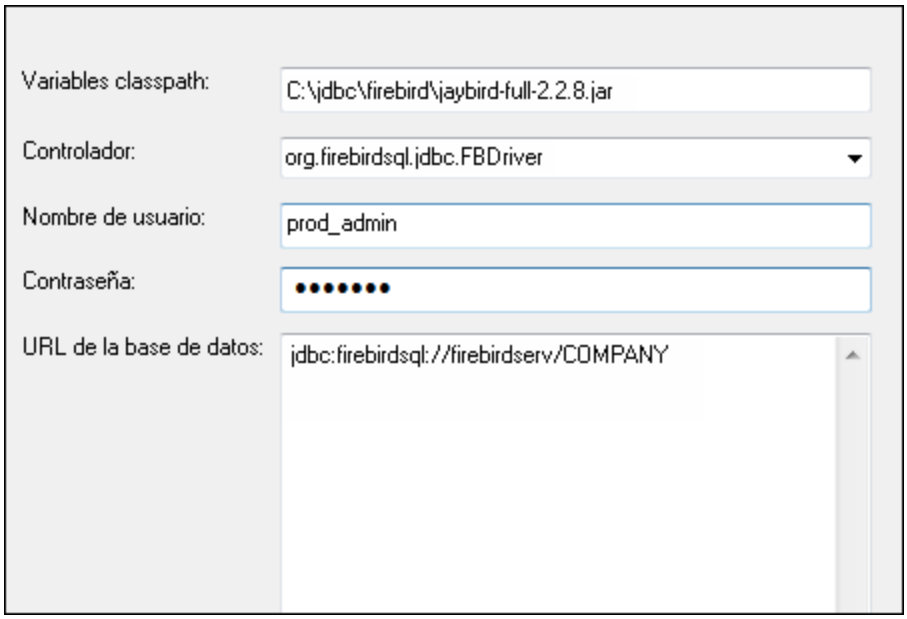
Requisitos:

- JRE (Java Runtime Environment) o Java Development Kit (JDK) está instalado. Este último puede ser el JDK de Oracle o uno de código abierto, como Oracle OpenJDK. MapForce identifica la ruta de acceso al equipo virtual Java (JVM) a partir de estas ubicaciones, en este orden: a) la ruta personal de acceso al JVM que puede indicar en en las **Opciones** (véase el apartado Java 1052); b) la ruta de acceso al JVM que se encuentra en el registro; c) la variable de entorno `JAVA_HOME`.
- Asegúrese de que la plataforma de MapForce (32 o 64 bits) es la misma que la de JRE/JDK.
- El controlador Firebird JDBC debe estar disponible en el sistema operativo (se trata de un archivo .jar que ofrece conectividad con la base de datos). El controlador se puede descargar del sitio web de Firebird (https://www.firebirdsql.org/). En este ejemplo usamos el controlador *Jaybird 2.2.8*.
- Disponer de los datos de conexión: host, ruta de acceso (o alias) de la base de datos, nombre de usuario y contraseña.

Para conectarse a Firebird por JDBC:

1. Inicie el asistente para la conexión de base de datos 163.
2. Haga clic en **Conexiones JDBC**.
3. En el campo *Variables classpath* debe introducir la ruta de acceso del archivo .jar que ofrece conectividad con la base de datos. Si fuera necesario, también puede introducir una lista de rutas de archivo .jar separadas por caracteres de punto y coma. En este ejemplo, el archivo .jar está ubicado en esta ruta de acceso: `C:\jdbc\firebird\jaybird-full-2.2.8.jar`. Tenga en cuenta que este campo puede dejarse en blanco si añadió la ruta de acceso de los archivos .jar a la variable de entorno CLASSPATH del sistema operativo (*véase Configurar la variable CLASSPATH* 186).
4. En el campo *Controlador* seleccione **org.firebirdsql.jdbc.FBDriver**. Recuerde que esta entrada solo estará disponible si se encuentra una ruta de archivo .jar válida en el campo Variables *classpath* o en la variable de entorno CLASSPATH del sistema operativo.

© 2016-2022 Altova GmbH



5. Introduzca el nombre de usuario y la contraseña de la base de datos.
6. Introduzca la cadena de conexión para el servidor de BD en el cuadro de texto *URL de la base de datos* (reemplace lo valores resaltados con los de su base de datos).

```
jdbc:firebirdsql://<host>[:<puerto>]/<ruta o alias de la BD>
```

7. Haga clic en **Conectarse**.

4.2.1.10.2 Firebird (ODBC)

Este ejemplo explica cómo conectarse a una base de datos Firebird 2.5.4 de un servidor Linux.

Requisitos:

- El servidor de BD Firebird está configurado para aceptar conexiones TCP/IP desde clientes.
- El controlador ODBC de Firebird está instalado en el sistema. Este ejemplo usa la versión 2.0.3.154 del controlador (https://www.firebirdsql.org/).
- El cliente Firebird está instalado en el sistema. Recuerde que el cliente forma parte del paquete de instalación del servidor Firebird. Puede descargar este paquete del sitio web de Firebird (https://www.firebirdsql.org/) y buscar el instalador para Windows (Windows executable installer for full Superclassic/Classic or Superserver). Para instalar los archivos del cliente solamente elija la opción **Minimum client install - no server, no tools** del asistente para la instalación.

Nota:

- La plataforma del controlador ODBC de Firebird y del cliente (de 32 o 64 bits) debe coincidir con la plataforma de MapForce.
- La versión del cliente Firebird debe coincidir con la versión del servidor Firebird al que desea conectarse.

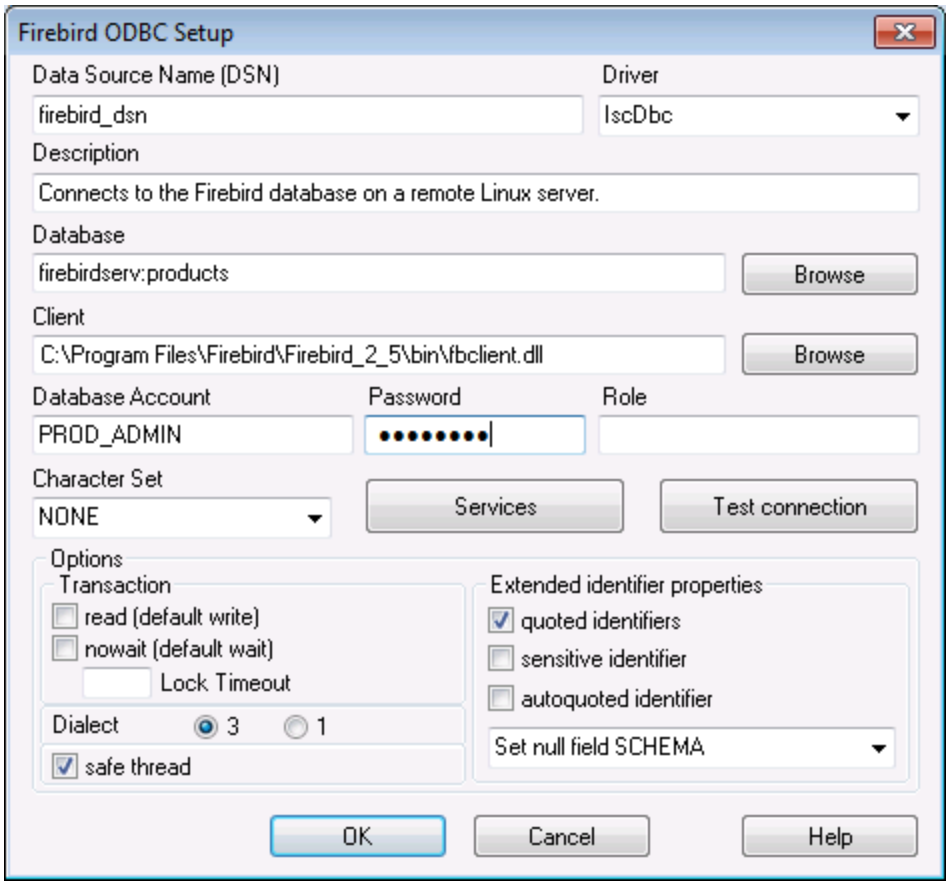
- Disponer de los datos de conexión: nombre de host o dirección IP, ruta de acceso (o alias) de la base de datos en el servidor, nombre de usuario y contraseña.

Para conectarse a Firebird por ODBC:

1. Inicie el asistente para la conexión de base de datos 163.
2. Haga clic en **Conexiones ODBC**.
3. Seleccione la opción *DSN de usuario* (o *DSN de sistema* si tiene privilegios de administrador) y después haga clic en **Agregar** .

4. Seleccione el controlador Firebird y después haga clic en *DSN de usuario* (o *DSN de sistema* dependiendo de la opción seleccionada en el paso anterior). Si el controlador Firebird no aparece en la lista, compruebe que está instalado en el sistema.

 © 2016-2022 Altova GmbH



5. Introduzca los datos de conexión:

Nombre del origen de datos (DSN)	Introduzca un nombre para el origen de datos que desea crear.

Base de datos	Introduzca el nombre de host o dirección IP del servidor, seguida de dos puntos, seguido del alias (o ruta de acceso) de la base de datos. En este ejemplo el nombre de host es `firebirdserv` y el alias de la BD es `products`:
	
	`firebirdserv:products`
	
	Usamos el alias de la BD porque damos por hecho que en el lado servidor el administrador de la BD ha configurado el alias *products* para apuntar al archivo de BD Firebird (.fdb) en el servidor.
	
	En lugar del nombre de host también puede usar la dirección IP del servidor. Y en lugar del alias puede usar una ruta de acceso. Por tanto, también podría usar estas otras dos cadenas de conexión:
	
	`firebirdserver:/var/Firebird/databases/butterflies.fdb`
	`127.0.0.1:D:\Misc\Lenders.fdb`

	Si la base de datos está en el equipo Windows local, haga clic en **Examinar** y seleccione el archivo de base de datos directamente.
Cliente	Introduzca la ruta de acceso del archivo `fbclient.dll`. Su ubicación predeterminada es el subdirectorio `bin` del directorio de instalación de Firebird.
Cuenta de base de datos	Introduzca el nombre de usuario que recibió del administrador de la BD (en este ejemplo es `PROD_ADMIN`).
Contraseña	Introduzca la contraseña de la BD que recibió del administrador.

6. Haga clic en **Aceptar**.

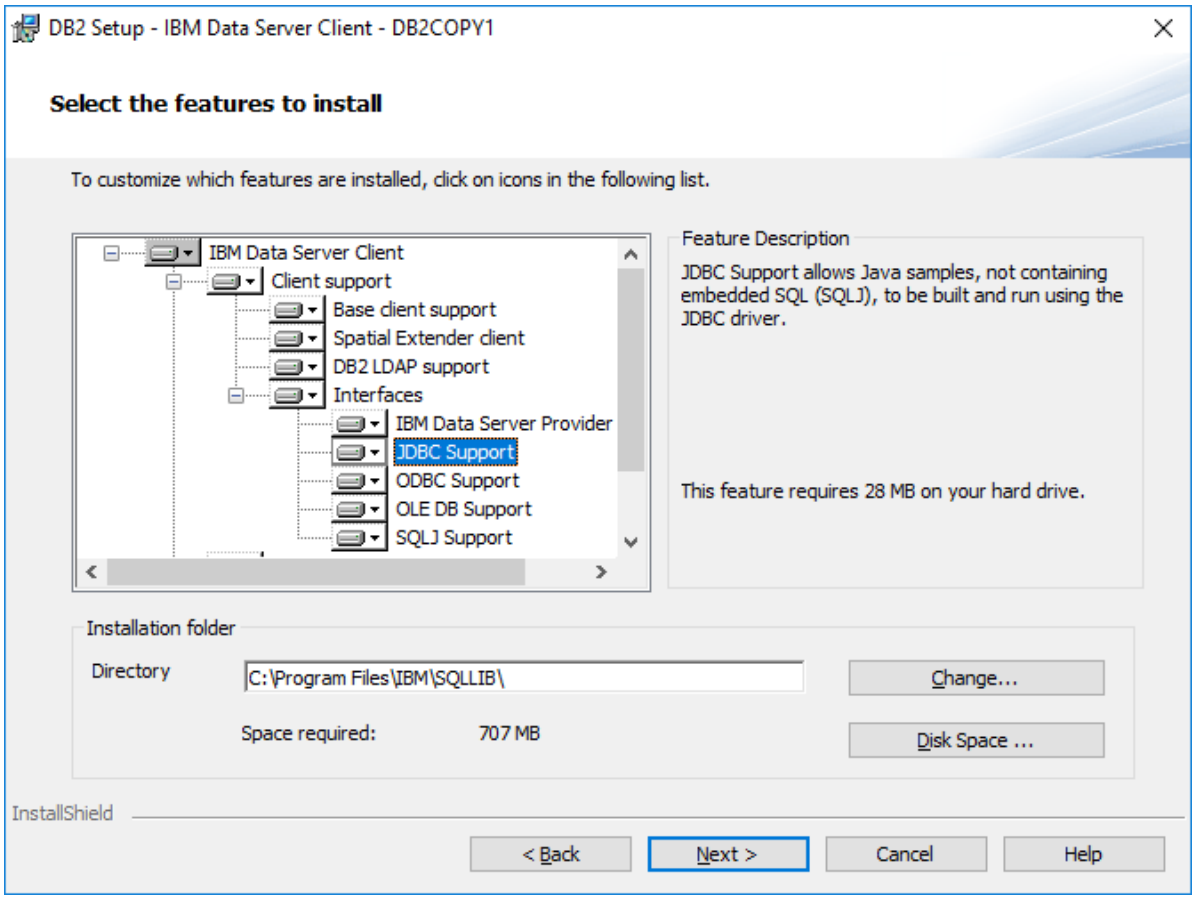
4.2.1.10.3 IBM DB2 (JDBC)

Este ejemplo explica cómo conectarse a una base de datos IBM DB2 por JDBC.

Requisitos:

- JRE (Java Runtime Environment) o Java Development Kit (JDK) está instalado. Este último puede ser el JDK de Oracle o uno de código abierto, como Oracle OpenJDK. MapForce identifica la ruta de acceso al equipo virtual Java (JVM) a partir de estas ubicaciones, en este orden: a) la ruta personal de acceso al JVM que puede indicar en en las **Opciones** (véase el apartado Java[1052]); b) la ruta de acceso al JVM que se encuentra en el registro; c) la variable de entorno `JAVA_HOME`.
- Asegúrese de que la plataforma de MapForce (32 o 64 bits) es del mismo tipo que la de JRE/JDK. Este ejemplo usa Oracle OpenJDK 11.0 de 64 bits, por lo que usa la versión de 64 bits de MapForce.
- El controlador JDBC (uno o más archivos .jar que permiten conectarse a la BD) debe estar disponible en el sistema operativo. Este ejemplo usa el controlador JDBC que está disponible tras instalar la versión 10.1 del **IBM Data Server Client** (64 bits). Al instalar los controladores, elija la instalación típica o seleccione esa opción en el asistente de instalación.

 © 2016-2022 Altova GmbH

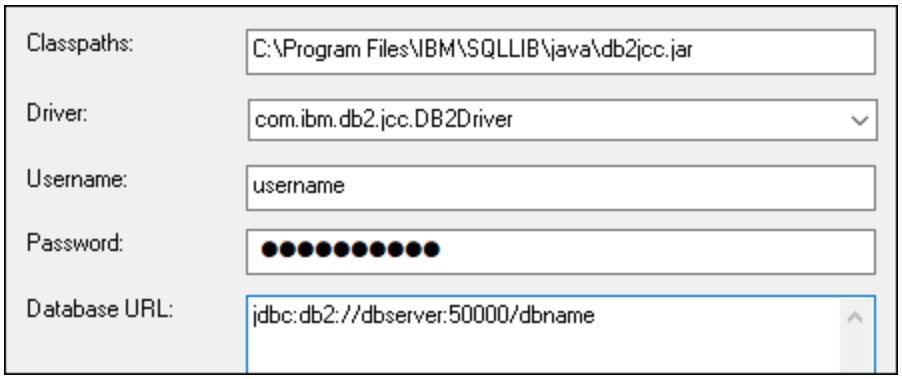


Si no ha modificado la ruta predeterminada de la instalación, una vez esta haya finalizado los archivos .jar requeridos están en el directorio **C:\Program Files\IBM\SQLLIB\java**.

- Necesitará esta información sobre la conexión a la BD: host, puerto, nombre de la BD, nombre de usuario y contraseña.

Para conectarse a IBM DB2 for i por JDBC:

1. Inicie el asistente para la conexión de base de datos 163.
2. Haga clic en **Conexiones JDBC**.
3. Junto a "Variables classpath" introduzca la ruta al archivo .jar que permite conectarse a la BD. En este ejemplo la ruta es **C:\Program Files\IBM\SQLLIB\java\db2jcc.jar**. Puede que necesite hacer referencia al controlador **db2jcc4.jar** en función de la versión del servidor de BD. Si tiene dudas sobre si el controlador es compatible consulte la documentación de IBM (http://www-01.ibm.com/support/docview.wss?uid=swg21363866). Puede dejar el campo "Variables classpath" vacío si ha añadido la ruta de acceso del archivo .jar (también pueden ser varios) a la variable de entorno CLASSPATH del sistema operativo (*véase también el apartado Configurar la variable CLASSPATH* 186).
4. En el campo "Controlador" seleccione **com.ibm.db2.jcc.DB2Driver**. Esta entrada solo está disponible si se encuentra una ruta de acceso válida a un archivo .jar en el campo "Variables classpath" o en la variable de entorno CLASSPATH del sistema operativo (*véase el paso anterior*).



5. Introduzca el nombre de usuario y la contraseña del usuario de la BD en los campos correspondientes.
6. Introduzca la cadena de conexión JDBC en el campo **URL de la BD**. Asegúrese de reemplazar los detalles de la conexión con los de su servidor de BD.

```
jdbc:db2://hostName:port/databaseName
```

7. Haga clic en **Conectarse**.

4.2.1.10.4 IBM DB2 (ODBC)

Este ejemplo explica cómo conectarse a una base de datos IBM DB2 por ODBC.

Requisitos:

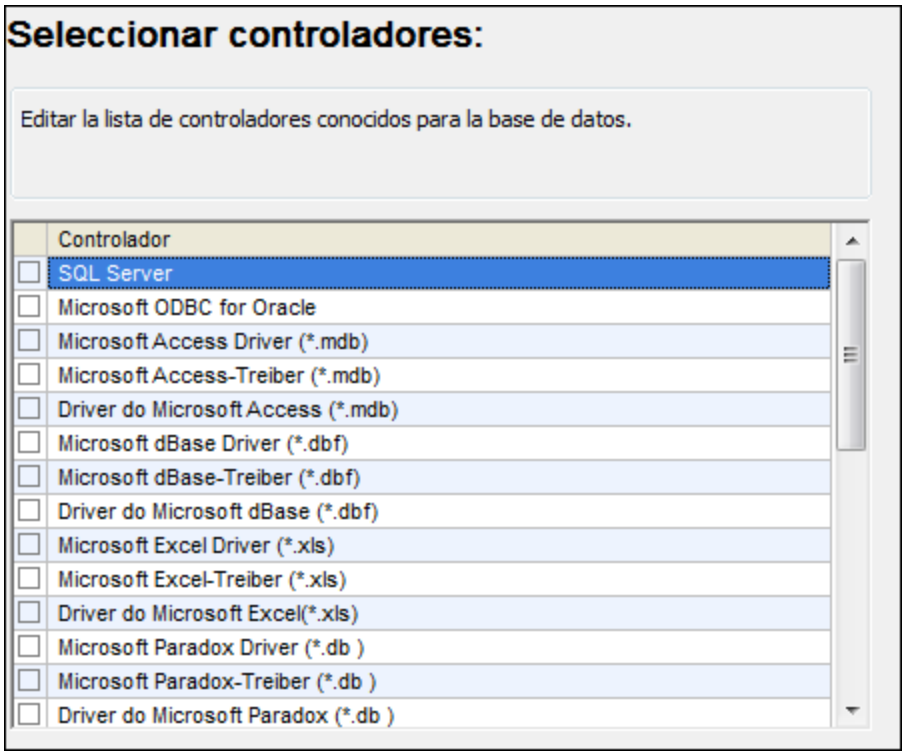
- El cliente IBM Data Server está instalado y configurado en el sistema (en este ejemplo usamos IBM Data Server Client 9.7). Después de instalar el cliente IBM Data Server compruebe que los controladores ODBC están disponibles en el equipo (consulte Ver los controladores ODBC disponibles 182).
- Tiene un alias para la base de datos. El alias se puede crear de varias maneras:
  - Con el asistente de configuración de IBM DB2
  - Con el procesador de línea de comandos de IBM DB2
  - Con el asistente para orígenes de datos ODBC (instrucciones más abajo).
- Disponer de los datos de conexión: host, base de datos, puerto, nombre de usuario y contraseña.

Para conectarse a IBM DB2:

1. Inicie el asistente para la conexión de base de datos 163 y seleccione la opción *IBM DB2 (ODBC/JDBC)*.
2. Haga clic en **Siguiente**.

 © 2016-2022 Altova GmbH

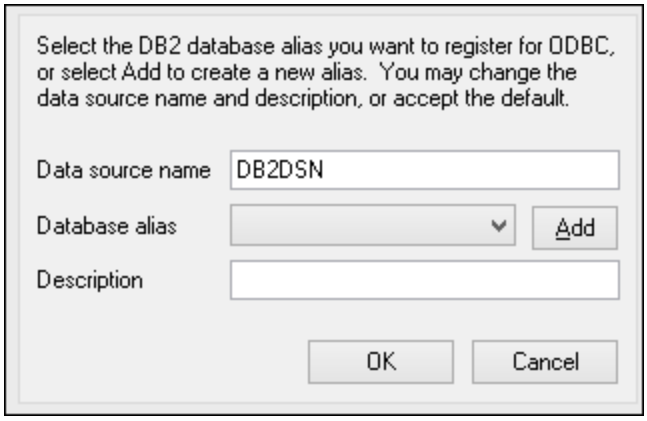
3. Seleccione la opción *ODBC* y haga clic en **Siguiente**. Si necesita editar la lista de controladores conocidos para la base de datos, seleccione los controladores que corresponden a IBM DB2 (ver Requisitos 198) y haga clic en **Siguiente**.



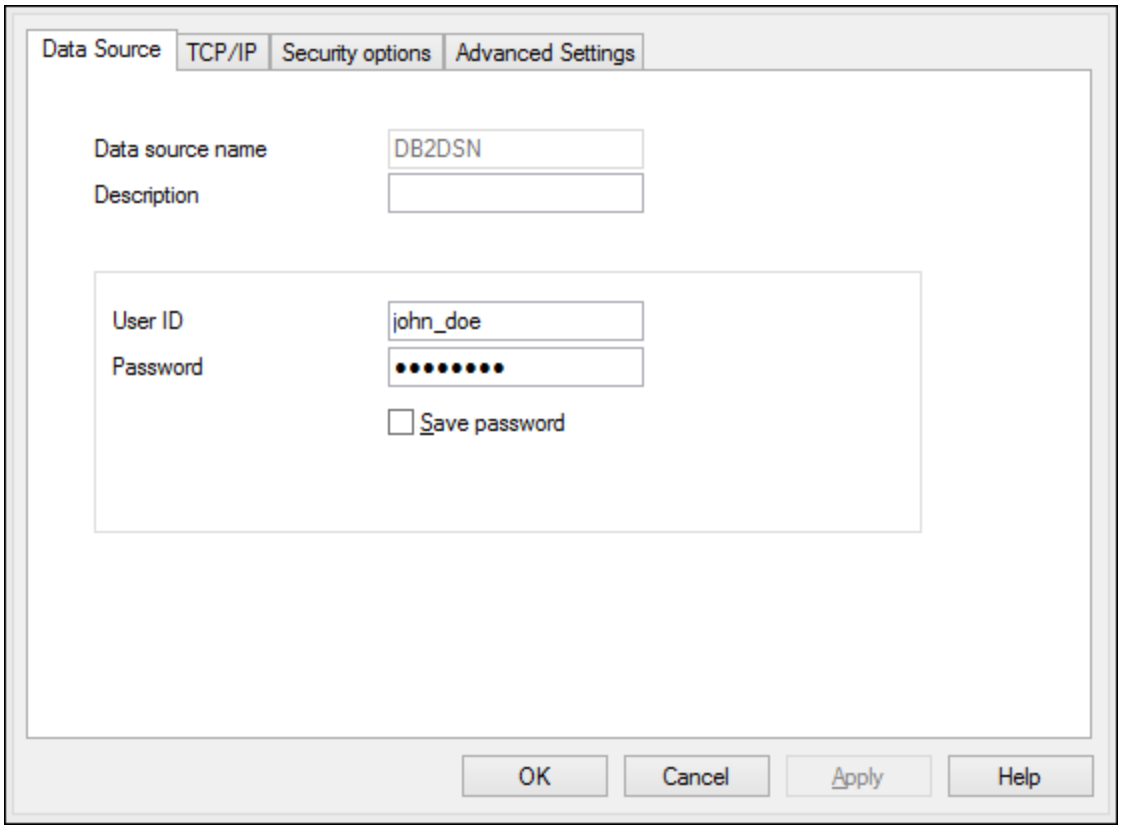
4. Seleccione el controlador de la lista y haga clic en **Conectarse**. (Para editar la lista de controladores disponibles haga clic en **Editar controladores** y active/desactive los controladores que desea agregar o eliminar)

© 2016-2022 Altova GmbH

5. Introduzca el DSN (**DB2DSN**) y haga clic en **Agregar**.

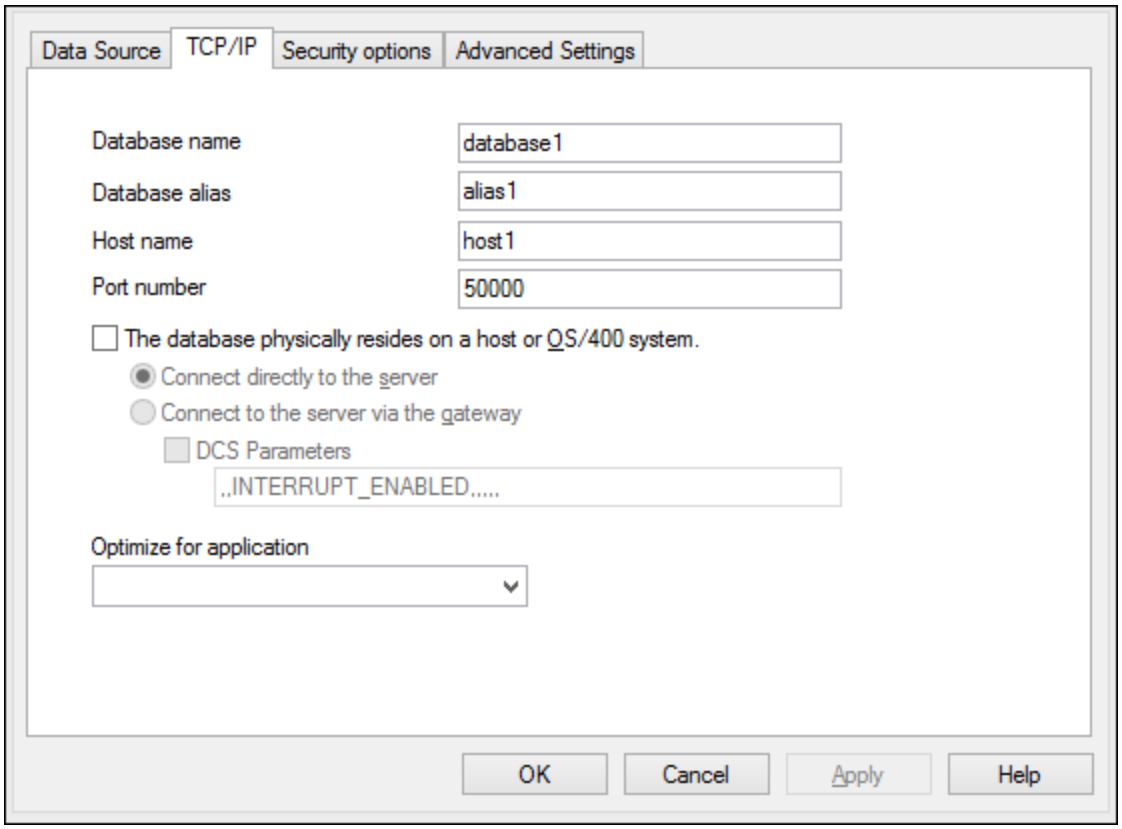


6. En la pestaña *Origen de datos* introduzca el nombre de usuario y la contraseña de la base de datos.

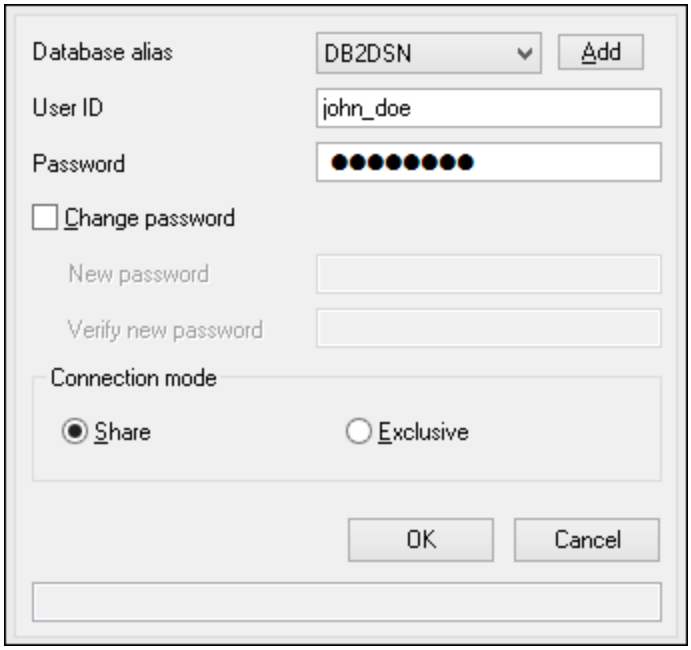


7. En la pestaña **TCP/IP** introduzca el nombre de la base de datos, un nombre para el alias, el nombre de host y el número de puerto. Después haga clic en **Aceptar**.

© 2016-2022 Altova GmbH



8. Vuelva a introducir el nombre de usuario y la contraseña y haga clic en **Aceptar**.



4.2.1.10.5 IBM DB2 para i (JDBC)

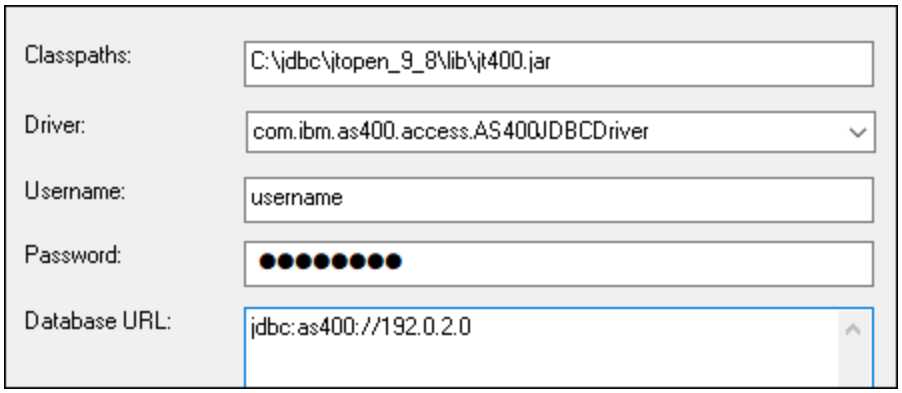
Este ejemplo explica cómo conectarse a una base de datos IBM DB2 for i por JDBC.

Requisitos:

- JRE (Java Runtime Environment) o Java Development Kit (JDK) está instalado. Este último puede ser el JDK de Oracle o uno de código abierto, como Oracle OpenJDK. MapForce identifica la ruta de acceso al equipo virtual Java (JVM) a partir de estas ubicaciones, en este orden: a) la ruta personal de acceso al JVM que puede indicar en en las **Opciones** (véase el apartado Java[1052]); b) la ruta de acceso al JVM que se encuentra en el registro; c) la variable de entorno `JAVA_HOME`.
- Asegúrese de que la plataforma de MapForce (32 o 64 bits) es del mismo tipo que la de JRE/JDK. Este ejemplo usa Oracle OpenJDK 11.0 de 64 bits, por lo que usa la versión de 64 bits de MapForce.
- El controlador JDBC (uno o más archivos .jar que permitan conectarse a la BD) debe estar disponible en su sistema operativo. En este ejemplo se usa la biblioteca de código abierto **Toolbox for Java/JTOpen** versión 9.8 (http://jt400.sourceforge.net/). Una vez haya descargado el paquete y lo haya desempaquetado en un directorio local, los archivos .jar necesarios estarán disponibles en el subdirectorio **lib**.
- Necesitará esta información sobre la conexión a la BD: host, nombre de usuario y contraseña.

Para conectarse a IBM DB2 for i con JDBC:

1. Inicie el asistente para la conexión de base de datos[163].
2. Haga clic en **JDBC Connections**.
3. Junto a "Variables classpath" introduzca la ruta de acceso al archivo .jar que permite conectarse a la BD. En este ejemplo la ruta es **C:\jdbc\jtopen_9_8\jt400.jar**. Tenga en cuenta que puede dejar vacío el campo "Variables classpath" si ha añadido la ruta de acceso al archivo .jar (también pueden ser varios) a la variable de entorno CLASSPATH del sistema operativo (*véase también Configurar la variable CLASSPATH*[186]).
4. En el campo "Controlador" seleccione **com.ibm.as400.access.AS400JDBCDriver**. Esta entrada solo está disponible si se encuentra una ruta de acceso válida a un archivo .jar en el campo "Variables classpath" o en la variable de entorno CLASSPATH del sistema operativo (*véase el paso anterior*).



5. Introduzca el nombre de usuario y la contraseña del usuario de la BD en los campos correspondientes.
6. Introduzca la cadena de conexión JDBC en el campo **URL de la BD**. Asegúrese de reemplazar `host` con el nombre de host o la dirección IP de su servidor de BD.

 © 2016-2022 Altova GmbH

```
jdbc:as400://host
```

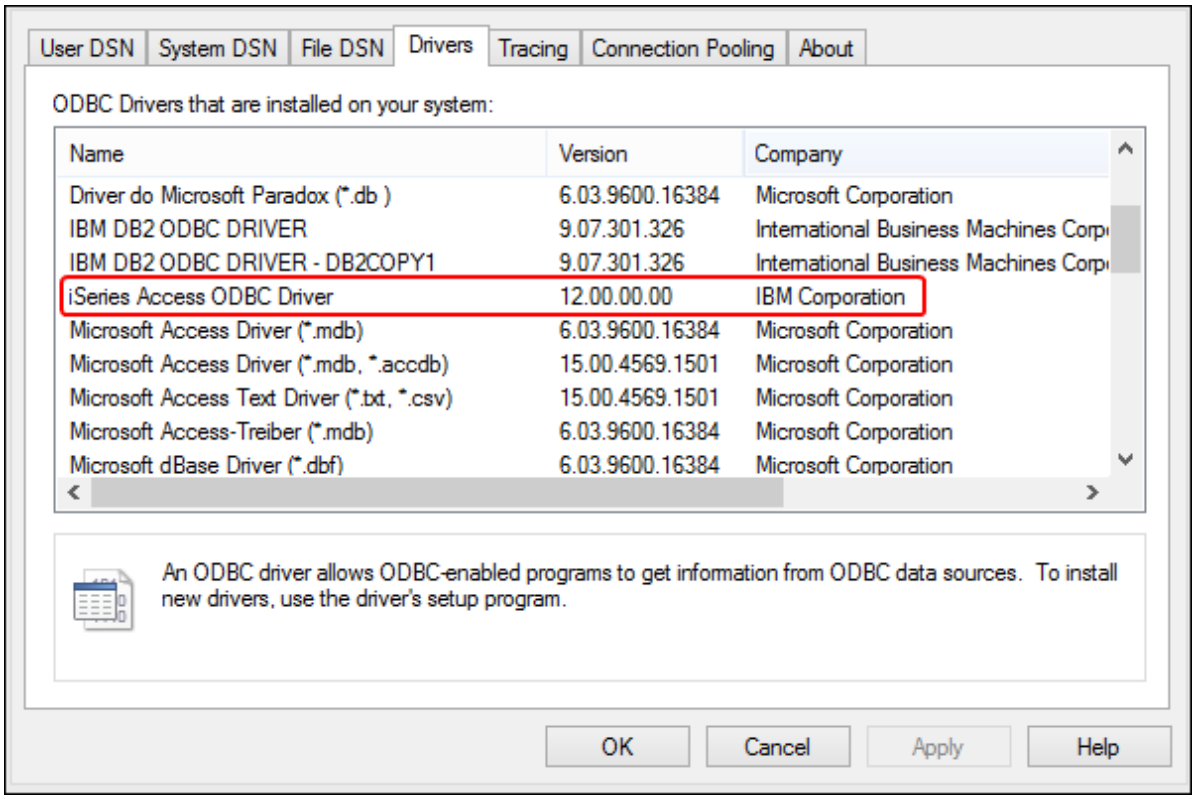
7. Haga clic en **Conectarse**.

4.2.1.10.6 IBM DB2 para i (ODBC)

Este ejemplo explica cómo conectarse a una base de datos IBM DB2 for i por ODBC.

Requisitos:

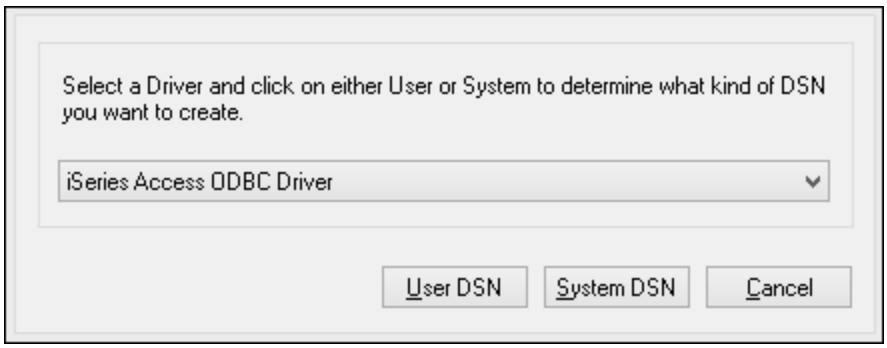
- *IBM System i Access for Windows* está instalado en el sistema (para este ejemplo usamos *IBM System i Access for Windows V6R1M0*). Compruebe que el controlador ODBC está en el equipo (consulte Ver los controladores ODBC disponibles 182).



- Disponer de los datos de conexión: dirección IP del servidor de BD, nombre de usuario y contraseña.
- Ejecute *System i Navigator* y siga las instrucciones que aparecen en pantalla para crear una conexión nueva. Llegado el momento de especificar un sistema, introduzca la dirección IP del servidor de BD. Tras crear la conexión se recomienda verificarla (haga clic en la conexión y seleccione **Archivo** > **Diagnóstico** > **Verificar conexión**). Si recibe errores de conexión, póngase en contacto con el administrador del servidor de BD.

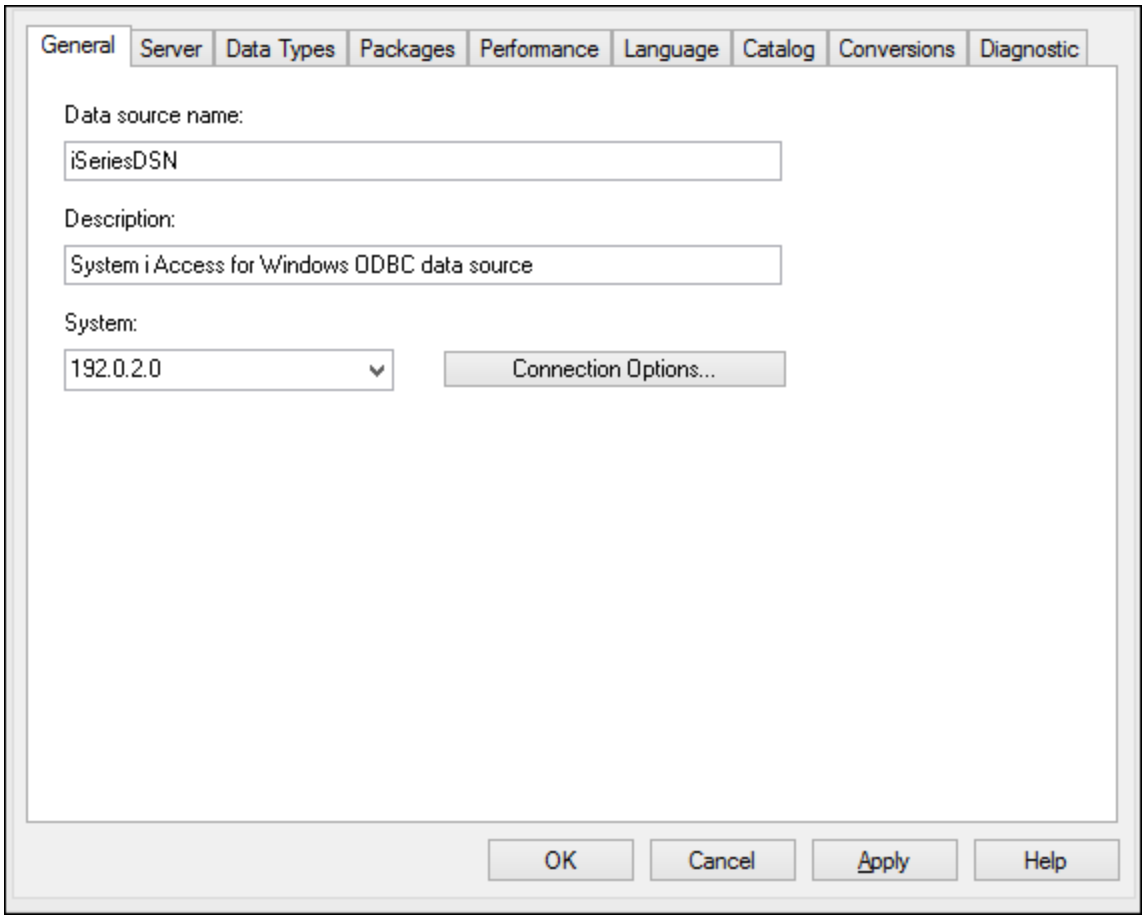
Para conectarse a IBM DB2 for i:

1. Inicie el asistente para la conexión de base de datos 163.
2. Haga clic en **Conexiones ODBC**.
3. Seleccione *DSN de usuario* (o *DSN de sistema* o *DSN de archivo*).
4. Haga clic en **Agregar** .
5. Seleccione el controlador **iSeries Access ODBC Driver** de la lista y después haga clic en **DSN de usuario** (o **DSN de sistema**).



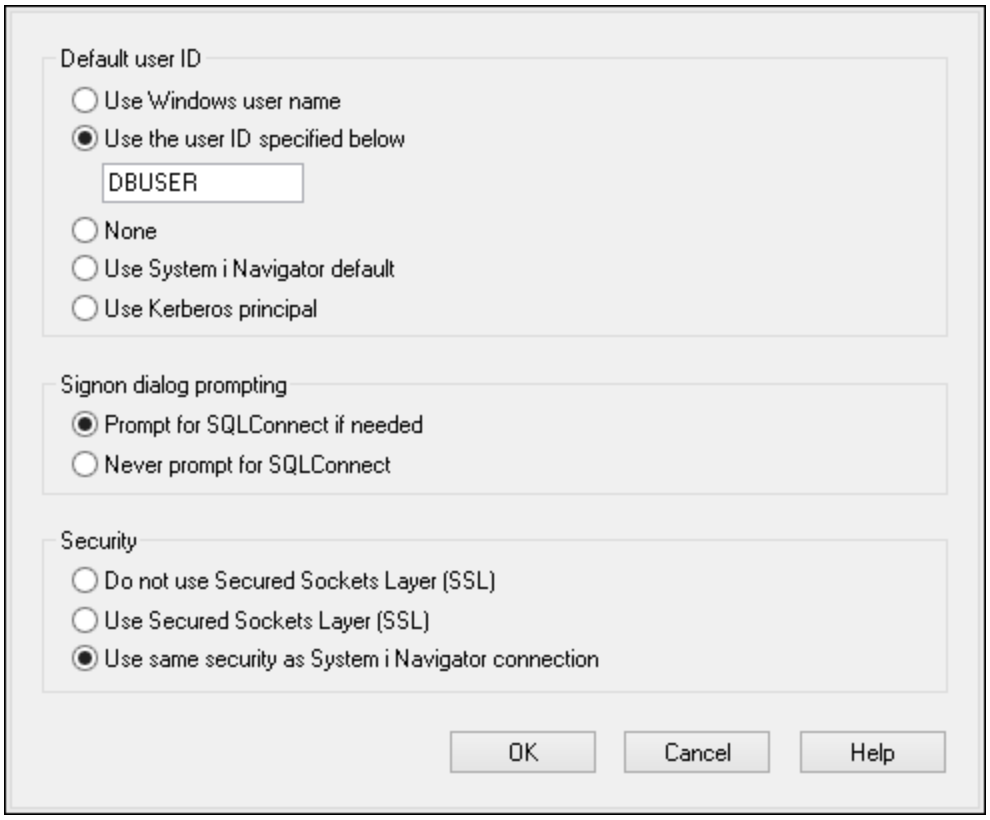
6. Introduzca el DSN y seleccione la conexión del cuadro combinado *Sistema*. En este ejemplo el DSN es **iSeriesDSN** y el sistema es **192.0.2.0**.

 © 2016-2022 Altova GmbH



Nota: cuando se añade un origen de datos ODBC para una base de datos *IBM DB2 for i*, se establece una marca predeterminada que permite tiempos de espera de consulta. Esta opción de configuración se puede deshabilitar para que los archivos de asignación de MapForce carguen correctamente. Para deshabilitar esta opción haga clic en la pestaña de rendimiento Performance, después en el botón de opciones avanzadas **Advanced** y desactive la casilla **Allow query timeout** (permitir tiempo de espera de consulta).

7. Haga clic en el botón **Opciones de conexión** y seleccione *Utilizar el Id. de usuario indicado* y escriba el nombre del usuario de la BD (en este ejemplo es **DBUSER**).



8. Haga clic en **Aceptar**. El origen de datos nuevo aparece ahora en la lista de DSN.
9. Haga clic en **Conectarse**.
10. Introduzca el nombre de usuario y la contraseña de la BD cuando sea necesario y después haga clic en **Aceptar**.

4.2.1.10.7 IBM Informix (JDBC)

Este ejemplo explica cómo conectarse a una base de datos IBM por JDBC.

Requisitos:

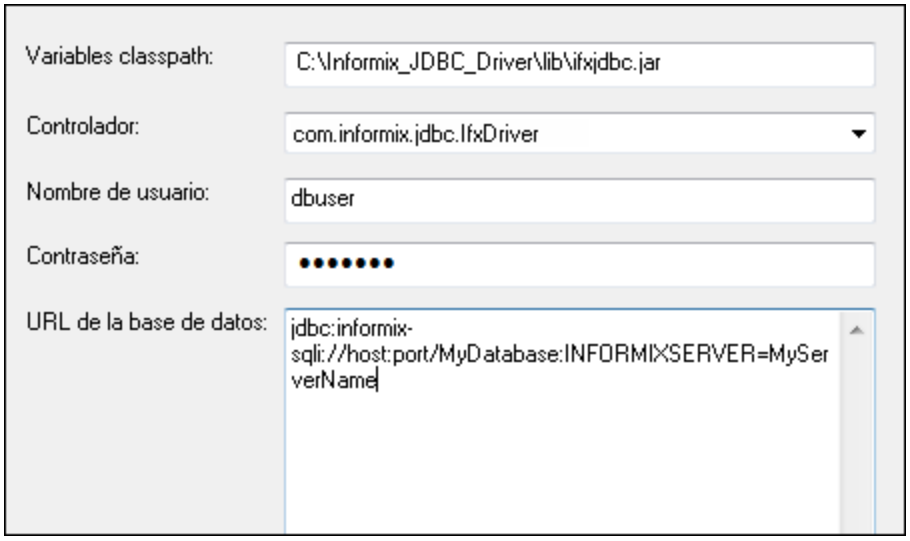
- Debe tener instalado Java Runtime Environment (JRE) o Java Development Kit (JDK). Este último puede ser el JDK de Oracle o uno de código abierto, como Oracle OpenJDK. MapForce identifica la ruta de acceso al equipo virtual Java (JVM) a partir de estas ubicaciones, en este orden: a) la ruta personal de acceso al JVM que puede indicar en en las **Opciones** (véase el apartado Java[1052]); b) la ruta de acceso al JVM que se encuentra en el registro; c) la variable de entorno `JAVA_HOME`.
- Asegúrese de que la plataforma de MapForce (32 o 64 bits) es la misma que la de JRE/JDK.
- El controlador JDBC (se trata de un archivo .jar o varios archivos .jar que ofrecen conectividad con la base de datos) debe estar disponible en el sistema operativo. En este ejemplo se utiliza el controlador JDBC IBM Informix versión 3.70. Consulte la documentación del controlador para ver las instrucciones de instalación o la guía *IBM Informix JDBC Driver Programmer's Guide*.

 © 2016-2022 Altova GmbH

- Disponer de los datos de conexión: host, nombre del servidor Informix, base de datos, puerto, nombre de usuario y contraseña.

Para conectarse a IBM Informix por JDBC:

1. Inicie el asistente para la conexión de base de datos 163.
2. Haga clic en **Conexiones JDBC**.
3. En el campo *Variables classpath* debe introducir la ruta de acceso del archivo .jar que ofrece conectividad con la base de datos. Si fuera necesario, también puede introducir una lista de rutas de archivo .jar separadas por caracteres de punto y coma. En este ejemplo, el archivo .jar está ubicado en esta ruta de acceso: `C:\Informix_JDBC_Driver\lib\ifxjdbc.jar`. Tenga en cuenta que este campo puede dejarse en blanco si añadió la ruta de acceso de los archivos .jar a la variable de entorno CLASSPATH del sistema operativo (*véase Configurar la variable CLASSPATH* 186).
4. En el campo *Controlador* seleccione **com.informix.jdbc.IfxDriver**. Recuerde que esta entrada solo estará disponible si se encuentra una ruta de archivo .jar válida en el campo Variables *classpath* o en la variable de entorno CLASSPATH del sistema operativo.



5. Introduzca el nombre de usuario y la contraseña de la base de datos.
6. Introduzca la cadena de conexión para el servidor de BD en el cuadro de texto *URL de la base de datos* (reemplace lo valores resaltados con los de su base de datos).

```
jdbc:informix-sqli://nombreHost:puerto/nombreBD:INFORMIXSERVER=myserver;
```

7. Haga clic en **Conectarse**.

4.2.1.10.8 MariaDB (ODBC)

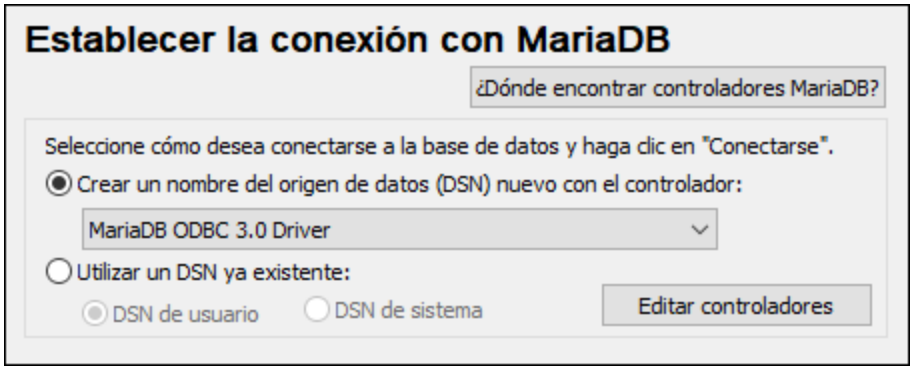
Este ejemplo explica cómo conectarse a un servidor de base de datos MariaDB por ODBC.

Requisitos:

- Debe tener instalado el conector de MariaDB para ODBC (https://downloads.mariadb.org/connector-odbc/).
- Disponer de los datos de conexión: host, base de datos, puerto, nombre de usuario y contraseña.

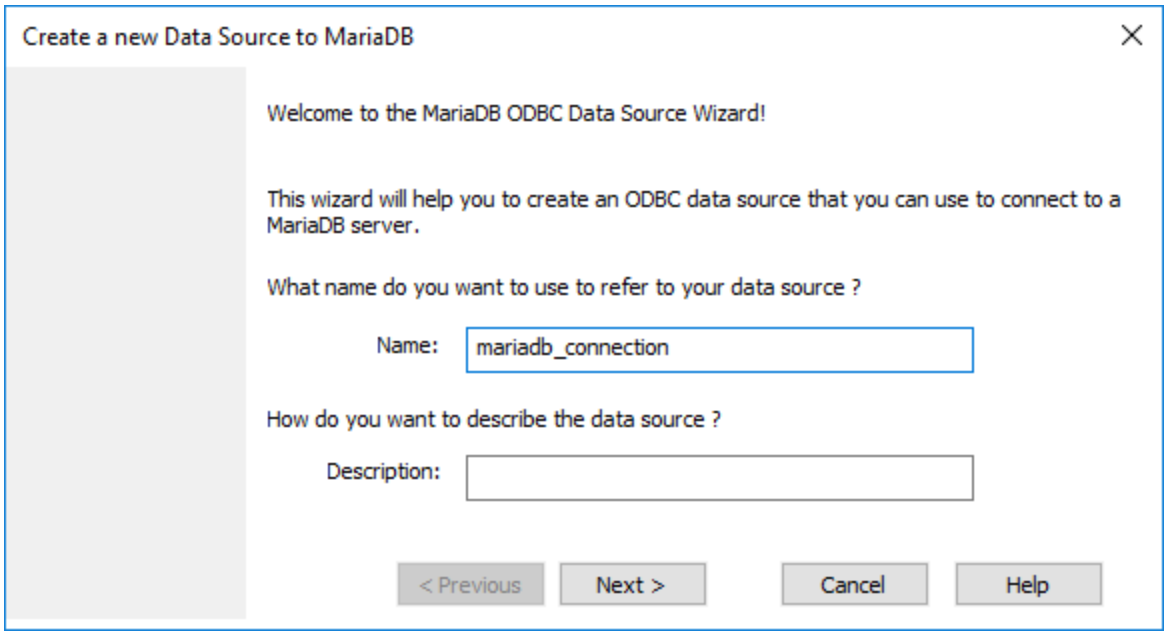
Para conectarse a MariaDB por ODBC:

1. Inicie el asistente para la conexión de base de datos 163.
2. Seleccione *MariaDB (ODBC)* y después haga clic en **Siguiente**.

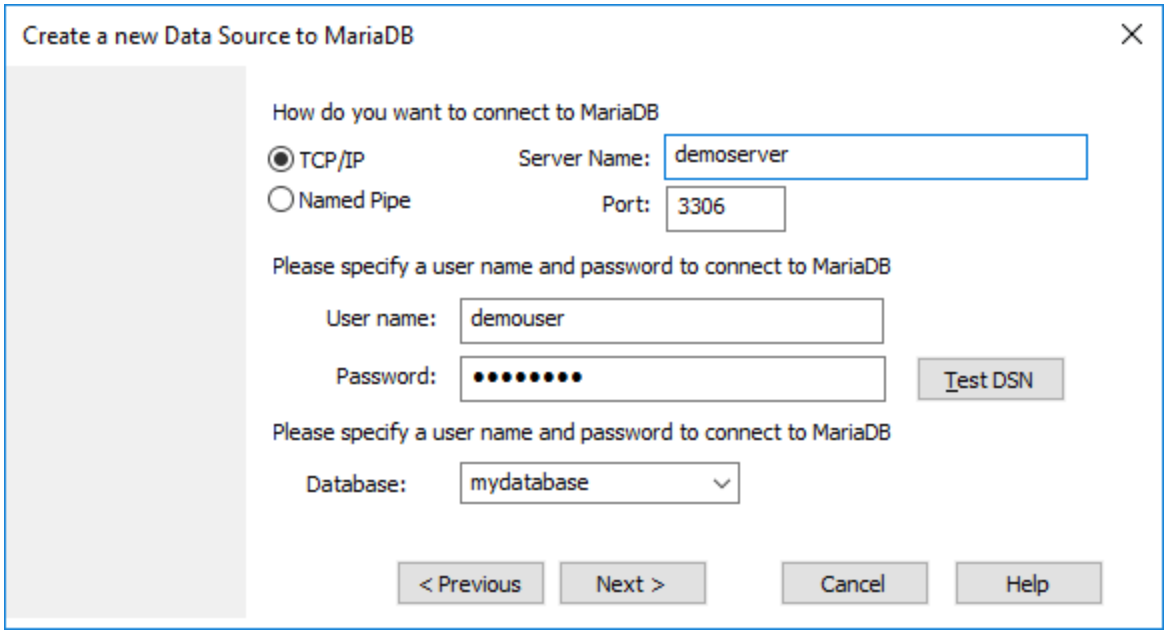


3. Seleccione *Crear un nombre del origen de datos (DSN) nuevo con el controlador* y elija el controlador **MariaDB ODBC 3.0 Driver**. Si este controlador no aparece en la lista, haga clic en **Editar controladores** y seleccione cualquier controlador MariaDB (la lista contiene todos los controladores ODBC que están instalados en el sistema operativo).
4. Haga clic en **Conectarse**.

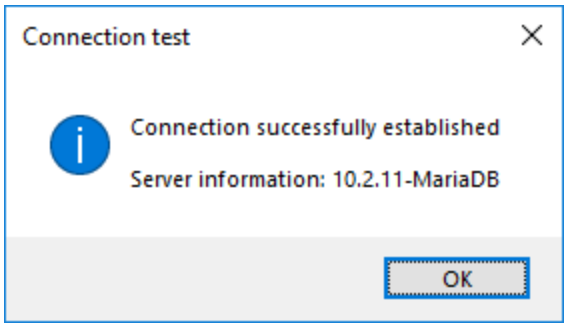
 © 2016-2022 Altova GmbH



5. Introduzca el nombre y, si quiere, una descripción que le ayude a identificar este origen de datos ODBC más adelante.



6. Rellene las credenciales de la conexión de base de datos (servidor TCP/IP, usuario, contraseña), seleccione una base de datos y después haga clic en **Probar DSN**. Cuando se establezca la conexión aparecerá este mensaje:



7. Haga clic en **Siguiente** y siga los pasos del asistente hasta el final. Dependiendo del caso, puede que sean necesarios más parámetros (p. ej. certificados SSL si se conecta a MariaDB a través de una conexión segura).

Nota: si el servidor de base de datos es remoto, deberá estar configurado por el administrador para que acepte conexiones remotas desde la dirección IP de su equipo.

4.2.1.10.9 Microsoft Access (ADO)

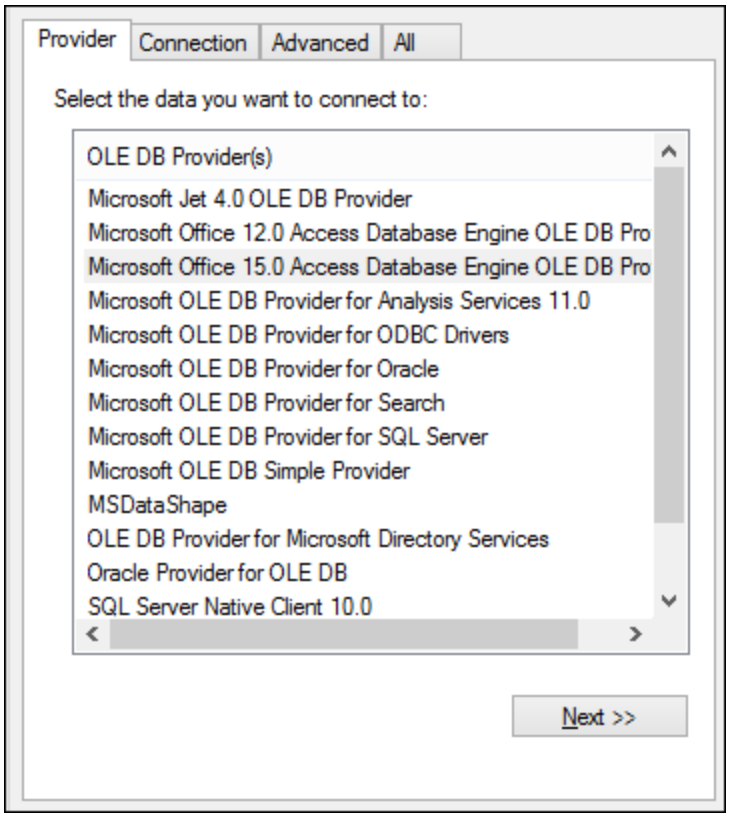
Lo más sencillo para conectarse a una base de datos Microsoft Access es seguir las instrucciones del asistente para la conexión de base de datos y buscar el archivo de base de datos, como se muestra en el apartado Conectarse a una base de datos Microsoft Access 170. También puede configurar explícitamente una conexión ADO, como muestra este ejemplo. Esta segunda opción se recomienda si la base de datos está protegida con contraseña.

También puede conectarse a Microsoft Access por ODBC, pero esto implica algunas restricciones por lo que recomendamos evitar este tipo de conexión.

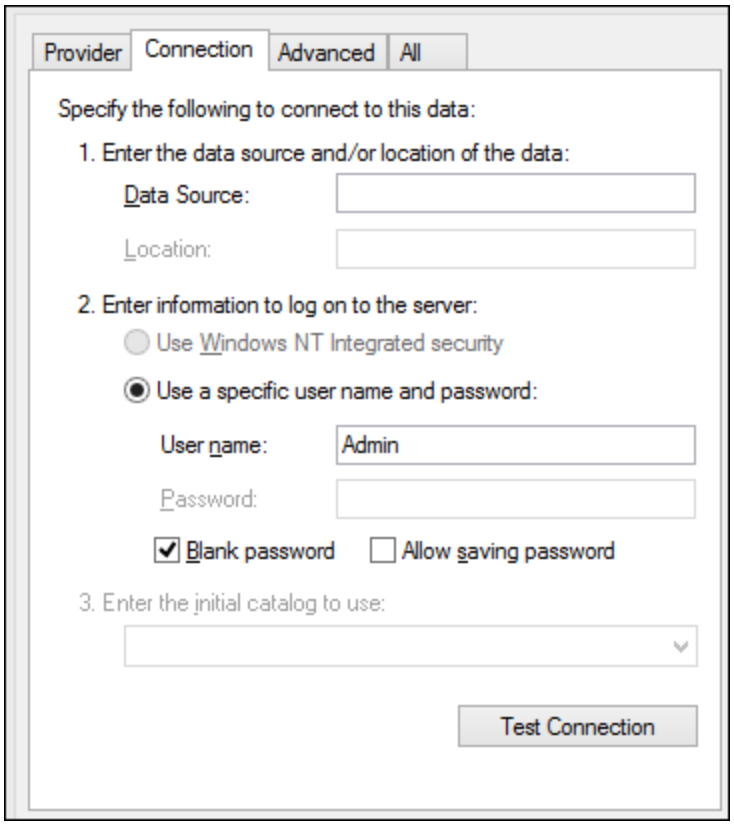
Para conectarse a una base de datos Microsoft Access protegida con contraseña:

1. Inicie el asistente para la conexión de base de datos 163.
2. Haga clic en **Conexiones ADO**.
3. Ahora haga clic en **Generar**.

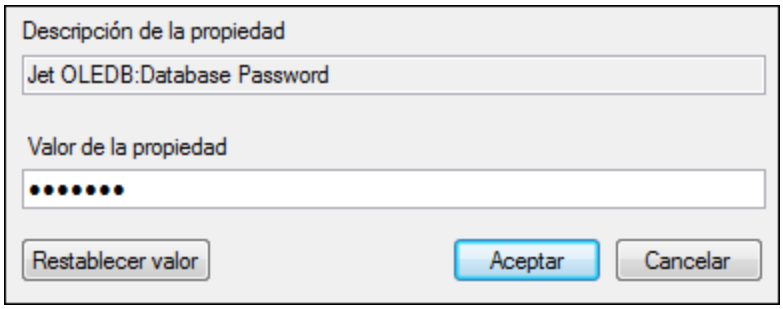
 © 2016-2022 Altova GmbH



4. Seleccione **Microsoft Office 15.0 Access Database Engine OLE DB Provider** y haga clic en **Siguiente**.



5. En el cuadro de texto Origen de datos introduzca la ruta de acceso del archivo de Microsoft Access en formato UNC, por ejemplo, `\\myserver\\mynetworkshare\Reports\Revenue.accdb`, donde `myserver` es el nombre del servidor y `mynetworkshare` el nombre del recurso compartido de red.
6. En la pestaña *Todos* haga doble clic en la propiedad **Jet OLEDB:Database Password** e introduzca la contraseña de la base de datos.



Nota: si no es capaz de establecer la conexión, busque el archivo de información del grupo de trabajo (`System.MDW`) de su perfil de usuario y establezca el valor de la propiedad **Jet OLEDB: System database** en la ruta de acceso del archivo `System.MDW`.

4.2.1.10.10 Microsoft SQL Server (ADO)

En este ejemplo explicamos cómo conectarse a una base de datos SQL Server con ADO. Estas instrucciones asumen que usa el driver recomendado para SQL Server **Microsoft OLE DB Driver for SQL Server**

 © 2016-2022 Altova GmbH

(MSOLEDBSQL), que puede descargar en https://docs.microsoft.com/en-us/sql/connect/oledb/download-oledb-driver-for-sql-server?view=sql-server-ver15.

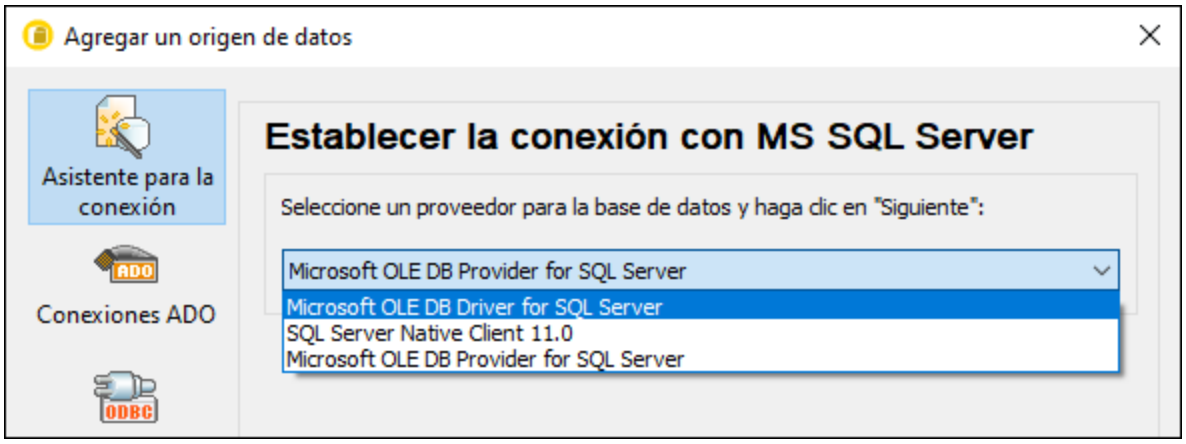
Antes se seguir estas instrucciones, asegúrese de que descarga e instala el proveedor que mencionado más arriba en su equipo de trabajo. El proveedor ADO debe ser de la misma versión de plataforma que MapForce (32 bits o 64 bits).

Si quiere usar otro proveedor ADO, como **QL Server Native Client (SQLNCLI)** or **Microsoft OLE DB Provider for SQL Server (SQLOLEDB)**, las instrucciones son parecidas; sin embargo, estos proveedores están obsoletos, por lo que no se recomiendan. Además, para poder conectarse correctamente con un proveedor obsoleto puede que necesite configurar otras propiedades de conexión, como se describe en Configurar las propiedades de vínculo de datos de SQL Server 171.

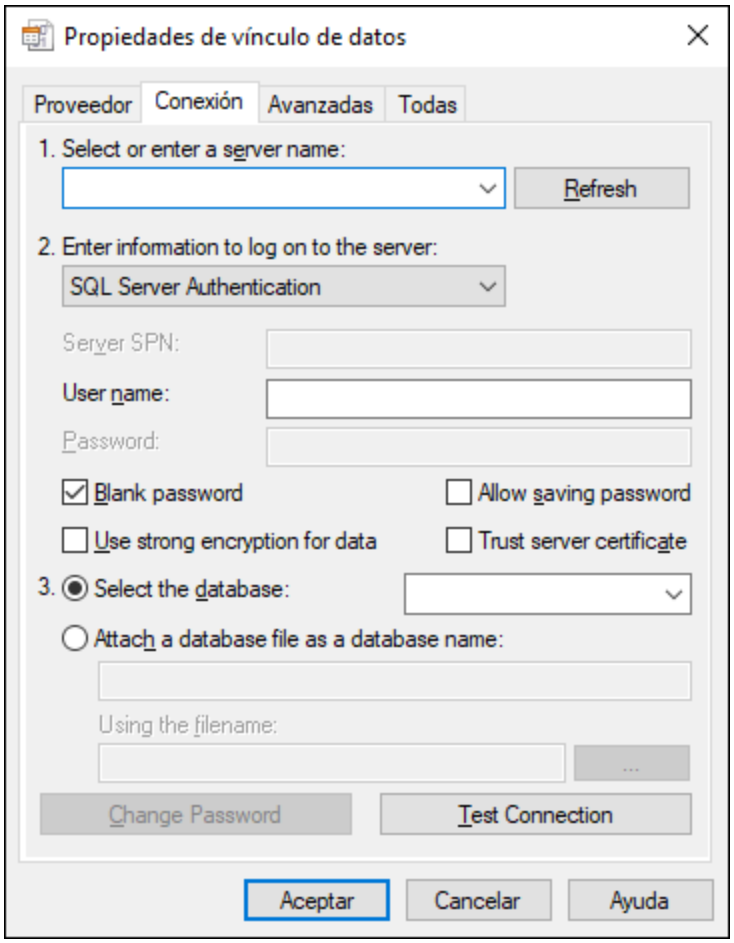
Es un problema conocido que el proveedor de BD para SQL Server **Microsoft OLE DB Provider for SQL Server (SQLOLEDB)** tiene dificultades para enlazar parámetros de consultas complejas como las expresiones comunes de tabla (CTE) e instrucciones SELECT anidadas.

Para conectarse a SQL Server:

1. Inicie el asistente de conexión 163.
2. Seleccione **Microsoft SQL Server (ADO)** y después haga clic en **Siguiente**. Verá que aparece la lista de proveedores ADO disponibles. En este ejemplo usaremos **Microsoft OLE DB Driver for SQL Server**. Si no está en la lista, asegúrese de que está instalado en su equipo, como hemos mencionado antes.

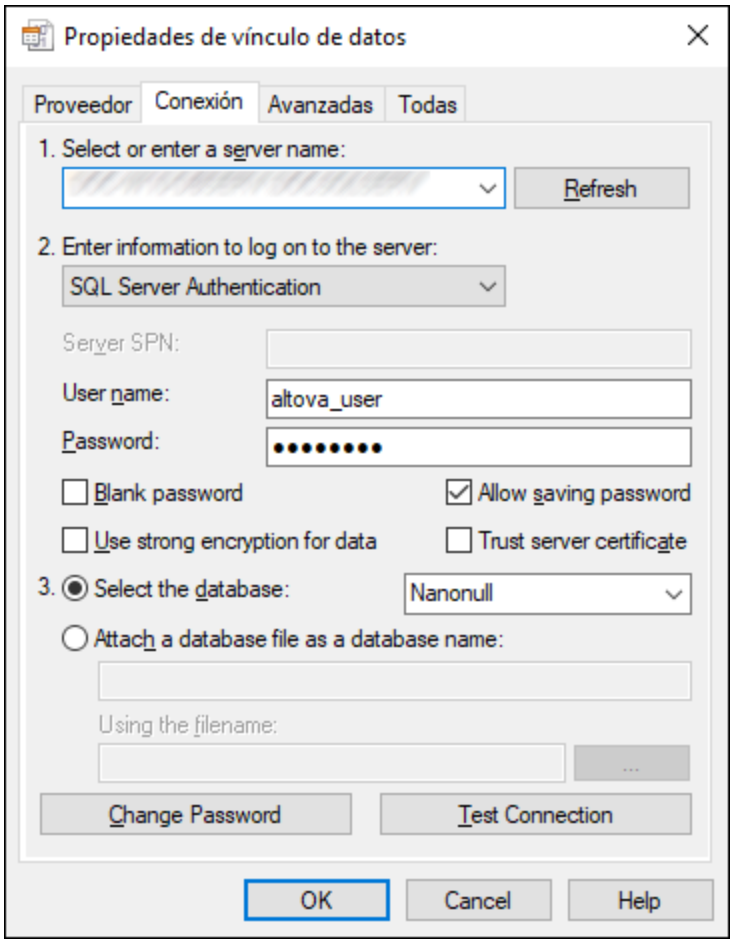


3. Haga clic en **Siguiente**. Se abre el cuadro de diálogo "Propiedades de enlace de datos".



4. Seleccione o introduzca el nombre del servidor de BD, por ejemplo **SQLSERV01**. Si se conecta a una instancia de SQL Server con nombre, el nombre del servidor se parecerá a: **SQLSERV01\SOMEINSTANCE**.
5. Si configuró el servidor de BD para que permita conexiones de usuarios autenticados en el dominio Windows, seleccione **Autenticación Windows**. De lo contrario seleccione **Autenticación SQL Server**, desmarque la casilla *Contraseña en blanco* e introduzca las credenciales de BD en los campos correspondientes.
6. Marque la casilla *Permitir guardar contraseña* y seleccione la BD a la que se quiere conectar (en este ejemplo, "Nanonull").

 © 2016-2022 Altova GmbH



7. Para comprobar la conexión haga clic en **Comprobar conexión**. Este paso es opcional, pero recomendamos no saltárselo.
8. Haga clic en **Aceptar**.

4.2.1.10.11 Microsoft SQL Server (ODBC)

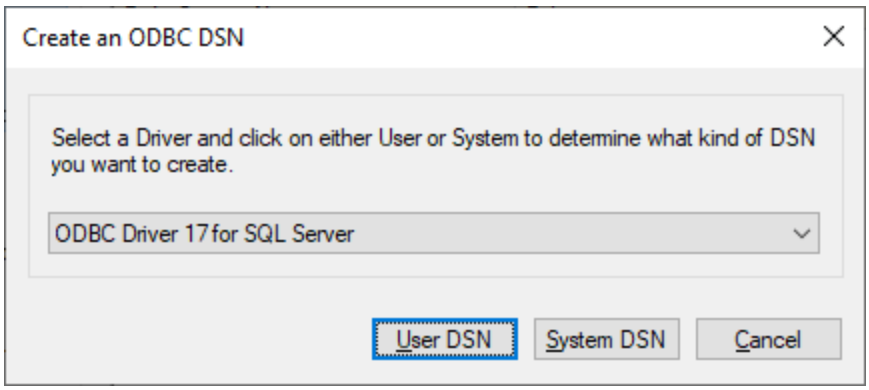
En este ejemplo aprenderá a conectar una base de datos a SQL Server con ODBC.

Requisitos previos:

- Descargue e instale el controlador **Microsoft ODBC Driver for SQL Server**, que encontrará en el sitio web de Microsoft (*véase* *https://docs.microsoft.com/en-us/SQL/connect/odbc/download-odbc-driver-for-sql-server*).Este ejemplo usa **Microsoft ODBC Driver 17 for SQL Server** para conectarse a la base de datos **SQL Server 2016**. Puede descargar un controlador distinto en función de la versión de SQL Server a la que se quiera conectar. Para información sobre las versiones del controlador ODBC compatibles con su base de datos SQL Server consulte los requisitos del sistema del controlador.

Para conectarse a SQL Server con ODBC:

1. Inicie el asistente para la conexión de base de datos 163.
2. Haga clic en **Conexiones ODBC**.
3. Seleccione **DSN de usuario** (o **DSN de sistema** si tiene privilegios de administrador) y haga clic en **Agregar** .
4. Seleccione el controlador de la lista. Tenga en cuenta que este solo aparece en esta lista una vez se ha instalado.

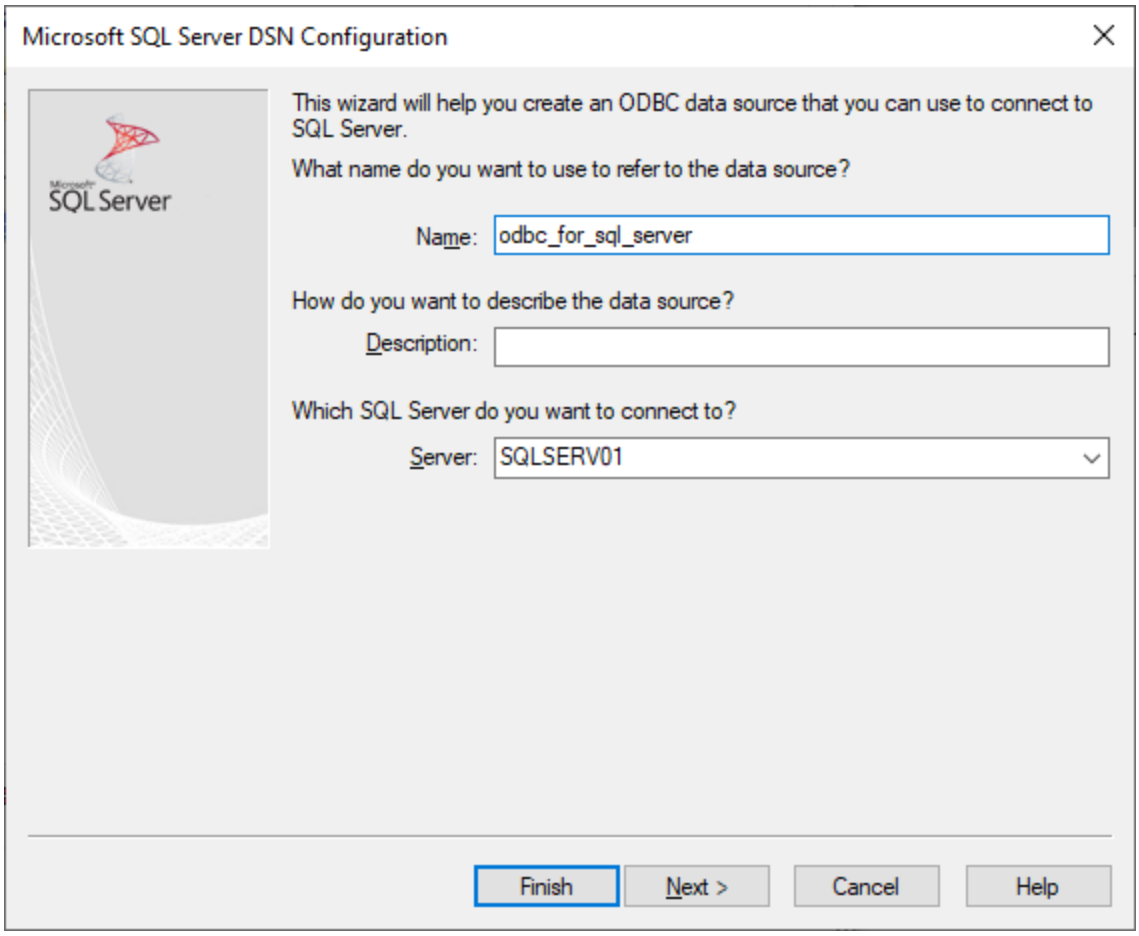


5. Haga clic en **DSN de usuario** (o en **DSN de sistema** si está creando un sistema DNS).

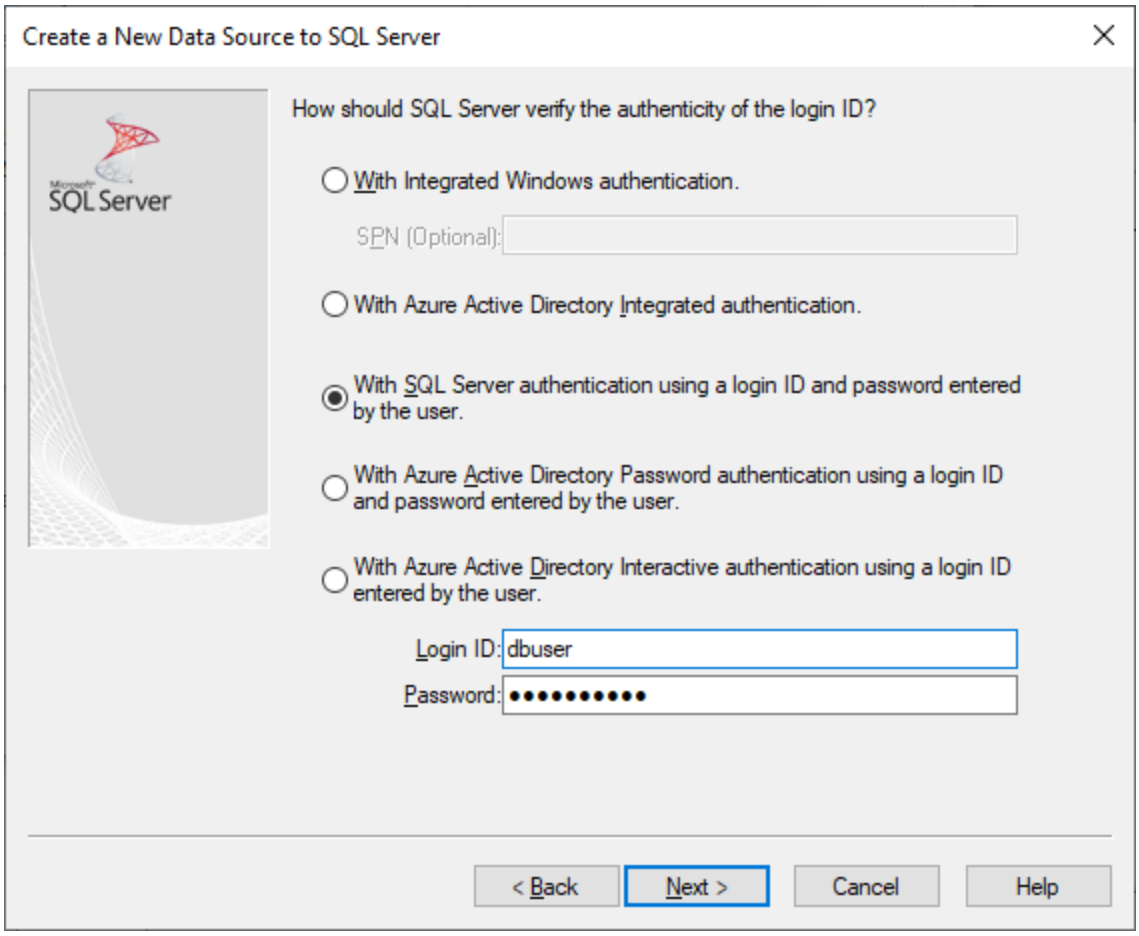
> Para crear un **Sistema DSN** necesita ejecutar MapForce como administrador. Si lo que quiere es crear este sistema, salga del asistente y vuelva a ejecutar MapForce como administrador y siga los pasos desde el principio.

6. Introduzca un nombre y, si quiere, una descripción que identifique esta conexión. A continuación seleccione en la lista el SQL Server al que se quiere conectar (en este ejemplo, **SQLSERV01**).

 © 2016-2022 Altova GmbH

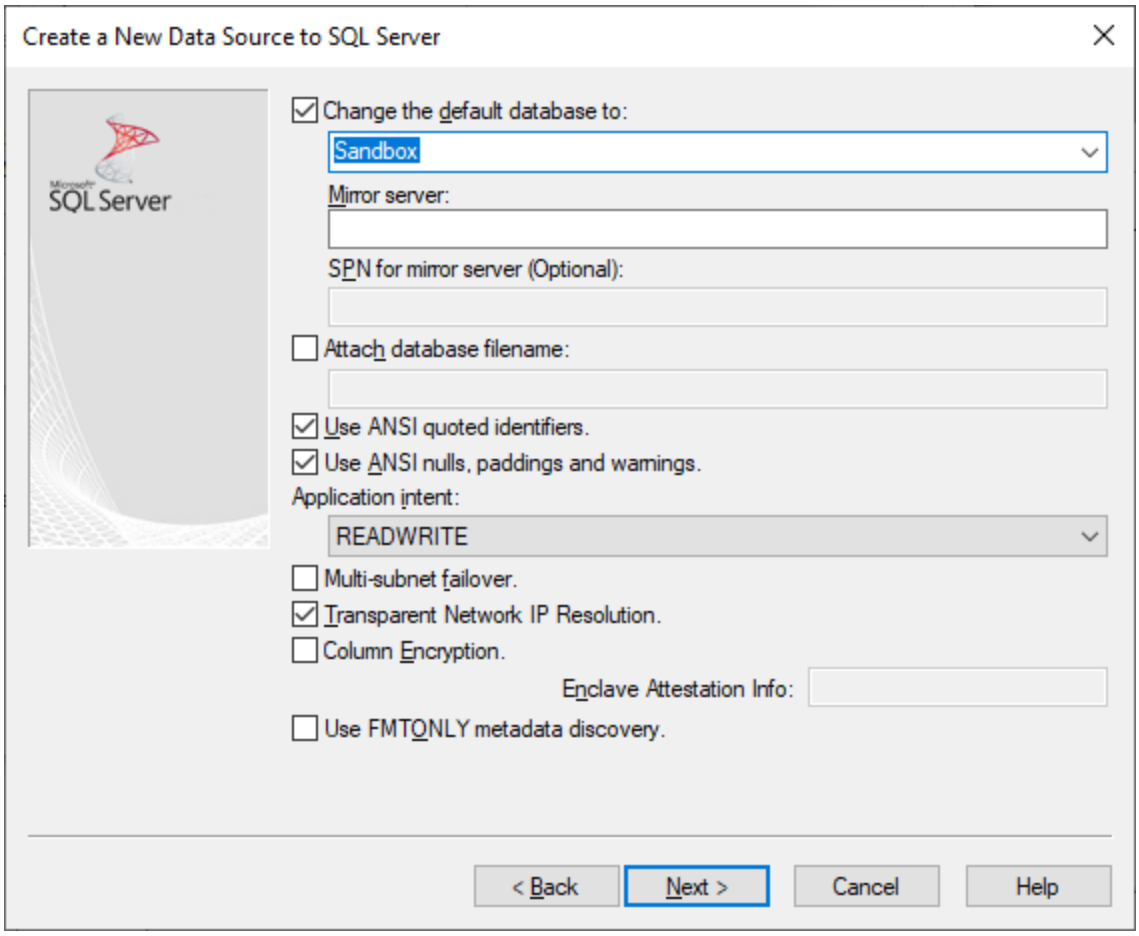


7. Si el servidor de BD se configuró para que permita conexiones de usuarios autenticados en el dominio Windows, seleccione **Con autenticación integrada de Windows**. De lo contrario, seleccione la opción que necesite. En este ejemplo usamos **Con autenticación de SQL Server...**, que necesita que se introduzcan el nombre de usuario y la contraseña en los campos correspondientes.

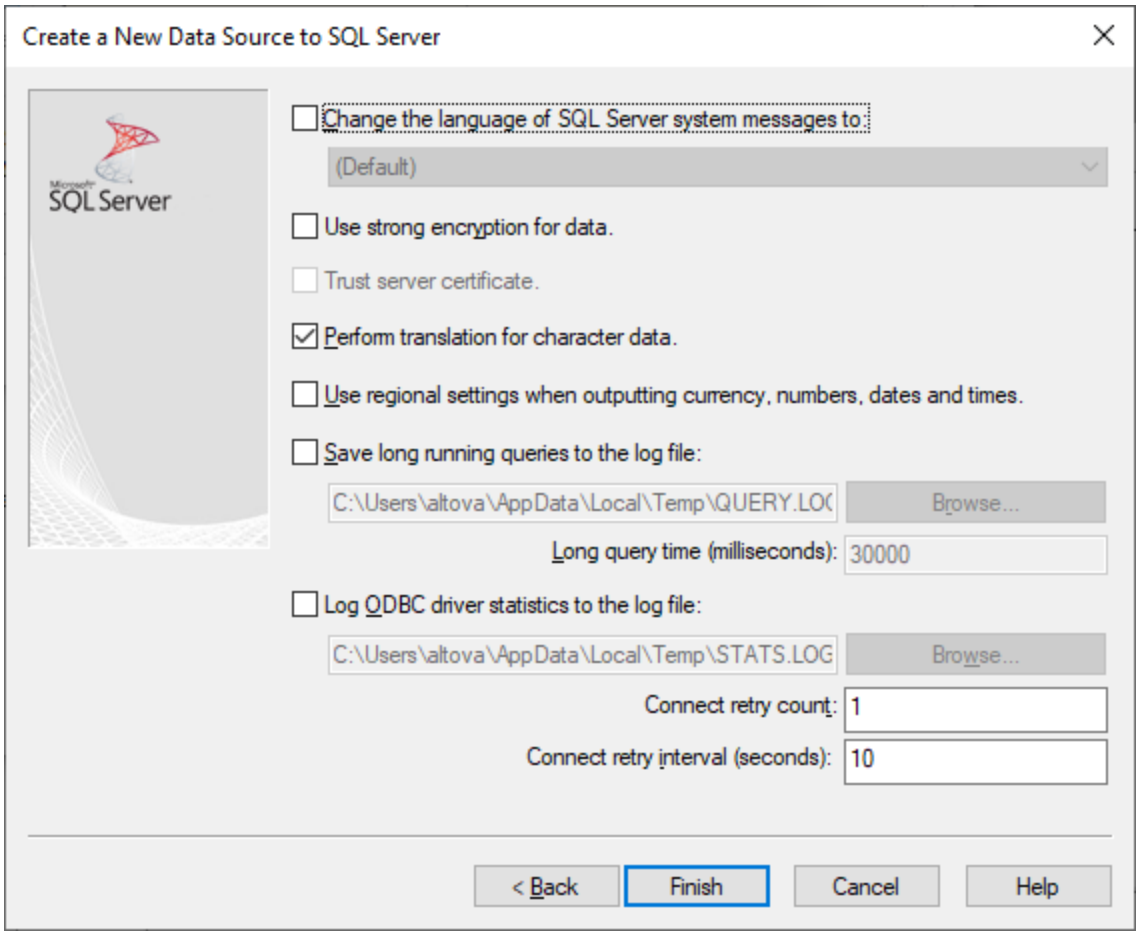


8. También puede marcar la casilla **Cambiar la base de datos predeterminada a** e introducir el nombre de la base de datos a la que se va a conectar (en este ejemplo, **Sandbox**).

© 2016-2022 Altova GmbH

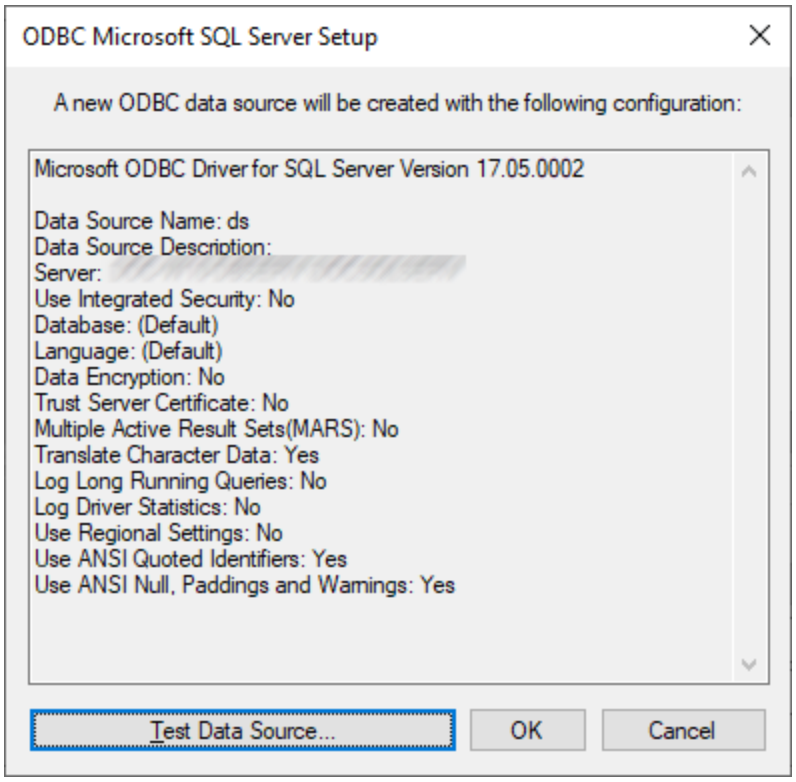


9. Haga clic en **Siguiente** y, si quiere, configure el resto de los parámetros de esta conexión.

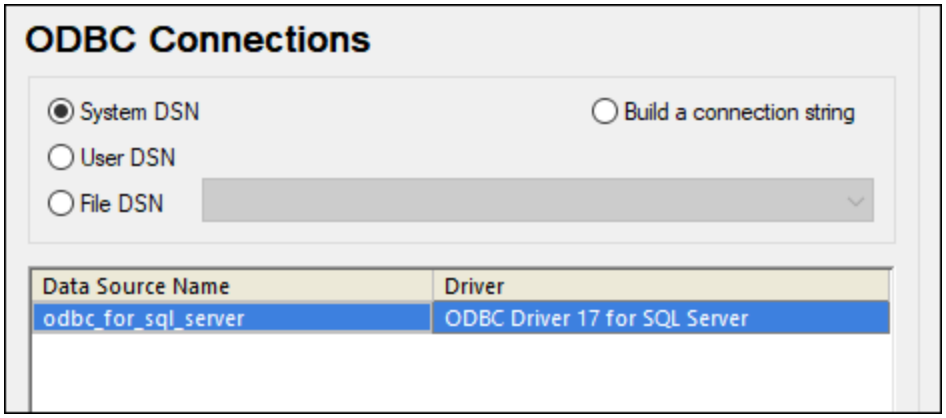


10. Haga clic en **Finalizar**. Aparece un cuadro de diálogo de conformación que contiene los detalles de la conexión.

© 2016-2022 Altova GmbH



11. Haga clic en **Aceptar**. Ahora la fuente de datos aparece en la lista de fuentes de datos de **Usuario** o **Sistema**, según la configuración; por ejemplo:



4.2.1.10.12 MySQL (ODBC)

Este ejemplo explica cómo conectarse a un servidor de base de datos MySQL desde un equipo Windows con el controlador ODBC. El controlador ODBC MySQL no está disponible en Windows así que deberá descargarlo e instalarlo por separado. En este ejemplo usamos el conector MySQL/ODBC 8.0.

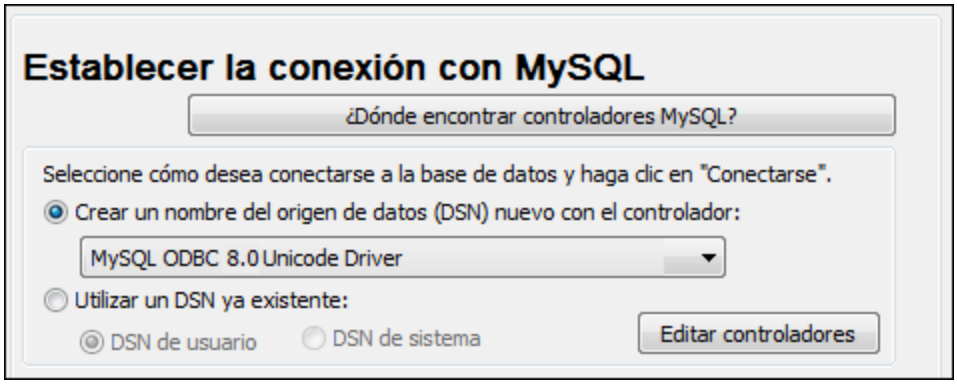
Requisitos previos:

- El controlador MySQL ODBC está instalado en el sistema. Consulte la documentación de MySQL para conocer qué controlador debe usar para su versión del servidor de base de datos (*véase https://dev.mysql.com/downloads/connector/odbc/*).
- Disponer de los datos de conexión: host, base de datos, puerto, nombre de usuario y contraseña.

Conector MySQL/ODBC 8.0
Si instala el controlador MySQL ODBC para plataformas de 64 bits, asegúrese de que también instala la versión de MapForce para plataformas de 64 bits.

Para conectarse a MySQL por ODBC:

1. Inicie el asistente para la conexión de base de datos 163.
2. Seleccione la opción *MySQL (ODBC)* y después haga clic en **Siguiente**.

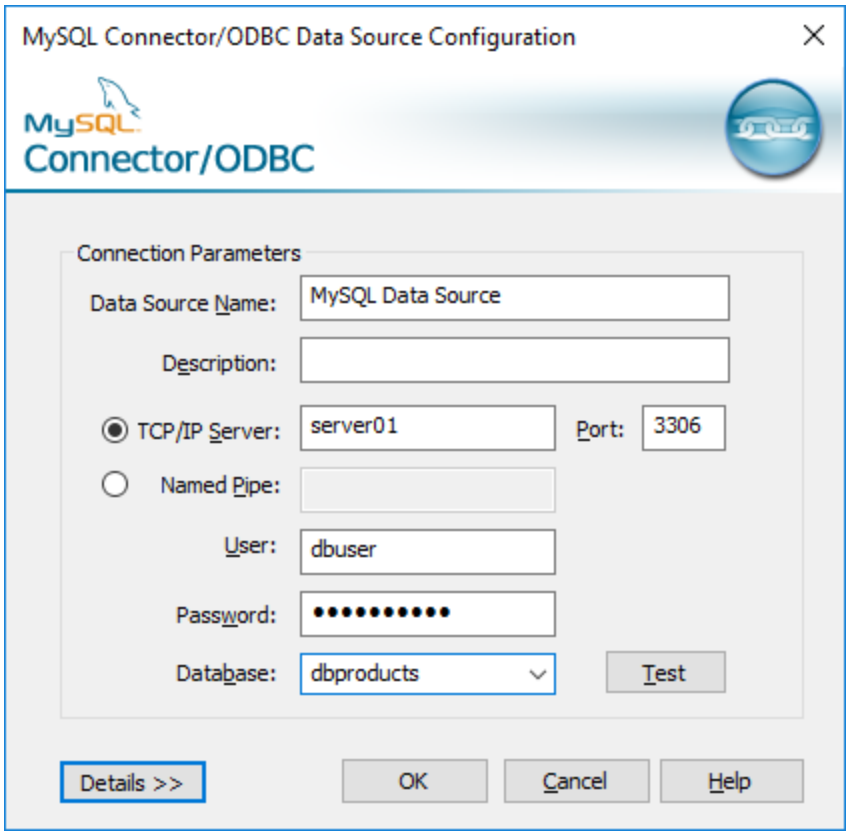


3. Seleccione la opción *Crear un DNS nuevo con el controlador* y seleccione un controlador MySQL. Si no hay ningún controlador MySQL en la lista, haga clic en **Editar controladores** y seleccione uno (la lista contiene todos los controladores ODBC que están instalados en el sistema).

   Si instaló la versión de MapForce para plataformas de 64 bits se mostrarán los controladores ODBC en la lista. De lo contrario se mostrarán los controladores para la versión de 32 bits. Consulte también el apartado Ver los controladores ODBC disponibles 182.

4. Haga clic en **Conectarse**.

 © 2016-2022 Altova GmbH



5. En el cuadro de texto *Nombre del origen de datos* introduzca un nombre que le ayude a identificar este origen de datos ODBC más adelante.
6. Rellene las credenciales de la conexión de BD (servidor TCP/IP, usuario, contraseña), seleccione una base de datos y haga clic en **Aceptar**.

Nota: si el servidor de BD es remoto, el administrador del servidor debe configurarlo para que acepte conexiones remotas desde la dirección IP de su equipo. Además, si hace clic en **Detalles>>**, podrá configurar algunos parámetros más. Consulte la documentación del controlador antes de cambiar los valores predeterminados.

4.2.1.10.13 Oracle (JDBC)

Este ejemplo explica cómo conectarse a un servidor de base de datos Oracle desde un equipo cliente, usando la interfaz JDBC. La conexión se crea como una conexión Java pura, usando el paquete **Oracle Instant Client Package (Basic)** que se puede descargar del sitio web de Oracle. La ventaja de este tipo de conexión es que solamente exige el entorno Java y las bibliotecas .jar que vienen con el paquete Oracle Instant Client Package. Es decir, no es necesario instalar ni configurar clientes de base de datos más complejos.

Requisitos:

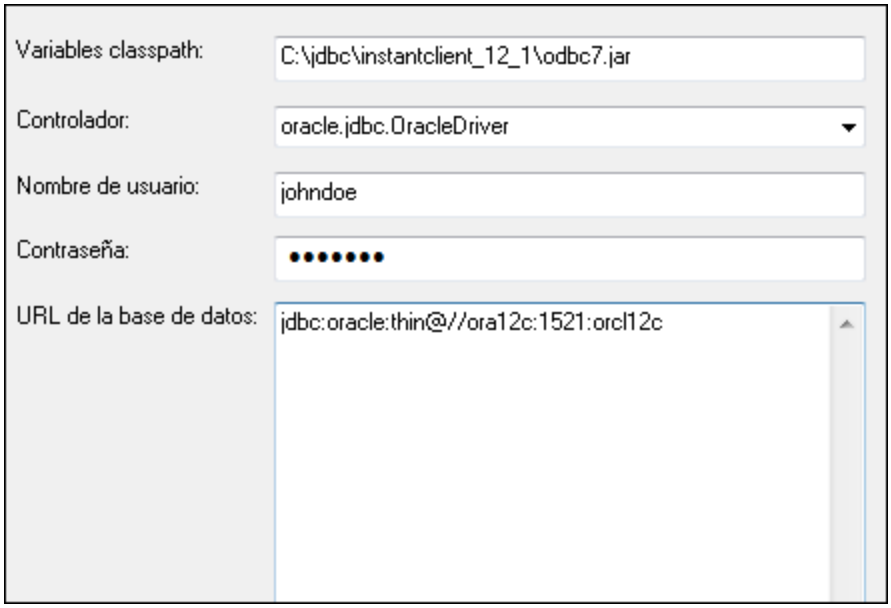
- JRE (Java Runtime Environment) o Java Development Kit (JDK) está instalado. Este último puede ser el JDK de Oracle o uno de código abierto, como Oracle OpenJDK. MapForce identifica la ruta de

acceso al equipo virtual Java (JVM) a partir de estas ubicaciones, en este orden: a) la ruta personal de acceso al JVM que puede indicar en en las **Opciones** (véase el apartado Java[1052]); b) la ruta de acceso al JVM que se encuentra en el registro; c) la variable de entorno `JAVA_HOME`.

- Asegúrese de que la plataforma de MapForce (32 o 64 bits) es la misma que la de JRE/JDK.
- El paquete **Oracle Instant Client Package (Basic)** está disponible en el sistema operativo. El paquete se puede descargar del sitio web oficial de Oracle. En este ejemplo utilizamos la versión 12.1.0.2.0 para Windows de 32 bits y, por tanto, Oracle JDK de 32 bits.
- Disponer de los datos de conexión: host, puerto, nombre del servicio, nombre de usuario y contraseña.

Para conectarse a Oracle a través del paquete Instant Client Package:

1. Inicie el asistente para la conexión de base de datos[163].
2. Haga clic en **Conexiones JDBC**.
3. En el campo *Variables classpath* debe introducir la ruta de acceso del archivo .jar que ofrece conectividad con la base de datos. Si fuera necesario, también puede introducir una lista de rutas de archivo .jar separadas por caracteres de punto y coma. En este ejemplo, el archivo .jar está ubicado en esta ruta de acceso: `C:\jdbc\instantclient_12_1\ojdbc7.jar`. Tenga en cuenta que este campo puede dejarse en blanco si añadió la ruta de acceso de los archivos .jar a la variable de entorno CLASSPATH del sistema operativo (*véase Configurar la variable CLASSPATH*[186]).
4. En el campo *Controlador* seleccione **oracle.jdbc.OracleDriver** o **oracle.jdbc.driver.OracleDriver**. Recuerde que esta entrada solo estará disponible si se encuentra una ruta de archivo .jar válida en el campo Variables *classpath* o en la variable de entorno CLASSPATH del sistema operativo.
5. Introduzca el nombre de usuario y contraseña de la base de datos.



6. Introduzca la cadena de conexión para el servidor de BD en el cuadro de texto *URL de la base de datos* (reemplace lo valores resaltados con los de su servidor de base de datos).

```
jdbc:oracle:thin:@//host:puerto:servicio
```

7. Haga clic en **Conectarse**.

 © 2016-2022 Altova GmbH

4.2.1.10.14 Oracle (ODBC)

Este ejemplo explica cómo conectarse a un servidor de base de datos Oracle desde MapForce. En este ejemplo la base de datos Oracle está en un equipo de la red y la conexión se establece a través de un cliente de base de datos Oracle instalado en el equipo local.

En este ejemplo ofrecemos instrucciones para configurar un DSN ODBC con el asistente para la conexión de base de datos de MapForce. Si ya tiene un DSN o prefiere crear uno desde el administrador de orígenes de datos ODBC de Windows, puede seleccionarlo desde el asistente. Para más información consulte el apartado Conexiones ODBC 180.

Requisitos:

- El cliente de BD Oracle (que incluye el controlador ODBC Oracle) está instalado y configurado en el sistema. Para más información consulte la documentación del software de Oracle.
- El archivo `tnsnames.ora` ubicado en el directorio de inicio de Oracle contiene una entrada que describe los parámetros de conexión de la base de datos:

```
ORCL =
  (DESCRIPTION =
    (ADDRESS_LIST =
      (ADDRESS = (PROTOCOL = TCP)(HOST = server01)(PORT = 1521))
    )
    (CONNECT_DATA =
      (SID = orcl)
      (SERVER = DEDICATED)
    )
  )
```

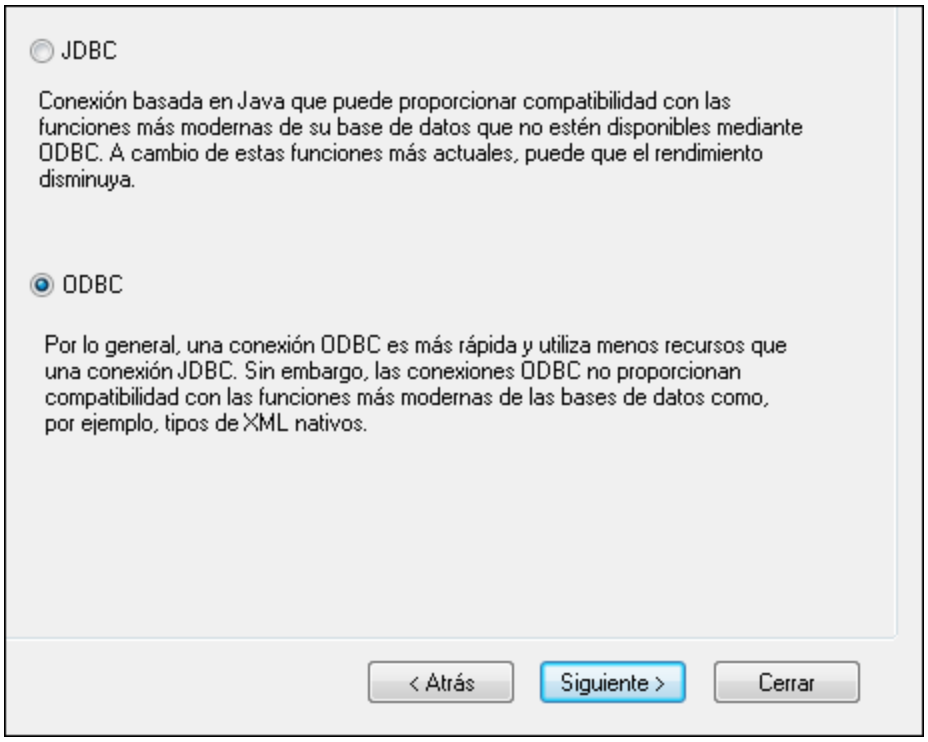
  La ruta de acceso del archivo `tnsnames.ora` depende de la ubicación del directorio de inicio de Oracle. Por ejemplo, en el caso del cliente de base de datos Oracle 11.2.0, la ruta de acceso predeterminada del directorio de inicio podría ser:

```
C:\app\nombreUsuario\product\11.2.0\client_1\network\admin\tnsnames.ora
```

  En el archivo `tnsnames.ora` puede introducir entradas nuevas, pegando los datos de conexión y guardando el archivo o ejecutando el asistente *Net Configuration Assistant* de Oracle (si está disponible). Si quiere que estos valores aparezcan en las listas desplegables durante el proceso de configuración, puede que necesite añadir la ruta de acceso a la carpeta de administrador como una variable de entorno **TNS_ADMIN**.

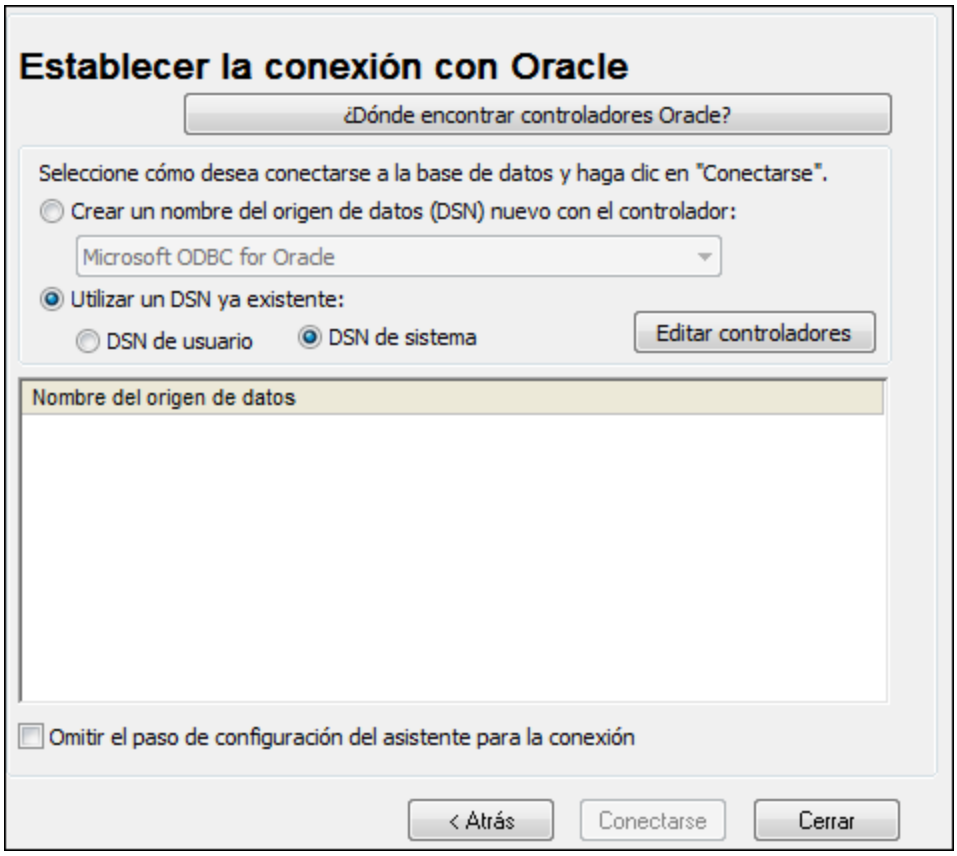
Para conectarse a Oracle por ODBC:

1. Inicie el asistente para la conexión de base de datos 163.
2. Seleccione la opción *Oracle (ODBC / JDBC)* y después haga clic en **Siguiente**.

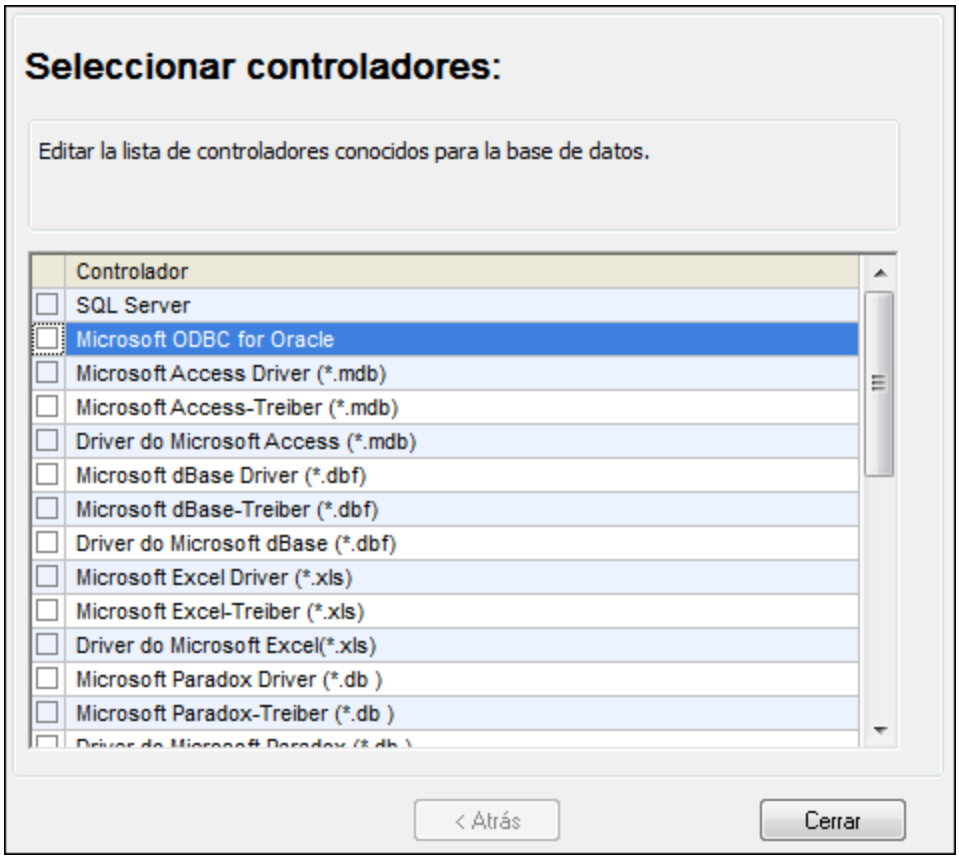


3. Seleccione el botón de opción *ODBC*.

 © 2016-2022 Altova GmbH

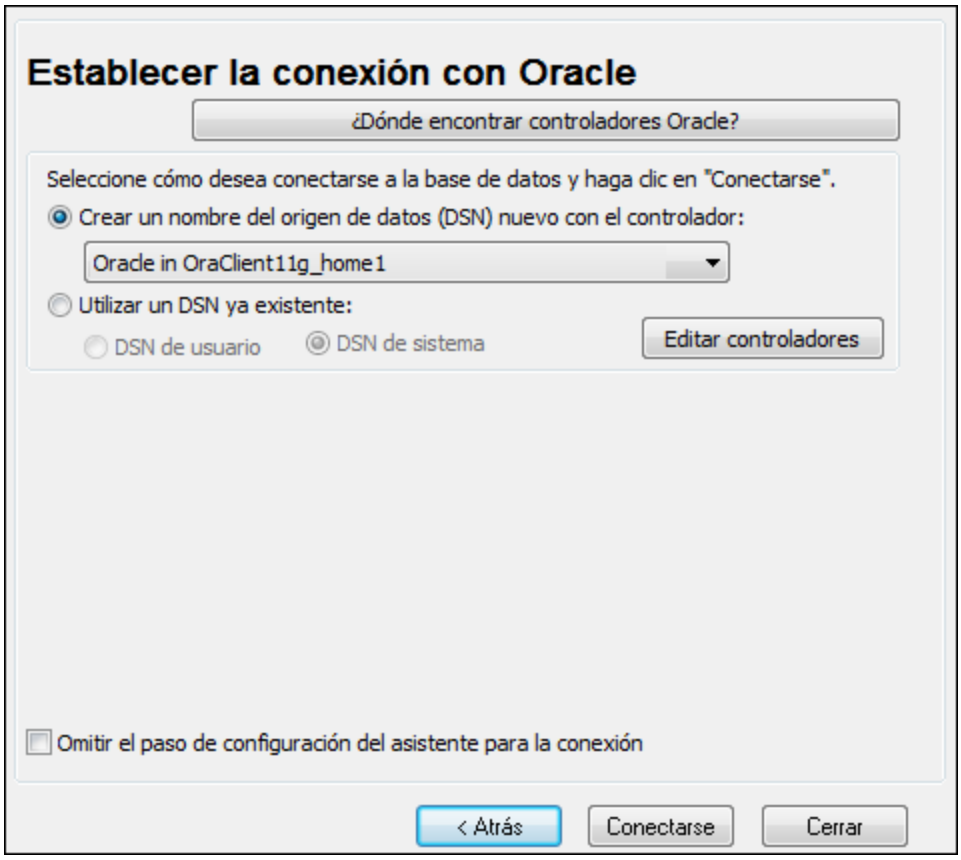


4. Haga clic en **Editar controladores**.



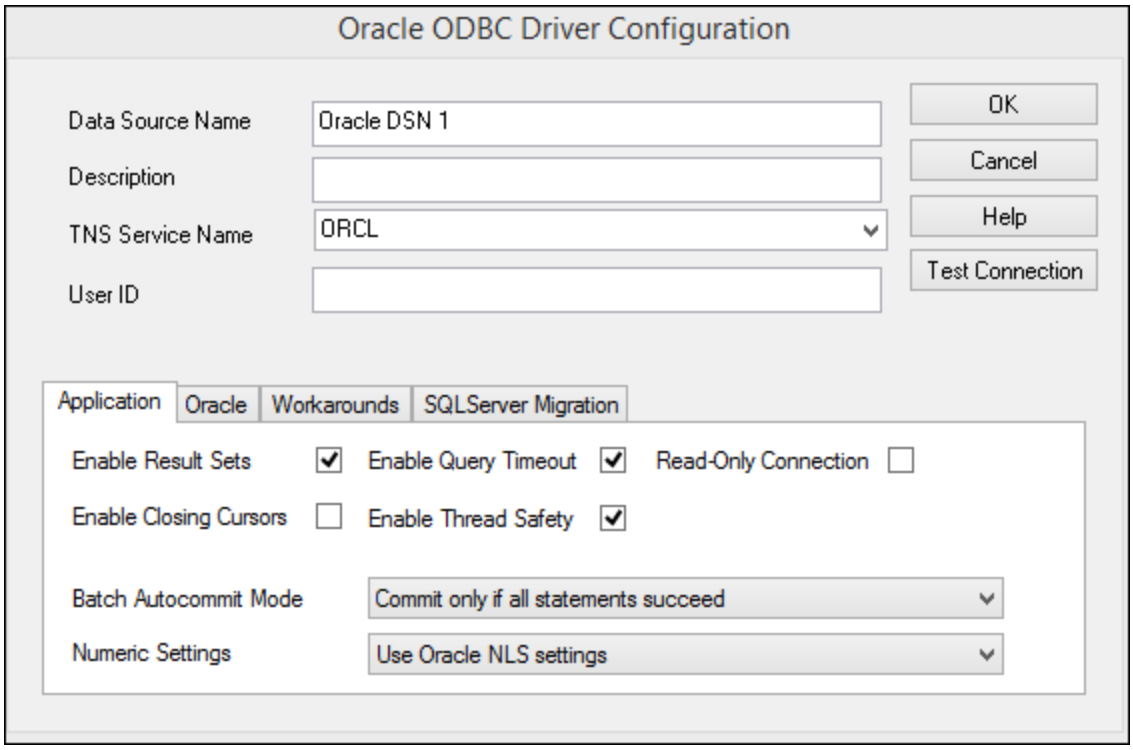
5. Seleccione los controladores Oracle que desea usar (en este ejemplo usamos **Oracle in OraClient11g_home1**). La lista incluye todos los controladores Oracle que están disponibles en el sistema después de instalar el cliente Oracle.
6. Haga clic en **Atrás**.
7. Seleccione la opción *Crear un DSN nuevo con el controlador* y después seleccione el controlador de Oracle elegido en el paso nº 4.

© 2016-2022 Altova GmbH

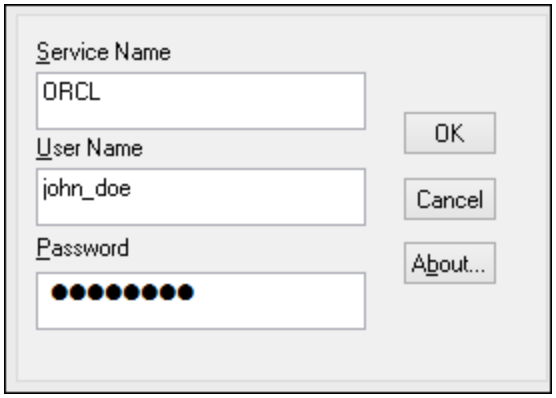


Recomendamos no utilizar el controlador **Microsoft ODBC for Oracle**. Microsoft recomienda utilizar el controlador ODBC que ofrece Oracle (consulte http://msdn.microsoft.com/en-us/library/ms714756%28v=vs.85%29.aspx)

8. Haga clic en **Conectarse**.



9. En el cuadro de texto *Nombre del origen de datos* introduzca un nombre que le ayude a identificar el origen de datos más adelante (en este ejemplo **Oracle DSN 1**).
10. En el cuadro de texto *Nombre del servicio TNS* introduzca el nombre de la conexión tal y como se define en el archivo `tnsnames.ora` (ver Requisitos 227). En este ejemplo el nombre de la conexión es **ORCL**.
11. Haga clic en **Aceptar**.



12. Escriba el nombre de usuario y la contraseña de la base de datos y haga clic en **Aceptar** para terminar.

© 2016-2022 Altova GmbH

4.2.1.10.15 PostgreSQL (ODBC)

Este ejemplo explica cómo conectarse a un servidor de base de datos PostgreSQL desde un equipo Windows con el controlador ODBC. El controlador ODBC PostgreSQL no está disponible en Windows así que deberá descargarlo e instalarlo por separado. En este ejemplo usamos la versión 11.0 del controlador psqlODBC, que se puede descargar del sitio web oficial (consulte el apartado Resumen de controladores de base de datos 165).

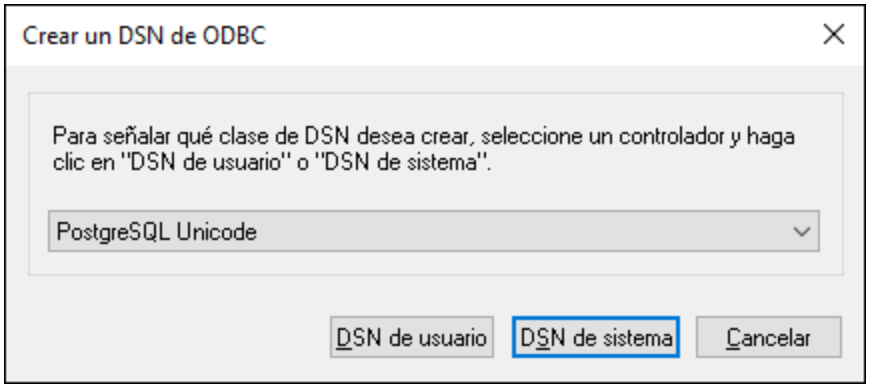
Nota: también se puede conectar a un servidor de base de datos PostgreSQL directamente (es decir, sin el controlador ODBC). Consulte el apartado Conexiones PostgreSQL 188 para más información.

Requisitos:

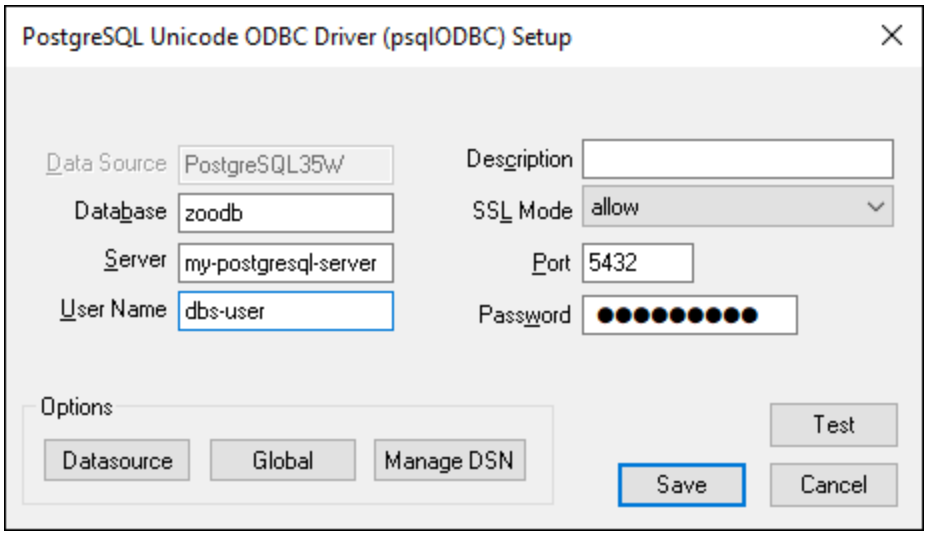
- El controlador *psqlODBC* está instalado en el sistema.
- Disponer de los datos de conexión: servidor, puerto, base de datos, nombre de usuario y contraseña.

Para conectarse a PostgreSQL por ODBC:

1. Inicie el asistente para la conexión de base de datos 163.
2. Haga clic en **Conexiones ODBC**.
3. Seleccione la opción *DNS de usuario*.
4. Haga clic en **Crear DNS nuevo** y seleccione el controlador en la lista desplegable. Si no hay ningún controlador PostgreSQL en la lista, asegúrese de que el controlador PostgreSQL está instalado en su sistema operativo, tal y como se indica más arriba en los requisitos.

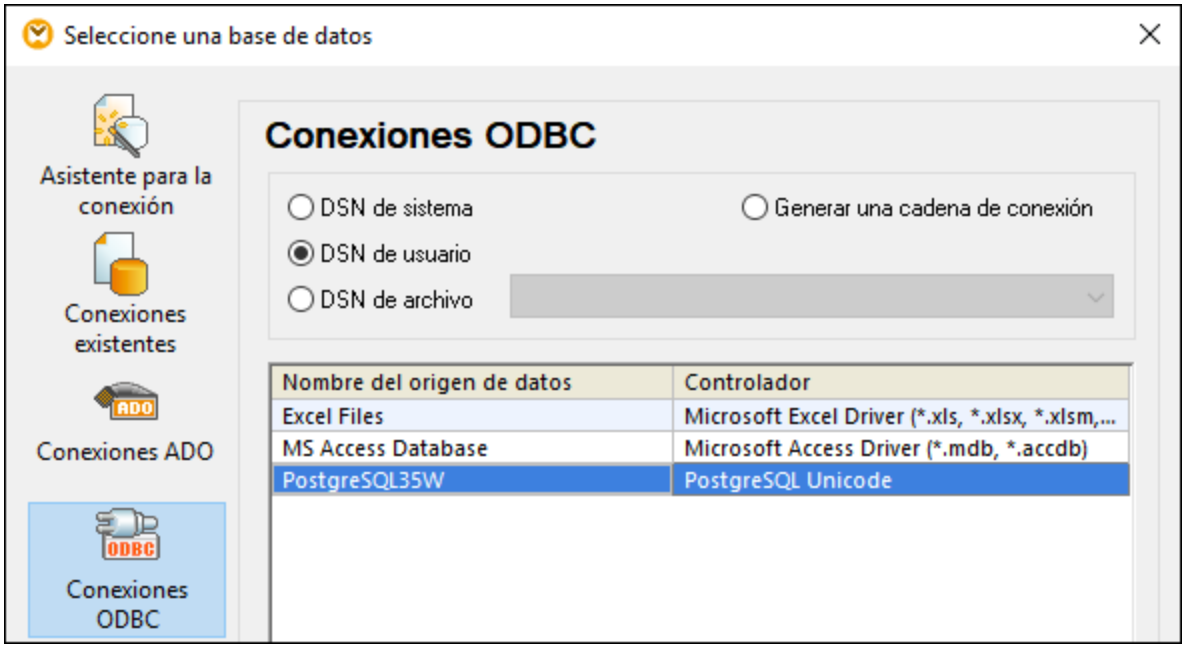


5. Haga clic en **DNS de usuario**.



6. Rellene las credenciales de la conexión de base de datos (las debe proporcionar el propietario de la base de datos) y haga clic en **Aceptar**.

Ahora la conexión está disponible en la lista de conexiones ODBC. Para conectarse a la base de datos puede hacer doble clic en la conexión o seleccionarla y hacer clic en **Conectarse**.



4.2.1.10.16 Progress OpenEdge (JDBC)

En este apartado encontrará instrucciones para conectarse a un servidor de base de datos Progress OpenEdge 11.6 por JDBC.

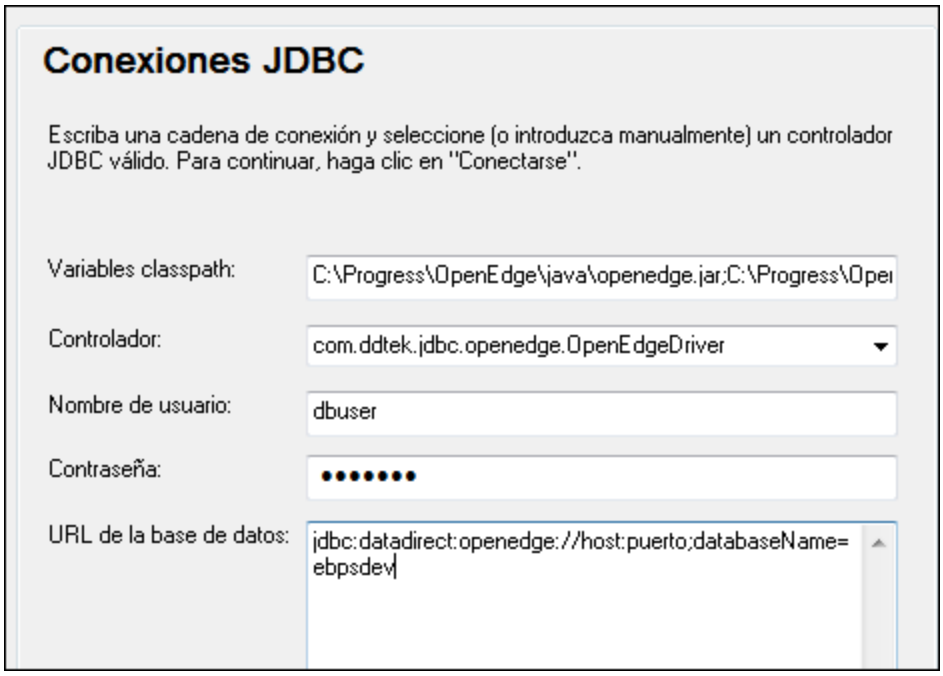
 © 2016-2022 Altova GmbH

Requisitos:

- Debe tener instalado Java Runtime Environment (JRE) o Java Development Kit (JDK). Este último puede ser el JDK de Oracle o uno de código abierto, como Oracle OpenJDK. MapForce identifica la ruta de acceso al equipo virtual Java (JVM) a partir de estas ubicaciones, en este orden: a) la ruta personal de acceso al JVM que puede indicar en en las **Opciones** (véase el apartado Java 1052); b) la ruta de acceso al JVM que se encuentra en el registro; c) la variable de entorno `JAVA_HOME`.
- Asegúrese de que la plataforma de MapForce (32 o 64 bits) es la misma que la de JRE/JDK.
- La variable de entorno `PATH` del sistema incluye la ruta al directorio `bin` del directorio de instalación de JRE o JDK (p. ej. `C:\Archivos de programa (x86)\Java\jre1.8.0_51\bin`).
- El controlador JDBC Progress OpenEdge está disponible en el sistema operativo. En este ejemplo la conexión por JDBC se consigue con los archivos de controlador **openedge.jar** y **pool.jar** disponibles en `C:\Progress\OpenEdge\java` y que se instalan con OpenEdge SDK.
- Disponer de los datos de conexión: host, puerto, nombre de la base de datos, nombre de usuario y contraseña.

Para conectarse a Progress OpenEdge por JDBC:

1. Inicie el asistente para la conexión de base de datos 163.
2. Haga clic en **Conexiones JDBC**.
3. En el campo *Variables classpath* debe introducir la ruta de acceso del archivo .jar que ofrece conectividad con la base de datos. Si fuera necesario, también puede introducir una lista de rutas de archivo .jar separadas por caracteres de punto y coma. En este ejemplo, los archivos .jar están ubicados en esta ruta de acceso: **`C:\Progress\OpenEdge\java\openedge.jar;C:\Progress\OpenEdge\java\pool.jar;`**. Tenga en cuenta que este campo puede dejarse en blanco si añadió la ruta de acceso de los archivos .jar a la variable de entorno CLASSPATH del sistema operativo (*véase Configurar la variable CLASSPATH* 186).
4. En el campo *Controlador* seleccione **com.ddtek.jdbc.openedge.OpenEdgeDriver**. Recuerde que esta entrada solo estará disponible si se encuentra una ruta de archivo .jar válida en el campo Variables *classpath* o en la variable de entorno CLASSPATH del sistema operativo.



5. Introduzca el nombre de usuario y la contraseña de la base de datos.
6. Introduzca la cadena de conexión para el servidor de BD en el cuadro de texto *URL de la base de datos* (reemplace lo valores resaltados con los de su base de datos).

```
jdbc:datadirect:openedge://host:puerto;databaseName=nombre_BD
```

7. Haga clic en **Conectarse**.

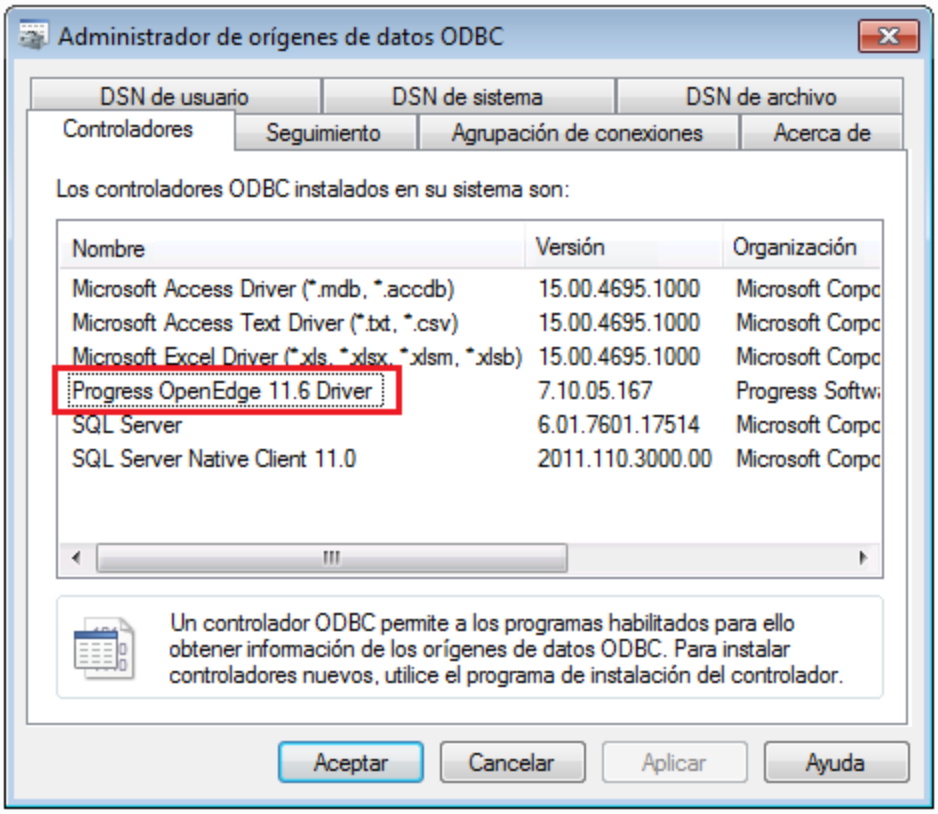
4.2.1.10.17 Progress OpenEdge (ODBC)

En este apartado encontrará instrucciones para conectarse a un servidor de base de datos Progress OpenEdge por medio del controlador ODBC Progress OpenEdge 11.6.

Requisitos:

- El controlador *ODBC Connector for Progress OpenEdge* está instalado en el sistema operativo. Este controlador se puede descargar del sitio web del proveedor (consulte la lista del apartado Resumen de controladores de base de datos 165). Si trabaja con la versión de 32 bits de MapForce, descargue el controlador de 32 bits. Por el contrario, si usa la versión de 64 bits, descargue el controlador de 64 bits. Una vez finalizada la instalación, compruebe que el controlador ODBC está disponible en el equipo (*véase* *Ver los controladores ODBC disponibles* 182).

 © 2016-2022 Altova GmbH

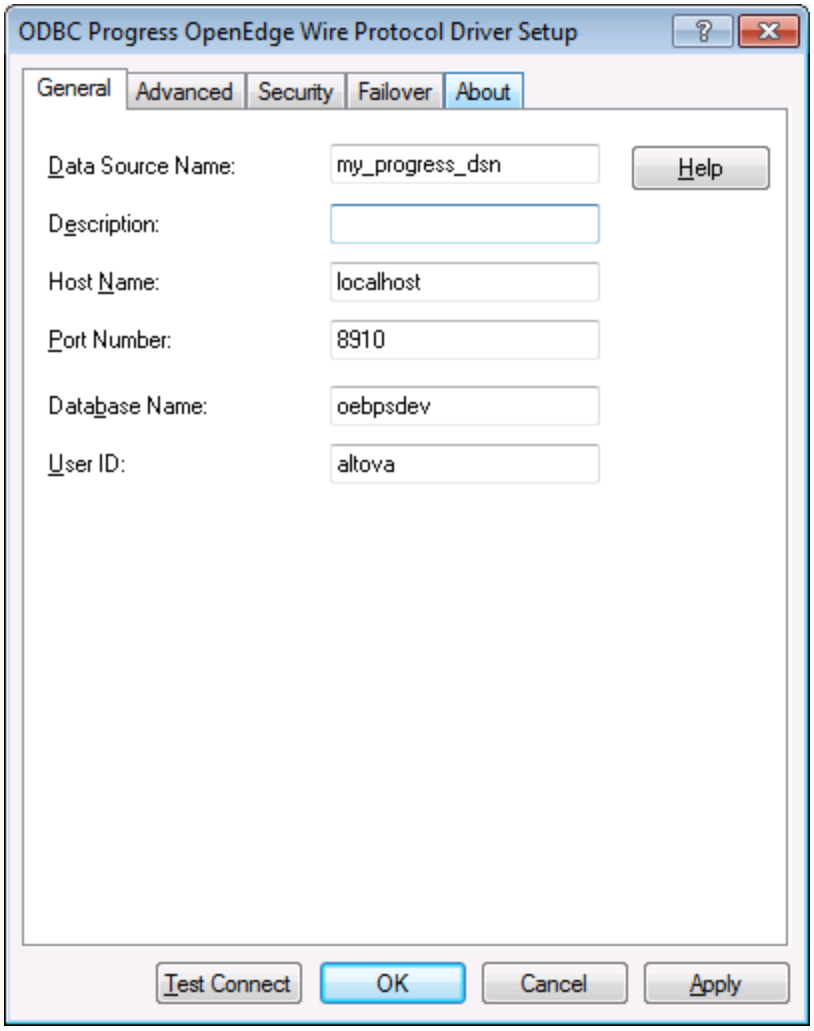


- Disponer de los datos de conexión: nombre de host, número de puerto, nombre de la base de datos, id. de usuario y contraseña.

Para conectarse a Progress OpenEdge por ODBC:

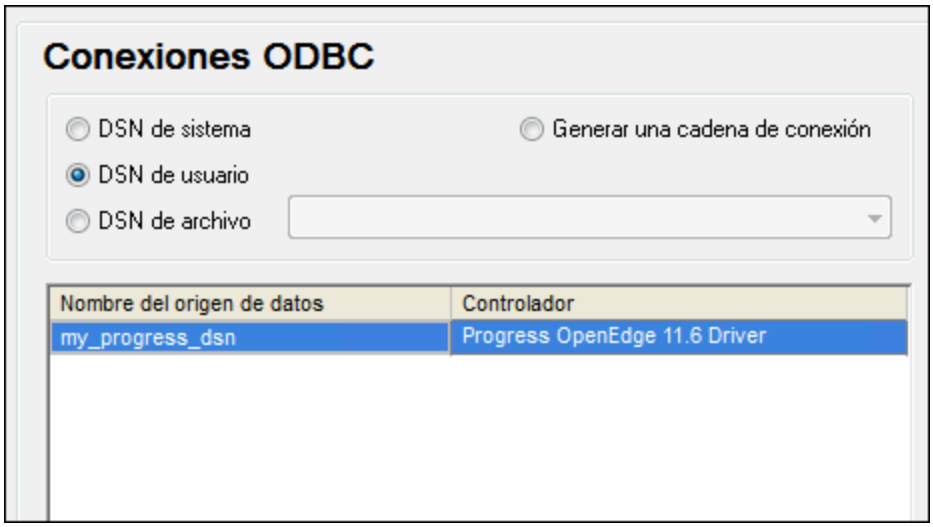
1. Inicie el asistente para la conexión de base de datos 163.
2. Haga clic en el botón Conexiones ODBC.
3. Haga clic en *DSN de usuario* (o en *DSN de sistema* o *DSN de archivo*).
4. Ahora haga clic en el icono **Agregar** .
5. Seleccione el controlador **Progress OpenEdge** de la lista y haga clic en el botón **DSN de usuario** (o **DSN de sistema** según corresponda).

6. Rellene las credenciales para la conexión con la base de datos (base de datos, servidor, puerto, nombre de usuario y contraseña) y haga clic en **Aceptar**. Para probar la conexión antes de guardar los datos introducidos haga clic en el botón **Probar conexión**.



7. Haga clic en **Aceptar**. El origen de datos nuevo aparece ahora en la lista de orígenes de datos ODBC.

© 2016-2022 Altova GmbH



8. Para terminar haga clic en **Conectarse**.

4.2.1.10.18 Sybase (JDBC)

Este ejemplo explica cómo conectarse a un servidor de base de datos Sybase por JDBC.

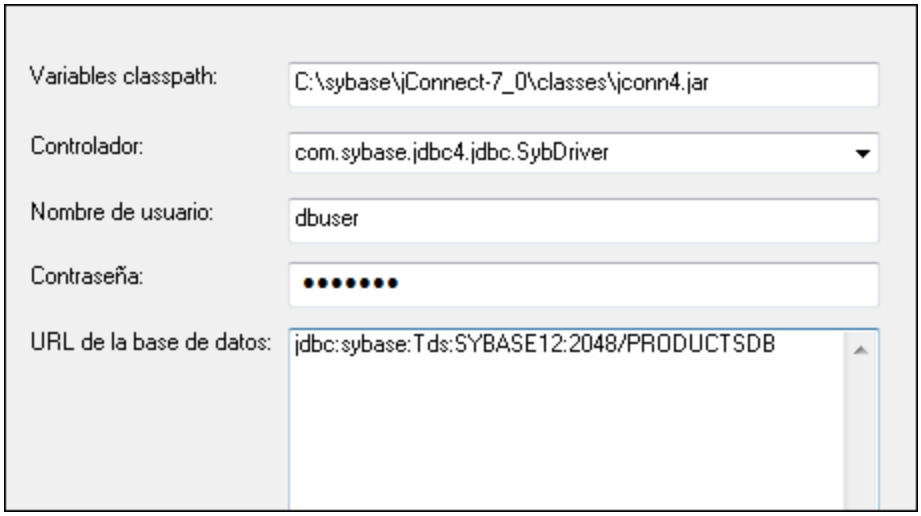
Requisitos:

- Debe tener instalado Java Runtime Environment (JRE) o Java Development Kit (JDK). Este último puede ser el JDK de Oracle o uno de código abierto, como Oracle OpenJDK. MapForce identifica la ruta de acceso al equipo virtual Java (JVM) a partir de estas ubicaciones, en este orden: a) la ruta personal de acceso al JVM que puede indicar en en las **Opciones** (véase el apartado Java 1052); b) la ruta de acceso al JVM que se encuentra en el registro; c) la variable de entorno `JAVA_HOME`.
- Asegúrese de que la plataforma de MapForce (32 o 64 bits) es la misma que la de JRE/JDK.
- El componente Sybase *jConnect* está instalado en el sistema (en este ejemplo usamos *jConnect 7.0* que se instala con el cliente *Sybase Adaptive Server Enterprise PC Client*. Consulte la documentación de Sybase para obtener más información.
- Disponer de los datos de conexión: host, puerto, nombre de la base de datos, nombre de usuario y contraseña.

Para conectarse a Sybase por JDBC:

1. Inicie el asistente para la conexión de base de datos 163.
2. Haga clic en **Conexiones JDBC**.
3. En el campo *Variables classpath* debe introducir la ruta de acceso del archivo .jar que ofrece conectividad con la base de datos. Si fuera necesario, también puede introducir una lista de rutas de archivo .jar separadas por caracteres de punto y coma. En este ejemplo, el archivo .jar está ubicado en esta ruta de acceso: `C:\sybase\jConnect-7_0\classes\jconn4.jar`. Tenga en cuenta que este campo puede dejarse en blanco si añadió la ruta de acceso de los archivos .jar a la variable de entorno CLASSPATH del sistema operativo (*véase Configurar la variable CLASSPATH* 186).

4. En el campo *Controlador* seleccione **com.sybase.jdbc4.jdbc.SybDriver**. Recuerde que esta entrada solo estará disponible si se encuentra una ruta de archivo .jar válida en el campo Variables *classpath* o en la variable de entorno CLASSPATH del sistema operativo.



5. Introduzca el nombre de usuario y la contraseña de la base de datos.
6. Introduzca la cadena de conexión para el servidor de BD en el cuadro de texto *URL de la base de datos* (reemplace lo valores resaltados con los de su base de datos).

```
jdbc:sybase:Tds:nombreHost:puerto/nombreBD
```

7. Haga clic en **Conectarse**.

4.2.1.10.19 Teradata (JDBC)

Este ejemplo explica cómo conectarse a un servidor de base de datos Teradata por JDBC.

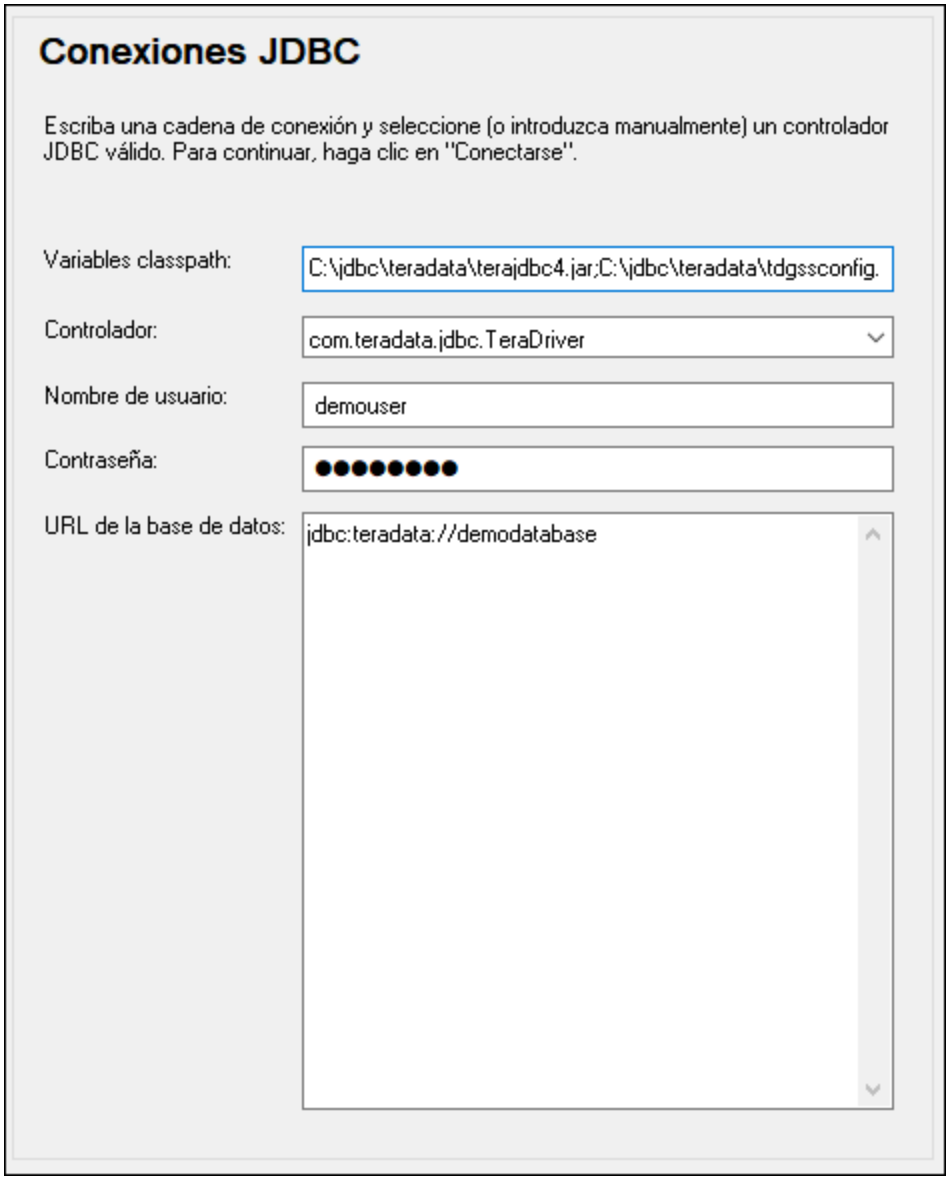
Requisitos:

- Debe tener instalado Java Runtime Environment (JRE) o Java Development Kit (JDK). Este último puede ser el JDK de Oracle o uno de código abierto, como Oracle OpenJDK. MapForce identifica la ruta de acceso al equipo virtual Java (JVM) a partir de estas ubicaciones, en este orden: a) la ruta personal de acceso al JVM que puede indicar en en las **Opciones** (véase el apartado Java[1052]); b) la ruta de acceso al JVM que se encuentra en el registro; c) la variable de entorno `JAVA_HOME`.
- Asegúrese de que la plataforma de MapForce (32 o 64 bits) es la misma que la de JRE/JDK.
- Debe contar con el controlador JDBC (archivos .jar que permiten conectarse a la base de datos) en el sistema operativo. Para este ejemplo usamos el controlador JDBC de Teradata 16.20.00.02 (https://downloads.teradata.com/download/connectivity/jdbc-driver).
- Disponer de los datos de conexión: host, base de datos, puerto, nombre de usuario y contraseña.

Para conectarse a Teradata por JDBC:

1. Inicie el asistente para la conexión de base de datos[163].

2. Haga clic en **Conexiones JDBC**.
3. Junto a *Variables Classpath* introduzca la ruta de acceso del archivo .jar que permite conectarse a la base de datos. Si es necesario, también puede introducir una lista de rutas de archivo .jar separadas por punto y comas. Para este ejemplo los archivos .jar están ubicados en esta ruta de acceso: **C:\jdbc\teradata**. Recuerde que puede dejar vacío el cuadro *Variables Classpath* si añadió las rutas de archivo .jar a la variable de entorno CLASSPATH del sistema operativo (véase Configurar la variable CLASSPATH 186).
4. En el cuadro *Controlador* seleccione **com.teradata.jdbc.TeraDriver**. Recuerde que esta entrada está disponible si en el cuadro *Variables Classpath* o en la variable de entorno CLASSPATH del sistema operativo se encuentra una ruta de archivo .jar válida (ver paso anterior).



5. Introduzca el nombre de usuario y la contraseña de la base de datos en las casillas correspondientes.
6. Introduzca la cadena de conexión en el cuadro de texto *URL de la base de datos* (reemplazando el valor resaltado con el valor correspondiente).

```
jdbc:teradata://nombreServidorBaseDatos
```

7. Por último, haga clic en **Conectarse**.

4.2.1.10.20 Teradata (ODBC)

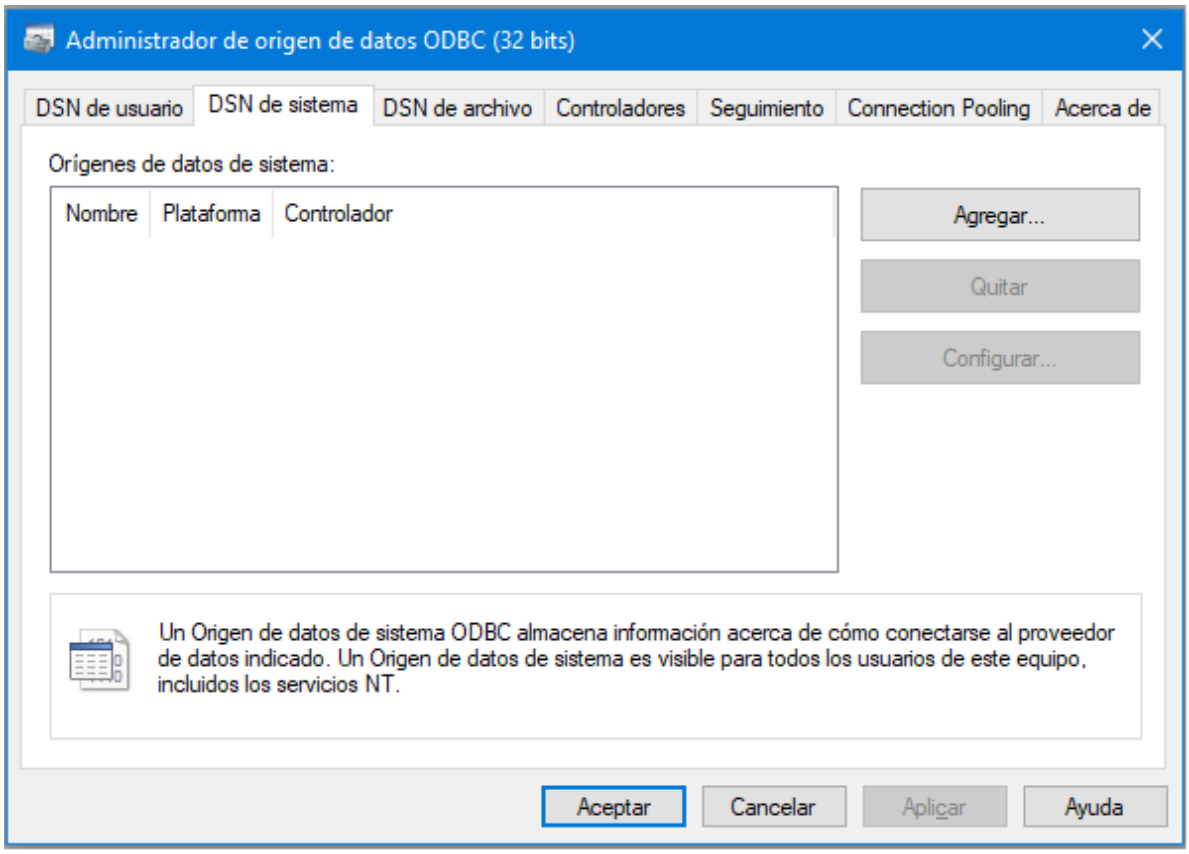
Este ejemplo explica cómo conectarse a un servidor de base de datos Teradata por ODBC.

Requisitos:

- Debe tener instalado el controlador ODBC de Teradata (https://downloads.teradata.com/download/connectivity/odbc-driver/windows). Para este ejemplo se utiliza el controlador ODBC de Teradata para Windows versión 16.20.00.
- Disponer de los datos de conexión: host, nombre de usuario y contraseña.

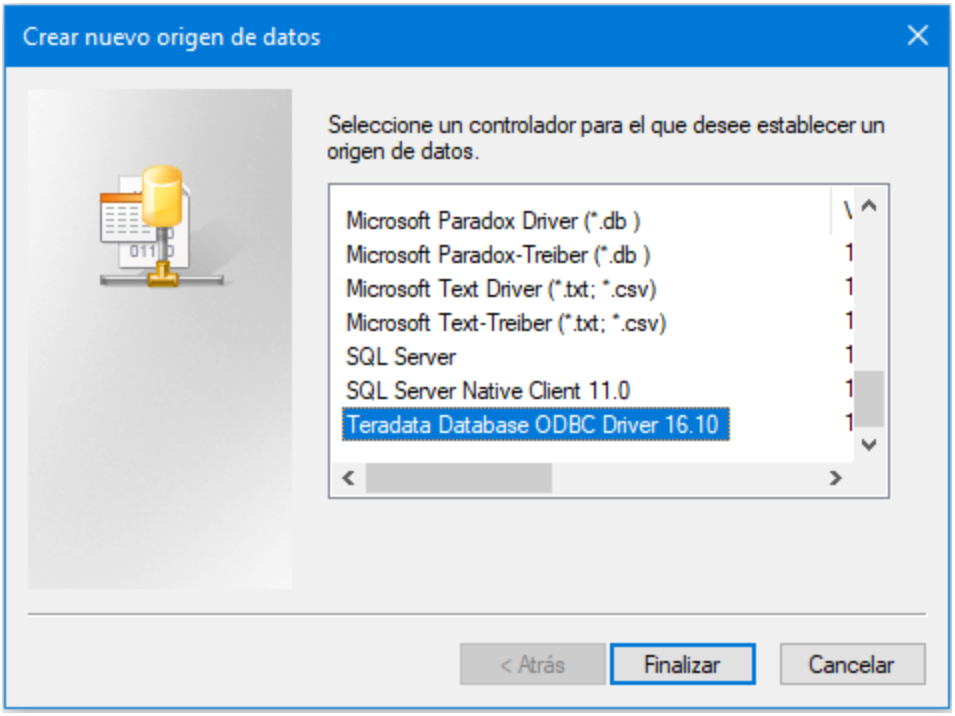
Para conectarse a Teradata por ODBC:

1. Pulse la tecla Windows, teclee "ODBC" y seleccione **Configurar orígenes de datos ODBC (32 bits)** en la lista de sugerencias. Si tiene un controlador ODBC de 64 bits, seleccione la opción **Configurar orígenes de datos ODBC (64 bits)** y utilice MapForce de 64 bits a partir de ese momento.

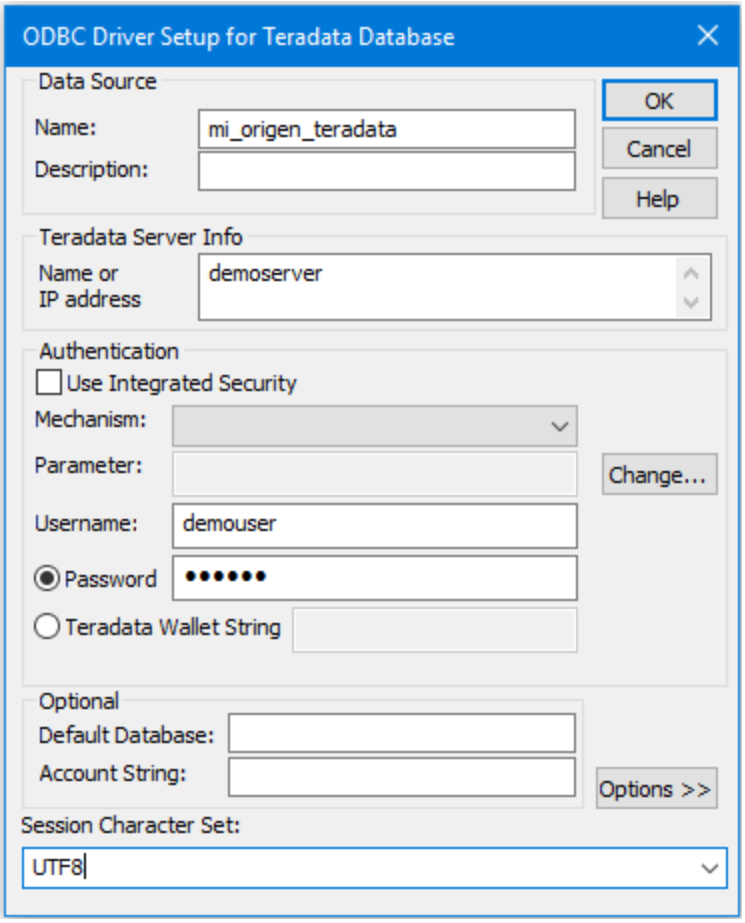


 © 2016-2022 Altova GmbH

2. Haga clic en la pestaña *DSN de sistema* y después haga clic en **Agregar**.

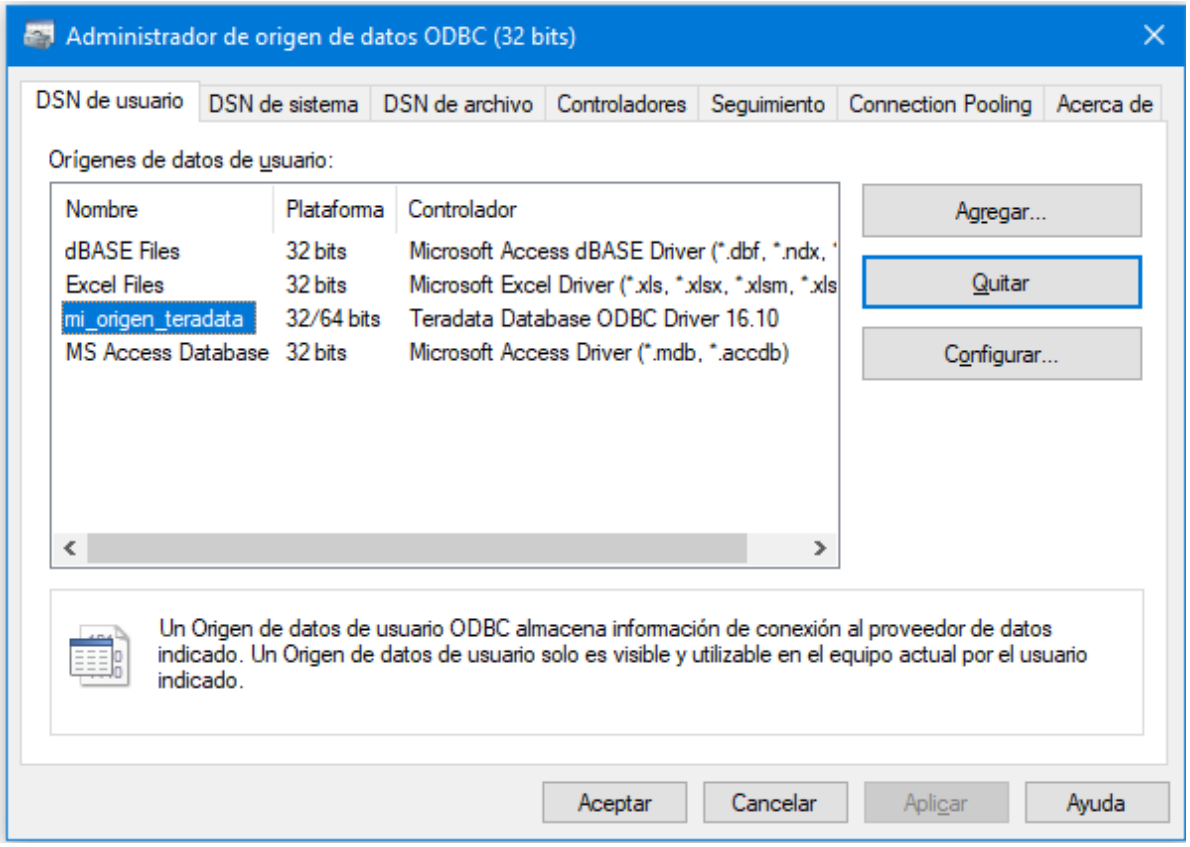


3. Seleccione **Teradata Database ODBC Driver** y después haga clic en **Finalizar**.

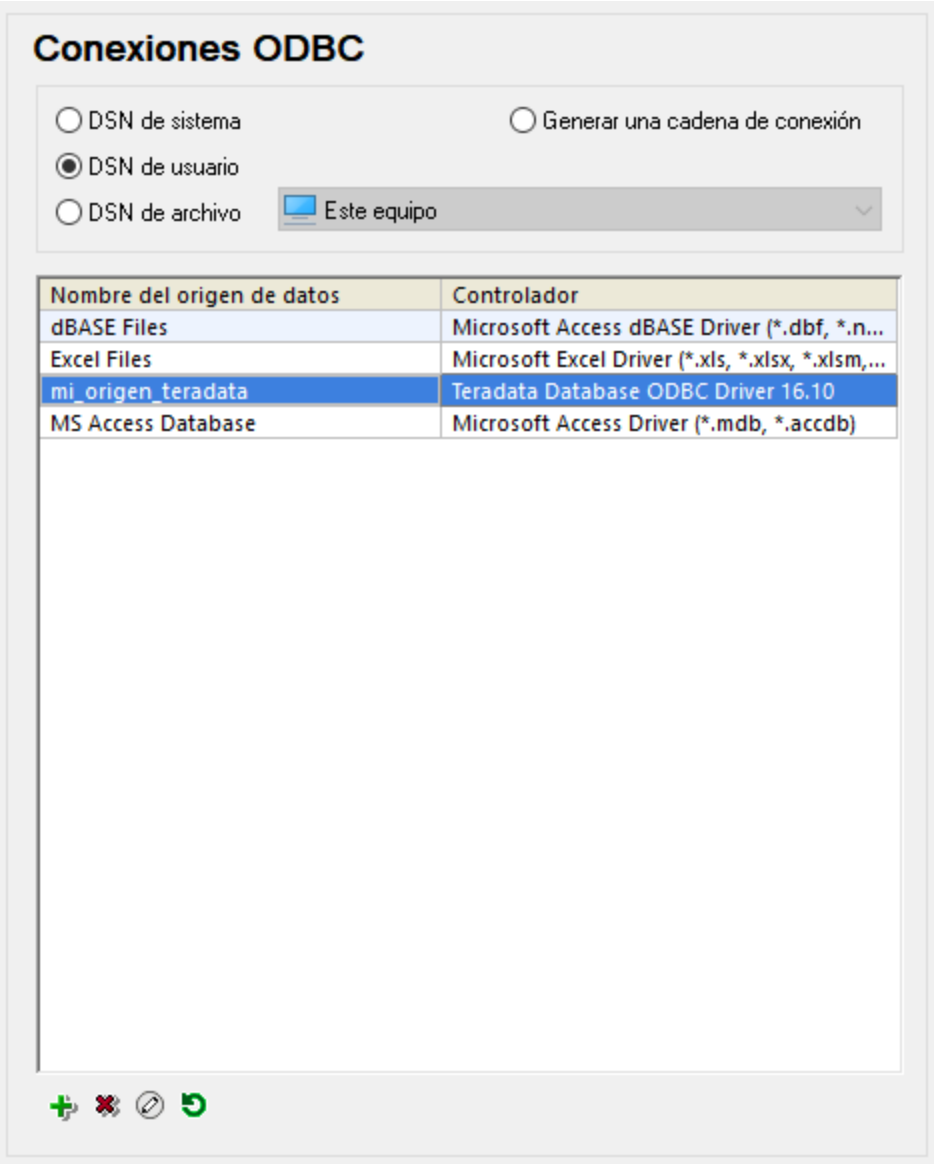


4. Introduzca el nombre y, si quiere, una descripción que le ayude a identificar este origen de datos ODBC más adelante. Además, deberá introducir las credenciales de la conexión de base de datos (servidor de BD, usuario y contraseña) y, si quiere, seleccione una base de datos.
5. Haga clic en **Aceptar**. El origen de datos aparecerá en la lista.

 © 2016-2022 Altova GmbH



6. Ejecute MapForce e inicie el asistente para la conexión de base de datos 163.
7. Haga clic en **Conexiones ODBC**.



8. Haga clic en *DSN de sistema,* seleccione el origen de datos creado en los pasos anteriores y después haga clic en el botón **Conectarse**.

Nota: si recibe el mensaje de error "Controlador devuelto no válido (o error en la devolución) SQL_DRIVER_ODBC_VER: 03.80", compruebe que la ruta de acceso del cliente ODBC (p. ej. **C:\Archivos de programa\Teradata\Client\16.10\bin**) existe en su variable de entorno PATH del sistema. Si falta esta ruta de acceso, entonces deberá añadirla a mano.

4.2.2 Introducción a las asignaciones de datos de BD

Esta sección explica a grandes rasgos cómo trabajar con bases de datos en MapForce y cómo trabajar con una base de datos después de establecer la conexión con la base de datos correctamente (*véase* *Conectarse a una base de datos* 161). Más concretamente, encontrará información sobre cómo seleccionar los objetos de

la BD que deben aparecer en la asignación, cómo gestionar las relaciones de base de datos y cómo configurar la base de datos para el proceso de asignación. Además encontrará varios ejemplos prácticos.

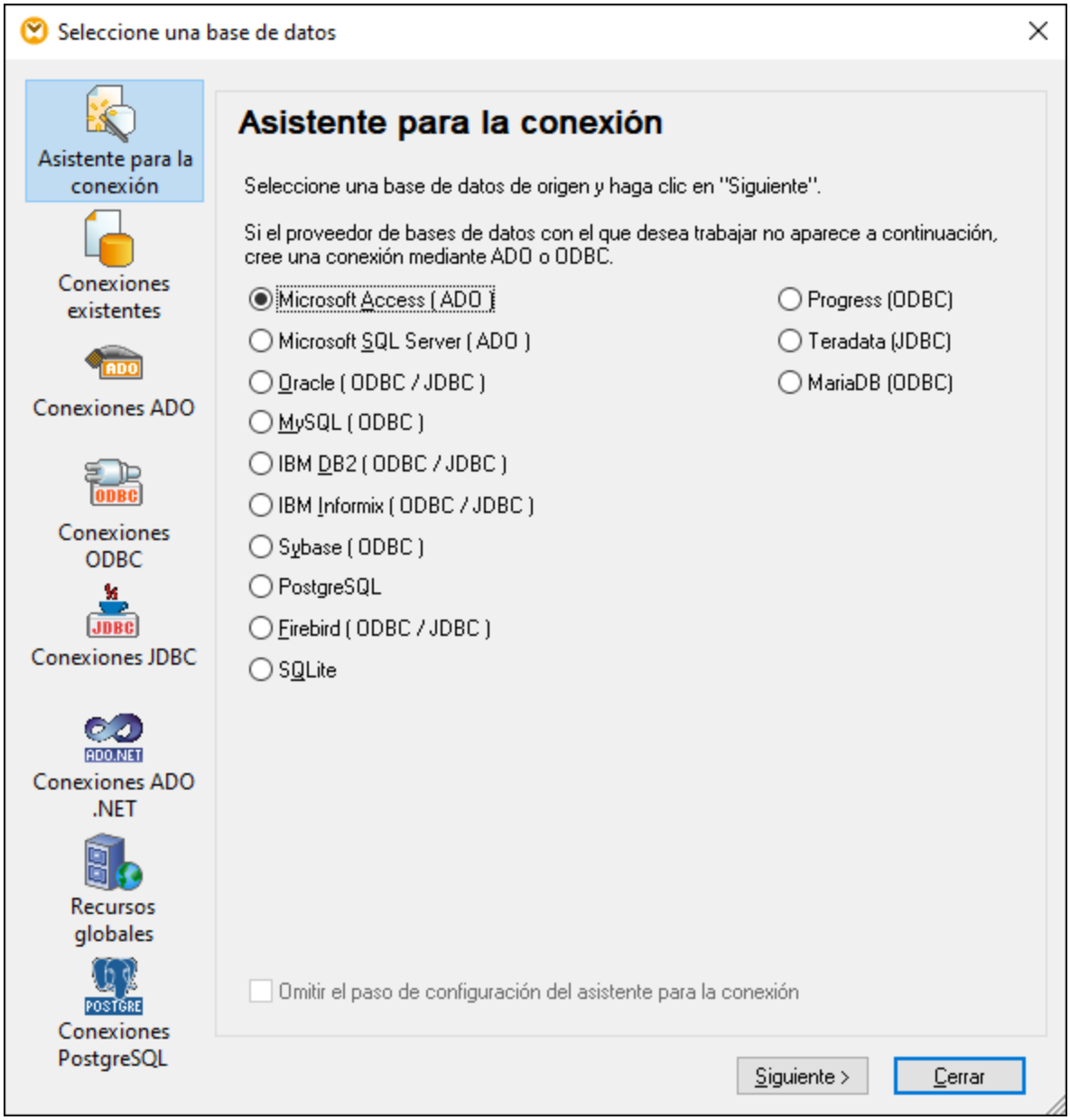
4.2.2.1 Agregar bases de datos a la asignación

Antes de agregar una base de datos a la asignación, asegúrese de que selecciona un lenguaje de transformación compatible con las asignaciones de datos de base de datos: el motor de transformación integrado o C++, C# o Java (*véase* *Seleccionar un lenguaje de transformación* 24). Recuerde que, si tiene pensado implementar la asignación en FlowForce Server o ejecutarla con MapForce Server o si tiene pensado usar características como la transferencia masiva o los procedimientos almacenados, entonces deberá seleccionar el motor de transformación integrado (BUILT-IN).

Tras seleccionar el lenguaje de transformación adecuado podrá agregar la base de datos a la asignación. Esto puede hacerse de dos maneras:

- Con el comando **Base de datos** del menú **Insertar**.
- Con el botón Insertar base de datos de la barra de herramientas.

Esto abre el asistente para la conexión a bases de datos, que le guiará en el proceso de conexión.



Asistente para la conexión a bases de datos

Nota: las bases de datos también se pueden agregar a la asignación en forma de variable (*véase* *Usar variables* 492). Cuando se añade una estructura de base de datos como variable, aparece el mismo asistente de conexión.

Para aprender a configurar una conexión con una de las bases de datos compatibles con MapForce debe consultar la sección Conectarse a una base de datos 161.

Una vez establecida la conexión con la base de datos, deberá seleccionar qué objetos de la base de datos deben aparecer en la asignación (*véase* *Agregar, editar y eliminar objetos de base de datos* 252).

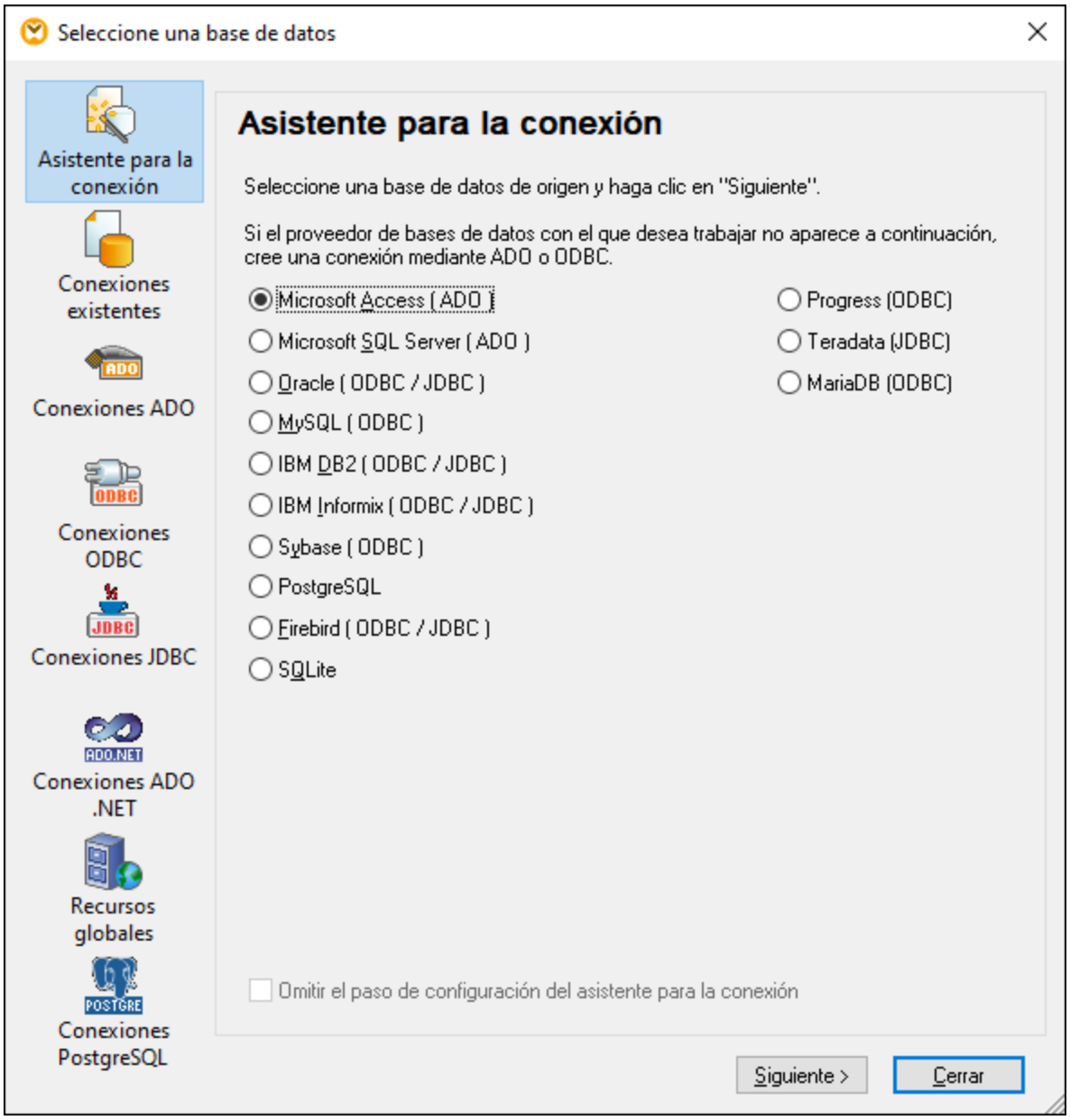
 © 2016-2022 Altova GmbH

4.2.2.2 Ejemplo: agregar la base de datos "altova.mdb" a la asignación

Este ejemplo explica cómo agregar una base de datos Microsoft Access a la asignación. La base de datos se llama **altova.mdb** y está en la carpeta **<Documentos>\Altova\MapForce2023\MapForceExamples**. La base de datos **altova.mdb** permite trabajar con varias acciones y conceptos descritos en esta documentación.

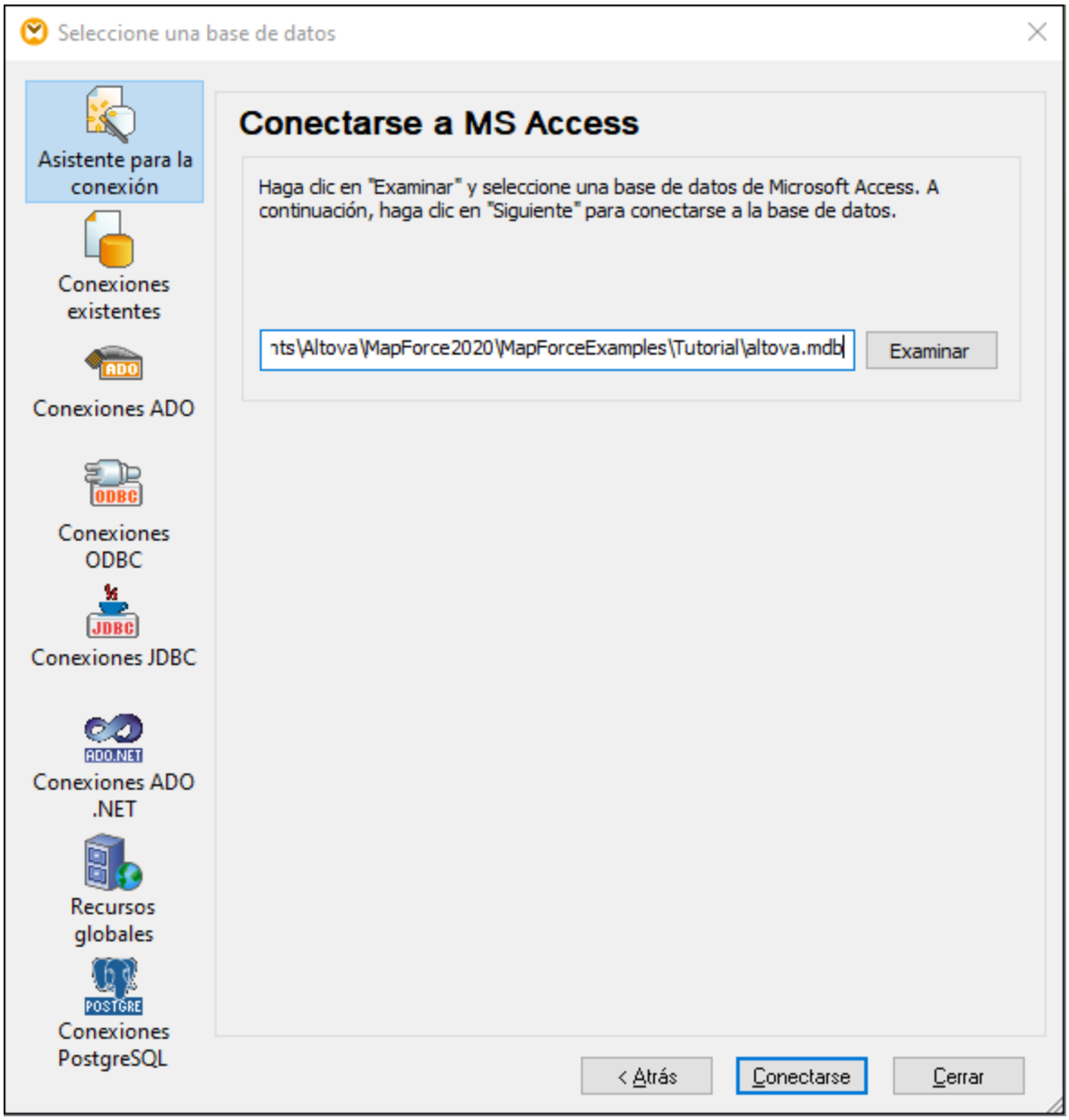
Para agregar la base de datos altova.mdb a la asignación:

1. En el menú **Insertar** haga clic en el comando **Base de datos** (o haga clic en el botón **Insertar base de datos** de la barra de herramientas).



2. En el asistente para la conexión elija el botón de opción *Microsoft Access (ADO)* y haga clic en **Siguiente**.

 © 2016-2022 Altova GmbH



3. Navegue hasta el archivo **altova.mdb** de la carpeta **<Documentos>\Altova\MapForce2023\MapForceExamples** y haga clic en **Conectarse**.
4. Cuando la aplicación le pida que seleccione objetos de la base de datos, marque la casilla de `Tablas de usuario`.

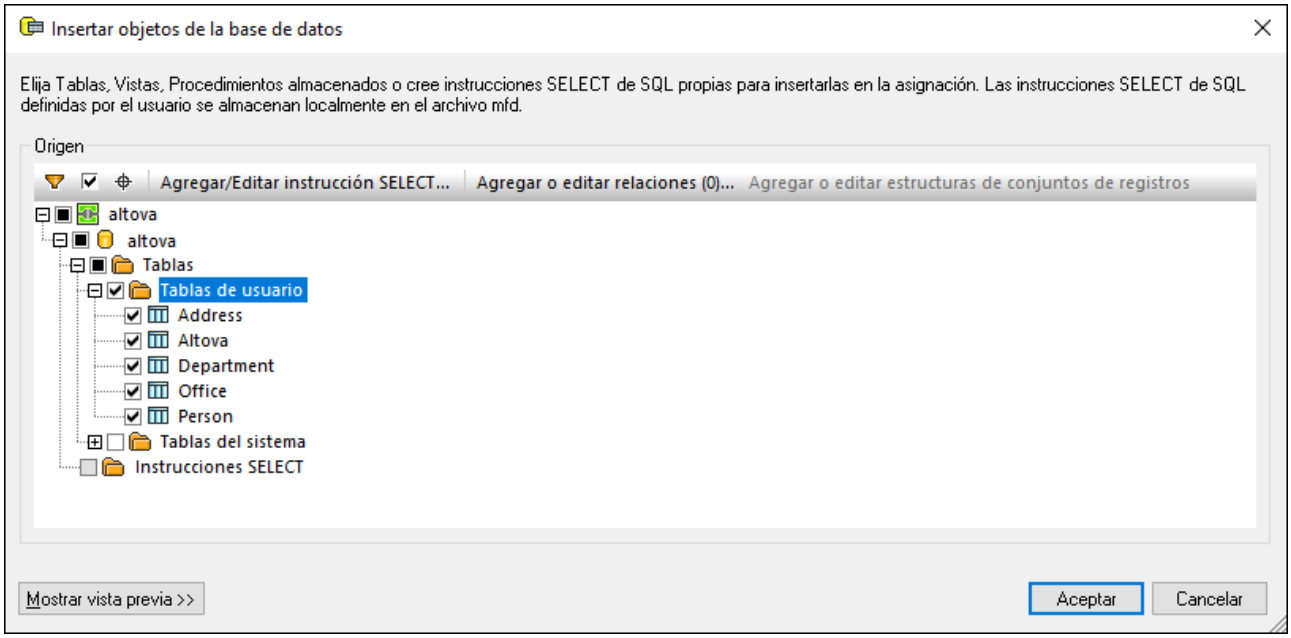
4.2.2.3 Agregar, editar y quitar objetos de BD

Algunas bases de datos pueden tener gran número de objetos (esquemas, tablas, vistas, etc.). En este apartado explicamos cómo incluir en la asignación solamente los objetos que necesitamos. Para ello utilizaremos una base de datos Access, pero puede seguir las mismas instrucciones con cualquier tipo de base de datos.

1. En el menú **Insertar** haga clic en el comando **Base de datos**.
2. En el asistente para la conexión seleccione el botón de opción *Microsoft Access (ADO)*.
3. Haga clic en **Siguiente** y navegue hasta el archivo **altova.mdb** de la carpeta **<Documentos>\Altova\MapForce2023\MapForceExamples**.

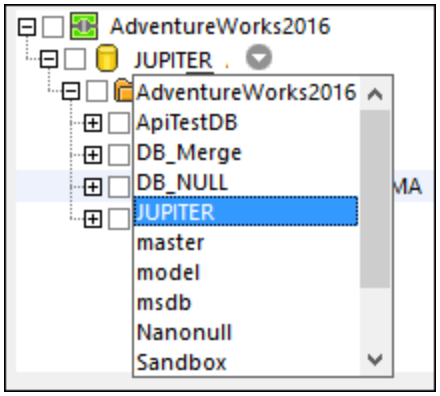
Aparece un cuadro de diálogo donde puede seleccionar qué objetos de la base de datos se incluyen en la asignación.

 © 2016-2022 Altova GmbH



El nodo superior de la estructura indica la conexión de BD. El resto de la estructura varía en función del tipo de base de datos. Por ejemplo, las bases de datos Oracle y IBM DB2 tienen un nodo de esquema bajo el nodo de conexión, mientras que los otros tipos de BD tienen un nodo de catálogo . El texto en negrita indica que se trata del catálogo o esquema predeterminado (BD), según corresponda.

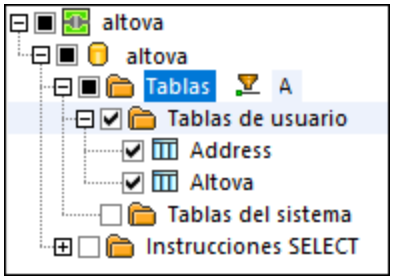
Si la cuenta del usuario de la BD tiene acceso a varias bases de datos o varios esquemas en el servidor, puede cambiar al que necesite haciendo clic en el icono . El ejemplo siguiente muestra el servidor SQL Server; la estructura puede ser distinta en otros tipos de BD.



Para incluir un objeto de la base de datos (p. ej. una tabla) en la asignación basta con marcar su casilla. Por ejemplo, vamos a marcar la casilla de `Tablas de usuario`.

El botón **Localizador de objetos** del cuadro de diálogo permite buscar elementos concretos de la base de datos. Seleccione un objeto (o escriba su nombre) en el cuadro combinado que aparece en la parte inferior del cuadro de diálogo.

El botón **Filtrar** sirve para filtrar objetos por nombre. Cuando se hace clic en el botón **Filtrar**, aparece un icono en forma de embudo junto a los objetos que se pueden filtrar (p. ej. `Tablas`).



Haga clic en el icono en forma de embudo para elegir si el nombre del objeto debe empezar o terminar con la cadena de búsqueda o ser igual o contener la cadena de búsqueda. Ahora puede introducir la cadena de búsqueda junto al filtro (p. ej. "A"):

El botón **Mostrar objetos seleccionados** sirve para ver solamente los elementos cuya casilla esté marcada.

El botón **Agregar/Editar instrucción SELECT** sirve para agregar o editar instrucciones SELECT personales para la base de datos actual. Los datos que devuelvan dichas instrucciones podrán utilizarse como componentes de origen en la asignación (*véase* *Instrucciones SQL SELECT como tablas virtuales* 359).

El botón **Agregar o editar relaciones** sirve para definir relaciones de clave principal o foránea entre los campos de la base de datos (*véase* *Definir relaciones locales* 267).

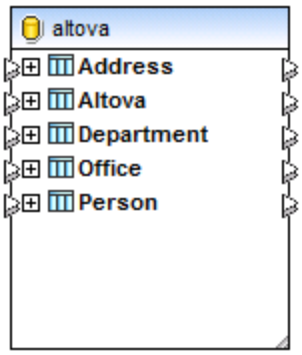
El botón **Agregar o editar estructuras de conjuntos de registros** solamente se habilita si la base de datos admite procedimientos almacenados y si se seleccionó un procedimiento almacenado en el cuadro de diálogo (*véase* *Procedimientos almacenados* 395).

El botón **Mostrar vista previa** sirve para obtener una vista previa de los datos de la tabla o vista que está seleccionada en ese momento. Recuerde que con el Explorador de bases de datos podrá examinar y consultar bases de datos fuera del proceso de asignación (*véase* *Examinar y consultar bases de datos* 380).

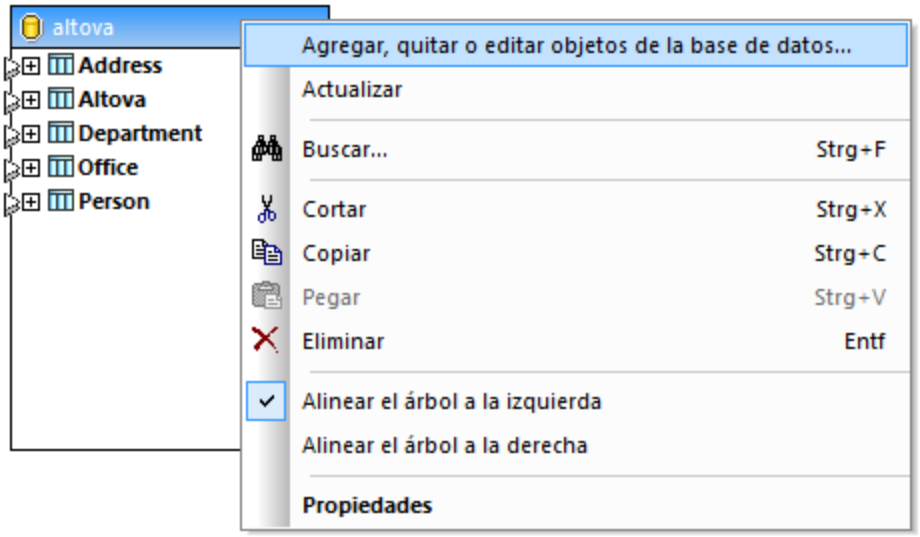
La casilla *Quitar esquema del nombre de la tabla* se habilita solo para bases de datos compatibles con objetos con nombre relativo. Esto es útil si piensa cambiar la BD más tarde. Para más información consulte Cambiar de BD y esquema 255.

Cuando esté todo listo para agregar los objetos de la base de datos a la asignación, haga clic en **Aceptar**. En el componente de base de datos solamente aparecerán las tablas, vistas, etc. seleccionadas y podrá dibujar conexiones de asignación entre ellas y los demás componentes de MapForce.

 © 2016-2022 Altova GmbH



Cuando desee cambiar los objetos de la base de datos que incluye el componente, haga clic con el botón derecho en el componente y seleccione **Agregar, quitar o editar objetos de la base de datos**.



4.2.2.4 Cambiar de BD y esquema

Para cambiar un componente de BD en una asignación para que use una conexión de BD nueva abra el cuadro de diálogo "Configuración del componente" 278, haga clic en **Cambiar** y siga las instrucciones del asistente para conectarse a una base de datos nueva. Si solo necesita cambiar los objetos de BD que aparecen en la asignación haga clic con el botón derecho en la barra del título del componente de BD y seleccione **Agregar, quitar o editar objetos de la base de datos** en el menú contextual. Véase también Agregar, editar y quitar objetos de BD 252.

Tenga en cuenta que, por defecto, aunque cambie la base de datos, esto no reasigna todas las conexiones a o desde el componente de BD. Sin embargo, con la mayoría de tipos de BD se pueden conservar las conexiones de la asignación después de cambiar una base de datos. Para ello debe seguir algunos pasos previos, que describimos a continuación.

Cuando seleccione objetos de la base de datos como se describe en el apartado Agregar, editar y quitar objetos de BD 252, puede elegir si quiere que los nombres de los objetos de BD se consideren como relativos a un esquema predeterminado. En el contexto de las bases de datos, un esquema es una agrupación lógica de objetos de BD, generalmente por razones de seguridad, como la asignación de permisos. Los esquemas se comportan de forma distinta en función del tipo de base de datos.

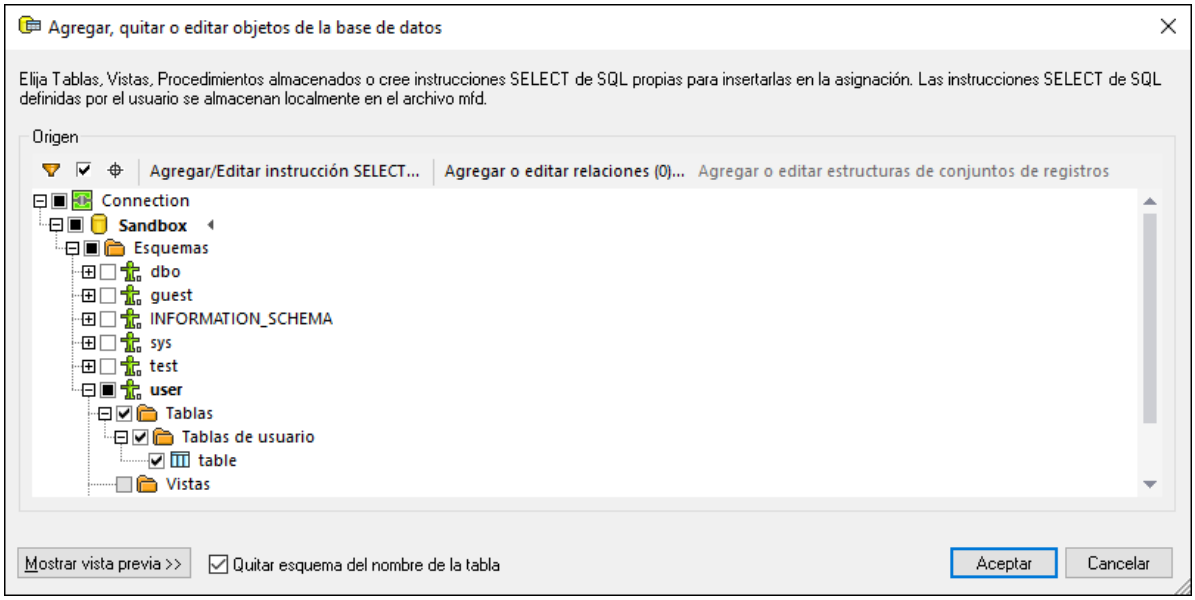
En MapForce es importante tratar los nombres de los objetos de BD como relativos a un esquema si quiere cambiar la BD de la asignación más adelante (por ejemplo, a una BD de producción). También es útil si se cambia el nombre del esquema de BD en el servidor en algún momento y necesita actualizar la asignación en consecuencia. Si el esquema nuevo tiene la misma estructura que el que se usó en tiempo de ejecución, puede cambiar a este sin tener que cambiar manualmente todas las conexiones de la asignación. Esto solo es posible si configura los nombres de los objetos como relativos a un esquema predeterminado, como se describe más abajo.

Tenga en cuenta que:

- se pueden usar nombres de objetos relativos a un esquema predeterminado en las bases de datos que son compatibles con los esquemas: IBM DB2, IBM Informix, IBM Db2 for i (iSeries), Oracle, PostgreSQL, Progress OpenEdge, SQL Server y Sybase.
- esta característica existe en MapForce a partir de su versión 2020.
- no se pueden usar nombres relativos si el componente de BD incluye relaciones locales 267 o instrucciones SQL SELECT como tablas virtuales 359.

Para tratar los nombres de los objetos de BD como relativos al esquema predeterminado:

1. Haga clic con el botón derecho en el componente de BD de la asignación y seleccione Agregar, editar y quitar objetos de BD 252 en el menú contextual.
2. Seleccione uno o más objetos que pertenezcan al esquema predeterminado o al catálogo y al esquema predeterminados (BD), según corresponda. La BD y el esquema predeterminados aparecen en negrita. En el ejemplo siguiente el catálogo predeterminado es **Sandbox** y el quema predeterminado es **user**. Esta estructura es específica para bases de datos de SQL Server y puede ser distinta en otros tipos de BD. Para más información sobre la estructura consulte Agregar, editar y quitar objetos de BD 252.



3. Marque la casilla *Usar nombres de archivos relativos al esquema predeterminado*. Tenga en cuenta que la casilla aparecerá deshabilitada si la base de datos no es compatible con nombres relativos de objetos.

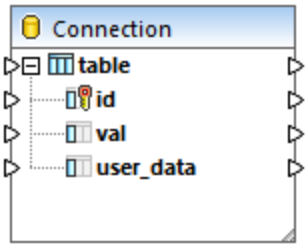
Si los objetos que necesita la asignación están en un esquema distinto (no en el predeterminado) tiene varias alternativas:

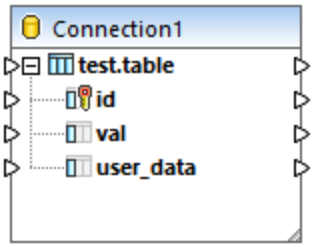
- conectarse como otro usuario de la BD que tenga acceso al esquema predeterminado que necesita.
- volver a configurar el servidor de BD para cambiar el esquema predeterminado del usuario de BD que ya existe, siempre que tenga los privilegios necesarios.

Por ejemplo, en SQL Server un administrados podría cambiar el esquema predeterminado de un usuario de BD con una instrucción como la que sigue (que asume que el catálogo actual es "Sandbox" y que tanto el usuario como el esquema ya existen).

```
USE [Sandbox]
GO
ALTER USER [test_user] WITH DEFAULT_SCHEMA=[test_schema]
GO
```

Los objetos de BD que son relativos al esquema predeterminado aparecen en la asignación con un nombre relativo. Observe las diferencias entre estos dos componentes de BD:





En el componente de la izquierda se marcó la casilla *Usar nombres de archivos relativos al esquema predeterminado*, por lo que la tabla se muestra con un nombre relativo. En el componente de la derecha no se marcó esta casilla, por lo que el nombre de la tabla incluye el nombre del esquema.

Cuando los objetos de BD son referenciados de forma relativa se pueden cambiar la BD o el esquema de la asignación *sin perder las conexiones*:

- abra el cuadro de diálogo "Configuración del componente" 278 y haga clic en **Cambiar**. Siga las instrucciones del asistente para conectarse a una BD nueva como usuario nuevo. Si los objetos tienen un nombre relativo y la BD nueva la misma estructura, entonces todas las conexiones de la asignación se actualizarán automáticamente. En concreto, ahora coincidirán con el catálogo y el esquema predeterminados del usuario nuevo de BD.
- si necesita realizar este cambio varias veces recomendamos que defina la conexión de BD como un recurso global 969, ya que este puede tener dos configuraciones: una predeterminada para la BD de desarrollo y una configuración de producción. Si asumimos que las dos bases de datos tienen la misma estructura, puede cambiar de una BD a otra cambiando de configuración en la lista desplegable:

Para más información sobre este tema consulte Ejemplo: cambiar de base de datos 980.

Si hay objetos de la BD que aparecen en rojo después del cambio significa que no existen en el esquema de BD nuevo.

Generación de código

La casilla *Usar nombres de archivos relativos al esquema predeterminado* afecta al código de programa generado en C#, C++ o Java. Cuando se marca esta casilla todas las referencias de BD se convierten en relativas también en el código generado. El siguiente fragmento de código muestra una instrucción SQL (en C#) si no se marca la casilla. Observe que el nombre de la tabla incluye el nombre del esquema.

```
var3_NewStatement = (Altova.Db.Dbs.NewStatement(closure.var1_Connection_Instance, 1,
"SELECT [id], [val], [user_data] FROM [user].[table]"));
```

El siguiente fragmento de código es un ejemplo de una instrucción SQL (en C#) si se marca la casilla:

```
var3_NewStatement = (Altova.Db.Dbs.NewStatement(closure.var1_Connection_Instance, 1,
"SELECT [id], [val], [user_data] FROM [table]"));
```

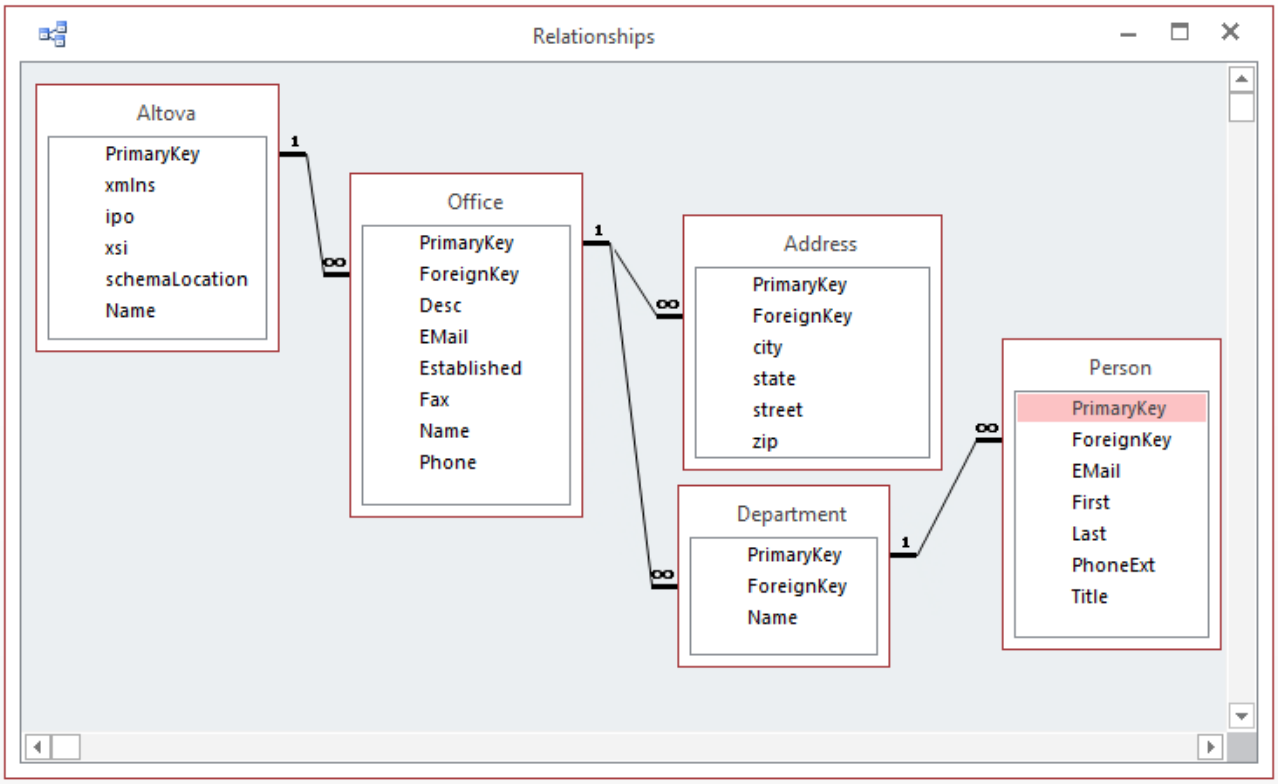
Nota: existe un método residual de quitar los nombres de los esquemas del código generado, que es marcando la casilla *Eliminar los nombres de esquema de los nombres de tabla* del cuadro de diálogo "Configuración del componente" 278. Este método solamente existe para garantizar la compatibilidad con archivos de versiones anteriores de la aplicación y no se recomienda en otros casos.

4.2.2.5 Gestionar relaciones de BD

Las bases de datos relacionales, tal y como indica su nombre, suelen tener relaciones definidas entre sus tablas. Por ejemplo, la base de datos **altova.mdb** de la carpeta **<Documentos>\Altova\MapForce2023\MapForceExamples\Tutorial** tiene estas relaciones:

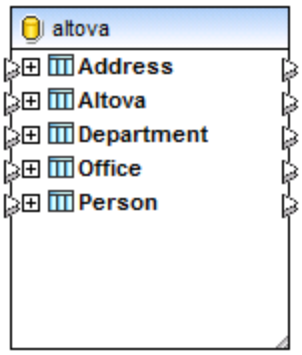
- La compañía del ejemplo (correspondiente a la tabla "Altova") está compuesta como mínimo por una oficina (p. ej. Brenton y Vereno). En las bases de datos esto se denomina *relación de uno a varios* y este tipo de relación existe entre las tablas "Altova" y "Office". En otras palabras por cada registro **PrimaryKey** de la tabla "Altova" puede haber varios registros **ForeignKey** en la tabla "Office". Por tanto, los registros de "Office" donde el valor de ForeignKey se corresponda con el valor de PrimaryKey de "Altova" se considerarán oficinas de "Altova".
- Cada oficina está compuesta como mínimo por un departamento (p. ej. "Marketing", "IT", "Development"). Una vez más, podemos decir que existe una *relación de uno a varios* entre las tablas "Office" y "Department".
- Por último, cada departamento está compuesto como mínimo por un empleado. Por tanto, existe una *relación de uno a varios* entre las tablas "Department" y "Person".

 © 2016-2022 Altova GmbH



Relaciones de tablas en la base de datos altova.mdb (vista "Relationships" de Microsoft Access)

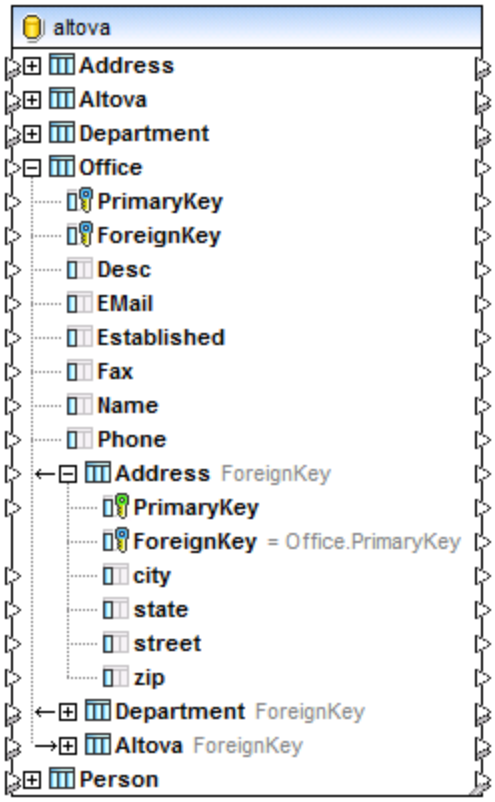
Las relaciones entre las tablas de la base de datos son muy importantes en las asignaciones de datos porque MapForce controla dichas relaciones de base de datos cuando la base de datos se añade al diseño de asignación. Esto permite conservar las relaciones mientas se crean las asignaciones de datos. Veamos un ejemplo para comprenderlo mejor: añada la base de datos **altova.mdb** a la asignación (con el comando de menú **Insertar | Base de datos**). A las tablas que aparecen a continuación las llamaremos tablas *raíz*:



Tablas raíz

Si expandimos cualquier tabla raíz, aparecen todas las tablas relacionadas. Por ejemplo, si expandimos la tabla **Office**, aparece la jerarquía de tablas relacionadas con esta tabla:

- La flecha ← de una tabla indica que la tabla es una tabla secundaria. Por ejemplo, **Address** es una tabla secundaria de **Office** y **Department** es una tabla secundaria de **Office**, así como una tabla hermana de **Address**, por lo que ambas tienen el mismo nivel de sangría. Como puede ver, la relación en la asignación de datos se corresponde con el diagrama que aparece al principio de este apartado.
- La flecha → indica que la tabla es una tabla primaria. Por ejemplo, **Altova** es la tabla primaria de **Office**.



Relaciones de tabla en MapForce (base de datos altova.mdb)

Esta representación jerárquica de las tablas ayuda a conservar las relaciones que existen en la base de datos cuando la asignación lee o escribe datos en una base de datos. Por ejemplo, imaginemos que queremos obtener todos los registros de la tabla **Person** y escribirlos en un archivo XML agrupados por departamento. Concretamente, el archivo XML debería vincular todos los empleados con un departamento (más o menos como en la base de datos **altova.mdb** utilizada en este ejemplo):

 © 2016-2022 Altova GmbH

Department

PrimaryKey	ForeignKey	Name
1	1	Administratior

PrimaryKey	EMail	First	Last	PhoneExt	Title
1	v.callaby@nan	Vernon	Callaby	582	Office Manage
2	f.further@nan	Frank	Further	471	Accounts Rece
3	l.matise@nan	Loby	Matise	963	Accounting Ma

PrimaryKey	ForeignKey	Name
2	1	Marketing

PrimaryKey	EMail	First	Last	PhoneExt	Title
4	j.firstbread@n	Joe	Firstbread	621	Marketing Mar
5	s.sanna@nano	Susi	Sanna	753	Art Director

PrimaryKey	ForeignKey	Name
3	1	Engineering

PrimaryKey	EMail	First	Last	PhoneExt	Title
6	f.landis@nano	Fred	Landis	951	Program Mana
7	m.landis@nan	Michelle	Butler	654	Software Engir
8	t.little@nanor	Ted	Little	852	Software Engir
9	a.way@nanon	Ann	Way	951	Technical Writ
10	l.gardner@nar	Liz	Gardner	753	Software Engir
11	p.smith@nanc	Paul	Smith	334	Software Engir

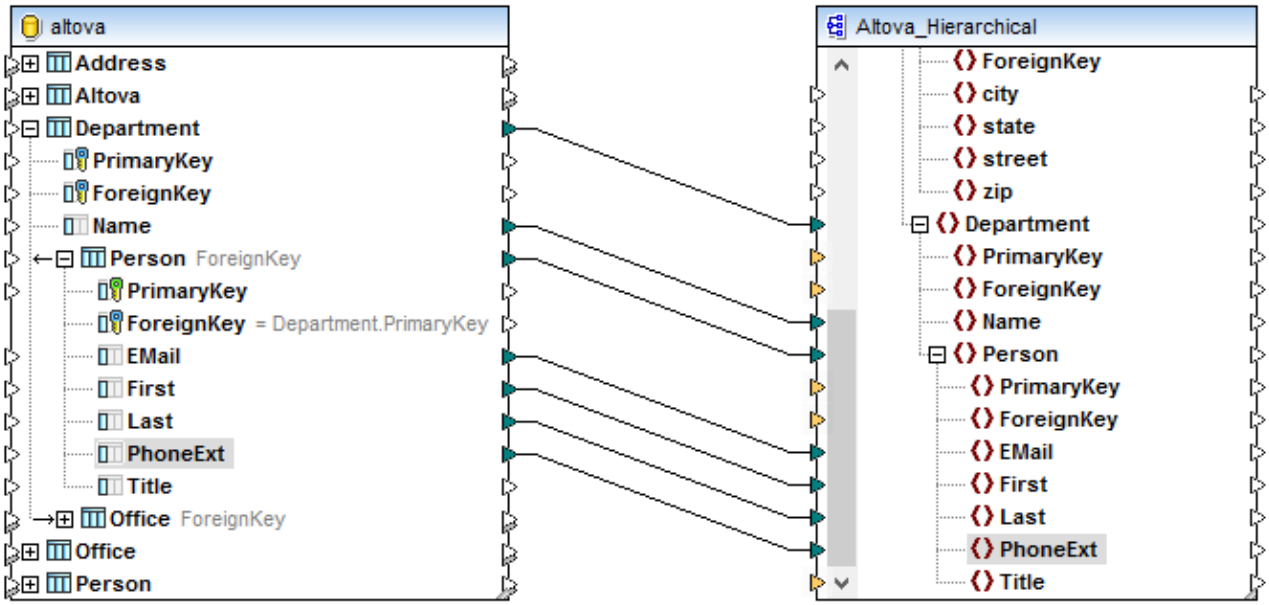
PrimaryKey	ForeignKey	Name
4	1	IT & Technical

PrimaryKey	EMail	First	Last	PhoneExt	Title
12	a.martin@nan	Alex	Martin	778	IT Manager
13	g.hammer@na	George	Hammer	223	Web Develope
14	j.band@nanor	Jessica	Bander	241	Support Engine
15	l.king@nanou	Lui	King	345	Support Engine

Record: 1 of 3 No Filter Search

Como muestra la imagen, el departamento "Administration" tiene tres empleados, "Marketing" tiene dos, "Engineering" tiene seis, etc.

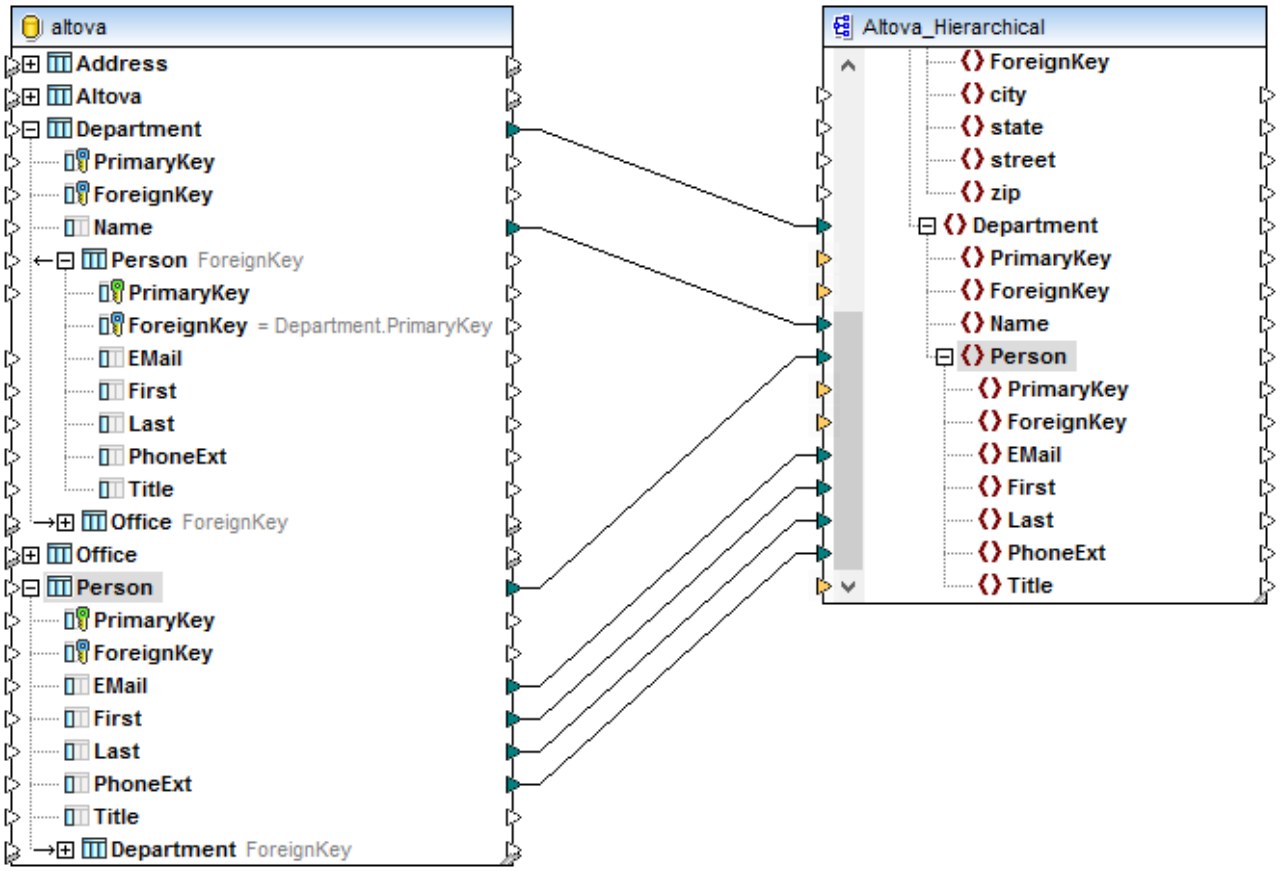
Si queremos que cada empleado se asigne al departamento correcto, debemos usar como tabla raíz la tabla **Department** y después crear asignaciones de datos desde la tabla **Person**, que es una tabla secundaria de **Department**.



La asignación de la imagen anterior es una versión modificada del archivo **DB_Altova_Hierarchical.mfd** de la carpeta **<Documentos>\Altova\MapForce2023\MapForceExamples**. En la vista previa de resultados veríamos que los empleados se agrupan por departamentos tal y como queríamos. Es decir, "Administration" tiene tres empleados, "Marketing" tiene dos, "Engineering" tiene seis, etc.

```
1    <?xml version="1.0" encoding="UTF-8"?>
2    <Altova xsi:noNamespaceSchemaLocation="file:///C:
3      <Office>
4        <Department>
5          <Name>Administration</Name>
6          <Person> ... </Person>
12         <Person> ... </Person>
18         <Person> ... </Person>
24       </Department>
25       <Department>
26         <Name>Marketing</Name>
27         <Person> ... </Person>
33         <Person> ... </Person>
39       </Department>
40       <Department>
41         <Name>Engineering</Name>
42         <Person> ... </Person>
48         <Person> ... </Person>
54         <Person> ... </Person>
60         <Person> ... </Person>
66         <Person> ... </Person>
72         <Person> ... </Person>
78       </Department>
79       <Department> ... </Department>
106      <Department> ... </Department>
121      <Department> ... </Department>
130      <Department> ... </Department>
151    </Office>
152  </Altova>
```

Ahora veamos otra versión de la misma asignación en la que las conexiones dibujadas convierten las tablas **Department** y **Person** en tablas raíz.



En esta ocasión, en la vista previa de resultados aparecen todos los empleados (sin importar su departamento de origen) en cada departamento de destino. Es decir, "Administration" tiene 21 empleados, "Marketing" tiene 21, "Engineering" tiene 21, etc.

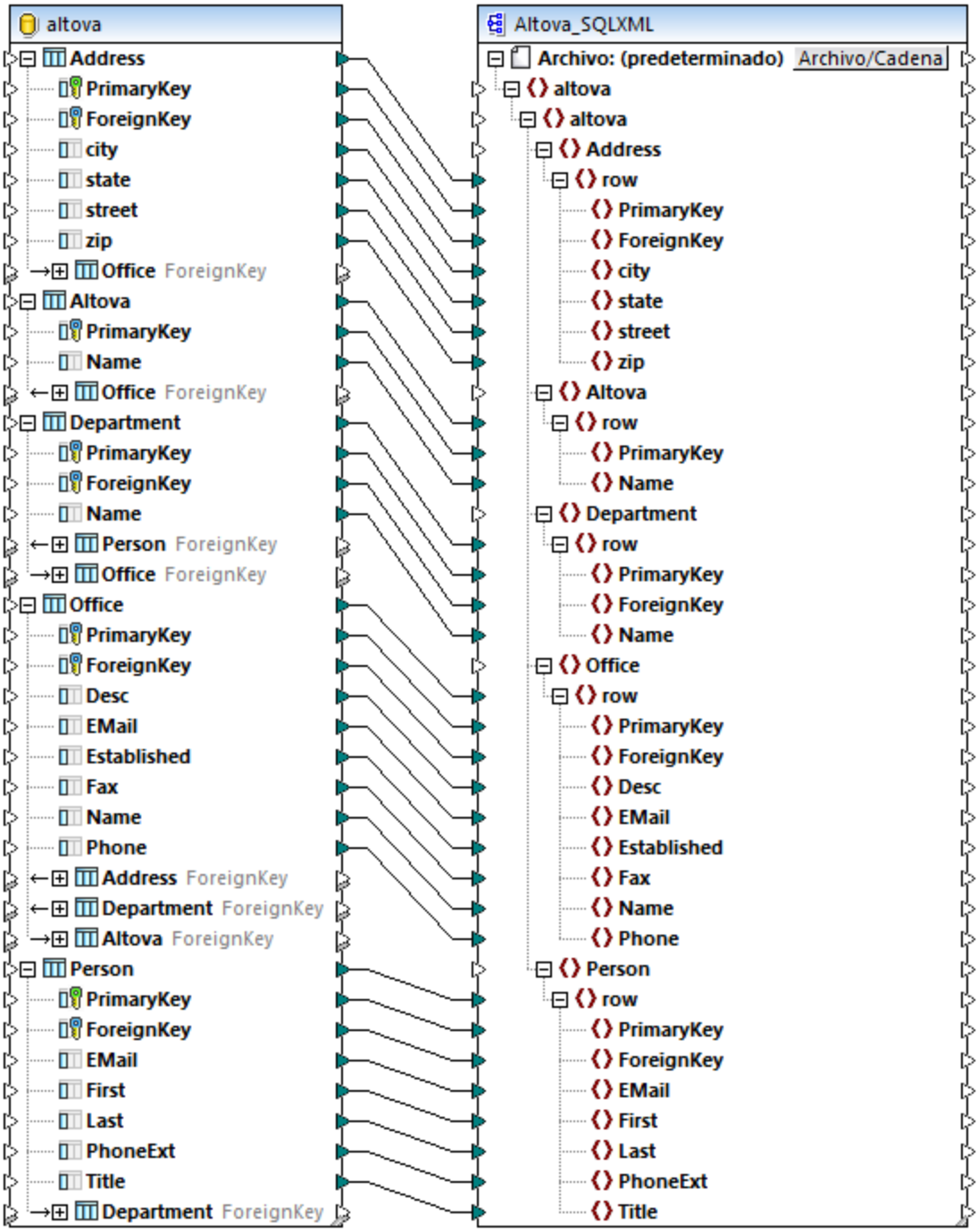
```
1    <?xml version="1.0" encoding="UTF-8"?>
2    <Altova xsi:noNamespaceSchemaLocation="file:///C:/
3      <Office>
4        <Department>
5          <Name>Administration</Name>
6          <Person> ... </Person>
12         <Person> ... </Person>
18         <Person> ... </Person>
24         <Person> ... </Person>
30         <Person> ... </Person>
36         <Person> ... </Person>
42         <Person> ... </Person>
48         <Person> ... </Person>
54         <Person> ... </Person>
60         <Person> ... </Person>
66         <Person> ... </Person>
72         <Person> ... </Person>
78         <Person> ... </Person>
84         <Person> ... </Person>
90         <Person> ... </Person>
96         <Person> ... </Person>
102        <Person> ... </Person>
108        <Person> ... </Person>
114        <Person> ... </Person>
120        <Person> ... </Person>
126        <Person> ... </Person>
132      </Department>
133      <Department> ... </Department>
262      <Department> ... </Department>
391      <Department> ... </Department>
520      <Department> ... </Department>
649      <Department> ... </Department>
778      <Department> ... </Department>
907    </Office>
908  </Altova>
```

En este segundo ejemplo, debido al modo en que se dibujaron las conexiones, MapForce no tiene en cuenta las relaciones de la base de datos.

Por tanto, si queremos conservar las relaciones de la base de datos, debemos asegurarnos de dibujar conexiones entre las mismas tablas raíz, pues estas contienen las tablas secundarias cuyas relaciones deseamos conservar, tanto si se trata de la base de datos de origen como de la de destino. Para ver ejemplos de asignaciones de datos que conservan las relaciones de las bases de datos consulte los archivos **DB_Altova_Hierarchical.mfd** y **Altova_Hierarchical_DB.mfd** de la carpeta **<Documentos>\Altova\MapForce2023\MapForceExamples** (*véase* *Insertar datos en tablas vinculadas* 286).

También puede darse el caso de que no sea necesario o conveniente conservar las relaciones de la base de datos. Por ejemplo, imaginemos que queremos exportar todos los datos de la base de datos **altova.mdb** a un archivo XML plano conforme a la especificación SQL/XML (parte 14 de la especificación SQL). Este tipo de asignación puede verse en el diseño **DB_Altova_SQLXML.mfd** de la carpeta **<Documentos>\Altova\MapForce2023\MapForceExamples**. El objetivo de la asignación es obtener los datos de la base de datos en forma de archivo XML plano. El esquema SQL/XML de destino se generó con XMLSpy (con el comando **Conversión | Crear esquema XML a partir de la estructura de la BD**).

 © 2016-2022 Altova GmbH



DB_Altova_SQLXML.mfd

Como muestra la imagen, a cada tabla de la base de datos le corresponde un elemento en el XML de destino. En la vista previa de resultados podemos ver que las filas de cada tabla de la base de datos se escriben en elementos "row" en el XML de destino.

```
1   <?xml version="1.0" encoding="UTF-8"?>
2   <altova xsi:noNamespaceSchemaLocation="file:///
3     <altova>
4       <Address>
5         <row> ... </row>
13        <row> ... </row>
21      </Address>
22      <Altova>
23        <row> ... </row>
27      </Altova>
28      <Department>
29        <row> ... </row>
34        <row> ... </row>
39        <row> ... </row>
44        <row> ... </row>
49        <row> ... </row>
54        <row> ... </row>
59        <row> ... </row>
64      </Department>
65      <Office>
66        <row> ... </row>
76        <row> ... </row>
86      </Office>
87      <Person>
88        <row> ... </row>
97        <row> ... </row>
106       <row> ... </row>
115       <row> ... </row>
124       <row> ... </row>
133       <row> ... </row>
142       <row> ... </row>
151       <row> ... </row>
160       <row> ... </row>
169       <row> ... </row>
178       <row> ... </row>
187       <row> ... </row>
196       <row> ... </row>
205       <row> ... </row>
214       <row> ... </row>
223       <row> ... </row>
232       <row> ... </row>
241       <row> ... </row>
250       <row> ... </row>
259       <row> ... </row>
268       <row> ... </row>
277     </Person>
278   </altova>
279 </altova>
```

Como puede verse en el XML de salida, no existen jerarquías entre los elementos XML, es decir, se trata de una estructura SQL/XML plana. Las relaciones de la base de datos se pasaron por alto porque se asignaron datos de varias tablas raíz de forma deliberada.

© 2016-2022 Altova GmbH

4.2.2.6 Definir relaciones locales

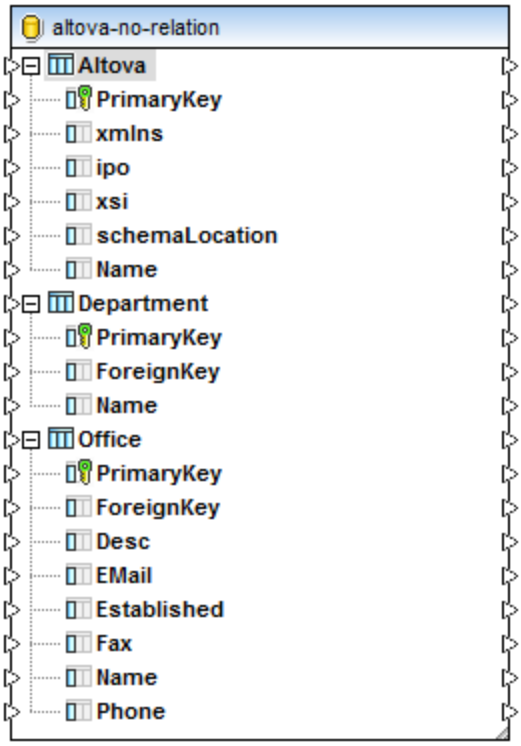
Cuando las tablas de una base de datos no tienen definidas relaciones explícitas, podemos definirlas en MapForce directamente. Concretamente, podemos crear relaciones de clave principal y foránea entre columnas de distintas tablas sin que ello afecte a la base de datos. Como claves principales y foráneas se puede usar cualquier columna de la base de datos. Además, se pueden crear relaciones nuevas además de las que ya existen en la base de datos. Las relaciones definidas localmente desde MapForce se guardan con la asignación.

En MapForce estas relaciones instantáneas se denominan ***relaciones locales***. La siguiente tabla enumera todos los campos posibles entre los que puede definir relaciones locales. Es posible crear relaciones mixtas (por ejemplo, asignar los resultados de un procedimiento almacenado a una columna de BD). Los campos que forman parte de la relación deben tener el mismo tipo de datos o uno compatible.

Clave principal/única	Clave foránea
• Columna de una tabla o vista de BD • Parámetro de salida o valor devuelto de un procedimiento almacenado (*consulte también Procedimientos almacenados* 395) • Columna de conjunto de registros devuelta por procedimiento almacenado* • Columna de una instrucción SELECT definida por el usuario (*consulte también Instrucciones SQL SELECT como tablas virtuales* 359).	• Columna de tabla o vista de BD • Parámetro de entrada de un procedimiento almacenado • Parámetro de entrada de una instrucción SELECT definida por el usuario

** Se aplica si se llama al procedimiento almacenado como fuente de datos (sin parámetros) o como función (con parámetros de entrada y salida). Para que el conjunto de registros esté disponible y se pueda seleccionar debe ejecutar el procedimiento almacenado una vez para obtener ese conjunto de registros.*

La base de datos **altova-no-relation.mdb** utilizada en este ejemplo es una versión simplificada de la base de datos **altova.mdb** que viene con MapForce. Las tablas "Person" y "Address", al igual que todas las relaciones de tabla, se han eliminado en Microsoft Access. Como podemos ver en la imagen siguiente, ninguna tabla del componente **altova-no-relation** tiene tablas secundarias. Es decir, todas están en el mismo nivel raíz. El contenido de cada tabla se limita a los campos que contiene.

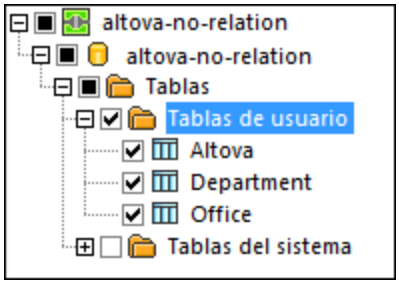


Estructura de base de datos sin relaciones explícitas

Imaginemos que queremos ver las oficinas de "Altova" y los departamentos de cada oficina. Recuerde que en **altova-no-relation.mdb** no existen relaciones de clave principal y foránea explícitas, pero existen de forma lógica (implícitamente), así que las recrearemos localmente en MapForce para alcanzar nuestro objetivo.

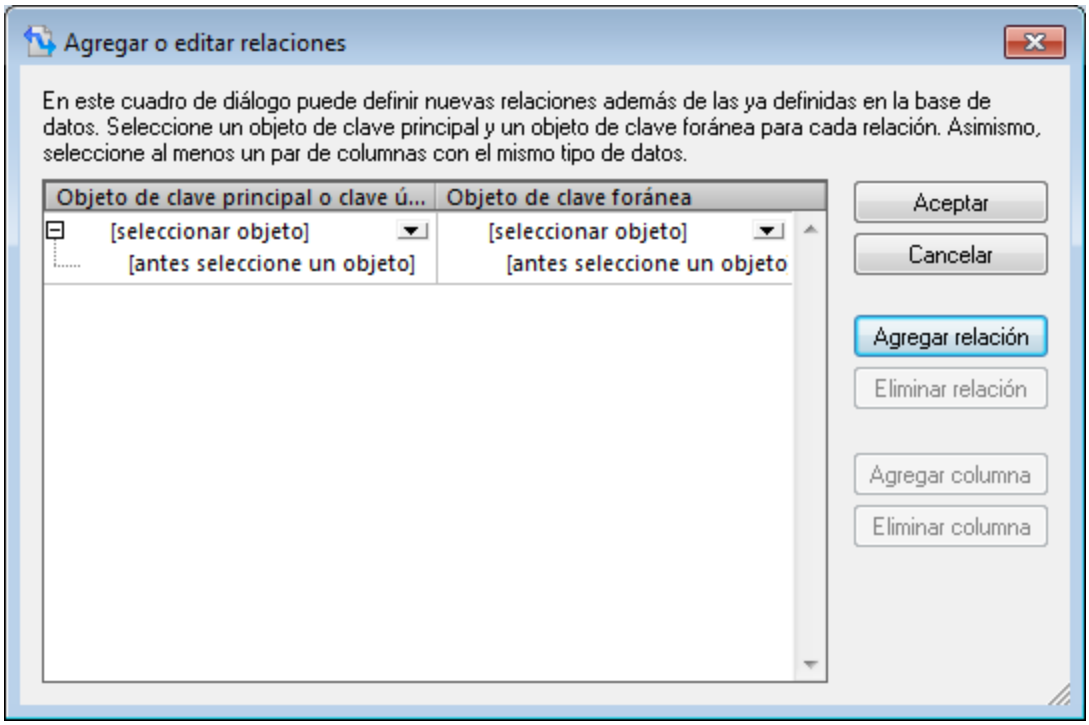
Podemos definir relaciones locales mientras añadimos la base de datos a la asignación o haciendo clic con el botón derecho en un componente de base de datos y eligiendo el comando **Agregar, quitar o editar objetos de la base de datos** del menú contextual:

1. En el menú **Insertar** haga clic en el comando **Base de datos**.
2. En el asistente para la conexión elija la opción *Microsoft Access (ADO)* y haga clic en **Siguiente**.
3. Navegue hasta la base de datos **altova-no-relation.mdb** de la carpeta **<Documentos>\Altova\MapForce2023\MapForceExamples\Tutorial** y haga clic en **Conectarse**.
4. Marque la casilla de `Tablas de usuario`.

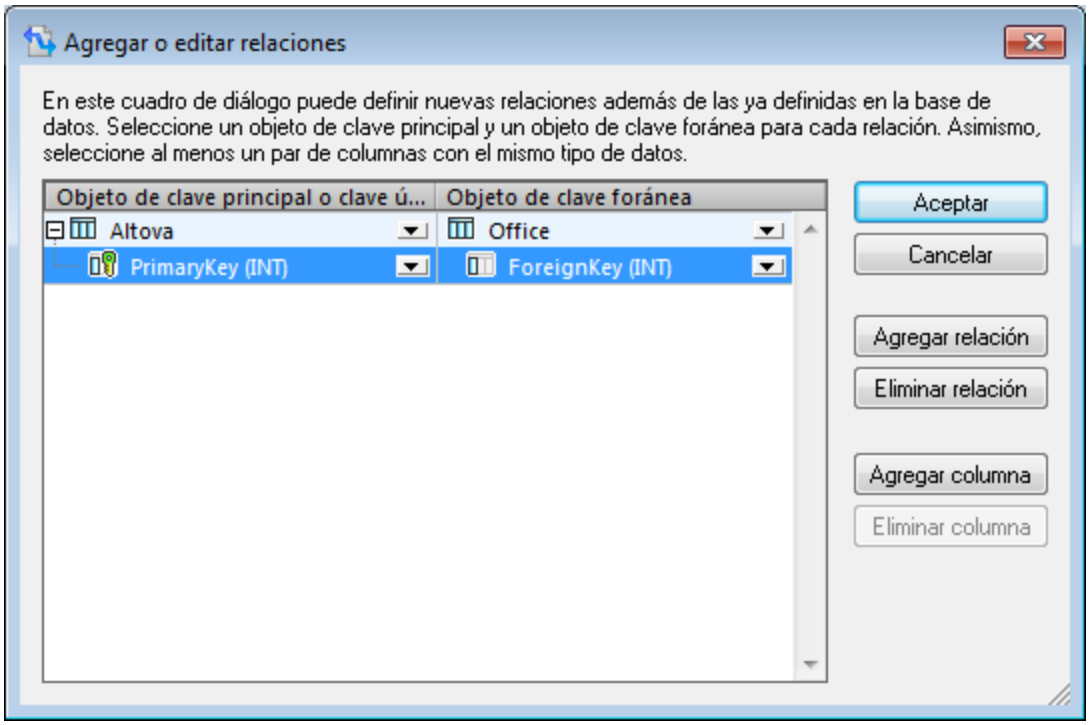


5. Haga clic en el botón **Agregar o editar relaciones** de la barra de herramientas.
6. Aparece el cuadro de diálogo "Agregar o editar relaciones". Haga clic en el botón **Agregar relación**.

 © 2016-2022 Altova GmbH

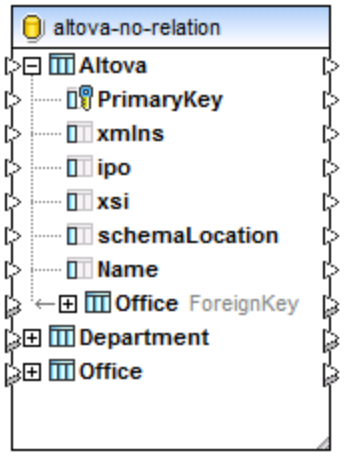


7. Seleccione valores en las dos listas desplegables para crear una relación de clave principal y foránea entre las tablas "Altova" y "Office" (*imagen siguiente*). En las dos listas desplegables puede seleccionar las tablas u objetos de base de datos para los que deben crearse relaciones. La lista de la izquierda especifica el objeto que almacena la clave principal/única, mientras que la derecha especifica el objeto de clave foránea. El objeto de clave principal/única será el objeto primario en MapForce, mientras que el objeto de clave foránea aparecerá como secundario en el componente de base de datos (*véase* *Gestionar relaciones de BD* 258).



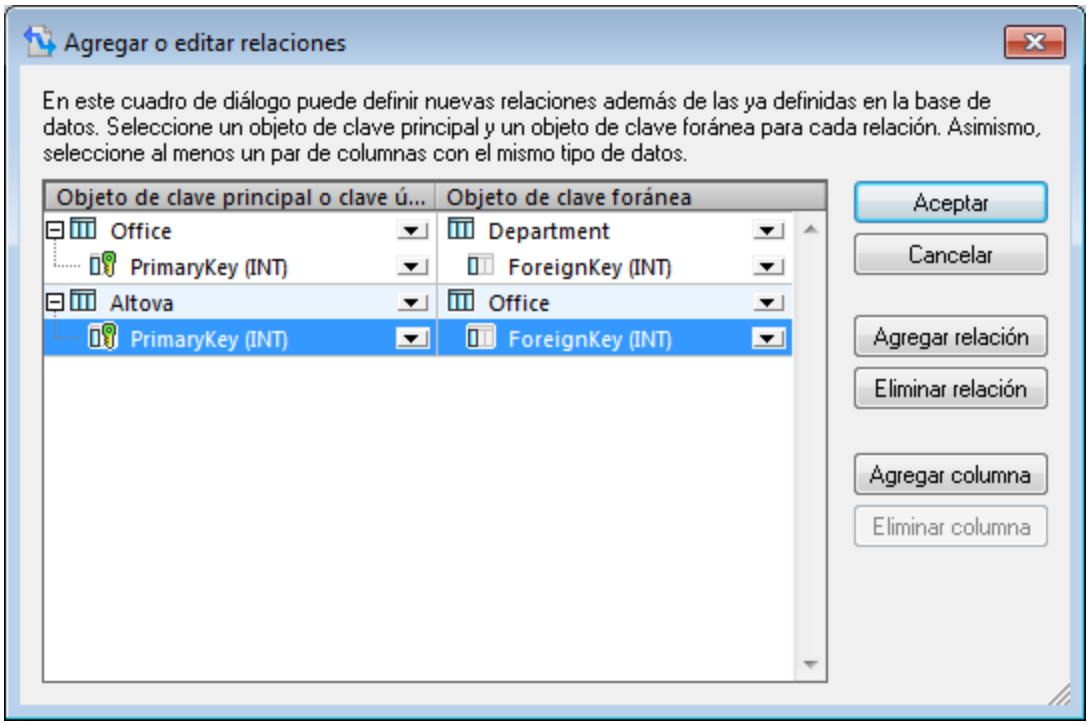
8. Haga clic en **Aceptar** para terminar de crear la definición de relación local y después haga clic en el botón **Insertar** para insertar la base de datos en el área de asignación.

Llegados a este punto, tenemos una relación local entre la columna **PrimaryKey** de la tabla "Altova" y la columna **ForeignKey** de la tabla "Office". Como muestra la imagen siguiente, la tabla raíz "Altova" ahora es la tabla primaria de la tabla "Office". Es decir, la tabla "Office" está relacionada con la tabla "Altova" y aparece con un icono de expansión.



Sin embargo, todavía no hemos terminado de crear relaciones. Todavía debemos crear una relación entre las tablas **Office** y **Department** (*imagen siguiente*).

 © 2016-2022 Altova GmbH

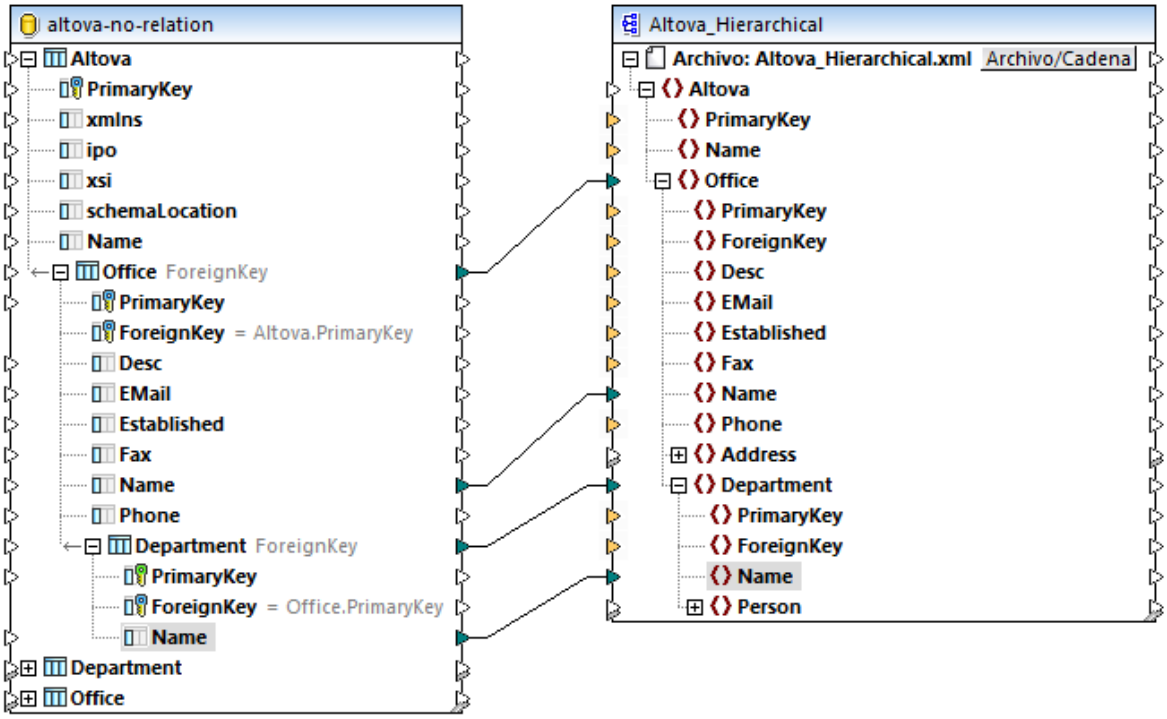


Para volver a abrir el cuadro de diálogo "Agregar o editar relaciones" haga clic con el botón derecho en el componente de base de datos y seleccione **Agregar, quitar o editar objetos de la base de datos** en el menú contextual.

Para terminar solo falta agregar el esquema de destino a la asignación:

1. En el menú **Insertar** haga clic en el comando **Insertar archivo o esquema XML**.
2. Navegue hasta el archivo **Altova_Hierarchical.xsd** de la carpeta **<Documentos>\Altova\MapForce2023\MapForceExamples**. Cuando la aplicación solicite un archivo XML, haga clic en **Omitir**. Cuando la aplicación solicite un elemento raíz, seleccione `Altova`.

Observe que, para poder conservar las relaciones que existen entre las tablas en el XML de destino, todas las conexiones se dibujaron desde la misma tabla raíz y jerárquicamente (desde la tabla "Altova". Para más información consulte el apartado Gestionar relaciones de BD 258.



Si abrimos el panel *Resultados*, veremos que los elementos Department aparecen anidados en la oficina correspondiente, tal y como queríamos.

 © 2016-2022 Altova GmbH

```
1   <?xml version="1.0" encoding="UTF-8"?>
2   <Altova xsi:noNamespaceSchemaLocation="file:///C:/U
3     <Office>
4       <Name>Nanonull, Inc.</Name>
5       <Department>
6         <Name>Administration</Name>
7       </Department>
8       <Department>
9         <Name>Marketing</Name>
10      </Department>
11      <Department>
12        <Name>Engineering</Name>
13      </Department>
14      <Department>
15        <Name>IT &amp; Technical Support</Name>
16      </Department>
17    </Office>
18    <Office>
19      <Name>Nanonull Partners, Inc.</Name>
20      <Department>
21        <Name>Administration</Name>
22      </Department>
23      <Department>
24        <Name>Marketing</Name>
25      </Department>
26      <Department>
27        <Name>IT &amp; Technical Support</Name>
28      </Department>
29    </Office>
30  </Altova>
```

4.2.2.7 Ejecutar asignaciones que modifican BD

Cuando una asignación de datos modifica los datos de una BD (p. ej. insertando, actualizando o eliminando registros), los cambios los aplica el motor que ejecuta la asignación. El motor que ejecuta la asignación puede ser MapForce, MapForce Server (independientemente o bajo la dirección de FlowForce Server) o el entorno de ejecución del código generado para C++, C# o Java.

Cuando se consulta la vista previa de resultados de una asignación en MapForce (en el panel *Resultados*), se obtiene un script de actualización. Este script no se llega a ejecutar en la base de datos, sino que se trata de una simple vista previa.

```
1   /*
2   Las instrucciones SQL que aparecen a continuación se presentan únicamente de forma preliminar y no se pueden ejecutar en otra herramienta de consultas SQL.
3   Para ejecutar estas instrucciones, utilice la función "Ejecutar script SQL" del menú "Resultados".
4   Conéctese a la base de datos usando esta cadena de conexión:
5   Data Source=C:\Users\Documents\Altova\MapForce2018\MapForceExamples\AltovaTarget.mdb;Provider=Microsoft.Jet.OLEDB.4.0
6   */
7
8   -- begin transaction
9
10  DELETE FROM [Person]
11
12  DELETE FROM [Department]
13
14  DELETE FROM [Address]
15
16  DELETE FROM [Office]
17
18  DELETE FROM [Altova]
19
20  SELECT IIF(MAX([Altova].[PrimaryKey]) IS NULL,0,MAX([Altova].[PrimaryKey]))+1 AS [PrimaryKey] FROM [Altova]
21  -->>> %PrimaryKey1%
22
23  INSERT INTO [Altova] ([Name], [PrimaryKey]) VALUES ('Organization Chart', '%PrimaryKey1%')
24
25  SELECT IIF(MAX([Office].[PrimaryKey]) IS NULL,0,MAX([Office].[PrimaryKey]))+1 AS [PrimaryKey] FROM [Office]
26  -->>> %PrimaryKey2%
```

Vista previa de resultados de una asignación que modifica una base de datos (Altova_Hierarchical_DB.mfd)

El script muestra instrucciones pseudo SQL de carácter puramente informativo; estas instrucciones coinciden con las que se van a ejecutar. Por ejemplo, si se definen varias acciones en una tabla (p. ej. "Actualizar si... Insertar el resto"), en la vista previa solo aparecerá la primera acción porque la segunda acción se ejecuta de forma condicional.

No aplique el pseudo script SQL manualmente a la base de datos con herramientas SQL distintas a las que hemos mencionado más arriba. El script puede contener formatos de valores no interpretables por editores SQL externos.

Si quiere aplicar los cambios de la asignación a la base de datos desde MapForce directamente, haga clic en el comando **Ejecutar script SQL** del menú **Resultados**. Recuerde que esta acción modificará realmente la base de datos inmediatamente.

Cuando la asignación se ejecuta con MapForce Server (independientemente o bajo la dirección de FlowForce Server), los cambios se aplican inmediatamente en la base de datos. Los mismo ocurre en el código generado: los cambios de la base de datos se aplican cuando se compila y ejecuta el código (p. ej. con el comando **Ejecutar** de Visual Studio).

Su instalación de MapForce incluye varios archivos de base de datos de ejemplo (Microsoft Access o SQLite), que están disponibles en la carpeta **<Documentos>\Altova\MapForce2023\MapForceExamples**. No se recomienda aplicar cambios en las bases de datos de ejemplo desde MapForce con el comando **Ejecutar script SQL** porque puede que tras aplicar los cambios estas bases de datos sean inservibles. Lo más recomendable es crear una copia de seguridad de la carpeta **<Documentos>\Altova\MapForce2023\MapForceExamples** antes de actualizar ningún archivo de esta carpeta.

En estos apartados de la documentación encontrará información sobre la ejecución de asignaciones de datos en entornos de ejecución externos a MapForce:

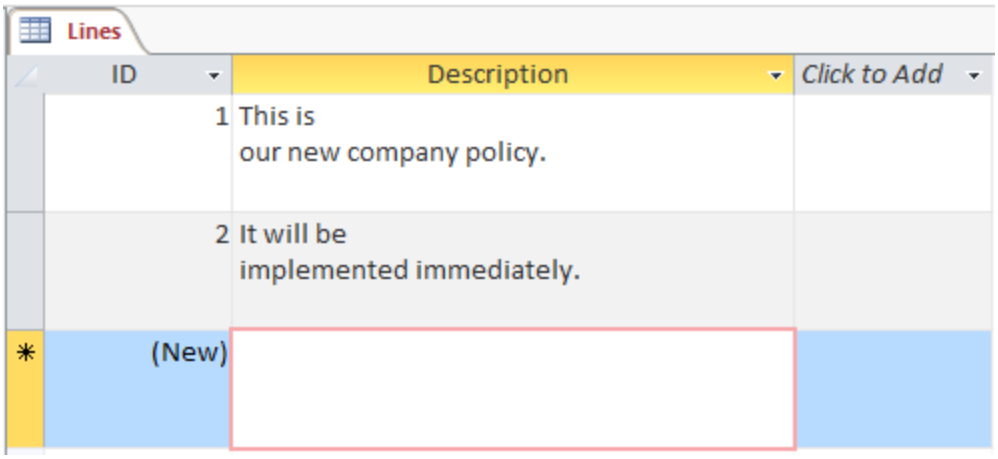
- Implementar asignaciones en FlowForce Server 958

- Compilar asignaciones en archivos de ejecución de MapForce Server 955
- Generador de código 1067

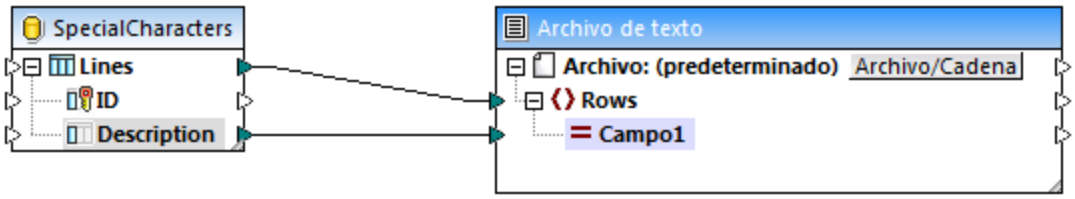
4.2.2.8 Reemplazar caracteres especiales

Cuando quiera transformar datos de una base de datos, primero puede ser necesario quitar ciertos caracteres especiales (como los caracteres de retorno de carro o de salto de línea) de la fuente de datos. Esto se puede hacer con la función **char-from-code** 707 de MapForce.

Imagine que tiene una base de datos Microsoft Access compuesta por una tabla llamada "Lines" que tiene dos columnas: "ID" y "Description".



El objetivo es extraer cada descripción a un archivo CSV (donde hay una descripción por línea). La asignación de datos que necesitamos para conseguir este objetivo es:

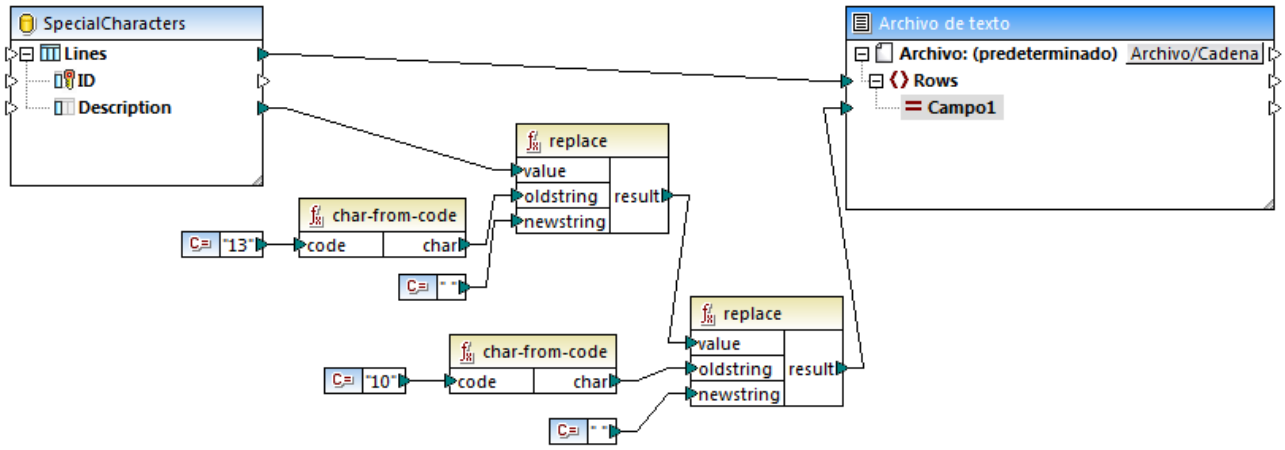


Sin embargo, como cada fila "Description" de Access contiene varias líneas separadas por caracteres CR/LF, el resultado de la asignación incluye también saltos de línea y este no es el resultado que queremos.

```
1   "This is
2   our new company policy."
3   "It will be
4   implemented immediately."
5
```

Para resolver este problema añadiremos las funciones **char-from-code** y **replace** de la biblioteca de MapForce (*véase* *Agregar una función* 550). Cada descripción debe procesarse de modo que, cuando se detecten caracteres especiales, éstos se reemplazarán con un espacio.

En el gráfico Unicode (http://www.unicode.org/charts/), los caracteres LF y CR corresponden a los caracteres **hex 0A | dec 10** y **hex 0D | dec 13** respectivamente. Por tanto, la asignación debe modificarse para convertir los valores Unicode decimales 13 y 10 en una cadena para que la función **replace** los pueda procesar.



En la vista previa de resultados de esta asignación podemos ver que los caracteres CR/LF de cada campo de la base de datos se reemplazaron con un espacio.

```
1   This is  our new company policy.
2   It will be  implemented immediately.
3
```

4.2.2.9 Gestionar valores null

Para comprobar en tiempo de ejecución de la asignación si un campo de la base de datos es NULL basta con usar las funciones **is-null** 723 y **is-not-null** 723 de la biblioteca de MapForce. Para ver en MapForce si una tabla tiene campos NULL basta con consultarla en el Explorador de bases de datos (*véase Examinar y consultar bases de datos* 380).

Para establecer un campo de base de datos como NULL basta con usar la función **set-null** 723.

Para reemplazar valores de base de datos NULL con una cadena podemos usar la función **substitute-null** 724 (puede ver un ejemplo de uso en el archivo **DB_ApplicationList.mfd** de la carpeta **<Documentos>\Altova\MapForce2023\MapForceExamples**).

Para aprender a trabajar con comparaciones de valores NULL en asignaciones que actualizan bases de datos consulte Gestionar valores null en acciones de tabla de BD 324.

Para obtener información sobre cómo gestionar valores NULL en asignaciones entre bases de datos y documentos XML consulte Valores Nil y Nillable 131.

 © 2016-2022 Altova GmbH

4.2.2.10 Generar valores secuenciales y únicos

Cuando se insertan datos o se actualiza una base de datos, puede ser necesario crear valores secuenciales o únicos de forma instantánea para los campos de base de datos que no reciben datos de ningún componente de origen. En este tipo de casos podemos usar estas funciones de la biblioteca integrada de MapForce:

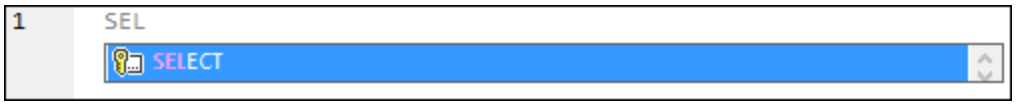
- **auto-number** 660 (de la biblioteca **core | generator functions**): suele utilizarse para generar valores de clave principal para un campo numérico.
- **create-guid** 746 (de la biblioteca **lang | generator functions**): esta función crea un identificador único global (como una cadena con codificación hexadecimal) para el campo especificado.

Tenga en cuenta que los valores para campos de base de datos se pueden escribir usando valores generados con la base de datos. Esta opción está disponible en el cuadro de diálogo "Acciones de tabla de la base de datos" (*véase Configurar acciones de tabla de BD* 327) y puede ser muy práctica a la hora de generar claves principales.

4.2.2.11 Sugerencias de finalización automática SQL

Cuando se teclean instrucciones SQL en determinados contextos, MapForce puede sugerir entradas de texto automáticamente. La finalización automática está disponible en estos contextos:

- En el editor SQL (*véase Examinar y consultar bases de datos* 380).
- En el cuadro de texto *Instrucción SQL personalizada* del cuadro de diálogo "Acciones de tabla de la base de datos" (*véase Configurar acciones de tabla de BD* 327).
- En el cuadro de diálogo "Introducir una instrucción SQL SELECT" (*véase Crear instrucciones SELECT* 360).



Finalización automática

Con las teclas **Arriba** y **Abajo** del teclado podrá navegar por la lista de sugerencias de la finalización automática. Para usar una entrada haga clic en ella o pulse **Entrar**.

Para deshabilitar las sugerencias de finalización automática:

1. En el menú **Herramientas** haga clic en el comando **Opciones** (o pulse **Ctrl+Alt+O**).
2. Dentro de **Base de datos** haga clic en **Editor SQL**.
3. En el grupo de opciones *Ayudantes de entrada* desactive la casilla *Abrir automáticamente*.

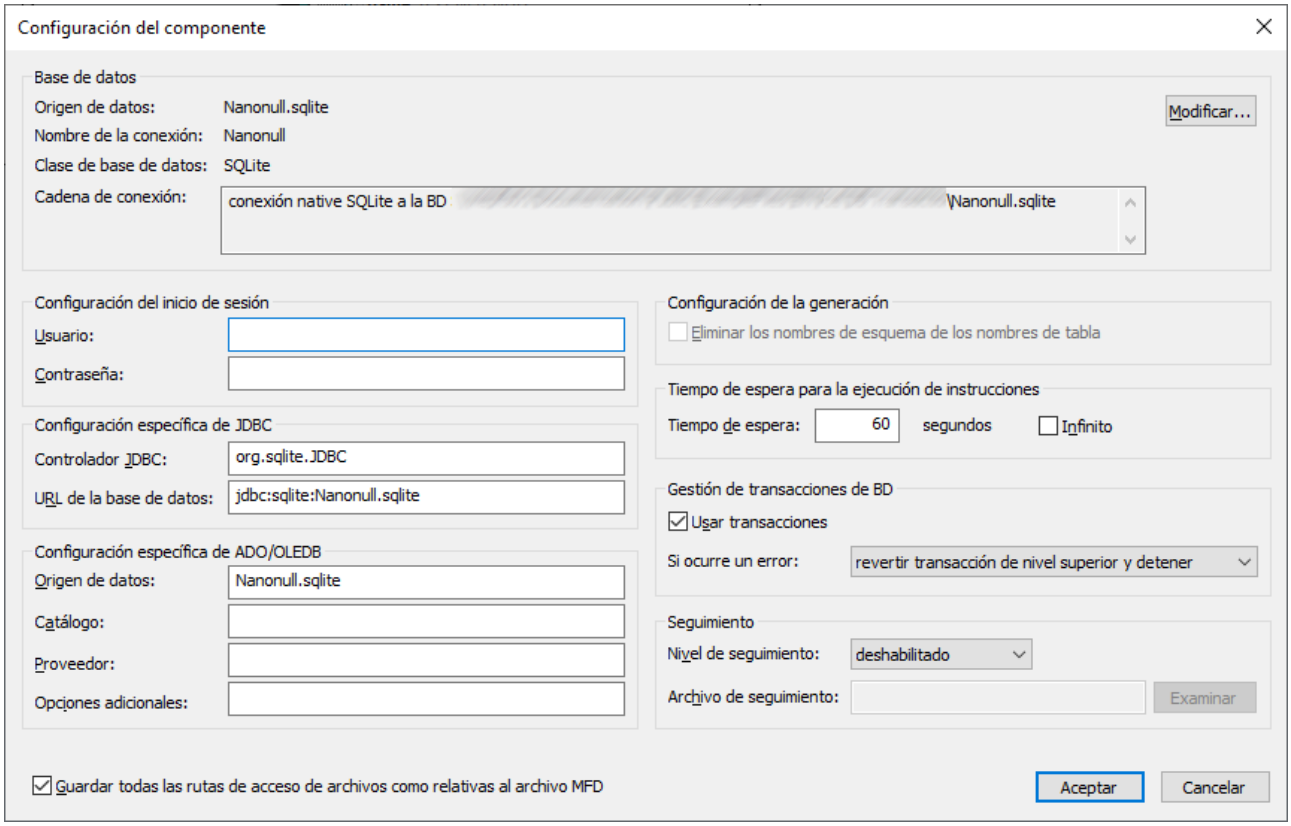
Para invocar las sugerencias de finalización automática a mano:

- Pulse **Ctrl+Espacio**.

4.2.2.12 Configurar componentes de BD

Tras agregar un componente de base de datos a la asignación, podrá editar su configuración desde el cuadro de diálogo "Configuración del componente". Este cuadro de diálogo se puede abrir de varias maneras:

- Seleccionando el componente en la asignación y después ejecutando el comando **Propiedades** del menú **Componentes**.
- Haciendo doble clic en la barra del título del componente.
- Haciendo clic con el botón derecho en el componente y eligiendo **Propiedades** en el menú contextuales.



Cuadro de diálogo "Configuración del componente" de un componente de base de datos

Estas son las opciones que se pueden configurar en este cuadro de diálogo.

Base de datos

En este panel de opciones aparece información de conexión de la base de datos. Haga clic en **Modificar** para seleccionar una base de datos distinta o para volver a definir los objetos actuales del componente de base de datos. Tras la modificación se conservarán los conectores de las tablas que tengan el mismo nombre. También pueden modificarse las tablas del componente (haciendo clic con el botón derecho en el componente y seleccionando **Agregar, quitar o editar objetos de la base de datos** del menú contextual).

 © 2016-2022 Altova GmbH

Origen de datos	Especifica el nombre del origen de datos actual. Si se trata de una base de datos basada en archivos, puede ser una ruta de acceso del sistema de archivos. Este campo indica si la base de datos basada en archivos se agregó a la asignación usando una ruta de acceso absoluta o relativa. Si se usó una ruta relativa, este campo contiene una ruta de acceso. Si se usó una ruta absoluta, el campo contiene el nombre de archivo de la base de datos.
Nombre de la conexión	Especifica el nombre de la conexión. Este nombre lo genera automáticamente MapForce. Normalmente es el mismo nombre que el del origen de datos, pero también puede ser un "alias" si realiza la conexión con los recursos globales de Altova 969. Si hay varios componentes de BD con el mismo nombre de conexión en la asignación, entonces el nombre cambiará a "<connection1>", "<connection2>", etc.
Clase de base de datos	Especifica el tipo de base de datos.
Cadena de conexión	Muestra la cadena de conexión de la base de datos actual. Es un campo de solo lectura que se genera a partir de la información indicada durante la fase de creación o modificación de la conexión.

Configuración del inicio de sesión

Estas opciones se usan para los destinos de la generación de código y para el motor de ejecución integrado.

Usuario	Permite cambiar el nombre de usuario que se utiliza para conectarse a la base de datos. Es obligatorio si la base de datos exige un nombre de usuario.
Contraseña	Permite cambiar la contraseña que se utiliza para conectarse a la base de datos. Es obligatorio si la base de datos exige una contraseña.

Configuración específica de JDBC

Estas opciones sirven para conectarse a la base de datos si la asignación contiene una conexión JDBC y se ejecuta con el código Java generado o con MapForce Server.

Nota: las conexiones ADO, ADO.NET y ODBC se convierten en JDBC (y, por tanto, se aplican estas opciones de configuración) cuando la asignación se ejecuta en un equipo Linux o macOS (*véase Bases de datos en MapForce* 160).

Controlador JDBC	Muestra el controlador que está activo para el componente de base de datos. Cuando se define un componente de base de datos se introduce automáticamente el controlador predeterminado, pero el valor de este campo se puede modificar

	si es necesario. Asegúrese de que la sintaxis del campo *URL de la base de datos* concuerda con el controlador elegido.
URL de la base de datos	URL de la base de datos que está seleccionada. Asegúrese de que su sintaxis concuerda con la del controlador JDBC (el controlador que aparece en el campo *Controlador JDBC*).

Configuración específica de ADO/OLEDB

Estas opciones sirven para conectarse a la base de datos si la asignación contiene una conexión ADO y se ejecuta con código C# o C++ o con MapForce Server desde Windows (*véase* *Bases de datos en MapForce* 160). Las opciones *Origen de datos* y *Catálogo* no se utilizan con el motor de ejecución integrado.

Origen de datos	Muestra el nombre del origen de datos ADO.
Catálogo	Muestra el nombre del catálogo ADO.
Proveedor	Muestra el proveedor que está activo para el componente de base de datos.
Opciones adicionales	Muestra opciones adicionales de la base de datos.

Configuración de la generación

Estas opciones son relevantes para todos los destinos de generación de código, así como para el motor de ejecución integrado.

Eliminar los nombres de esquema de los nombres de tabla	Permite eliminar los nombres de esquema de base de datos en el código generado y conservar solamente los nombres de tabla, lo cual permite una mayor flexibilidad. Recuerde que esta opción solamente funciona con las instrucciones SQL Select generadas con MapForce. Las instrucciones SQL definidas por el usuario, mientras se crean tablas virtuales 359, no se modificarán. Esta opción existe para garantizar la compatibilidad con versiones anteriores, por lo que debe evitarla si ese no es su caso. Para tratar a los objetos de BD como relativos al esquema predeterminado recomendamos que use el método que se describe en Cambiar de BD y esquema 255.

Tiempo de espera para la ejecución de instrucciones

Cuando se usa una base de datos como componente de destino, pueden superarse los tiempos de espera de ejecución debido a problemas de disponibilidad del servidor o de tráfico, a disparadores de larga ejecución, etc. Estas opciones permiten definir el tiempo de espera máximo antes de que se cierre la conexión. La configuración definida en este grupo de opciones tienen efecto durante las consultas de datos de la base de datos y en el código C#, Java y C++ generado.

© 2016-2022 Altova GmbH

Tiempo de espera	Define el plazo de tiempo máximo (en segundos) que debe esperar el motor de ejecución hasta recibir una respuesta de la base de datos antes de anular la ejecución de la instrucción. El valor predeterminado es 60 segundos.
Infinito	Si se marca esta casilla, el motor de ejecución no agotará el tiempo de espera nunca.

Nota: el tiempo de espera para la ejecución de instrucciones no afecta a las bases de datos SQLite.

Gestión de transacciones de BD

Usar transacciones	Habilita el procesamiento de transacciones cuando se usa una base de datos como destino. Cuando se detecta un error aparece un cuadro de diálogo donde el usuario puede elegir cómo se procede. Si se marca esta casilla, el procesamiento de transacciones se habilita para todas las tablas del componente de base de datos. Para más información consulte Revertir transacciones 422.

Seguimiento

Nivel de seguimiento	Esta opción se aplica a los componentes de BD de destino. Si se activa el seguimiento, las acciones que ejecute la asignación en la BD se almacenan en un archivo. Puede escoger entre registrar todas las acciones, solo los errores o no registrar nada. Para más información consulte Seguimiento de BD y registro de errores 435.
Archivo de seguimiento	Indica el archivo en el que se guarda la información del seguimiento una vez se ejecute la asignación. El archivo de seguimiento está en formato XML (véase Estructura del archivo de seguimiento 438). Esta ruta de acceso puede ser absoluta o relativa en función de si marca o no la casilla *Guardar todas las rutas de acceso de archivos como relativas al archivo MFD*.

Guardar todas las rutas de acceso de archivos como relativas al archivo MFD

Cuando se habilita esta opción, MapForce guarda las rutas de los archivos que aparezcan en el cuadro de diálogo "Configuración del componente" como relativas a la ubicación del archivo de diseño de MapForce (.mfd). Use rutas relativas si quiere ejecutar la asignación con MapForce Server en un sistema operativo distinto. Consulte también Usar rutas de acceso relativas en un componente 82.

4.2.3 Asignar datos a bases de datos

Esta sección incluye instrucciones y ejemplos para transferir datos desde cualquier origen de datos compatible con MapForce (p. ej. archivos XML) a una base de datos de destino. A continuación puede ver un resumen de todas las opciones disponibles.

Si quiere...	Consulte este tema...
Insertar datos en una tabla de base de datos de destino basada en datos suministrados por la asignación de datos...	• Insertar datos en una tabla 282
Controlar cómo se crean los valores de clave principal...	• Insertar datos en una tabla Table 282 • Insertar datos en varias tablas vinculadas 286
Ejecutar una instrucción SQL preliminar que se debe ejecutar antes de que la asignación modifique una tabla (p. ej. eliminar todos los registros de la tabla o una instrucción SQL personalizada)...	• Insertar datos en varias tablas vinculadas 286 • Configurar acciones de tabla de BD 327
Conservar la relación jerárquica existente entre los registros de tablas vinculadas por claves foráneas...	• Insertar datos en varias tablas vinculadas 286
Actualizar una tabla de forma condicional...	• Actualizar una tabla 291
Combinar registros en una tabla de BD (actualizar algunos registros e insertar otros en la misma tabla) basándose en una condición...	• Acción "Actualizar si... insertar restantes" 294 • Instrucciones de combinación MERGE 301
Preservar la integridad de la BD cuando se actualizan tablas que están vinculadas a otras tablas a través de relaciones de clave foránea...	• Opciones para tablas secundarias cuando se actualiza la tabla principal 303
Definir varias acciones en la misma tabla (p. ej. eliminar un registro si se cumple una condición o, de lo contrario, insertar un registro nuevo)...	• Acción "Eliminar si..." 311 • Acción "Omitir si..." 315
Preservar la integridad de los datos en caso de que falle la ejecución de la asignación...	• Gestionar excepciones de BD 422
Insertar varios registros en una tabla de BD de forma masiva (combinando varias instrucciones INSERT en una sola consulta)...	• Inserciones masivas (con MapForce Server) 320
Evitar resultados no deseados cuando se asignen datos a tablas de BD de destino que contengan valores null...	• Gestionar valores null en acciones de tabla de BD 324
Ver un ejemplo de cómo crear una asignación que actualice una base de datos...	• Ejemplo: asignar datos XML a SQLite 330

4.2.3.1 Insertar datos en una tabla

Una asignación de datos puede insertar datos en una tabla de BD desde cualquier tipo de componente de origen compatible con MapForce, incluidas otras bases de datos. Además puede configurar cómo se crea la clave principal de registros nuevos recién insertados. Por ejemplo, la clave principal se puede tomar de la asignación, generada por la base de datos, o se puede calcular a partir de valores de clave ya existentes en la tabla de BD.

 © 2016-2022 Altova GmbH

A continuación explicamos cómo insertar registros nuevos en una tabla de BD desde un archivo XML. También explicamos cómo se configura la generación de claves principales. Para ello nos serviremos de estos archivos:

- **altova-cmpy.xml**: que contiene los datos de origen que se deben insertar en la base de datos.
- **Altova_Hierarchical.xsd**: el esquema utilizado para validar el archivo de instancia anterior.
- **altova.mdb**: la base de datos de destino que se debe actualizar.

Todos estos archivos están en la carpeta **<Documentos>\Altova\MapForce2023\MapForceExamples** y, para simplificar, en las instrucciones omitimos su ruta de acceso.

La asignación de este ejemplo modifica un archivo de base de datos de ejemplo. Lo más recomendable es crear una copia de seguridad de la base de datos original y empezar con una copia nueva antes de seguir las instrucciones que aparecen más abajo. Así se evitarán posibles modificaciones en los ejemplos originales y obtendrá el mismo resultado que en este ejemplo. Para más información consulte Ejecutar asignaciones que modifican bases de datos 273.

El objetivo de la asignación de datos que nos ocupa es insertar compañías que se encuentran en el archivo **Altova-cmpy.xml** como registros nuevos en la tabla "Altova" dela base de datos **altova.mdb**. Si abre el archivo XML de origen, observará que solamente contiene una compañía llamada "Microtech OrgChart". Por tanto, la asignación de datos debe insertar un registro nuevo en la tabla "Altova" que se llamará "Microtech OrgChart". Además será necesario generar su clave principal.

Para conseguir este objetivo debemos seguir estos pasos:

Paso nº1: insertar el componente XML de origen

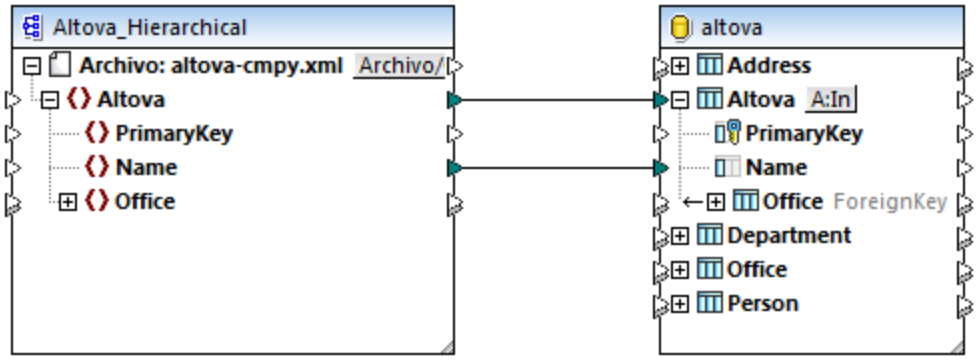
- En el menú **Insertar** haga clic en el comando **Archivo o esquema XML** y navegue hasta el esquema **Altova_Hierarchical.xsd**. Cuando la aplicación solicite un archivo de instancia, navegue hasta el archivo **Altova-cmpy.xml**.

Paso nº2: insertar la base de datos de destino

- En el menú **Insertar** haga clic en el comando **Base de datos** y siga los pasos del asistente para conectarse a **altova.mdb** (*véase* *Ejemplo: agregar la base de datos "altova.mdb" a la asignación* 249).

Paso nº3: dibujar las conexiones

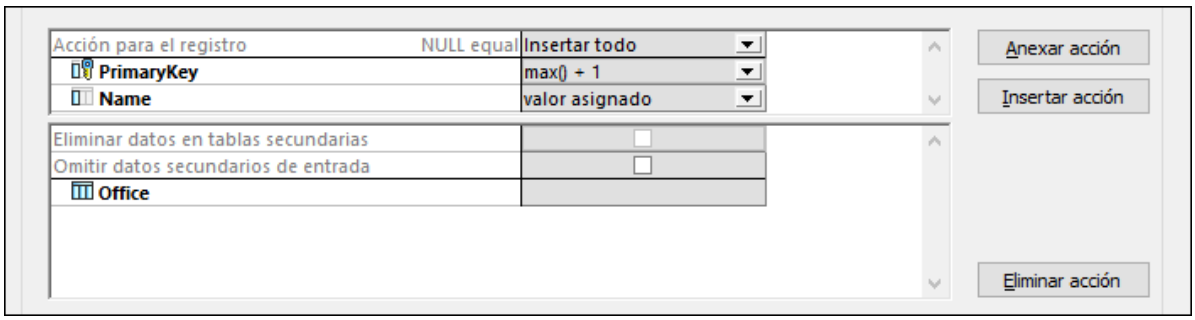
- Dibuje las conexiones de asignación de datos que aparecen a continuación.



Nota: si se dibujan conexiones no deseadas automáticamente para los elementos descendientes, esto significa que está activa la opción *Conectar automáticamente los secundarios equivalentes*. Si esto as así, para deshacer la última acción, debe seleccionar el comando Edición **| Deshacer**. Para deshabilitar la opción de conexión automática de secundarios, haga clic en el comando **Conexión | Conectar automáticamente los secundarios equivalentes**.

Paso nº4: configurar la acción Insertar

1. Observe que el componente de destino tiene un botón **Acción: Insertar** (A:In). Este botón aparece junto a cada tabla que tiene una conexión con el componente de origen (en este caso, la tabla "Altova"). Haga clic en este botón para configurar más detalladamente la acción de base de datos que se debe ejecutar (en este caso, la acción de inserción). Aparece el cuadro de diálogo "Acciones de tabla de la base de datos".
2. En el cuadro de diálogo, debajo del campo *Insertar todo* y junto a **`PrimaryKey`**, seleccione la opción **`max()` + 1**.



Las opciones que ofrecen estas listas desplegables son:

Opción	Descripción
valor asignado	Permite asignar los datos de origen al campo de base de datos directamente y se trata de la opción estándar para todos los campos de BD. También se puede usar un procedimiento almacenado para suministrar un valor de clave definiendo una relación (*véase* *Usar procedimientos almacenados para generar claves principales* 417).

max() + 1	Genera los valores de clave a partir de las claves que ya existen en la base de datos. Por ejemplo, si la tabla tiene tres registros con calves principales 1, 2 y 3, entonces **max() + 1** es 4. En este ejemplo, la tabla "Altova" solo tiene un registro con clave principal 1, así que **max() + 1** es 2, que es el valor esperado de la nueva clave principal.
generada por la BD	La base de datos usa la función de identidad para generar valores de clave.

La opción valor asignada elegida para **`Name`** significa que esta columna recibirá el valor de la asignación directamente. Para más información sobre las demás opciones consulte Configurar acciones de tabla de BD 327.

Paso nº5: vista previa de la asignación y actualizar la base de datos

Abra el panel *Resultados* para consultar una vista previa de los resultados de la asignación. La asignación genera un script SQL que contiene las acciones que se deben ejecutar en la base de datos. El script no ha modificado todavía la base de datos. Solamente se trata de una vista previa.

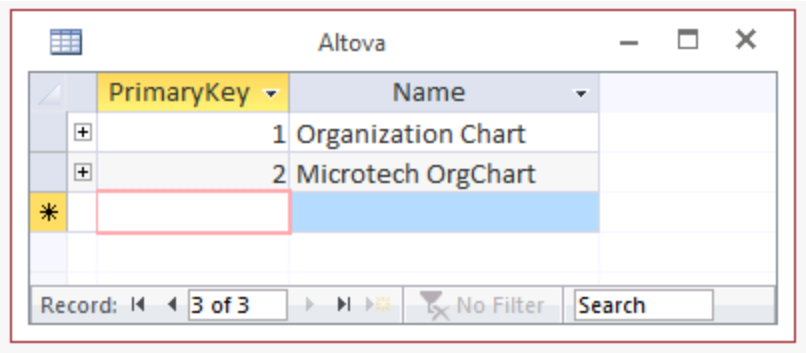
```
1   /*
2   Las instrucciones SQL que aparecen a continuación se presentan únicamente de forma preliminar y no se pueden ejecutar en otra herramienta de consultas SQL.
3   Para ejecutar estas instrucciones, utilice la función "Ejecutar script SQL" del menú "Resultados".
4   Conéctese a la base de datos usando esta cadena de conexión:
5   Data Source=S:\MapForce\DB Insert\altova.mdb;Provider=Microsoft.Jet.OLEDB.4.0
6   */
7
8   SELECT IIF(MAX([Altova].[PrimaryKey]) IS NULL,0,MAX([Altova].[PrimaryKey]))+1 AS [PrimaryKey] FROM [Altova]
9   --> > > %PrimaryKey1%
10
11  INSERT INTO [Altova] ([Name], [PrimaryKey]) VALUES ('Microtech OrgChart', '%PrimaryKey1%')
12
```

Para ejecutar el script en la base de datos:

- En el menú **Resultados** haga clic en **Ejecutar script SQL**.

Nota: el script SQL se puede ejecutar desde MapForce directamente pero hay otras maneras de actualizar la base de datos (*véase* *Ejecutar asignaciones que modifican bases de datos* 273).

Para ver el resultado abra la base de datos **altova.mdb** en DatabaseSpy o Access y observe que se añadió el registro "Microtech OrgChart" a la tabla "Altova" con la nueva clave principal 2. Los datos de este registro proceden de la instancia XML de entrada.



Hemos conseguido crear una asignación de datos que inserta datos en una tabla de base de datos. Para ver un ejemplo de asignación que inserte datos tanto en la tabla actual como en una tabla secundaria dependiente puede consultar el apartado Insertar datos en tablas vinculadas 286.

4.2.3.2 Insertar datos en tablas vinculadas

Las tablas de base de datos pueden ser tablas primarias, es decir, pueden ser referente de otras tablas de la base de datos, que le harán referencia mediante relaciones de clave foránea. Cuando la BD contenga este tipo de tablas podrá configurar la asignación de datos para que inserte registros en la tabla primaria pero también en sus tablas secundarias dependientes. Por ejemplo, cuando quiera insertar un registro "company" nuevo en una tabla de base de datos, también podrá insertar registros para las oficinas vinculadas a esta compañía, así como sus departamentos secundarios, sus empleados, etc.

A continuación explicamos cómo insertar datos en varias tablas sin perder las relaciones de la base de datos. Se trata de una versión algo más elaborada del ejemplo anterior (véase Insertar datos en una tabla 282. Para ello nos serviremos de estos archivos:

- **Altova_Hierarchical.mfd**: el archivo de asignación de datos propiamente dicho.
- **Altova_Hierarchical.xml**: el archivo XML que contiene los datos de origen que se deben insertar en la base de datos.
- **Altova_Hierarchical.xsd**: el esquema utilizado para validar el archivo de instancia anterior.
- **AltovaTarget.mdb**: la base de datos de destino que se debe actualizar.

Todos estos archivos están en la carpeta **<Documentos>\Altova\MapForce2023\MapForceExamples** y, para simplificar, en las instrucciones omitimos su ruta de acceso.

> La asignación de este ejemplo modifica un archivo de base de datos de ejemplo. Lo más recomendable es crear una copia de seguridad de la base de datos original y empezar con una copia nueva antes de seguir las instrucciones que aparecen más abajo. Así se evitarán posibles modificaciones en los ejemplos originales y obtendrá el mismo resultado que en este ejemplo. Para más información consulte Ejecutar asignaciones que modifican bases de datos 273.

El objetivo de la asignación de datos que nos ocupa es reemplazar datos de la base de datos de destino (**AltovaTarget.mdb**) con datos del archivo XML de origen. La estructura del archivo XML se corresponde con la estructura jerárquica de las tablas de la base de datos. Se trata de un organigrama que está estructurado de

 © 2016-2022 Altova GmbH

la siguiente manera: el elemento de nivel superior es una compañía compuesta por dos oficinas. Cada oficina está formada por varios departamentos y cada departamento por múltiples empleados. La misma jerarquía existe en la base de datos **AltovaTarget.mdb**, cuya tabla "Altova" corresponde a la compañía. Esta tabla está vinculada por medio de una relación de clave foránea a los registros de la tabla "Office". Igualmente "Office" está vinculada a "Department" y "Department" a "Person". Para ver un diagrama de relaciones de la base de datos **AltovaTarget.mdb**, ábrala en la vista "Relationships" de Access (*véase* *Gestionar relaciones de BD* 258).

Para conseguir este objetivo debemos seguir estos pasos:

Paso nº1: insertar el componente XML de origen

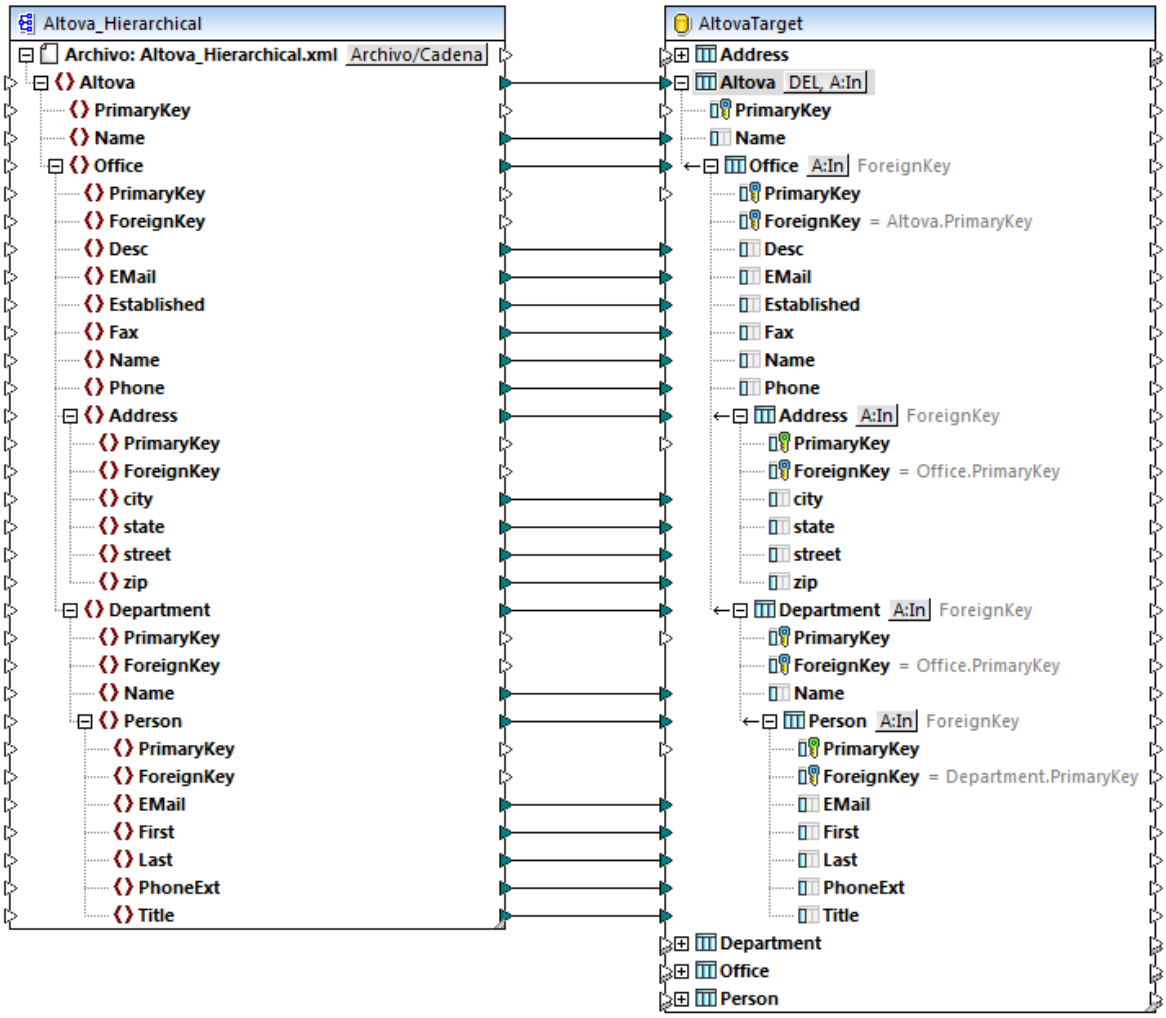
- En el menú **Insertar** haga clic en el comando **Archivo o esquema XML** y navegue hasta el esquema **Altova_Hierarchical.xsd**. Cuando la aplicación solicite un archivo de instancia, navegue hasta el archivo **Altova_Hierarchical.xml**.

Paso nº2: insertar la base de datos de destino

- En el menú **Insertar** haga clic en el comando **Base de datos** y siga los pasos del asistente para conectarse a **AltovaTarget.mdb** (*véase* *Ejemplo: agregar la base de datos "altova.mdb" a la asignación* 249).

Paso nº3: dibujar las conexiones

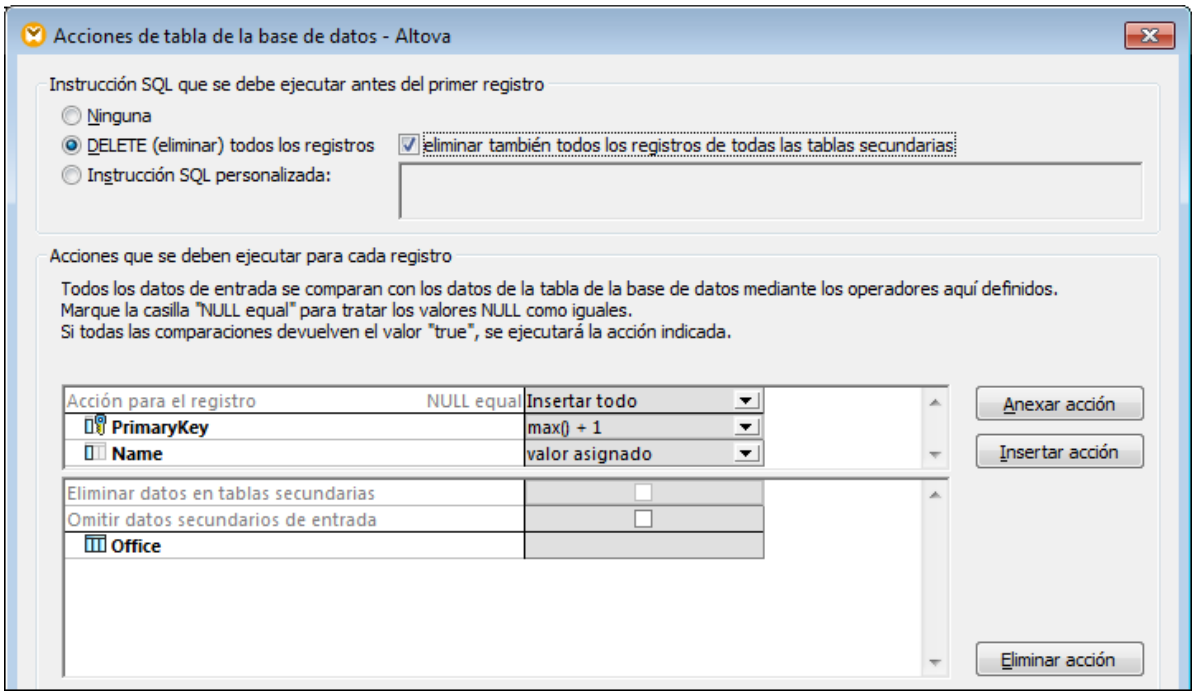
- Dibuje las conexiones de asignación de datos que aparecen en la imagen. Observe que las claves principal y foránea no están asignadas porque se generarán de forma instantánea (ver más abajo).



Nota: si se dibujan conexiones no deseadas automáticamente para los elementos descendientes, esto significa que está activa la opción *Conectar automáticamente los secundarios equivalentes*. Si esto as así, para deshacer la última acción, debe seleccionar el comando Edición **| Deshacer**. Para deshabilitar la opción de conexión automática de secundarios, haga clic en el comando **Conexión | Conectar automáticamente los secundarios equivalentes**.

Paso nº4: configurar las acciones Insertar

1. En el componente de destino haga clic en el botón **Acción: Insertar** (A:In) situado junto a la tabla "Altova" y configure la opción **max() + 1** de la clave principal (*imagen siguiente*). Esta opción se explica con más detalle en el apartado anterior (*véase* *Insertar datos en una tabla* 282).

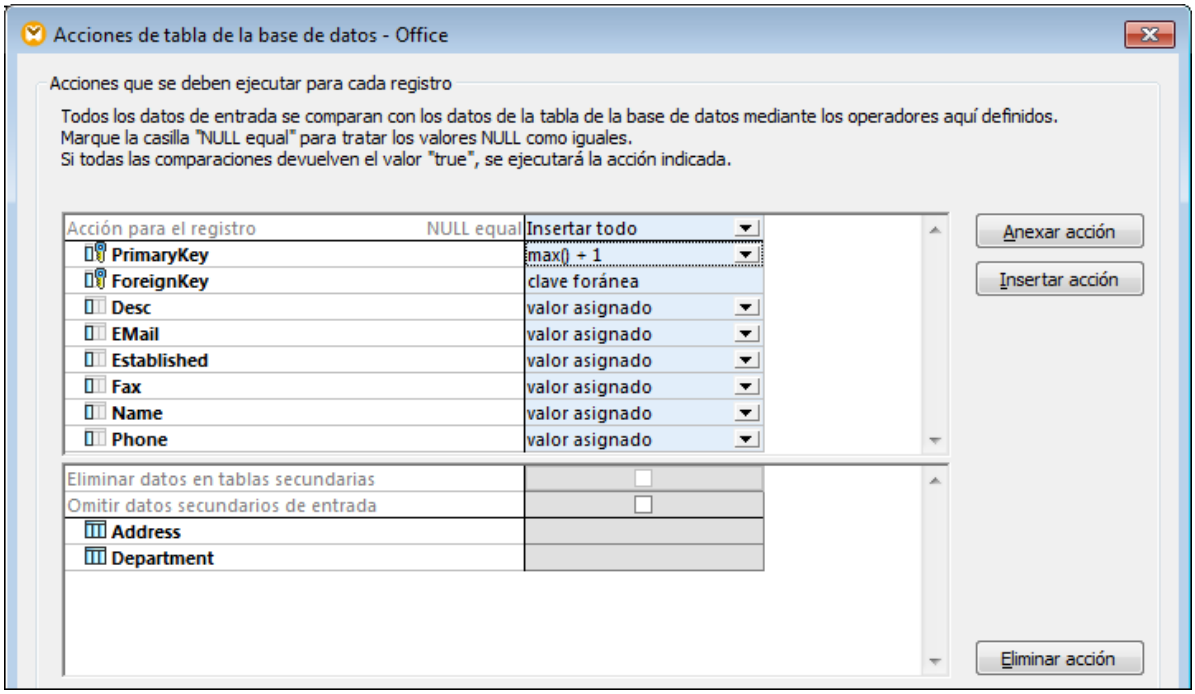


Observe que la opción *ELIMINAR todos los registros* está habilitada. Esto borra todos los registros que existen en la tabla antes de introducir los nuevos, tal y como queremos. Si deshabilita esta opción, los nuevos registros (con una clave principal nueva) se añadirán a la base de datos junto a los que ya existen cada vez que se ejecute la asignación. Pero esto no es lo que queremos.

En nuestro ejemplo también está habilitada la opción *eliminar también todos los registros de todas las tablas secundarias*. Esto garantiza que solamente se eliminen los registros de la tabla "Altova", pero también todos los registros que estén vinculados a la tabla "Altova" por medio de relaciones de clave foránea. Si las tablas secundarias tienen tablas secundarias propias, sus registros también se eliminarán y así sucesivamente hasta llegar al final de la jerarquía. Si intenta eliminar los registros de la tabla raíz "Altova" solamente, se comprometerá la integridad de la base de datos y la asignación no se podrá ejecutar.

Para obtener más información sobre las demás opciones del cuadro de diálogo "Acciones de tabla de la base de datos" consulte el apartado Configurar acciones de tabla de BD 327.

2. Haga clic en **Aceptar** para cerrar el cuadro de diálogo. Observe que en el área de asignación el aspecto del botón es DEL, A:In. Esto indica que se configuró una instrucción Delete para que la eliminación tenga lugar antes de la acción Insert.
3. Haga clic en el botón A:In de la tabla "Office" y configure la opción **max() + 1** de la clave principal.



4. Repita el paso nº3 para cada tabla a partir de la tabla "Office", es decir, en las tablas "Address", "Department" y "Person". Asegúrese de que todas estas tablas son descendientes inmediatos de la tabla raíz "Altova". Si necesita información sobre qué es una tabla raíz y por qué se necesita, consulte Gestionar relaciones de BD 258.

Paso nº5: vista previa de la asignación y actualizar la base de datos

Abra el panel *Resultados* para consultar una vista previa de los resultados de la asignación. La asignación genera un script SQL que contiene las acciones que se deben ejecutar en la base de datos. El script no ha modificado todavía la base de datos. Solamente se trata de una vista previa.

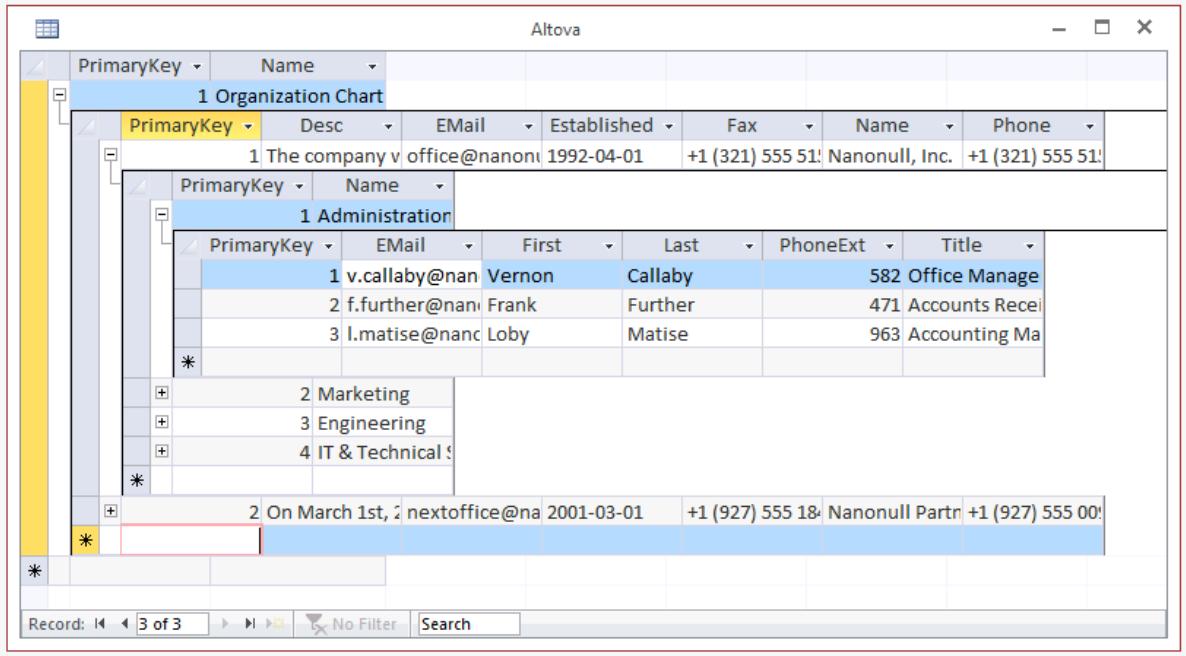
Para ejecutar el script en la base de datos:

- En el menú **Resultados** haga clic en **Ejecutar script SQL**.

Nota: el script SQL se puede ejecutar desde MapForce directamente pero hay otras maneras de actualizar la base de datos (*véase* *Ejecutar asignaciones que modifican bases de datos* 273).

Para ver el resultado abra la tabla "Altova" en Microsoft Access y observe cómo las relaciones del archivo XML se han copiado en la base de datos, desde la tabla "Altova" hasta la tabla "Person".

 © 2016-2022 Altova GmbH



Hemos conseguido crear una asignación de datos que inserta datos en varias tablas de la base de datos sin comprometer la integridad de las relaciones existentes entre las tablas.

4.2.3.3 Actualizar una tabla

Con este ejemplo explicamos cómo actualizar datos de una tabla de base de datos con datos procedentes de un XML de origen. Para ello nos serviremos de estos archivos:

- **altova-cmpy.xml**, que contiene los datos de origen que se deben insertar en la base de datos.
- **Altova_Hierarchical.xsd**, el esquema que se utilizará para validar el archivo de instancia anterior.
- **altova.mdb**, la base de datos de destino que se debe actualizar.

Todos estos archivos están en la carpeta **<Documentos>\Altova\MapForce2023\MapForceExamples** y, para simplificar, en las instrucciones omitimos su ruta de acceso.

> La asignación de este ejemplo modifica un archivo de base de datos de ejemplo. Lo más recomendable es crear una copia de seguridad de la base de datos original y empezar con una copia nueva antes de seguir las instrucciones que aparecen más abajo. Así se evitarán posibles modificaciones en los ejemplos originales y obtendrá el mismo resultado que en este ejemplo. Para más información consulte Ejecutar asignaciones que modifican bases de datos 273.

El objetivo de la asignación de datos que nos ocupa es actualizar todos los registros de la tabla "Person" con instancias de "Person" del documento XML. Cada persona del archivo XML tiene un elemento secundario llamado `PrimaryKey`. Cada persona de la tabla "Person" tiene una columna llamada `PrimaryKey`. Solamente se deben actualizar los registros que correspondan a personas del archivo XML con un secundario `PrimaryKey` que coincida con la columna `PrimaryKey` de una persona de la base de datos.

Para conseguir este objetivo debemos seguir estos pasos:

Paso nº1: insertar el componente XML de origen

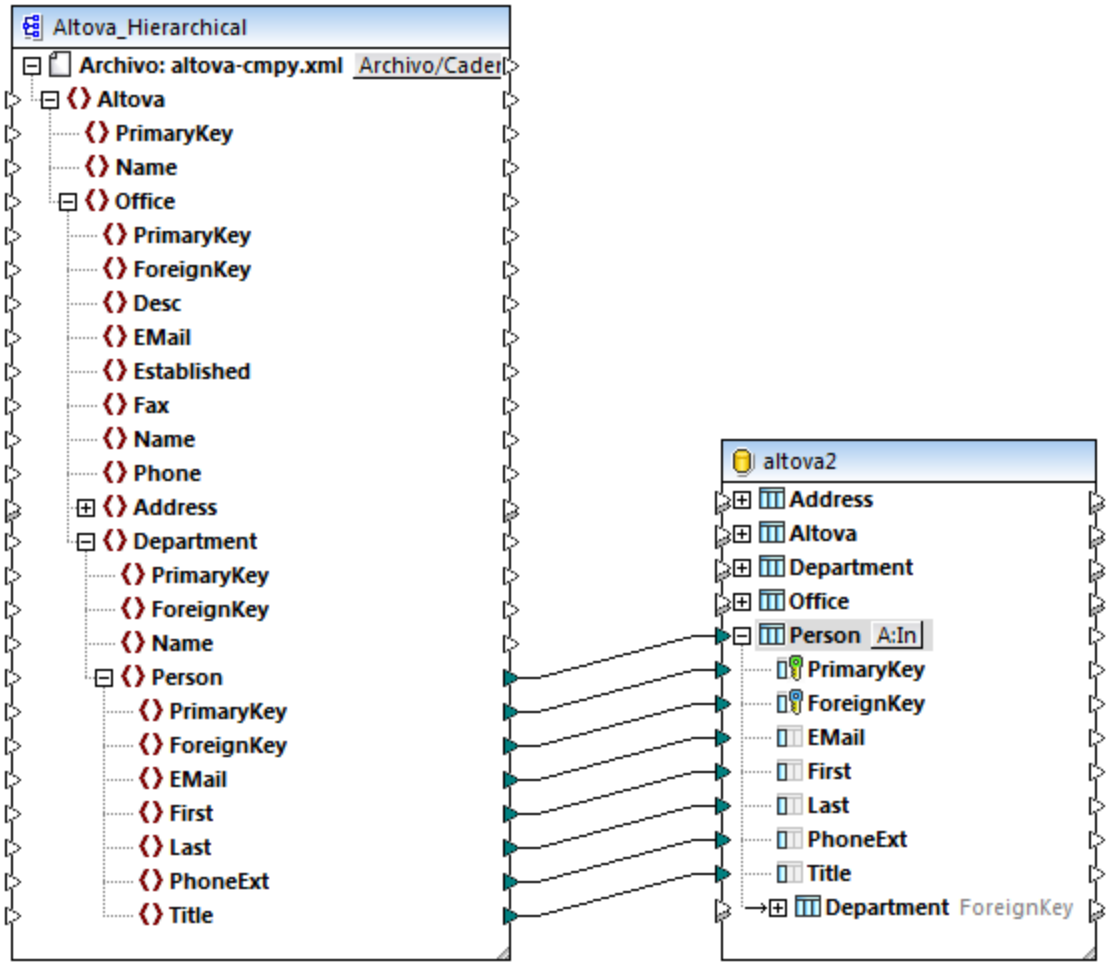
- En el menú **Insertar** haga clic en el comando **Archivo o esquema XML** y navegue hasta el esquema **Altova_Hierarchical.xsd**. Cuando la aplicación solicite un archivo de instancia, navegue hasta el archivo **altova-cmpy.xml**.

Paso nº2: insertar la base de datos de destino

- En el menú **Insertar** haga clic en el comando **Base de datos** y siga los pasos del asistente para conectarse a **altova.mdb** (*véase* *Ejemplo: agregar la base de datos "altova.mdb" a la asignación* 249).

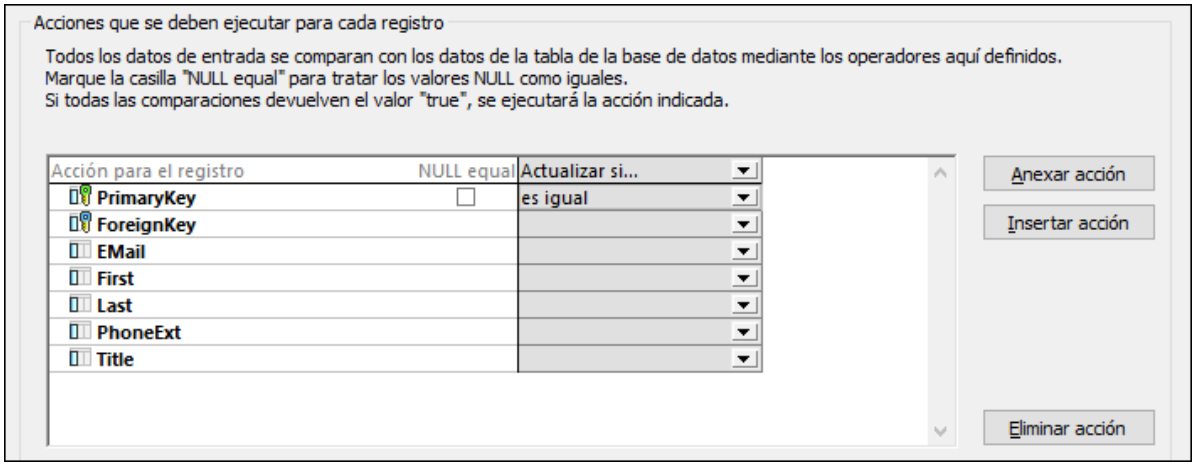
Paso nº3: dibujar las conexiones

- Dibuje las conexiones de asignación de datos que aparecen en la imagen.



Paso nº4: configurar la acción Actualizar

1. En el componente de destino haga clic en el botón **Acción: Insertar** A:In situado junto a la tabla "Person".
2. Junto al campo *Acción para el registro* seleccione la opción **Actualizar si...**, que cambiará la acción de tabla de base de datos a una acción de actualización condicional. Es decir, el registro actual solamente se actualizará si se cumple determinada condición (ver paso siguiente).
3. Junto al campo **`PrimaryKey`**, seleccione el valor **`es igual`** (*imagen siguiente*). Esto define la condición de actualización: es decir, el registro de la base de datos solamente se actualizará cuando el valor de su **`PrimaryKey`** sea igual al valor de **`PrimaryKey`** que procede de la asignación.



En este ejemplo el operador de igualdad se aplica al campo **`PrimaryKey`**, que suele ser el caso en tareas de actualización de bases de datos. Tenga en cuenta que también se pueden definir condiciones en otros campos y no solo en claves principales. Por ejemplo, si selecciona el valor **`es igual`** de los campos **`First`** y **`Last`**, podría actualizar los registros cuyo nombre y apellido coincida con los del XML de origen.

Puede optar por marcar la casilla *Null equal*. En este ejemplo esta casilla es irrelevante, ya que ni el campo *ClavePrimaria* del XML de origen ni el campo *ClavePrimaria* de la BD contienen valores `null`. Sin embargo, debería marcar eta casilla si sus datos contienen valores `null` y quiere que estos sean tratados como iguales; de lo contrario puede que no obtenga los resultados deseados. Para más información consulte el apartado Gestionar valores null en acciones de tabla de BD 324.

4. Haga clic en Aceptar para cerrar el cuadro de diálogo. Observe que en la asignación el botón **Acción: Insertar** A:In ahora se ha transformado en el botón **Acción: Actualizar** A:Up para indicar que se definió una acción de actualización para esta tabla.

Paso nº5: vista previa de la asignación y actualizar la base de datos

Abra el panel *Resultados* para consultar una vista previa de los resultados de la asignación. La asignación genera un script SQL que contiene las acciones que se deben ejecutar en la base de datos. El script no ha modificado todavía la base de datos. Solamente se trata de una vista previa.

```
1   /*
2   Las instrucciones SQL que aparecen a continuación se presentan únicamente de forma preliminar
    y no se pueden ejecutar en otra herramienta de consultas SQL.
3   Para ejecutar estas instrucciones, utilice la función "Ejecutar script SQL" del menú "Resultados".
4   Conéctese a la base de datos usando esta cadena de conexión:
5   Data Source=S:\Documentation\Public\ExampleFiles\EN\MapForce\DB Update
    if\altova.mdb;Provider=Microsoft.Jet.OLEDB.4.0
6   */
7
8   UPDATE [Person] SET [ForeignKey] = 1, [EMail] = 'A.Aldrich@microtech.com', [First] = 'Albert', [Last] =
     'Aldrich', [PhoneExt] = 582, [Title] = 'Manager' WHERE ([Person].[PrimaryKey] = 1)
9
10  UPDATE [Person] SET [ForeignKey] = 1, [EMail] = 'b.bander@microtech.com', [First] = 'Bert', [Last] =
    'Bander', [PhoneExt] = 471, [Title] = 'Accounts Receivable' WHERE ([Person].[PrimaryKey] = 2)
11
12  UPDATE [Person] SET [ForeignKey] = 1, [EMail] = 'c.clovis@microtech.com', [First] = 'Clive', [Last] =
    'Clovis', [PhoneExt] = 963, [Title] = 'Accounting Manager' WHERE ([Person].[PrimaryKey] = 3)
13
14  UPDATE [Person] SET [ForeignKey] = 2, [EMail] = 'd.Durnell@microtech.com', [First] = 'Dave', [Last] =
    'Durnelll', [PhoneExt] = 621, [Title] = 'Marketing Manager Europe' WHERE ([Person].[PrimaryKey] = 4)
15
16  UPDATE [Person] SET [ForeignKey] = 2, [EMail] = 'e.ellas@microtech.com', [First] = 'Eve', [Last] =
    'Ellas', [PhoneExt] = 753, [Title] = 'Art Director' WHERE ([Person].[PrimaryKey] = 5)
17
18  UPDATE [Person] SET [ForeignKey] = 3, [EMail] = 'f.fortunas@microtech.com', [First] = 'Fred', [Last] =
    'Fortunas', [PhoneExt] = 951, [Title] = 'Program Manager' WHERE ([Person].[PrimaryKey] = 6)
19
20  UPDATE [Person] SET [ForeignKey] = 3, [EMail] = 'g.gundall@microtech.com', [First] = 'Gerry', [Last] =
     'Gundall', [PhoneExt] = 654, [Title] = 'Software Engineer' WHERE ([Person].[PrimaryKey] = 7)
```

Para ejecutar el script en la base de datos:

- En el menú **Resultados** haga clic en **Ejecutar script SQL**.

Nota: el script SQL se puede ejecutar desde MapForce directamente pero hay otras maneras de actualizar la base de datos (*véase* *Ejecutar asignaciones que modifican bases de datos* 273).

4.2.3.4 Acción "Actualizar si... Insertar restantes"

A veces no solo es necesario actualizar registros de una tabla sino insertar registros nuevos en la misma tabla. En casos así podemos usar la acción **Actualizar si... Insertar restantes** de MapForce, que funciona de la siguiente manera:

- Si se cumple la condición **Actualizar si**, el registro de la base de datos se actualiza con los datos de la asignación.
- Si no se cumple la condición **Actualizar si** y existe una condición **Insertar restantes**, se inserta un registro nuevo.
- Si en la base de datos existen registros que no tienen equivalentes en el archivo de origen, estos registros no se modifican.

Nota sobre MySQL ODBC

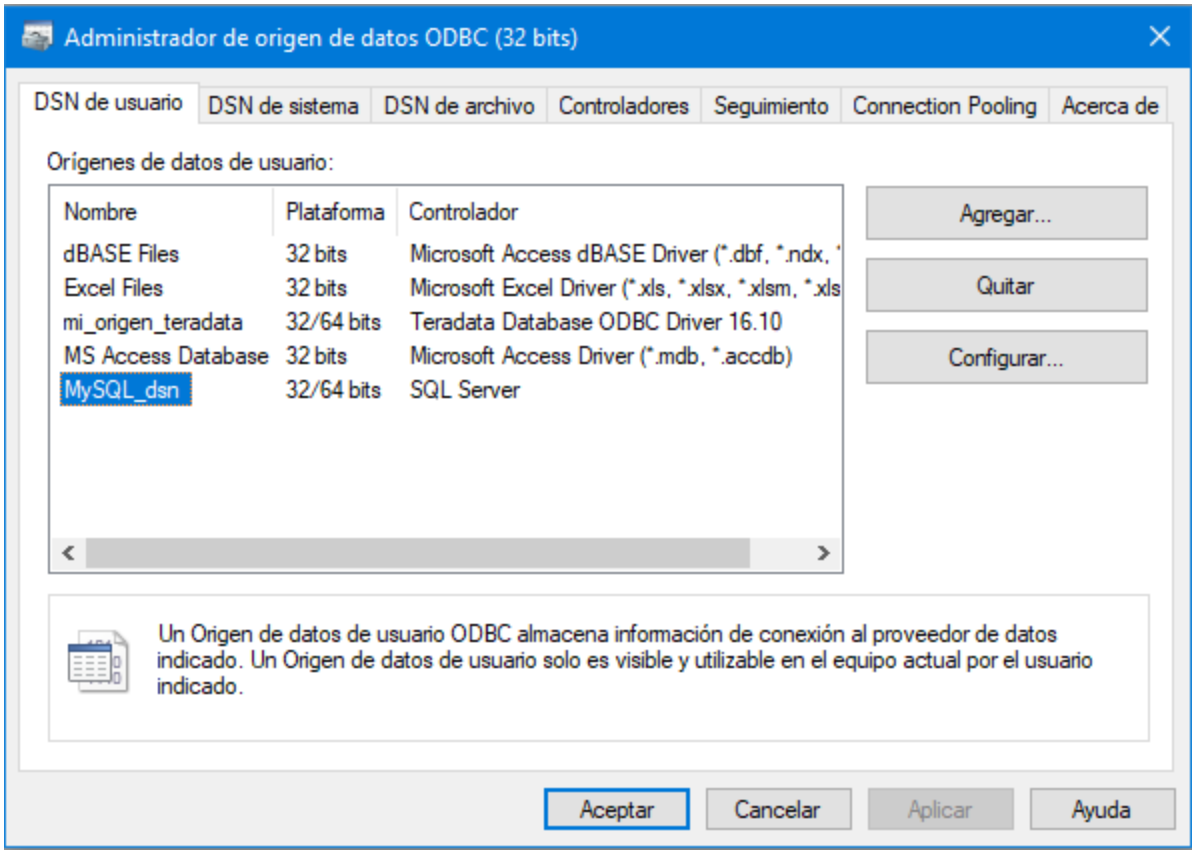
Si la base de datos de destino es MySQL con conexión por ODBC, deberá habilitar la opción *Return*

 © 2016-2022 Altova GmbH

matched rows instead of affected rows (Devolver filas coincidentes en lugar de filas afectadas) de la pestaña **Cursor/Results** del conector MySQL ODBC. Si lo prefiere e introduce la cadena de conexión a mano en el asistente de conexión para bases de datos, añada `Option=2` a la cadena de conexión (p. ej. `Dsn=mydsn;Option=2;`)

Para habilitar esta opción desde el conector MySQL ODBC:

1. Pulse la tecla **Windows** y teclee "ODBC".
2. Ejecute el administrador de fuentes de datos ODBC (versión de 32 bits o de 64 bits, dependiendo de la plataforma del conector MySQL ODBC instalado).
3. Haga clic en el nombre de origen de datos (DSN) utilizado por la asignación de MapForce y después haga clic en el botón **Configure** para modificar la configuración.



4. Haga clic en **Details >>** para acceder a las opciones avanzadas de configuración.
5. Haga clic en la pestaña **Cursors/Results** y después marque la casilla *Return matched rows instead of affected rows*.

Ejemplo

El siguiente ejemplo explica cómo combinar (actualizar e insertar) datos de un XML de origen en una tabla de una base de datos. Para ello se usan estos archivos:

- **altova-cmpy-extra.xml**, que contiene los datos de origen que se deben insertar en la base de datos.
- **Altova_Hierarchical.xsd**, que es el esquema utilizado para validar el archivo de instancia anterior.

- **altova.mdb**, que es la base de datos de destino que se debe actualizar.

Todos estos archivos están en la carpeta **<Documentos>\Altova\MapForce2023\MapForceExamples** y, para simplificar, en las instrucciones omitimos su ruta de acceso.

La asignación de este ejemplo modifica un archivo de base de datos de ejemplo. Lo más recomendable es crear una copia de seguridad de la base de datos original y empezar con una copia nueva antes de seguir las instrucciones que aparecen más abajo. Así se evitarán posibles modificaciones en los ejemplos originales y obtendrá el mismo resultado que en este ejemplo. Para más información consulte Ejecutar asignaciones que modifican bases de datos 273.

El objetivo de la asignación que nos ocupa es combinar todos los registros de un documento XML de origen en una tabla de destino llamada "Person". Es decir, por cada registro de los datos XML de origen, la asignación deberá hacer lo siguiente:

- Si el campo `PrimaryKey` de la persona en el archivo XML se corresponde con el campo `PrimaryKey` de la persona en la base de datos, la asignación deberá actualizar el registro.
- Los registros de la tabla "Person" que no cumplan esta condición no se modificarán.
- Si el campo `PrimaryKey` de la persona en el archivo XML no tiene equivalentes en la tabla de base de datos de destino, la asignación añadirá un registro nuevo en la tabla.

Para conseguir este objetivo debemos seguir estos pasos:

Paso nº1: insertar el componente XML de origen

- En el menú **Insertar** haga clic en el comando **Archivo o esquema XML** y navegue hasta el esquema **Altova_Hierarchical.xsd**. Cuando la aplicación solicite un archivo de instancia, navegue hasta el archivo **altova-cmpy-extra.xml**.

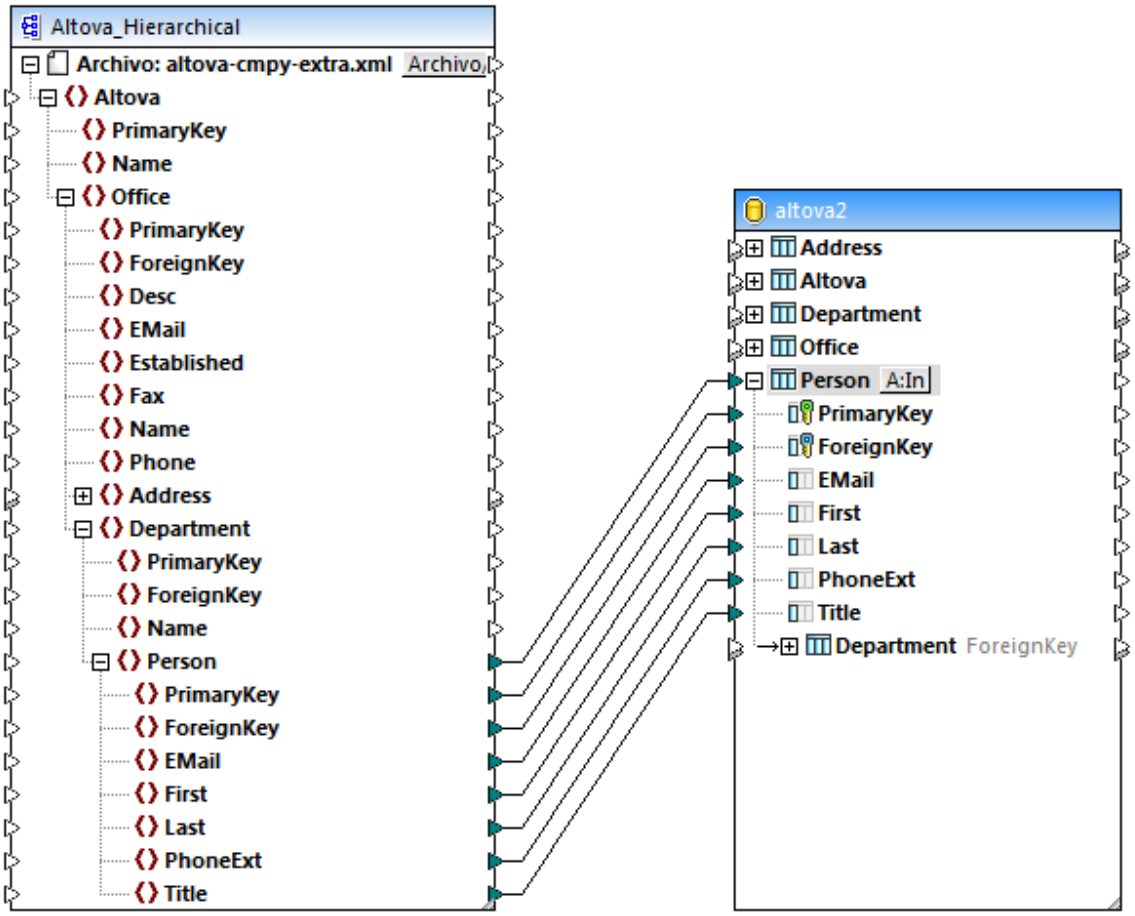
Paso nº2: insertar la base de datos de destino

- En el menú **Insertar** haga clic en el comando **Base de datos** y siga los pasos del asistente para conectarse a **altova.mdb** (*véase* *Ejemplo: agregar la base de datos "altova.mdb" a la asignación* 249).

Paso nº3: dibujar las conexiones

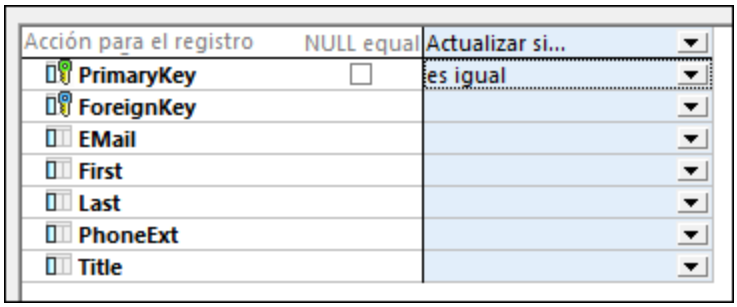
- Dibuje las conexiones de asignación de datos que aparecen en la imagen.

 © 2016-2022 Altova GmbH

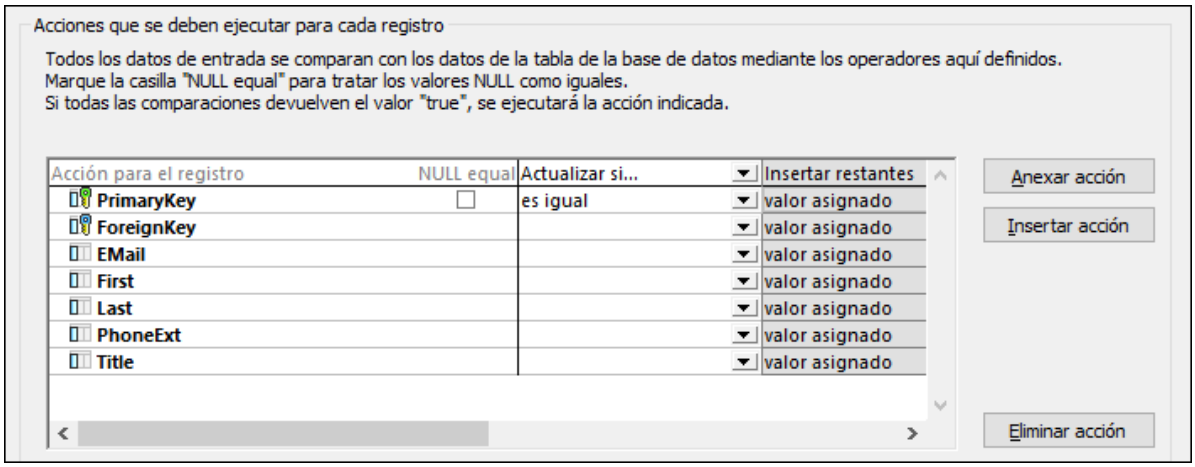


Paso nº4: configurar las acciones "Actualizar si... Insertar restantes"

1. En el componente de destino haga clic en el botón **Acción: Insertar** A:In situado junto a la tabla "Person".
2. Junto al campo *Acción para el registro* seleccione **Actualizar si...**, lo cual cambia la acción de tabla de base de datos por una acción de actualización condicional. Es decir, el registro actual solamente se actualizará si se cumple una condición concreta (paso siguiente).
3. Junto al campo **`PrimaryKey`**, seleccione el valor es igual (*imagen siguiente*). Esto define la condición de actualización: el registro de la base de datos solamente se actualizará si su valor **`PrimaryKey`** es igual al valor **`PrimaryKey`** procedente de la asignación.



4. Puede optar por marcar la casilla *Null equal*. En este ejemplo esta casilla es irrelevante, ya que ni el campo *ClavePrimaria* del XML de origen ni el campo *ClavePrimaria* de la BD contienen valores `null`. Sin embargo, debería marcar eta casilla si sus datos contienen valores `null` y quiere que estos sean tratados como iguales; de lo contrario puede que no obtenga los resultados deseados. Para más información consulte el apartado Gestionar valores null en acciones de tabla de BD 324.
5. Haga clic en el botón **Anexar acción**. Esto añade una acción nueva a la derecha de la acción **Actualizar si...** actual. Configure esta acción nueva como **Insertar restantes**.



En la imagen anterior, las acciones de tabla de base de datos están configuradas de acuerdo con los objetivos de esta asignación. Es decir, el registro solamente se actualizará si se cumple la condición **Actualizar si**. De lo contrario, el registro se insertará. La opción valor asignado especifica que los valores de la asignación se usarán para rellenar todos los campos del registro.

También se pueden definir más de dos acciones en la misma tabla de base de datos (aunque en nuestro ejemplo esto no es necesario). Cuando se ejecute la asignación, las acciones se ejecutarán de izquierda a derecha. Se entiende que la última acción **Insertar** es la última acción y las acciones que se añadan después se pasarán por alto.

Tenga en cuenta que el botón **Anexar acción** del cuadro de diálogo añade una acción nueva *después* de la acción seleccionada. El botón **Insertar acción** añade una acción nueva *antes* de la acción seleccionada. Para eliminar una acción haga clic en ella y después en el botón **Eliminar acción**.

6. Haga clic en **Aceptar** para cerrar el cuadro de diálogo. Observe cómo, en la asignación, el botón **Acción:Insertar** A:In se ha convertido en **Acción: Actualizar; Insertar** A:Up,In. Esto indica que se configuró una acción de actualización y de inserción para esta tabla.

Paso nº5: vista previa de la asignación y actualizar la base de datos

Abra el panel *Resultados* para consultar una vista previa de los resultados de la asignación. La asignación genera un script SQL que contiene las acciones que se deben ejecutar en la base de datos. El script no ha modificado todavía la base de datos. Solamente se trata de una vista previa.

```
1   /*
2   Las instrucciones SQL que aparecen a continuación se presentan únicamente de forma preliminar y no
    se pueden ejecutar en otra herramienta de consultas SQL.
3   Para ejecutar estas instrucciones, utilice la función "Ejecutar script SQL" del menú "Resultados".
4   Conéctese a la base de datos usando esta cadena de conexión:
5   Data Source=S:\ExampleFiles\EN\MapForce\DB Update if... Insert
    Rest\altova.mdb;Provider=Microsoft.Jet.OLEDB.4.0
6   */
7
8   UPDATE [Person] SET [ForeignKey] = 1, [EMail] = 'A.Aldrich@microtech.com', [First] = 'Albert', [Last] =
    'Aldrich', [PhoneExt] = 582, [Title] = 'Manager' WHERE ([Person].[PrimaryKey] = 1)
9
10  UPDATE [Person] SET [ForeignKey] = 1, [EMail] = 'b.bander@microtech.com', [First] = 'Bert', [Last] =
    'Bander', [PhoneExt] = 471, [Title] = 'Accounts Receivable' WHERE ([Person].[PrimaryKey] = 2)
11
12  UPDATE [Person] SET [ForeignKey] = 1, [EMail] = 'c.clovis@microtech.com', [First] = 'Clive', [Last] = 'Clovis'
    , [PhoneExt] = 963, [Title] = 'Accounting Manager' WHERE ([Person].[PrimaryKey] = 3)
13
14  UPDATE [Person] SET [ForeignKey] = 1, [EMail] = 'c.Cicada@microtech.com', [First] = 'Camilla ', [Last] =
    'Cicada', [PhoneExt] = 765, [Title] = 'HR' WHERE ([Person].[PrimaryKey] = 30)
15
16  UPDATE [Person] SET [ForeignKey] = 1, [EMail] = 'c.corrigan@microtech.com', [First] = 'Carol', [Last] =
    'Corrigan', [PhoneExt] = 629, [Title] = 'Admin' WHERE ([Person].[PrimaryKey] = 31)
17
18  UPDATE [Person] SET [ForeignKey] = 2, [EMail] = 'd.Durnell@microtech.com', [First] = 'Dave', [Last] =
    'DurnellI', [PhoneExt] = 621, [Title] = 'Marketing Manager Europe' WHERE ([Person].[PrimaryKey] = 4)
19
20  UPDATE [Person] SET [ForeignKey] = 2, [EMail] = 'e.ellas@microtech.com', [First] = 'Eve', [Last] = 'Ellas',
    [PhoneExt] = 753, [Title] = 'Art Director' WHERE ([Person].[PrimaryKey] = 5)
21
22  UPDATE [Person] SET [ForeignKey] = 4, [EMail] = 'n.newbury@microtech.com', [First] = 'Nira', [Last] =
    'Newbury', [PhoneExt] = 241, [Title] = 'Support Engineer' WHERE ([Person].[PrimaryKey] = 14)
23
24  UPDATE [Person] SET [ForeignKey] = 4, [EMail] = 'o.origone@microtech.com', [First] = 'Olanda', [Last] =
    'Origone', [PhoneExt] = 345, [Title] = 'Support Engineer' WHERE ([Person].[PrimaryKey] = 15)
```

Script SQL (vista parcial) antes de actualizarse la base de datos

Puede que en el script de vista previa no aparezca ninguna instrucción INSERT. Esto es totalmente normal y se debe a que los registros se insertan de forma condicional y a que las instrucciones INSERT dependen del resultado de la acción **Actualizar si** (lo cual no ocurre hasta que se ejecuta la asignación).

Nota: para determinadas bases de datos MapForce crea instrucciones de combinación MERGE en lugar de instrucciones UPDATE. Para más información consulte el apartado Instrucciones de combinación MERGE 301.

Para ejecutar el script en la base de datos:

- En el menú **Resultados** haga clic en **Ejecutar script SQL**.

Una vez ejecutada la asignación de datos y aplicado el script a la base de datos, podemos observar cómo las instrucciones INSERT ahora aparecen en el panel *Resultados*.

```
1   /*
2   Las instrucciones SQL que aparecen a continuación se ejecutaron durante la función "Generar
    resultados".
3   Cada resultado aparece escrito a la derecha de la cadena "-->>>".
4   Estas instrucciones se presentan únicamente de forma preliminar y no se pueden ejecutar en otra
    herramienta de consultas SQL.
5   La base de datos se conectó usando esta cadena de conexión:
6   Data Source=S:\ExampleFiles\EN\MapForce\DB Update if... Insert
    Rest\altova.mdb;Provider=Microsoft.Jet.OLEDB.4.0
7   */
8
9   UPDATE [Person] SET [ForeignKey] = 1, [EMail] = 'A.Aldrich@microtech.com', [First] = 'Albert', [Last] =
    'Aldrich', [PhoneExt] = 582, [Title] = 'Manager' WHERE ([Person].[PrimaryKey] = 1)
10  -->>> OK. 1 fila(s).
11
12  UPDATE [Person] SET [ForeignKey] = 1, [EMail] = 'b.bander@microtech.com', [First] = 'Bert', [Last] =
    'Bander', [PhoneExt] = 471, [Title] = 'Accounts Receivable' WHERE ([Person].[PrimaryKey] = 2)
13  -->>> OK. 1 fila(s).
14
15  UPDATE [Person] SET [ForeignKey] = 1, [EMail] = 'c.clovis@microtech.com', [First] = 'Clive', [Last] = 'Clovis'
    , [PhoneExt] = 963, [Title] = 'Accounting Manager' WHERE ([Person].[PrimaryKey] = 3)
16  -->>> OK. 1 fila(s).
17
18  UPDATE [Person] SET [ForeignKey] = 1, [EMail] = 'c.Cicada@microtech.com', [First] = 'Camilla ', [Last] =
    'Cicada', [PhoneExt] = 765, [Title] = 'HR' WHERE ([Person].[PrimaryKey] = 30)
19  -->>> OK. 0 fila(s).
20
21  INSERT INTO [Person] ([PrimaryKey], [ForeignKey], [EMail], [First], [Last], [PhoneExt], [Title]) VALUES (30, 1,
    'c.Cicada@microtech.com', 'Camilla ', 'Cicada', 765, 'HR')
22  -->>> OK. 1 fila(s).
23
24  UPDATE [Person] SET [ForeignKey] = 1, [EMail] = 'c.corrigan@microtech.com', [First] = 'Carol', [Last] =
    'Corrigan', [PhoneExt] = 629, [Title] = 'Admin' WHERE ([Person].[PrimaryKey] = 31)
25  -->>> OK. 0 fila(s).
26
27  INSERT INTO [Person] ([PrimaryKey], [ForeignKey], [EMail], [First], [Last], [PhoneExt], [Title]) VALUES (31, 1,
    'c.corrigan@microtech.com', 'Carol', 'Corrigan', 629, 'Admin')
```

Script SQL (vista parcial) después de actualizarse la base de datos

Nota: el script SQL se puede ejecutar desde MapForce directamente pero hay otras maneras de actualizar la base de datos (*véase* *Ejecutar asignaciones que modifican bases de datos* 273).

Si abrimos la tabla "Person" en el panel *Consulta de BD* de MapForce (*véase* *Examinar y consultar bases de datos* 380), podrá ver el resultado de la asignación:

- Se actualizaron todos los registros de la base de datos que tenían claves principales equivalentes en el archivo XML (p. ej. los registros con clave principal 1, 2, 3, 4 y 5).
- No se modificaron los registros de la base de datos que no tenían claves equivalentes en el archivo XML (p. ej. los registros con clave principal 6, 7, 8 y 9).
- Se insertaron registros nuevos en la tabla "Person", cuando la clave no existía en la base de datos (p. ej. los registros con clave principal 30 y 31).

 © 2016-2022 Altova GmbH

	PrimaryKey	ForeignKey	EMail	First	Last	PhoneExt	Title
1	1	1	A.Aldrich@microtech.com	Albert	Aldrich	582	Manager
2	2	1	b.bander@microtech.com	Bert	Bander	471	Accounts Receivable
3	3	1	c.clovis@microtech.com	Clive	Clovis	963	Accounting Manager
4	4	2	d.Durnell@microtech.com	Dave	Durnelll	621	Marketing Manager Europe
5	5	2	e.ellas@microtech.com	Eve	Ellas	753	Art Director
6	6	3	f.landis@nanonull.com	Fred	Landis	951	Program Manager
7	7	3	m.landis@nanonull.com	Michelle	Butler	654	Software Engineer
8	8	3	t.little@nanonull.com	Ted	Little	852	Software Engineer
9	9	3	a.way@nanonull.com	Ann	Way	951	Technical Writer
10	10	3	l.gardner@nanonull.com	Liz	Gardner	753	Software Engineer
11	11	3	p.smith@nanonull.com	Paul	Smith	334	Software Engineer
12	12	4	a.martin@nanonull.com	Alex	Martin	778	IT Manager
13	13	4	g.hammer@nanonull.com	George	Hammer	223	Web Developer
14	14	4	n.newbury@microtech.com	Nira	Newbury	241	Support Engineer
15	15	4	o.origone@microtech.com	Olanda	Origone	345	Support Engineer
16	16	5	s.meier@microtech.com	Paul	Ponetti	114	Office Manager
17	17	5	t.bone@microtech.com	Quang	Qui	331	Accounts Receivable
18	18	6	m.nafta@microtech.com	Rita	Rotter	122	PR & Marketing Manager US
19	19	7	v.bass@microtech.com	Silla	Semua	716	IT Manager
20	20	7	c.franken@microtech.com	Tom	Tinker	147	Support Engineer
21	21	7	m.redgreen@microtech.com	Uri	Ugoner	152	Support Engineer
22	30	1	c.Cicada@microtech.com	Camilla	Cicada	765	HR
23	31	1	c.corrigan@microtech.com	Carol	Corrigan	629	Admin

Tabla "Person" después de actualizarse la base de datos

4.2.3.5 Instrucciones de combinación MERGE

En algunas asignaciones de datos que actualizan e insertan datos en una tabla de base de datos (*véase* *Acción "Actualizar si... Insertar restantes"* 294). MapForce genera instrucciones de combinación MERGE que se ejecutarán en la base de datos cuando se ejecute la asignación. El motor de ejecución no tiene por qué ser MapForce (*véase* *Ejecutar asignaciones que modifican BD* 273).

Las instrucciones de combinación MERGE son compatibles con estas bases de datos:

- SQL Server 2008 y superior
- Oracle
- DB2
- Firebird

Las instrucciones MERGE reducen el número de llamadas al servidor de la base de datos porque compaginan instrucciones INSERT y UDPATE en una sola instrucción. Además, en el caso de las instrucciones MERGE la comprobación de coherencia la lleva a cabo la propia base de datos.

Para comprobar si la asignación ejecutará instrucciones MERGE en tiempo de ejecución (en lugar de instrucciones INSERT y UPDATE):

1. Cree una asignación que utilice acciones **Actualizar si...** y acciones **Insertar restantes** (*véase* *Acción "Actualizar si... Insertar restantes"* 294).
2. Consulte la vista previa de resultados de la asignación (abriendo el panel *Resultados*).

Si la base de datos admite instrucciones MERGE, el script SQL que genera MapForce incluirá instrucciones MERGE. Por ejemplo:

```
1   /*
2   Las instrucciones SQL que aparecen a continuación se presentan únicamente de forma preliminar y no se pueden
    ejecutar en otra herramienta de consultas SQL.
3   Para ejecutar estas instrucciones, utilice la función "Ejecutar script SQL" del menú "Resultados".
4   Conéctese a la base de datos usando esta cadena de conexión:
5   Data Source=                          ;Initial
    Catalog=DB_MERGE;MultipleActiveResultSets=True;Password=•••••;Persist Security Info=True;User ID=
6   */
7
8   SET QUOTED_IDENTIFIER ON
9
10  UPDATE [dbo].[Users] SET [FirstName] = (CAST('Despine' AS nvarchar(50))), [LastName] = (CAST('Buttler' AS
    nvarchar(50))) WHERE ([dbo].[Users].[UserID] = 1)
11
12  SELECT [UserID] FROM [dbo].[Users] WHERE ([UserID] = 1)
13  -->>> %UserID1%
14
15  DELETE FROM [dbo].[Addresses] WHERE EXISTS(SELECT * FROM [dbo].[Users] WHERE [dbo].[Users].[UserID] = [dbo].
    [Addresses].[UserID] AND ([dbo].[Users].[UserID] = '%UserID1%'))
16
17  MERGE INTO [dbo].[Addresses] AS T USING ( VALUES ( '%UserID1%', 1, 1, (CAST('Home' AS nvarchar(20))),
    (CAST('Louisville' AS nvarchar(50))), (CAST('Elm Street' AS nvarchar(50))), (CAST('12' AS nvarchar(20))) ) ) AS
    S ( [UserID], [IsShipping], [IsBilling], [AddressType], [City], [Street], [Number] ) ON ( (S.[UserID] = T.
    [UserID]) ) WHEN MATCHED THEN UPDATE SET [IsShipping] = S.[IsShipping], [IsBilling] = S.[IsBilling],
    [AddressType] = S.[AddressType], [City] = S.[City], [Street] = S.[Street], [Number] = S.[Number] WHEN NOT
    MATCHED THEN INSERT ( [UserID], [IsShipping], [IsBilling], [AddressType], [City], [Street], [Number] ) VALUES
    ( S.[UserID], S.[IsShipping], S.[IsBilling], S.[AddressType], S.[City], S.[Street], S.[Number] );
18
```

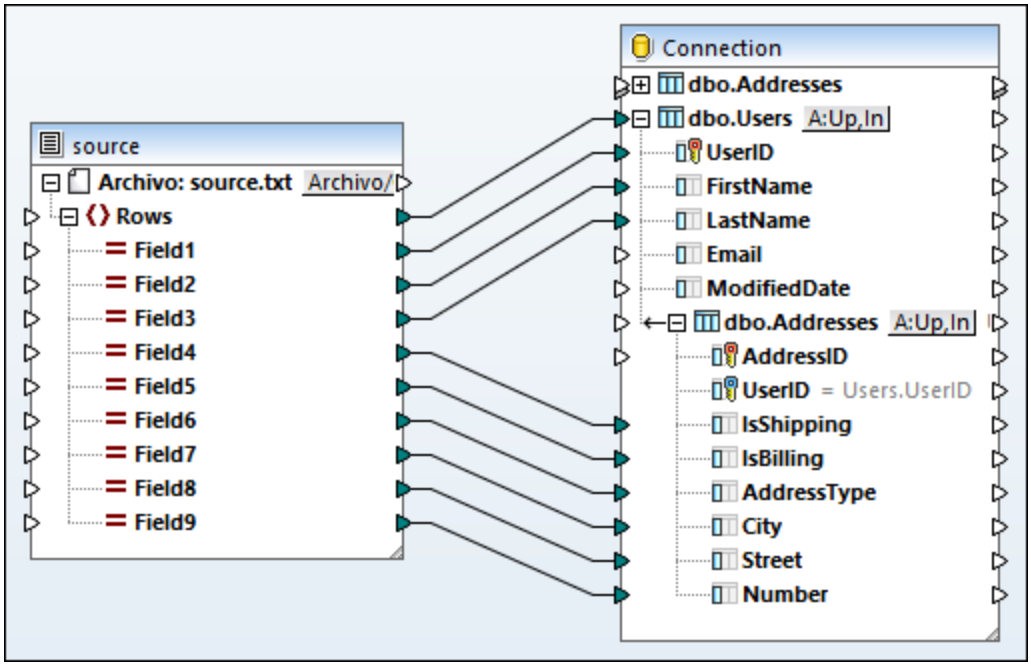
Asignación | Consulta de la BD | **Resultados**

Si la base de datos no admite instrucciones MERGE, el script SQL generado incluirá instrucciones UPDATE solamente (la vista previa no incluirá instrucciones INSERT porque estas instrucciones solamente se ejecutan si no se cumple la condición **Actualizar si...** (y esto no se sabe antes de ejecutar la asignación).

Notas

- MapForce crea instrucciones MERGE automáticamente cuando detecta un tipo de base de datos compatible; es decir, se trata de un proceso que no se puede alterar de forma manual.
- Si está actualizando varias tablas entre las que existen relaciones primaria-secundaria, entonces solo se crean instrucciones MERGE para las tablas "leaf". Una tabla "leaf" de la asignación que está anidado a mayor nivel. Por ejemplo, en la siguiente asignación las acciones Actualizar si... insertar restantes... se han definido tanto para la tabla `Users` como para la tabla `Addresses`. Sin embargo, las

instrucciones MERGE solo se generarán para la tabla "leaf", `Addresses`. Las instrucciones para la tabla primaria, `Users`, serán UPDATE o INSERT en lugar de MERGE.



- Con las instrucciones MERGE la opción Usar transferencia masiva (*véase* *Inserciones masivas (MapForce Server)* 320) solo es compatible con conexiones de BD ODBC y JDBC.

4.2.3.6 Opciones para tablas secundarias cuando se actualiza la tabla principal

Cuando la asignación actualiza una tabla que es tabla *principal* (es decir, una tabla que tiene relaciones de clave foránea con otras tablas), podrá configurar cómo ocuparse de los registros dependientes tanto en el origen de datos como en la tabla de destino. Por ejemplo, imaginemos que desea actualizar la tabla "Department" de la base de datos **altova.mdb**. Como cada persona está vinculada a un departamento por medio de una clave foránea, lo más probable es que desee ejecutar acciones en la tabla "Person" también (p. ej. inserciones, actualizaciones o eliminaciones). Esto le ayudaría a mantener la integridad de la base de datos y prevenir errores de asignación.

En este tema se describen las opciones disponibles para la tabla "Person" cuando se actualiza la tabla principal "Department" y para ello utilizamos estos archivos de ejemplo:

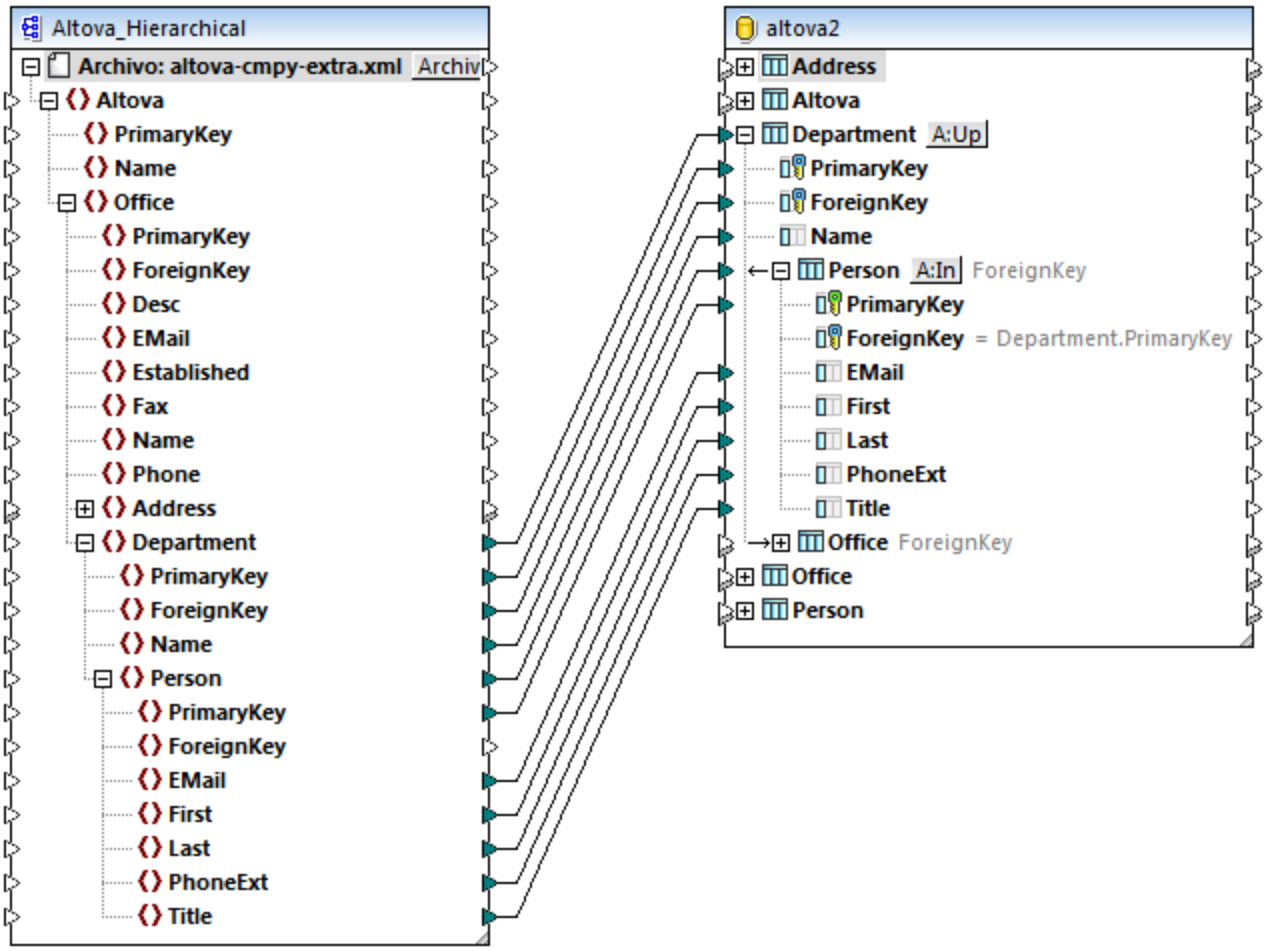
- **altova-cmpy-extra.xml**: contiene el origen de datos que se debe insertar en la base de datos.
- **Altova_Hierarchical.xsd**: el esquema utilizado para validar el archivo de instancia anterior.
- **altova.mdb**: la base de datos de destino que se debe actualizar.

Todos estos archivos están en la carpeta **<Documentos>\Altova\MapForce2023\MapForceExamples** y, para simplificar, en las instrucciones omitimos su ruta de acceso.

La asignación de este ejemplo modifica un archivo de base de datos de ejemplo. Lo más recomendable es

crear una copia de seguridad de la base de datos original y empezar con una copia nueva antes de seguir las instrucciones que aparecen más abajo. Así se evitarán posibles modificaciones en los ejemplos originales y obtendrá el mismo resultado que en este ejemplo. Para más información consulte Ejecutar asignaciones que modifican bases de datos 273.

En primer lugar, añada el esquema y la instancia XML de origen así como la base de datos de destino a la asignación de datos (*véase Ejemplo: agregar la base de datos "altova.mdb" a la asignación* 249). Siga los mismos pasos que aparecen en el apartado Acción "Actualizar si... Insertar restantes" 294. En segundo lugar, dibuje las conexiones que aparecen en esta imagen:

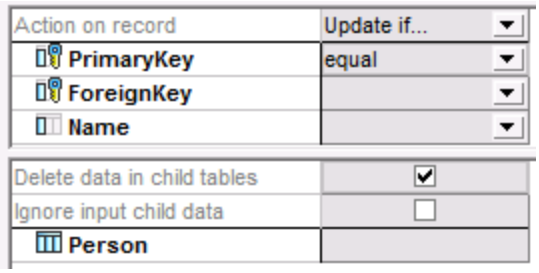
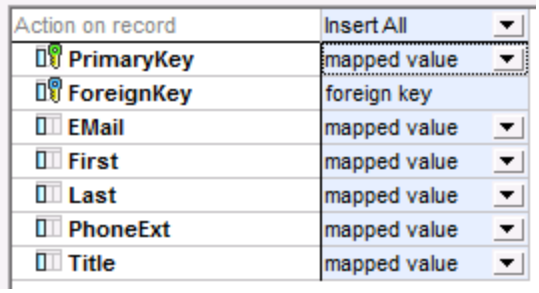


Como puede ver en la imagen, la asignación actualiza la tabla "Department" en la base de datos de destino. Esta tabla se eligió como tabla raíz. Para más información sobre las tablas raíz y por qué son necesarias consulte Gestionar relaciones de BD 258. La acción que debe ejecutarse en la tabla secundaria "Person" es el tema que nos ocupa en este apartado.

Las tablas que aparecen más abajo ilustran varias opciones de configuración y el resultado correspondiente. Estas opciones se pueden seleccionar en el cuadro de diálogo "Acciones de tabla de la base de datos" de la tabla principal "Department" y de la tabla secundaria "Person".

 © 2016-2022 Altova GmbH

Configuración A

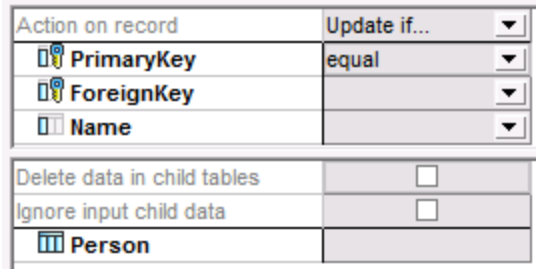
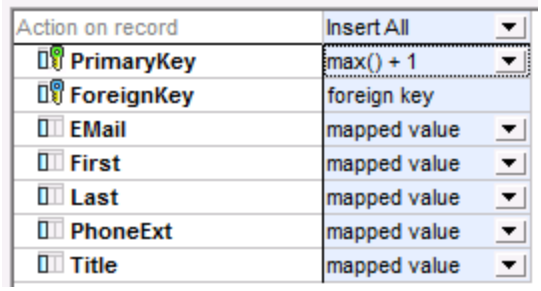
Opciones	Resultado de la asignación
 *Tabla "Department"*  *Tabla "Person"*	• Actualiza los registros de "Department" donde `PrimaryKey` corresponde en el XML de origen a `PrimaryKey` en la tabla de BD. • No actualiza los registros de "Department" que no tengan un equivalente en el archivo XML de entrada (dicha clave no existe en el origen). • Inserta, desde la instancia XML de entrada, todos los registros de "Person" que todavía no existen en la base de datos. • Elimina los datos secundarios (registros de "Person") de los registros de "Department" que cumplen con la condición **Actualizar si...** definida.

Configuración B

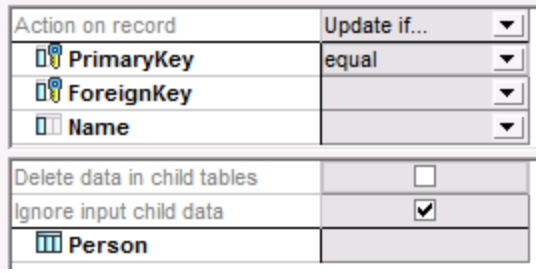
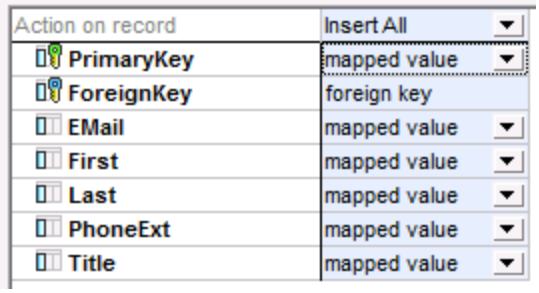
Opciones	Resultado de la asignación
Action on record: Update if... PrimaryKey: equal ForeignKey: Name: Delete data in child tables: ☐ Ignore input child data: ☐ Person *Tabla "Department"* Action on record: Insert All PrimaryKey: mapped value ForeignKey: foreign key EMail: mapped value First: mapped value Last: mapped value PhoneExt: mapped value Title: mapped value *Tabla "Person"*	La asignación falla y emite un mensaje de error de ejecución SQL. El motivo es que la asignación intenta insertar registros nuevos de "Person" con la misma clave principal que los registros de "Person" que ya existen. Si desea insertar registros desde el XML de entrada y añadirlos a los que ya existen en la base de datos, debe consultar la siguiente configuración.

© 2016-2022 Altova GmbH

Configuración C

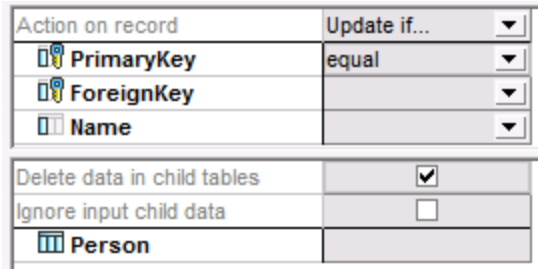
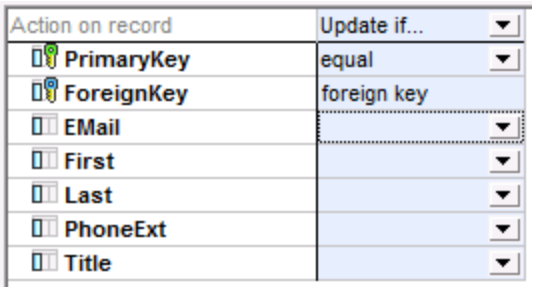
Opciones	Resultado de la asignación
 *Tabla "Department"*  *Tabla "Person"*	• Actualiza los registros de "Department" cuyo registro `PrimaryKey` corresponde en el XML de origen al registro `PrimaryKey` de la tabla de la BD. • No actualiza los registros de "Department" que no tengan equivalente en el XML de entrada (dicha clave no existe en el origen). • Se insertan registros nuevos de "Person" (con claves principales generadas automáticamente) en la tabla "Person".

Configuración D

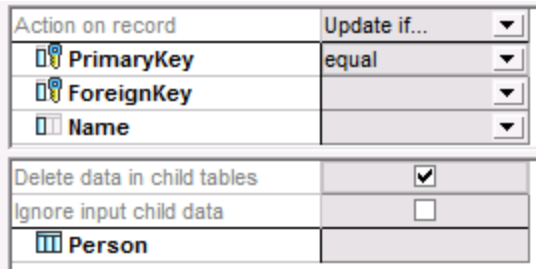
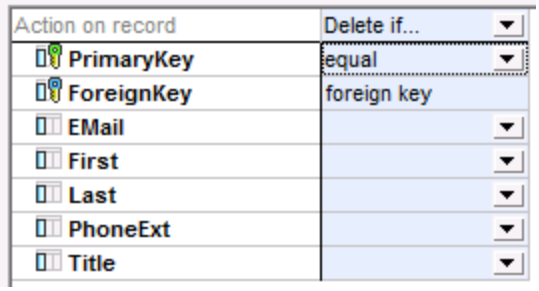
Opciones	Resultado de la asignación
 *Tabla "Department"*  *Tabla "Person"*	• Actualiza los registros de "Department" cuyo `PrimaryKey` corresponda en el XML de origen al `PrimaryKey` en la tabla de la BD. • No actualiza los registros de "Department" que no tengan equivalente en el XML de entrada (dicha clave no existe en el origen). • No se insertan registros en la tabla "Person" porque la opción *Omitir datos secundarios de entrada* está habilitada para la tabla "Departments" principal.

© 2016-2022 Altova GmbH

Configuración E

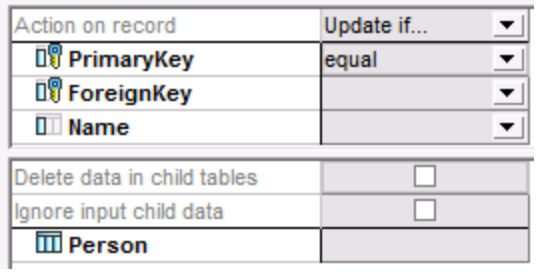
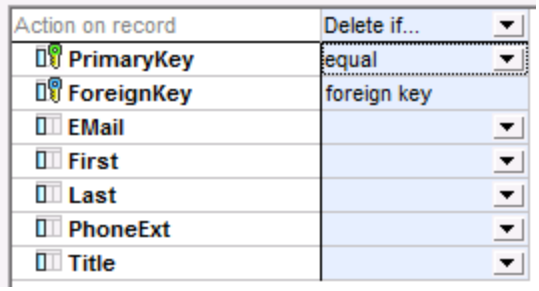
Opciones	Resultado de la asignación
 *Tabla "Department"*  *Tabla "Person"*	• Actualiza los registros de "Department" cuyo `PrimaryKey` corresponda en el XML de origen al `PrimaryKey` en la tabla de la BD. • No actualiza los registros de "Department" que no tengan equivalente en el XML de entrada (dicha clave no existe en el origen). • Elimina todos los registros de "Person" vinculados a un "Department" que tenga el correspondiente `PrimaryKey` en el XML de origen. El motivo es que la opción *Eliminar datos en tablas secundarias* está habilitada para la tabla principal "Department". • Se conservan en la base de datos los registros de "Person" vinculados a un departamento que no cumpla con la condición **Actualizar si...** definida. • No se actualiza ningún registro de la tabla "Person".

Configuración F

Opciones	Resultado de la asignación
 *Tabla "Department"*  *Tabla "Person"*	• Actualiza los registros de "Department" cuyo `PrimaryKey` corresponda en el XML de origen al `PrimaryKey` en la tabla de la BD. • No actualiza los registros de "Department" que no tengan equivalente en el XML de entrada (dicha clave no existe en el origen). • Elimina todos los registros de "Person" vinculados a un "Department" que tenga el correspondiente `PrimaryKey` en el XML de origen. El motivo es que la opción *Eliminar datos en tablas secundarias* está habilitada para la tabla principal "Department". • Se conservan en la base de datos los registros de "Person" vinculados a un departamento que no cumpla con la condición **Actualizar si...** definida.

© 2016-2022 Altova GmbH

Configuración G

Opciones	Resultado de la asignación
 *Tabla "Department"*  *Tabla "Person"*	• Actualiza los registros de "Department" cuyo `PrimaryKey` corresponda en el XML de origen al `PrimaryKey` en la tabla de la BD. • No actualiza los registros de "Department" que no tengan equivalente en el XML de entrada (dicha clave no existe en el origen). • Elimina todos los registros de "Person" que cumplan con estas dos condiciones: a. El registro de "Person" está vinculado a un "Department" que tiene un `PrimaryKey` correspondiente en el XML de origen y b. El registro "Person" tiene un `PrimaryKey` correspondiente en el XML de origen.

4.2.3.7 Acción "Eliminar si..."

La acción **Eliminar si...** se usa para borrar datos de una tabla de BD de forma condicional. Puede definir esta acción en el cuadro de diálogo "Acciones de tabla de la base de datos" (véase Configurar acciones de tabla de BD 327). Por ejemplo, al asignar datos desde un XML de origen a una BD de destino puede configurar una condición **Eliminar si...** para comprobar si un campo concreto del XML de origen es igual que un campo en el registro de la BD de destino (por lo general, un valor de clave principal). Si la condición **Eliminar si...** se cumple (es decir, si los dos campos son iguales), el registro de DB se eliminará cuando se ejecute la asignación.

Nota: la acción **Eliminar si...** no debe confundirse con la opción **Eliminar datos en tablas secundarias** del cuadro de diálogo "Acciones de tabla de la base de datos". La acción **Eliminar si...** solo afecta a la tabla para la que se ha definido la acción; el resto de tablas permanecen intactas.

Este ejemplo explica cómo eliminar datos de una tabla de BD de forma condicional, así como a insertar registros en esa misma tabla de BD basándose en el resultado de la condición para eliminar.

Este ejemplo usa los siguientes archivos:

- **altova-cmpy-extra.xml**: contiene los datos de origen que deben insertarse en la base de datos.
- **Altova_Hierarchical.xsd**: el esquema que usaremos para validar el archivo de instancia de la línea anterior.

- **altova.mdb**: la base de datos de destino que vamos a actualizar.

Todos los archivos se encuentran en la carpeta **<Documentos>\Altova\MapForce2023\MapForceExamples**. Para simplificar, más adelante omitiremos la ruta completa.

> La asignación de este ejemplo modifica un archivo de base de datos de ejemplo. Lo más recomendable es crear una copia de seguridad de la base de datos original y empezar con una copia nueva antes de seguir las instrucciones que aparecen más abajo. Así se evitarán posibles modificaciones en los ejemplos originales y obtendrá el mismo resultado que en este ejemplo. Para más información consulte Ejecutar asignaciones que modifican bases de datos 273.

El objetivo de la asignación es el siguiente:

- si existen registros "person" con el mismo `PrimaryKey` tanto en el XML de origen como en la tabla "Person" de destino, es necesario borrarlos de esta última.
- el resto de los registros del XML de origen deben insertarse en la tabla "Person".

Para alcanzar el objetivo de la asignación, seguiremos los pasos que se describen a continuación.

Paso 1: insertar el componente XML de origen

- En el menú **Insertar**, haga clic en **Archivo o esquema XML...** y busque el archivo **Altova_Hierarchical.xsd**. Cuando se le pida que indique un archivo de instancia, busque **altova-cmpy-extra.xml**.

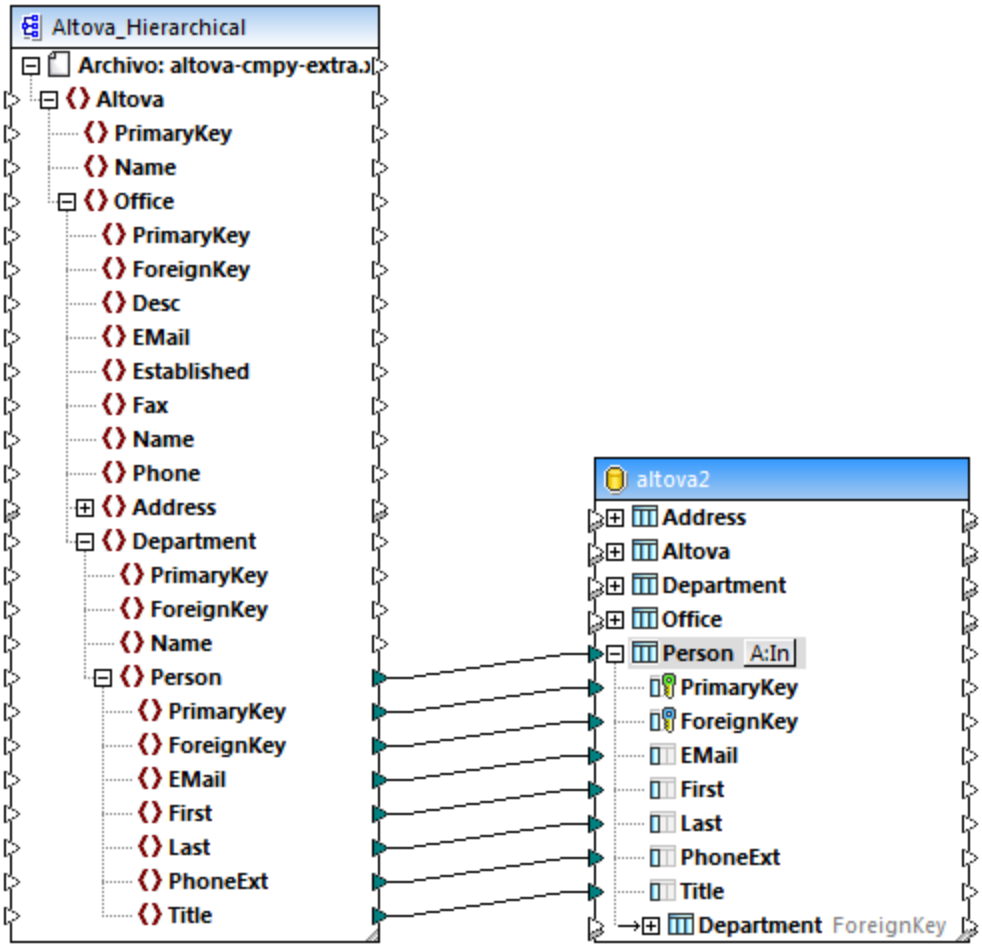
Paso 2: insertar la BD de destino

- En el menú **Insertar**, haga clic en **Base de datos** y siga los pasos del asistente para conectarse a **altova.mdb** (*véase* *Ejemplo: agregar la base de datos "altova.mdb" a la asignación* 249).

Paso 3: dibujar las conexiones

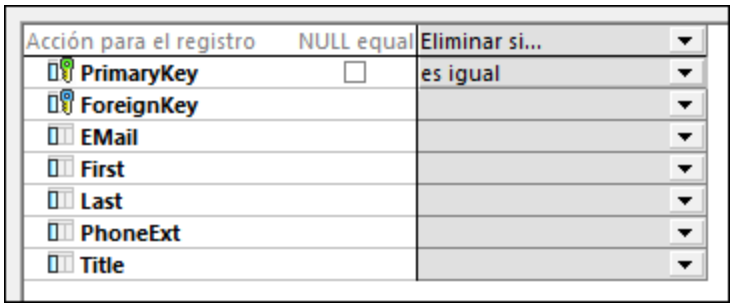
- Dibuje las conexiones de la asignación como se muestra a continuación.

 © 2016-2022 Altova GmbH

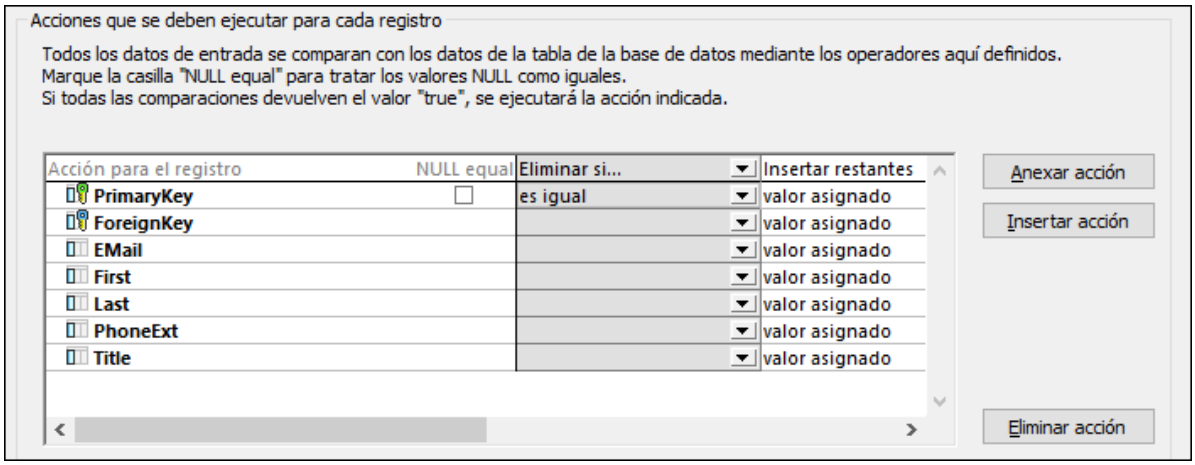


Paso 4: configurar las acciones "Eliminar si..." e "Insertar restantes"

1. En el componente de destino, haga clic en el botón **Acción: Insertar** (A:In), junto a la tabla "Person".
2. Junto a **Acción para el registro**, seleccione **Eliminar si...** . Esto cambia la acción de la tabla de la base de datos a una acción de eliminación condicional. Es decir, el registro actual solo se eliminará si se cumple una condición (*véase el siguiente paso*).
3. Junto a ClavePrincipal, seleccione el valor `es igual`, como se muestra más abajo. Esto define la condición de la actualización: el registro de BD se eliminará solo si el valor **ClavePrincipal** es igual al valor **ClavePrincipal** que proviene de la asignación.



4. Puede optar por marcar la casilla *Null equal*. En este ejemplo esta casilla es irrelevante, ya que ni el campo *ClavePrimaria* del XML de origen ni el campo *ClavePrimaria* de la BD contienen valores `null`. Sin embargo, debería marcar eta casilla si sus datos contienen valores `null` y quiere que estos sean tratados como iguales; de lo contrario puede que no obtenga los resultados deseados. Para más información consulte el apartado Gestionar valores null en acciones de tabla de BD 324.
5. Haga clic en **Anexar acción**. Esto añade una nueva acción a la derecha de la acción ya existente **Eliminar si**. Configure la nueva acción como **Insertar restantes**.



En la imagen anterior, las acciones de la tabla de la BD se han configurado conforme a los objetivos de la asignación. Es decir, que el registro solo se elimina si se cumple la condición asignada a la acción **Eliminar si...** ; de lo contrario, se inserta. La opción "valor asignado" indica que los valores de la asignación se usarán para rellenar todos los campos del registro.

6. Haga clic en **Aceptar** para cerrar el cuadro de diálogo. Observe que en la asignación al botón **Acción: Insertar** A:In ahora lo ha reemplazado a **Acción: Eliminar; Insertar** (A:De,In). Esto indica que para esta tabla hay configuradas una acción de eliminar y una de insertar.

Paso 5: previsualizar la asignación y actualizar la BD

Haga clic en la pestaña *Resultados* para previsualizar la asignación. Se genera un script SQL que contiene las acciones que se van a ejecutar en la BD. El script todavía no ha modificado la base de datos, se trata únicamente de una vista previa. Para ejecutar el script:

- en el menú **Resultados**, haga clic en **Ejecutar script SQL**.

 © 2016-2022 Altova GmbH

Nota: ejecutar el script SQL directamente desde MapForce es solo una de las formas que existen de actualizar la BD. Consulte también Ejecutar asignaciones que modifican BD 273.

Si abre la tabla "Person" en la pestaña *Consulta de la BD* de MapForce (*véase Examinar y consultar bases de datos* 380) puede ver el resultado de la asignación:

- todos los registros de la BD cuyas claves principales tienen correspondencia en el archivo XML se eliminan. Esto ocurre, por ejemplo, con los registros con claves principales 1, 2, 3, 4 y 5.
- todos los registros de BD que no tienen claves principales correspondientes en el archivo XML permanecen intactas. Esto ocurre, por ejemplo, con los registros con claves principales 6, 7, 8, 9, 10, 11, 12 y 13.
- se insertan nuevos registros en la tabla "Person" (en los casos en los que no existe ya una clave en la base de datos). Esto ocurre, por ejemplo, con los registros con claves principales 30 y 31.

	PrimaryKey	ForeignKey	EMail	First	Last	PhoneExt	Title
1	6	3	f.landis@nanonull.com	Fred	Landis	951	Program Manager
2	7	3	m.landis@nanonull.com	Michelle	Butler	654	Software Engineer
3	8	3	t.little@nanonull.com	Ted	Little	852	Software Engineer
4	9	3	a.way@nanonull.com	Ann	Way	951	Technical Writer
5	10	3	l.gardner@nanonull.com	Liz	Gardner	753	Software Engineer
6	11	3	p.smith@nanonull.com	Paul	Smith	334	Software Engineer
7	12	4	a.martin@nanonull.com	Alex	Martin	778	IT Manager
8	13	4	g.hammer@nanonull.com	George	Hammer	223	Web Developer
9	30	1	c.Cicada@microtech.com	Camilla	Cicada	765	HR
10	31	1	c.corrigan@microtech.com	Carol	Corrigan	629	Admin

La tabla "Person" después de actualizar la DB

4.2.3.8 Acción "Omitir si..."

La acción **Omitir si...** consiste en una condición que se define para evitar que se actualicen ciertos registros en una tabla de BD. La acción **Omitir si...** está pensada para que se use en combinación con otra acción de tabla de base de datos (como **Insertar restantes**). Por ejemplo, al asignar datos desde un XML de origen a una base de datos de destino puede configurar una condición **Omitir si...** para comprobar si un campo en concreto del XML de origen es igual que otro campo en el registro de BD de destino (por lo general una clave principal). Si se cumple la condición **Ignore if...** (es decir, si los dos campos son iguales) el registro de la BD se ignorará al ejecutar la asignación y se ejecutará la siguiente acción definida (por ejemplo, **Insertar restantes**).

Este ejemplo muestra cómo insertar registros en una tabla de BD basándose en el resultado de la condición **Omitir si...**, para lo que usaremos los siguientes archivos:

- **altova-cmpy-extra.xml**: contiene los datos de origen que deben insertarse en la base de datos.
- **Altova_Hierarchical.xsd**: el esquema que usaremos para validar el archivo de instancia de la línea anterior.
- **altova.mdb**: la base de datos de destino que vamos a actualizar.

Todos los archivos se encuentran en la carpeta **<Documentos>\Altova\MapForce2023\MapForceExamples**. Para simplificar, más adelante omitiremos la ruta completa.

La asignación de este ejemplo modifica un archivo de base de datos de ejemplo. Lo más recomendable es crear una copia de seguridad de la base de datos original y empezar con una copia nueva antes de seguir las instrucciones que aparecen más abajo. Así se evitarán posibles modificaciones en los ejemplos originales y obtendrá el mismo resultado que en este ejemplo. Para más información consulte Ejecutar asignaciones que modifican bases de datos 273.

El objetivo de la asignación es el siguiente:

- si existen registros "person" con el mismo `PrimaryKey` tanto en el XML de origen como en la tabla "Person" de destino, no se ejecuta ninguna acción que les afecte (es decir, se ignoran).
- si en la tabla "Person" existe algún registro que no cumpla la condición que hemos explicado, tampoco se ejecuta ninguna acción que les afecte.
- a los registros del XML de origen que no tengan equivalente (clave principal) en la tabla "Person" se les considera como nuevos y se insertan en la tabla "Person" con una clave principal.

Para alcanzar el objetivo de la asignación, seguiremos los pasos que se describen a continuación.

Paso 1: insertar el componente XML de origen

- En el menú **Insertar**, haga clic en **Archivo o esquema XML...** y busque el archivo **Altova_Hierarchical.xsd**. Cuando se le pida que indique un archivo de instancia, busque **altova-cmpy-extra.xml**.

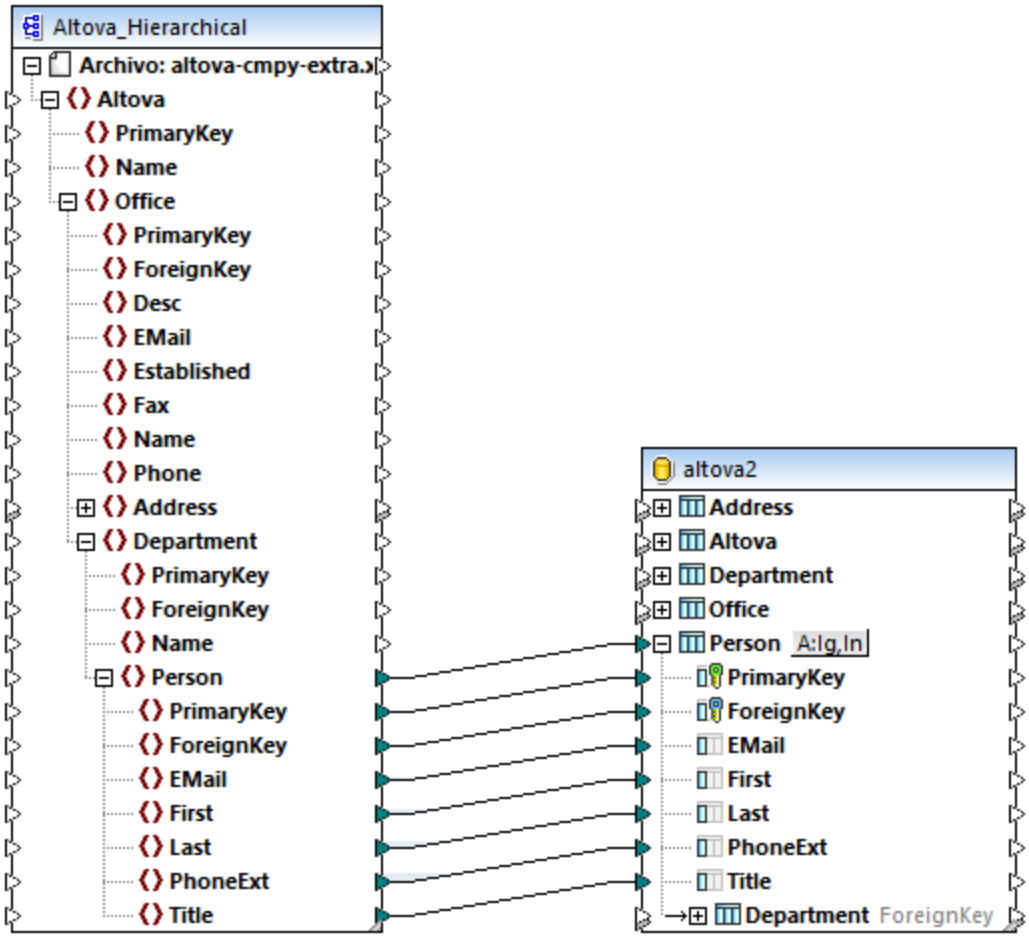
Paso 2: insertar la BD de destino

- En el menú **Insertar**, haga clic en **Base de datos** y siga los pasos del asistente para conectarse a **altova.mdb** (*véase* *Ejemplo: agregar la base de datos "altova.mdb" a la asignación* 249).

Paso 3: dibujar las conexiones

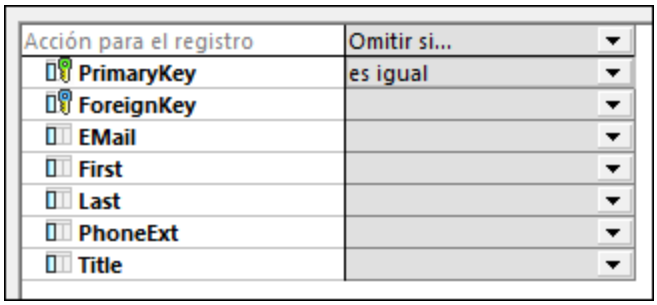
- Dibuje las conexiones de la asignación como se muestra a continuación.

 © 2016-2022 Altova GmbH

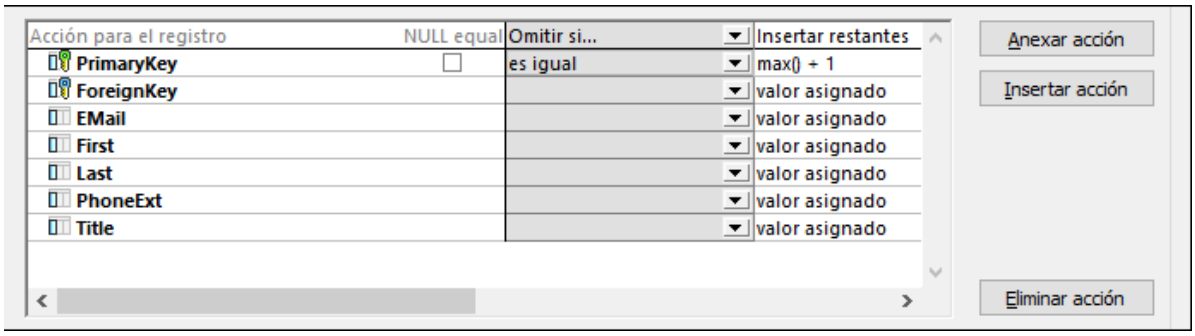


Paso 4: configurar las acciones "Omitir si..." e "Insertar restantes"

1. En el componente de destino, haga clic en el botón **Acción: Insertar** (A:In), junto a la tabla "Person".
2. Junto a **Acción para el registro**, seleccione **Omitir si...** . Esto cambiará la acción de la tabla a una acción condicional de ignorar. Es decir, el registro actual solo se omitirá si se cumple una condición (*véase el siguiente paso*).
3. Junto a **Clave principal**, seleccione el valor `es igual`, como se muestra más abajo. Esto define la condición de la acción de omitir: el registro de BD se omitirá solo si el valor **ClavePrincipal** es igual al valor **ClavePrincipal** que proviene de la asignación.



4. Puede optar por marcar la casilla *Null equal*. En este ejemplo esta casilla es irrelevante, ya que ni el campo *ClavePrimaria* del XML de origen ni el campo *ClavePrimaria* de la BD contienen valores `null`. Sin embargo, debería marcar eta casilla si sus datos contienen valores `null` y quiere que estos sean tratados como iguales; de lo contrario puede que no obtenga los resultados deseados. Para más información consulte el apartado Gestionar valores null en acciones de tabla de BD 324.
5. Haga clic en **Anexar acción**. Esto añade una nueva acción a la derecha de la acción ya existente **Omitir si**. Configure la nueva acción como **Insertar restantes**, con la clave principal en `max() + 1`, como se muestra a continuación:



En la imagen anterior las acciones de la BD se han configurado conforme a los objetivos de la asignación. Es decir, el registro solo se omite si se cumple la condición asignada a la acción **Omitir si...** ; de lo contrario, el registro se inserta.La opción `valor asignado` indica que los valores de la asignación se usarán para rellenar todos los campos del registro. La opción `max() + 1` genera una nueva clave principal única para el registro.

6. Haga clic en **Aceptar** para cerrar el cuadro de diálogo. Observe que en la asignación al botón **Acción: Insertar** A:In ahora lo ha reemplazado a **Acción: Omitir; Insertar**. Esto indica que para esta tabla hay configuradas una acción de omitir y una de insertar.

Paso 5: previsualizar la asignación y actualizar la BD

Haga clic en la pestaña *Resultados* para previsualizar la asignación. Se genera un script SQL que contiene las acciones que se van a ejecutar en la BD. El script todavía no ha modificado la base de datos, se trata únicamente de una vista previa. Para ejecutar el script:

- en el menú **Resultados**, haga clic en **Ejecutar script SQL**.

Nota: ejecutar el script SQL directamente desde MapForce es solo una de las formas que existen de actualizar la BD. Consulte también Ejecutar asignaciones que modifican BD 273.

Si abre la tabla "Person" en la pestaña *Consulta de la BD* de MapForce (*véase* *Examinar y consultar bases de datos* 380) puede ver el resultado de la asignación:

- todos los registros de la BD cuyas claves principales tienen correspondencia en el archivo XML cumplen con la condición de la acción **Omitir si...** , por lo que permanecen intactas. Esto ocurre, por ejemplo, con los registros con claves principales 1, 2, 3, 4 y 5.
- todos los registros de BD que no tienen claves principales correspondientes en el archivo XML no cumplen con la condición **Omitir si...** pero también permanecen intactas. Esto ocurre, por ejemplo, con los registros con claves principales 6, 7, 8, 9, 10, 11, 12 y 13.
- se insertan nuevos registros en la tabla "Person" (en los casos en los que no existe ya una clave en la base de datos). Esto ocurre, por ejemplo, con los registros del archivo XML de origen con claves principales 30 y 31. Estas se insertan en la BD con la respectivas claves principales nuevas 22 y 23.

	PrimaryKey	ForeignKey	EMail	First	Last	PhoneExt	Title
1	1	1	v.callaby@nanonull.com	Vernon	Callaby	582	Office Manager
2	2	1	f.further@nanonull.com	Frank	Further	471	Accounts Receivable
3	3	1	l.matise@nanonull.com	Loby	Matise	963	Accounting Manager
4	4	2	j.firstbread@nanonull.com	Joe	Firstbread	621	Marketing Manager Europe
5	5	2	s.sanna@nanonull.com	Susi	Sanna	753	Art Director
6	6	3	f.landis@nanonull.com	Fred	Landis	951	Program Manager
7	7	3	m.landis@nanonull.com	Michelle	Butler	654	Software Engineer
8	8	3	t.little@nanonull.com	Ted	Little	852	Software Engineer
9	9	3	a.way@nanonull.com	Ann	Way	951	Technical Writer
10	10	3	l.gardner@nanonull.com	Liz	Gardner	753	Software Engineer
11	11	3	p.smith@nanonull.com	Paul	Smith	334	Software Engineer
12	12	4	a.martin@nanonull.com	Alex	Martin	778	IT Manager
13	13	4	g.hammer@nanonull.com	George	Hammer	223	Web Developer
14	14	4	j.band@nanonull.com	Jessica	Bander	241	Support Engineer
15	15	4	l.king@nanonull.com	Lui	King	345	Support Engineer
16	16	5	s.meier@nanonull.com	Steve	Meier	114	Office Manager
17	17	5	t.bone@nanonull.com	Theo	Bone	331	Accounts Receivable
18	18	6	m.nafta@nanonull.com	Max	Nafta	122	PR & Marketing Manager US
19	19	7	v.bass@nanonull.com	Valentin	Bass	716	IT Manager
20	20	7	c.franken@nanonull.com	Carl	Franken	147	Support Engineer
21	21	7	m.redgreen@nanonull.com	Mark	Redgreen	152	Support Engineer
22	22	1	c.Cicada@microtech.com	Camilla	Cicada	765	HR
23	23	1	c.corrigan@microtech.com	Carol	Corrigan	629	Admin

La tabla "Person" tras actualizar la BD

4.2.3.9 Inserciones masivas

La opción *Usar transferencia masiva* sirve para insertar datos de un componente de MapForce (`txt`, `csv`, `dat`, etc.) en una BD a gran velocidad. Con esta opción puede acelerar significativamente el proceso de inserción de datos porque basta con ejecutar una sola instrucción en lugar de muchas.

La opción *Usar transferencia masiva* se puede habilitar en MapForce al diseñar la asignación, como mostramos más abajo. Una asignación que tenga habilitada esta opción se puede ejecutar en MapForce, pero la transferencia masiva no se aplica en esta fase, sino que ocurre cuando se ejecuta la asignación en MapForce Server.

Para poder usar transferencias masivas es necesario cumplir los siguientes requisitos:

- El lenguaje de transformación de la asignación es BUILT-IN. Para más información consulte Seleccionar el lenguaje de transformación 24.
- La ejecución la ejecuta MapForce Server (solo o gestionado por FlowForce Server). Esto significa que la asignación debe bien compilarse a formato .mfx o implementarse en FlowForce Server. Para más información consulte Compilar asignaciones en archivos de ejecución de MapForce Server 955 y Implementar asignaciones en FlowForce Server 958.
- La licencia de MapForce Server no se limita a la "ejecución por subprocesos simples" en un equipo multinúcleo. Es decir, la casilla ***Limit to single thread execution*** (limitar a ejecución de subprocesos) en la pestaña *Gestión de servidores* del servidor Altova LicenseServer debe estar inactiva.
- La acción de BD es "Insertar todo". Consulte también Configurar de acciones de tabla de BD 327.
- La tabla en la que se debe cargar la inserción masiva de datos debe ser una tabla *leaf*, es decir, una tabla que esté en el nivel más inferior de la jerarquía de la BD. En la asignación no debería haber tablas, vistas ni procedimientos almacenados relacionados que hagan referencia a la tabla.
- El controlador de la BD admiten el uso de inserciones masivas en condiciones WHERE.

La siguiente tabla es un resumen de en qué casos se pueden usar las inserciones masivas dependiendo del tipo de BD y del controlador.

	ADO	ODBC	JDBC	ADO.NET	Nativa
Access	No	No	n/a	n/a	n/a
DB2	No	Sí	Sí	Sí	n/a
Firebird	n/a	Sí	Sí	No	n/a
Informix	No	Sí	Sí	Sí	n/a
iSeries	No	Sí	Sí	Sí	n/a
MariaDB	No	Sí	Sí	No	n/a
MySQL	n/a	Sí*	Sí	No	n/a
Oracle	No	Sí	Sí	Sí	n/a
PostgreSQL	n/a	Sí	Sí	n/a	Sí
Progress	n/a	Sí	Sí	n/a	n/a

 © 2016-2022 Altova GmbH

	ADO	ODBC	JDBC	ADO.NET	Nativa
SQL Server	Sí	Sí	Sí	Sí	n/a
SQLite	n/a	n/a	n/a	n/a	No
Sybase	No	Sí	Sí	n/a	n/a
Teradata	n/a	Sí	Sí	n/a	n/a

** se necesita MySQL versión 5 o superior.*

Nota: para poder realizar inserciones masivas en MySQL y MariaDB con JDBC use la opción de conexión `rewriteBatchedStatements=true`.

Ejemplo de inserción masiva

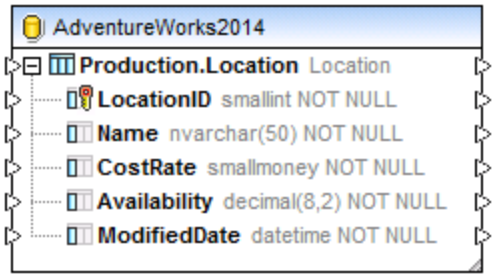
Este ejemplo explica cómo crear una asignación que cargue inserciones masivas de datos desde un archivo de muestra `source.txt` a una BD. El ejemplo usa SQL Server 2014 y la BD AdventureWorks 2014. Esta última se puede descargar en los ejemplos de AdventureWorks, en la página web de GitHub (https://github.com/Microsoft/sql-server-samples/releases/tag/adventureworks).

```
Location A,15.3,39
Location B,46,34
Location C,56.33,0
Location D,0,399
Location E,0,97.43
```

source.txt

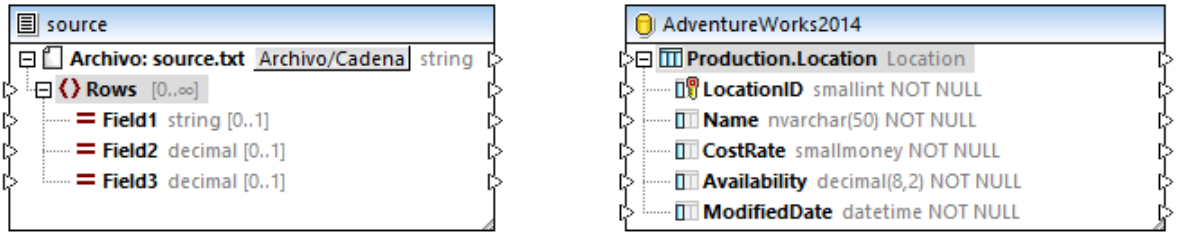
Para definir una inserción masiva:

1. Seleccione BUILT-IN ( ) como lenguaje de transformación.
2. Conéctese a la BD AdventureWorks 2014 y añada la tabla "Production.Location" a la asignación. Para más información consulte Agregar bases de datos a la asignación 247.

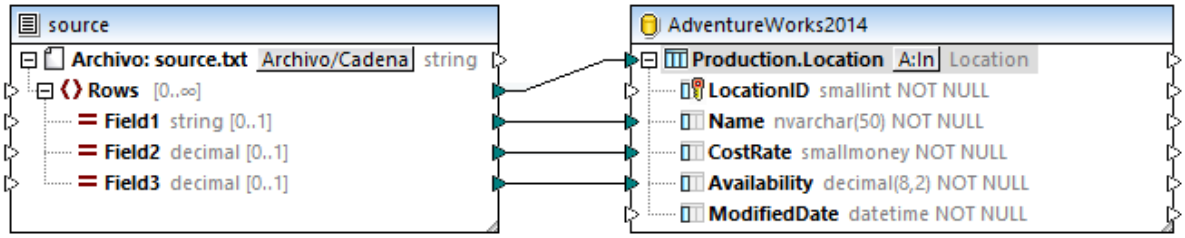


3. En el menú **Insertar**, haga clic en **Texto** y añada un archivo de texto origen (como el ejemplo anterior `source.txt`). Para más información consulte Archivos CSV y archivos de texto 445. Asegúrese de que los tipos de datos de los componentes de origen y de destino son compatibles. Puede ver los tipos de

datos en el componente habilitando el botón de la barra de herramientas **Mostrar tipos de datos** ( ).

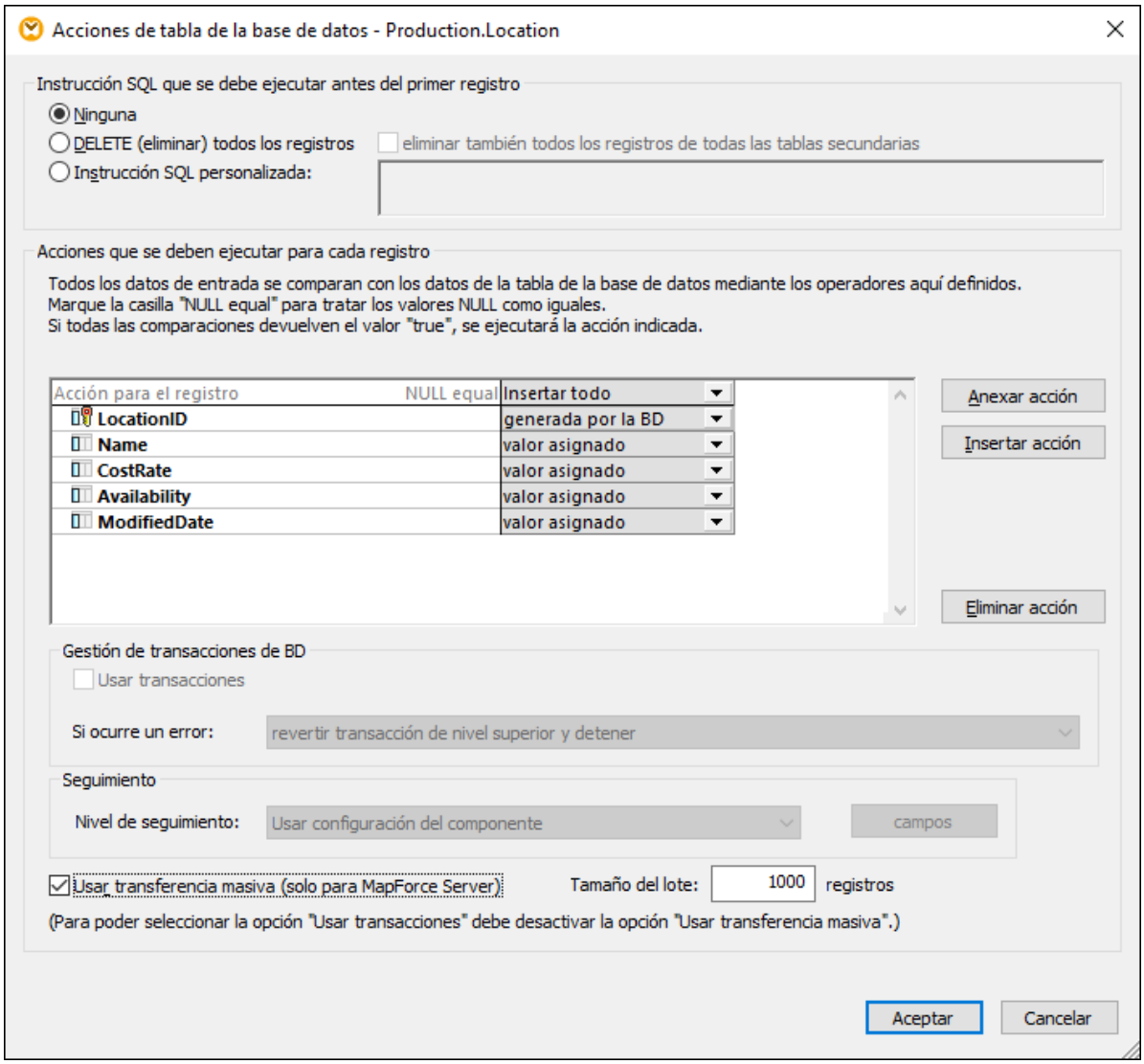


4. Dibuje las conexiones de la asignación como mostramos a continuación. Observe que el botón **Acciones de BD** A:In ahora es visible a la derecha del nombre de la tabla.



5. Haga clic en el botón **Acciones de BD** (A:In), marque la casilla **Usar inserciones masivas** y haga clic en **Aceptar**.

© 2016-2022 Altova GmbH



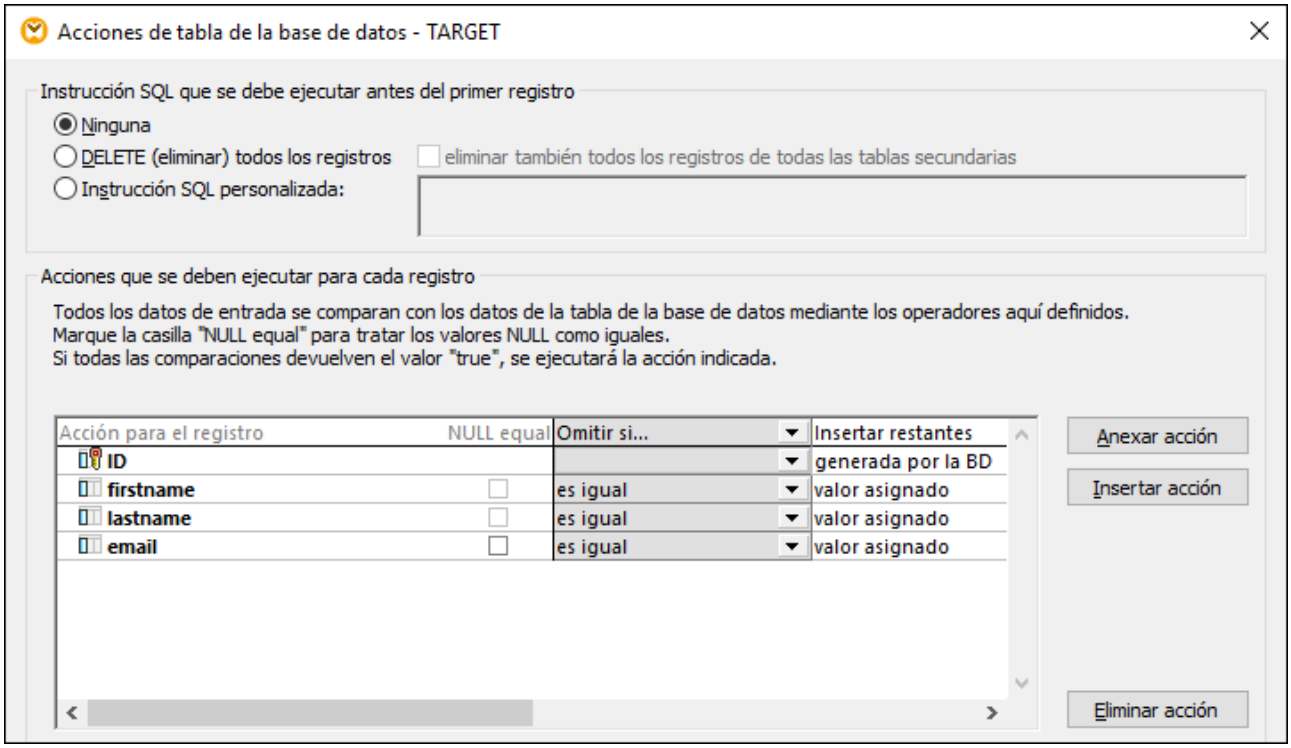
En el cuadro de diálogo anterior, observe que la "Acción para el registro" es "Insertar todo". El campo *Tamaño del lote* indica el número de registros que se insertan con cada acción.

Nota: cuando se habilita la opción *Usar transferencia masiva* se deshabilita automáticamente la opción *Usar transacciones* y viceversa. Si quiere procesar transacciones, desmarque la casilla *Usar transferencia masiva*.

Ahora que están habilitadas las inserciones masivas, el siguiente paso es ejecutar la asignación en MapForce Server (sea solo o gestionado con FlowForce Server). Para más información consulte Compilar asignaciones en archivos de ejecución de MapForce Server 955 y Implementar asignaciones en FlowForce Server 958.

4.2.3.10 Gestionar valores null en acciones de tabla de BD

Cuando una asignación actualiza una BD de destino con acciones de tabla como **Omitir si...**, **Actualizar si...** o **Eliminar si...**, MapForce compara los datos de origen con los de destino y genera consultas internas de actualización de BD como resultado. Estas consultas internas se pueden previsualizar en el panel *Resultados* de MapForce, véase Ejecutar asignaciones que modifican BD 273. Las consultas generadas reflejan las condiciones de comparación que se definieron en el cuadro de diálogo "Acciones de tabla de BD".



Cuadro de diálogo Acciones de tabla de BD

Si los datos de origen o de destino contienen campos que aceptan valores `null`, tiene dos opciones para comparar los valores `null` de los datos de origen con los de destino:

1. tratar los valores `null` como iguales
2. tratar los valores `null` como no iguales

En el cuadro de diálogo anterior aparece una casilla *Null equal* junto a algunos de los campos. El que marque o no esta casilla puede afectar al resultado de la asignación. Tenga en cuenta que solo se puede marcar la casilla en los campos que admiten valores `null` y si al menos una acción de la tabla tiene una condición "es igual" o "es igual (omitir may/min)".

Para evitar obtener resultados no deseados debe marcar la casilla *Null equal* si se cumplen todas estas condiciones:

1. El cuadro de diálogo "Acciones de tabla de BD" contiene acciones **Omitir si**, **Actualizar si** o **Eliminar si**, y
2. Estas acciones se van a ejecutar en registros que puede que contengan valores `null` y

 © 2016-2022 Altova GmbH

3. Los valores `null` de la tabla de origen se deben tratar como iguales a los de la tabla de destino.

La configuración predeterminada es no tener en cuenta los valores `null`. Si se cumplen los requisitos antes mencionados y no está habilitada la opción de tener en cuenta los valores `null`, puede haber casos en los que la tabla de BD no se actualice correctamente (por ejemplo, se insertan o actualizan más filas de las necesarias). Esto ocurre porque los valores `null` afectan a la comparación de datos y podrían interferir con los resultados. Por ejemplo, en la imagen anterior, imaginemos que un registro tiene un valor `null` en el campo *email* tanto en los datos de origen como en los de destino. Si marca la casilla *Null equal* ese registro cumplirá la condición de la acción **Omitir si...** y no se tendrá en cuenta. Sin embargo, si no marca la casilla *Null equal* el registro ya no cumplirá esa condición y se insertará en la BD.

Ejemplo

Para entender mejor la gestión de los valores `null` en asignaciones de datos vamos a analizar un ejemplo en el que encontraremos comparaciones de valores `null`. En este ejemplo usamos una BD de Microsoft SQL Server, pero se podría aplicar a cualquier otro tipo de base de datos compatible. O, si tiene Microsoft SQL Server, puede crear las tablas y los datos de este ejemplo ejecutando el siguiente script de BD: **<Documentos>\Altova\MapForce2023\MapForceExamples\Tutorial\CreateNullableFields.sql**.

Por comodidad reproducimos a continuación las tablas de la BD. Ambas tablas contienen datos de personas y tienen las mismas columnas. Además, la columna `email` puede contener valores `null` en ambas tablas.

```
+----+-----------+-----------+-------------------------+
| id | firstname | lastname  | email                   |
+----+-----------+-----------+-------------------------+
| 1  | Toby      | Hughey    | t.hughey@nanonull.com   |
| 2  | Mia       | Dahill    | NULL                    |
| 3  | Fred      | Weinstein | f.weinstein@nanonull.com |
+----+-----------+-----------+-------------------------+
```

La tabla SOURCE

```
+----+-----------+-----------+-------------------------+
| id | firstname | lastname  | email                   |
+----+-----------+-----------+-------------------------+
| 1  | Mia       | Dahill    | NULL                    |
| 2  | Fred      | Weinstein | f.weinstein@nanonull.com |
+----+-----------+-----------+-------------------------+
```

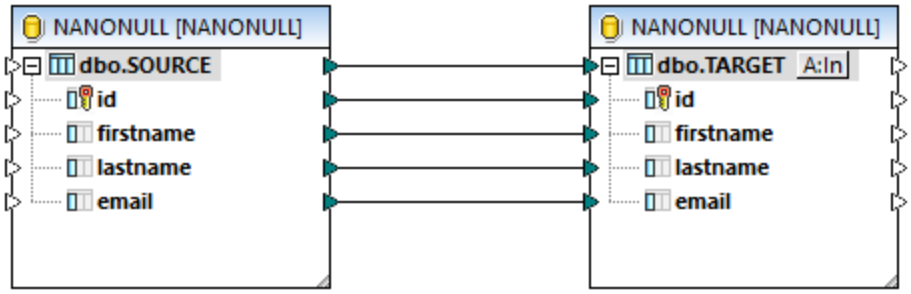
La tabla TARGET

Imaginemos que quiere combinar datos de la tabla `SOURCE` con la tabla `TARGET`. En la tabla `TARGET` se insertan únicamente los registros nuevos (en este ejmplo "Tobie Hughey"). Los registros que existen en las dos tablas ("Mia Dahill" y "Fred Weinstein") se ignoran.

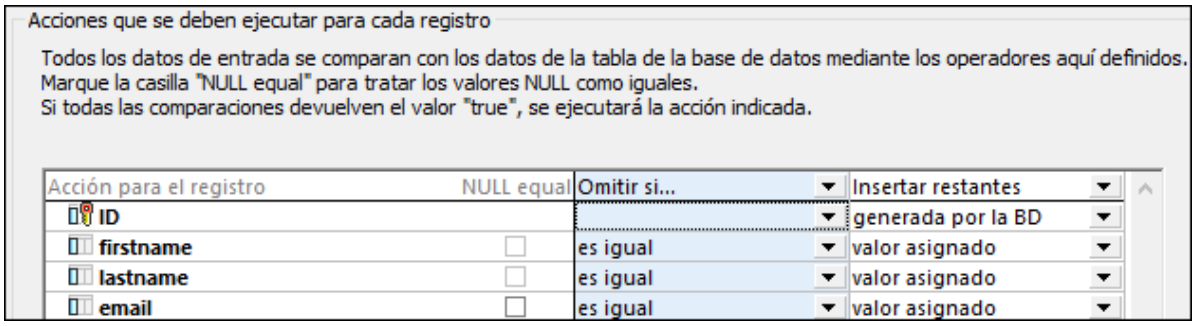
Para llevar a cabo esta tarea seguiremos los pasos que se describen a continuación:

1. En el menú **Insertar**, seleccione **Base de datos**. Siga las instrucciones del asistente para conectarse a la base de datos (*véase también* *Conectarse a una base de datos* 161). Cuando deba añadir objetos de BD, seleccione la tabla `SOURCE`.
2. En el menú **Insertar**, seleccione **Base de datos**. Vuelva a conectarse a la BD y añada la tabla `TARGET` a la asignación.

3. Dibuje las conexiones de la asignación entre los componentes de origen y de destino.



4. Haga clic en el botón **Acción: Insertar** A:In y configure las acciones de la tabla de BD como sigue:



La configuración combina las acciones **Omitir si** e **Insertar restantes**, lo que significa que por cada registro la asignación comprueba si:

- `firstname` es en el origen igual que `firstname` en destino, y
- `lastname` es en el origen igual que `lastname` en destino, y
- `email` es en el origen igual que `email` en el destino.

Si se cumplen todas estas condiciones, se ignora el registro. De lo contrario se inserta un nuevo registro en la tabla de destino. El `id` del nuevo registro lo genera la BD, mientras que el resto de campos (`firstname`, `lastname`, `email`) se rellenan con valores asignados desde el origen.

La casilla junto al campo `email` habilita o deshabilita la función de comparar teniendo en cuenta los valores `null` para este campo. En este ejemplo debemos marcar esta casilla porque el campo `email` puede contener valores `null` (concretamente, "Mia Dahill" tiene una dirección de correo electrónico `null`). Para ver cómo afecta a la comparación esta casilla, pruebe a actualizar la base de datos dos veces: en la primera, marque la casilla pero en la segunda no.

Para actualizar la base de datos, haga clic en la pestaña *Resultados* y ejecute el comando de menú **Resultados | Ejecutar script SQL**.

Si la casilla está marcada estamos indicando a MapForce explícitamente que trate los valores `null` como iguales. Por lo tanto, el registro "Mia Dahill" no se inserta en la tabla de destino, que es lo que buscamos.

Si la casilla no está marcada, el registro "Mia Dahill" se inserta en la tabla de destino (a pesar de que ya existe), que no es lo que pretendemos. Esto ocurre porque no le hemos dado a MapForce una indicación explícita de que queremos que trate los valores `null` como iguales. Ocurre algo parecido si ejecutamos la siguiente consulta en la base de datos (esta consulta no obtiene registros porque el valor `null` se compara con el operador "=", por lo que no tiene en cuenta los valores `null`).

 © 2016-2022 Altova GmbH

```
SELECT firstname, lastname, email FROM TARGET WHERE firstname = 'Mia' AND lastname =
'Dahill' AND email = NULL;
```

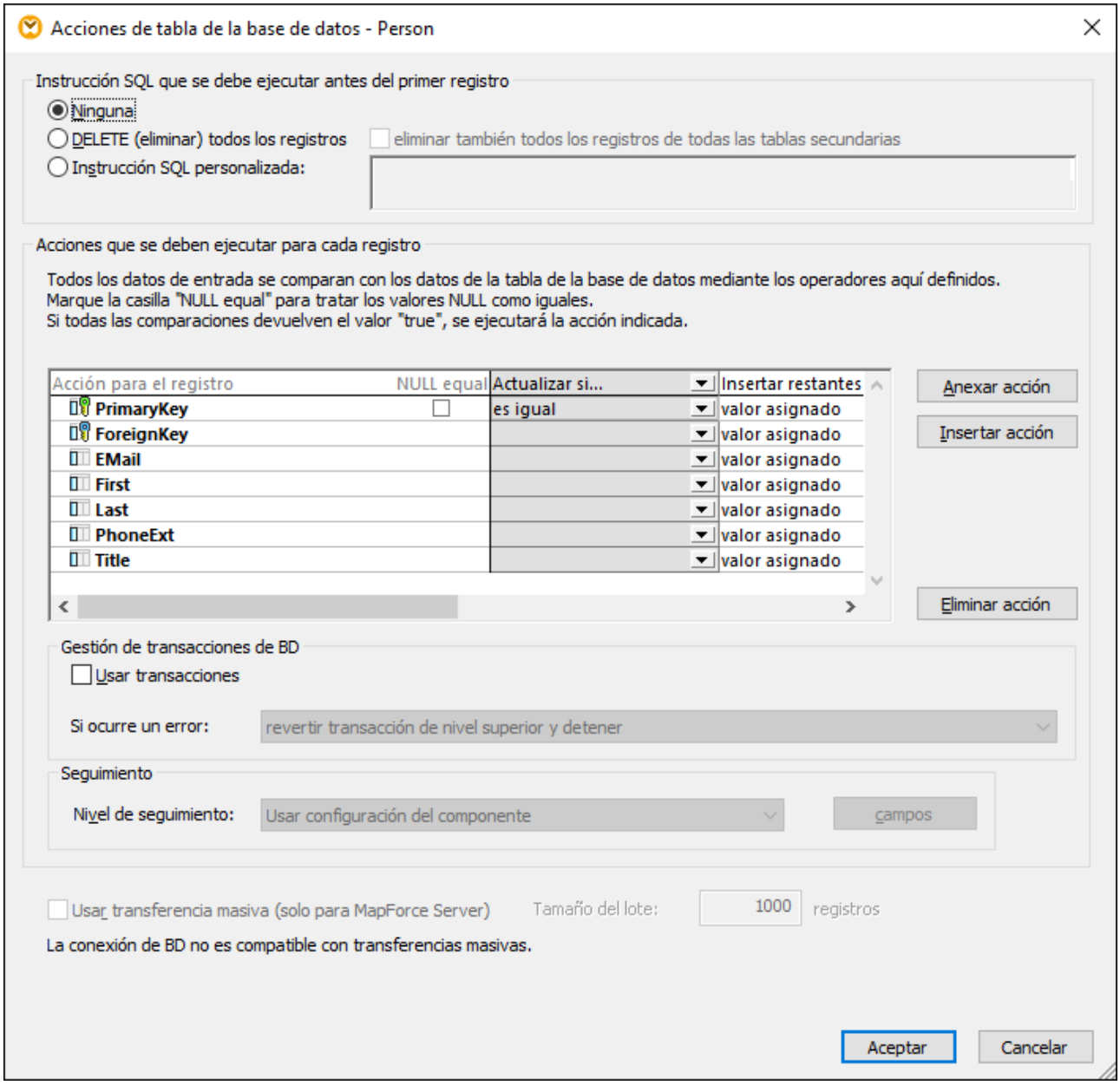
Para que tenga en cuenta los valores `null`, deberíamos reescribir la consulta anterior así:

```
SELECT firstname, lastname, email FROM TARGET WHERE firstname = 'Mia' AND lastname =
'Dahill' AND email IS NULL;
```

Nota: las consultas anteriores tienen propósitos únicamente ilustrativos y no reflejan la sintaxis real de las consultas internas que genera MapForce. MapForce adapta la sintaxis de las consultas que genera según el tipo de base de datos (ya que cada proveedor de bases de datos enfoca la comparación de valores `null` de forma distinta).

4.2.3.11 Configurar acciones de tabla de BD

Cada vez que se crea una conexión de asignación de datos con una BD aparece un botón **Acciones de la BD** junto a la tabla en cuestión. Al hacer clic en este botón se abre el cuadro de diálogo "Acciones de tabla de BD", desde donde puede configurar las acciones de BD insertar, actualizar y eliminar, además de otras opciones.



Cuadro de diálogo Acciones de tabla de BD

A continuación aparece una descripción de las opciones que puede configurar en el cuadro de diálogo "Acciones de tabla de BD".

Ejecutar una instrucción SQL antes del primer registro

En este cuadro de diálogo puede definir instrucciones SQL que se ejecuten antes que las acciones definidas en **Acciones que se deben ejecutar para cada registro**. Puede seleccionar cualquiera de estos botones:

- **Ninguna**: no se lleva a cabo ninguna acción. Esta es la configuración predeterminada.

 © 2016-2022 Altova GmbH

- **DELETE (eliminar) todos los registros**: se eliminan todos los registros de la tabla seleccionada antes de que se lleve a cabo ninguna acción definida en el cuadro de diálogo **Acciones que se deben ejecutar para cada registro**. Si también quiere eliminar los datos almacenados en las tablas secundarias de la tabla seleccionada, marque la casilla **eliminar también todos los registros de todas las tablas secundarias**. Si quiere ver un ejemplo, consulte Insertar datos en tablas vinculadas 286.
- **Instrucción SQL personalizada**: escriba una instrucción personalizada que afecte a toda la tabla. Tenga en cuenta que podrá o no incluir varias instrucciones SQL en una misma consulta dependiendo de la BD, del método de conexión y del driver que use.

Acciones que se deben ejecutar para cada registro

Este grupo de opciones indica las acciones de BD que deben llevarse a cabo en esta tabla cuando se ejecute la asignación. Para gestionar las acciones de la tabla, haga clic en los botones **Anexar acción**, **Insertar acción** o **Eliminar acción**. Si es necesario se pueden definir varias acciones (por ejemplo una acción **Actualizar** si seguida de una acción **Insertar restantes**).

Las acciones de tabla definidas se procesan de izquierda a derecha. En el ejemplo anterior se procesa primero la acción **Actualizar si**. Si no se cumple la condición para actualizar se procesa la siguiente acción (en este ejemplo la acción **Insertar restantes**). Tenga en cuenta lo siguiente:

- Para que se ejecute una acción deben cumplirse todas las condiciones definidas para esa acción. Si se cumplen todas las condiciones, se actualizan todos los campos para los que exista una conexión entre los elementos de origen y destino en la asignación. Cualquier otra acción de tabla (a la derecha de una acción cuya condición se cumple) se ignora para ese registro.
- Si no se cumple la condición definida, entonces la acción de la tabla se omite y se ejecuta la siguiente acción (a la derecha).
- Si no se cumple ninguna de las condiciones, no se ejecuta ninguna acción.

Cualquier acción que se defina después de **Insertar todo** o **Insertar restantes** no se ejecutará nunca porque no hay condiciones de columna para acciones de insertar. En este caso aparece un cuadro de diálogo informando de que se eliminarán las columnas de acción de tabla posteriores.

La casilla Null Equal junto a cada registro, cuando proceda, indica a MapForce que los valores `null` del registro de origen deben ser tratados como iguales a los valores `null` del registro de destino. Si no se selecciona esta casilla es posible que los resultados no sean correctos (*véase Gestionar valores null en acciones de tabla de BD* 324).

Cuando la asignación actualiza una tabla que tiene relaciones de claves foráneas con otras tablas se pueden usar las siguientes opciones:

Eliminar datos en tablas secundarias	Esta opción cobra importancia si selecciona la acción **Actualizar si** en una tabla primaria. Puede que sea necesario si el número de registros del archivo de origen no es el mismo que el de la BD de destino y quiere mantener las bases de datos sincronizadas (sin datos huérfanos en las tablas secundarias). Consulte también Opciones para tablas secundarias cuando se actualiza la tabla principal 303.
Omitir datos secundarios de entrada	Use esta opción si quiere actualizar una tabla primaria de destino sin que eso afecte al resto de tablas o de registros de esa tabla. Consulte

	también Opciones para tablas secundarias cuando se actualiza la tabla principal 303.

Aquí tiene varios ejemplos que ilustran distintas combinaciones de acciones:

- Insertar datos en una tabla 282
- Insertar datos en tablas vinculadas 286
- Actualizar una tabla 291
- Opciones para tablas secundarias cuando se actualiza la tabla principal 303
- Acción "Actualizar si... Insertar restantes" 294
- Acción "Eliminar si..." 311
- Acción "Omitir si..." 315

Usar transacciones

Habilita las transacciones de BD para esta acción de tabla en particular. Para más información consulte Gestionar excepciones de BD 422.

Usar transferencia masiva

Habilita las transferencias masivas (varias instrucciones INSERT en una consulta). Se pueden usar transferencias masivas si la asignación se ejecuta en MapForce Server y la acción de BD es **Insertar todo**. Para más información consulte Inserciones masivas 320.

4.2.3.12 Ejemplo: asignar datos XML a datos SQLite

Este ejemplo explica qué pasos se han de seguir para crear una asignación de MapForce que lea datos de un archivo XML y los escriba en una BD SQL. El ejemplo incluye un diseño simple de asignación (.mfd). Si quiere ver este archivo antes de empezar con el ejemplo, lo puede encontrar aquí: **<Documentos>\Altova\MapForce2023\MapForceExamples\Tutorial\XMLtoSQLite.mfd**.

El objetivo del ejemplo es insertar datos en una BD SQLite desde un archivo XML. Para alcanzar este objetivo necesitará una base de datos SQLite vacía en la que escribir los datos. Como se muestra más abajo, puede crear y explorar la base de datos SQLite tanto con Altova DatabaseSpy como con la shell de línea de comandos que puede encontrar en la página oficial de SQLite.

Para crear la base de datos SQLite:

Si DatabaseSpy está instalado en su ordenador (solo o como parte de Altova Missionkit), puede crear la nueva base de datos SQLite como sigue:

1. Ejecute DatabaseSpy.
2. En el menú **Archivo**, haga clic en **Crear una conexión a una base de datos**.
3. Haga clic en **Asistente para la conexión** y luego en **SQLite**.
4. Haga clic en **Crear una base de datos SQLite nueva** e introduzca como ruta **c:\sqlite\articles.sqlite**. Haga clic en **Conectarse**.
5. Cuando deba introducir un nombre de BD, deje el predeterminado.
6. Abra un nuevo editor de SQL (**Ctrl+N**) y ejecute la siguiente instrucción en la BD:

 © 2016-2022 Altova GmbH

```
create table articles (number smallint, name varchar(10), singleprice real);
```

Otra alternativa para crear la base de datos es la siguiente:

1. Descargue la shell de línea de comandos SQLite para Windows de la página de descargas de SQLite (http://www.sqlite.org/download.html) y extraiga los archivos del .zip a un directorio de su equipo local (para este ejemplo, use **C:\sqlite**).
2. Ejecute **c:\sqlite\sqlite3.exe** e introduzca la siguiente instrucción:

   ```
   create table articles (number smallint, name varchar(10), singleprice real);
   ```

   Esto crea la tabla `articles` en la base de datos en memoria. La tabla `articles` consiste en tres columnas: *number*, *name* y *singleprice*. Estas columnas sirven para almacenar datos de los elementos con el mismo nombre definidos en el esquema XML de origen. Cada columna tiene designado un tipo de datos compatible con los datos que se van a guardar en esa columna.
3. Ejecute el siguiente comando:

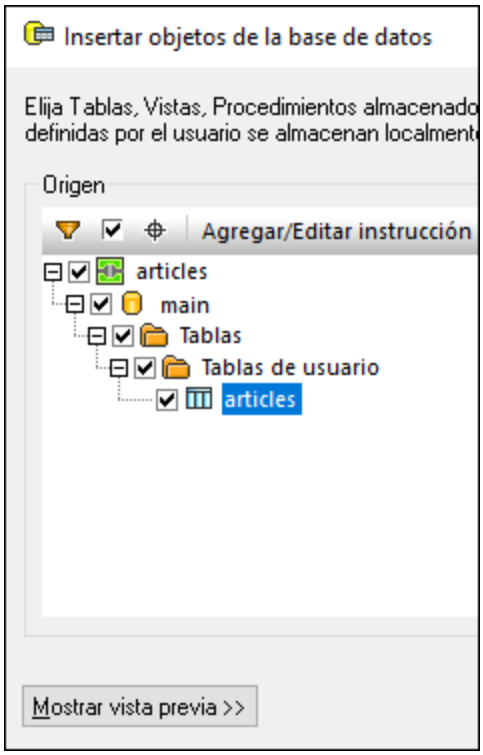
   ```
   .save articles.sqlite
   ```

   Este comando guarda la base de datos en memoria en la siguiente ruta: **c:\sqlite\articles.sqlite**. Tenga en cuenta que en los pasos siguientes tendrá que hacer referencia a esta ruta.

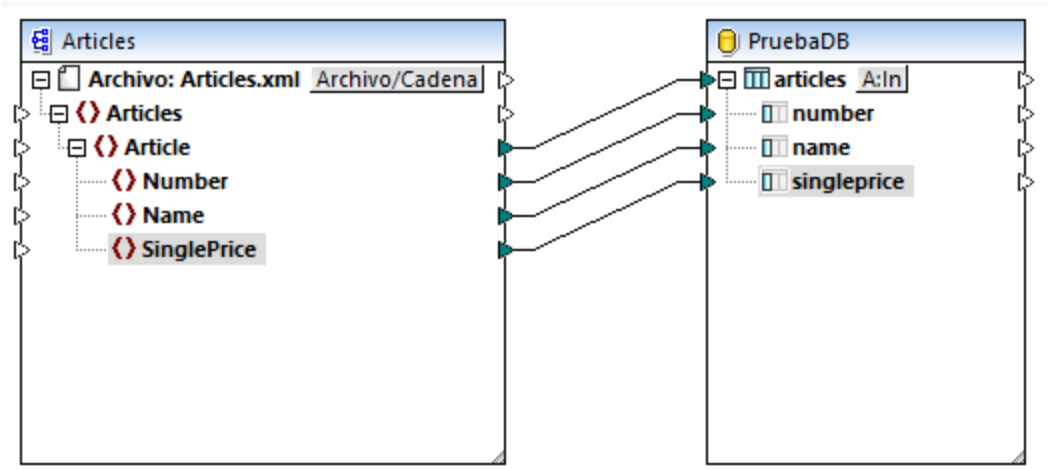
Ahora ha terminado de crear la base de datos SQLite de prueba que necesitamos para este ejemplo.

Para crear el diseño de asignación de XML a SQLite:

1. Ejecute MapForce y asegúrese de que selecciona el motor de transformación integrado (use el comando de menú **Resultados | Built-in Execution Engine**).
2. Añada al área de asignación el archivo `Articles.xml`, que encontrará en la carpeta **<Documentos>\Altova\MapForce2023\MapForceExamples\Tutorial** (use el comando de menú **Insertar | Archivo o esquema XML**).
3. Añada al área de asignación la base de datos `articles.sqlite` creada en los pasos anteriores (use el comando de menú **Insertar | Base de datos**) y seleccione SQLite.
4. Haga clic en **Conectarse**. Cuando deba escojer objetos de BD, seleccione la tabla `articles`.

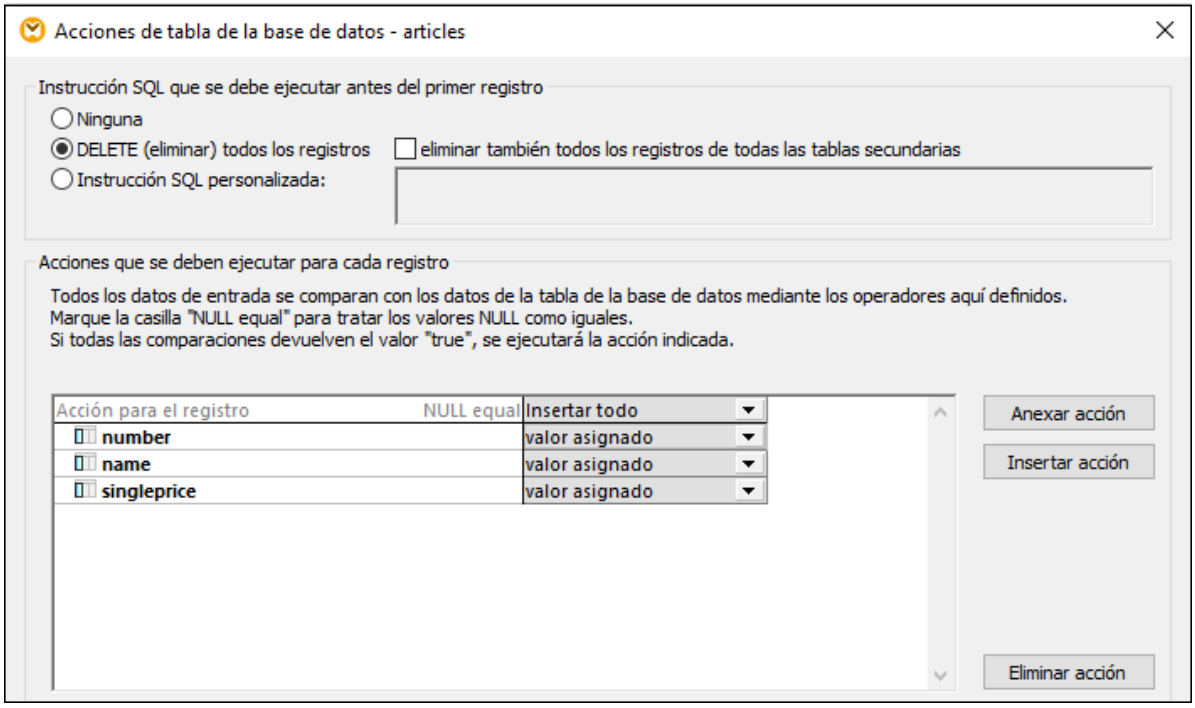


5. Dibuje las conexiones como se muestra a continuación:



6. Haga clic en el botón **A:In** en el componente de BD y seleccione la opción **Eliminar todos los registros**. Esto garantiza que cada vez que se ejecute la asignación, todas las filas de BD ya existentes se borran para evitar duplicados.

© 2016-2022 Altova GmbH



7. Haga clic en la pestaña *Resultados* de la ventana principal de la asignación. MapForce ejecuta la asignación usando el motor de ejecución integrado y muestra la consulta realizada en la ventana Resultados.

```
/*
Las instrucciones SQL que aparecen a continuación se presentan únicamente de forma preliminar y no
se pueden ejecutar en otra herramienta de consultas SQL.
Para ejecutar estas instrucciones, utilice la función "Ejecutar script SQL" del menú "Resultados".
Conéctese a la base de datos usando esta cadena de conexión:
C:/Users/m.merodio/Desktop/PruebaDB
*/

PRAGMA foreign_keys = ON;

DELETE FROM "articles"

INSERT INTO "articles" ("number", "name", "singleprice") VALUES (1, 'T-Shirt', 25)

INSERT INTO "articles" ("number", "name", "singleprice") VALUES (2, 'Socks', 2.3)

INSERT INTO "articles" ("number", "name", "singleprice") VALUES (3, 'Pants', 34)

INSERT INTO "articles" ("number", "name", "singleprice") VALUES (4, 'Jacket', 57.5)
```

8. Ejecute el script SQL para rellenar la BD (use el comando de menú **Resultados | Ejecutar script SQL**). Si MapForce no encuentra ningún error de tiempo de ejecución, los registros se insertan en la base de datos SQLite.

```
1   /*
2   Las instrucciones SQL que aparecen a continuación se ejecutaron durante la función "Generar
    resultados".
3   Cada resultado aparece escrito a la derecha de la cadena "-->>>".
4   Estas instrucciones se presentan únicamente de forma preliminar y no se pueden ejecutar en otra
    herramienta de consultas SQL.
5   La base de datos se conectó usando esta cadena de conexión:
6   C:/Users/m.merodio/Desktop/PruebaDB
7   */
8
9   PRAGMA foreign_keys = ON;
10  -->>> OK. 0 fila(s).
11
12  DELETE FROM "articles"
13  -->>> OK. 0 fila(s).
14
15  INSERT INTO "articles" ("number", "name", "singleprice") VALUES (1, 'T-Shirt', 25)
16  -->>> OK. 1 fila(s).
17
18  INSERT INTO "articles" ("number", "name", "singleprice") VALUES (2, 'Socks', 2.3)
19  -->>> OK. 1 fila(s).
20
21  INSERT INTO "articles" ("number", "name", "singleprice") VALUES (3, 'Pants', 34)
22  -->>> OK. 1 fila(s).
23
24  INSERT INTO "articles" ("number", "name", "singleprice") VALUES (4, 'Jacket', 57.5)
25  -->>> OK. 1 fila(s).
26
```

Para comprobar si los datos se han insertado correctamente en la base de datos SQLite:

1. Haga clic en la pestaña **Consulta de la BD** de MapForce.
2. Seleccione la la entrada `articles` de la lista desplegable en la parte superior del panel **Consulta de la BD**.
3. Introduzca la siguiente consulta en el editor SQL:

```
select * from articles;
```

También puede optar por este otro método:

1. Ejecute el archivo **c:\sqlite\sqlite3.exe** y abra la BD con el siguiente comando:

```
.open articles.sqlite
```

2. Ejecute la siguiente instrucción:

```
select * from articles;
```

Independientemente del método que escoja para seleccionar los datos de la BD (MapForce o la shell de SQLite), la consulta debería devolver cuatro filas, que corresponden con el número de registros en el archivo XML de origen, que es lo que pretendíamos con este ejemplo.

 © 2016-2022 Altova GmbH

4.2.4 Combinar datos de BD

En las asignaciones de datos que leen datos de bases de datos podrá combinar objetos de la base de datos, como tablas o vistas, con solo añadir un componente de combinación a la asignación. Por ejemplo, puede combinar datos de dos o más tablas vinculadas por medio de relaciones de clave foránea, la manera más corriente de almacenar datos en bases de datos relacionales. El resultado sería idéntico al que se obtendría si se ejecutara una consulta SQL que uniera dos o más tablas por medio de una operación INNER JOIN en la base de datos.

Dependiendo del tipo de datos que se conecten al componente de combinación, la operación de combinación puede tener lugar en el modo estándar (no SQL) o en el modo SQL. Las combinaciones en modo no SQL las lleva a cabo MapForce, mientras que las combinaciones en modo SQL las lleva a cabo la base de datos en la que lee datos la asignación.

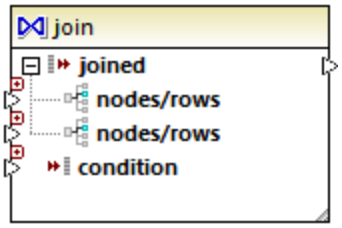
Las combinaciones en modo no SQL son más flexibles porque admiten más tipos de componentes de entrada (p. ej. la combinación puede ser entre tablas de bases de datos distintas o entre estructuras XML y tablas de base de datos). Para ver un ejemplo de combinación no SQL consulte el apartado Ejemplo: combinar estructuras XML 523. Por otro lado, las combinaciones no SQL exige al motor de asignación comparaciones que requieren un gran uso de memoria (porque el número total de comparaciones representa la combinación cruzada o producto cartesiano de todas las estructuras combinadas). Este proceso se lleva a cabo a gran velocidad por lo general y no se aprecia en las asignaciones que no hacen un uso intensivo de los datos. Sin embargo, si los orígenes de datos combinados están compuestos por un gran número de registros, puede que la asignación necesite mucho tiempo para ejecutarlos. Si la asignación que tiene entre manos debe procesar un gran número de registros, considere obtener una licencia de MapForce Server Advanced Edition, que optimiza especialmente las combinaciones para acelerar la ejecución de asignaciones de datos.

Las combinaciones en modo SQL solamente aceptan determinados objetos de base de datos como entrada (p. ej. tablas o vistas), así que no son tan flexibles como las combinaciones no SQL. Sin embargo, ofrecen un mayor rendimiento de la asignación porque la base de datos las ejecuta de forma nativa. Para más información consulte el apartado Combinaciones en modo SQL 336.

Nota: los componentes de combinación no son el único método posible para combinar tablas o vistas de base de datos. También puede realizar combinaciones de datos de base de datos con ayuda de instrucciones SQL SELECT (*véase* *Instrucciones SQL SELECT como tablas virtuales* 359). La principal diferencia entre las instrucciones SQL SELECT y los componentes de combinación es que aquellas se escriben a mano y permiten mayor flexibilidad. Los componentes de combinación son una alternativa sencilla para quienes no dominen SQL.

Para agregar un componente de combinación:

1. Active el lenguaje de transformación BUILT-IN (haciendo clic en el botón de la barra de herramientas o con el comando de menú **Resultados | Motor de ejecución integrado**).
2. Haga clic en el comando de menú **Insertar | Combinación** o en el botón **Combinación** de la barra de herramientas. El componente de combinación aparece en la asignación y por defecto acepta datos de dos estructuras, así que tiene dos entradas `nodes/rows`. Si lo necesita, puede agregar entradas nuevas en la combinación con ayuda del botón **Agregar entrada** (*véase* *Combinar tres o más estructuras* 522).



3. Conecte las estructuras que se deben combinar con las entradas `nodes/rows` del componente de combinación.
4. Añada la condición para la combinación (o varias condiciones). Esto se consigue haciendo clic con el botón derecho en el componente de combinación y seleccionando **Propiedades**. También puede añadir condiciones de combinación desde la asignación directamente: conecte el resultado booleano de una función con la entrada `condition` del componente de combinación. Cuando se están combinando tablas de base de datos, MapForce puede crear condiciones de combinación automáticamente (*véase* *Agregar condiciones de combinación* 519).

Notas:

- Se admite el uso de componentes de combinación cuando el lenguaje de destino de la asignación es el motor de ejecución integrado (BUILT-IN). No es compatible con la generación de código C#, C++ o Java.
- Cuando una estructura no es una fuente de entrada válida ni compatible para la combinación, MapForce emite sugerencias en la asignación directamente o, cuando se valide la asignación, en la ventana Mensajes (*véase* *Validar asignaciones* 105).
- No se deberían conectar componentes de combinación con parámetros de entrada ni resultados de funciones definidas por el usuario inline. Si existen dichas conexiones, se producirán errores de validación.
- Cuando conecte componentes de base de datos (tablas o vistas, por ejemplo) a un componente de combinación directamente, en la esquina superior derecha del componente de combinación se habilitará el botón **Modo SQL** SQL. Si se activa, este botón ofrece funciones SQL especiales para trabajar con la operación de combinación (*véase* *Combinaciones en modo SQL* 336).
- No se puede conectar la salida del nodo `joined` con otro componente de combinación, pero si lo necesita puede conectar un resultado parcial de una combinación con otra.

4.2.4.1 Combinaciones en modo SQL

Cuando se conectan los componentes de base de datos apropiados (p. ej. tablas o vistas) a un componente de combinación, en la esquina superior derecha del componente de combinación aparece el botón **Modo SQL** SQL. Cuando se habilita el modo SQL, la operación de combinación la lleva a cabo la base de datos de la que lee datos la asignación. En otras palabras, MapForce enviará una consulta a la base de datos compuesta con la sintaxis SQL adecuada para seleccionar y combinar datos de todas las tablas que participan en la combinación. Y, lo que es más importante, no será necesario escribir código SQL porque la consulta se produce en base al diseño visual que tenga el componente de combinación en la asignación de datos (*ver ejemplos más abajo*).

Para poder habilitar el modo SQL deben cumplirse estos requisitos:

1. Los dos objetos que se deben combinar (tablas o vistas) deben ser de la misma base de datos.

 © 2016-2022 Altova GmbH

2. Los dos objetos que se deben combinar deben proceder del mismo componente de MapForce (recuerde que puede agregar/quitar objetos de base de datos de un componente haciendo clic con el botón derecho en el componente y seleccionando **Agregar, quitar o editar objetos de la base de datos** en el menú contextual).
3. La condición (o las condiciones) de combinación debe definirse desde las propiedades del componente solamente (haciendo clic con el botón derecho en el componente de combinación y seleccionando **Propiedades**) y no en la asignación (*véase* *Agregar condiciones de combinación* 519).

Nota: cuando se combinan tablas de base de datos en modo SQL, MapForce crea las condiciones de combinación automáticamente a partir de las relaciones de clave foránea detectadas entre las tablas. Para que MapForce pueda crear condiciones de combinación automáticas las tablas de la base de datos deben estar en una relación secundario/primario en el componente de MapForce (es decir, una tabla debe ser el primario o secundario de otra tabla en el componente). Consulte el apartado Ejemplo: combinar tablas en modo SQL 344 para obtener más información.

4. Las tablas de la base de datos no pueden estar todavía en el contexto de destino actual. Cuando el resultado de la combinación se usa en un componente de destino ninguna de las tablas combinadas puede estar conectada directa o indirectamente a ninguno de los nodos primarios de destino. Para más información consulte Reglas y estrategias de asignación de datos 871.

El modo SQL se puede controlar y consultar con ayuda del botón **SQL** SQL situado en la esquina superior derecha del componente de combinación:

SQL El modo SQL está deshabilitado (la combinación la ejecutará MapForce o, si procede, MapForce Server).

SQL El modo SQL está habilitado (la combinación la ejecutará la base de datos).

Si falta el botón SQL, significa que el modo SQL no es relevante o no es compatible con los datos que se pretenden combinar.

En algunos casos es necesario deshabilitar el modo SQL explícitamente, por ejemplo:

- Cuando la asignación necesite condiciones de combinación fuera de las propiedades del componente de combinación (es decir, condiciones definidas en la asignación y conectadas al elemento `condition` del componente de combinación).
- Cuando necesite combinar tablas de bases de datos distintas. En este caso deberá usar una combinación estándar (no SQL).

Cambiar el modo de combinación

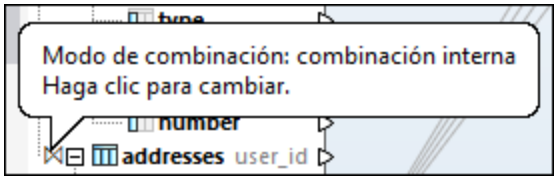
Cuando el componente de combinación está en Modo SQL SQL puede combinar varias tablas o vistas de BD en una de varias formas:

- Vínculo interno: el componente de combinación solo devuelve los registros que cumplan la condición en ambos conjuntos de entrada.
- Vínculo externo izquierdo: el componente de combinación incluye todos los registros de la tabla más a la izquierda (en MapForce esta es la tabla más alta de un componente de combinación), además de los registros de la tabla que se combina posteriormente y que cumplan con la condición de combinación.

Para ver el modo de combinación de una tabla o vista en el componente de combinación, observe el icono que aparece junto a la tabla o vista combinada. Estos iconos pueden aparecer en cualquier tabla combinada excepto en la primera:

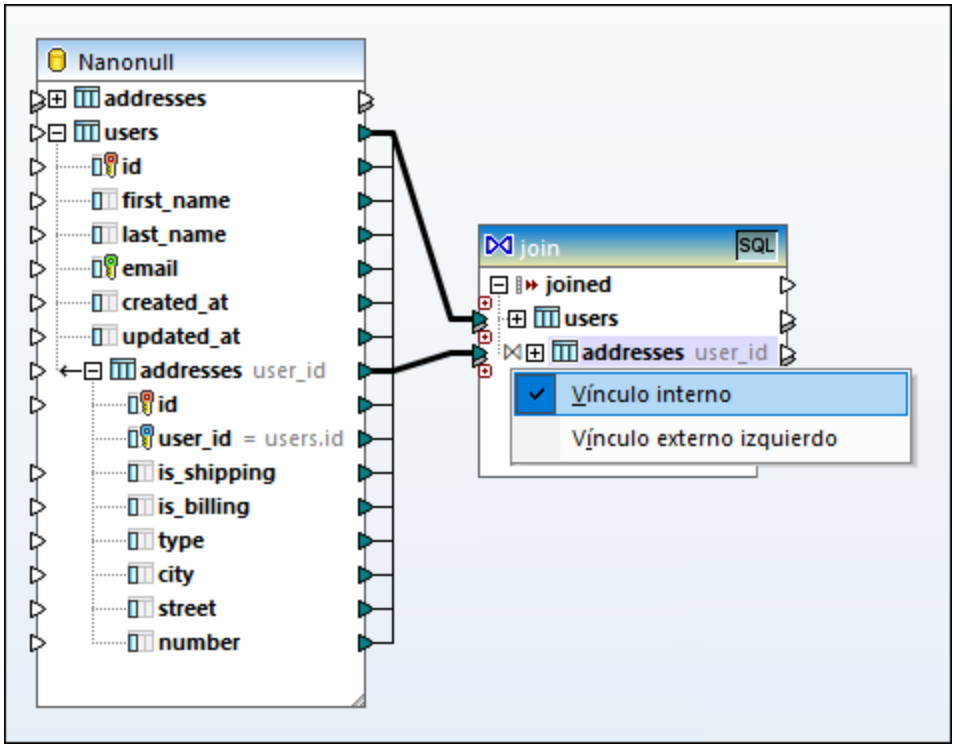
- Vínculo interno ⋈
- Vínculo externo izquierdo ⋈

Para ver información rápida sobre la combinación mueva el cursor del ratón sobre el icono:



Para cambiar el modo de combinación elija una opción:

- Haga clic en el icono **Vínculo interno** ⋈ o **Vínculo externo izquierdo** ⋈ que hay en cada tabla o vista combinadas y seleccione **Vínculo interno** o **Vínculo externo izquierdo** en el menú contextual.
- Haga clic con el botón derecho en la segunda (o tercera, cuarta, etc.) tabla o vista combinadas en el componente de combinación y seleccione **Tipo de vínculo | Vínculo interno** o **Tipo de vínculo | Vínculo externo izquierdo** en el menú contextual.



Tenga en cuenta que:

- Si cambia el modo de combinación al vínculo externo izquierdo, entonces la tabla o vista superior representa la parte izquierda de la combinación.

 © 2016-2022 Altova GmbH

- Cambiar el modo de combinación afecta a los datos que devuelve el componente de combinación de la misma forma que el vínculo interno o el vínculo externo izquierdo afectan a los resultados de una consulta SQL en una BD.

Nombres de alias

A menudo las tablas o vistas de base de datos combinadas contienen nombres de campo idéntico en ambas estructuras. Cuando se habilita el modo SQL, estos elementos aparecen en el componente con el prefijo "AS". Por ejemplo, si dos tablas combinadas contienen un campo llamado "id", este campo aparecerá como "id" en la primera tabla combinada y como "id AS id2" en la segunda tabla combinada. Las tablas combinadas también pueden dar lugar a nombres de alias (p. ej. cuando se combina una tabla consigo misma).

Los nombres de alias de campo o de tabla son importantes para hacerles referencia más adelante en la asignación. Por ejemplo, imagínese que quiere filtrar u ordenar el resultado de la combinación. Para conseguirlo deberá conectar el resultado del componente de combinación a un componente WHERE/ORDER de SQL, donde deberá introducir las cláusulas WHERE y ORDER BY de SQL.

Para hacer referencia al campo desde la cláusula WHERE deberá escribir el nombre de la tabla seguido de un punto y del nombre del campo. Para hacer referencia al alias de una tabla deberá usar el nombre de alias tal y como aparece en el componente de combinación. En la cláusula ORDER BY podrá usar la misma técnica (`tabla.campo`) o escribir solo el nombre de alias de campo (el nombre que aparece después de "AS").

Para ver una asignación que utilice cláusulas WHERE/ORDER de SQL consulte el apartado Ejemplo: combinar tablas en modo SQL 344.

Nota: no está permitido crear componentes WHERE/ORDER de SQL entre una tabla de base de datos y el componente de combinación. Es decir, solamente se pueden añadir después (pero no antes) del componente de combinación. Para más información consulte el apartado Filtrar y ordenar datos de BD (SQL WHERE/ORDER) 352.

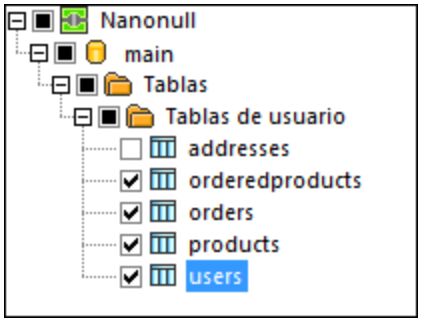
4.2.4.2 Ejemplo: crear un informe CSV a partir de varias tablas

Este ejemplo demuestra cómo combinar varias tablas de base de datos para extraer datos y guardarlos en un solo informe en formato CSV. La base de datos utilizada en este ejemplo se llama **Nanonull.sqlite** y está guardada en este directorio: **<Documentos>\Altova\MapForce2023\MapForceExamples\Tutorial**. Esta base de datos almacena información sobre una empresa ficticia (pedidos, productos, usuarios y direcciones de los usuarios). Como en cualquier otra base de datos relacional, la información de esta base de datos está normalizada y repartida en varias tablas. Por ejemplo, la tabla `users` almacena datos personales de los usuarios (su nombre, apellido y correo electrónico). La base de datos también almacena información sobre los productos adquiridos por los usuarios en dos tablas distintas: `orders` (que incluye el identificador único del pedido y la hora en la que se realizó el pedido) y `orderedproducts` (que incluye una lista de productos pedidos y su cantidad). Además, el nombre de los productos se almacena en una tabla llamada `products`.

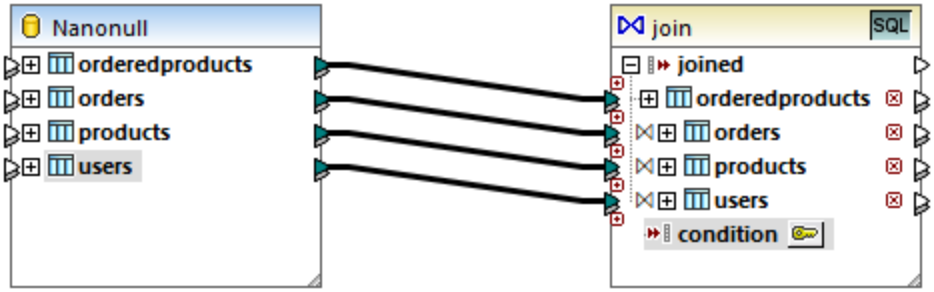
El objetivo de nuestra asignación de datos será producir un informe basado en los datos extraídos de varias tablas y así conocer quién pidió ciertos productos, cuándo y en qué cantidad. Para alcanzar este objetivo debemos seguir estos pasos:

1. En el menú **Insertar** haga clic en el comando **Base de datos**.
2. Cuando llegue el momento de seleccionar un tipo de base de datos, haga clic en la opción *SQLite* y después en el botón **Siguiente**.

3. Navegue hasta la base de datos **Nanonull.sqlite** que mencionamos al principio y haga clic en **Conectarse**.
4. En el cuadro de diálogo que aparece seleccione las tablas `orderedproducts`, `orders`, `products` y `users` y haga clic en **Aceptar**.



5. Añada un componente de combinación y después cree cuatro elementos `nodes/rows` (haciendo clic en el botón **Agregar entrada** del componente de combinación).
6. Conecte las cuatro tablas del componente de base de datos con los elementos de entrada del componente de combinación.



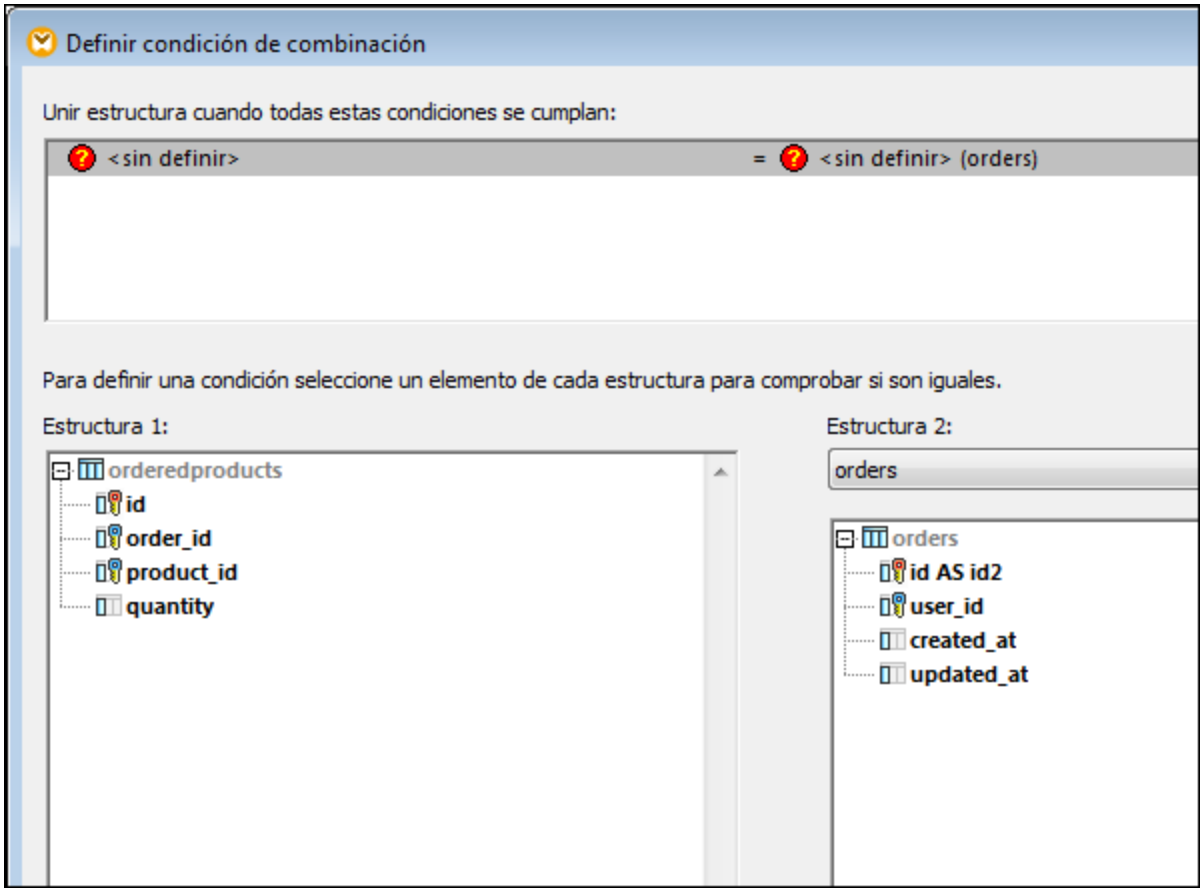
Nota: **otro método alternativo sería conectar la tabla** `orderedproducts` al componente de combinación, después la tabla `orders` (la que tiene anidada y no la que está al mismo nivel) y así sucesivamente hasta que todas las tablas combinadas estén anidadas bajo la misma tabla raíz (*véase* *Gestionar relaciones de BD* 258). El resultado de la asignación sería idéntico al que se obtendría al combinar las tablas de esta forma. La diferencia es que en este ejemplo las condiciones de combinación deben crearse a mano (ver más abajo), mientras que con el método alternativo recién descrito las condiciones de combinación las crearía MapForce automáticamente. Para ver un ejemplo en el que se combinan tablas sin necesidad de definir condiciones de combinación a mano consulte el apartado Ejemplo: combinar tablas en modo SQL 344. También puede consultar la asignación de datos **<Documentos>\Altova\MapForce2023\MapForceExamples\DB_Denormalize.mfd**, donde todas las tablas combinadas se encuentran bajo la misma tabla raíz.

En el ejemplo que nos ocupa las tablas conectadas al componente de combinación siguen este orden:

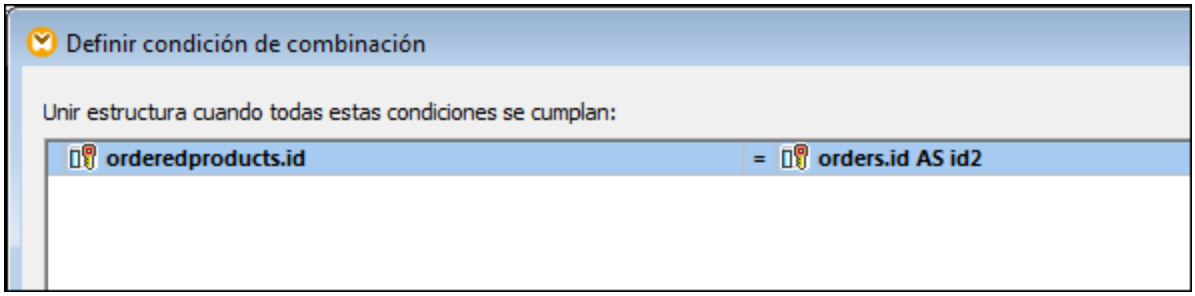
1. `orderedproducts`
2. `orders`
3. `products`
4. `users`

Este orden afecta a la presentación de las correspondientes estructuras en el cuadro de diálogo "Definir condición de combinación" (*imagen siguiente*), que se abre al hacer clic en el botón **Definir condición de**

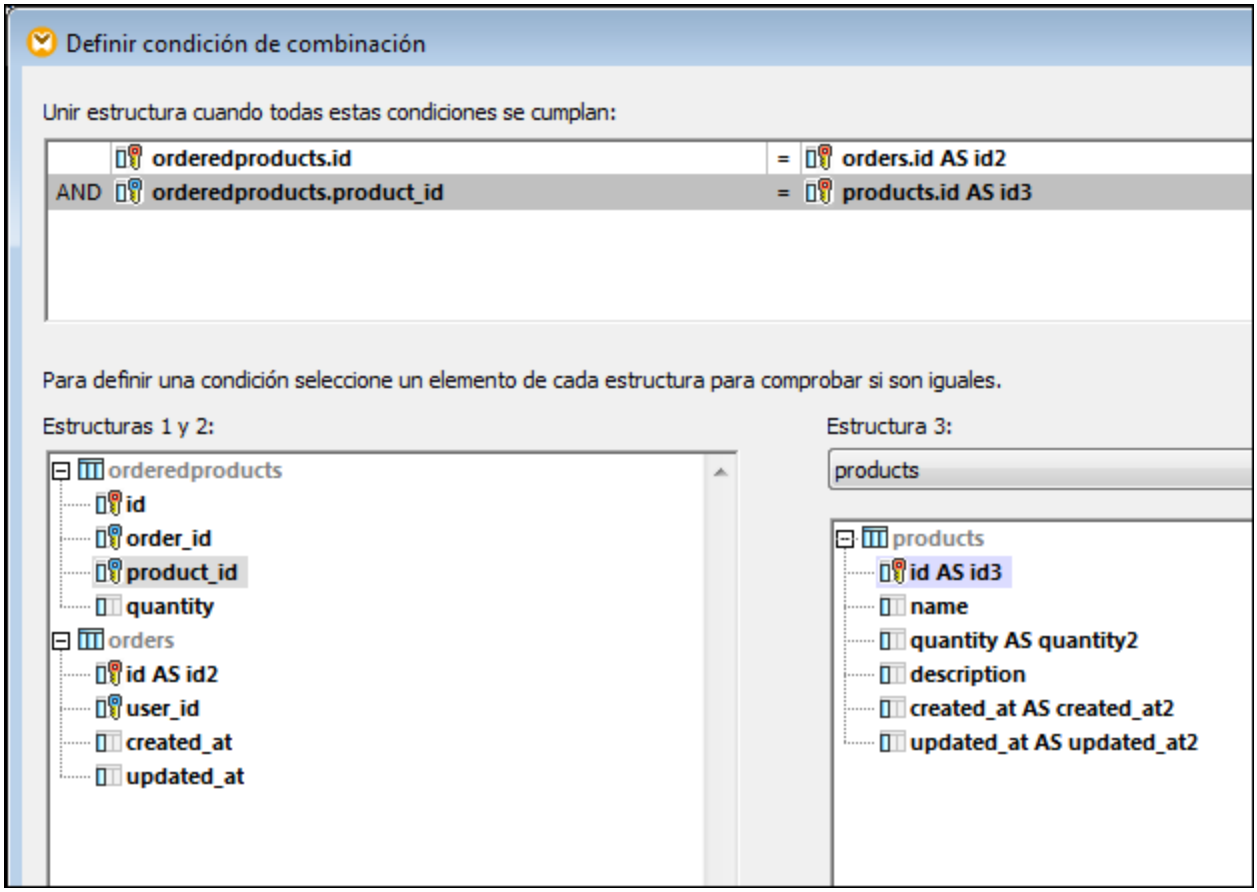
combinación . Es decir, la primera tabla (`orderedproducts`) aparece por defecto bajo *Estructura 1* y la tabla inmediatamente posterior a ella (`orders`) aparece bajo *Estructura 2*.



Para definir la primera condición de combinación haga clic en el elemento `order_id` del panel izquierdo y en el elemento `id` del panel derecho. Esto empareja los campos `orderedproducts.order_id` y `orders.id`:

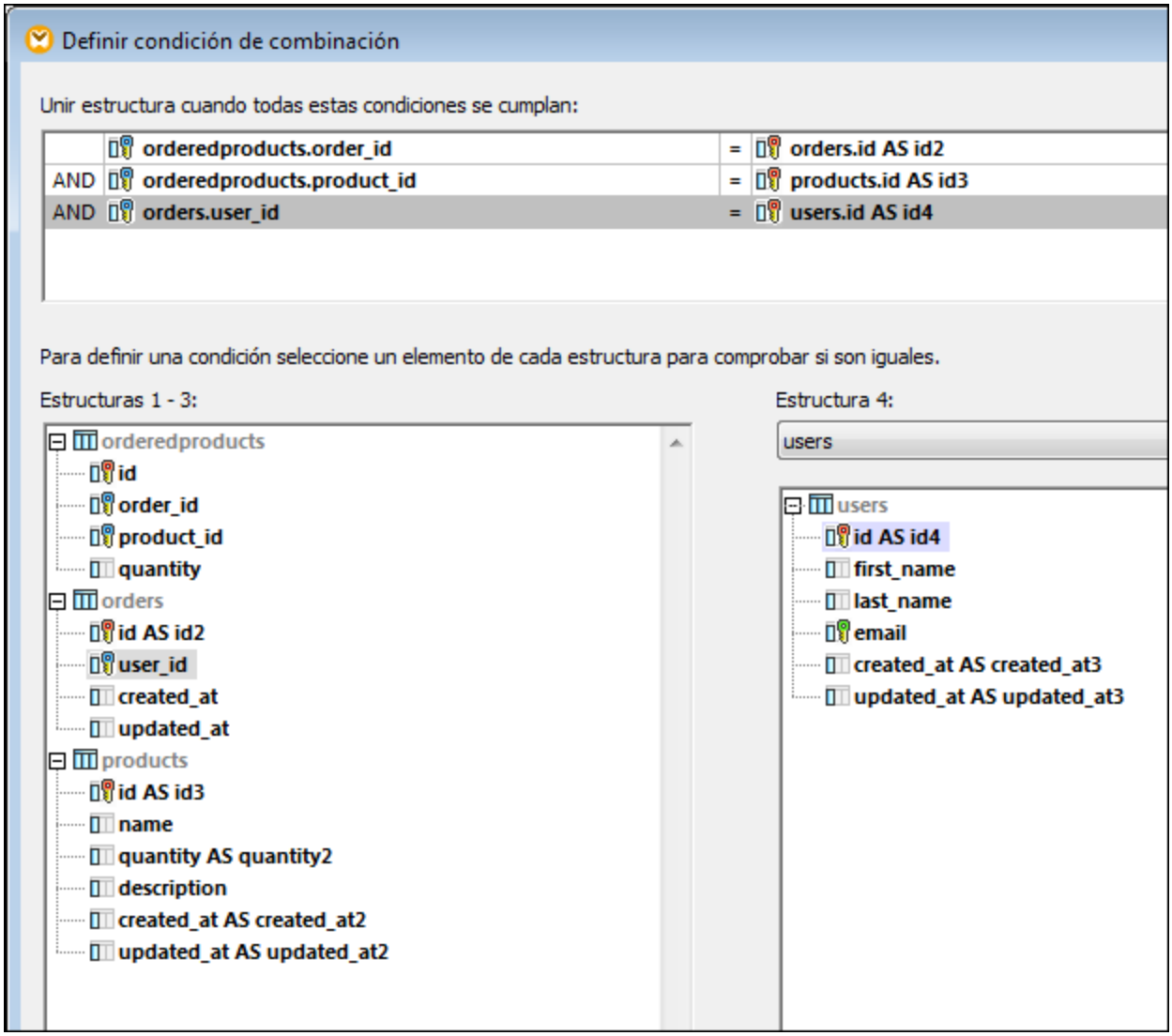


Hasta ahora hemos combinado solamente dos tablas. Para definir condiciones de combinación que comprendan una tercera tabla es necesario seleccionar la tabla correspondiente en la lista desplegable situada sobre el panel derecho. En este caso el panel izquierdo muestra todas las tablas que aparecen después de dicha tabla en el componente de combinación. Por ejemplo, si seleccionamos `products` en el panel derecho, el panel izquierdo presentará `orderedproducts` y `orders` (porque estas dos tablas aparecen antes de `products` en el componente de combinación). Ahora podemos emparejar campos de la tabla `products` con campos de tablas precedentes (en este caso `orderedproducts.product_id` y `products.id`).

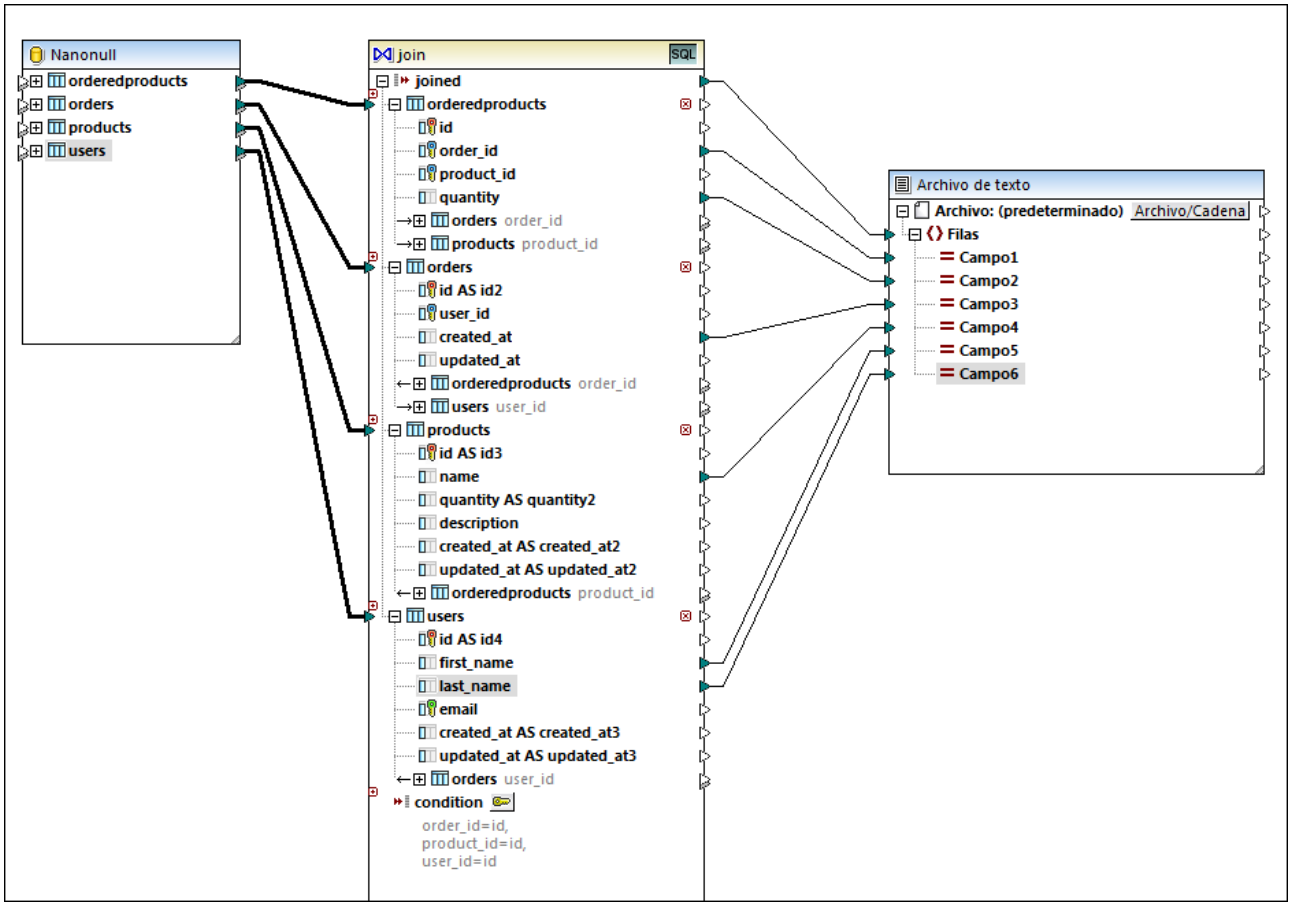


Para combinar la cuarta tabla (`users`) basta con seleccionar la tabla `users` en la lista desplegable y podremos emparejar los campos `orders.user_id` con los campos `users.id`.

© 2016-2022 Altova GmbH



Ahora que todas las condiciones de combinación están definidas, podremos crear asignaciones de datos entre los elementos de componente de combinación y los del componente de destino. Para terminar de diseñar la asignación de datos añada un componente CSV (*véase* *Archivos CSV y archivos de texto* 445) y conecte los elementos del componente de combinación con el componente CSV de destino tal y como muestra la imagen siguiente.



Esta asignación de datos genera un informe en formato CSV a partir de las cuatro tablas que participan en la combinación y que incluye:

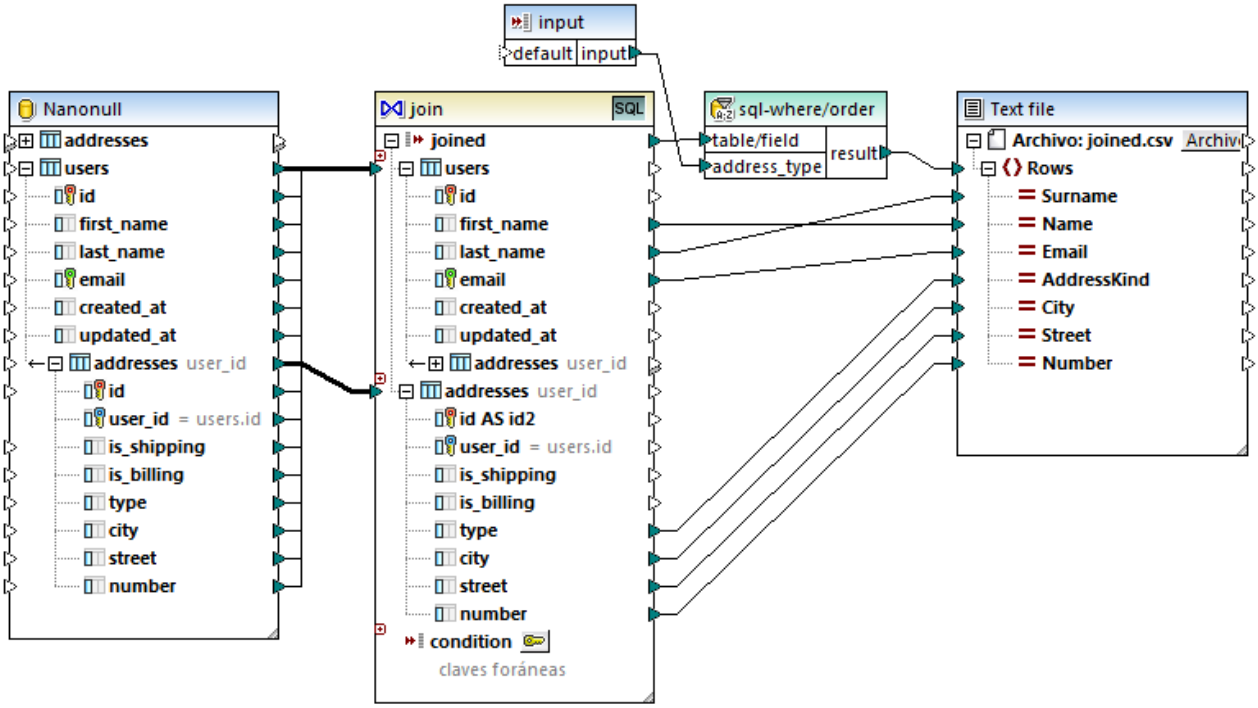
- el identificador del pedido (procedente de la tabla `orderedproducts`)
- la cantidad de artículos pedidos (procedente de la tabla `orderedproducts`)
- la hora a la que se realizó el pedido (procedente de la tabla `orders`)
- el nombre del producto que se pidió (procedente de la tabla `products`)
- el nombre y el apellido del usuario que hizo el pedido (procedente de la tabla `users`)

Todas las tablas de este ejemplo están combinadas con el modo del vínculo interno. Para más información sobre cómo cambiar el modo al vínculo externo izquierdo consulte Cambiar el modo de combinación 337.

4.2.4.3 Ejemplo: combinar tablas en modo SQL

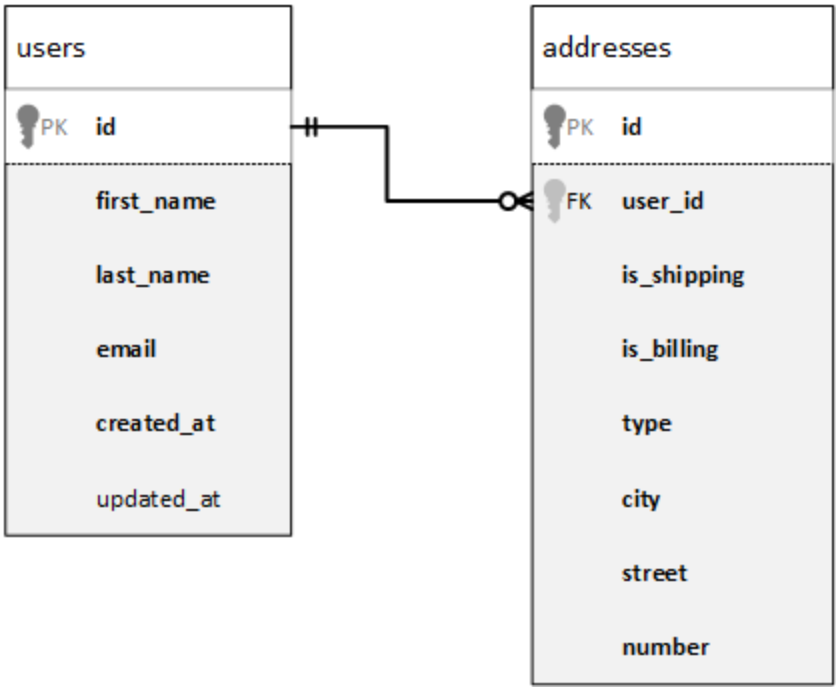
Este ejemplo demuestra cómo combinar datos de dos tablas de base de datos con ayuda de un componente de combinación de MapForce. La operación de combinación se lleva a cabo en modo SQL (*véase* *Combinaciones en modo SQL* 336). Recuerde que también se pueden combinar tres o más tablas de una forma muy parecida (*véase* *Ejemplo: crear un informe CSV a partir de varias tablas* 339).

El ejemplo que aparece a continuación se sirve de la asignación de datos **<Documentos>\Altova\MapForce2023\MapForceExamples\Tutorial\JoinDatabaseTables.mfd**.



JoinDatabaseTables.mfd

El objetivo de la asignación es combinar datos de dos tablas de base de datos de origen en un solo archivo CSV de destino. Como puede ver en el esquema que aparece a continuación, la primera tabla (`users`) almacena direcciones de personas y la segunda (`addresses`) almacena nombres y direcciones de correo de personas. Estas dos tablas están enlazadas por medio de un campo común (el campo `id` de `users` corresponde al campo `user_id` de `addresses`). En bases de datos este tipo de relación recibe el nombre de *relación de clave foránea*.



A continuación puede ver los datos que contienen las dos tablas.

© 2016-2022 Altova GmbH

users

id	first_name	last_name	email	created_at	updated_at
1	Marquita	Bailey	m.bailey@nanonull.com	2016-12-29 14:37:14	[NULL]
2	Sharda	Junker	s.junker@nanonull.com	2016-12-29 14:37:14	[NULL]
3	Totie	Rea	t.rea@nanonull.com	2016-12-29 14:37:14	[NULL]
4	Tobie	Hughey	t.hughey@nanonull.com	2016-12-29 14:37:14	[NULL]
5	Eadith	Lafreniere	e.lafreniere@nanonull.com	2016-12-29 14:37:14	[NULL]
6	Yehudi	Sponga	y.sponga@nanonull.com	2016-12-29 14:37:14	[NULL]
7	Laurianne	Huisman	l.huisman@nanonull.com	2016-12-29 14:37:14	[NULL]
8	Fred	Weinstein	f.weinstein@nanonull.com	2016-12-29 14:37:14	[NULL]
9	Mia	Dahill			
10	June	Leiker			
11	Benedick	Kocyk			
12	Andrej	Hildebrand			
13	Ariel	Phelan			
14	Matthaeus	Hulick			
15	Lotta	Mendes			
16	Jessey	Decelles			
17	Hilda	Lees			
18	Mark	Marzolla			
19	Dannie	Vignola			
20	Lanita	Krysiak			

addresses

id	user_id	is_shipping	is_billing	type	city	street	number
1	1	1	0	work	Bridgedell	Maple Lane	1
2	1	0	1	home	Bridgedell	Olive Street	6
3	3	1	1	home	Roseford	Evergreen Lane	34
4	4	1	1	work	Beardale	Route 44	9
5	6	1	1	home	Johnson City	Franklin Avenue	11
6	7	1	1	home	North Kingstown	Beach Alley	5
7	8	1	1	home	Merrowmeadow	Freybeach Street	85
8	10	1	1	work	Barrowedge	Penn Street	8
9	12	1	1	home	Elfville	Creek Road	3
10	13	1	1	home	Roseford	Bowman Ave.	853
11	14	1	1	work	Beardale	Iroquois Street	98
12	17	1	1	home	Bridgedell	Smith Road	7
13	18	1	0	home	Roseford	Wood Street	7
14	18	0	1	work	Johnson City	Thorne Lane	9677
15	20	1	1	home	Mechanicsville	Vine Street	9065

Cada registro de usuario de la tabla `users` puede tener cero o más direcciones en la tabla `addresses`. Por ejemplo, un usuario puede tener una dirección de tipo "home" o dos direcciones (una de tipo "home" y otra de tipo "work") o no tener dirección alguna.

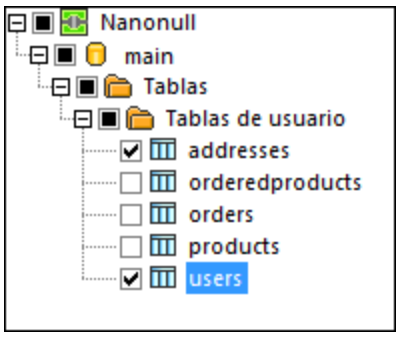
El objetivo de la asignación es recuperar todos los datos (nombre, apellido, correo electrónico, ciudad, calle y número) de todos los usuarios que tengan como mínimo una dirección en la tabla `addrresses`. También debería ser posible recuperar solamente cierto tipo de direcciones (p. ej. solo las de tipo "home" o solo las de tipo "work"). El tipo de dirección que se debe recuperar ("home" o "work") debe especificarse como parámetro en la asignación. Los registros recuperados deberán ordenarse alfabéticamente según el apellido de la persona.

Todo esto se conseguirá con ayuda de un componente de combinación.

Nota: los componentes de combinación no son el único método posible para combinar tablas o vistas de base de datos. También puede realizar combinaciones de datos de base de datos con ayuda de instrucciones SQL SELECT (*véase* *Instrucciones SQL SELECT como tablas virtuales* 359). La principal diferencia entre las instrucciones SQL SELECT y los componentes de combinación es que aquellas se escriben a mano y permiten mayor flexibilidad. Los componentes de combinación son una alternativa sencilla para quienes no dominen SQL.

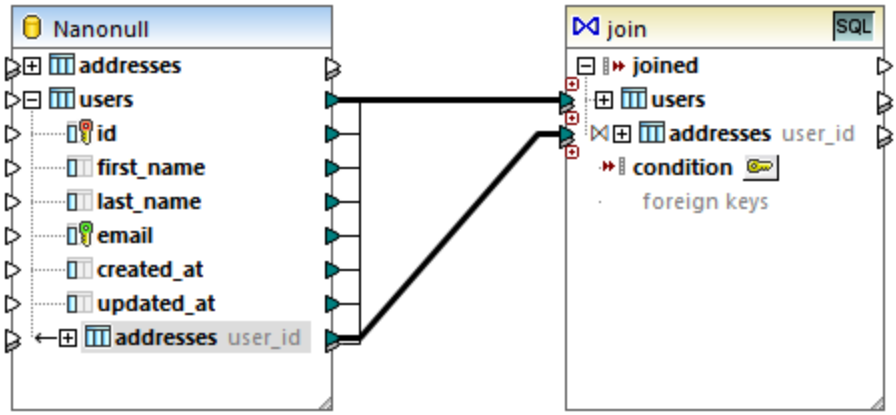
Paso nº1: agregar la base de datos de origen

1. En el menú **Insertar** haga clic en el comando **Base de datos** (o en el botón **Insertar base de datos** de la barra de herramientas).
2. Seleccione el tipo de base de datos *SQLite* y haga clic en **Siguiente**.
3. Navegue hasta el archivo **Nanonull.sqlite** de la carpeta **<Documentos>\Altova\MapForce2023\MapForceExamples\Tutorial** y haga clic en **Conectarse**.
4. Cuando la aplicación lo solicite, seleccione las tablas `addresses` y `users`.



Paso nº2: agregar el componente de combinación

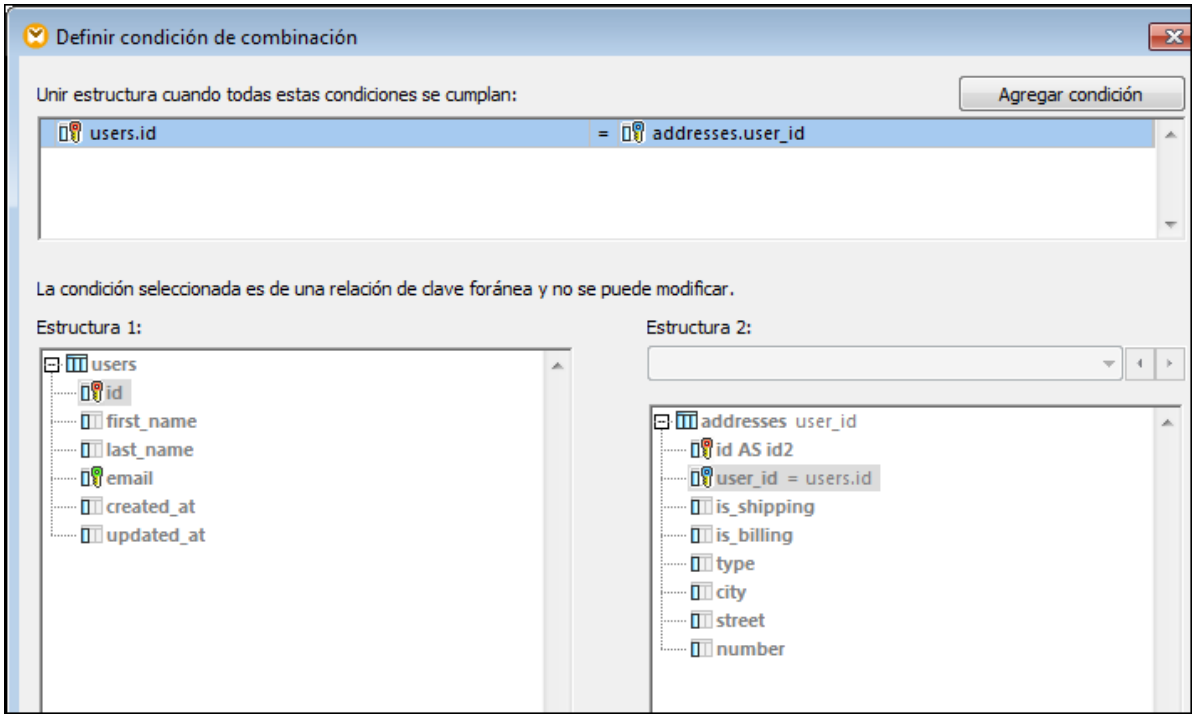
1. En el menú **Insertar** haga clic en **Combinación** (o en el botón **Insertar componente de combinación** de la barra de herramientas).
2. Dibuje una conexión entre la tabla `users` y la primera entrada del componente de combinación.
3. Expanda la tabla `users` y dibuje una conexión entre la tabla `addresses` (secundaria de `users`) y la segunda entrada del componente de combinación. El botón sirve para añadir más tablas si las necesita. No obstante, en este ejemplo solamente combinaremos dos tablas.



Nota: también se puede agregar la conexión desde la tabla `addresses` directamente (la que no es tabla secundaria de `users`). Sin embargo, en el ejemplo que nos ocupa las condiciones de combinación deberían definirse a mano (*véase* *Agregar condiciones de combinación* 519). Para este ejemplo asegúrese de crear las conexiones que aparecen en la imagen anterior. Eso garantizará que la condición de combinación necesaria se crea de forma automática.

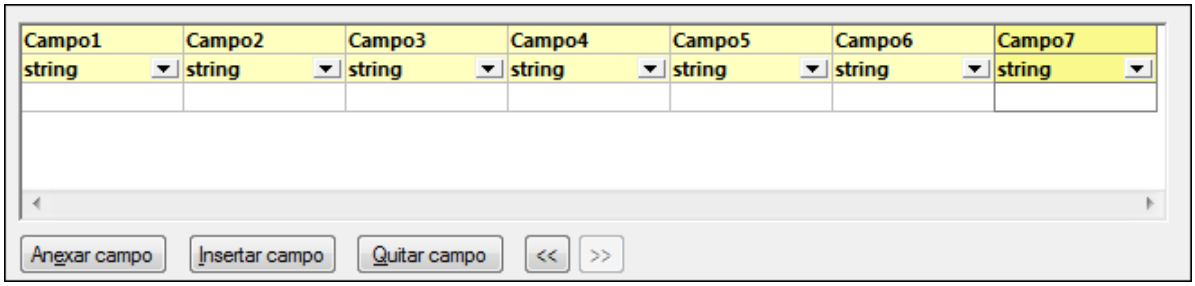
 © 2016-2022 Altova GmbH

4. Haga clic en el botón **Definir condición de combinación** del componente de combinación. Observe que la condición de combinación se crea de forma automática (`users.id = addresses.user_id`).

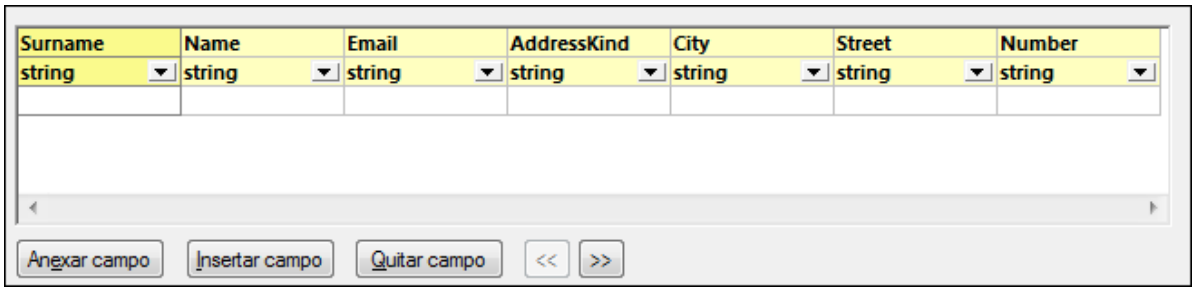


Paso nº3: agregar el componente CSV de destino

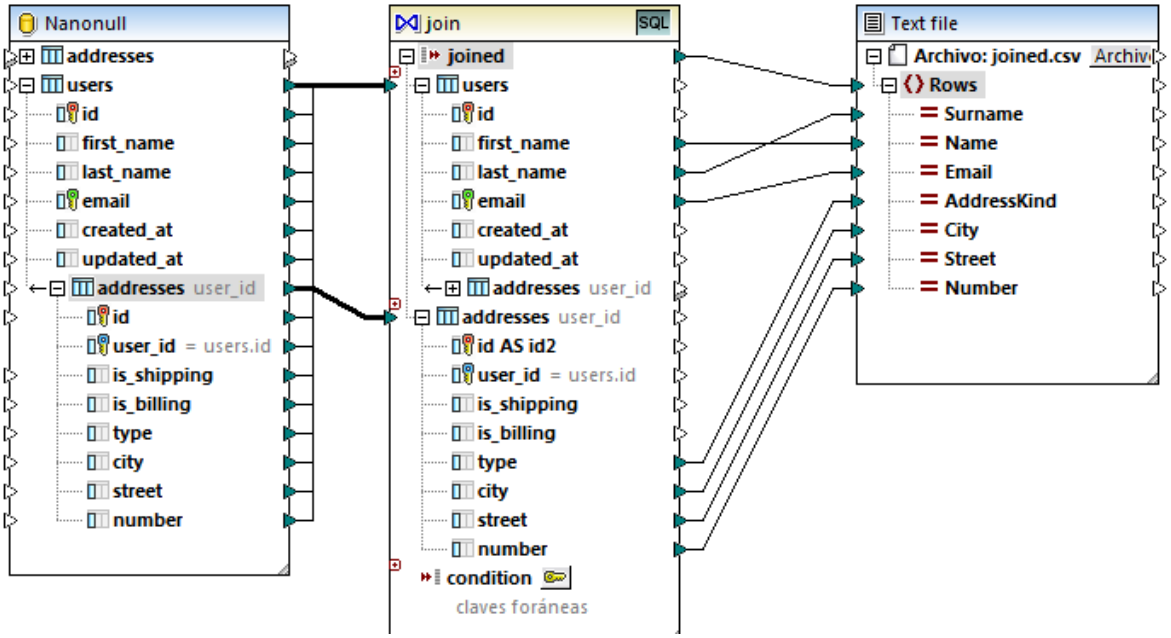
1. En el menú **Insertar** haga clic en **Archivo de texto** (o en el botón **Insertar archivo de texto** de la barra de herramientas).
2. Cuando la aplicación le pida que elija un modo de procesamiento, seleccione la opción *Utilizar procesamiento básico...* en el cuadro de diálogo.
3. Haga clic en **Anexar campo** varias veces para crear siete campos CSV. Deje la demás opciones del cuadro de diálogo como están.



4. Ahora haga doble clic en la celda que contiene el título de los campos recién creados y modifique el nombre de las celdas (*imagen siguiente*).



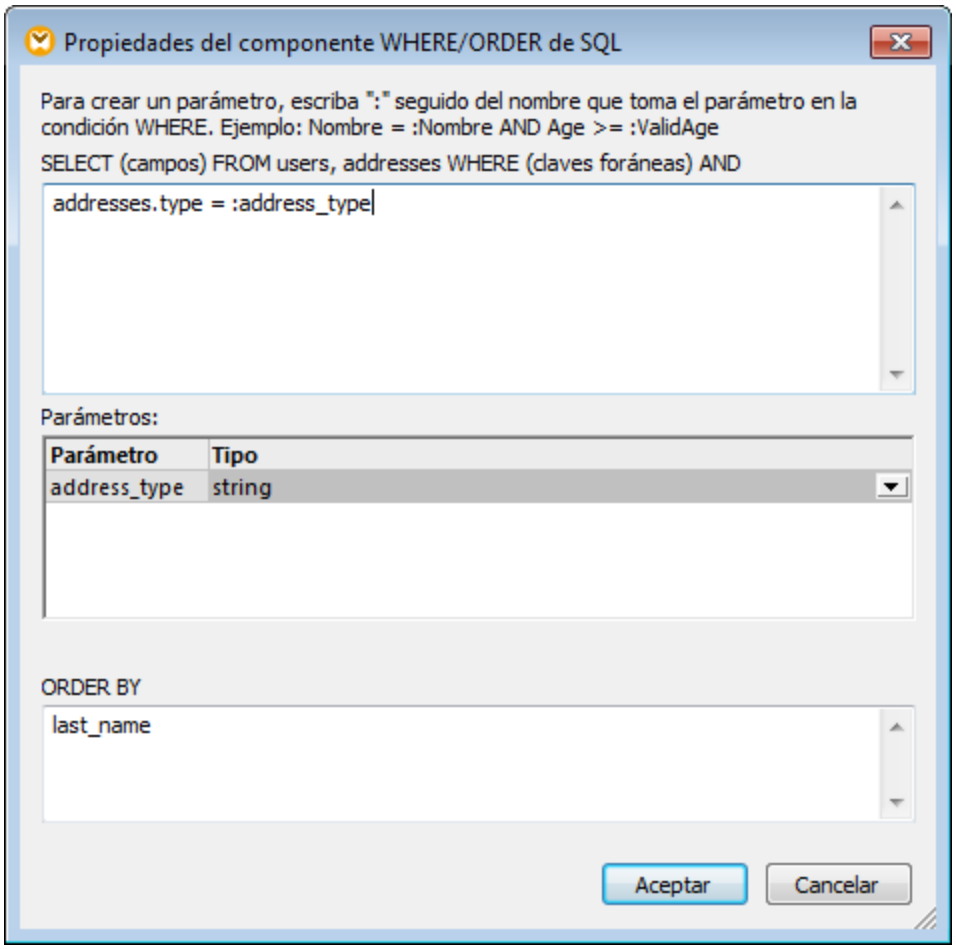
5. Dibuje las conexiones que pueden verse en la siguiente imagen entre el componente de combinación y el componente CSV. La conexión que une el elemento `joined` del componente de combinación y el elemento `Rows` del componente CSV de destino da la orden de *crear tantos registros (filas) en el destino como registros existan que cumplan la condición de combinación*.



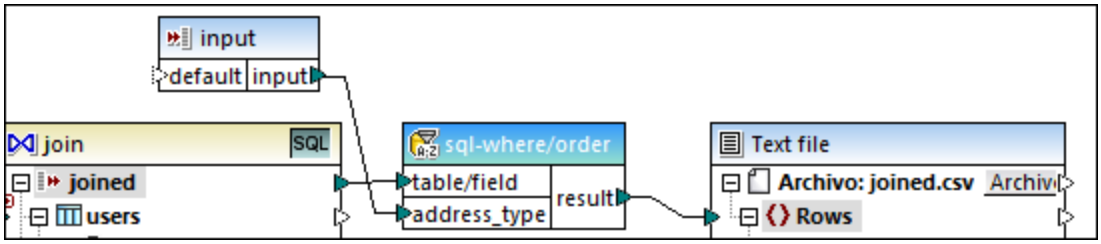
Paso nº4: agregar la condición WHERE/ORDER de SQL y el parámetro de entrada

1. Haga clic con el botón derecho en la conexión que une el elemento `joined` del componente de combinación y el elemento `Rows` del componente CSV de destino y después seleccione **Insertar WHERE/ORDER de SQL**.
2. Introduzca las cláusulas WHERE y ORDER BY que pueden verse en la siguiente imagen.

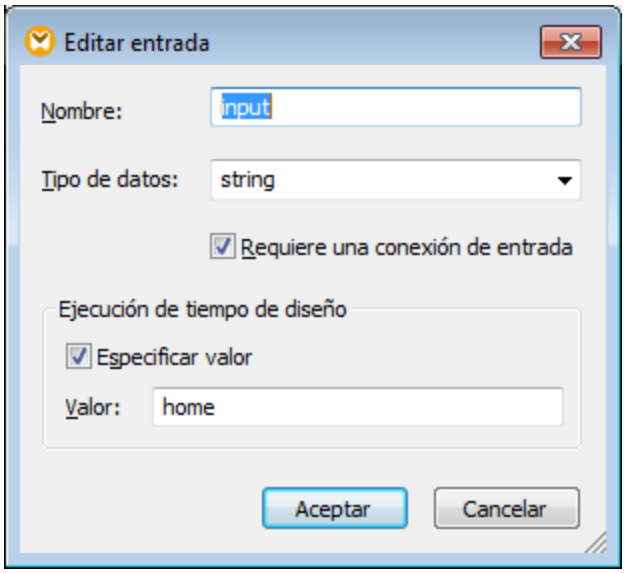
 © 2016-2022 Altova GmbH



3. En la asignación añada un componente de entrada (con el comando de menú **Insertar | Insertar componente de entrada**) y conecte su resultado con el parámetro `address_type` creado en el paso anterior.



4. Haga doble clic en el componente de entrada y configúrelo como muestra la siguiente imagen. Para poder consultar la vista previa de resultados de la asignación en MapForce es necesario tener un valor en tiempo de diseño (en este caso "home"). Si quiere que la vista previa recupere las direcciones de tipo "work", reemplace este valor con "work".



Funcionamiento de esta asignación de datos

La condición de combinación creada de forma automática en el paso nº2 se asegura de que en el destino solo se copien los registros que cumplan la condición de combinación `users.id = addresses.user_id`. La condición de combinación se añadió automáticamente porque las dos tablas están unidas por una relación de clave foránea y porque se dibujaron las correspondientes conexiones de asignación de datos (*véase* *Gestionar relaciones de BD* 258). Aunque en nuestro caso concreto no sea relevante, también se pueden definir condiciones de combinación a mano (*véase* *Ejemplo: crear un informe CSV a partir de varias tablas* 339).

Las dos tablas de origen son de la misma base de datos y del mismo componente así que esta combinación puede hacerse en modo SQL SQL. Como el modo SQL está habilitado, la operación de combinación la lleva a cabo la base de datos y no MapForce. En otras palabras, MapForce genera una instrucción de combinación interna INNER JOIN y la envía a la base de datos para que la ejecute.

El componente WHERE/ORDER de SQL que se añadió en el paso nº4 permite filtrar (recuperar las direcciones de tipo "home" o "work") y ordenar el conjunto de registros. Tenga en cuenta que la cláusula WHERE creó un parámetro llamado **`:address_type`** de tipo `string`. Este parámetro permite suministrar el tipo de dirección (home o work) desde la asignación. Para más información consulte Filtrar y ordenar datos de BD (SQL WHERE/ORDER) 352.

Por último, el componente de entrada permite suministrar el valor de parámetro actual cuando se ejecuta la asignación. Recuerde que, cuando la asignación se ejecuta fuera de MapForce (p. ej. cuando se ejecuta con MapForce Server en otro equipo), la entrada debe suministrarse en tiempo de ejecución como parámetro de la línea de comandos, en cuyo caso se omitirá el valor mencionado. Para más información consulte Pasar parámetros a la asignación 477.

4.2.5 Filtrar y ordenar datos de BD (SQL WHERE/ORDER)

Cuando necesite filtrar y ordenar datos de base de datos, utilice un componente **WHERE/ORDER de SQL/NoSQL**. Esto le permitirá introducir a mano (desde la interfaz gráfica del usuario de MapForce) una cláusula **WHERE de SQL** que filtre los datos. También tiene la opción de especificar una cláusula ORDER BY

 © 2016-2022 Altova GmbH

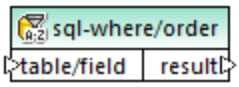
si desea ordenar el conjunto de registros según cierto campo de la base de datos, en orden ascendente o descendente.

El componente **WHERE/ORDER de SQL/NoSQL** debe estar conectado a una tabla o campo de un componente de base de datos. También puede conectarse a un componente de combinación cuando se necesita filtrar un conjunto de registros o registros combinados previamente (*véase* *Combinar datos de BD* 335).

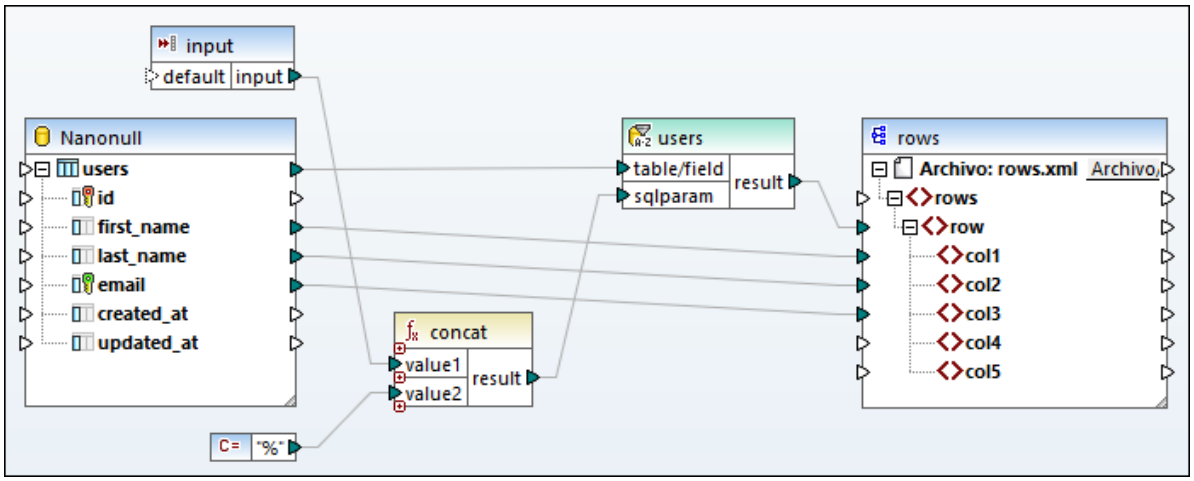
Agregar un componente WHERE/ORDER de SQL/NoSQL en la asignación

Para añadir un componente **WHERE/ORDER de SQL/NoSQL** a la asignacón:

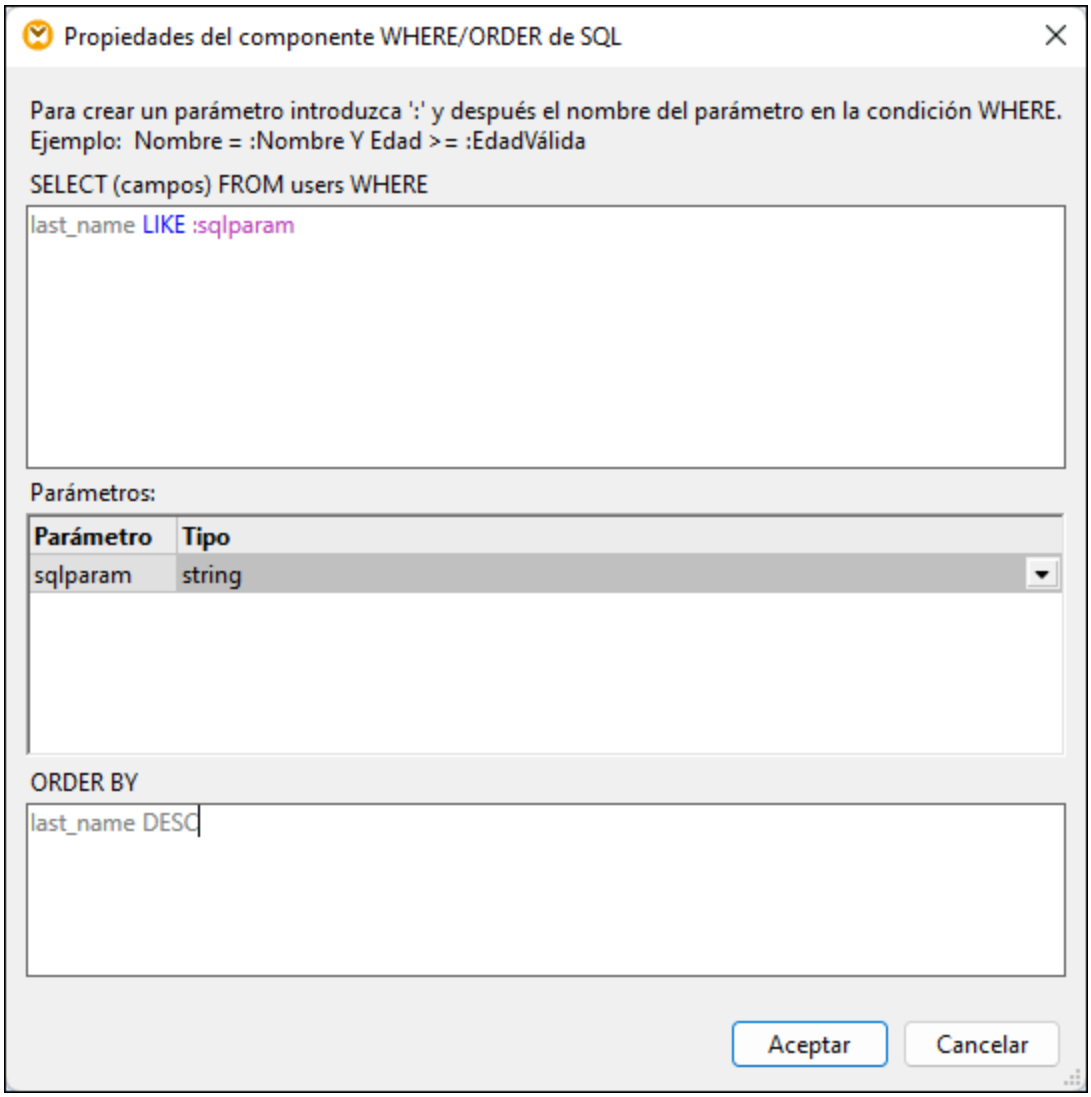
1. En el menú **Insertar** seleccione el comando **WHERE/ORDER de SQL/NoSQL**. Este componente presenta esta estructura predeterminada:



2. Conecte una tabla o un campo de la tabla de base de datos de origen con el nodo `table/field` del componente **WHERE/ORDER de SQL/NoSQL**. Por ejemplo, la asignación de ejemplo **`FilterDatabaseRecords.mfd`** de la carpeta **`<Documentos>\Altova\MapForce2023\MapForceExamples\Tutorial\`** incluye un componente WHERE/ORDER de SQL que sirve para filtrar los registros de la tabla de origen `Person` cuyo apellido empiece por la letra B.



3. Haga doble clic en el título del componente WHERE/ORDER de SQL (o clic con el botón derecho y después clic en **Propiedades**). Esto abre el cuadro de diálogo "Propiedades del componente WHERE/ORDER de SQL".



4. Escriba la cláusula WHERE de SQL en el cuadro de texto superior. También tiene la opción de introducir la cláusula ORDER BY. En la imagen anterior puede ver las cláusulas WHERE y ORDER BY definidas en la asignación **DB_PhoneList.mfd**. Encontrará más ejemplos en el apartado Crear cláusulas WHERE y ORDER BY 356.

Suministrar parámetros a componentes WHERE/ORDER de SQL

El componente componente WHERE/ORDER de SQL de la asignación **DB_PhoneList.mfd** define esta cláusula WHERE:

```
Last LIKE :Name
```

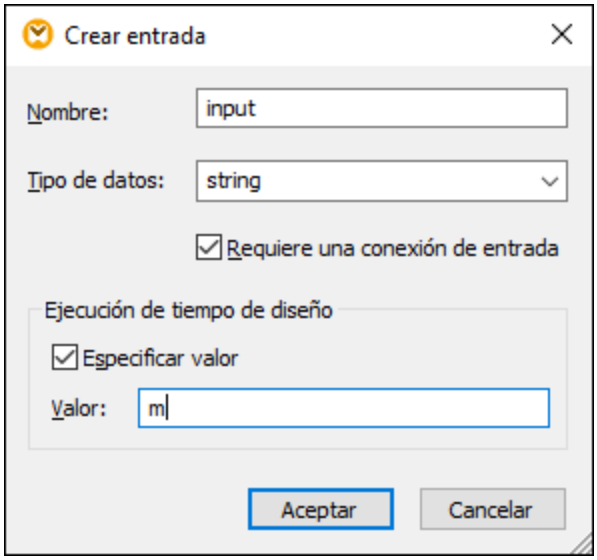
"Last" se refiere al nombre de un campo de la BD de la tabla que está conectada. "LIKE" es un operador SQL y ":Name" crea un parámetro llamado "Name" en la asignación.

 © 2016-2022 Altova GmbH

Los parámetros son opcionales en los componentes WHERE/ORDER de SQL. Son prácticos si se quiere pasar un valor a la cláusula WHERE desde la asignación. La misma cláusula WHERE se puede escribir sin parámetros de la siguiente manera:

```
Last LIKE "B%"
```

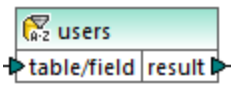
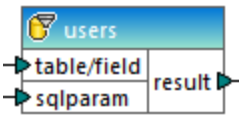
Esto recuperaría todas las personas cuyo apellido empiece con la letra B. Para poder hacer esta consulta más flexible añadimos un parámetro en lugar de "B%". Esto nos permitirá usar cualquier letra desde la asignación (p. ej. "C" para recuperar las personas cuyo apellido empiece por la letra "C" con solo cambiar una constante o un parámetro de entrada).

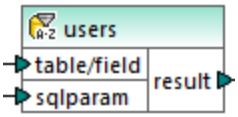


En la asignación, el carácter comodín SQL % viene dado por una constante y después se concatena con el valor del parámetro con ayuda de la función **concat**. La ventaja es que no es necesario teclear comodines SQL en la línea de comandos si se ejecuta esta asignación en otro entorno, como MapForce Server.

Aspecto de los componentes WHERE/ORDER de SQL

Algo importante que se debe tener en cuenta sobre los componentes WHERE/ORDER de SQL es que su aspecto cambia dependiendo de su configuración. Esto permite saber a simple vista qué papel desempeña el componente WHERE/ORDER de SQL en la asignación. Por ejemplo:

	Se definió una cláusula WHERE.
	Se definió una cláusula WHERE con un parámetro. El parámetro "Name" aparece debajo del nodo "table/field".

	Se definió una cláusula WHERE con un parámetro y una cláusula ORDER BY. El icono A-Z indica que se definió una ordenación.

Con solo pasar el puntero del ratón por encima del título del componente podrá ver qué cláusulas se han definido exactamente.

4.2.5.1 Crear cláusulas WHERE y ORDER BY

Tras añadir un componente WHERE/ORDER de SQL a la asignación deberá añadir una condición WHERE para especificar cómo se deben filtrar los datos que están conectados al componente exactamente. la condición WHERE debe introducirse en el cuadro de diálogo "Propiedades del componente WHERE/ORDER de SQL".

En MapForce las condiciones WHERE se escriben de forma similar a como se escriben las mismas cláusulas SQL fuera de MapForce, usando la sintaxis que corresponda al dialecto SQL de la base de datos. Por ejemplo, puede usar operadores, comodines, subselecciones o funciones de agregado. Para crear parámetros que se puedan pasar desde la asignación introduzca un carácter de dos puntos (:) seguido del nombre del parámetro-

Nota: cuando termine de escribir la cláusula WHERE y haga clic en **Aceptar**, MapForce validará la integridad de la instrucción SQL final y, si se detectan errores sintácticos, emitirá mensajes en un cuadro de diálogo.

En esta tabla puede ver los operadores más frecuentes que se pueden usar en la cláusula WHERE:

Operador	Descripción
=	Igual
<>	No igual
<	Menor que
>	Mayor que
>=	Mayor o igual
<=	Menor o igual
IN	Recupera un valor conocido de una columna
LIKE	Busca un patrón determinado
BETWEEN	Busca dentro de un intervalo

Use el comodín `%` (porcentaje) para representar cualquier número de caracteres de un patrón. Por ejemplo, para recuperar todos los registros que empiecen por R de un campo llamado `lastname`, debe usar esta expresión:

 © 2016-2022 Altova GmbH

```
lastname = "%r"
```

Cuando consulte bases de datos compatibles con la ordenación y consulta de datos de base de datos XML (p. ej. IBM DB2, Oracle, SQL Server), podrá usar funciones y palabras clave XML que correspondan a dicho tipo de base de datos. Por ejemplo:

```
xmlexists('$c/Client/Address[zip>"55116"]' passing USER.CLIENTS.CONTACTINFO AS "c")
```

Consulte también el apartado Ejemplo: extraer datos de columnas tipo XML de IBM DB2 377.

Además, si quiere ordenar el conjunto de registros recuperado usando un campo como criterio de ordenación, añada una cláusula ORDER BY en el cuadro correspondiente del cuadro de diálogo "Propiedades del componente WHERE/ORDER de SQL". Para ordenar los registros usando varios campos como criterio de ordenación, separe los nombres de campo por comas. Para cambiar el orden (ascendente/descendente) use las palabras clave `ASC` y `DESC`. Por ejemplo, la cláusula ORDER BY que aparece a continuación recupera registros ordenados usando el campo `lastname` y después el campo `firstname` como criterios de ordenación en orden descendente:

```
lastname, firstname DESC
```

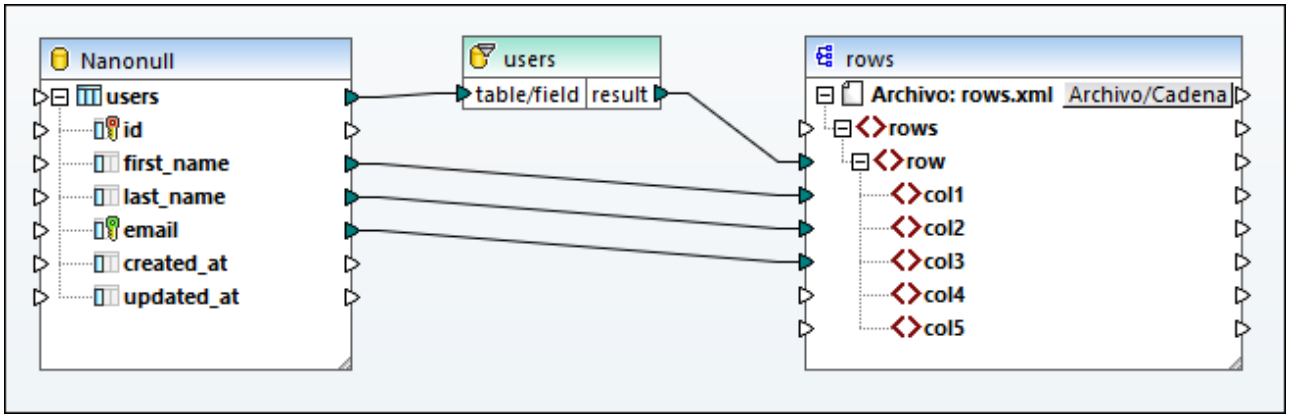
Ejemplo nº1

La condición WHERE que aparece a continuación está vinculada a la tabla `users` del componente de BD **Nanonull.sqlite**. Recupera los registros en los que `last_name` es mayor que la letra "M". En otras palabras, todos los nombres a partir del usuario "Marzolla".

```
last_name > "M"
```

Observe las conexiones:

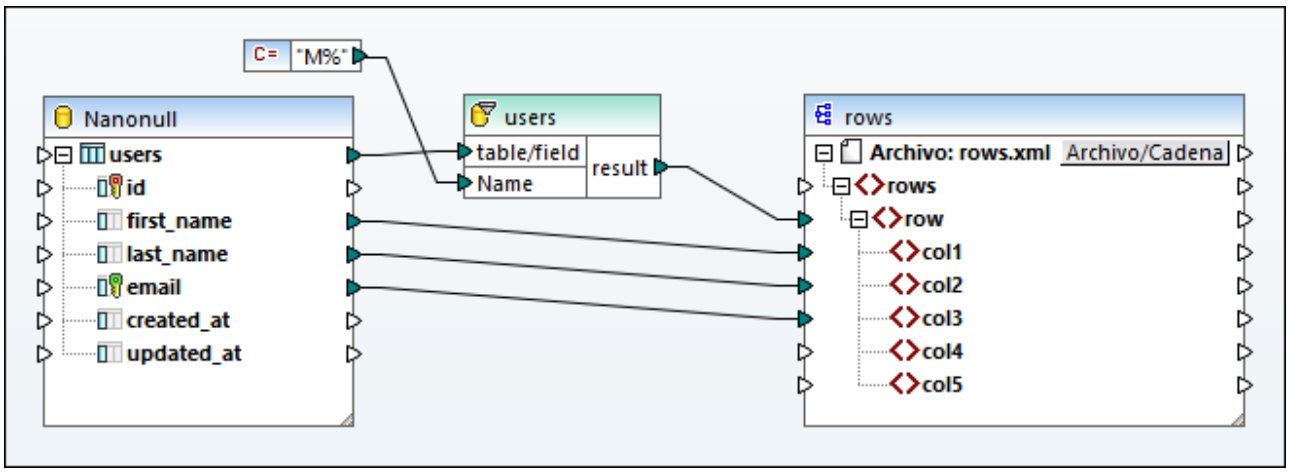
- La conexión **table/field** tiene origen en la tabla en la que quiere realizar la consulta, "users" en este caso.
- La salida **result** está conectada a un elemento "primario" de los campos en los que se consulta/filtra, en este caso al elemento **row**.



Ejemplo nº2

La condición WHERE que aparece a continuación crea un parámetro **Name** que aparece en el componente WHERE/ORDER de SQL de la asignación.

```
last_name LIKE :param
```



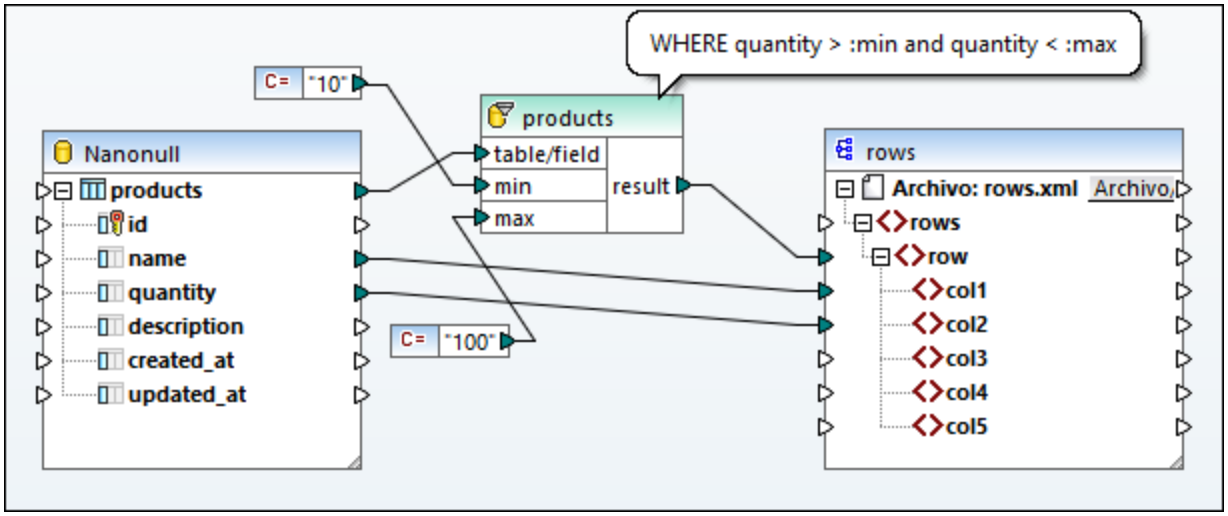
El componente de constante `%M` aporta el valor del parámetro **Name**. El comodín % denota cualquier número de caracteres. Esto hace que la asignación busque un parámetro en la columna "last_name" (todos los apellidos que acaben en "M").

Ejemplo nº3

La condición WHERE que aparece a continuación crea dos parámetros: `min` y `max`. Los valores actuales de `quantity` se comparan con estos dos parámetros. Los valores máximo (`max`) y mínimo (`min`) se indican por medio de constantes.

```
quantity > :min and quantity < :max
```

 © 2016-2022 Altova GmbH



La condición WHERE de este ejemplo también se podría escribir con el operador BETWEEN:

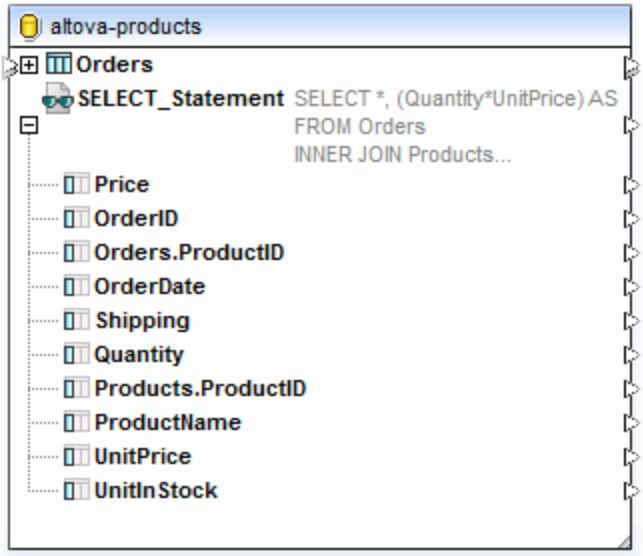
```
quantity BETWEEN :min and :max
```

4.2.6 Instrucciones SQL SELECT como tablas virtuales

MapForce permite crear instrucciones SELECT de SQL con parámetros dentro de componentes de BD. Se trata de estructuras tipo tabla que contienen los campos del conjunto de resultados generado por la instrucción SELECT. Estas estructuras se pueden usar como origen de datos de la asignación, igual que cualquier otra tabla o vista definida en la BD.

- Cuando use combinaciones internas/externas (INNER/OUTER JOIN) en la instrucción SELECT, en el componente se incluyen los campos de todas las tablas.
- Las expresiones con nombres de correlación (que usan la palabra clave SQL "AS") también aparecen en el componente como elementos asignables.
- Las vistas de la BD también se pueden usar en la cláusula FROM.
- Las instrucciones SELECT pueden tener parámetros que usan la misma sintaxis que el componente WHERE/ORDER de SQL 352.

Las instrucciones SQL SELECT son compatibles con los lenguajes C++, C#, Java y BUILT-IN, siempre que no contengan parámetros de entrada. Las instrucciones SQL SELECT con parámetros de entrada solo son compatibles con el lenguaje de transformación BUILT-IN.



El número de líneas visibles de la instrucción SELECT se puede configurar. Para definir qué número de líneas desea ver en el componente abra la pestaña *Generales* del cuadro de diálogo "Opciones" (**Herramientas | Opciones**) e introduzca el número de líneas en el grupo *Visualización de asignaciones*.

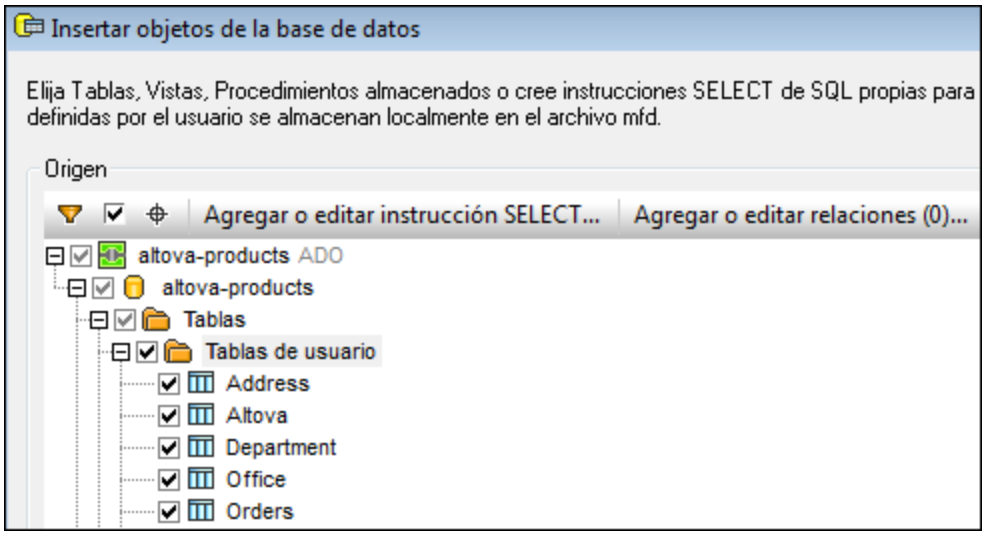
4.2.6.1 Crear instrucciones SELECT

Puede crear instrucciones SELECT en cualquier asignación de datos que tenga un componente de base de datos. Si la asignación no contiene todavía ninguna base de datos, primero deberá añadir una (*véase* *Conectarse a una base de datos* 161). Para este ejemplo haga clic en el comando de menú **Insertar | Insertar base de datos** y, siguiendo los pasos del Asistente para la conexión, conéctese al archivo de base de datos **altova-products.mdb** de la carpeta `<Documentos>\Altova\MapForce2023\MapForceExamples\Tutorial\`.

Para crear una instrucción SELECT:

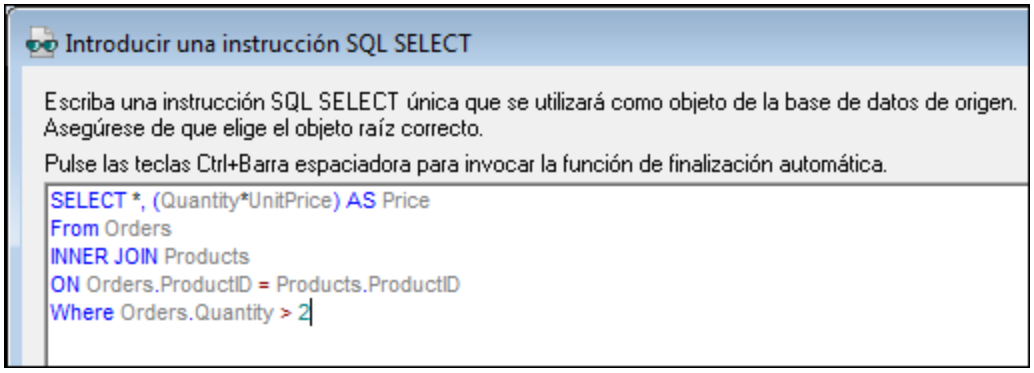
1. Haga clic con el botón derecho en el título del componente de base de datos y elija el comando **Agregar, quitar o editar objetos de la base de datos**. Si lo prefiere, también puede seleccionar el componente de base de datos y después hacer clic en el comando de menú **Componente | Agregar, quitar o editar objetos de la base de datos**.

 © 2016-2022 Altova GmbH

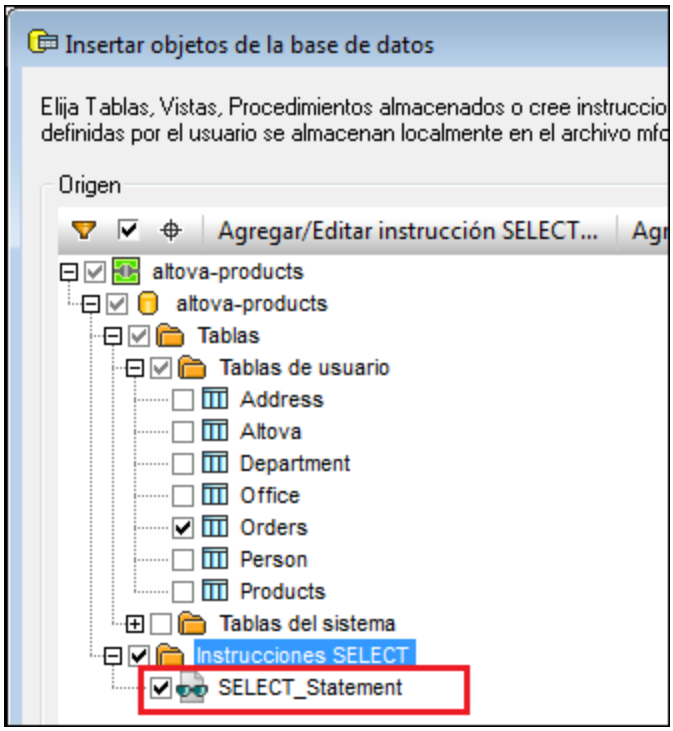


2. Ahora tiene dos opciones:
   - Para generar la instrucción SELECT a partir de una tabla actual: haga clic con el botón derecho en cualquier tabla y seleccione **Generar y agregar una instrucción SELECT...** en el menú contextual. Más tarde podrá editar la instrucción generada.
   - Para escribir una instrucción SELECT personalizada, haga clic en el botón **Agregar/Editar instrucción SELECT**.
3. Edite o cree la instrucción en función de sus requisitos. Por ejemplo, la instrucción SELECT que aparece a continuación es válida para el archivo **altova-products.mdb** de la carpeta `<Documentos>\Altova\MapForce2023\MapForceExamples\Tutorial\`. El campo **Price** es el producto de los dos campos (**Quantity** y **UnitPrice**) y se declara como nombre de correlación (`AS Price`).

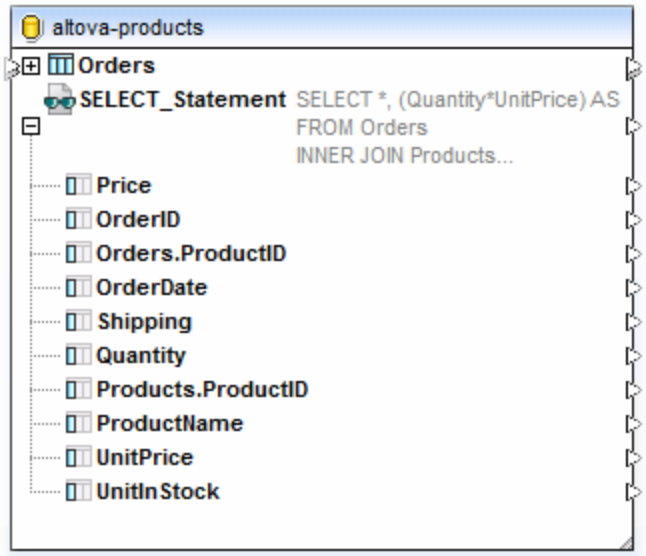
```
SELECT *, (Quantity*UnitPrice) AS Price
From Orders
INNER JOIN Products
ON Orders.ProductID = Products.ProductID
Where Orders.Quantity > 2
```



4. Ahora haga clic en el botón **Agregar instrucción SELECT**. Observe que ahora la instrucción SELECT aparece como objeto de la base de datos, como si se tratase de una tabla, una vista o un procedimiento.



5. Para terminar haga clic en **Aceptar**. La instrucción SELECT aparecerá también en el componente de base de datos y podrá crear asignaciones de datos desde cualquier campo que devuelva la consulta SELECT.



Notas:

- Todas las expresiones calculadas en la instrucción SELECT deben tener un nombre de correlación único (como "AS Price" en el ejemplo anterior). De lo contrario, no estarán disponibles como elemento asignable.

 © 2016-2022 Altova GmbH

- Si se conecta a una base de datos Oracle o IBM DB2 con JDBC, la instrucción SELECT no puede terminar con un punto y coma.

Para eliminar instrucciones SELECT añadidas previamente:

1. Haga clic con el botón derecho en el título del componente de base de datos y elija el comando **Agregar, quitar o editar objetos de la base de datos**.
2. Haga clic con el botón derecho en la instrucción SELECT que desea eliminar y seleccione el comando **Eliminar instrucción SELECT**.

4.2.6.2 Ejemplo: SELECT con parámetros

Con este ejemplo aprenderá a crear una asignación de MapForce que lee datos de una base de datos Microsoft Access y los escribe en un archivo CSV. En concreto, en este ejemplo utilizamos una consulta SELECT de base de datos personalizada con un parámetro. La instrucción SELECT combina datos de varias tablas diferentes. Y después los resultados se envían a la asignación para seguir procesando los datos.

Este ejemplo viene acompañado de un diseño de asignación, cuya ruta de acceso es **<Documentos>\Altova\MapForce2023\MapForceExamples\Tutorial\select-component.mfd**. Puede abrir este diseño y analizarlo o seguir los pasos que aparecen a continuación para crear el mismo diseño desde cero.

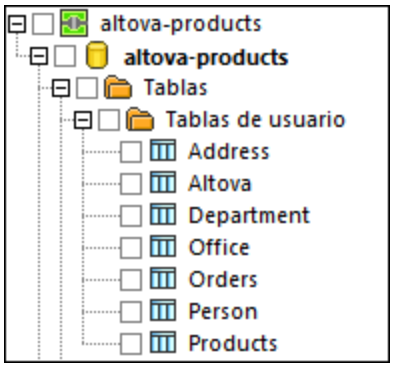
Aunque para este ejemplo usamos una base de datos Microsoft Access, el proceso es idéntico con otras bases de datos. Consulte la sección Conectarse a una base de datos 161 para obtener información sobre cómo conectarse a los distintos tipos de base de datos.

El objetivo de nuestro diseño de asignación es:

1. Seleccionar en la base de datos solamente aquellos pedidos cuyo número de productos pedidos supere determinado valor. Este valor vendrá dado por un parámetro. Para conseguir nuestro objetivo crearemos una instrucción SELECT que tome un parámetro de entrada.
2. En la base de datos Access el formato de fecha es `YYYY-MM-DD HH-MI-SS`. En el archivo CSV de destino la parte de hora debe eliminarse para que el formato sea `YYYY-MM-DD`. Para conseguir nuestro objetivo usaremos la función `date-from-datetime` 731 de MapForce.
3. El archivo CSV resultante debe llamarse **OrdersReport.csv**.

Paso n°1: agregar la estructura SELECT

1. En el menú **Insertar** seleccione el comando **Base de datos**.
2. Seleccione **Microsoft Access (ADO)** y siga los pasos del asistente para conectarse a la base de datos **altova-products.mdb** que está en la carpeta `<Documentos>\Altova\MapForce2023\MapForceExamples\Tutorial\`.

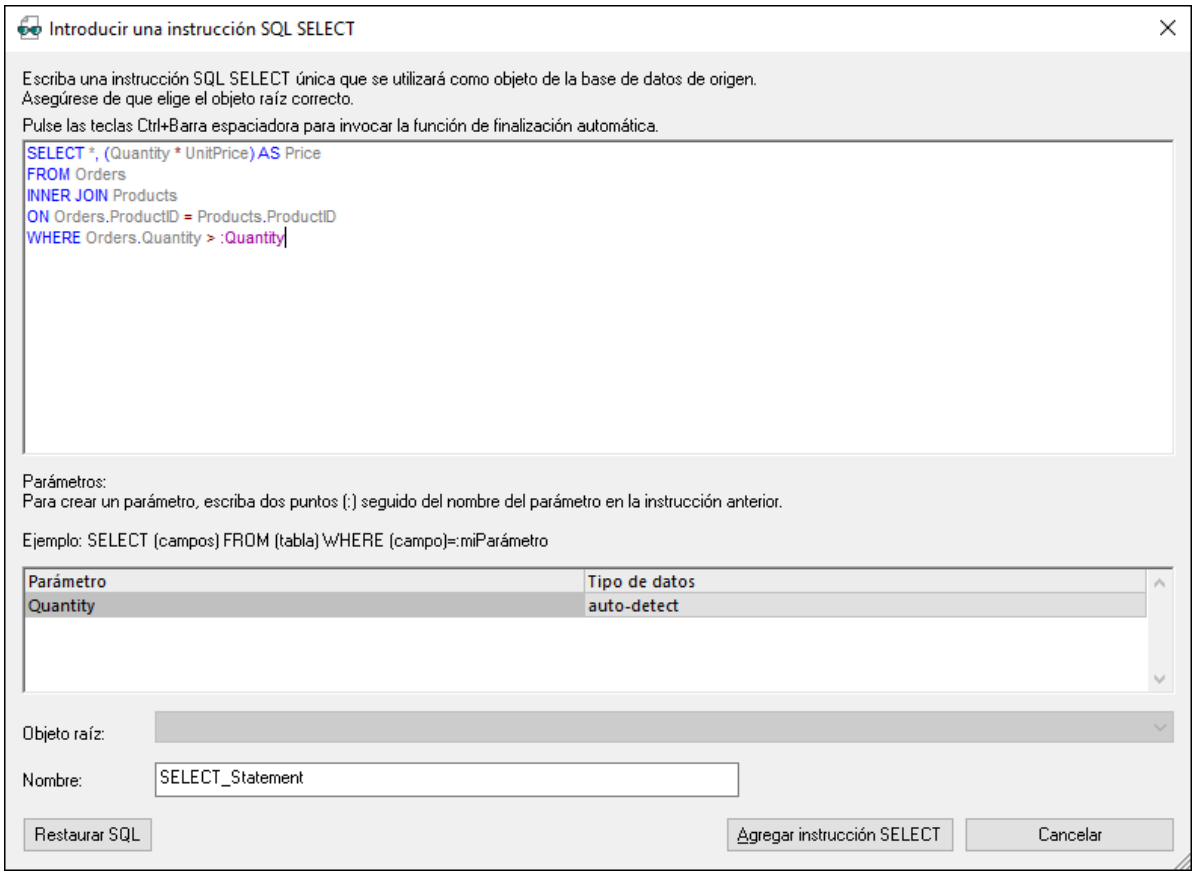


3. En el cuadro de diálogo "Insertar objetos de la base de datos" haga clic en el botón **Agregar/Editar instrucción SELECT** e introduzca esta consulta:

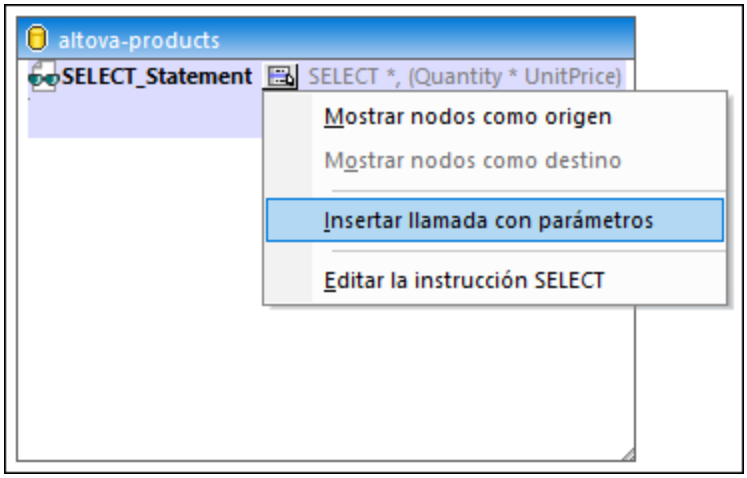
```
SELECT *, (Quantity * UnitPrice) AS Price
FROM Orders
INNER JOIN Products
ON Orders.ProductID = Products.ProductID
WHERE Orders.Quantity > :Quantity
```

Esta consulta usa una combinación interna entre las tablas Orders y Products y recupera todos los campos (`*`) y un valor calculado (`AS Price`). La consulta también especifica el parámetro **:Quantity** en la cláusula WHERE.

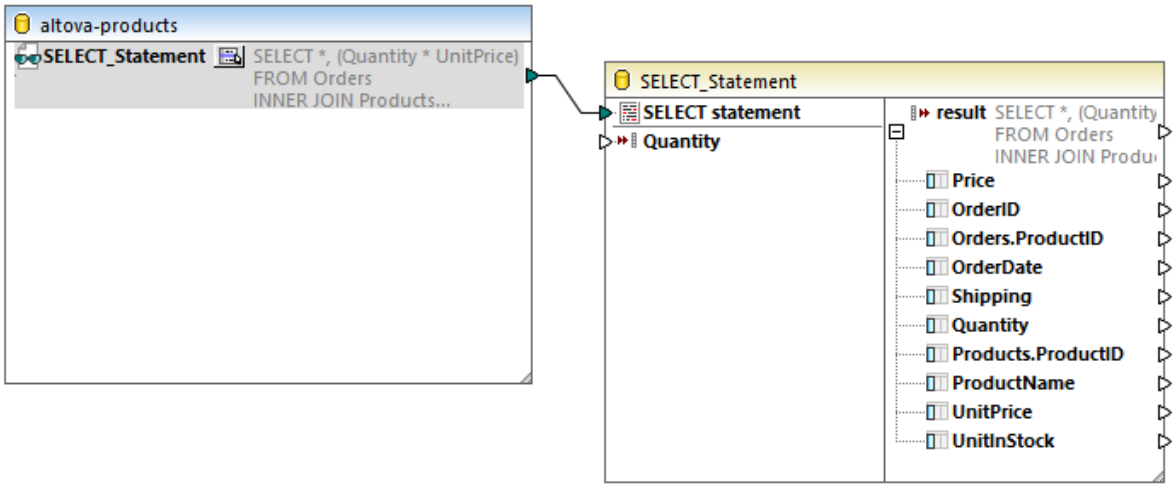
 © 2016-2022 Altova GmbH



4. Ahora haga clic en el botón **Agregar instrucción SELECT**.
5. Haga clic en **Aceptar** para terminar. El componente **altova-products** se añadió al área de asignación.
6. Ahora haga clic en el botón de este componente y seleccione **Insertar llamada con parámetros** en el menú contextual.

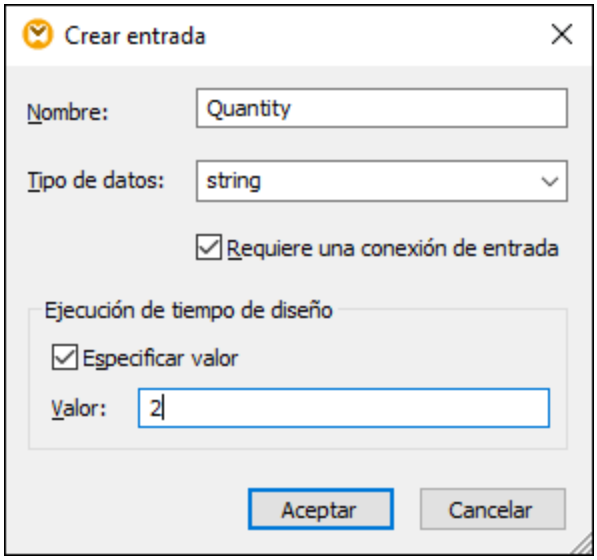


La asignación incluye ahora una estructura nueva (SELECT_Statement) que está dividida en dos paneles: el panel izquierdo aporta los conectores de entrada y el derecho los conectores de salida. Observe que el panel izquierda también incluye el parámetro **`Quantity`** que definimos en pasos anteriores.



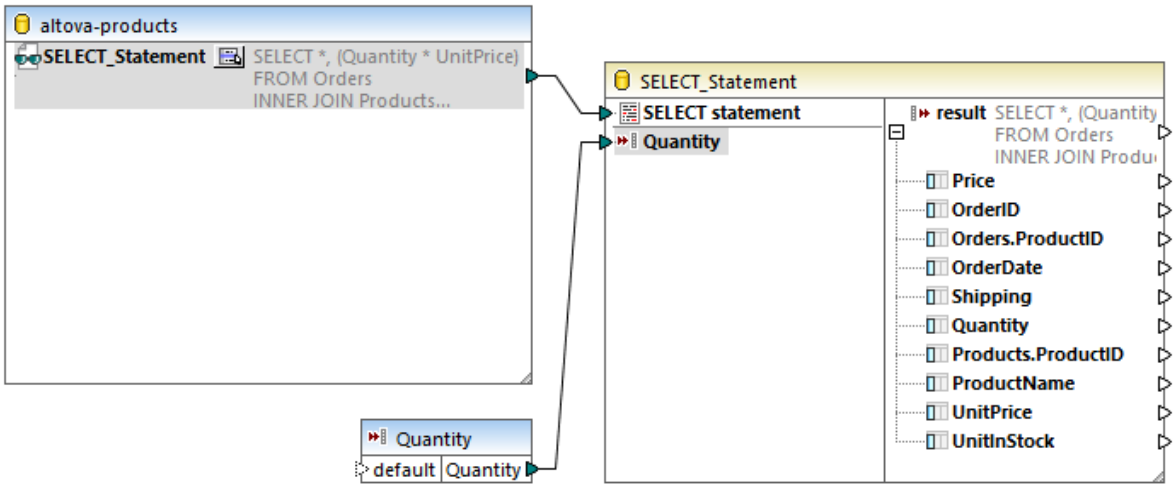
Paso n°2: agregar el parámetro de entrada

1. En el menú **Insertar** haga clic en el comando **Insertar componente de entrada**.
2. Introduzca el nombre "Quantity".
3. En el grupo de opciones *Ejecución de tiempo de diseño* introduzca el valor de parámetro que se debe usar para ejecutar la asignación durante la fase de diseño (p. ej. 2). Para más información consulte la sección Pasar parámetros a la asignación 477.



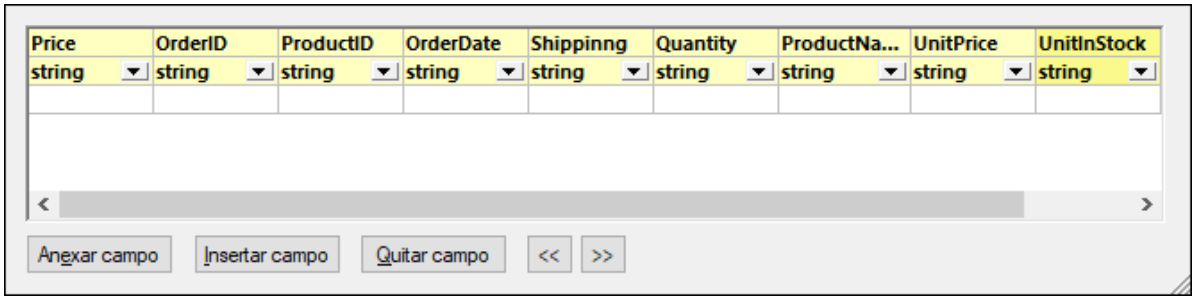
Ahora puede conectar el parámetro de entrada con la estructura de llamada a la base de datos (*imagen siguiente*).

 © 2016-2022 Altova GmbH

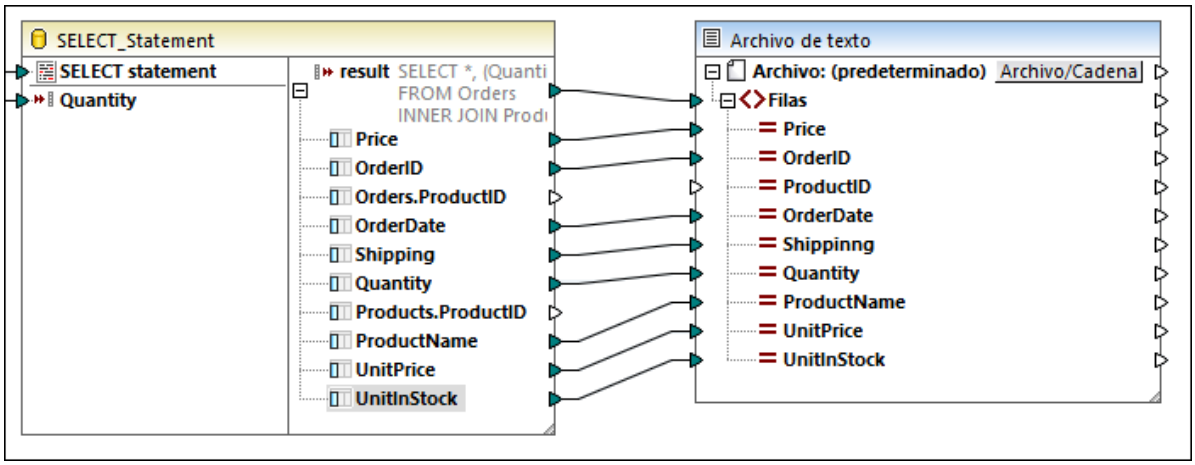


Paso n°3: agregar el componente CSV de destino

1. En el menú **Insertar** haga clic en el comando **Archivo de texto**.
2. Seleccione la opción *Utilizar procesamiento básico para archivos CSV estándar* y después haga clic en **Continuar**.
3. En el cuadro de diálogo "Configuración del componente" haga clic en el botón **Anexar campo** y añada nueve campos nuevos. Recomendamos que los campos CSV tengan el mismo nombre que los campos de la base de datos (*imagen siguiente*). Así podrá ahorrar tiempo a la hora de crear asignaciones. Consulte el apartado Opciones de configuración de componentes CSV 454 para obtener más información.



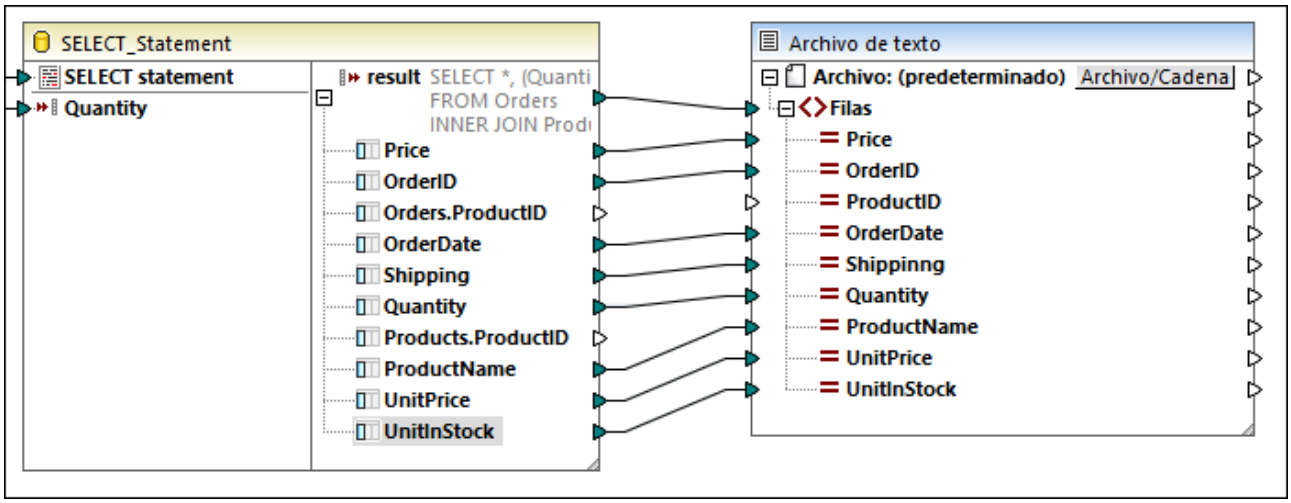
4. Cree una conexión entre el nodo **result** de la estructura SELECT y el nodo **Rows** del componente CSV.



Como la mayoría de los campos del componente CSV se llaman igual que su equivalente de la base de datos, cuando conecte **result** con **Rows**, probablemente MapForce cree las demás conexiones automáticamente. Si no lo hace, seleccione el comando Conexión y habilite la opción *Conectar automáticamente los secundarios equivalentes*. El único elemento que deberá conectar a mano será **ProductID** porque en la estructura SELECT no hay ningún campo con este nombre.

Paso n°4: convertir la fecha

En la ventana Bibliotecas busque la función `date-from-datetime` y arrástrela hasta el área de asignación. Después conecte su entrada y su salida tal y como se aprecia en esta imagen:



Paso n°5: establecer el nombre del archivo de salida

Para definir el nombre **OrdersReport.csv** para el archivo de salida basta con hacer doble clic en la barra del título del componente CSV. Se abre el cuadro de diálogo "Configuración del componente", donde puede introducir el valor en el cuadro de texto que corresponde al archivo de salida *(marcado en rojo en la imagen siguiente)*.

 © 2016-2022 Altova GmbH

4.2.7 Asignaciones entre datos XML y campos de BD

En MapForce puede crear asignaciones de datos con campos (columnas) de BD que almacenen contenido XML. Esto significa que los datos XML almacenados en el campo (columna) de BD se pueden extraer y escribir en cualquier otro tipo de estructura compatible con MapForce. Estas son las asignaciones de datos que se pueden crear en MapForce:

1. Asignaciones de datos con campos de un tipo XML especializado (p. ej. `Xml` en SQL Server y `XMLType` en Oracle). La lectura y escritura de datos XML en campos XML especializados es pertinente en bases de datos con compatibilidad nativa para XML (como IBM DB2, Oracle y SQL Server).
2. Asignaciones de datos con campos de texto que almacenen contenido XML (p. ej. `Text, Varchar`). Esto es pertinente en bases de datos con campos de texto de longitud suficiente como para almacenar un documento XML.

En ambos casos es necesario que exista un esquema XML válido para cada columna de BD a la que se desea asignar datos. Cuando una columna de BD almacena XML, MapForce ofrece la opción de asignar un esquema XML desde la BD directamente (si la BD lo admite) o seleccionar un esquema de un archivo externo. Puede asignar un esquema XML por cada columna de BD. Si el esquema tiene varios elementos raíz, podrá seleccionar un solo elemento raíz del esquema.

Cuando XML se almacena como campo de cadena en una BD, la codificación de caracteres del documento XML es la del campo de cadena subyacente. Si el campo de BD no almacena texto en Unicode, algunos caracteres no se podrán representar.

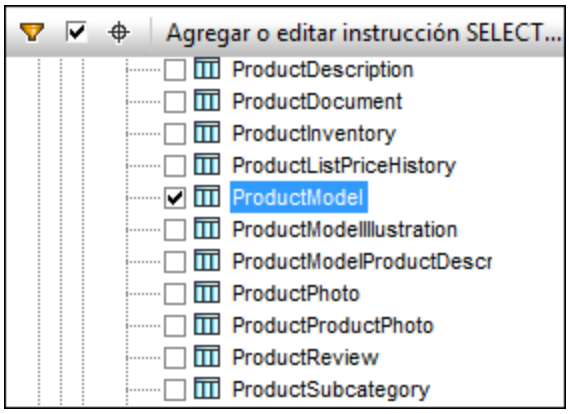
Algunas bases de datos admiten la codificación XML para campos XML (que no es necesariamente la misma que la del conjunto de caracteres de la BD). Si la base de datos lo admite, MapForce entiende que la declaración de codificación del documento XML es la que se declaró en el campo XML. Para más información sobre la compatibilidad de codificación XML consulte la documentación de cada base de datos.

4.2.7.1 Asignar un esquema XML a un campo de BD

En este tema explicamos cómo asignar un esquema a un campo que está definido de forma nativa como tipo XML en la base de datos. Para explicarlo usamos SQL Server 2017 y la base de datos Adventure Works 2014. Esta última se puede descargar de la página de muestras de Adventure Works en GitHub (https://github.com/Microsoft/sql-server-samples/releases/tag/adventureworks). Recuerde que la asignación de datos a campos XML funciona igual que la asignación a otros tipos de base de datos que admiten campos XML.

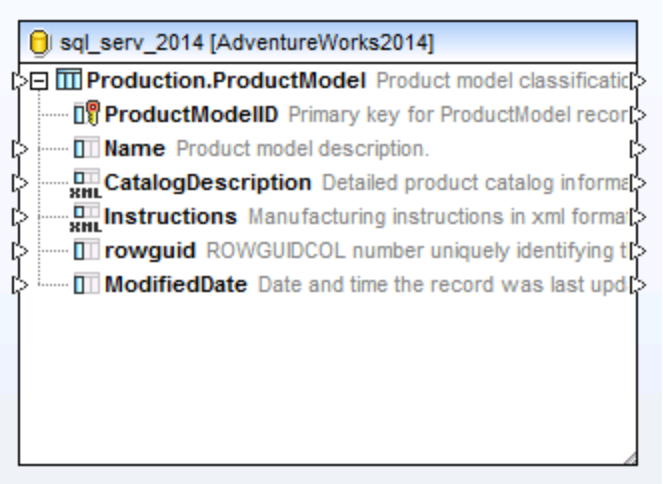
Para agregar la base de datos Adventure Works 2014 como componente de asignación:

1. En el menú Insertar haga clic en el comando Base de datos. Siga los pasos del Asistente para conectarse a la base de datos con ADO o con ODBC. Para más información consulte los temas Conectarse a Microsoft SQL Server (ADO) 168 y Conectarse a Microsoft SQL Server (ODBC) 180. **Nota:** si usa el controlador **SQL Server Native Client**, quizás necesite establecer un carácter de espaciado como valor de la propiedad **Seguridad integrada** (*véase Configurar las propiedades de vínculo de datos de SQL Server* 171).
2. En el cuadro de diálogo "Insertar objeto de base de datos" expanda el esquema **Production** y después marque la casilla de la tabla **ProductModel**.



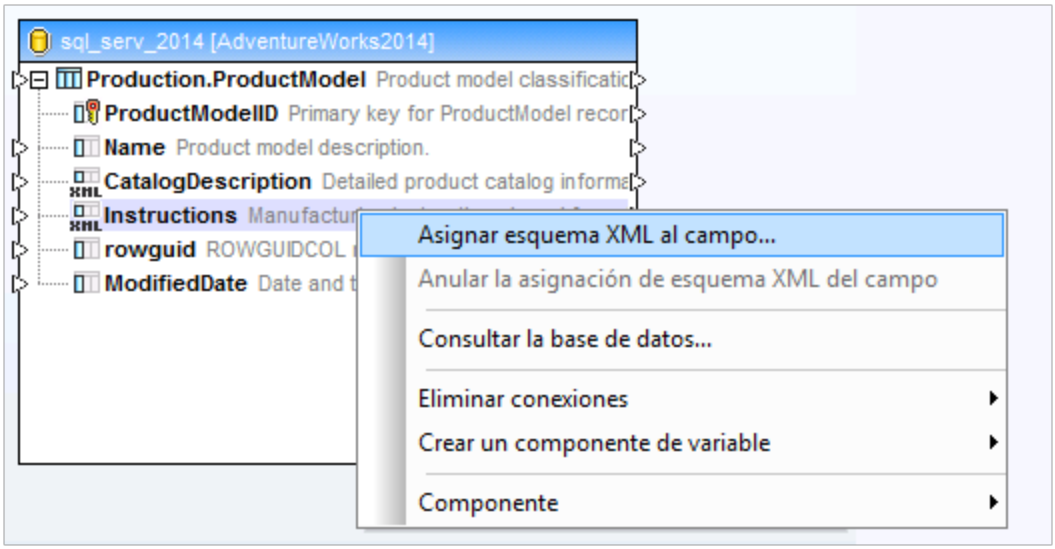
3. Para terminar haga clic en **Aceptar**.

La tabla de base de datos se añade al área de asignación. Observe que esta tabla tiene dos campos de tipo XML, **CatalogDescription** e **Instructions**:

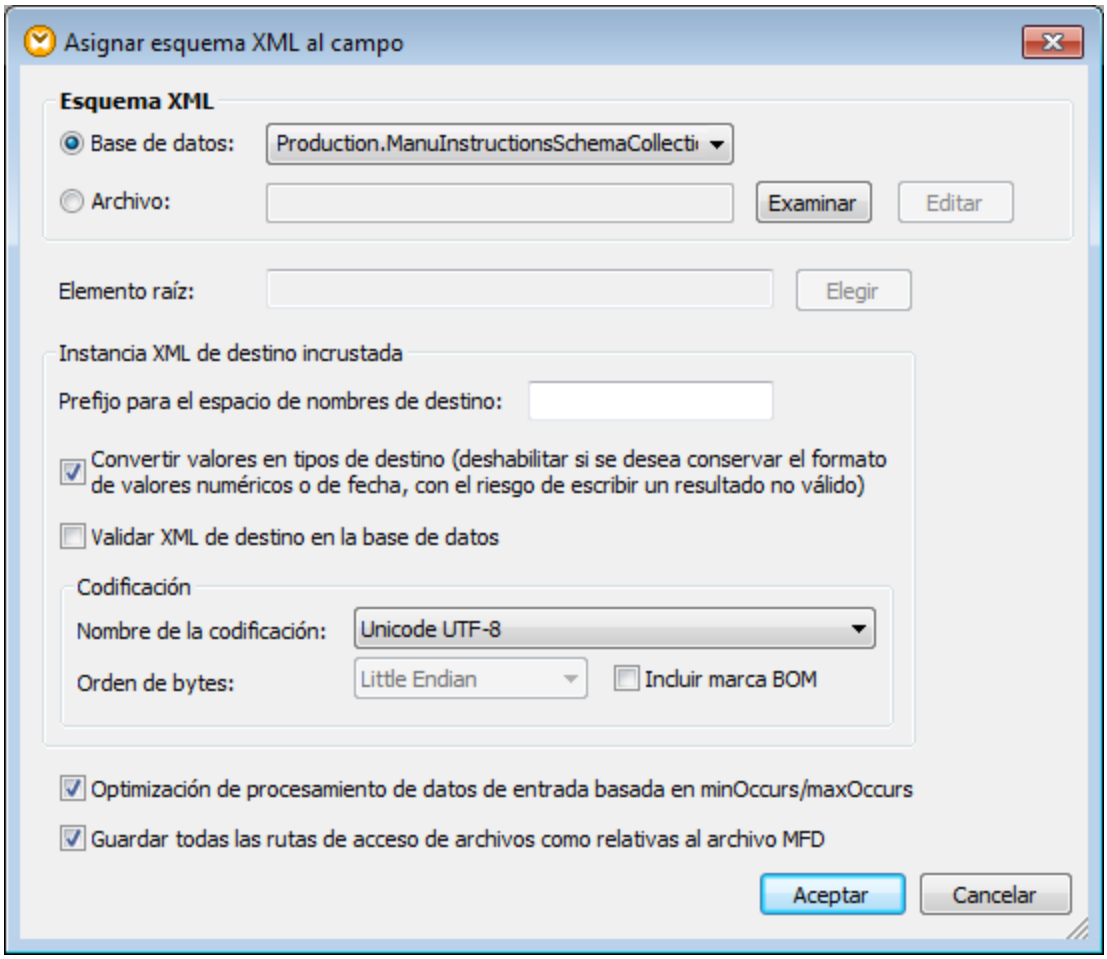


Para que la estructura de los campos XML aparezcan en la asignación es necesario el esquema XML del contenido del campo. Haga clic con el botón derecho en el campo *Instructions* y seleccione **Asignar esquema XML al campo** en el menú contextual.

 © 2016-2022 Altova GmbH

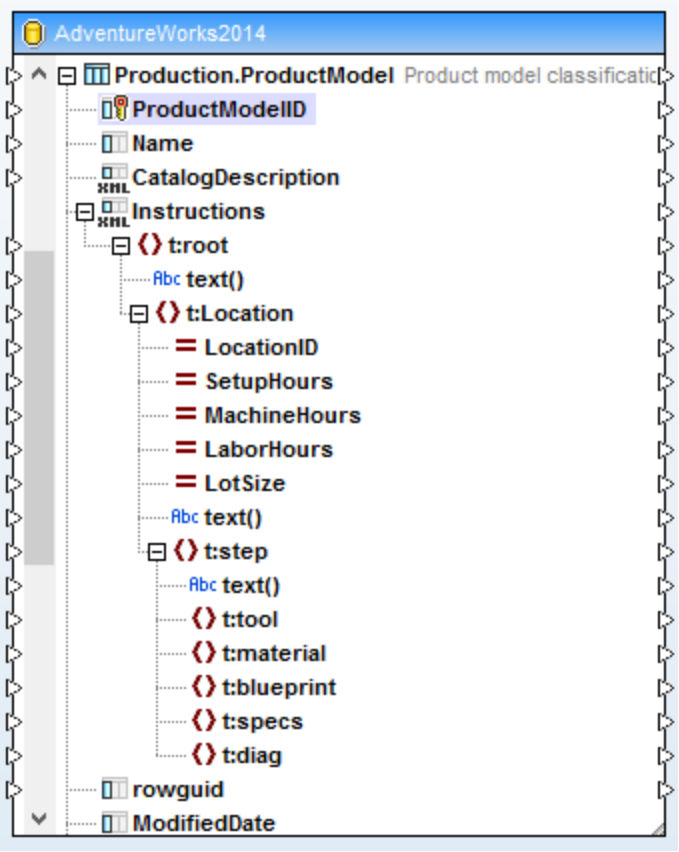


En este ejemplo concreto asignaremos un esquema al campo **Instructions** desde la BD directamente. Para ello seleccionamos el elemento **Production.ManuInstructionsSchemaCollection** en el cuadro combinado de la opción *Base de datos:* y hacemos clic en **Aceptar**.



La estructura del campo XML ya aparece en el componente y se pueden crear conexiones (y asignaciones de datos) entre este campo y otros elementos.

 © 2016-2022 Altova GmbH



4.2.7.2 Ejemplo: escribir datos XML a un campo SQLite

Este ejemplo explica paso a paso cómo crear una asignación de MapForce que lee datos de varios archivos XML y los escribe en una base de datos SQLite. El objetivo de la asignación es crear un registro de BD nuevo en la base de datos SQLite por cada archivo XML de origen. Cada registro almacenará el documento XML como campo TEXT.

Todos los archivos utilizados en este ejemplo están en la ruta de acceso `<Documentos>\Altova\MapForce2023\MapForceExamples\Tutorial\`:

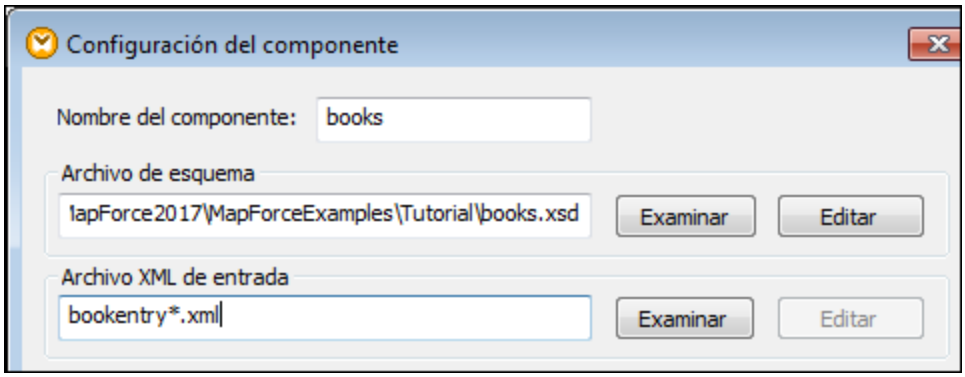
El archivo de diseño de asignación	• **XmlToSqliteField.mfd**
Los archivos XML de origen	• **bookentry1.xml** • **bookentry2.xml** • **bookentry3.xml**
El esquema XML utilizado para la validación	• **books.xsd**
La base de datos SQLite de destino	• **Library.sqlite**

Para conseguir el objetivo de la asignación tendremos que seguir estas instrucciones:

1. Agregar el componente XML y configurarlo para que lea datos de varios archivos.
2. Agregar el componente de BD SQLite y asignar un esquema XML al campo TEXT de destino.
3. Crear las conexiones de asignación y configurar la acción INSERT de la base de datos.

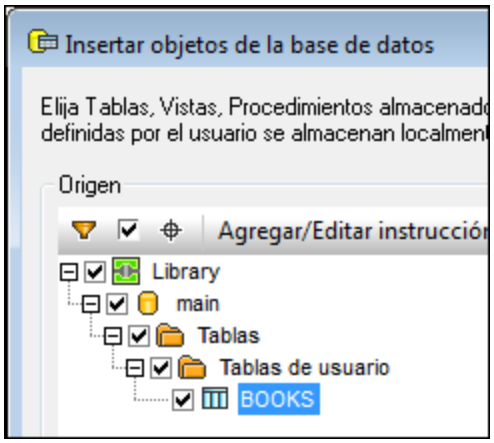
Paso nº1: agregar el componente XML

1. En el menú **Insertar** haga clic en el comando **Archivo o esquema XML...** y navegue hasta el esquema `books.xsd` ubicado en el directorio `<Documentos>\Altova\MapForce2023\MapForceExamples\Tutorial\`. Cuando MapForce solicite un archivo XML de muestra, haga clic en **Omitir**. Cuando MapForce solicite un elemento raíz, seleccione **Books**.
2. Haga doble clic en el título del componente e introduzca **bookentry*.xml** en el cuadro de texto *Archivo XML de entrada*. Esto indica que MapForce debe leer todos los archivos XML del directorio de origen cuyo nombre empiece por "bookentry-". Para más información consulte la sección Procesar varios archivos de entrada o salida simultáneamente 892.



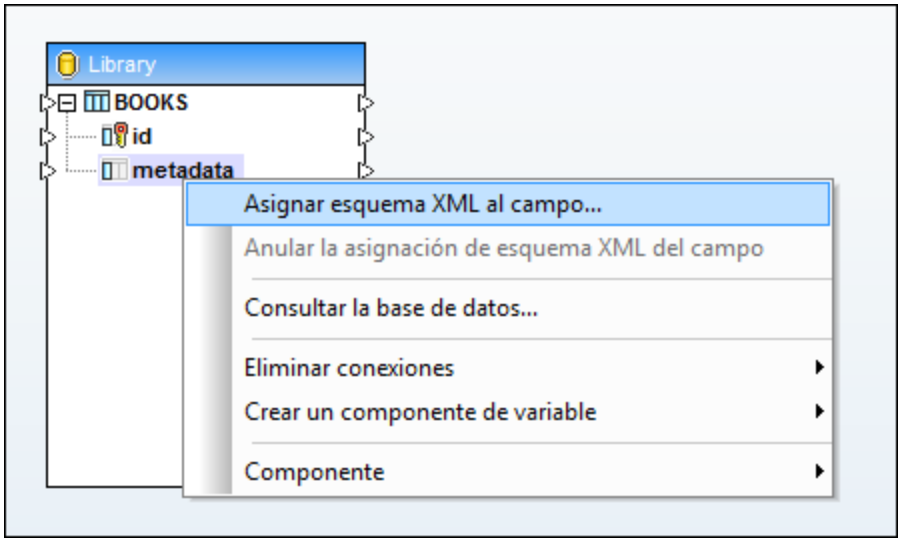
Paso nº2: agregar el componente SQLite

En el menú **Insertar** haga clic en **Base de datos** y siga los pasos del Asistente para conectarse al archivo de base de datos **Library.sqlite** del directorio `<Documentos>\Altova\MapForce2023\MapForceExamples\Tutorial\` directory (*véase* *Conectarse a una base de datos SQLite* 190). Llegada la hora de seleccionar objetos de la BD, seleccione la tabla `BOOKS`.

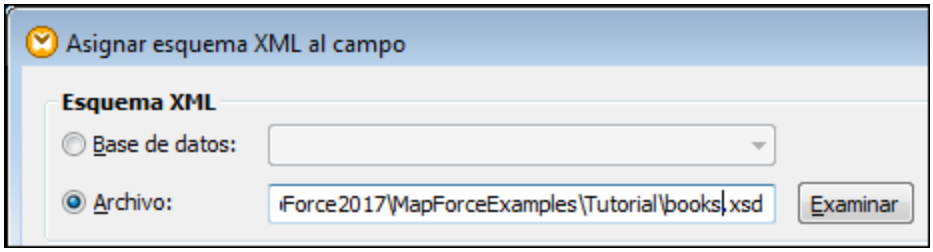


 © 2016-2022 Altova GmbH

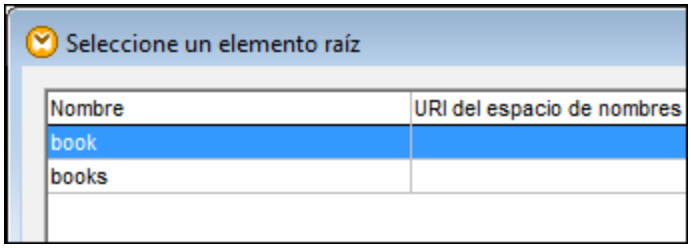
El campo de BD donde se escribirá el contenido XML se llama `metadata`. Para asignar un esquema XML a este campo haga clic con el botón derecho y seleccione **Asignar esquema XML al campo** en el menú contextual.



En este tutorial el esquema asignado al campo `metadata` es el mismo que se usa para validar los archivos XML de origen. Haga clic en **Examinar** y seleccione el esquema `books.xsd` del directorio `<Documentos>\Altova\MapForce2023\MapForceExamples\Tutorial\`:

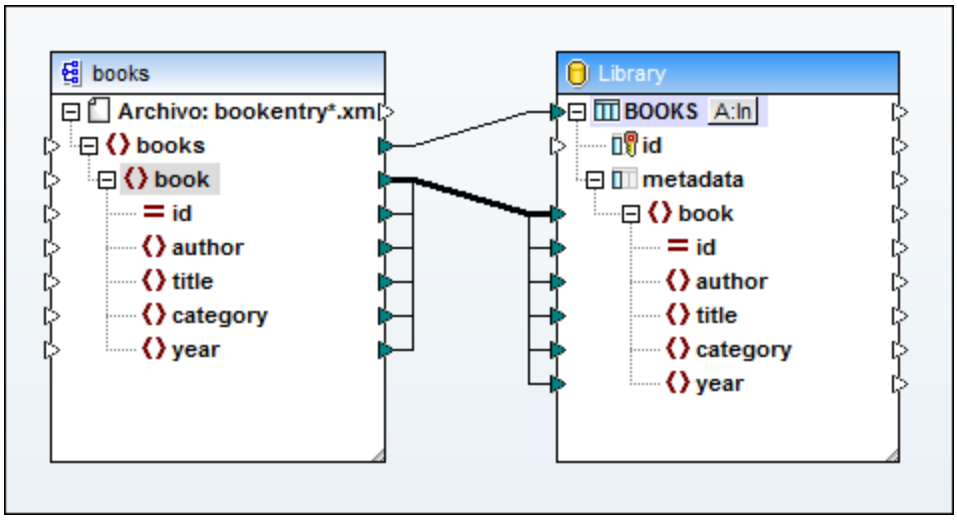


El esquema **books.xsd** tiene dos elementos con declaración global: `book` y `books`. En este ejemplo estableceremos `book` como elemento raíz de los datos XML escritos en el campo de la BD. Haga clic en **Elegir** y seleccione `book` como elemento raíz:



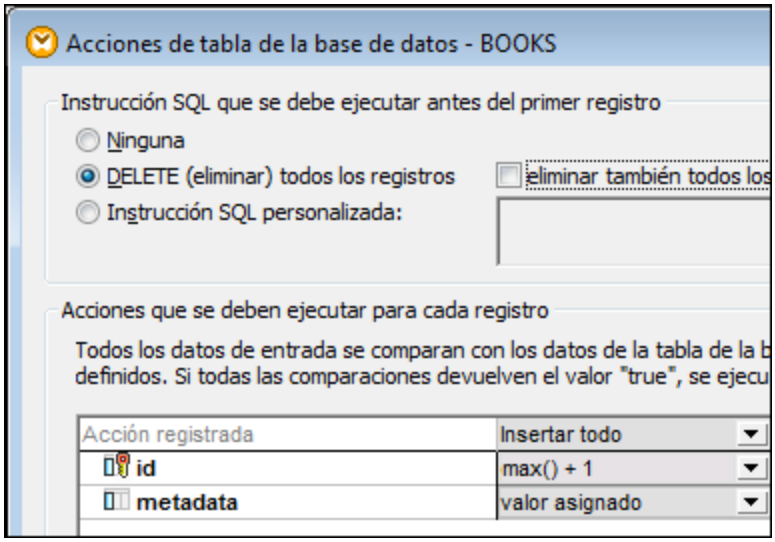
Paso nº3: crear las conexiones de asignación y configurar la acción INSERT

Cree estas conexiones de asignación:



Como se aprecia en la imagen anterior, la conexión entre `book` y `book` es una conexión de copia total porque tanto el origen como el destino usan el mismo esquema y los elementos secundarios tienen nombres idénticos. Para más información consulte el apartado Conexiones de copia total 95.

La conexión de nivel superior (entre `books` y `BOOKS`) recorre cada elemento `books` del origen y escribe un registro nuevo en la tabla BOOKS. Haga clic en el botón `A:In` del componente de BD y defina estas opciones de configuración para la actualización de la base de datos:



La opción **DELETE (eliminar) todos los registros** indica que MapForce debe eliminar el contenido de la tabla `BOOKS` antes de insertar otros registros.

Las acciones **Insertar todo** especifican que tendrá lugar una consulta `INSERT` de base de datos. El campo `id` se genera a partir de la BD propiamente dicha, mientras que el campo `metadata` se rellenará con el valor que viene dado por la asignación.

Recuerde que antes de ejecutar la asignación deberá guardarla.

 © 2016-2022 Altova GmbH

Para ejecutar la asignación y ver el resultado generado haga clic en el panel *Resultados*. No olvide que esta acción no actualiza inmediatamente la base de datos. Cuando esté listo para ejecutar el script de base de datos generado, seleccione el comando de menú **Resultados | Ejecutar script SQL** (o haga clic en el botón de la barra de herramientas).

4.2.7.3 Ejemplo: extraer datos de columnas tipo XML de IBM DB2

Este ejemplo explica cómo extraer datos de columnas de bases de datos IBM DB2 de tipo XML y escribirlos en un archivo CSV de destino. También explica cómo usar instrucciones XQuery incrustadas en código SQL para recuperar datos XML de forma condicional. Para seguir este ejemplo se necesita tener acceso a una base de datos IBM DB2 con permiso para crear y rellenar tablas.

Primero debemos preparar la base de datos par que contenga datos XML. Esto se puede hacer desde la herramienta de administración propia de la base de datos o en MapForce directamente. para hacerlo en MapForce debe seguir estas instrucciones:

1. Cree una asignación nueva y haga clic en el panel *Consulta de la BD*.
2. Haga clic en el botón **Conexión rápida** y siga los pasos del asistente para crear una conexión de base de datos nueva (*véase Ejemplos de conexión a bases de datos*).
3. Pegue este texto en el editor SQL. Esta consulta SQL crea una tabla de base de datos llamada `ARTICLES` y la rellena con datos.

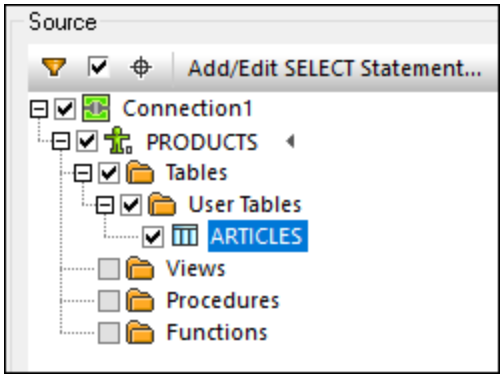
```
-- Create the table
CREATE TABLE
    ARTICLES (
        id INTEGER NOT NULL,
        article XML ) ;
-- Populate the table
INSERT INTO ARTICLES VALUES
   (1, '<Article>
      <Number>1</Number>
      <Name>T-Shirt</Name>
      <SinglePrice>25</SinglePrice>
   </Article>'),
(2, '<Article>
      <Number>2</Number>
      <Name>Socks</Name>
      <SinglePrice>230</SinglePrice>
   </Article>'),
(3, '<Article>
      <Number>3</Number>
      <Name>Pants</Name>
      <SinglePrice>34</SinglePrice>
   </Article>'),
 (4, '<Article>
      <Number>4</Number>
      <Name>Jacket</Name>
      <SinglePrice>5750</SinglePrice>
   </Article>');
```

4. Haga clic en el botón **Ejecutar** . El resultado de la ejecución de la consulta aparece en la ventana *Resultados*. Si la consulta se ejecuta correctamente, la tabla recién creada contará con cuatro filas.

Ahora debemos crear una asignación de datos que recupere datos XML de forma condicional de la tabla `ARTICLES` que acabamos de crear. El objetivo es recuperar productos que tengan un importe superior a 100 en la columna `ARTICLES`.

Paso nº1: agregar la base de datos

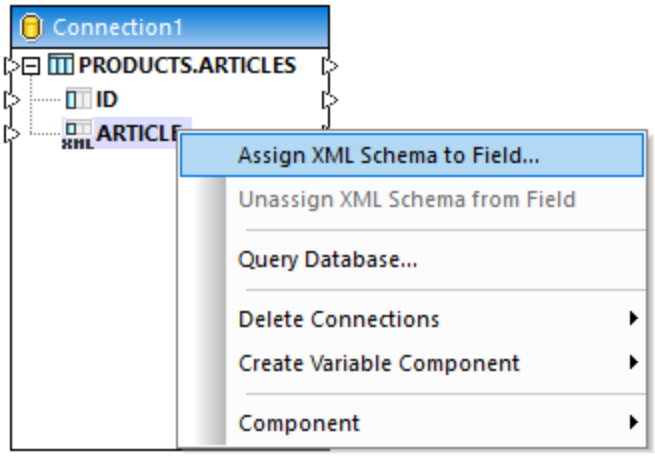
1. Vuelva al panel *Asignación*.
2. En el menú **Insertar** seleccione el comando **Base de datos** y siga los pasos del asistente para conectarse a la base de datos.
3. Cuando llegue el momento de seleccionar objetos de la base de datos, seleccione la tabla `ARTICLES` que se creó en el paso anterior.



Paso nº2: asignar el esquema al campo de tipo XML

1. Haga clic con el botón derecho en el elemento `ARTICLE` del componente y seleccione **Asignar esquema XML al campo** en el menú contextual.

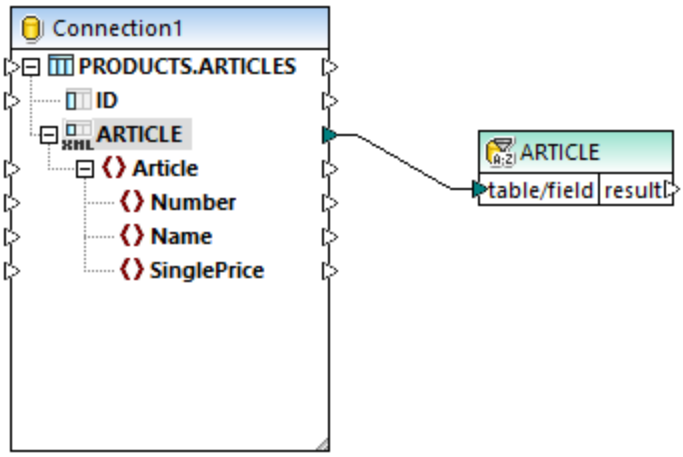
© 2016-2022 Altova GmbH



2. Seleccione *Archivo* y navegue hasta este esquema: **<Documentos>\Altova\MapForce2023\MapForceExamples\Tutorial\DB2xsd.xsd**.

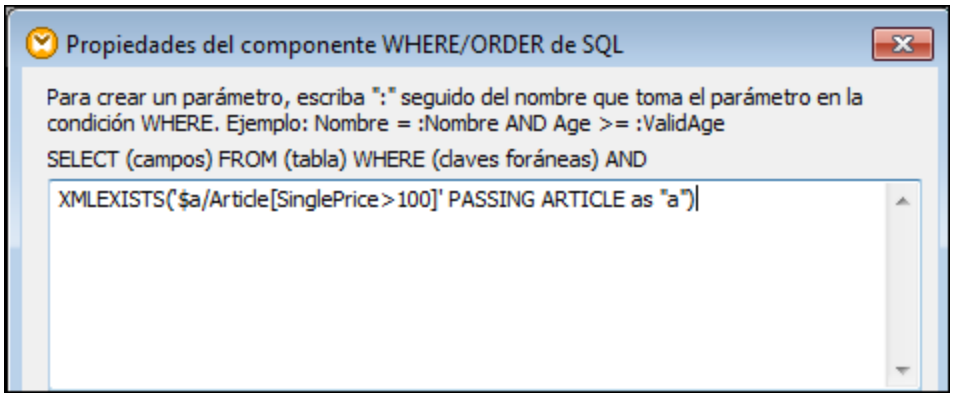
Paso nº3: agregar el componente WHERE/ORDER de SQL

1. En el menú **Insertar** seleccione el comando **WHERE/ORDER de SQL**.
2. Conecte la columna de tipo XML `ARTICLE` a la entrada del componente WHERE/ORDER de SQL.



3. En el cuadro de diálogo "Propiedades del componente WHERE/ORDER de SQL" introduzca este texto:

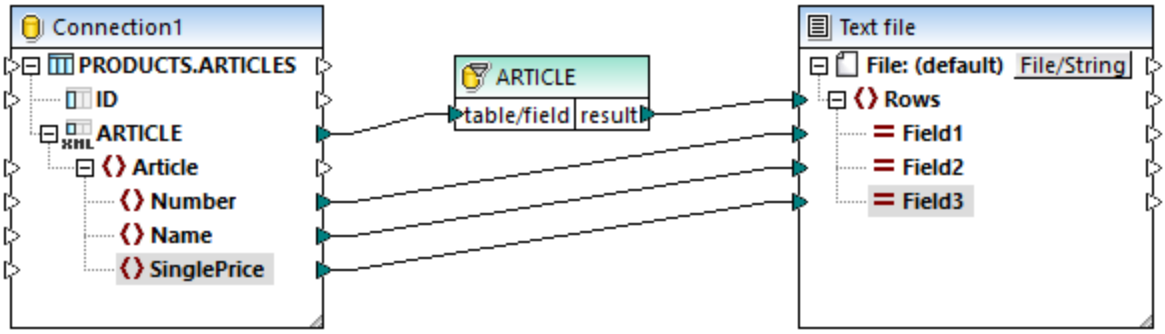
```
XMLEXISTS('$a/Article[SinglePrice>100]' PASSING ARTICLE as "a")
```



Este texto representa la parte WHERE de la consulta SQL. Cuando se ejecute la asignación, la parte WHERE se combinará con la parte SELECT que aparece en el cuadro de diálogo. Esta instrucción usa la función `XMLEXISTS` y sintaxis propia de las bases de datos IBM DB2.

Paso nº4: agregar el archivo CSV de destino

1. En el menú **Insertar** seleccione el comando **Archivo de texto**.
2. Después seleccione la opción *Utilizar procesamiento básico para archivos CSV....* y haga clic en **Continuar**.
3. Haga clic tres veces en **Anexar campo** para agregar tres campos que almacenarán el número de producto, el nombre y el precio respectivamente. Deje las demás opciones de configuración como están.
4. Dibuje las conexiones que aparecen en la imagen siguiente.



Ahora puede generar una vista previa de los resultados de la asignación de datos (abriendo el panel *Resultados*). Como esperábamos, en el resultado solo aparecen los productos con un precio superior a 100.

```
1    2,Socks,230
2    4,Jacket,5750
```

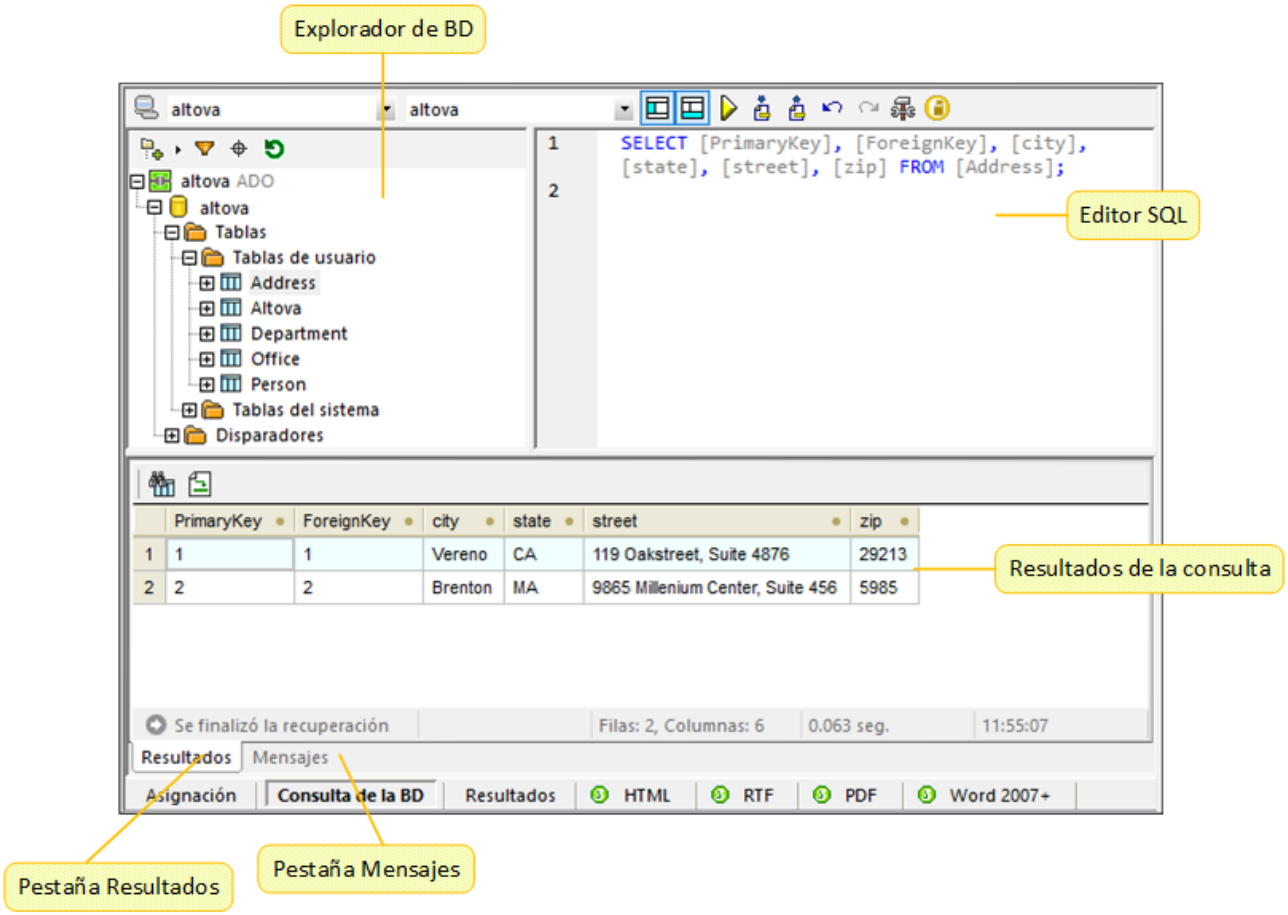
4.2.8 Examinar y consultar bases de datos

MapForce tiene un panel especial de consulta de bases de datos (llamado ***Consulta de la BD***) que sirve para consultar bases de datos por separado del proceso de asignación. Estas consultas directas no se guardan

 © 2016-2022 Altova GmbH

con la asignación en el archivo *.mfd, sino que son solo una manera de examinar y modificar el contenido de bases de datos desde MapForce directamente.

MapForce ofrece un panel de consulta de la BD por cada diseño de asignación que esté activo. Puede crear varias conexiones activas con varias bases de datos distintas dentro de cada panel *Consulta de la BD*. Recuerde que las conexiones creadas desde el panel *Consulta de la BD* no forman parte de la asignación y, por tanto, no se conservan cuando se cierra la aplicación, a no ser que las defina como recursos globales (*véase* *Recursos globales* 969).



El panel *Consulta de la BD* se divide en varias vistas:

- **Explorador de la BD:** muestra información sobre la conexión y las tablas de la BD.
- **Editor SQL:** aquí puede escribir sus consultas SQL.
- **Pestaña *Resultados*:** muestra los resultados de las consultas en forma de tabla.
- **Pestaña *Mensajes*:** muestra advertencias y mensajes de error.

La parte superior del panel *Consulta de la BD* contiene botones de control para definir las bases de datos de trabajo, la conexión y los esquemas de la base de datos.

4.2.8.1 Seleccionar o conectarse a una base de datos

Por cada base de datos que necesite consultar deberá crear una conexión de base de datos. Si la asignación ya incluye un componente de base de datos, puede seleccionar la conexión de base de datos actual en la parte superior del panel *Consulta de la BD* (por defecto el panel tiene el estado *Sin conexión*) y empezar a explorar los objetos de la BD y a ejecutar consultas.

Si la asignación no incluye ningún componente de base de datos todavía o si desea conectarse a una base de datos nueva, haga clic en el botón **Conexión rápida** y siga los pasos del asistente para crear una conexión de base de datos nueva (aquí puede ver algunos ejemplos). Si procede, también puede seleccionar una conexión actual de los recursos globales (*véase* *Recursos globales* 969).

Tras conectarse a la base de datos podrá crear consultas de base de datos de la siguiente manera:

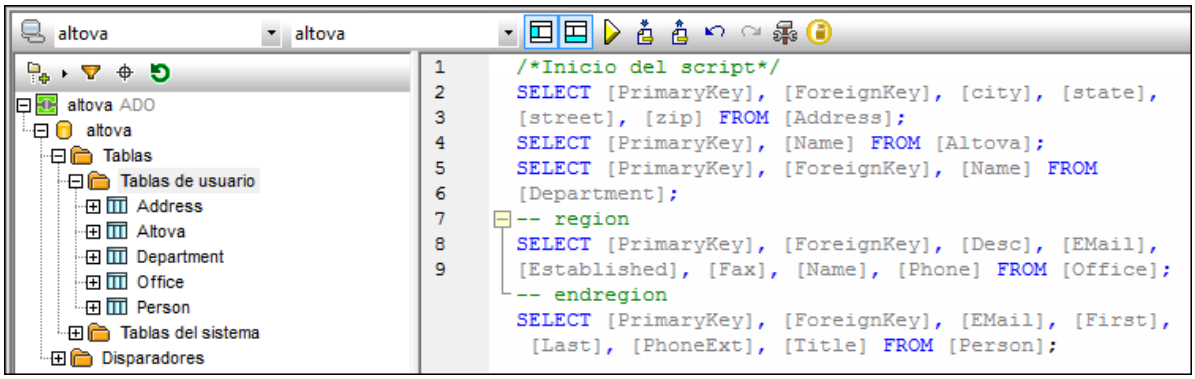
- Importando la consulta SQL al panel Editor SQL desde un archivo SQL ya existente.
- Escribiendo la consulta en el panel Editor SQL.
- Haciendo clic con el botón derecho en un objeto del panel Explorador y generando una consulta (generalmente SELECT).

Cuando termine de crear la consulta en el panel Editor SQL, haga clic en el botón **Ejecutar**. Los datos de la base de datos se recuperan y aparecen en el panel *Resultados* en forma de tabla. Recuerde que en la barra de estado aparecerá el mensaje Se finalizó la recuperación y cualquier otra información relevante sobre los resultados de la consulta.

Después de que aparezca el mensaje Se finalizó la recuperación podrá buscar resultados, ordenarlos y copiarlos en el portapapeles (*véase* *Copiar, ordenar y buscar resultados de la consulta* 392).

4.2.8.2 Crear y editar instrucciones SQL

El Editor SQL sirve para escribir y ejecutar instrucciones SQL, tanto generadas automáticamente, como cargadas desde scripts SQL o escritas a mano. El editor SQL incluye características muy prácticas como la finalización automática (*véase* *Finalización automática* 277), las regiones y los comentarios en bloque o en línea.



Estos son los botones de la barra de herramientas del Editor SQL:

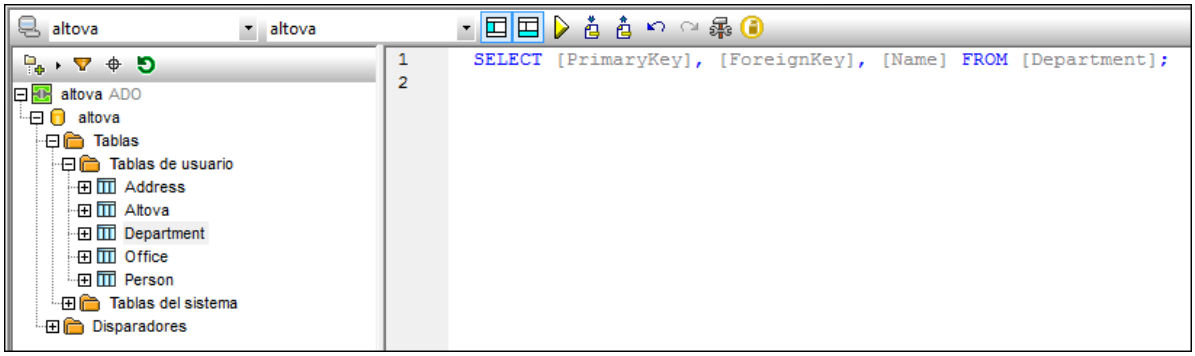
	Mostrar u ocultar Explorador: muestra u oculta el panel Explorador.
	Mostrar u ocultar Resultados: muestra u oculta la pestaña *Resultados*.
	Ejecutar (F5): ejecuta las instrucciones SQL que estén seleccionadas en el editor. Si existen varias instrucciones y no se seleccionó ninguna, entonces se ejecutan todas.
	Deshacer: permite deshacer un número ilimitado de operaciones de edición en el Editor SQL.
	Rehacer: permite rehacer comandos Deshacer previos. Con estos comandos puede avanzar y retroceder en el historial de operaciones Deshacer.
	Importar archivo SQL: abre un archivo SQL en el Editor SQL. A continuación se podrá ejecutar.
	Exportar archivo SQL: guarda las consultas SQL para poder utilizarlas más tarde.
	Abrir script SQL en DatabaseSpy: abre el script en el editor SQL de Altova DatabaseSpy.
	Opciones: abre el cuadro de diálogo "Opciones", donde se pueden definir opciones de configuración generales sobre la consulta de bases de datos y opciones de configuración del editor SQL.

4.2.8.2.1 Generar instrucciones SQL

Puede generar instrucciones SQL automáticamente desde el panel Explorador, cargarlas desde scripts o introducirlas a mano.

Para generar instrucciones SELECT SQL desde el Explorador:

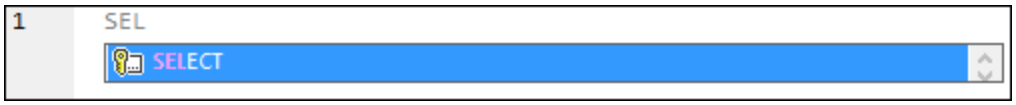
- Haga clic en un objeto de la BD (como una tabla o una vista) o en una carpeta del Explorador y arrástrela hasta el Editor SQL.



- Haga clic con el botón derecho en un objeto de la base de datos en el Explorador y seleccione **Mostrar en Editor SQL | SELECT** en el menú contextual.

Para crear instrucciones SQL a mano:

1. Empiece a teclear la instrucción SQL en el Editor SQL. Si la finalización automática está habilitada, aparecerá una lista desplegable con sugerencias mientras teclea en el editor.



2. Con las teclas de dirección **Arriba** y **Abajo** del teclado puede seleccionar la sugerencia y pulsar Entrar para insertar la opción resaltada (*véase* *Sugerencias de finalización automática SQL* 277).

4.2.8.2.2 Ejecutar instrucciones SQL

Las instrucciones SQL que aparecen en el Editor SQL se pueden ejecutar en la base de datos y con efecto inmediato. El resultado de la consulta SQL y el número de filas afectadas aparecen en la pestaña *Mensajes* del panel *Consulta de la BD*.

Cuando hay varias instrucciones SQL en el Editor SQL, solamente se ejecutarán las instrucciones que estén seleccionadas. Para seleccionar una instrucción SQL:

- Arrastre el puntero por encima de la instrucción sin dejar de pulsar el botón principal del ratón.
- Haga clic en un número de línea del Editor SQL.
- Haga triple clic en una instrucción.

Para ejecutar una instrucción SQL:

1. Introduzca o seleccione la instrucción SQL en el Editor SQL (*véase* *Generar instrucciones SQL* 383).
2. Haga clic en el botón **Ejecutar** .

4.2.8.2.3 Importar y exportar scripts SQL

Puede guardar el código SQL de la ventana Editor SQL en un archivo y usarlo más tarde.

Para exportar el contenido del Editor SQL a un archivo:

- Haga clic en **Exportar archivo SQL** e introduzca un nombre para el script SQL.

 © 2016-2022 Altova GmbH

Para importar un archivo SQL:

- Haga clic en **Importar archivo SQL** y seleccione el archivo SQL que desea abrir.

4.2.8.2.4 Agregar y quitar comentarios SQL

En el editor SQL de MapForce puede convertir instrucciones, parte de instrucciones o grupos de instrucciones en comentarios. Las instrucciones convertidas en comentarios se ignoran cuando se ejecuta el script SQL.

Para convertir una sección del texto en comentario:

1. Seleccione una instrucción o parte de la instrucción.
2. Haga clic con el botón derecho en el texto seleccionado y elija la opción **Convertir el bloque en comentario/quitar comentario**.

```
1
2   /*
3   SELECT [PrimaryKey], [ForeignKey], [city], [state],
    [street], [zip] FROM [Address];
4   SELECT [PrimaryKey], [Name] FROM [Altova];
5   SELECT [PrimaryKey], [ForeignKey], [Name] FROM
    [Department];*/
6
7   SELECT [PrimaryKey], [ForeignKey], [Desc], [EMail],
    [Established], [Fax], [Name], [Phone] FROM [Office];
8   SELECT [PrimaryKey], [ForeignKey], [EMail], [First],
    [Last], [PhoneExt], [Title] FROM [Person];
9
```

Para convertir una línea en comentario:

Haga clic con el botón derecho en la posición que desea convertir en comentario y seleccione la opción **Convertir la línea en comentario/quitar comentario**. A partir de la posición del cursor la instrucción se convierte en comentario.

Para quitar un comentario (de bloque o de línea):

1. Seleccione la parte de la instrucción que es un comentario.
   Si quiere quitar un comentario de línea, basta con borrar las marcas de comentario - - situadas al principio del comentario.

2. Haga clic con el botón derecho y seleccione la opción **Convertir el bloque (o la línea) en comentario/quitar comentario**.

4.2.8.2.5 Uso de marcadores

Los marcadores sirven para marcar puntos de interés en los scripts de gran tamaño.

Para agregar un marcador:

Haga clic con el botón derecho en la línea en la que desea insertar el marcador y seleccione **Insertar o quitar marcador** en el menú contextual.

```
1
2   SELECT [PrimaryKey], [ForeignKey], [city], [state],
    [street], [zip] FROM [Address];
3   SELECT [PrimaryKey], [Name] FROM [Altova];
4   SELECT [PrimaryKey], [ForeignKey], [Name] FROM
    [Department];
5   SELECT [PrimaryKey], [ForeignKey], [Desc], [EMail],
    [Established], [Fax], [Name], [Phone] FROM [Office];
6   SELECT [PrimaryKey], [ForeignKey], [EMail], [First],
    [Last], [PhoneExt], [Title] FROM [Person];
7
```

El icono aparece en el margen de la línea que incluye el marcador.

Para quitar un marcador:

Haga clic con el botón derecho en la línea que incluye el marcador que desea eliminar y seleccione **Insertar o quitar marcador** en el menú contextual.

Para navegar de un marcador a otro:

- Para pasar al siguiente marcador, haga clic con el botón derecho y seleccione **Ir al siguiente marcador**.
- Para ir al marcador anterior, haga clic con el botón derecho y seleccione **Ir al marcador anterior**.

© 2016-2022 Altova GmbH

Para quitar todos los marcadores:

Haga clic con el botón derecho en el **Editor SQL** y seleccione **Quitar todos los marcadores**.

4.2.8.2.6 Insertar regiones

Las regiones son partes del texto que se marcan y se declaran como una unidad. Estas unidades sirven para estructurar los scripts SQL. Las regiones se pueden expandir y contraer para mostrar u ocultar partes del script SQL. También se pueden anidar regiones dentro de otras.

Cuando se crea una región, se inserta un icono de expansión/contracción y el comentario `-- region` antes del texto seleccionado.

Nota: puede cambiar el nombre de una región anexando un texto descriptivo al comentario `--region`. Pero no elimine la palabra `region`. Un ejemplo sería `-- region DB2query`.

Para crear una región:

1. En el **Editor SQL** seleccione las instrucciones que desea insertar en una región.
2. Haga clic con el botón derecho y seleccione la opción **Agregar región** del menú contextual.
   Las instrucciones seleccionadas se convierten en una región que se puede expandir o contraer.

```
1
2   SELECT [PrimaryKey], [ForeignKey], [city], [state],
    [street], [zip] FROM [Address];
3   -- region
4   SELECT [PrimaryKey], [Name] FROM [Altova];
5   SELECT [PrimaryKey], [ForeignKey], [Name] FROM
    [Department];
6   -- endregion
7   SELECT [PrimaryKey], [ForeignKey], [Desc], [EMail],
    [Established], [Fax], [Name], [Phone] FROM [Office];
8   SELECT [PrimaryKey], [ForeignKey], [EMail], [First],
    [Last], [PhoneExt], [Title] FROM [Person];
9
```

3. Haga clic en el icono **+** o **-** para expandir o contraer la región.

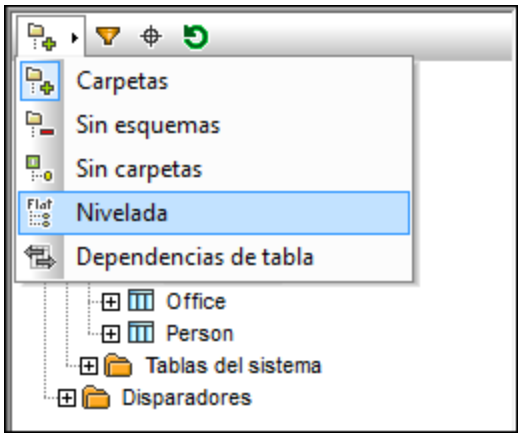
Para quitar una región:

Elimine las marcas de comentario `-- region` y `-- endregion`.

4.2.8.3 Examinar objetos de la base de datos

El panel Explorador muestra con detalle los objetos de las bases de datos que están conectadas, incluyendo sus tablas, vistas, procedimientos, etc. Además, cuando se trata de una base de datos compatible con XML, el Explorador muestra los esquemas XML registrados en una carpeta distinta.

En la parte superior del panel Explorador puede elegir una disposición para todos estos objetos.



Para seleccionar una disposición haga clic en el botón **Disposición en carpetas** y elija una opción de la lista desplegable. Observe que el aspecto del botón cambia dependiendo de la opción elegida.

- La **Disposición en carpetas** organiza los objetos de la BD en carpetas en función del tipo de objeto y estas carpetas aparecen organizadas de forma jerárquica (esta es la disposición predeterminada).
- La **Disposición sin esquemas** es similar a la disposición en carpetas, pero no presenta carpetas para los esquemas de BD. Por tanto, las tablas no se pueden categorizar según el esquema de la BD.
- La **Disposición sin carpetas** muestra todos los objetos de la BD en una jerarquía pero sin usar carpetas.
- La **Disposición nivelada** divide los objetos de la BD por tipos en el primer nivel jerárquico. Por ejemplo, las columnas no aparecen dentro de su tabla, sino que todas las columnas aparecen en una carpeta llamada `Columnas`.
- La **Disposición según dependencias** categoriza las tablas en función de la relación que tengan con las demás tablas. Hay categorías para tablas con claves foráneas, para tablas a las que hagan referencia claves foráneas y para tablas que no tengan relación alguna con otras tablas.

Desde el Explorador también puede:

- Generar instrucciones SQL 383.
- Filtrar y buscar objetos de la base de datos 389).
- Dividir las tablas en tablas del sistema y tablas del usuario.
- Actualizar el objeto raíz del origen de datos activo.

Para dividir tablas en tablas del sistema y tablas del usuario:

- En el Explorador haga clic con el botón derecho en la carpeta `Tablas` y seleccione **Ordenar por tablas de usuario y del sistema** en el menú contextual.

 © 2016-2022 Altova GmbH

Nota: esta función solo funciona si está elegida la *disposición en carpetas*, la *disposición sin esquemas* o la *disposición nivelada*.

Para actualizar el objeto raíz del origen de datos activo:

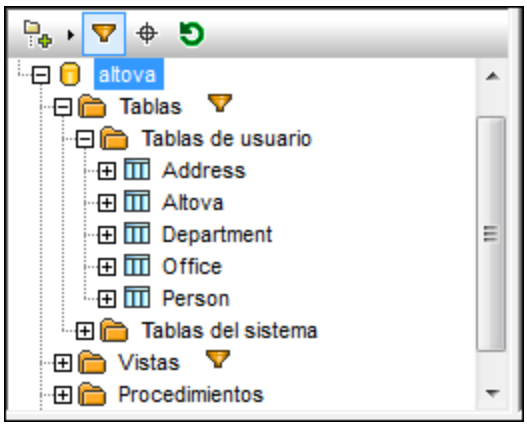
- Haga clic en el botón **Actualizar** de la barra de herramientas del Explorador.

4.2.8.3.1 Filtrar y buscar objetos de la base de datos

En el Explorador puede filtrar los objetos de base de datos (esquemas, tablas, vistas, etc.) por nombre o parte del nombre. Los objetos se filtran a medida que se teclea y el filtrado tiene en cuenta el uso de mayúsculas y minúsculas por defecto. Nota: recuerde que la función de filtrado no funciona con la *disposición sin carpetas*.

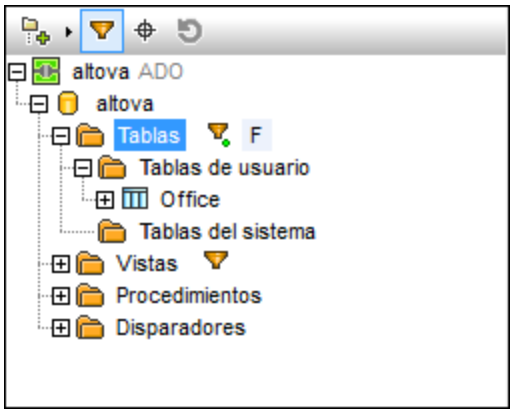
Filtrar objetos de base de datos

1. En la barra de herramientas del Explorador haga clic en **Filtrar contenido de carpeta**. Ahora aparece un icono en forma de embudo junto a cada una de las carpetas de la jerarquía.



2. Haga clic en el icono en forma de embudo de la carpeta que desea filtrar y seleccione una opción de filtrado en el menú contextual (p. ej. **Contiene**).

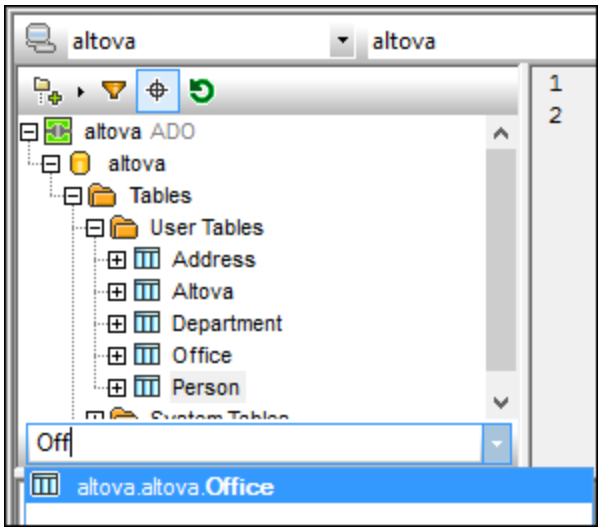
3. En el campo vacío que aparece junto al icono en forma de embudo introduzca el texto de búsqueda (p. ej. "F"). Los resultados aparecen a medida que se teclea.



Buscar objetos de base de datos

Para buscar por su nombre un elemento concreto de la base de datos puede usar funciones de filtrado o el **localizador de objetos**. Para buscar elementos de la base de datos con el localizador de objetos:

1. En la barra de herramientas del Explorador haga clic en **Localizador de objetos**.
2. En la lista desplegable que aparece introduzca el texto de búsqueda (p. ej. "Off").



3. Haga clic en un objeto de la lista de resultados para seleccionarlo en el Explorador.

4.2.8.3.2 Opciones contextuales del explorador

Las opciones del menú contextual del Explorador dependerán del tipo de objeto que esté seleccionado. Por ejemplo:

- Si se hace clic con el botón derecho en un objeto raíz, el menú contextual incluye la opción de actualizar la base de datos.

 © 2016-2022 Altova GmbH

- Si se hace clic con el botón derecho en una carpeta, el menú contextual incluye la opción **Expandir | Elementos del mismo nivel/Secundarios** y la opción **Contraer | Elementos del mismo nivel/Secundarios**.
- Si se hace clic con el botón derecho en un objeto de la base de datos, el menú contextual incluye el comando **Mostrar en Editor SQL** y algunos subcomandos (ver más abajo).

Para seleccionar varios objetos de la base de datos utilice **Mayús+Clic** o **Ctrl+Clic**.

Nota: la sintaxis de las instrucciones SQL puede depender de la base de datos que se utilice. La sintaxis utilizada en los siguientes ejemplos corresponde a Microsoft SQL Server 2014.

Estas opciones corresponden al comando **Mostrar en Editor SQL** del menú contextual del objeto raíz:

- **CREATE**: crea una instrucción CREATE para el objeto raíz seleccionado. Por ejemplo:
  `CREATE DATABASE [MYDB]`
- **DROP**: crea una instrucción DROP para el objeto raíz seleccionado. Por ejemplo:
  `DROP DATABASE [MYDB]`

Estas opciones corresponden al comando **Mostrar en Editor SQL** del menú contextual de tablas y vistas:

- **SELECT**: crea una instrucción SELECT que recupera datos de todas las columnas de la tabla de origen. Por ejemplo:
  `SELECT [DepartmentID], [Name], [GroupName], [ModifiedDate] FROM [MYDB].[HumanResources].[Department]`
- **Nombre**: devuelve el nombre de la tabla.
- **Ruta de acceso**: devuelve la ruta de acceso completa de las tablas, con este formato `NombreOrigenDatos.NombreBaseDatos.NombreEsquema.NombreTabla`.

Si seleccionó varias tablas, sus nombres y rutas de acceso se imprimen en líneas distintas y separadas por comas.

Estas opciones corresponden al comando **Mostrar en Editor SQL** del menú contextual de columnas:

- **SELECT**: crea una instrucción SELECT que recupera datos de todas las columnas seleccionadas de la tabla primaria. Por ejemplo:
  `SELECT [DepartmentID] FROM [MYDB].[HumanResources].[Department]`
- **Nombre**: devuelve el nombre de la columna seleccionada.
- **Ruta de acceso**: devuelve la ruta de acceso completa de la columna, con este formato `NombreOrigenDatos.NombreBaseDatos.NombreEsquema.NombreTabla.NombreColumna`

Si seleccionó varias columnas, sus nombres y rutas de acceso se imprimen en líneas distintas y separadas por comas.

Estas opciones corresponden al comando **Mostrar en Editor SQL** del menú contextual de restricciones:

- **Nombre**: devuelve el nombre de la restricción seleccionada.
- **Ruta de acceso**: devuelve la ruta de acceso completa de la restricción, con este formato `NombreOrigenDatos.NombreBaseDatos.NombreEsquema.NombreTabla.NombreRestricción`

Si seleccionó varias restricciones, sus nombres y rutas de acceso se imprimen en líneas distintas y separadas por comas.

Estas opciones corresponden al comando **Mostrar en Editor SQL** del menú contextual de índices:

- **Nombre**: devuelve el nombre del índice seleccionado.
- **Ruta de acceso**: devuelve la ruta de acceso completa del índice, con este formato `NombreOrigenDatos.NombreBaseDatos.NombreEsquema.NombreTabla.NombreÍndice`

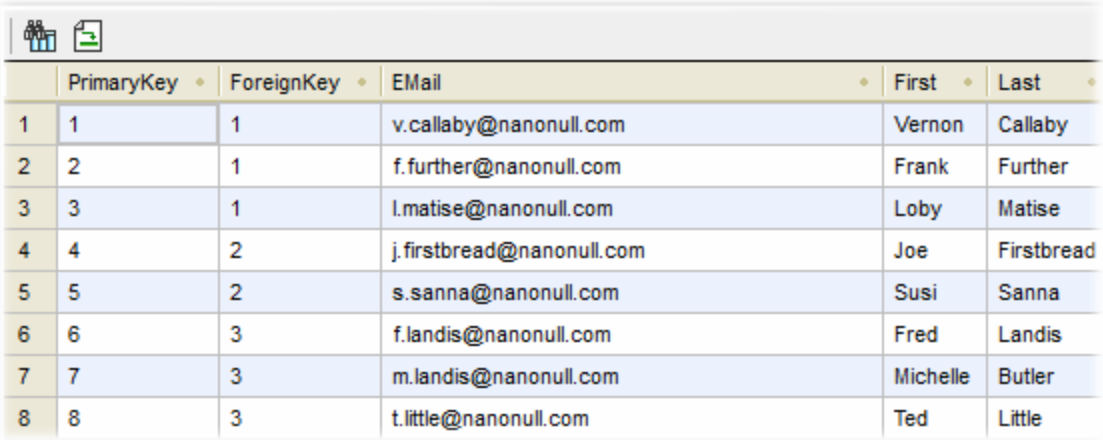
Si seleccionó varios índices, sus nombres y rutas de acceso se imprimen en líneas distintas y separadas por comas.

Si la base de datos admite esquemas XML, estas opciones corresponden al comando **Mostrar en Editor SQL** del menú contextual de los esquemas de la carpeta `Esquemas XML`:

- **Ver en XMLSpy**: abre el esquema de base de datos en XMLSpy (siempre y cuando esté instalado).
- **Administrar esquemas XML**: abre un cuadro de diálogo donde puede agregar o eliminar esquemas XML de base de datos.

4.2.8.4 Copiar, ordenar y buscar resultados de la consulta

La vista de resultados del panel *Consulta de la BD* muestra el conjunto de registros que se recuperó como resultado de la consulta de la base de datos.



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

Los botones de la barra de herramienta sirven para navegar por los resultados e instrucciones SQL y realizar búsquedas dentro de la tabla de resultados.

© 2016-2022 Altova GmbH

	Buscar: permite buscar determinado texto dentro de los resultados de la tabla. Pulse **F3** para ir a la siguiente instancia del término de búsqueda.
	Ir a la instrucción: salta al Editor SQL y resalta la instrucción SQL que produjo el resultado actual. Esta función puede ser muy práctica si el Editor SQL contiene un gran número de instrucciones.

Para seleccionar celdas de la tabla de resultados:

- Haga clic en el encabezado de columna para seleccionar una columna entera.
- Haga clic en un número de fila para seleccionar una fila entera.
- Haga clic en las celdas una por una o pulse la tecla **Ctrl** mientras hace clic para seleccionar varias celdas. Si una columna o celda contiene datos XML, también podrá copiar estos datos.

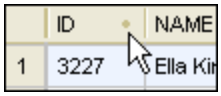
  Nota: también puede usar el menú contextual para seleccionar datos (**Seleccionar | Fila / Columna / Todo**.)

Para copiar las celdas seleccionadas en el portapapeles:

- Haga clic con el botón derecho y seleccione **Copiar celdas seleccionadas** en el menú contextual.

Para ordenar los datos.

- Haga clic con el botón derecho en cualquier parte de la columna que desea ordenar y seleccione **Orden | Ascendente** o **Descendente**.
- Haga clic en el icono de ordenación situado en el encabezado de la columna.



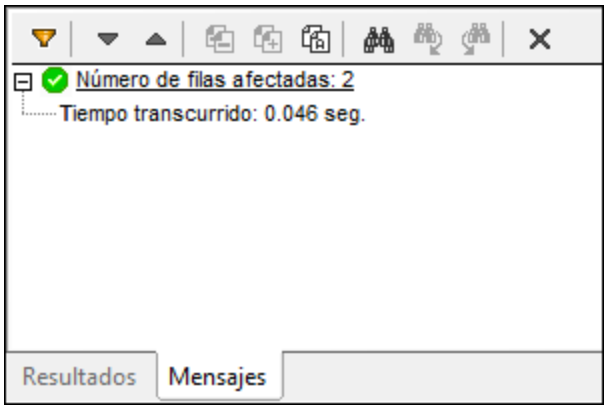
  Los datos se ordenan en función del contenido de la columna.

Para restaurar el orden predeterminado:

- Haga clic con el botón derecho en cualquier parte de la tabla y seleccione **Orden | Restaurar valor predeterminado** en el menú contextual.

4.2.8.5 Ver el estado de las consultas ejecutadas

La pestaña *Mensajes* del panel *Consulta de la BD* muestra información sobre la última instrucción SQL que se ejecutó y notifica errores y advertencias.



Puede personalizar la vista de la pestaña *Mensajes* usando diferentes filtros o navegar por los datos de fila en fila con los botones **Siguiente** y **Anterior**.

Estos son los comandos de la barra de herramientas de la pestaña *Mensajes*.

	Filtrar: abre un menú emergente donde puede filtrar el tipo de mensajes (**Resumen**, **Operación correcta**, **Advertencia**, **Error**, **Inserción automática**, **Progreso**). El tipo "Inserción automática" se refiere a aquellos mensajes provocados al insertar instrucciones SQL o construcciones SQL automáticamente en el editor SQL. El tipo "Progreso" indican el resultado de la conexión con la base de datos y el resultado de analizar datos SQL y cargar la estructura de datos. El menú emergente también incluye las opciones **Activar todos** y **Desactivar todos**.
	Siguiente: pasa al siguiente mensaje y lo resalta.
	Anterior: pasa al mensaje anterior y lo resalta.
	Copiar el mensaje seleccionado en el portapapeles
	Copiar el mensaje seleccionado en el portapapeles (incluyendo sus elementos secundarios)
	Copiar todos los mensajes en el portapapeles
	Buscar: abre el cuadro de diálogo "Buscar".
	Buscar anterior: pasa a la instancia anterior de la cadena especificada en el cuadro de diálogo "Buscar".
	Buscar siguiente: pasa a la instancia siguiente de la cadena especificada en el cuadro de diálogo "Buscar".
	Borrar: elimina todos los mensajes de la pestaña *Mensajes* del Editor SQL.

 © 2016-2022 Altova GmbH

4.2.9 Procedimientos almacenados

Los procedimientos almacenados son programas que están hospedados y se ejecutan en un servidor de BD. Es lo que se conoce como aplicaciones cliente. A menudo se escriben en algún dialecto extendido de SQL. Algunas bases de datos también admiten implementaciones en Java, .NET CLR y otros lenguajes de programación.

Entre los usos más frecuentes de los procedimientos almacenados se encuentran las consultas de la BD y la devolución de datos al cliente de llamada o la modificación de bases de datos tras la validación de los parámetros de entrada. Los procedimientos almacenados también pueden realizar otras acciones fuera de la base de datos, como enviar correos electrónicos.

Un procedimiento almacenado puede tener cero o más parámetros de entrada y salida y puede devolver cero o más conjuntos de datos valor de retorno predeterminado. En consecuencia, en MapForce puede llamar a un procedimiento almacenado de varias formas:

- Para obtener datos, como si fuera un componente de origen de la asignación. Esto se aplica a procedimientos que no toman parámetros de entrada. Cuando se ejecuta la asignación se llama al procedimiento y este devuelve un conjunto de datos o parámetros de salida. Puede asignar el conjunto de datos o los parámetros de salida, o ambos, a cualquier otro tipo de datos compatible con MapForce. Para ver un ejemplo consulte Procedimientos almacenados como fuente de datos 400.
- Como si fuera una llamada a una función, con parámetros. En este caso se suministran todos los parámetros de entrada necesarios desde la aplicación y también se puede asignar el conjunto de datos devuelto, o los parámetros de salida, o ambos, a otros destinos compatibles con MapForce. Para ver un ejemplo consulte Procedimientos almacenados con parámetros de entrada y salida 404.
- Como si fuera un componente de destino de la asignación. Un caso de uso típico sería llamar a un procedimiento almacenado con parámetros para modificar la base de datos (por ejemplo, insertar un registro). Este enfoque es adecuado si no necesita resultados del procedimiento almacenado. Además, con este enfoque puede ejecutar el procedimiento almacenado dentro de una transacción de BD que se puede revertir en caso de error. Para ver un ejemplo consulte Procedimientos almacenados en componentes de destino 408.

También puede haber casos en los que necesite llamar a procedimientos almacenados o ejecutar acciones en tablas de bases de datos en un orden concreto (primero insertar, luego actualizar, etc.). Por ejemplo, puede que necesite pasar el parámetro de salida de un procedimiento almacenado a otro procedimiento almacenado. O puede que necesite combinar los datos devueltos por un procedimiento almacenado con datos de una tabla. Ese tipo de acciones se pueden ejecutar con ayuda de las relaciones locales definidas en MapForce, incluso si la base de datos subyacente no aplica relaciones de claves principales/foráneas entre tablas. Para más información, consulte el apartado Procedimientos almacenados y relaciones locales 411.

Nota: en este apartado usaremos Microsoft SQL Server 2016 y la base de datos "AdventureWorks 2016" para explicar cómo se implementan los procedimientos almacenados en MapForce. La base de datos "AdventureWorks" se puede descargar del sitio web de CodePlex (https://github.com/Microsoft/sql-server-samples/releases/tag/adventureworks).

Notas sobre compatibilidad

- Los procedimientos almacenados solo se pueden usar en el motor de ejecución integrado BUILT-IN. Los procedimientos almacenados no son compatibles con la generación de código en C++, C# o Java.

- Tipos de parámetros de entrada/salida: por lo general no son compatibles los tipos definidos por el usuario, tipos cursor, tipos variant y muchos otros tipos de datos de BD menos comunes (como tipos matriz, geometry, CLR, etc.).

- No es compatible la sobrecarga de procedimientos y funciones (varias definiciones de rutinas con el mismo nombre y parámetros diferentes).

- Algunas bases de datos admiten el uso de valores predeterminados en los parámetros de entrada, pero MapForce no lo admite por ahora. No puede omitir parámetros de entrada en la asignación para usar valores predeterminados.

- Los procedimientos almacenados que devuelven varios conjuntos de registros son compatibles o no dependiendo de la combinación de controlador y API de base de datos (ODBC/ADO/ADO.NET/JDBC). Solamente son compatibles los procedimientos que devuelven el mismo número de conjuntos de registro con una estructura de columnas fijas.

- Recomendamos utilizar siempre que se pueda la versión más reciente del controlador nativo de BD que mantenga el proveedor de la BD. No recomendamos utilizar controladores puente como ODBC to ADO Bridge ni ODBC to JDBC Bridge.

- También puede habilitar las transacciones de BD para procedimientos almacenados a los que se llama como datos de destino (*consulte* *Procedimientos almacenados en componentes de destino* 408). Las transacciones no son compatibles con los procedimientos almacenados a los que se llama como datos de origen (sin parámetros de entrada) o aquellos a los que se llama como función (con parámetros de entrada y salida).

La siguiente tabla muestra información específica de cada una de las BD compatibles.

Base de datos	Notas sobre compatibilidad
Access	• Los procedimientos almacenados de las BD de Microsoft Access tienen una funcionalidad muy limitada y no son compatibles con MapForce.
DB2	• Compatibles con MapForce: procedimientos almacenados, funciones escalares y funciones con valores de tabla. • No se admiten los valores de retorno de los procedimientos almacenados de BD2 porque no se pueden leer con las APIs de BD que usa MapForce. • Los procedimientos almacenados de valores de rendimiento de DB2 no son compatibles porque no se pueden leer con las APIs de DB que usa MapForce. • Las funciones con valores de fila (RETURNS ROW) no son compatibles. • Se recomienda instalar como mínimo "IBM_DB2 9.7 Fix Pack 3a" para evitar que ocurra un error confirmado con el driver JDBC al leer errores/advertencias tras la ejecución. Esto también soluciona un problema con el proveedor ADO que resulta en que falte un bloque de filas del resultado.
Firebird	• Compatibles con MapForce: procedimientos almacenados y funciones con valores de tabla
Informix	• Compatibles con MapForce: procedimientos almacenados y funciones con valores de tabla.
MariaDB	• Compatibles con MapForce: procedimientos almacenados, funciones escalares

 © 2016-2022 Altova GmbH

Base de datos	Notas sobre compatibilidad
MySQL	• Compatibles con MapForce: procedimientos almacenados, funciones escalares • MySQL incluye total compatibilidad para procedimientos almacenados y funciones a partir de la versión 5.5. Si usa una versión anterior, la funcionalidad será limitada en MapForce.
Oracle	• Compatibles con MapForce: procedimientos almacenados, funciones escalares y funciones con valores de tabla. • Se recomienda usar un controlador Oracle nativo en lugar del proveedor **Microsoft OLE DB Provider for Oracle**. • Oracle tiene una forma especial de devolver conjuntos de resultados al cliente que consiste en usar parámetros de salida de tipo REF CURSOR. Esto es compatible con MapForce en el caso de los procedimientos almacenados, pero no en el de las funciones. Por tanto, los nombres y el número de los conjuntos de registros siempre son fijos para procedimientos almacenados de Oracle.
PostgreSQL	• Compatibles con MapForce: funciones escalares, funciones con valores de fila, funciones con valores de tabla. También son compatibles los procedimientos almacenados y las funciones independientes, así como los definidos dentro de un paquete Oracle. • En PostgreSQL los parámetros de salida definidos en una función describen las columnas del conjunto de resultados. Esta información se utiliza automáticamente en MapForce (no es necesaria la detección por ejecución ni la entrada manual de conjuntos de registros). Los parámetros de tipo refcursor no son compatibles.
Progress OpenEdge	• Compatibles con MapForce: procedimientos almacenados.
SQL Server	• Compatibles con MapForce: procedimientos almacenados, funciones escalares y funciones con valores de tabla. • Recomendamos usar el controlador **SQL Server Native Client** más reciente en lugar de **Microsoft OLE DB Provider for SQL Server**. • La API de ADO tiene algunas limitaciones de compatibilidad con algunos tipos de datos introducidos con SQL Server 2008 (`datetime2`, `datetimeoffset`). Si detecta problemas de truncamiento de datos con estos tipos temporales cuando use ADO con SQL Server Native Client, puede establecer el argumento de cadena de conexión `DataTypeCompatibility=80` o usar ODBC. • Los procedimientos de SQL Server tienen un parámetro de rendimiento implícito de tipo `int null` que está disponible para asignaciones. Si el procedimiento omite una instrucción RETURN el valor resultante es 0.
SQLite	• SQLite no usa procedimientos almacenados.
Teradata	• Compatibles con MapForce: procedimientos almacenados, macros. • No admite funciones escalares, funciones agregadas ni funciones de tabla • Problema conocido: el driver Teradata ODBC no rellena los valores de parámetros de salida después del procedimiento de llamada.

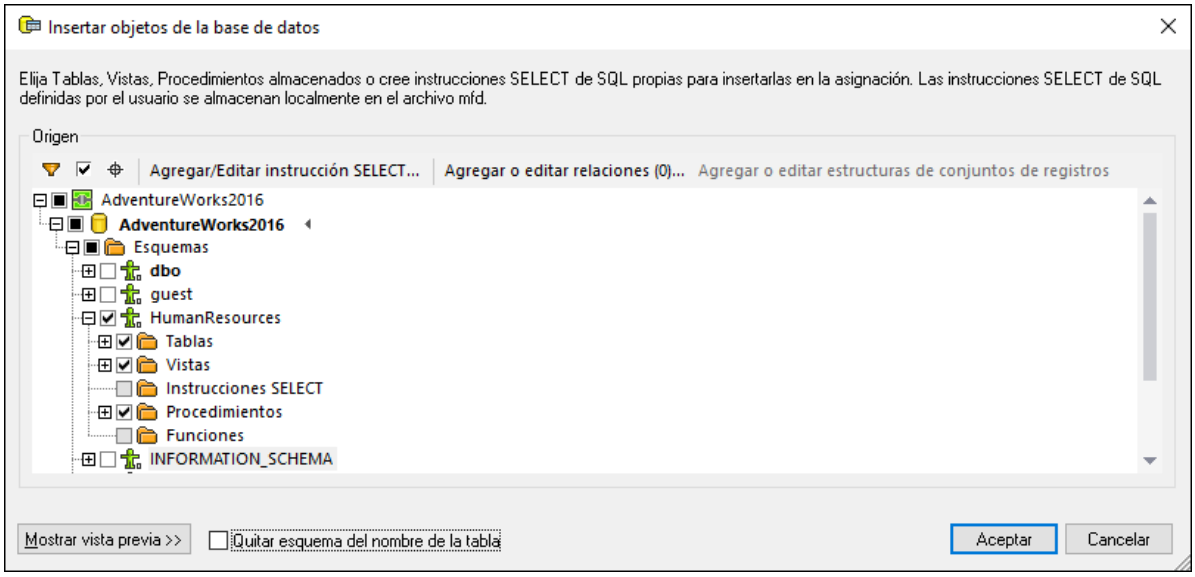
4.2.9.1 Agregar procedimientos almacenados a la asignación

En el área de la asignación los procedimientos almacenados se muestran como parte del componente de BD al que pertenecen. Para que los procedimientos almacenados sean visibles en el componente de BD debe seleccionarlos explícitamente al añadir el componente de BD a la asignación, como se muestra más abajo. En este ejemplo nos conectamos a la base de datos "AdventureWorks" que se está ejecutando en SQL Server. Las instrucciones son parecidas a las que debe seguir con otras bases de datos.

En el caso de las bases de datos Oracle, los procedimientos almacenados o las funciones pueden ser independientes o parte de paquetes Oracle. Puede añadir las dos categorías a la asignación. Los procedimientos almacenados o las funciones que pertenezcan a un paquete aparecen bajo el nombre de paquete correspondiente en el cuadro de diálogo "Insertar objetos de la base de datos" (*imagen siguiente*).

Para agregar procedimientos almacenados a la asignación:

1. Elija una de estas opciones:
   - En el menú **Insertar**, haga clic en **Base de datos**.
   - Haga clic en el botón de la barra de herramientas **Insertar base de datos** ( ).
2. Siga los pasos que le indique el asistente para conectarse a la base de datos hasta que llegue al cuadro de diálogo "Insertar objetos de la base de datos". Para leer instrucciones detalladas para cada tipo de base de datos consulte los ejemplos de la sección Ejemplos de conexión a bases de datos.



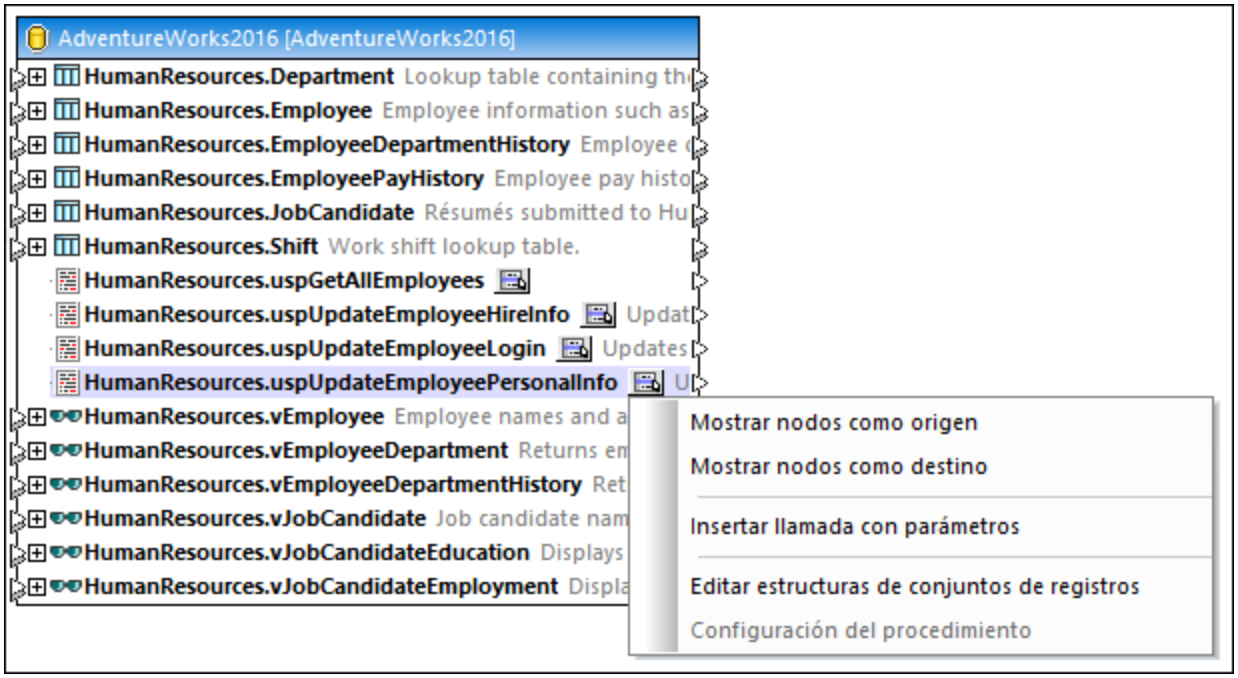
3. Marque las casillas de los objetos de BD que necesita que aparezcan en la asignación y haga clic en **Aceptar**. En este ejemplo hemos seleccionado todas las tablas, vistas y procedimientos almacenados disponibles en el esquema "HumanResources".

Notas

© 2016-2022 Altova GmbH

- Puede cambiar los objetos seleccionados más tarde haciendo clic con el botón derecho en la barra de título de un componente de BD y seleccionando **Componente | Agregar, quitar o editar componentes de la base de datos** del menú contextual.
- Su cuenta de usuario de BD debe tener derechos para ver y ejecutar procedimientos almacenados en la base de datos.

Ahora el componente de BD se ha agregado a la asignación. Observe que puede identificar los procedimientos almacenados por el icono . Además, las tablas, vistas y los procedimientos se ordenan alfabéticamente en el componente de BD.



El botón para mostrar el menú contextual ( ) que hay junto a cada procedimiento almacenado permite configurar la forma en que se llama a ese procedimiento almacenado, además de otras opciones relacionadas con él:

Opción	Uso
Mostrar nodos como origen	Seleccione esta opción si quiere llamar a un procedimiento almacenado *sin parámetros* para obtener datos de una BD y asignarlos a otro componente compatible con MapForce (XML, texto, EDI, etc.). Para ver un ejemplo consulte Procedimientos almacenados como fuente de datos 400.
Mostrar nodos como destino	Seleccione esta opción si quiere llamar a un procedimiento almacenado para modificar la BD o ejecutar otra acción específica para la que no necesite el resultado del procedimiento almacenado. Para ver un ejemplo consulte Procedimientos almacenados en componentes de destino 408.

Insertar llamada con parámetros	Seleccione esta opción si quiere llamar a un procedimiento almacenado *con parámetros* para asignar los datos devueltos a otro componente compatible con MapForce. Para ver un ejemplo consulte Procedimientos almacenados con parámetros de entrada y salida 404.
Editar estructuras de conjuntos de registros	Solo se aplica a procedimientos almacenados que devuelven conjuntos de datos. Seleccione esta opción para ejecutar los procedimientos almacenados una vez para que MapForce pueda determinar la estructura del conjunto de datos devuelto y mostrarla en la asignación. Otra opción, si no quiere ejecutar el procedimiento almacenado al crear el diseño, es definir manualmente la estructura del conjunto de datos.
Configuración del procedimiento	Solo se aplica a procedimientos almacenados que se hayan configurado como "destino" (es decir, los que actualizan la BD). Seleccione esta opción para configurar otras opciones relativas al procedimiento, como ejecutar una consulta SQL personalizada antes de llamar al procedimiento o habilitar transacciones de BD.

4.2.9.2 Procedimientos almacenados como fuente de datos

En este ejemplo explicamos cómo llamar a un procedimiento almacenado que no toma parámetros de entrada y únicamente obtiene datos de la base de datos. En este caso el procedimiento almacenado actúa como componente de origen de la asignación y los datos que obtiene se pueden asignar a otro componente de destino compatible con MapForce. Si necesita llamar a un procedimiento almacenado con parámetros de entrada consulte el apartado Procedimientos almacenados con parámetros de entrada y salida 404.

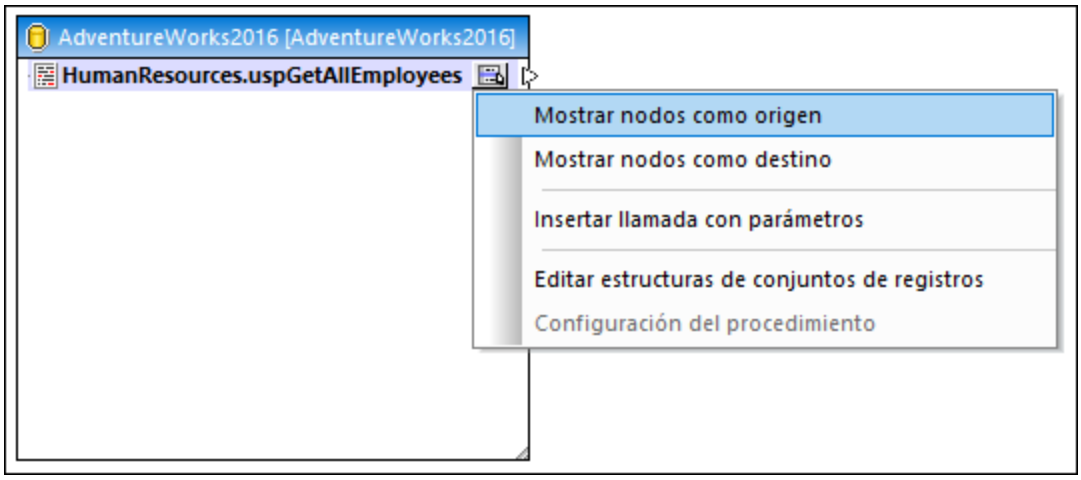
Primero vamos a crear el procedimiento almacenado de ejemplo en la base de datos "AdventureWorks". Para ello, ejecute el siguiente script en la BD. Puede hacerlo desde una ventana de consultas de la aplicación de **Microsoft SQL Server Management Studio** o directamente desde la pestaña *Consulta de la BD* de MapForce (*consulte Examinar y consultar bases de datos* 380). En ambos casos, asegúrese de que su cuenta de usuario de la base de datos tiene permisos para crear procedimientos almacenados.

```
CREATE PROCEDURE HumanResources.uspGetAllEmployees
AS
    SELECT LastName, FirstName, JobTitle, Department
    FROM HumanResources.vEmployeeDepartment
```

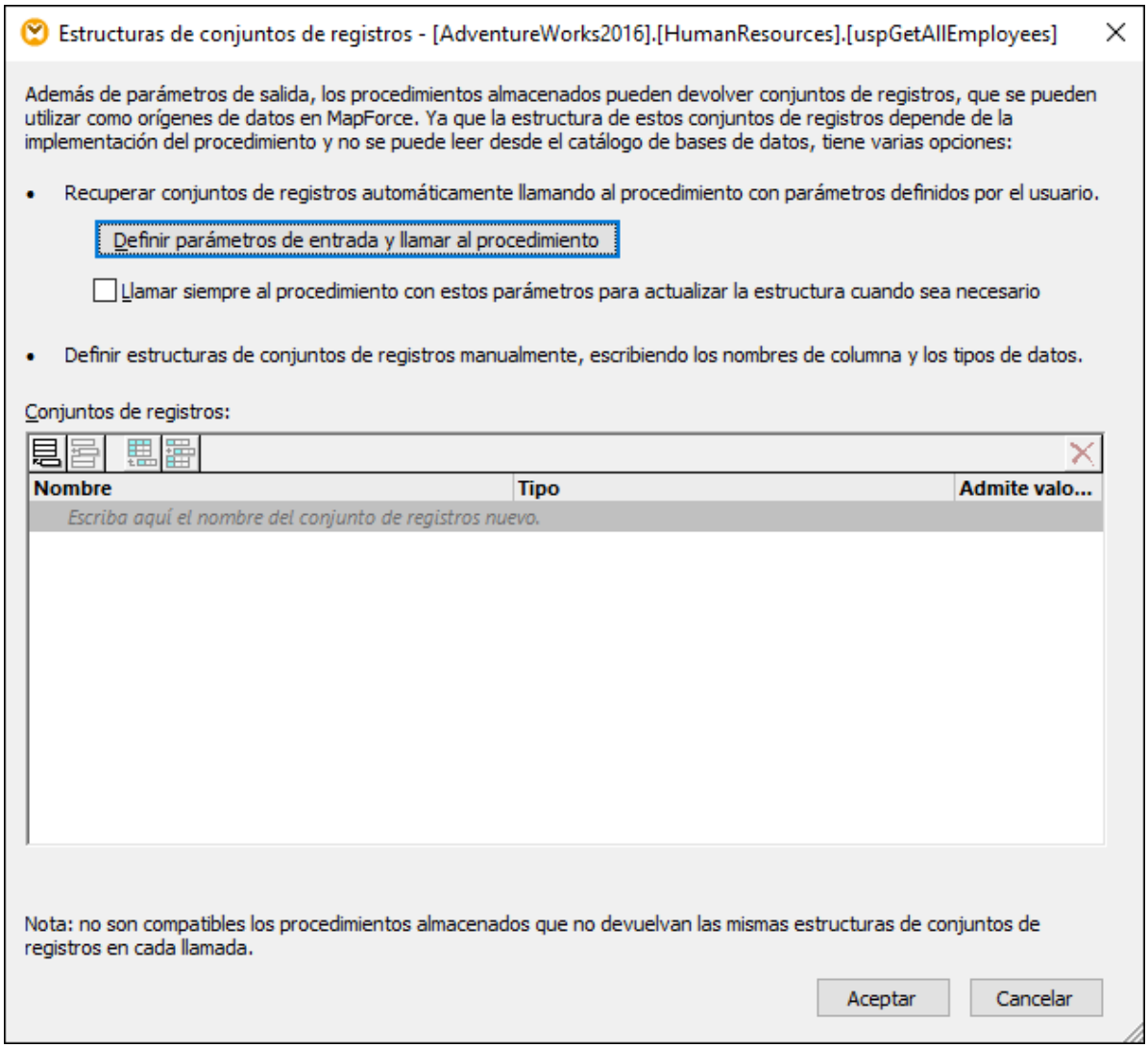
El procedimiento almacenado anterior devuelve información sobre los empleados de la vista **vEmployeeDepartment**. Los siguientes pasos muestran cómo crear una asignación que consume datos devueltos por este procedimiento.

1. Conéctese a la BD "AdventureWorks" desde MapForce y añada el procedimiento almacenado a la asignación como se describe en Agregar procedimientos almacenados a la asignación 398. Asegúrese de que su cuenta de usuario de la base de datos tiene permisos para ver y ejecutar procedimientos almacenados.
2. Haga clic en el botón para mostrar el menú contextual (), junto al procedimiento almacenado, y seleccione **Mostrar nodos como origen**.

 © 2016-2022 Altova GmbH



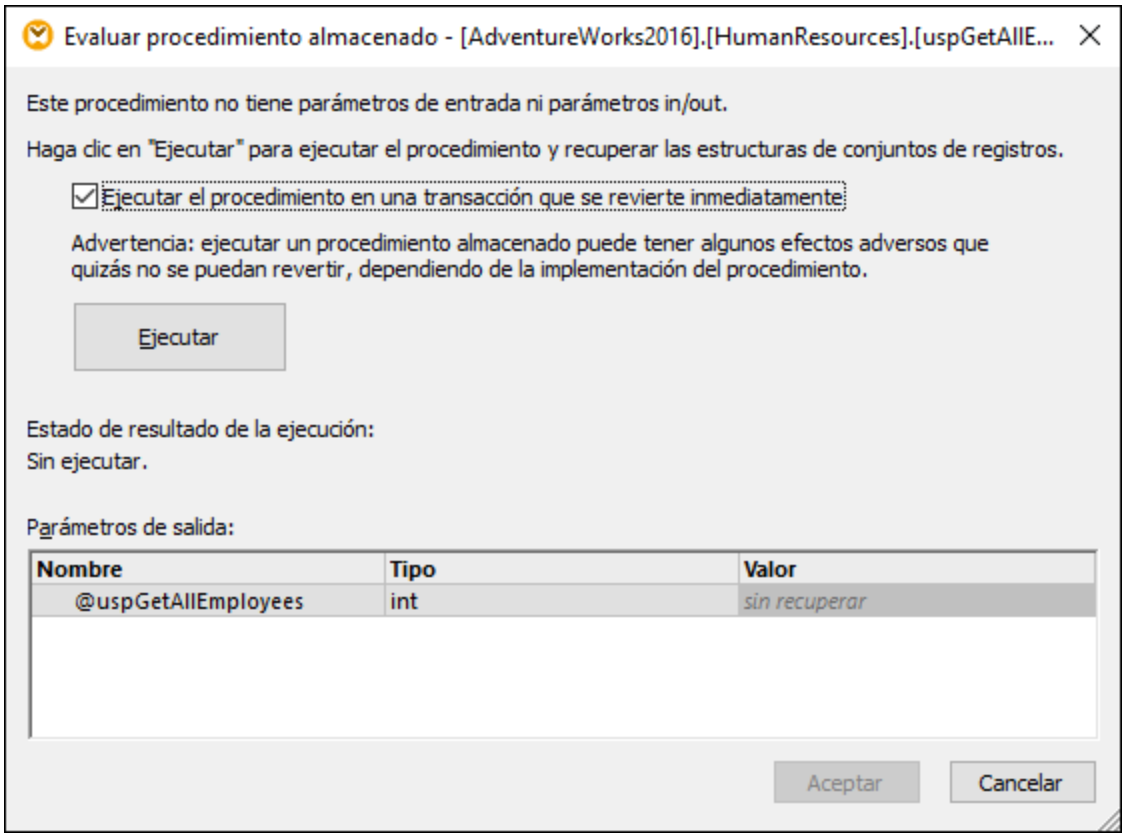
3. Vuelva a hacer clic en el botón para mostrar el menú contextual ( ) y seleccione **Editar estructuras de conjuntos de registros**. Aparecerá el cuadro de diálogo "Estructuras de conjuntos de registros".



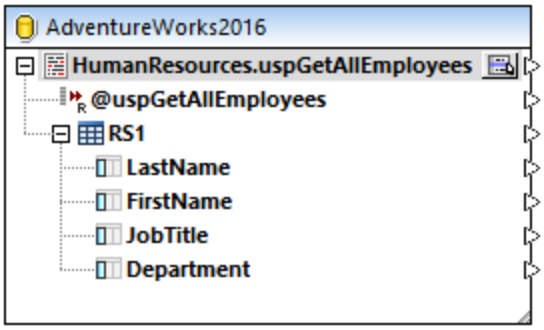
Llamar a un procedimiento almacenado en el momento de diseñar la asignación puede tener efectos secundarios (dependiendo de la implementación del procedimiento). Si no quiere ejecutar el procedimiento almacenado en el momento de diseñar la asignación, entonces no haga clic en **Ejecutar**, como se describe en los siguientes pasos. En vez de eso, defina el conjunto de registros esperado en el cuadro de diálogo "Estructuras de conjuntos de registros" añadiendo manualmente conjuntos de registros y sus columnas asociadas. Use los botones **Agregar conjunto** de registros y **Agregar columna** del cuadro de diálogo "Estructuras de conjuntos de registros".

4. Haga clic en **Definir parámetros de entrada y llamar al procedimiento** y después haga clic en **Aceptar**. Aparecerá el cuadro de diálogo "Evaluar procedimiento almacenado".

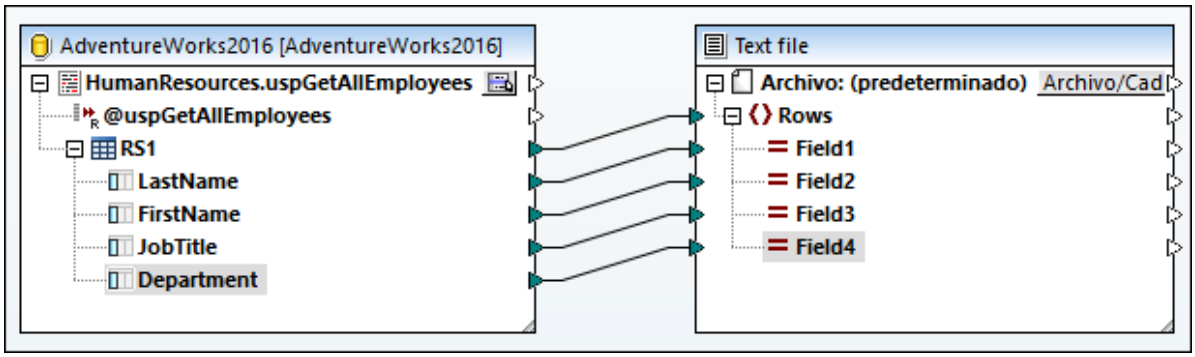
 © 2016-2022 Altova GmbH



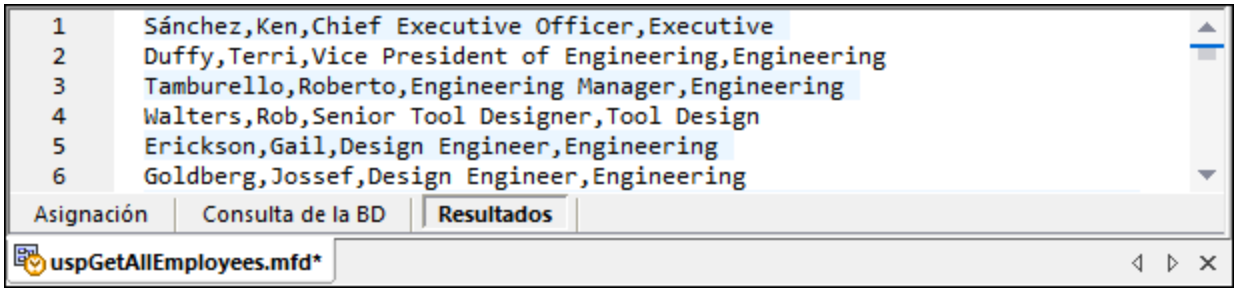
5. Haga clic en **Ejecutar** y después en **Aceptar**. La estructura del conjunto de datos ("RS1") ahora es visible tanto en cuadro de diálogo "Estructuras de conjuntos de registros" como en la asignación.



6. En este punto puede añadir un componente de destino en el que se escribirán los datos obtenidos. En este ejemplo los datos se escribirán en un archivo CSV. En el menú **Insertar**, haga clic en **Archivo de texto** y añada un componente CSV a la asignación. Para más información consulte el apartado Archivos CSV y archivos de texto 445.



Ahora puede obtener una vista previa de la asignación. Haga clic en el botón **Resultados** y observe el resultado de la asignación en el panel **Resultados**, por ejemplo:



4.2.9.3 Procedimientos almacenados con parámetros de entrada y salida

En este ejemplo explicamos cómo llamar a un procedimiento que toma parámetros de entrada y también obtiene resultados de la BD. En este caso, para llamar al procedimiento almacenado usaremos un método parecido al que usamos para llamar a un servicio web o una función, y los datos que obtenga el procedimiento se pueden asignar a cualquier otro componente de destino compatible con MapForce.

Primero vamos a crear el procedimiento almacenado de ejemplo en la base de datos "AdventureWorks". Para ello, ejecute el siguiente script en la BD. Puede hacerlo desde una ventana de consultas de la aplicación de **Microsoft SQL Server Management Studio** o directamente desde la pestaña *Consulta de la BD* de MapForce (*consulte* *Examinar y consultar bases de datos* 380). En ambos casos, asegúrese de que su cuenta de usuario de la base de datos tiene permisos para crear procedimientos almacenados.

```
CREATE PROCEDURE Production.uspSearchProducts
   @SearchString nvarchar(50)
   ,@MaxPrice money
   ,@ComparePrice money OUTPUT
AS
BEGIN
   SET NOCOUNT ON
   SELECT pr.[Name], pr.ListPrice FROM [Production].[Product] pr
   WHERE pr.[Name] like @SearchString AND  pr.ListPrice < @MaxPrice

   SET @ComparePrice = @MaxPrice
```

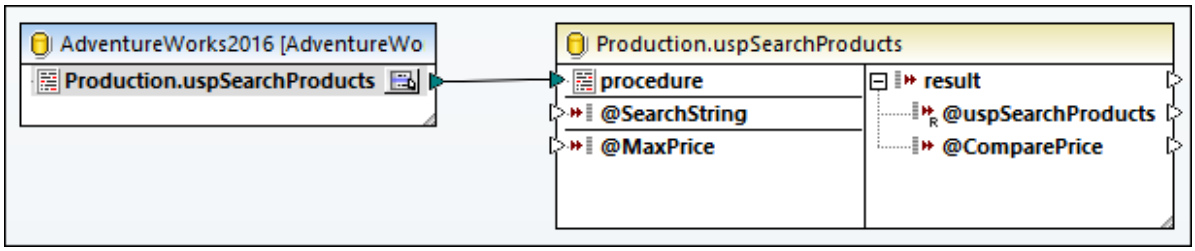
 © 2016-2022 Altova GmbH

```
    RETURN @ComparePrice
END
```

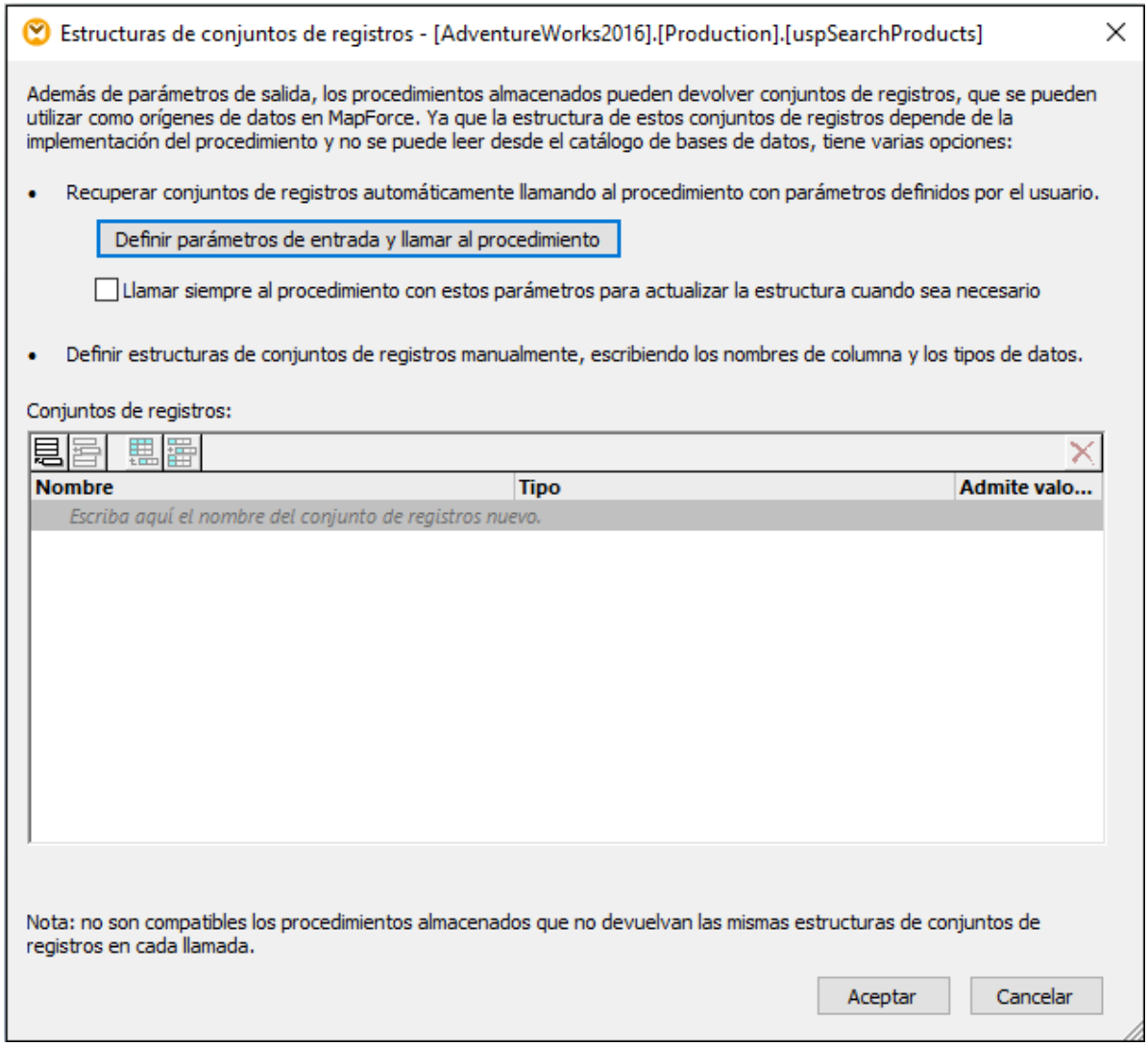
El procedimiento almacenado anterior obtiene un conjunto de registros que contienen información sobre productos. Toma dos parámetros como entrada: una cadena de texto con el nombre del producto (`@SearchString`) y el precio máximo de ese producto (`@MaxPrice`). Además del conjunto de registros y del parámetro de retorno predeterminado también obtiene un parámetro de salida (`@ComparePrice`).

Para crear una asignación que consuma datos devueltos por este procedimiento siga los pasos que explicamos a continuación:

1. Conéctese a la BD "AdventureWorks" desde MapForce y añada el procedimiento almacenado a la asignación como se describe en Agregar procedimientos almacenados a la asignación 398. Asegúrese de que su cuenta de usuario de la base de datos tiene permisos para ver y ejecutar procedimientos almacenados.
2. Haga clic en el botón para mostrar el menú contextual (), junto al procedimiento almacenado, y seleccione **Insertar llamada con parámetros**. El procedimiento almacenado ahora aparece en un componente distinto de la asignación que presenta dos partes: a la izquierda aparecen los parámetros de entrada y a la derecha están los resultados y el parámetro de salida.



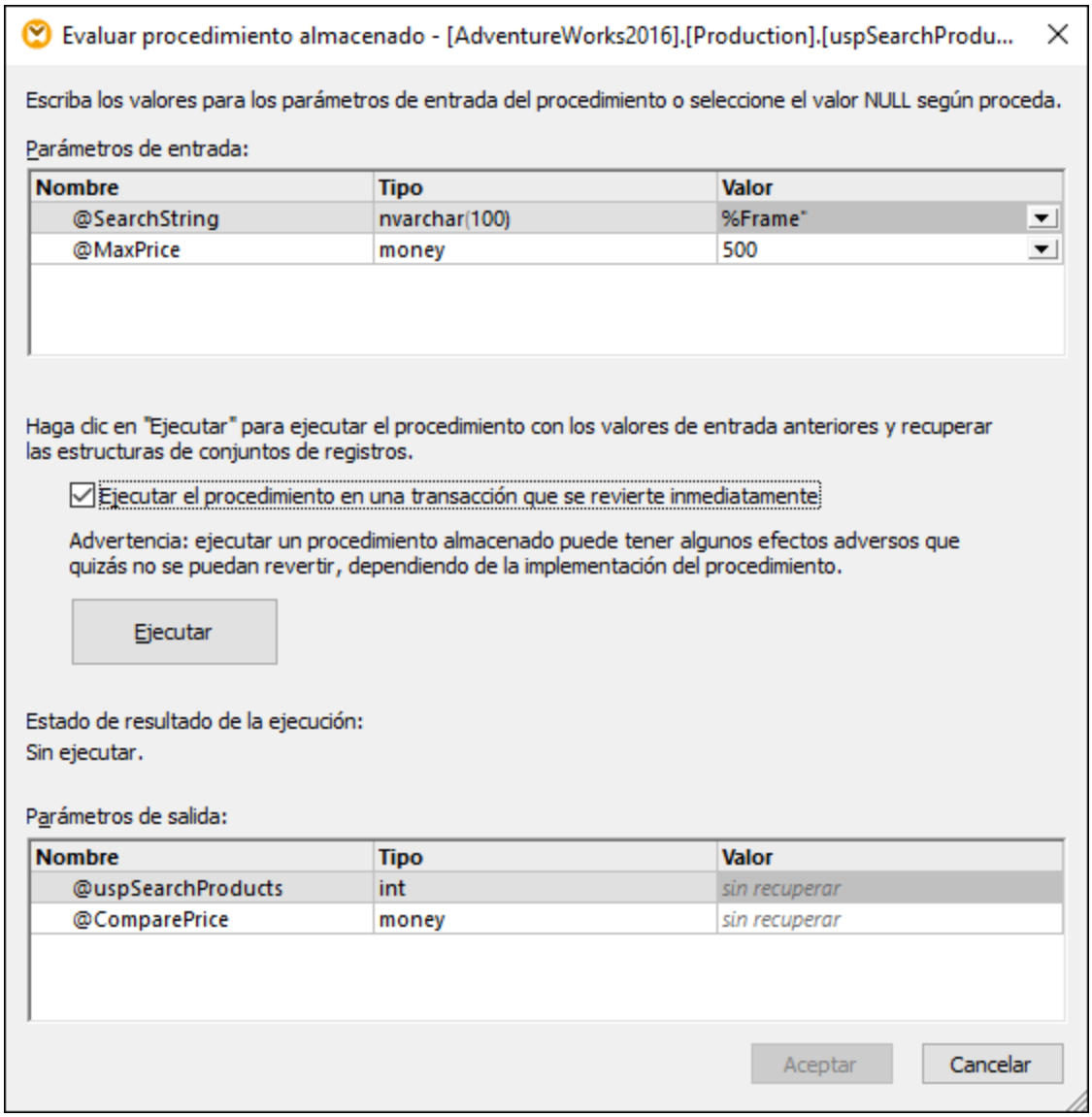
3. Vuelva a hacer clic en el botón para mostrar el menú contextual () y seleccione **Editar estructuras de conjuntos de registros**. Esto es necesario para suministrar a MapForce la información sobre la estructura de los registros devuelta por el procedimiento. Aparece el cuadro de diálogo "Estructuras de conjuntos de registros".



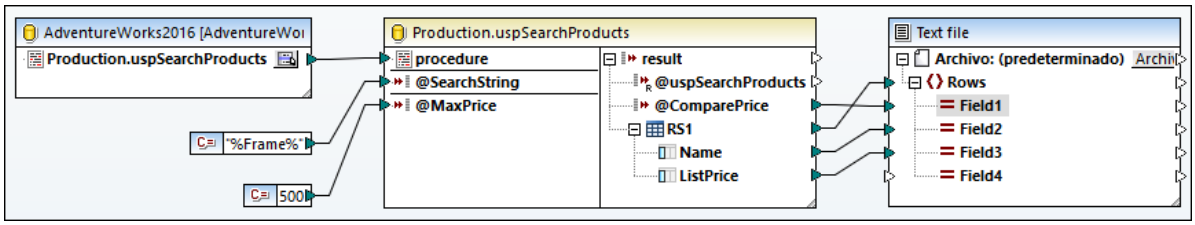
Llamar a un procedimiento almacenado en el momento de diseñar la asignación puede tener efectos secundarios (dependiendo de la implementación del procedimiento). Si no quiere ejecutar el procedimiento almacenado en el momento de diseñar la asignación, entonces no haga clic en **Ejecutar**, como se describe en los siguientes pasos. En vez de eso, defina el conjunto de registros esperado en el cuadro de diálogo "Estructuras de conjuntos de registros" añadiendo manualmente conjuntos de registros y sus columnas asociadas. Use los botones **Agregar conjunto** de registros y **Agregar columna** del cuadro de diálogo "Estructuras de conjuntos de registros".

4. Haga clic en **Definir parámetros de entrada y llamar al procedimiento**. Aparecerá el cuadro de diálogo "Evaluar procedimiento almacenado".

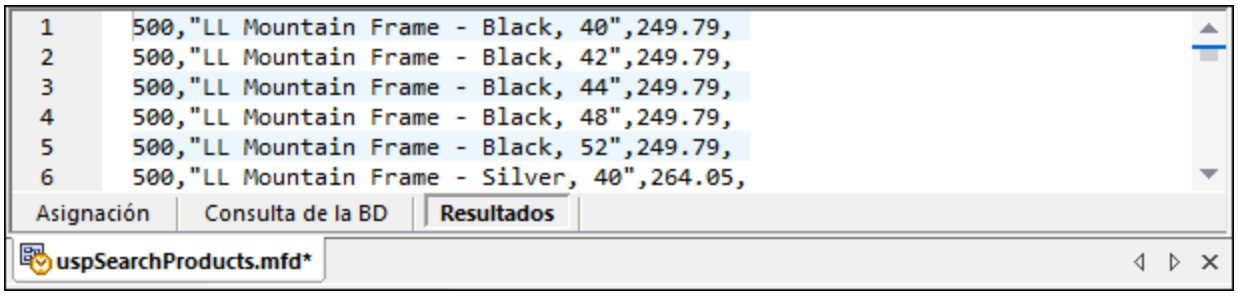
 © 2016-2022 Altova GmbH



5. Rellene los valores del parámetro como se muestra en la imagen anterior y haga clic en **Ejecutar**.
6. Haga clic en **Aceptar**. La estructura del conjunto de registros ("RS1") ahora es visible tanto en el cuadro de diálogo "Estructuras de conjuntos de registros" como en la asignación.
7. En este punto puede añadir un componente de destino en el que se escribirán los datos obtenidos. En este ejemplo los datos se escribirán en un archivo CSV. En el menú **Insertar**, haga clic en **Archivo de texto** y añada un componente CSV a la asignación (*véase también* *Archivos CSV y archivos de texto* 445). 445. A continuación dibuje las conexiones de la asignación como se muestra en la imagen siguiente. Observe que los parámetros de entrada del procedimiento se dan en forma de constantes. Para más información sobre constantes consulte el apartado Agregar una constante a una asignación de datos 550.



Ahora puede obtener una vista previa de la asignación. Haga clic en el botón **Resultados** y observe el resultado de la asignación en el panel **Resultados**, por ejemplo:



4.2.9.4 Procedimientos almacenados en componentes de destino

En este ejemplo explicamos cómo llamar a un procedimiento almacenado que toma parámetros de entrada y actualiza una base de datos. Al llamar al procedimiento con este método se pueden habilitar las transacciones y revertir la acción en caso de error, así como añadir una instrucción SQL personalizada antes de llamar al procedimiento. En este ejemplo se asume que el procedimiento actúa como un componente de destino en MapForce y que no nos interesan los resultados que este devuelve. Para aprender cómo pasar parámetros y también datos de asignación devueltos a un procedimiento almacenado, consulte el apartado Procedimientos almacenados con parámetros de entrada y salida 404.

Primero vamos a crear el procedimiento almacenado de ejemplo en la base de datos "AdventureWorks". Para ello, ejecute el siguiente script en la BD. Puede hacerlo desde una ventana de consultas de la aplicación de **Microsoft SQL Server Management Studio** o directamente desde la pestaña *Consulta de la BD* de MapForce (*consulte* *Examinar y consultar bases de datos* 380). En ambos casos, asegúrese de que su cuenta de usuario de la base de datos tiene permisos para crear procedimientos almacenados.

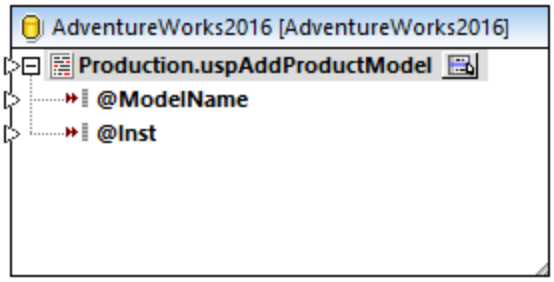
```
CREATE PROCEDURE Production.uspAddProductModel
   @ModelName nvarchar(50)
   ,@Inst xml
AS
BEGIN
INSERT INTO [Production].[ProductModel]
           ([Name]
           ,[Instructions]
           ,[rowguid]
           ,[ModifiedDate])
     VALUES
           (@ModelName
```

```
            ,@Inst
            ,NEWID()
            ,GETDATE())
END
```

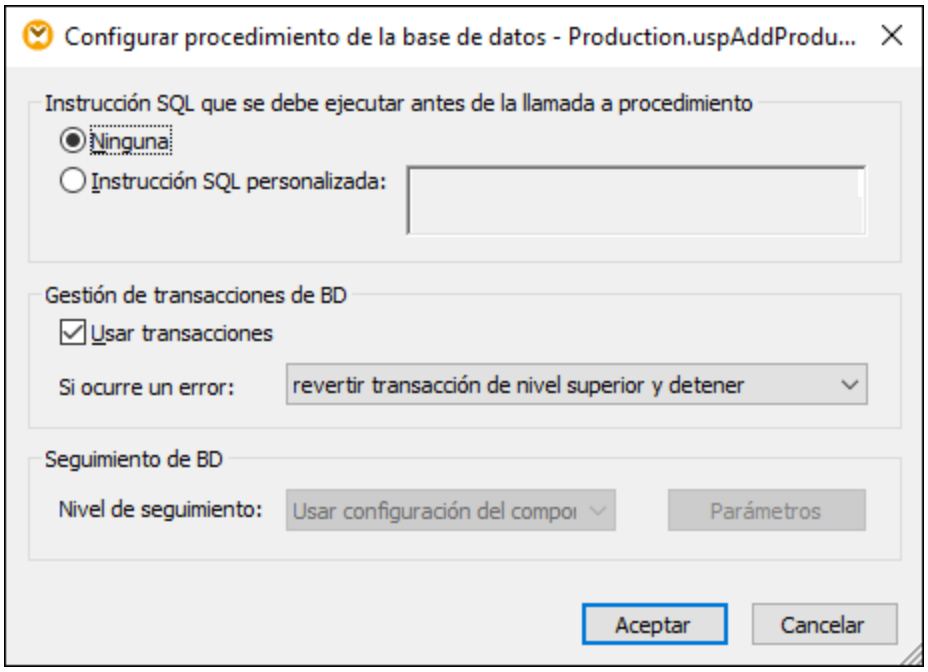
El procedimiento almacenado anterior toma dos parámetros (`@ModelName`, `@Inst`) como entrada e inserta los valores correspondientes en la tabla `ProductModel` de la BD AdventureWorks junto con otros datos generados por la base de datos.

Los siguientes pasos explican cómo crear una asignación que consuma fatos devueltos por este procedimiento.

1. Conéctese a la BD "AdventureWorks" desde MapForce y añada el procedimiento almacenado a la asignación como se describe en el apartado Agregar procedimientos almacenados a la asignación 398. Asegúrese de que su cuenta de usuario de la base de datos tiene permisos para ver y ejecutar procedimientos almacenados.
2. Haga clic en el botón para mostrar el menú contextual (), junto al procedimiento almacenado, y seleccione **Mostrar nodos como destino**. El procedimiento almacenado ahora aparece como componente de destino en la asignación, donde encontramos los parámetros de entrada se encuentran a la izquierda.

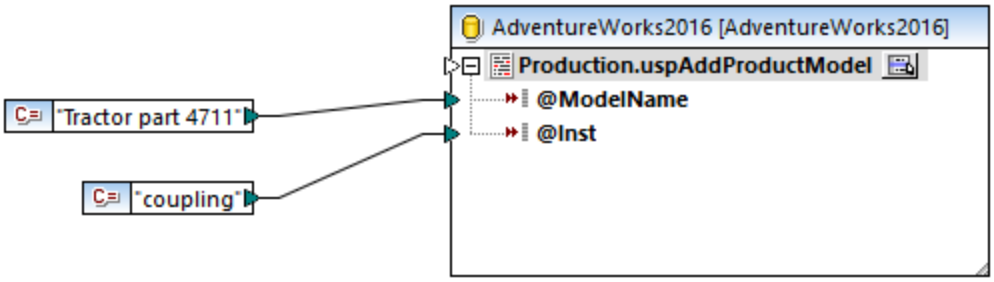


3. Vuelva a hacer clic en el botón para mostrar el menú contextual () y seleccione **Configuración del procedimiento**. Este paso opcional permite ejecutar el procedimiento almacenado dentro de una transacción que se puede revertir en caso de error. También puede añadir una instrucción SQL personalizada que se ejecute antes de llamar al procedimiento.
4. Marque la casilla **Usar transacciones**.



Nota: en este ejemplo hemos deshabilitado el seguimiento de BD a nivel de los componentes de la BD y no se ha indicado que se inicie ningún seguimiento. Sin embargo, puede habilitar el seguimiento de BD para procedimientos almacenados si lo necesita. Para más información consulte Seguimiento de BD y registro de errores 435.

5. Añada el componente de origen que proporciona los datos que se deben insertar en la BD. En este ejemplo los datos de origen vienen dados por constantes; sin embargo, cualquier otro componente de origen compatible con MapForce podría servir como datos de entrada. Para más información sobre constantes consulte el apartado Agregar una constante a una asignación de datos 550.



Esta asignación actualiza una base de datos, por lo que no existe una vista previa de sus resultados, como ocurre con otras asignaciones. Al hacer clic en el botón **Resultados**, en la pestaña correspondiente aparecerá una especie de SQL con pistas sobre cómo se modificará la BD. Si habilita las transacciones, estas ocurrirán como describen los comentarios de la pestaña **Resultados**.

 © 2016-2022 Altova GmbH

```
7
8   SET QUOTED_IDENTIFIER ON
9
10  -- begin transaction
11
12  EXECUTE NULL = [Production].[uspAddProductModel] 'Tractor part 4711', 'coupling'
13
14  -- commit transaction
15
```

Asignación | Consulta de la BD | Resultados

uspAddProductModel.mfd

Esa especie de SQL que aparece en el panel **Resultados** no muestra los comandos reales de las transacciones, sino solo pistas (como comentarios). Los comandos SQL de verdad se envían a la API de BD subyacente.

Puede ejecutar la asignación en la BD de dos maneras:

- En el menú **Resultados**, haga clic en **Ejecutar script SQL**.
- Haga clic en el botón de la barra de herramientas **Ejecutar script SQL** .

Procedimientos almacenados y entradas duplicadas

Si necesita asignar datos de distintos orígenes desde la asignación a un mismo procedimiento almacenado puede duplicar ese procedimiento almacenado para que acepte varios orígenes. Para ello, haga clic con el botón derecho en el elemento del procedimiento almacenado del componente y seleccione **Agregar un duplicado de entrada** en el menú contextual (véase también Duplicar entradas 81). Cuando ejecute la asignación, esta llamará a esos procedimientos almacenados una vez por cada duplicado de entrada.

Observe que el comando **Agregar un duplicado de entrada** está deshabilitado para los parámetros de procedimientos almacenados porque cada parámetro es un valor atómico (y también podría ser "nullable").

4.2.9.5 Procedimientos almacenados y relaciones locales

Las relaciones locales son relaciones lógicas entre campos de BD que puede crear en MapForce, lo que le evita tener que cambiar la BD subyacente (*consulte también* *Definir relaciones locales* 267). Puede definir relaciones locales no solo para campos de BD sino también para procedimientos almacenados, en componentes tanto de entrada como de salida.

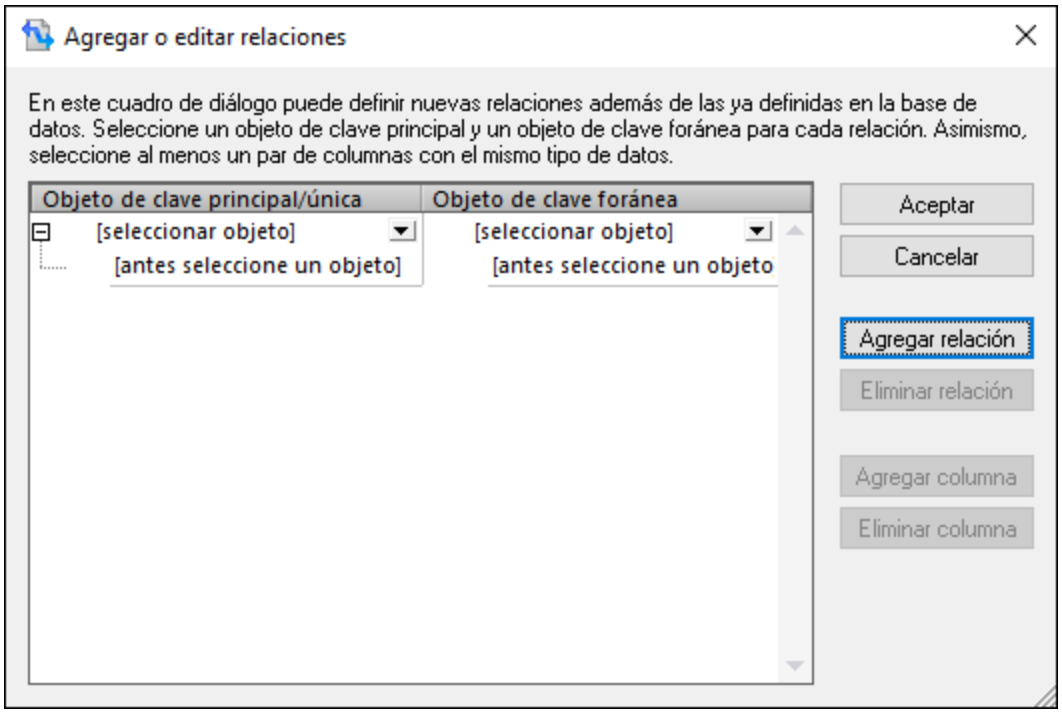
En componentes de entrada las relaciones locales permiten leer fácilmente datos de objetos relacionados, como leer IDs de una tabla de BD y llamar a un procedimiento almacenado con cada uno de estos IDs para obtener información al respecto. También se puede llamar a un procedimiento almacenado con datos obtenidos a partir de otro procedimiento.

En componentes de destino las relaciones locales permiten definir un orden jerárquico con el que se puede llamar a varios procedimientos relacionados. Por ejemplo, puede llamar primero a un procedimiento almacenado que cree un valor de ID y a otro que inserte información relacionada en una tabla. En las

relaciones locales también se pueden mezclar procedimientos almacenados y tablas. Por ejemplo, puede hacer la inserción directamente en una tabla relacionada en lugar de llamar a otro procedimiento (*véase* *Procedimientos almacenados para generar claves* 417).

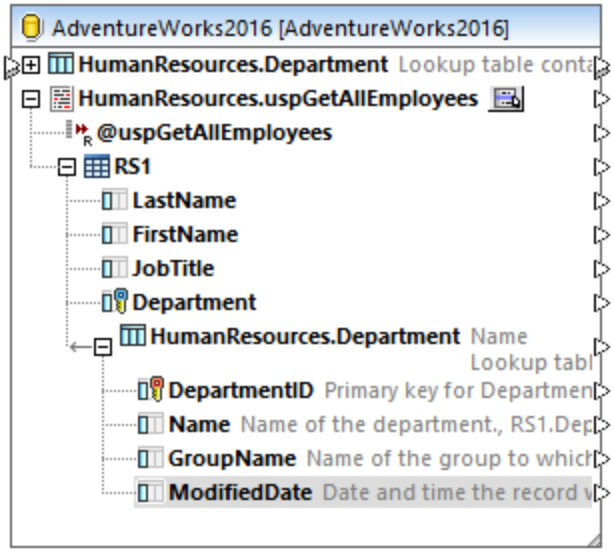
Para crear una relación local:

1. Haga clic con el botón derecho en la barra de título de un componente de BD y seleccione **Componente | Agregar, quitar o editar objetos de la base de datos** del menú contextual. Se abrirá el cuadro de diálogo "Agregar, quitar o editar objetos de la base de datos".
2. Haga clic en **Agregar o editar relaciones**.
3. Haga clic en **Agregar relación** y seleccione los objetos entre los que quiere crear esa relación.



Como ilustra la imagen anterior, una relación local consiste en un objeto de **clave principal/única** y un objeto de **clave foránea**. Piense en ello como una relación padre-hijo. En el componente de la asignación el objeto (tabla, vista, procedimiento, etc.) en el que está la clave principal/única aparecerá como objeto principal, mientras que el objeto en el que está la clave foránea aparecerá anidado debajo. Por ejemplo, en el componente de BD de la imagen siguiente se definió una relación local entre una columna de conjunto de registros (**RS1.Department**) y una columna de tabla (**Department.Name**). En consecuencia, la tabla **Department** aparece como subordinado al procedimiento almacenado de la asignación. Encontrará más detalles sobre este ejemplo en el apartado Relaciones locales en componentes de origen 413.

 © 2016-2022 Altova GmbH



La siguiente tabla enumera todos los campos posibles entre los que puede definir relaciones locales. Es posible crear relaciones mixtas (por ejemplo, asignar los resultados de un procedimiento almacenado a una columna de BD). Los campos que forman parte de la relación deben tener el mismo tipo de datos o uno compatible.

Clave principal/única	Clave foránea
• Columna de una tabla o vista de BD • Parámetro de salida o valor devuelto de un procedimiento almacenado (*consulte también* *Procedimientos almacenados* 395) • Columna de conjunto de registros devuelta por procedimiento almacenado* • Columna de una instrucción SELECT definida por el usuario (*consulte también* *Instrucciones SQL SELECT como tablas virtuales* 359).	• Columna de tabla o vista de BD • Parámetro de entrada de un procedimiento almacenado • Parámetro de entrada de una instrucción SELECT definida por el usuario

** Se aplica si se llama al procedimiento almacenado como fuente de datos (sin parámetros) o como función (con parámetros de entrada y salida). Para que el conjunto de registros esté disponible y se pueda seleccionar debe ejecutar el procedimiento almacenado una vez para obtener ese conjunto de registros.*

4.2.9.6 Relaciones locales en componentes de origen

En este ejemplo explicamos cómo combinar datos devueltos por un procedimiento almacenado con datos de una tabla de la misma base de datos, con ayuda de las relaciones locales.

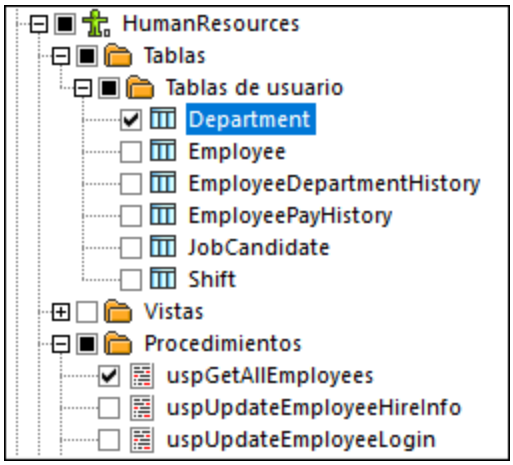
Si no lo hizo ya con el ejemplo anterior, ejecute el script siguiente para crear el procedimiento almacenado de ejemplo en la base de datos "AdventureWorks". Puede hacerlo desde una ventana de consultas de la aplicación de **Microsoft SQL Server Management Studio** o directamente desde la pestaña *Consulta de la*

BD de MapForce (*consulte* *Examinar y consultar bases de datos* 380). En ambos casos, asegúrese de que su cuenta de usuario de la base de datos tiene permisos para crear procedimientos almacenados.

```
CREATE PROCEDURE HumanResources.uspGetAllEmployees
AS
    SELECT LastName, FirstName, JobTitle, Department
    FROM HumanResources.vEmployeeDepartment
```

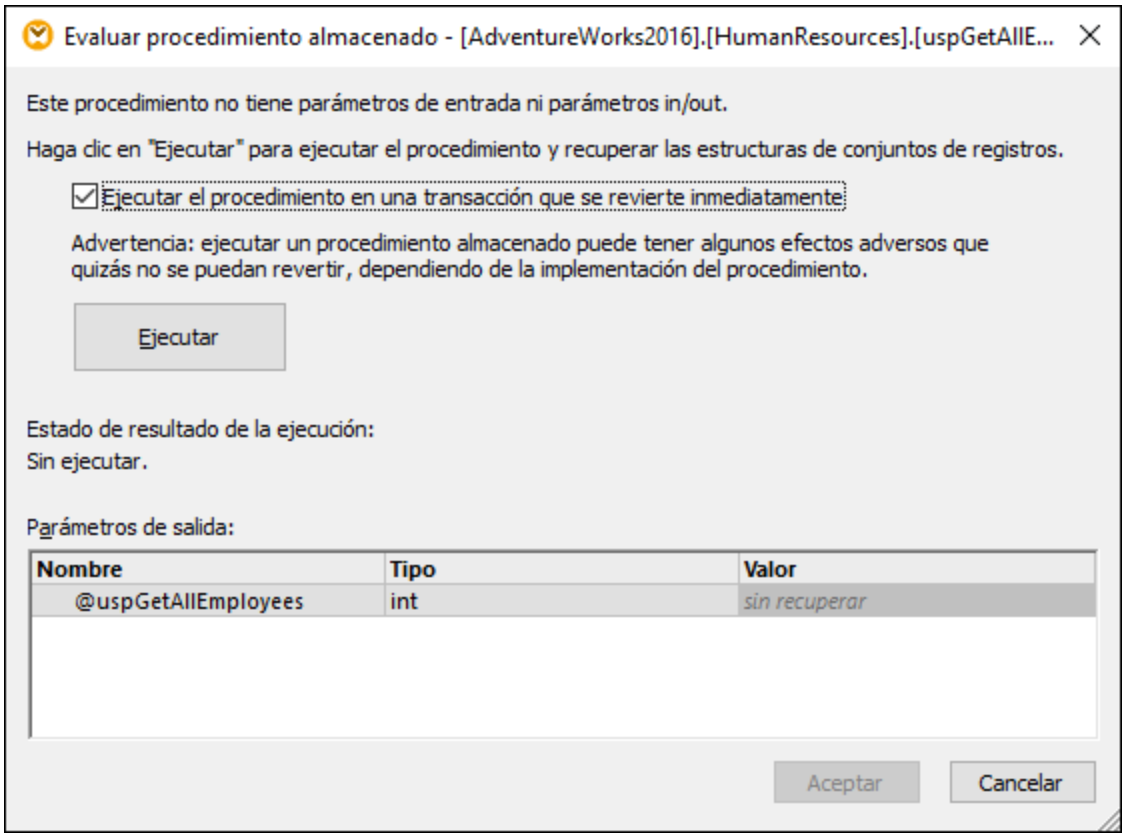
El procedimiento almacenado anterior devuelve información sobre los empleados de la vista **vEmployeeDepartment**. A continuación explicamos los pasos que debe seguir para crear una asignación que consuma los datos devueltos por este procedimiento.

1. Conéctese a la base de datos "AdventureWorks" desde MapForce, como se describe en el apartado Agregar procedimientos almacenados a la asignación 398. Asegúrese de que su cuenta de usuario de la base de datos tiene permisos para crear procedimientos almacenados.
2. Cuando la aplicación solicite que elija los objetos de BD, seleccione la tabla **Department** y el procedimiento almacenado **uspGetAllEmployees**.

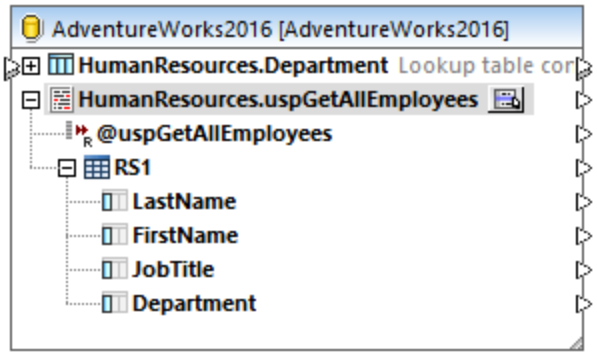


3. Haga clic en el botón para mostrar el menú contextual (), junto al procedimiento almacenado, y seleccione **Mostrar nodos como origen**.
4. Vuelva a hacer clic en el botón para mostrar el menú contextual () y seleccione **Editar estructuras de conjuntos de registros**. Aparece el cuadro de diálogo "Estructuras de conjuntos de registros".
5. Haga clic en **Definir parámetros de entrada y llamar al procedimiento** y después en **Aceptar**. Aparece el cuadro de diálogo "Evaluar procedimiento almacenado".

 © 2016-2022 Altova GmbH



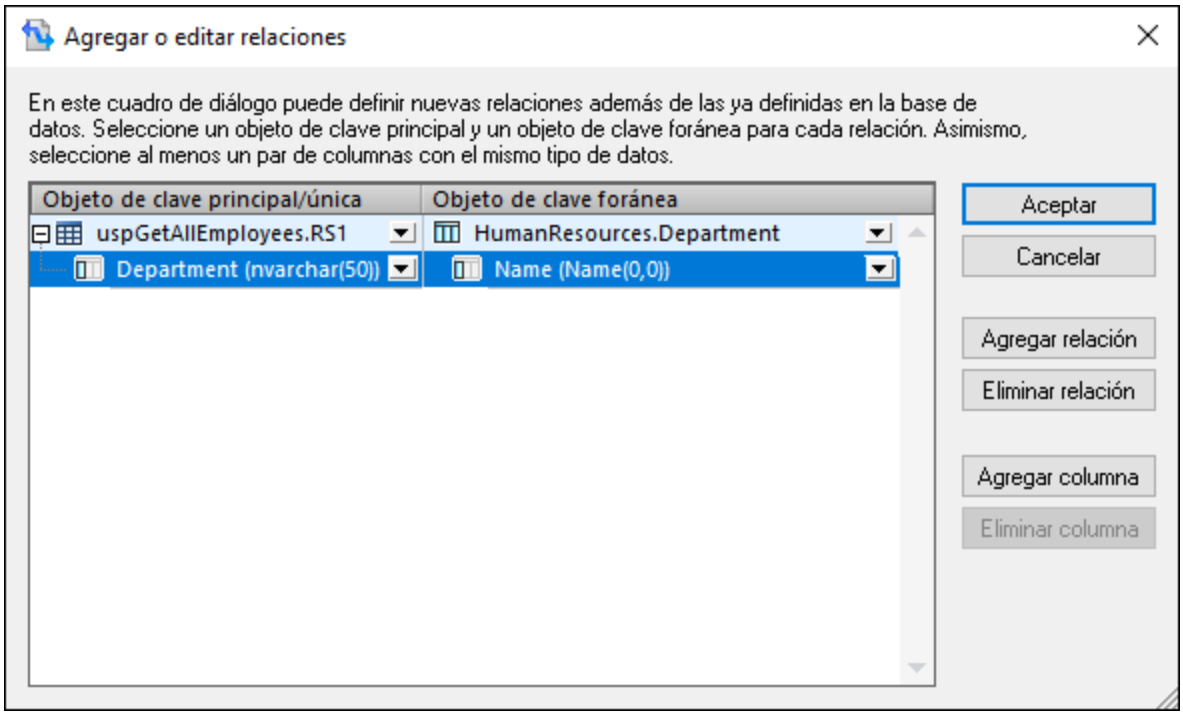
- Haga clic en **Ejecutar** y después en **Aceptar**. Ahora la estructura del conjunto de registros ("RS1") es visible tanto en el cuadro de diálogo "Estructuras de conjuntos de registros" como en la asignación.



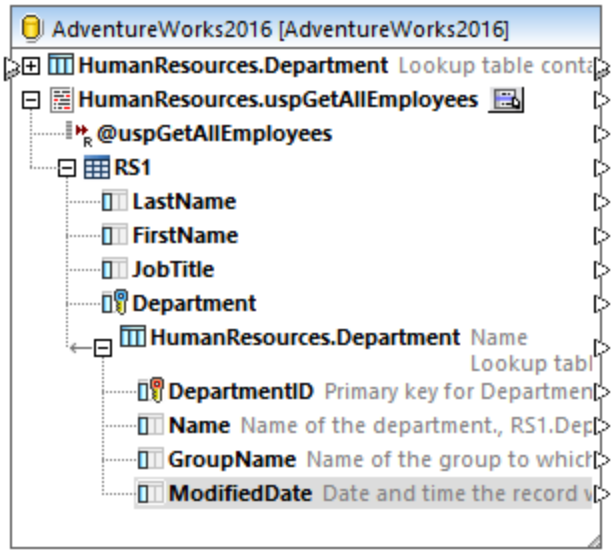
Definir relaciones locales

Ahora vamos a definir una relación local entre la columna **Department** del conjunto de registros devuelto y la columna **Name** de la tabla **Department**.

1. Haga clic con el botón derecho en la barra del título del componente de BD y seleccione **Componente | Agregar, quitar o editar objetos de la base de datos** del menú contextual.
2. Haga clic en **Agregar o editar relaciones** y después en **Agregar relación**. Defina las relaciones como se muestra a continuación.



3. Haga clic en **Aceptar** para cerrar el cuadro de diálogo. Observe que la tabla *Department* ahora se ha convertido en un secundario del conjunto de registros **RS1**.



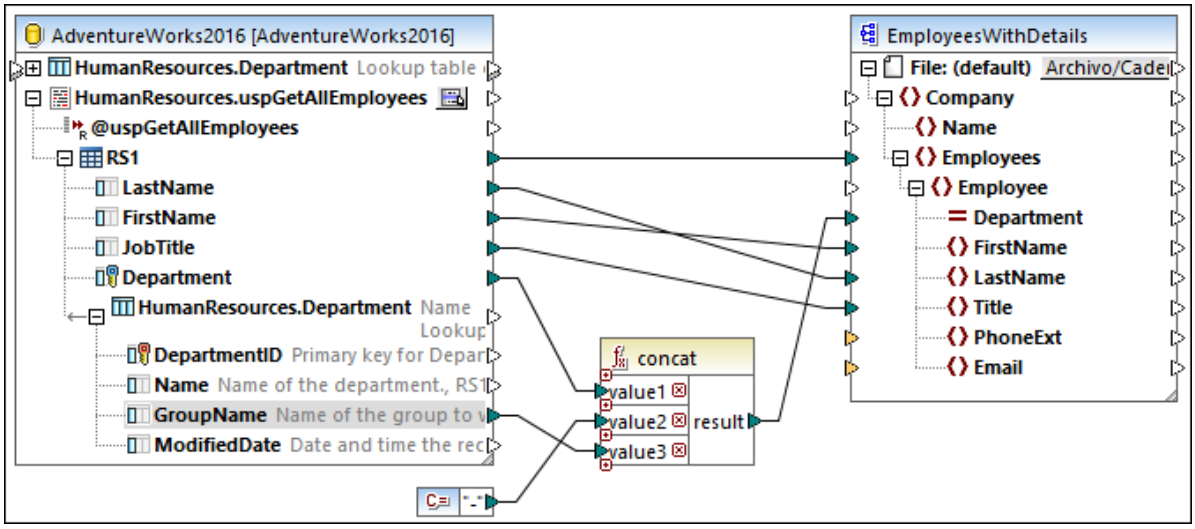
Completar la asignación

Gracias a la relación que acabamos de crear ahora podemos asignar datos del conjunto de registros en combinación con datos de la tabla. Para este ejemplo vamos a escribir datos en un archivo XML de destino:

1. En el menú Insertar, haga clic en **Archivo o esquema XML** y seleccione el siguiente archivo: **<Documentos>\Altova\MapForce2023\MapForceExamples\EmployeesWithDetails.xsd**.

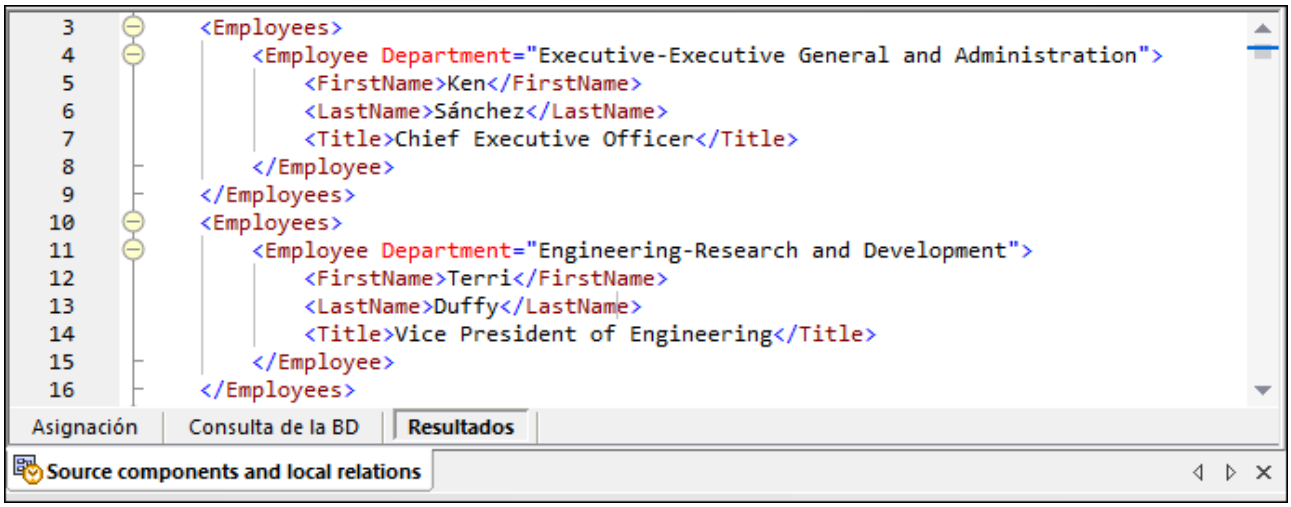
 © 2016-2022 Altova GmbH

2. Cuando la aplicación solicite que indique un archivo de instancia XML de muestra, haga clic en **Omitir**.
3. Dibuje las conexiones de la asignación como se indica a continuación.



La asignación de la imagen anterior escribe datos de la base de datos en un archivo XML de destino. Los datos de origen son una combinación de datos extraídos por el procedimiento almacenado con datos extraídos directamente de una tabla. La asignación usa la función **concat** 708 para generar una cadena de texto que contenga el nombre del departamento seguido por un guion, seguido por el nombre del grupo.

Para obtener la vista previa de la asignación haga clic en el botón **Resultados** y observe el resultado de la asignación en el panel *Resultados*, por ejemplo:



```
3   <Employees>
4       <Employee Department="Executive-Executive General and Administration">
5           <FirstName>Ken</FirstName>
6           <LastName>Sánchez</LastName>
7           <Title>Chief Executive Officer</Title>
8       </Employee>
9   </Employees>
10  <Employees>
11      <Employee Department="Engineering-Research and Development">
12          <FirstName>Terri</FirstName>
13          <LastName>Duffy</LastName>
14          <Title>Vice President of Engineering</Title>
15      </Employee>
16  </Employees>
```

Asignación | Consulta de la BD | Resultados

Source components and local relations

4.2.9.7 Procedimientos almacenados para generar claves

En este ejemplo explicamos cómo insertar una clave (ID) generada por un procedimiento almacenado en otra tabla, con ayuda de las relaciones locales.

Primero vamos a crear el procedimiento almacenado de ejemplo en la base de datos "AdventureWorks". Para

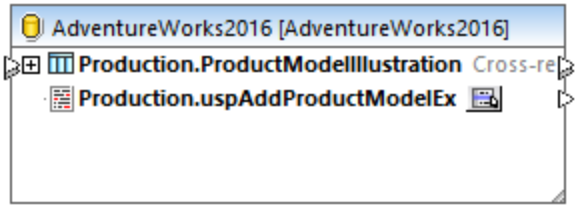
ello, ejecute el siguiente script en la BD. Puede hacerlo desde una ventana de consultas de la aplicación de **Microsoft SQL Server Management Studio** o directamente desde la pestaña *Consulta de la BD* de MapForce (*consulte* *Examinar y consultar bases de datos* 380). En ambos casos, asegúrese de que su cuenta de usuario de la base de datos tiene permisos para crear procedimientos almacenados.

```
CREATE PROCEDURE Production.uspAddProductModelEx
  @ModelName nvarchar(50)
  ,@Inst xml
  ,@ProductModelID int OUTPUT
AS
BEGIN
INSERT INTO [Production].[ProductModel]
           ([Name]
           ,[Instructions]
           ,[rowguid]
           ,[ModifiedDate])
     VALUES
           (@ModelName
           ,@Inst
           ,NEWID()
           ,GETDATE())
   SELECT @ProductModelID = SCOPE_IDENTITY()
END
```

El procedimiento almacenado anterior toma dos parámetros (`INSERT`) como entrada y ejecuta una operación `INSERT` en la tabla **ProductModel**. Entonces devuelve el ID generado `@ProductModelID` como parámetro de salida. El requisito es insertar el `@ProductModelID` devuelto por el procedimiento almacenado en la tabla **ProductModelIllustration**.

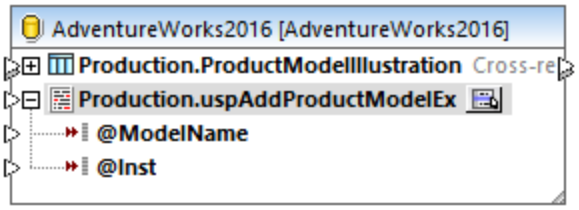
Para crear una asignación que cumpla con el requisito anterior siga los pasos que describimos a continuación.

1. Conéctese a la base de datos "AdventureWorks" desde MapForce, como se describe en el apartado Agregar procedimientos almacenados a la asignación 398. Asegúrese de que su cuenta de usuario de la base de datos tiene permisos para crear procedimientos almacenados.
2. Cuando la aplicación solicite que elija objetos de la BD, seleccione la tabla **ProductModelIllustration** y el procedimiento almacenado **uspAddProductModelEx**.

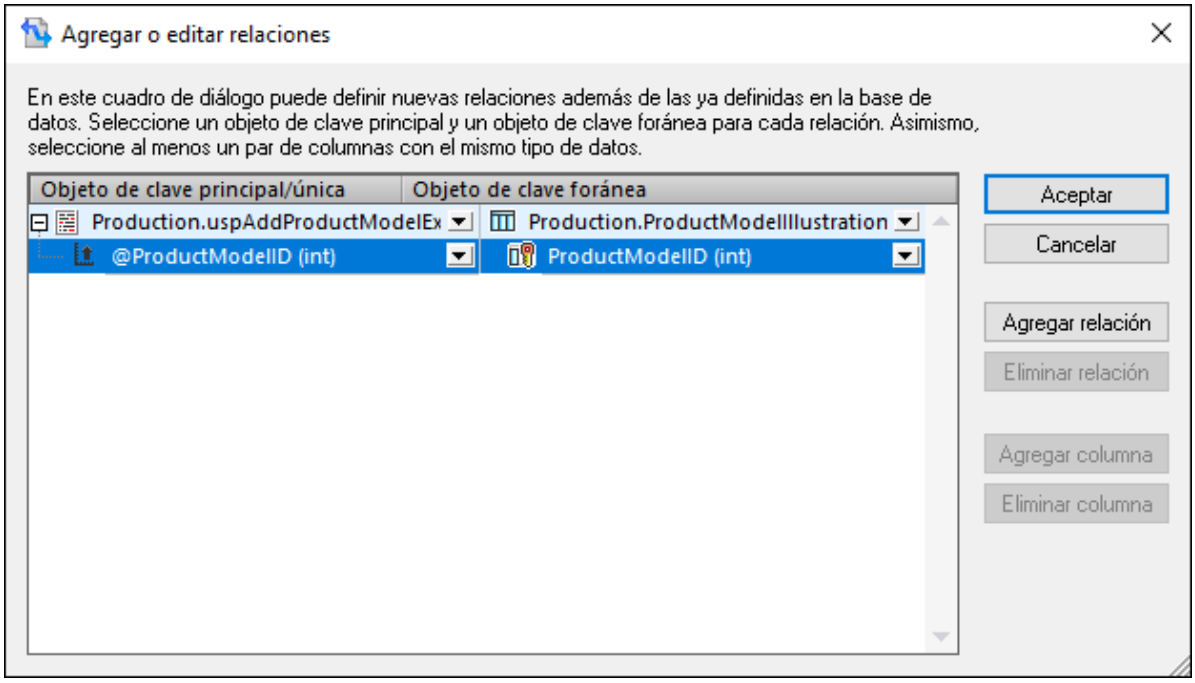


3. Haga clic en el botón para mostrar el menú contextual (), junto al procedimiento almacenado, y seleccione **Mostrar nodos como destino**. Ahora el procedimiento almacenado aparece como componente de destino en la aplicación, donde aparecen a la izquierda los parámetros de entrada.

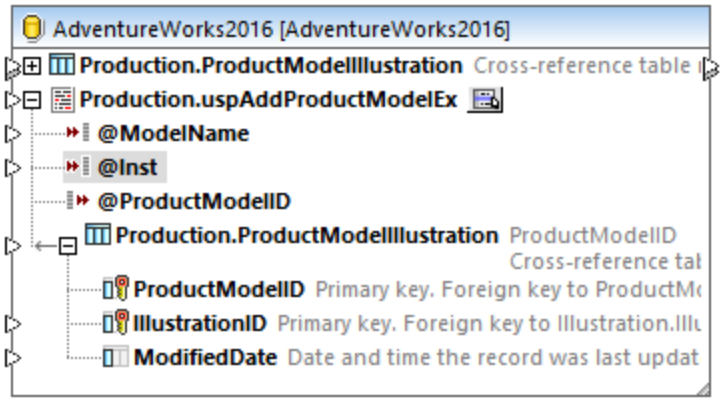
 © 2016-2022 Altova GmbH



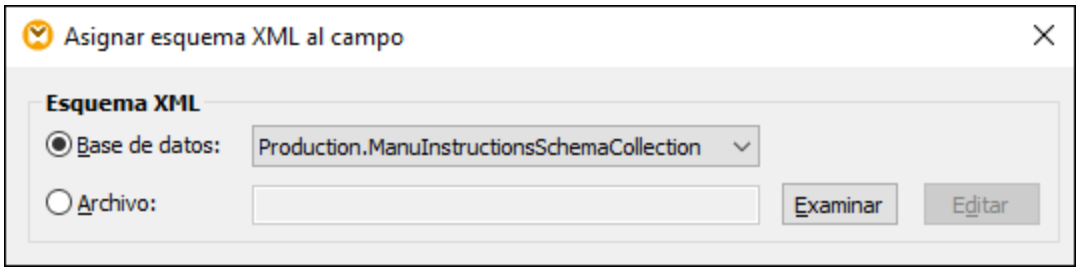
4. Otra opción, si quiere ejecutar el procedimiento almacenado dentro de una transacción, es volver a hacer clic en el botón para mostrar el menú contextual (), seleccionar **Configuración del procedimiento** y luego marcar la casilla **Usar transacciones**. Al definir la transacción para el procedimiento almacenado se asegura de que se obtiene la clave y se inserta el registro durante la misma transacción.
5. Haga clic con el botón derecho en la barra del título del componente de BD y seleccione **Componente | Agregar, quitar o editar objetos de la base de datos** del menú contextual.
6. Haga clic en **Agregar o editar relaciones** y después en **Agregar relación**. Defina las relaciones como se muestra a continuación.



7. Haga clic en **Aceptar** para cerrar el cuadro de diálogo. Observe que la tabla **ProductModelIllustration** ahora aparece como secundaria del procedimiento almacenado. El parámetro de salida del procedimiento almacenado (`@ProductModelID`) se muestra como un indicador que se usará en la relación local, pero no tiene conectores de entrada ni de salida.

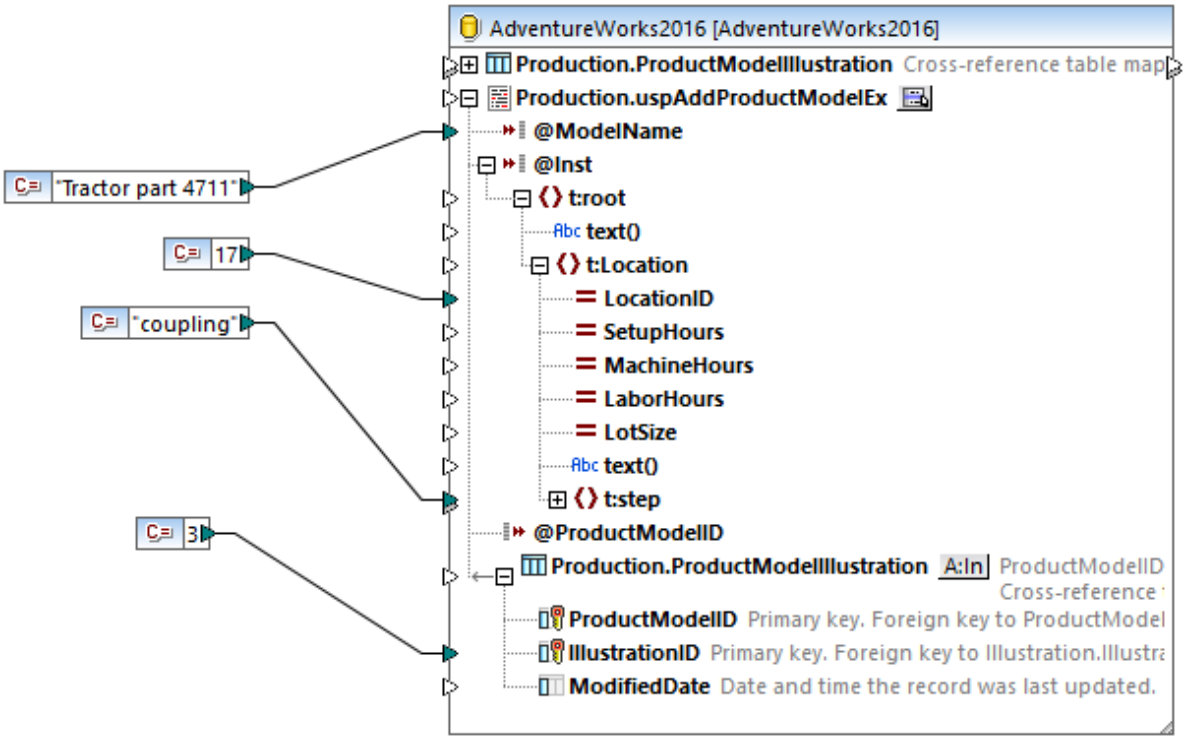


8. En este ejemplo el parámetro `@Inst` es de tipo XML. Haga clic con el botón derecho en el parámetro `@Inst` del componente y seleccione **Asignar esquema XML al campo** en el menú contextual. A continuación, seleccione el esquema **Production.ManuNstructionsSchemaCollection** de la BD. Cuando la aplicación solicite que seleccione un elemento raíz, deje el valor predeterminado como está y haga clic en **Aceptar**. Para más información sobre la asignación de datos a campos XML de BD consulte el apartado Asignaciones entre datos XML y campos de BD 369.



9. Añada el componente de origen que proporciona los datos que se deben insertar en la BD. En este ejemplo los datos se origen vienen dados por constantes; sin embargo, cualquier otro componente de origen compatible con MapForce podría servir como datos de entrada. Para más información sobre constantes consulte el apartado Agregar una constante a una asignación de datos 550.

 © 2016-2022 Altova GmbH



Esta asignación actualiza una base de datos, por lo que no existe una vista previa de sus resultados, como ocurre con otras asignaciones. Al hacer clic en el botón **Resultados**, en la pestaña correspondiente aparecerá una especie de SQL con pistas sobre cómo se modificará la BD. Si habilita las transacciones, estas ocurrirán como describen los comentarios de la pestaña **Resultados**.

```
10    -- begin transaction
11
12    NULL = {{{[Production].[uspAddProductModelEx]}}} ( 'Tractor part 4711','<root
      xmlns="http://schemas.microsoft.com/sqlserver/2004/07/adventure-works/
      ProductModelManuInstructions"><Location LocationID="17"><step>coupling</
      step></Location></root>',NULL )
13    -->>> %@uspAddProductModelEx1%
14    -->>> %@ProductModelID1%
15
16    INSERT INTO [Production].[ProductModelIllustration] ([ProductModelID],
      [IllustrationID]) VALUES ('%@ProductModelID1%', 11)
17
18    -- commit transaction
19
```

Asignación | Consulta de la BD | **Resultados**

uspAddProductModelEx

Esa especie de SQL que aparece en el panel **Resultados** no muestra los comandos reales de las transacciones, sino solo pistas (como comentarios). Los comandos SQL de verdad se envían a la API de BD subyacente.

Puede ejecutar la asignación en la BD de dos maneras:

- En el menú **Resultados**, haga clic en **Ejecutar script SQL**.
- Haga clic en el botón de la barra de herramientas **Ejecutar script SQL** .

4.2.10 Gestionar excepciones de BD

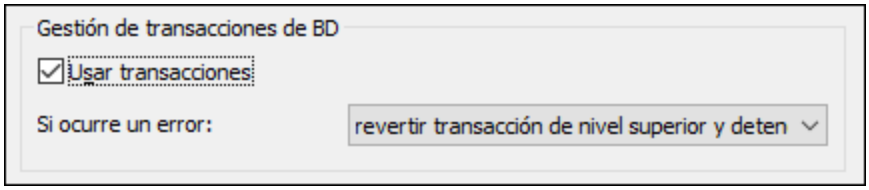
Al ejecutar asignaciones que modifican bases de datos, es posible que MapForce (o MapForce Server o un programa generado por MapForce) encuentre errores relacionados con la BD. Por ejemplo, puede la cuenta de la base de datos no tenga los privilegios suficientes para ejecutar una acción específica de base de datos o puede que existan restricciones de BD que impidan que la asignación inserte o actualice datos no válidos. Esto último puede ocurrir, por ejemplo, si una columna de tabla es obligatoria pero la asignación no proporciona un valor para ella.

Para que sea posible revertir datos si aparecen excepciones de base de datos puede configurar la asignación para que use transacciones de base de datos. Como explicamos a continuación, puede habilitar las transacciones a nivel de la base de datos y a nivel de acciones de tabla.

Transacciones a nivel de componente de BD

Para habilitar transacciones a nivel de componente de BD haga lo siguiente:

1. Haga clic con el botón derecho en el componente de la base de datos y seleccione **Propiedades** en el menú contextual (También puede hacer doble clic en la barra de título del componente de BD). Esto abre el cuadro de diálogo "Configuración del componente" correspondiente.



2. Marque la casilla **Usar transacciones**.

Esta acción agrupa todos los cambios realizados por el componente de base de datos en una transacción que se puede revertir en caso de error. Cuando ocurre un error al ejecutar una asignación, el resultado depende de la opción que elija en la lista desplegable:

- **revertir transacción de nivel superior y detener:** deshace la transacción que contiene todos los cambios hechos por el componente de BD y detiene la ejecución de la asignación.
- **revertir transacción de nivel superior y continuar:** hace lo mismo que la opción anterior pero permite que la ejecución de la asignación continúe después de revertir la transacción (por ejemplo, para procesar un segundo componente de destino, si lo hay).

Algunas asignaciones pueden contener varios componentes de base de datos que, a su vez, pueden usar todos la misma conexión de base de datos o conexiones completamente distintas. El resultado de esas asignaciones en caso de error depende del motor de ejecución:

- Si la asignación se ejecuta con MapForce, solo se puede procesar un componente de destino mientras se está ejecutando la asignación. Este es el componente para el que está habilitado el botón

 © 2016-2022 Altova GmbH

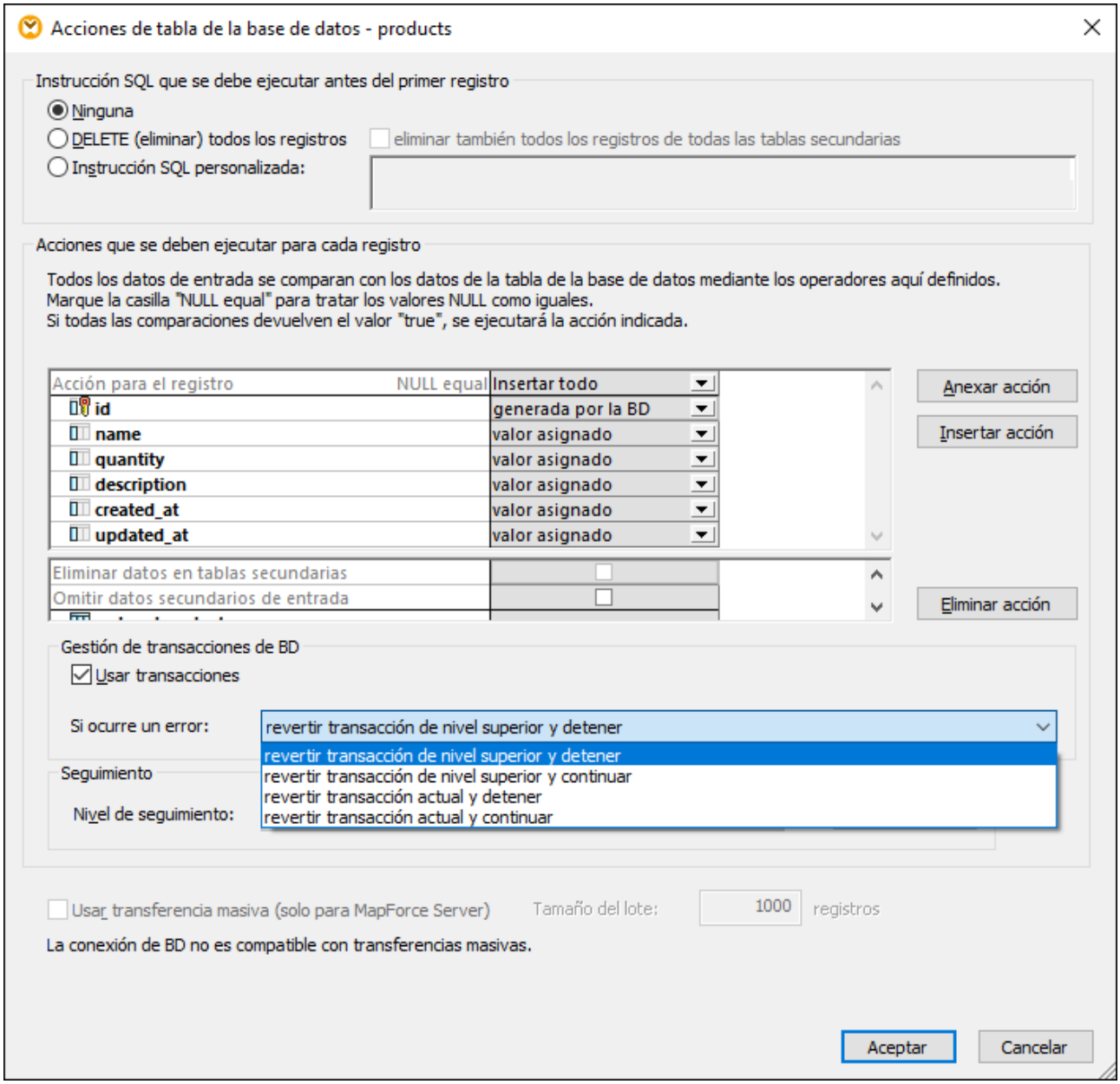
la Vista previa . Si ocurre un error de BD en ese componente y la casilla *Usar transacciones* está marcada, entonces se revertirán todos los cambios realizados por ese componente.

- Si la asignación se ejecuta con MapForce Server o con un programa generado por MapForce, se procesan todos los componentes de destino de forma secuencial. En este caso, si ocurre un error de BD la acción de revertir el componente de BD tendrá lugar donde ocurrió el error. La asignación se detendrá o pasará a ejecutar el siguiente componente de destino, según el valor que haya seleccionado en la lista desplegable de la opción **Si ocurre un error**.

Para consultar todas las opciones configurables del cuadro de diálogo "Configuración del componente" de BD (*véase* *Configurar componentes de BD* 278).

Transacciones a nivel de acciones de tabla

Las transacciones a nivel de acciones de tabla incluyen todas las acciones de tabla (insertar, actualizar, etc.) que hay dentro de una transacción. Con esta configuración, si alguna acción de insertar o actualizar falla en un registro, se puede revertir únicamente la acción que ha fallado y seguir procesando el resto de registros. Para habilitar las transacciones a nivel de acciones de tabla marque la casilla **Usar transacciones** en el cuadro de diálogo "Acciones de tabla de la base de datos" (*véase también* *Configurar de acciones de tabla de BD* 327).



Estas son las opciones de este cuadro de diálogo:

- **Revertir transacción de nivel superior y detener:** si la asignación va a actualizar varias tablas entre las que existen relaciones primaria-secundaria, se pueden habilitar las transacciones no solo para la tabla primaria, sino también para la tabla secundaria. Las transacciones se pueden habilitar a distintos niveles, por lo que "primario" hace referencia a la transacción al nivel más alto en la jerarquía. Esta es la transacción a nivel de componente de BD, si habilitó las transacciones a ese nivel. De lo contrario es la transacción definida en la acción de tabla de nivel superior, siempre y cuando el componente contenga varias acciones de tabla anidadas. Para entender esta opción, recuerde que un componente puede actualizar varias tablas de forma jerárquica, como se describe en Insertar datos en tablas vinculadas 286. Por ejemplo, un componente puede tener como elemento primario una acción

"Insertar" que inserta registros de personas y como elemento secundario otra acción "Insertar" que inserta una o más direcciones para cada persona en una tabla distinta. Como las transacciones se pueden habilitar para las dos acciones "insertar", "de nivel superior" se refiere al nivel más alto de la jerarquía (en este ejemplo la persona puede insertar la acción). Por tanto, si ocurre una excepción al insertar una dirección se revierte la acción de nivel superior (que en este ejemplo está a nivel de tabla de persona) y se detiene la ejecución. En otras palabras, si no se puede insertar la dirección no se puede insertar ningún registro de persona.

- **Revertir transacción de nivel superior y continuar:** ocurre como en la opción anterior, pero la asignación sigue ejecutándose después de revertir la transacción (por ejemplo, para procesar otro componente de destino, si lo hay).
- **Revertir transacción actual y detener:** si ocurre una transacción, se revierten únicamente los cambios contenidos en la transacción actual; los cambios realizados previamente fuera de la transacción actual se conservan.
- **Revertir transacción actual y continuar:** ocurre como en la opción anterior, pero la asignación continúa después de deshacer la transacción.

Transacciones almacenadas a nivel de procedimiento

También puede habilitar las transacciones de BD para los procedimientos almacenados a los que se llama como destino de los datos. Para ver un ejemplo, consulte Procedimientos almacenados en componentes de destino 408. Las transacciones no son compatibles con los procedimientos almacenados a los que se llama como fuente de datos (sin parámetros de entrada) o con aquellos a los que se llama como función (con parámetros de entrada y de salida).

Reversión

Para ver si las transacciones se van a llevar a cabo al ejecutar la asignación, haga clic en la pestaña *Resultados* y observe el pseudo código SQL que genera MapForce. La siguiente imagen muestra la vista previa de una asignación en la que se han habilitado las transacciones a nivel de componente y a nivel de acción de tabla. Como se ve, una transacción principal engloba toda la operación (entre los comentarios `--begin transaction` y `--commit transaction`). También existen otras transacciones menores que ocurren por cada operación "insertar" (identificadas por el comando `SAVEPOINT`).

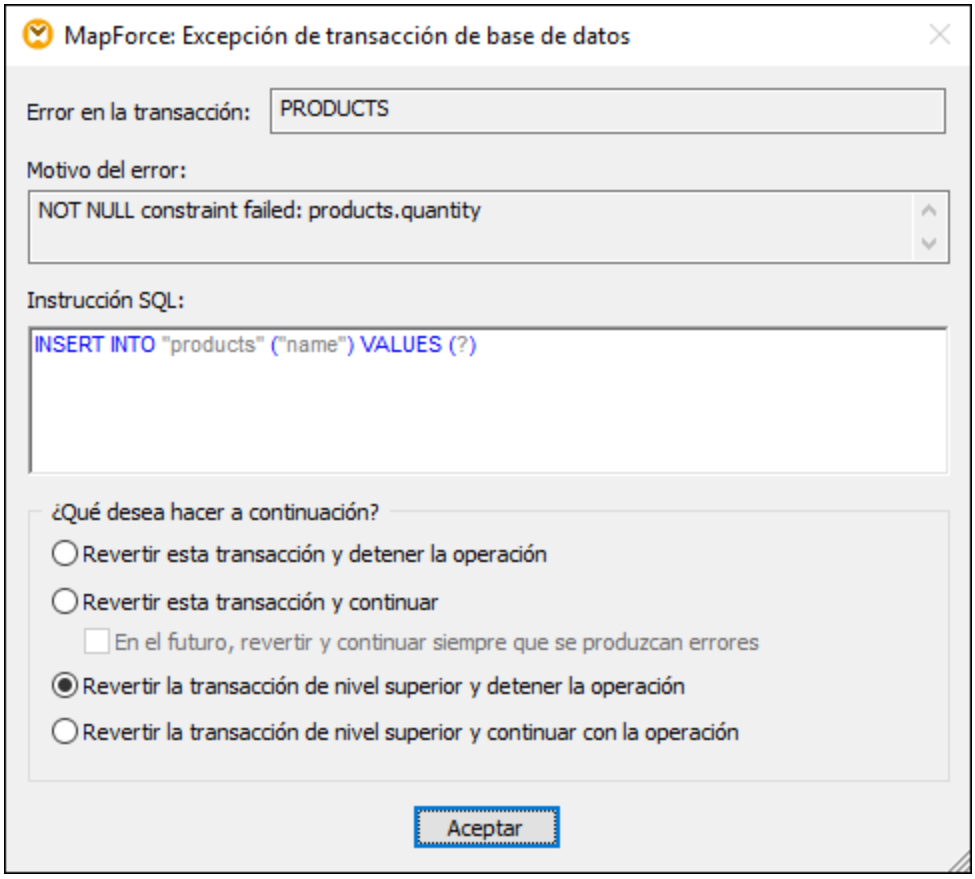
```
7
8   -- begin transaction
9
10  PRAGMA foreign_keys = ON;
11
12  SAVEPOINT PRODUCTS
13
14  INSERT INTO "products" ("name") VALUES ('Product A')
15
16  SAVEPOINT PRODUCTS
17
18  INSERT INTO "products" ("name") VALUES ('Product B')
19
20  SAVEPOINT PRODUCTS
21
22  INSERT INTO "products" ("name") VALUES ('Product C')
23
24  -- commit transaction
25
```

Asignación | Consulta de la BD | **Resultados**

Cuando la asignación se ejecuta en MapForce Server o desde un programa generado por MapForce, la reversión tiene lugar en función de las opciones que haya configurado al diseñar la asignación. Por ejemplo, si selecciona la opción **Deshacer transacción de nivel superior y continuar**, la inserción que haya fallado se revertirá y volverá al punto de recuperación más cercano, y la asignación seguirá intentando insertar registros.

Si ejecuta una asignación en la que se han habilitado las transacciones directamente en MapForce con el comando de menú **Resultados | Ejecutar script SQL** un cuadro de diálogo le informará cuando encuentre una excepción e BD, por ejemplo:

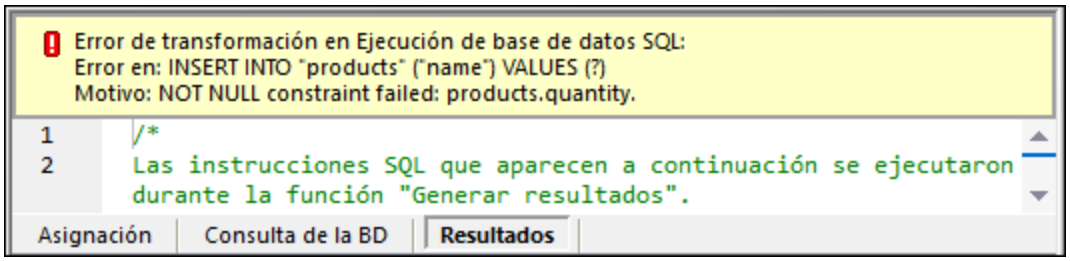
 © 2016-2022 Altova GmbH



Hay varias opciones para cambiar el comportamiento posterior:

- **Revertir esta transacción y detener la operación**

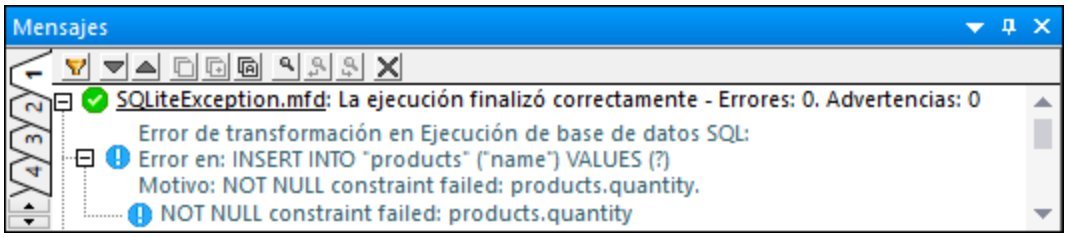
  - La transacción actual ya se ha revertido cuando aparece el cuadro de diálogo
  - Aparece un error en la parte superior de la ventana de vista previa y en la ventana *Mensajes*



  - Las transacciones de nivel superior sí se envían a la base de datos
  - La ejecución de la asignación se detiene con un código de error

Revertir esta transacción y continuar

- La transacción actual ya se ha revertido cuando aparece el cuadro de diálogo
- El error es tratado como una advertencia y aparece en la ventana *Mensajes* cuando termina de ejecutarse la asignación



- Las transacciones de nivel superior no se ven afectadas
- La asignación sigue ejecutándose con la siguiente entrada del registro (si la transacción no es a nivel de componente) o con el siguiente componente (solo en el caso de MapForce Server)

Revertir transacción de nivel superior y detener la operación

- La transacción actual ya se ha revertido cuando aparece el cuadro de diálogo
- Aparece un error en la parte superior de la ventana de vista previa y en la ventana *Mensajes*
- Las transacciones de nivel superior se revierten
- La ejecución de la asignación se detiene con un código de error

Nota: la opción **Revertir transacción de nivel superior y detener la operación** aparece en el cuadro de diálogo solo si hay transacciones anidadas (puntos de recuperación o *savepoints*).

Revertir transacción de nivel superior y continuar

- La transacción actual ya se ha revertido cuando aparece el cuadro de diálogo
- El error es tratado como una advertencia y aparece en la ventana *Mensajes* cuando termina de ejecutarse la asignación
- Las transacciones de nivel superior se revierten
- La asignación sigue ejecutándose con la siguiente entrada del registro (si la transacción no es a nivel de componente) o con el siguiente componente (solo en el caso de MapForce Server)

Nota: la opción **Revertir transacción de nivel superior y continuar** aparece en el cuadro de diálogo solo si hay transacciones anidadas (puntos de recuperación o *savepoints*).

4.2.10.1 Ejemplo: revertir transacciones

En este ejemplo aprenderá las distintas formas de gestionar errores de BD al ejecutar una asignación que modifica una base de datos. La BD que se usa en este ejemplo almacena registros de usuario (tabla `users`) y direcciones (tabla `addresses`). Cada usuario puede tener una dirección, dos o ninguna (por ejemplo, una dirección personal y una de trabajo). Cada una de las direcciones de la tabla contiene un campo `user_id` que apunta al campo `id` de la tabla `users`. Por esta razón, la tabla `users` es una tabla primaria para la tabla `addresses`. La lista siguiente muestra los scripts SQL de creación para ambas tablas (observe que la sintaxis se puede aplicar a una base de datos SQLite).

```
CREATE TABLE
       users (id INTEGER NOT NULL PRIMARY KEY,
       first_name TEXT NOT NULL,
       last_name TEXT NOT NULL,
       email TEXT UNIQUE NOT NULL);
```

```
CREATE TABLE
       addresses (id INTEGER NOT NULL PRIMARY KEY,
       user_id INTEGER NOT NULL,
       is_shipping INTEGER,
       is_billing INTEGER,
       type TEXT NOT NULL,
       city TEXT NOT NULL,
       street TEXT NOT NULL,
       number INTEGER NOT NULL,
       FOREIGN KEY (user_id) REFERENCES users (id) ) ;
```

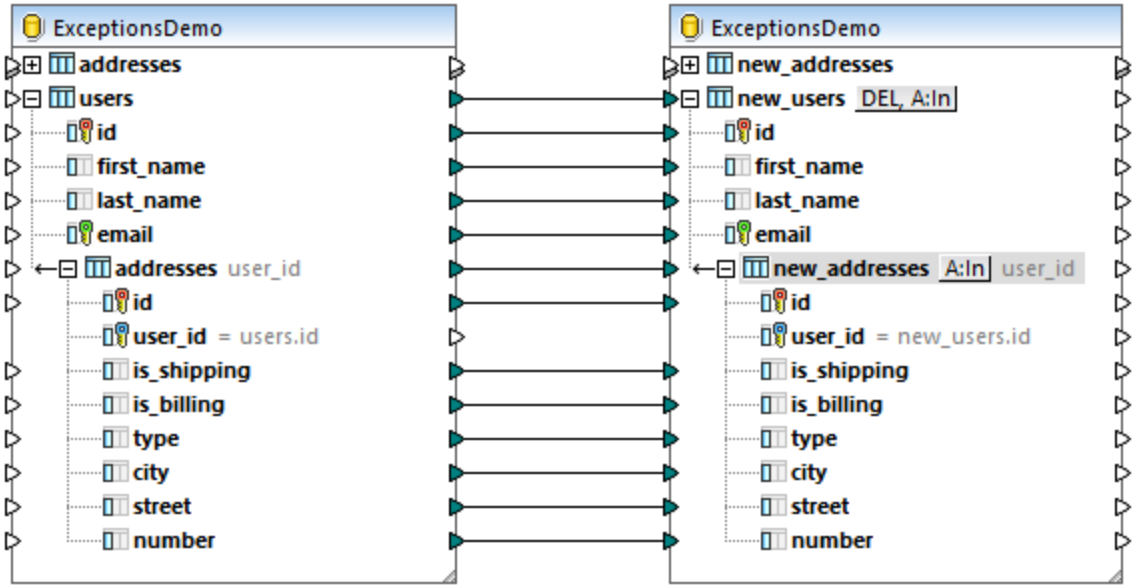
Tablas de origen

El objetivo de este ejemplo es copiar todos los datos de las tablas `users` y `addresses` a otras tablas nuevas: `new_users` y `new_addresses`. Estas tablas nuevas son prácticamente idénticas a las anteriores; la única diferencia es que la tabla `new_tables` no admite valores null para las columnas `is_shipping` o `is_billing`, como se ve resaltado en azul en el código de ejemplo siguiente. Esto significa que pueden ocurrir excepciones si aparecen valores null.

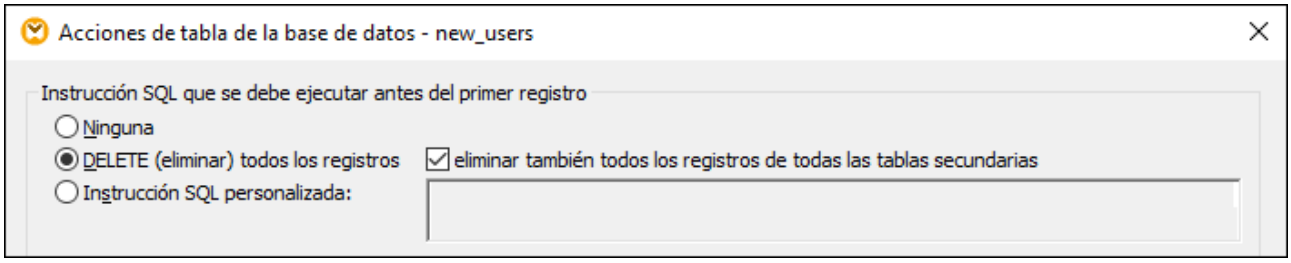
```
CREATE TABLE
       new_users (id INTEGER NOT NULL PRIMARY KEY,
       first_name TEXT NOT NULL,
       last_name TEXT NOT NULL,
       email TEXT UNIQUE NOT NULL);

CREATE TABLE
       new_addresses (id INTEGER NOT NULL PRIMARY KEY,
       user_id INTEGER NOT NULL,
       is_shipping INTEGER NOT NULL,
       is_billing INTEGER NOT NULL,
       type TEXT NOT NULL,
       city TEXT NOT NULL,
       street TEXT NOT NULL,
       number INTEGER NOT NULL,
       FOREIGN KEY (user_id) REFERENCES new_users (id) ) ;
```

Tablas de destino

A continuación mostramos la asignación que copia todos los datos de las tablas antiguas a las nuevas. Puede encontrar esta asignación en **<Documentos>\Altova\MapForce2023\MapForceExamples\Tutorial\DatabaseExceptions.mfd**.



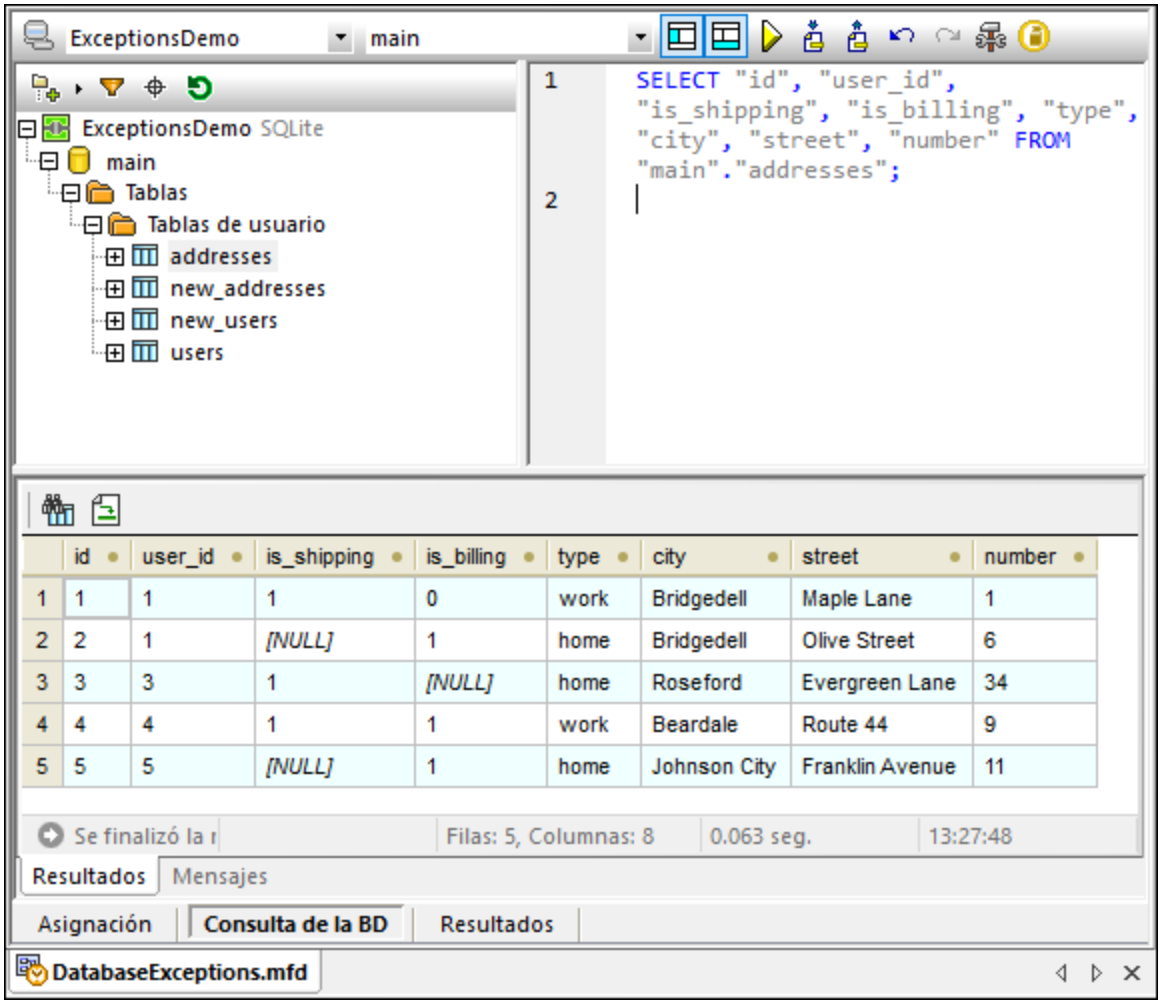
Como se ilustra en la imagen anterior, esta asignación está configurada para copiar datos de forma literal de un componente de BD de origen a uno de destino. Hemos omitido cualquier otra transformación para simplificar. En este ejemplo tanto el componente de origen como el de destino pertenecen a la misma base de datos y solo las tablas de origen y de destino son distintas. Haga clic en el botón DEL, A:In del componente de destino y observe que todas las entradas de registro se borran antes de cada acción "insertar". Esto garantiza que las tablas de destino estén siempre vacías antes de cada inserción.



Como hemos explicado antes, las excepciones ocurren si la tabla de origen `addresses` contiene valores null. Para comprobar si la tabla de origen contiene valores null:

1. Haga clic en la pestaña **Consulta de la DB**.
2. Haga clic con el botón derecho en la tabla `addresses` y seleccione **Mostrar en Editor SQL | SELECT**.
3. Haga clic en **Ejecutar consulta** .

 © 2016-2022 Altova GmbH



Como se ve más arriba, existen varios campos `NULL` en la tabla de origen `addresses`. Sabiendo esto, al ejecutar esta asignación tiene varias opciones para gestionar las excepciones, por ejemplo:

- A) Si aparece una excepción, se revierten todos los cambios. En otras palabras, no se inserta ninguna entrada de registro si los datos de origen contienen entradas de registro no válidas.
- B) si aparece una excepción, se omiten las entradas de registro en las que se da esa excepción pero se siguen insertando entradas de registro válidas.

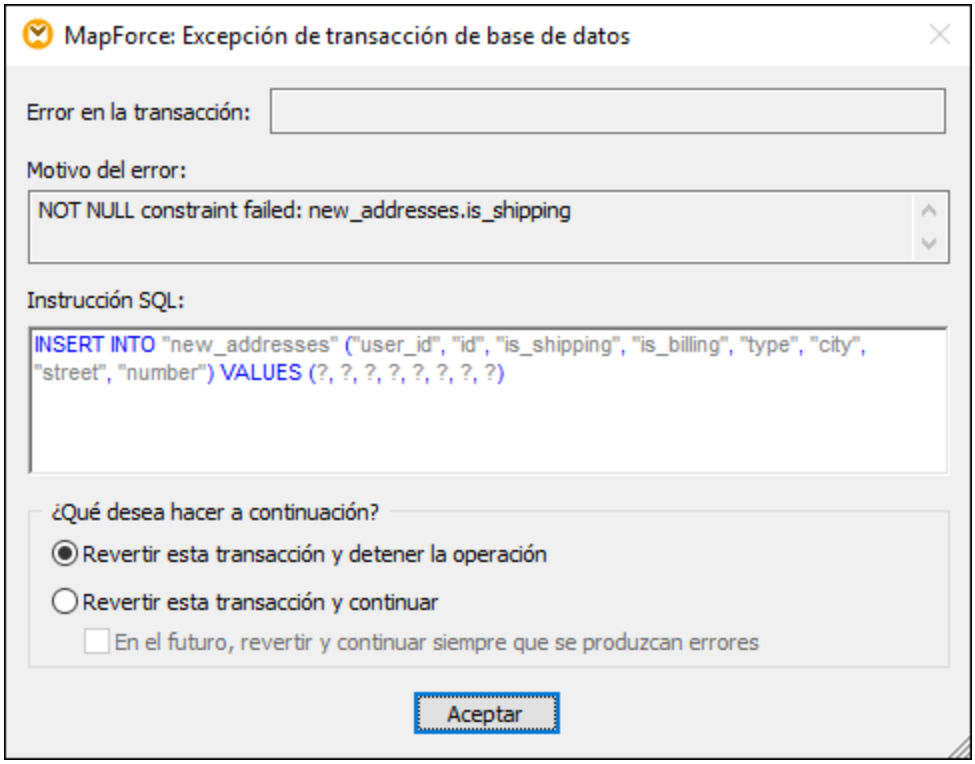
Caso A: revertir todos los cambios si se encuentra una excepción

El objetivo en este caso es revertir todos los cambios realizados a la base de datos si se encuentra una excepción. Para configurar la asignación para que eso ocurra:

1. Haga clic con el botón derecho en la barra de título del componente de BD de destino y seleccione **Propiedades** en el menú contextual. También puede hacer doble clic en la barra de título para abrir el cuadro de diálogo "Configuración del componente".
2. Marque la casilla *Usar transacciones* y seleccione **revertir transacción de nivel superior y detener**.

Cuando la asignación está configurada como en el ejemplo anterior, cualquier excepción que ocurra revertirá la transacción de nivel superior al nivel de los componentes de BD. Para ejecutar la asignación con esta configuración:

1. Haga clic en la pestaña *Resultados*.
2. En el menú Resultados, haga clic en **Ejecutar script SQL**. Ahora la asignación encuentra una excepción porque hay un valor null en la tabla de origen `addresses`. Aparece el cuadro de diálogo "Excepción de transacción de base de datos".



3. Deje la configuración como está y haga clic en **Aceptar**.

Este es el resultado:

- se revierten todos los cambios.
- no se inserta ninguna entrada de registro en la tabla `new_users`
- no se inserta ninguna entrada de registro en la tabla `new_addresses`

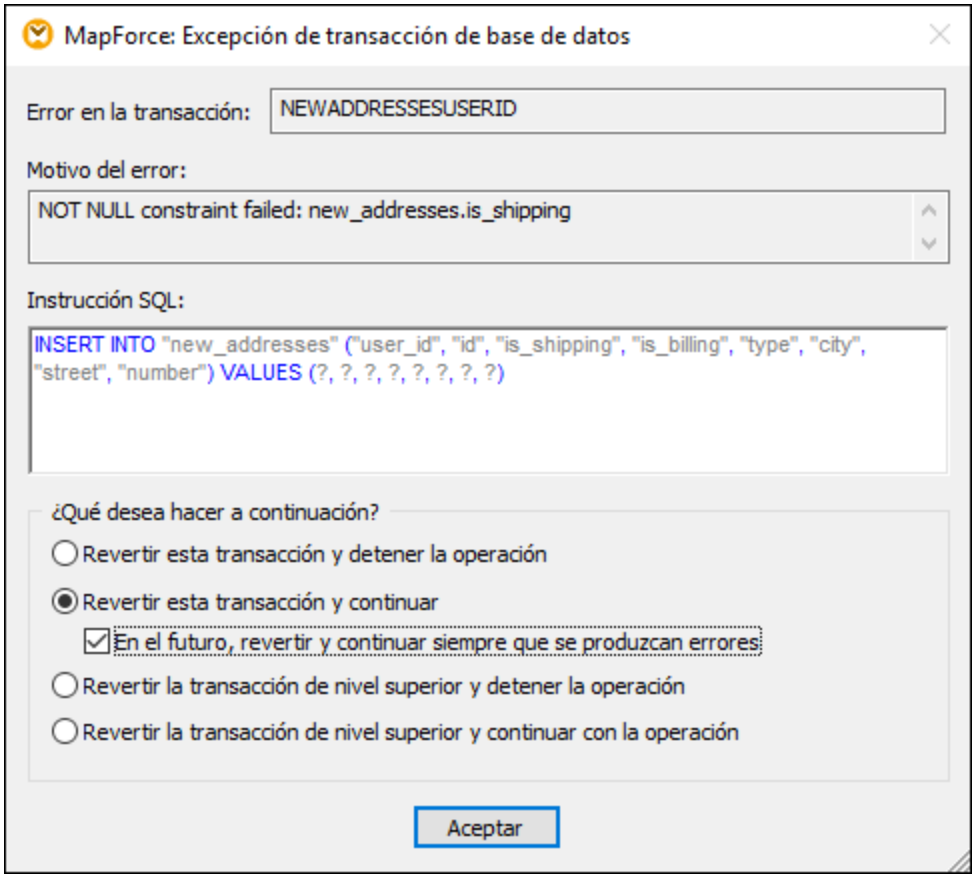
Caso B: revertir la transacción actual y continuar con la ejecución

El objetivo en este caso es omitir las entradas de registro que generen una excepción pero seguir insertando entradas de registro válidas. Para configurar la asignación para que eso ocurra:

1. Haga clic en el botón DEL, A:In junto a la tabla `new_users`, marque la casilla *Usar transacciones* y seleccione **revertir transacción actual y continuar**
2. Haga clic en el botón A:In junto a la tabla `new_addresses`, marque la casilla *Usar transacciones* y seleccione **revertir transacción actual y continuar**

 © 2016-2022 Altova GmbH

Cuando se ejecuta una asignación con esta configuración aparece el cuadro de diálogo "Excepción de transacción de base de datos". Seleccione la opción **Revertir esta transacción y continuar**, como se muestra en la imagen siguiente, para revertir únicamente esa transacción y seguir ejecutando la asignación.



Este es el resultado:

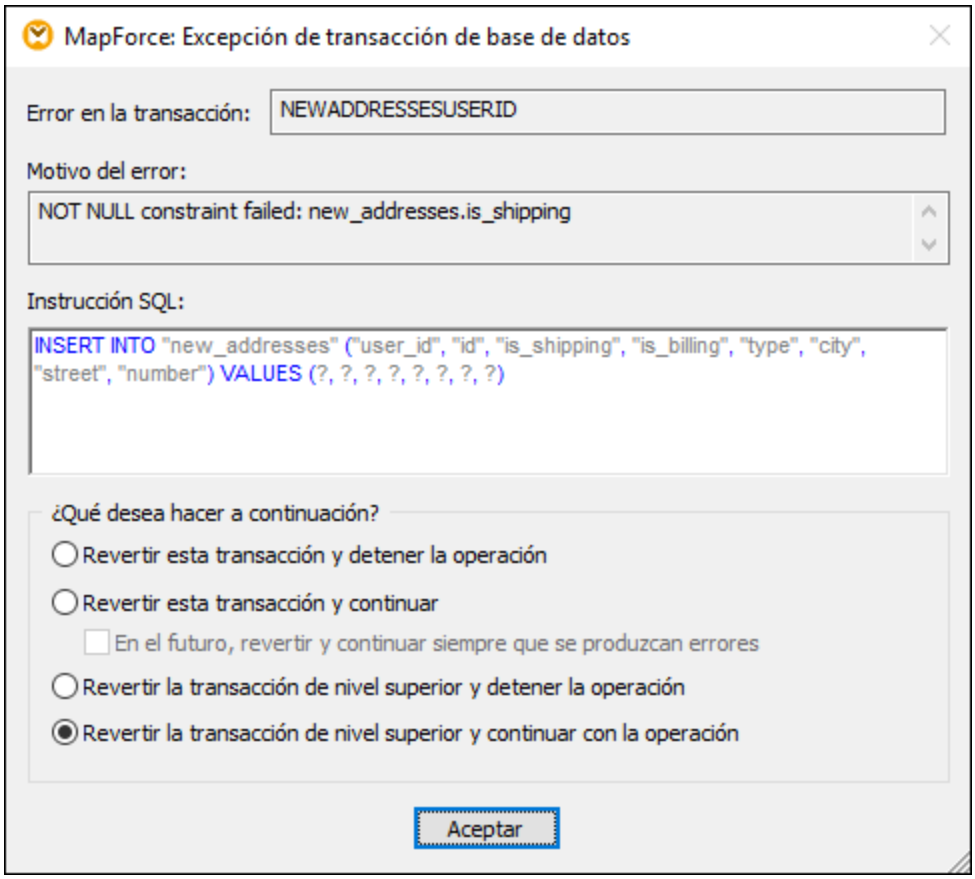
- se revierten todas las transacciones erróneas.
- se insertan 5 usuarios de 5 (porque ninguna de esas entradas de registro generó una excepción)
- se insertan 2 de 5 direcciones (porque 3 de ellas contenían valores null y generaron excepciones)

Caso C) Revertir transacción de nivel superior y continuar con la ejecución

Objetivo: si falta por insertar una dirección tampoco se inserta la entrada de registro del usuario primario. Para configurar la asignación con esta finalidad:

1. Haga doble clic en la barra del título del componente de BD y desmarque la casilla *Usar transacciones*.
2. Haga clic en el botón DEL, A:In junto a la tabla `new_users`, marque la casilla *Usar transacciones* y seleccione **revertir transacción actual y continuar**
3. Haga clic en el botón A:In junto a la tabla `new_addresses`, marque la casilla *Usar transacciones* y seleccione **revertir transacción de nivel superior y continuar**

Al ejecutar la asignación con esta configuración aparece el cuadro de diálogo "Excepción de transacción de base de datos". Configure las opciones como en la imagen siguiente y repita su selección tantas veces como se lo pida la aplicación:



Este es el resultado:

- se insertan 2 usuarios (Sharda Junker, Tobie Hughey)
- se inserta 1 dirección

Explicación: la usuaria "Sharda Junker" es la única que no tiene una dirección. Como no ocurrieron excepciones en este caso, la entrada de registro se insertó. El segundo usuario, "Tobie Hughey", se insertó porque es el único usuario sin excepciones a nivel de la dirección. El resto de entradas de registro no se insertaron porque había al menos una dirección en la que ocurrió una excepción, por lo que las transacciones se revirtieron.

Hay dos direcciones que no tienen valores null, por lo que no generan excepciones. Se trata de las direcciones con `id=1` e `id=4`. La transacción de inserción para la primera dirección se revirtió porque la transacción del usuario primario también debía revertirse. Por lo tanto, la única dirección que llegó a insertarse fue `id=4`.

Observe que se puede obtener el mismo resultado de esta otra manera:

1. Haga clic en el botón DEL, A:In junto a la tabla `new_users`, marque la casilla *Usar transacciones* y elija **revertir transacción actual y continuar**
2. Haga clic en el botón A:In junto a la tabla `new_addresses` y desmarque la casilla *Usar transacciones*.

 © 2016-2022 Altova GmbH

Ejecutar la asignación con MapForce Server

Si tiene una licencia para MapForce Server también puede ejecutar la asignación desde la línea de comandos tanto en Linux como en macOS y en Windows:

1. Compile la asignación en un archivo de ejecución de MapForce Server (.mfx) con el comando de menú **Archivo | Compilar en archivo de ejecución de MapForce Server** (véase también Compilar asignaciones en archivos de ejecución de MapForce Server 955).
2. Copie el archivo .mfx y la base de datos **ExceptionsDemo.sqlite** en un directorio en el equipo servidor al que llamaremos "directorio de trabajo".
3. Cambie el símbolo del sistema al directorio de trabajo y ejecute MapForce Server con el siguiente comando:

```
mapforceserver run DatabaseExceptions.mfx
```

Notas:

- **mapforceserver** es la ruta de acceso al ejecutable de MapForce Server con las modificaciones que requiera su sistema operativo. Cambie la ruta de acceso al archivo .mfx como convenga. Por ejemplo, en un equipo Linux la ruta sería: `/opt/Altova/MapForceServer2023/mapforceserver`.

Al ejecutar la asignación en el servidor también puede ejecutar asignaciones como una llamada API o como un trabajo de FlowForce, sea puntualmente o de forma recurrente. Para más información consulte el apartado Automatización con MapForce Server 948.

4.2.11 Seguimiento de BD y registro de errores

Cuando una asignación escribe datos en una BD se puede habilitar el seguimiento de BD y el registro de errores. El seguimiento es útil para monitorizar todos los cambios que la asignación hace en la base de datos al ejecutarse. Si se activa el seguimiento, acciones como las inserciones en la BD, las actualizaciones o los errores quedan registradas en un archivo XML que más tarde puede analizar o incluso seguir procesando de forma automatizada. Si prefiere que el archivo de registro esté en un formato distinto a XML, también puede asignar los datos que contiene a otro tipo de componente compatible con MapForce (por ejemplo, un archivo de texto o incluso otra base de datos).

Nota: para que una asignación admita el seguimiento y el registro de errores, el lenguaje de la asignación debe ser BUILT-IN. El registro de BD y el seguimiento no se admiten en C#, C++ o Java.

El seguimiento de BD se puede configurar, es decir, puede escoger entre hacer el seguimiento de todos los mensajes, de los errores solamente o puede deshabilitarlo por completo. También puede usar el seguimiento a otros niveles:

- A nivel de componente de BD. A este nivel puede habilitar o deshabilitar el seguimiento para ese componente de BD en concreto. El seguimiento a este nivel es útil para asignaciones que tienen varios componentes BD de destino de los que solo necesita monitorizar uno.
- A nivel de tabla o de procedimiento almacenado. En un componente de BD puede decidir si hace o no seguimiento de cada tabla o procedimiento almacenado. Si el seguimiento está habilitado a este nivel se incluyen en él acciones de tabla como "Insertar todos", "Actualizar si", "Omitir si" o "Eliminar si".

En el caso de los procedimientos almacenados el seguimiento conserva las acciones relacionadas con la llamada al procedimiento almacenado en cuestión.

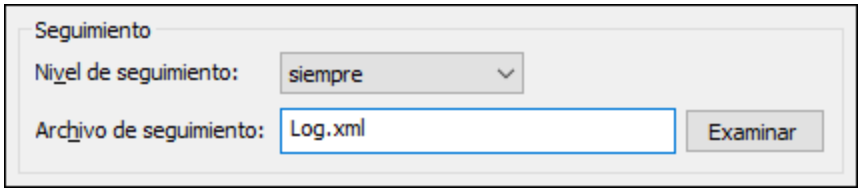
- A nivel de campo de BD. La configuración predeterminada hace seguimiento de todos los campos, pero puede excluir los que no quiere que se incluyan o puede indicar que ciertos campos se incluyan en el archivo de seguimiento solamente si ocurre un error.

Es importante recordar que los tres niveles que acabamos de mencionar son jerárquicos y que la configuración del seguimiento se aplica de arriba a abajo. Por lo tanto, si deshabilita el seguimiento a nivel del componente de BD no lo puede habilitar a nivel de tabla o de procedimiento almacenado. Asimismo, si deshabilita el seguimiento a nivel de tabla no es posible habilitarlo a nivel de campo de BD (columna). Lo mismo ocurre si restringe el seguimiento. Por ejemplo, si lo limita a solamente errores a nivel de los componentes de BD, entonces no puede usar el seguimiento completo a nivel de tabla o de procedimientos almacenados.

Seguimiento a nivel de componente de base de datos

Para habilitar el seguimiento a nivel de componente de BD:

1. Haga clic con el botón derecho en el componente de la base de datos y seleccione **Propiedades** en el menú contextual (también puede hacer doble clic en la barra del título del componente de la BD). Se abre el cuadro de diálogo "Configuración del componente".



2. Seleccione el nivel deseado de seguimiento (**Siempre**, **Errores**, **Deshabilitado**). Si selecciona **Errores** se creará un archivo de seguimiento solo en caso de que ocurran errores al ejecutar la asignación.
3. Junto a **Archivo de seguimiento**, teclee el nombre del archivo XML en el que se deben escribir los detalles del seguimiento (por ejemplo, **Log.xml**) o la ruta de acceso al mismo. Si introduce un nombre de archivo en lugar de una ruta completa, el archivo de seguimiento se creará en el mismo directorio en el que se está ejecutando la asignación.

Cuando se habilita el seguimiento, el componente de BD cambia su estructura en la asignación y muestra, además de la estructura de la base de datos, una nueva jerarquía de nodos en la que el elemento superior es el archivo de seguimiento. Para más información consulte el apartado Estructura del archivo de seguimiento 438
.

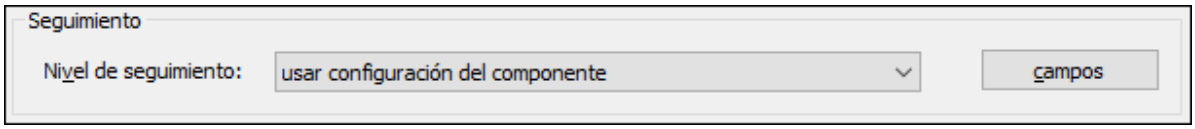
Al habilitar el seguimiento a nivel de componente de BD se habilita automáticamente también para todas las tablas o los procedimientos almacenados del componente que tienen una conexión con la asignación. De ser necesario, puede seguir configurando el seguimiento a nivel de tabla o de los procedimientos almacenados, como se describe más abajo. Observe que las tablas o los procedimientos almacenados que se muestran en el componente pero no participan en la asignación (es decir, que no tienen ninguna conexión con la asignación) no se monitorizan.

Seguimiento a nivel de tabla o de procedimiento almacenado

Para habilitar el seguimiento a nivel de tabla o de procedimiento almacenado:

 © 2016-2022 Altova GmbH

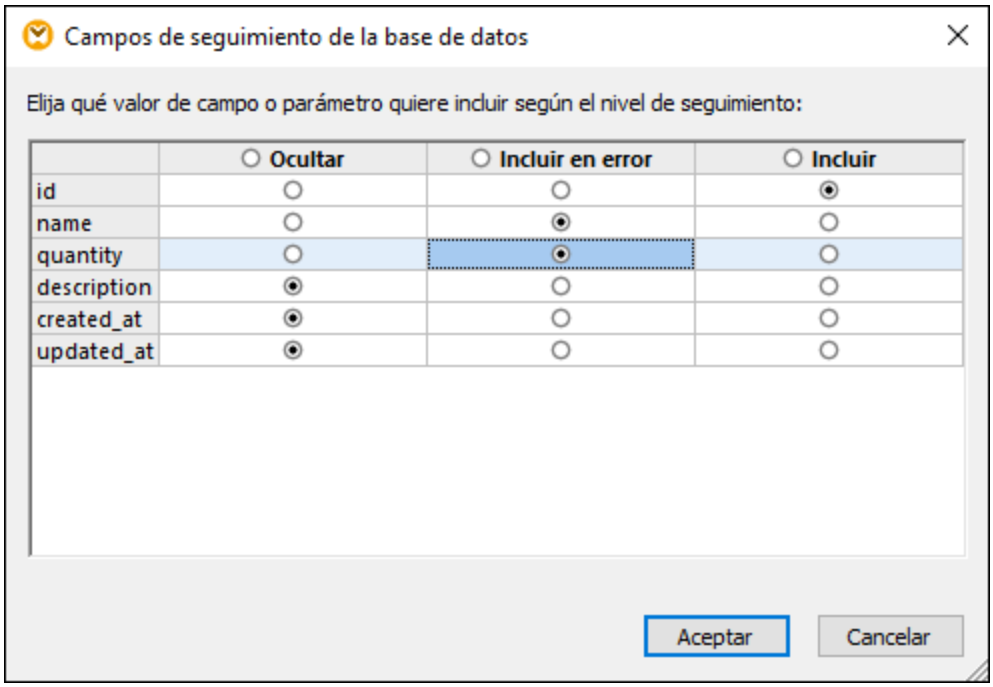
1. Asegúrese de que el nivel de seguimiento a nivel de los componentes de DB es **Siempre** o **Errores** (*véase más arriba*).
2. Elija una opción:
   - haga clic en el botón de acción de la tabla (por ejemplo A:In) junto a la tabla para la que quiere habilitar el seguimiento, o
   - si se trata de un procedimiento almacenado, haga clic en el botón **Mostrar menú contextual** y luego seleccione **Configuración del procedimiento** en el menú contextual.



3. Seleccione el nivel de seguimiento. La opción **Usar configuración del componente** hereda la misma configuración que se definió a nivel del componente. La opción **Limitar a los errores** restringe el seguimiento a los errores solamente. **Siempre deshabilitado** significa que no se hará ningún seguimiento para esa tabla o ese procedimiento almacenado.

Seguimiento a nivel de campo de la base de datos

Cuando se habilita el seguimiento a nivel de componente de BD y a nivel de tabla o de procedimiento almacenado, todos los campos (columnas de BD) se incluyen automáticamente en el informe de seguimiento. También puede limitar el seguimiento a solamente algunos campos de DB o parámetros de procedimientos almacenados en concreto. Para ello haga clic en el botón **Campos** de la Tabla de acciones de BD o del cuadro de diálogo "Configuración del procedimiento de la base de datos" (*véase más arriba*).

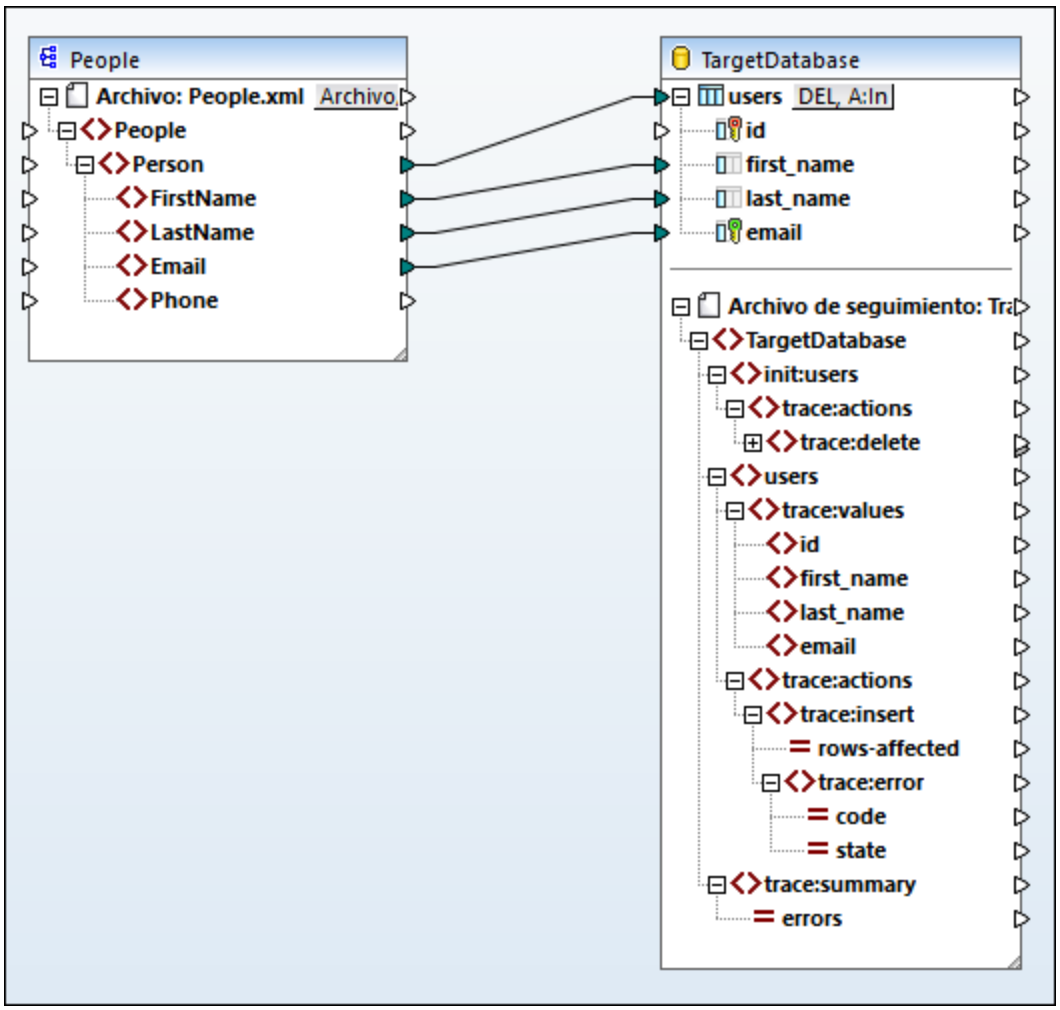


Los elementos que seleccione en este cuadro de diálogo afectan a la estructura del archivo de seguimiento. Por ejemplo, los campos que haya optado por ocultar no aparecerán en el archivo de seguimiento. Puede que quiera hacer esto, por ejemplo, para reducir el tamaño del archivo de registro.

Para ocultar o incluir varios campos al mismo tiempo haga clic en el encabezado de la opción correspondiente.

4.2.11.1 Estructura del archivo de seguimiento

Al habilitar el seguimiento para un componente de la base de datos aparece una estructura de seguimiento en la parte inferior del componente, por ejemplo:



Como se ve en la imagen anterior, el nodo superior de la estructura de seguimiento indica el nombre del archivo de seguimiento que se escribirá cuando se ejecute la asignación. El resto de la estructura de seguimiento se modela en función de las tablas de BD o de los procedimientos almacenados que participen en la asignación. En este ejemplo el nombre de la base de datos es "TargetDatabase", por lo que en la estructura de seguimiento también existe un elemento similar.

El elemento secundario de "TargetDatabase" es `users`, que corresponde con el "users" de la tabla de BD. En este ejemplo el componente de la BD solo tiene una tabla, "users". Sin embargo, en un componente con muchas tablas la estructura de seguimiento incluirá todas las tablas presentes en el componente. Si añade o elimina tablas o procedimientos almacenados del componente de BD, entonces la estructura de seguimiento se actualizará en consonancia. Encontrará las instrucciones en el apartado Agregar, editar y quitar objetos de BD 252.

En este ejemplo también aparece la estructura **init:users**. Esto ocurre si ha configurado la asignación para que ejecute alguna acción inicial antes de las acciones principales de insertar, actualizar, etc. Por ejemplo, si eliminó todas las entradas de registro que existían en la tabla antes de insertar nuevas, entonces estará presente la estructura **init**. Esta estructura recoge la información de seguimiento sobre esa acción inicial (como eliminar o puede que una consulta SQL personalizada) y cualquier error que pueda ocurrir en esa fase.

La estructura **trace:values** muestra todas las columnas de la tabla de BD (en este caso, las columnas de la tabla "users"). En el caso de los procedimientos almacenados esta estructura muestra los parámetros del procedimiento almacenado. Por defecto, todas las columnas están incluidas en el seguimiento, pero puede configurar de qué columnas se debe hacer seguimiento, como hemos explicado anteriormente.

Los elementos **trace:actions** indican qué acciones se han definido para que se ejecuten en la tabla de BD. En este ejemplo hay una acción de actualización de BD A:Up definida, por lo que existe el elemento **trace:update**. Si cambia el componente para que ejecute una acción de inserción en la BD A:In, entonces la estructura de seguimiento cambiaría a **trace:insert** y lo mismo ocurre con otros tipos de acciones. En el caso de los procedimientos almacenados la acción que aparece es **trace:execute**.

Cada acción de seguimiento tiene un atributo **rows-affected** que indica cuántas filas fueron afectadas por cada acción de BD.

El elemento **trace:error** solo se rellena si ocurre un error de BD durante la ejecución de la asignación. Tiene dos atributos: `code` y `state`. El texto del error y los valores del atributo vienen datos por el controlador de la base de datos, por lo que son distintos para cada tipo de BD.

El elemento **trace:summary** incluye un atributo `errors`, que informa del número de errores encontrados. El valor del atributo es **0** si no se encuentra ningún error.

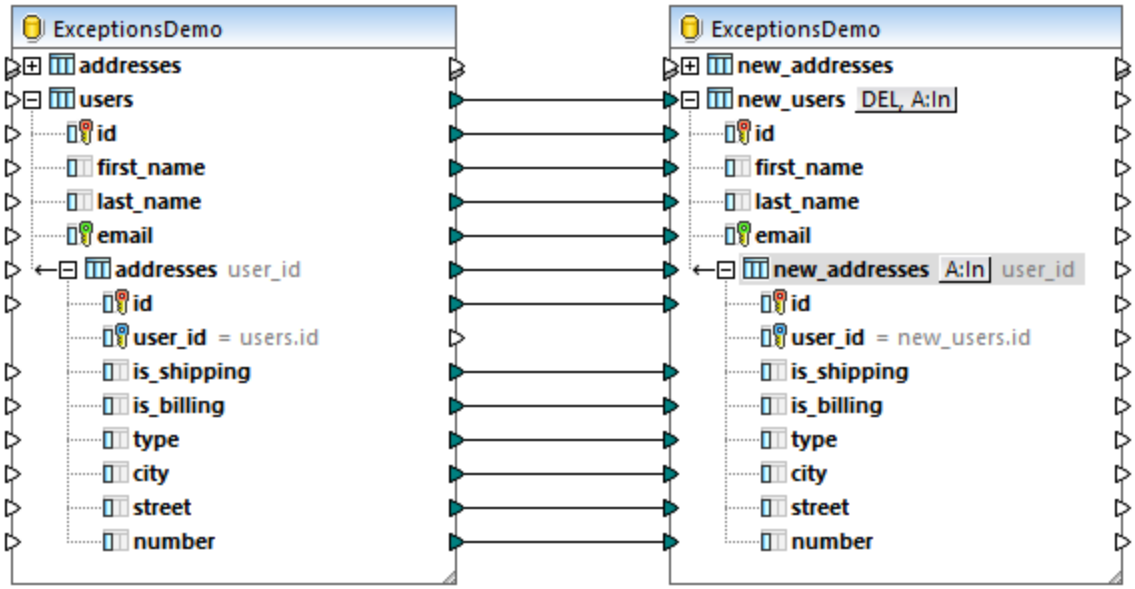
Vista previa de archivos de seguimiento

Cuando se habilita el seguimiento para un componente de BD de destino se puede acceder a una vista previa del mismo haciendo clic en la pestaña *Resultados*. Esto es útil si quiere ver qué aspecto aproximado tendrá el archivo. Tenga en cuenta que el archivo de seguimiento que se genera finalmente tendrá un aspecto distinto en función del resultado de la ejecución de la asignación.

4.2.11.2 Ejemplo: registrar errores de BD

Este ejemplo explica cómo configurar el seguimiento de BD y el registro de errores para asignaciones que insertan datos en una base de datos SQLite. Las instrucciones son parecidas a las que se usan para otros tipos de BD compatibles con MapForce y para otros tipos de acciones de BD (como "actualizar", "omitir", "eliminar", etc).

Puede encontrar la asignación que usamos en este ejemplo en **<Documentos>\Altova\MapForce2023\MapForceExamples\Tutorial\DatabaseExceptions.mfd**. Esta asignación inserta los datos de dos tablas de una BD en dos tablas nuevas en la misma BD; encontrará una descripción más detallada de la misma en el apartado Ejemplo: revertir transacciones 428. Observe que algunos de los registros de la BD son `null` a propósito para que se generen excepciones de BD.



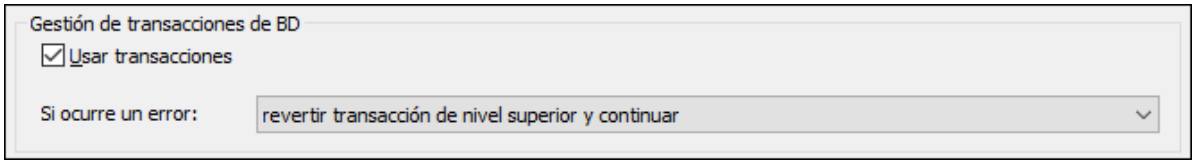
DatabaseExceptions.mfd

En este ejemplo el objetivo es habilitar el seguimiento para el componente de BD de destino para determinar cuántos errores se generaron y dónde ocurrieron exactamente. A continuación describimos los pasos que hay que seguir para alcanzar este objetivo.

Paso 1: habilitar la reversión de transacciones

Esta asignación de ejemplo está configurada para revertir transacciones a nivel de los componentes de BD. Esto significa que cuando aparece la primera excepción la asignación se detiene y revierte todos los datos de vuelta a su estado original. Primero vamos a cambiar las opciones de reversión de transacciones para que la asignación siga ejecutándose incluso aunque se encuentren excepciones para algunas de las entradas del registro:

1. Haga clic en el botón DEL, A:In junto a la tabls `new_users`, marque la casilla *Usar transacciones* y elija **revertir transacción actual y continuar**.



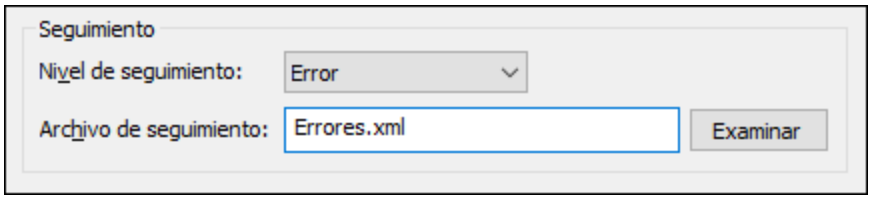
2. Haga clic en el botón A:In junto a la tabla `new_addresses`, marque la casilla *Usar transacciones* y elija **revertir transacción actual y continuar**.

También puede deshabilitar por completo la reversión de transacciones, pero entonces la asignación se detendría con el primer error que ocurriera y *no* se revertirían los cambios, por lo que no recomendamos esta opción. Con esta opción deshabilitada el seguimiento y el registro de errores solo reflejarían la ejecución de la asignación hasta ese primer error. Para más información sobre cómo revertir transacciones consulte el apartado Gestionar excepciones de BD 422.

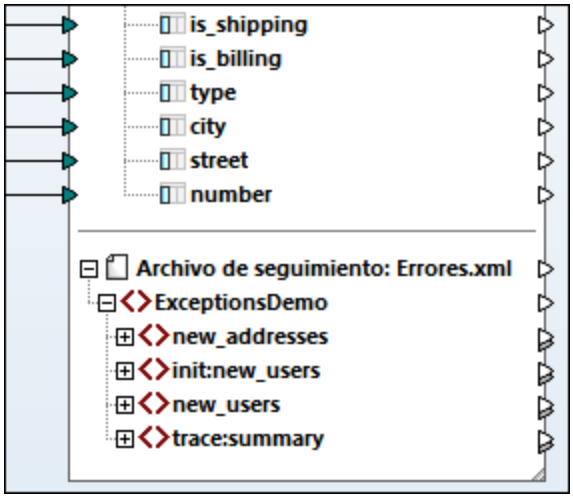
Paso 2: configurar el registro de errores

Para habilitar el seguimiento, siga estos pasos:

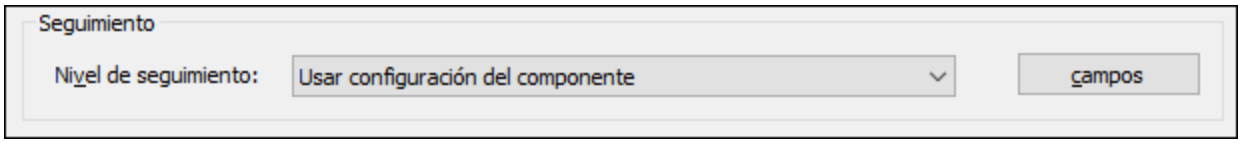
1. Haga clic con el botón derecho en la barra del título del componente de BD y seleccione **Propiedades** en el menú contextual (también puede hacer doble clic en la barra del título del componente de BD). Se abre el cuadro de diálogo "Configuración del componente".
2. Seleccione **Errores** en el nivel de seguimiento.
3. Junto a **Archivo de seguimiento**, teclee el nombre que quiere darle al archivo de seguimiento (en este ejemplo, **Errors.xml**). No introduzca una ruta absoluta si quiere ejecutar la asignación con MapForce Server en otros equipos o sistemas operativos.



Ahora la estructura del componente de BD de destino ha cambiado y muestra un compartimento más que contiene la estructura del archivo de seguimiento:



Hasta ahora hemos habilitado el seguimiento a nivel de los componentes de BD. En consecuencia, la configuración del seguimiento la heredarán todas las acciones de tabla de ese componente de BD. En este ejemplo la configuración la heredan las acciones "Insertar" de las tablas `new_users` y `new_addresses`. Si hace clic en el botón A:In de esas dos tablas, la configuración lo reflejará:



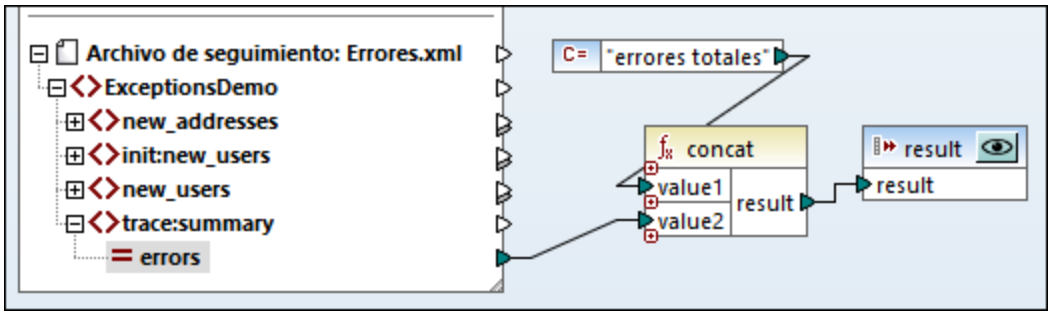
También puede hacer clic en el botón **Campos** para configurar qué campos se deben incluir u ocultar en el archivo de seguimiento. En este ejemplo se han incluido todos los campos.

Paso 3: configurar los resultados de la asignación

Este paso muestra cómo asignar los valores del archivo de seguimiento a otro tipo de componente (en este caso, un componente simple de salida). Tenga en cuenta que este paso es opcional; se crea un archivo de registro de errores aunque no asigne el archivo de seguimiento a otro componente. Sin embargo, en este ejemplo queremos configurar la asignación explícitamente para que devuelva el número total de errores encontrados en los resultados. Para ello vamos a seguir estos pasos:

1. Añada un componente simple de salida con el comando de menú **Insertar | Insertar componente de salida**. Para más información sobre componentes de salida consulte el apartado Obtener valores de cadena de una asignación 488.
2. Añada una constante con el comando de menú **Insertar | Constante**. La constante muestra el texto "Errores totales: " para que el texto de los resultados de la asignación resulte más claro.
3. Arrastre la función `concat` desde la ventana *Bibliotecas* a la asignación. Para más información sobre funciones integradas consulte el apartado Agregar una función integrada a una asignación de datos 550.

Como se muestra más arriba, la función `concat` concatena el texto de la constante con el valor **errors** suministrado por el archivo de seguimiento.



Con la configuración de la imagen anterior, cuando se ejecuta la asignación con MapForce los resultados mostrarán el texto "Errores totales: {value}", donde **value** representa el número de errores encontrados en el momento de la ejecución.

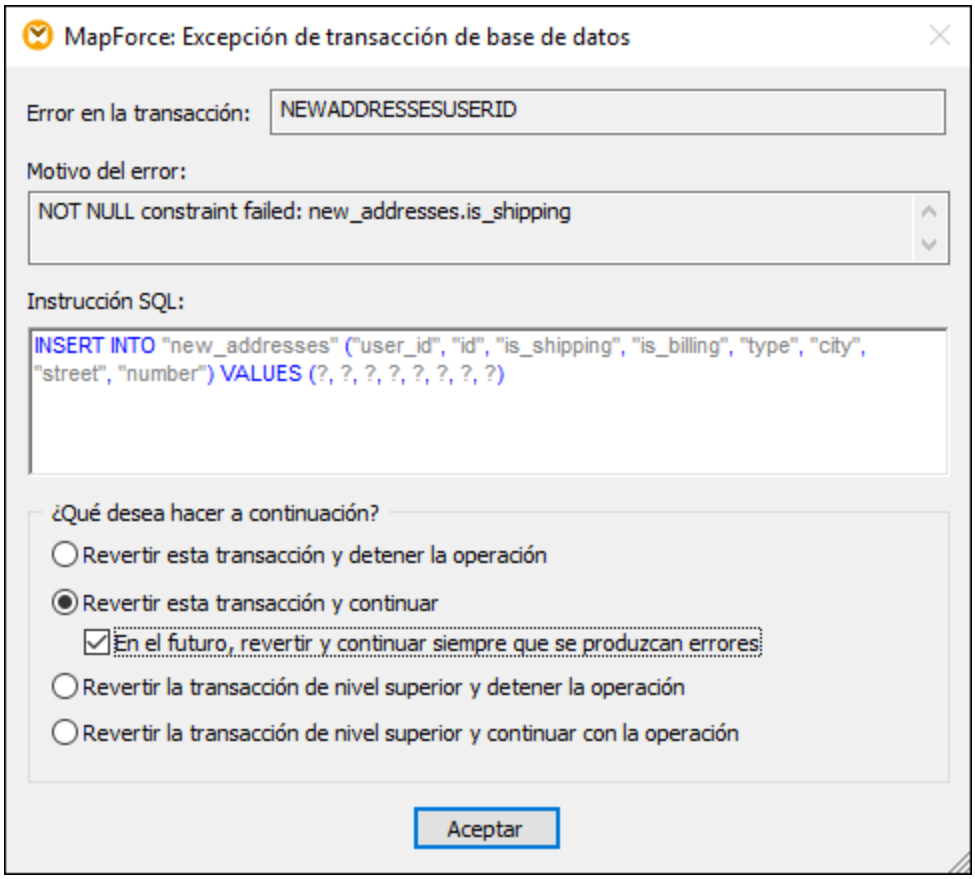
> En este momento no se puede ejecutar la asignación con el botón de vista previa activado en el componente "result" y hacerlo provocaría un error. Primero debe existir un archivo de seguimiento, ya que el componente simple de salida lee los datos de ese archivo de seguimiento. En el siguiente paso explicamos cómo crear uno.

Paso 4: ejecutar la asignación

Para ejecurtar esta asignación con MapForce:

1. Haga clic en el botón de **vista previa** del componente de BD.
2. Haga clic en la pestaña *Resultados*.
3. En el menú **Resultados**, haga clic en **Ejecutar script SQL**. En este momento la asignación se encuentra la primera excepción, que hemos incluido a propósito. Aparece el cuadro de diálogo "Excepción de transacción de BD".

 © 2016-2022 Altova GmbH



4. Seleccione **Revertir esta transacción y continuar** y haga clic en **Aceptar**.

Nota: si se ejecuta una asignación con MapForce Server no aparece ningún cuadro de diálogo y las transacciones con errores se revierten automáticamente de acuerdo a las opciones configuradas en MapForce.

En este ejemplo, una vez se ha completado la ejecución de la asignación se revierten todas las transacciones y solo se envían a la base de datos aquellas que no hayan tenido errores. Los errores encontrados se escriben en el archivo **Errors.xml**; a continuación mostramos un breve extracto del mismo:

```
11            <trace:insert rows-affected="1"/>
12        </trace:actions>
13        <new_addresses>
14            <trace:values>
15                <id>2</id>
16                <user_id>1</user_id>
17                <is_shipping xsi:nil="true"/>
18                <is_billing>1</is_billing>
19                <type>home</type>
20                <city>Bridgedell</city>
21                <street>Olive Street</street>
22                <number>6</number>
23            </trace:values>
24            <trace:actions>
25                <trace:insert>
26                    <trace:error code="19" state="1299">NOT NULL
constraint failed: new_addresses.is_shipping</trace:error>
27                </trace:insert>
28            </trace:actions>
29        </new_addresses>
30    </new_users>
31    <new_users>
```

Texto | Cuadrícula | Esquema | WSDL | XBRL | Authentic | Explorador

Errores.xml

Ahora que tenemos un archivo de registro se puede acceder a la vista previa que devuelve el número total de errores durante la ejecución en MapForce. Para ello:

1. En la asignación, haga clic en el botón de **vista previa** junto al componente "result".
2. Haga clic en la pestaña *Resultados* y observe los resultados.

```
1    errores totales3
```

Asignación | Consulta de la BD | Resultados

DatabaseExceptions.mfd*

Esta parte de la asignación lee datos del archivo de registro que creamos previamente. Si este archivo no existe se generará un error.

© 2016-2022 Altova GmbH

4.3 Archivos CSV y archivos de texto

MapForce admite asignaciones de datos entre formatos de archivo basados en texto como archivos de texto CSV (valores separados por comas) y archivos de texto FLF (campos de longitud fija). Por ejemplo, en MapForce puede crear estas transformaciones de datos:

- transformar esquemas XML en archivos planos y viceversa
- transformar bases de datos en archivos planos y viceversa

Recuerde que, en el caso del formato CSV, los archivos pueden usar delimitadores como comas, tabulaciones, punto y comas, espacios y otros valores.

Además de asignaciones de datos de archivos CSV y FLF, MapForce permite trabajar con estructuras más complejas gracias a la aplicación MapForce FlexText, un módulo disponible con la edición MapForce Enterprise Edition. FlexText permite definir la estructura de los datos de texto (con ayuda de una *plantilla FlexText)* y así poder asignarlos a otros formatos.

La asignación de datos de archivos de texto solamente es compatible con los lenguajes de transformación Java, C#, C++ y BUILT-IN.

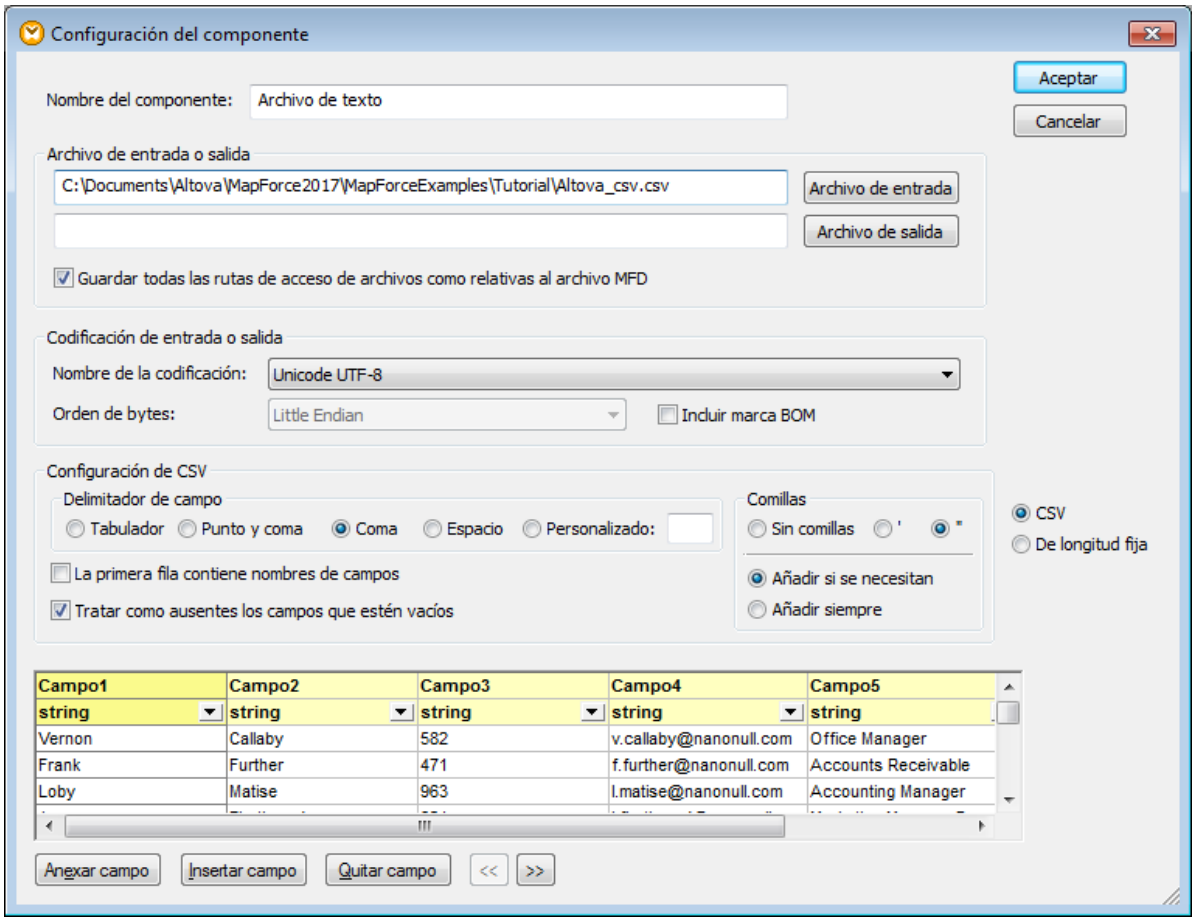
Hay dos maneras de generar resultados para asignaciones datos de archivos planos:

- Abriendo el panel *Resultados*: esto genera una vista previa con el motor de ejecución integrado. También puede guardar el resultado de la asignación con el comando de menú **Resultados | Guardar el archivo de salida...** o con el botón **Guardar resultado generado** de la barra de herramientas del panel *Resultados*.
- Seleccionando el comando de menú **Archivo | Generar código en | Java/C#/C++** y compilando y ejecutando el código generado.

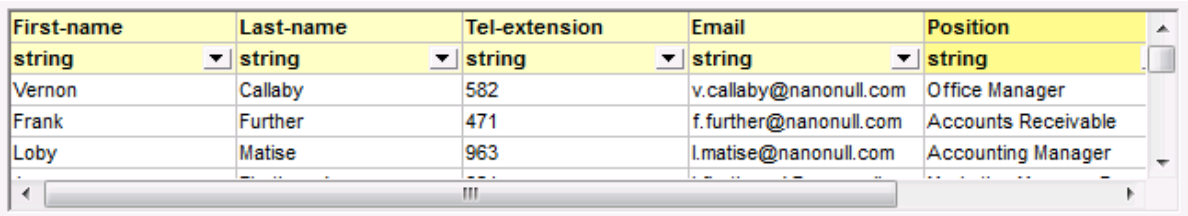
4.3.1 Ejemplo: asignar archivos CSV a XML

El objetivo de este ejemplo es crear una asignación que lea datos de un archivo CSV simple y los escriba en un archivo XML de destino. Los archivos utilizados en este ejemplo se encuentran en la carpeta `<Documentos>\Altova\MapForce2023\MapForceExamples\Tutorial\`.

1. Seleccione uno de estos lenguajes de transformación: Java, C#, C++ o BUILT-IN.
2. Añada un componente de archivo de texto al área de asignación (con el comando de menú **Insertar | Archivo de texto** o con el botón **Insertar archivo de texto** de la barra de herramientas).
3. En el cuadro de diálogo "Configuración del componente" haga clic en el botón Archivo de entrada y navegue hasta el archivo **Altova_csv.csv**. El contenido del archivo aparece ahora en la parte inferior del cuadro de diálogo. Observe que en el modo de vista previa solo aparecen las primeras 20 filas del archivo de texto.



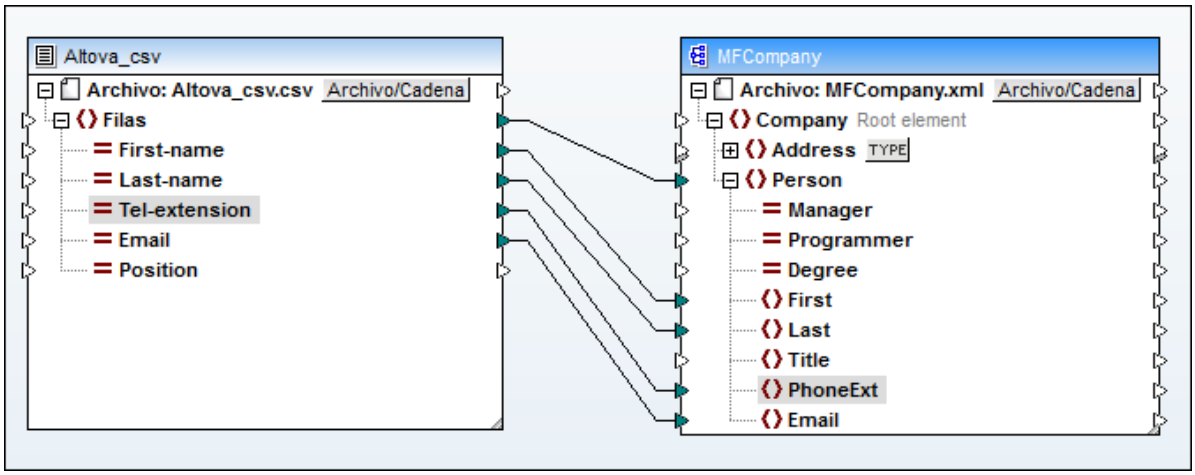
4. Haga clic dentro del encabezado Campo1 y cambie el texto por First-name. Haga lo mismo con el resto de campos: Campo 2 => Last-name, Campo 3 =>Tel-extension, Campo 4 => Email, Campo 5 => Position. **Nota:** pulse la tecla **Tabulación** para recorrer los campos.



5. Cuando termine, haga clic en **Aceptar**.
6. Cuando la aplicación solicite un nombre nuevo para el componente, haga clic en **Cambiar el nombre del componente**. El componente CSV aparece ahora en la asignación.
7. Añada **MFCompany.xsd** como componente XML de destino de la asignación (con el comando **Insertar | Archivo o esquema XML**).
8. Haga clic en **Omitir** cuando la aplicación solicite un archivo XML de muestra y seleccione el elemento raíz `Company`.
9. Realice las asignaciones correspondientes entre los dos componentes (asegurándose de asignar el elemento `Filas` al elemento `Person` del esquema de destino).

 © 2016-2022 Altova GmbH

El conector que une el elemento `Filas` del componente CSV con el elemento `Person` del esquema es esencial porque define qué elementos se deben recorrer. Es decir, por cada fila del archivo CSV, se creará un nuevo elemento `Person` en el archivo XML de salida.



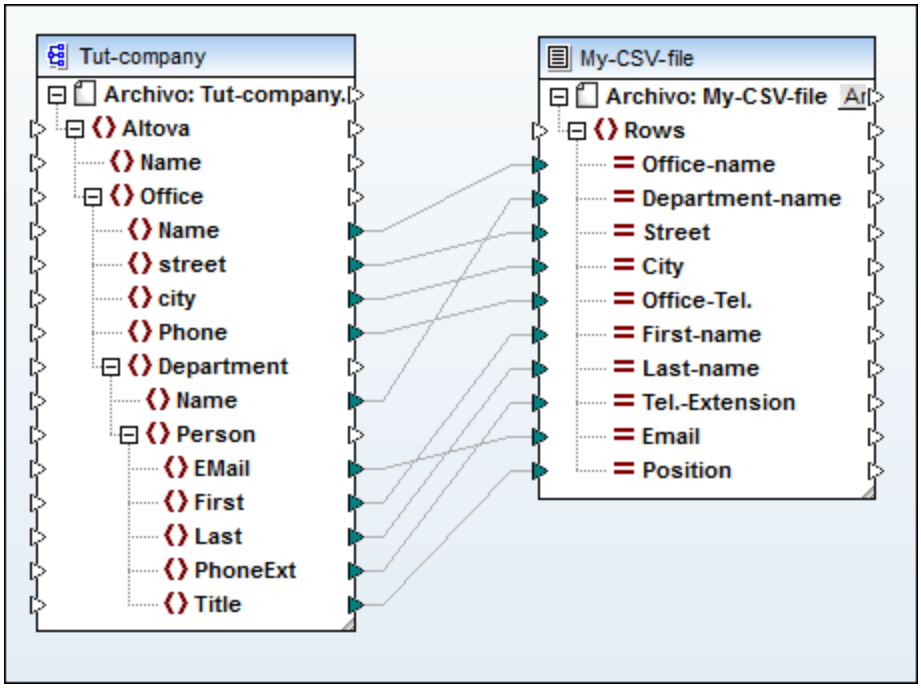
10. Abra el panel *Resultados* para ver el resultado de la asignación.

```
1   <?xml version="1.0" encoding="UTF-8"?>
2   <Company xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:
3     <Person Manager="true">
4       <First>Vernon</First>
5       <Last>Callaby</Last>
6       <PhoneExt>582</PhoneExt>
7       <Email>v.callaby@nanonull.com</Email>
8     </Person>
9     <Person Manager="true">
10      <First>Frank</First>
11      <Last>Further</Last>
12      <PhoneExt>471</PhoneExt>
13      <Email>f.further@nanonull.com</Email>
14    </Person>
15    <Person Manager="true">
```

Los datos del archivo CSV se asignaron correctamente al archivo XML de destino.

4.3.2 Ejemplo: recorrer elementos

Este ejemplo pretende explicar cómo se crean iteraciones (varias filas) en un archivo CSV de destino. El diseño de asignación que acompaña este ejemplo está en la ruta de acceso `<Documentos>\Altova\MapForce2023\MapForceExamples\Tutorial\Tut-xml2csv.mfd`.



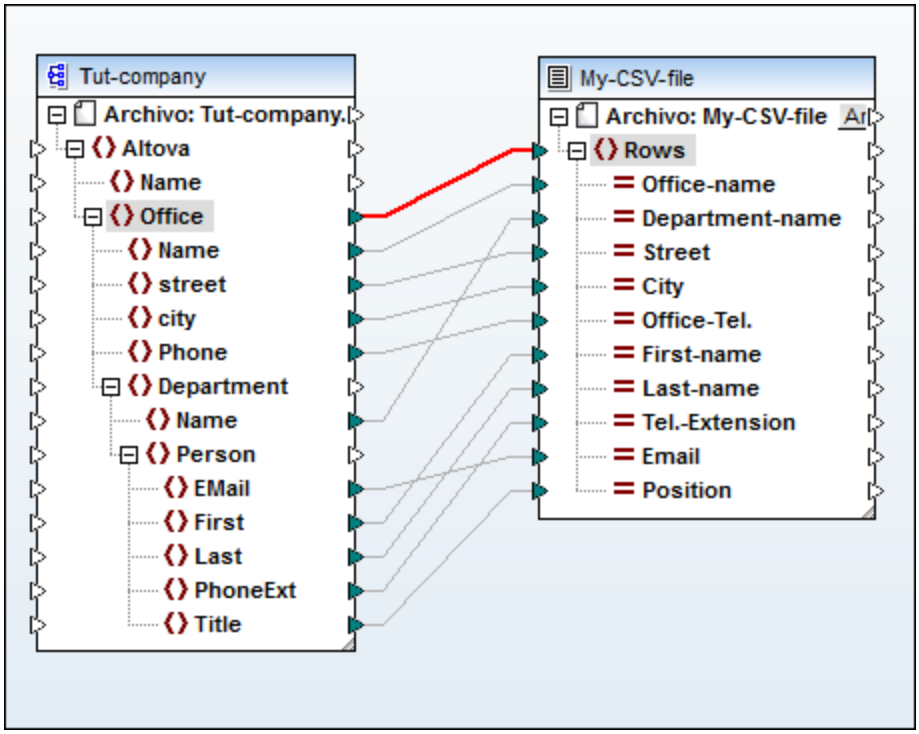
Tut-xml2csv.mfd

Tenga en cuenta que este diseño de asignación no está terminado a propósito. Si intenta validar el archivo de ejemplo con el comando de menú **Archivo | Validar asignación**, recibirá mensajes de advertencia. Además, si abre la vista previa de resultados, verá que solamente se genera una fila.

Imaginemos que lo que usted necesita es crear varias filas en el archivo CSV a partir de una secuencia de elementos del archivo XML. Esto se puede hacer dibujando una conexión con el elemento `Rows` del archivo CSV de destino.

Por ejemplo, para recorrer todas las oficinas y conseguir que el resultado aparezca en el archivo CSV es necesario conectar `Office` con `Rows`. Esta conexión indica que por cada elemento `Office` del archivo XML de origen MapForce debe crear una fila en el archivo CSV de destino.

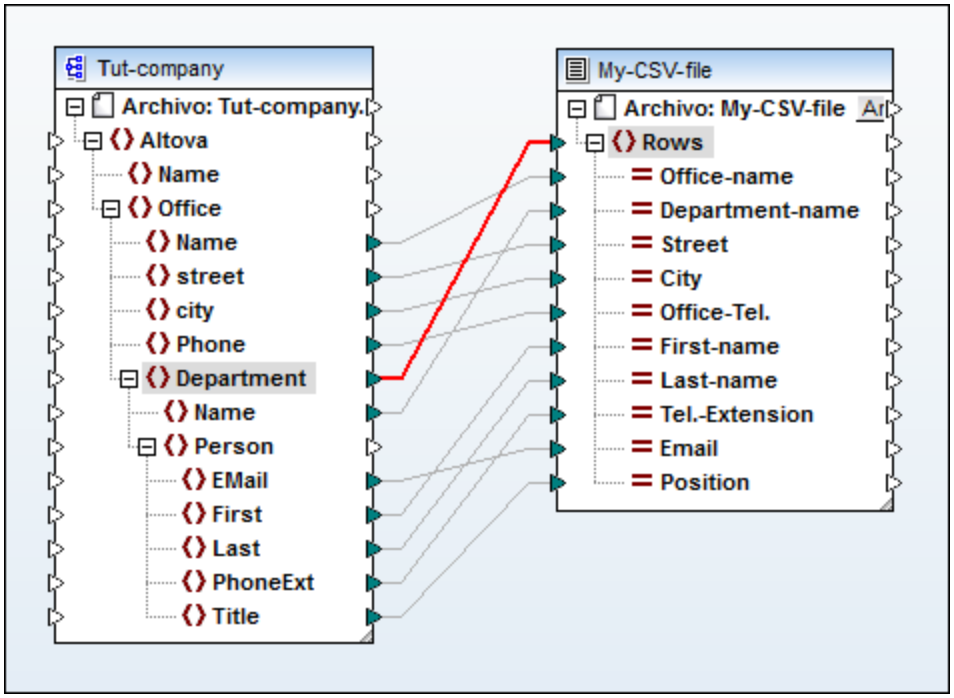
 © 2016-2022 Altova GmbH



El elemento `Rows` del componente CSV hace de iterador para la secuencia de elementos a los que está conectado. Por tanto, si conecta el elemento `Office`, el resultado crea una fila por cada oficina que se encontrara en el archivo XML de origen.

```
1   "Microtech, Inc.",Level 1 support,Major Ave 1,Vancouver,558833
2   "Microtech Partners, Inc.",Level 2 support,Perro Bvd 1324,Otto
3
```

Igualmente, si conecta **Department** a **Rows**, se generará una fila por cada departamento que se encuentre en el archivo XML de origen.



Este sería el resultado:

```
1  "Microtech, Inc.",Admin,Major Ave 1,Vancouver,5588339,Clive,Clo
2  "Microtech, Inc.",Sales and Marketing,Major Ave 1,Vancouver,558
3  "Microtech, Inc.",Manufacturing,Major Ave 1,Vancouver,5588339,K
4  "Microtech, Inc.",Level 1 support,Major Ave 1,Vancouver,5588339
5  "Microtech Partners, Inc.",Admin,Perro Bvd 1324,Ottowa,3549202,
6  "Microtech Partners, Inc.",Sales and Marketing,Perro Bvd 1324,O
7  "Microtech Partners, Inc.",Level 2 support,Perro Bvd 1324,Ottow
8
```

Por último, la asignación entre `Person` y `Rows` hace que en el resultado se generen todos los elementos `Person`. En este caso MapForce recorre los registros de esta manera: cada `Person` dentro de cada `Department`, dentro de cada `Office`.

4.3.3 Ejemplo: crear jerarquías a partir de archivos CSV y de texto de longitud fija

Este ejemplo es la asignación `<Documentos>\Altova\MapForce2023\MapForceExamples\Tutorial\Tut-headerDetail.mfd` y utiliza un archivo CSV llamado `Orders.csv` que tiene este formato:

- Campo 1: H define un registro de encabezado y D un registro detallado.
- Campo 2: una clave común tanto para los registros de encabezado como para los registros detallados.
- Cada registro de encabezado o detallado está en una línea distinta.

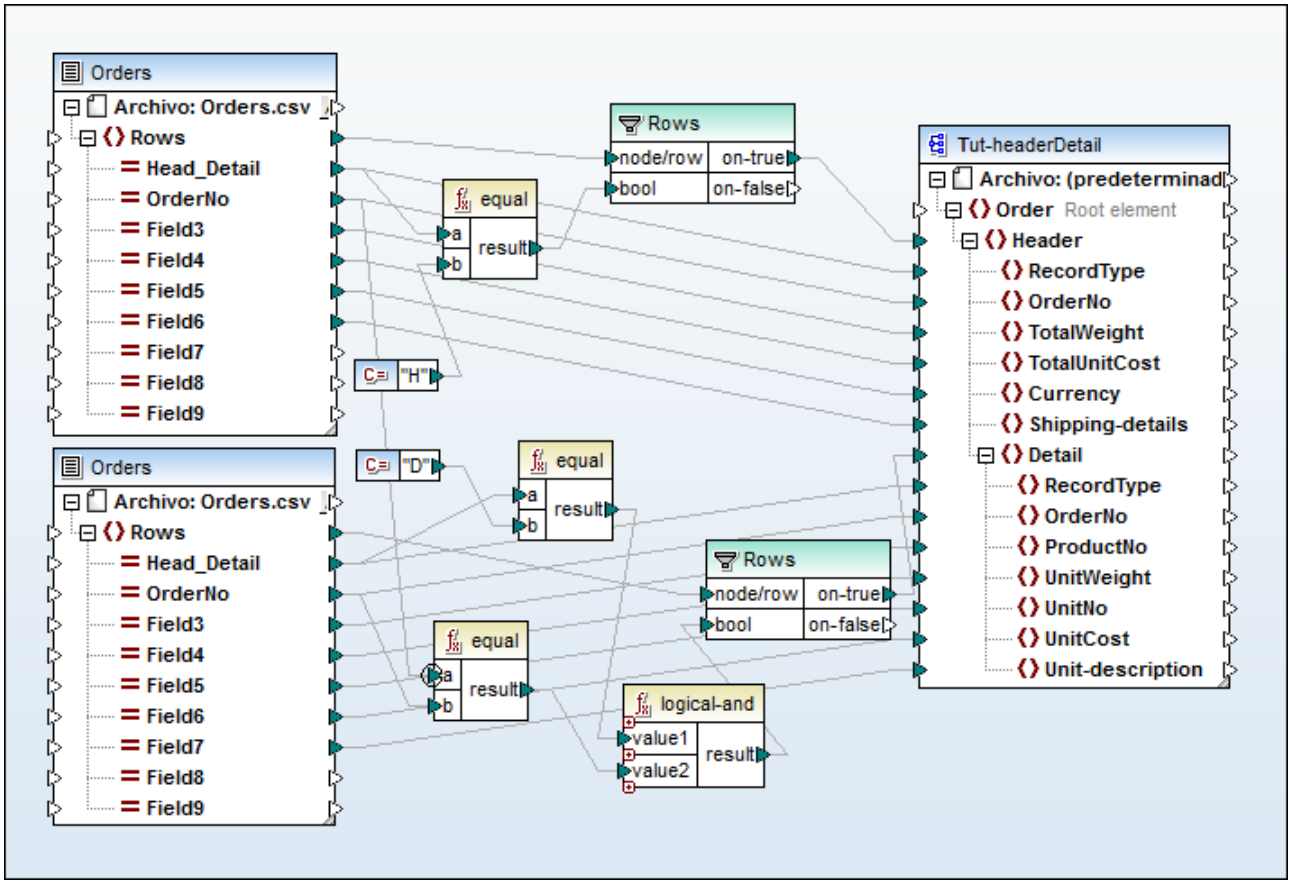
A continuación puede ver el contenido del archivo `Orders.csv`.

 © 2016-2022 Altova GmbH

```
H,111,332.1,22537.7,,Container ship,,,
D,111,A-1579-227,10,3,400,Microtome,,
D,111,B-152-427,7,6,1200,Miscellaneous,,
H,222,978.4,7563.1,,Air freight,,,
D,222,ZZ-AW56-1,10,5,10000,Gas Chromatograph,,
```

Nuestro diseño de asignación tiene varios objetivos:

- Asignar el archivo CSV plano a un archivo XML jerárquico.
- Filtrar los registros de encabezado (señalados con una H).
- Asociar los correspondientes registros detallados (señalados con una D) a cada uno de los registros de encabezado.



tut-headerDetail.mfd

Para conseguir nuestros objetivos primero debemos asegurarnos de que los registros de encabezado y los registros detallados tengan un campo común. En este caso, el campo común (también denominado *clave*) es el segundo campo del archivo CSV (es decir, `OrderNo`). En el archivo CSV, tanto el primer registro de encabezado como los dos registros detallados que le siguen contienen el valor común 111.

El archivo `Orders.csv` se insertó dos veces a fin de hacer el proceso de asignación más intuitivo.

El archivo de esquema **Tut-headerDetail.xsd** tiene una estructura jerárquica: `Order` es el elemento raíz, `Header` es su elemento secundario y `Detail` es un elemento secundario de `Header`.

El primer archivo `Orders.csv` aporta los registros de encabezado `Header` (y todos los campos asignados) al elemento `Header` del archivo de esquema de destino. El componente de filtrado se usa para filtrar solamente los registros que empiecen por H. El elemento `Rows` aporta estos registros filtrados al elemento `Header` del esquema de destino.

El segundo archivo `Orders.csv` aporta los registros detallados `Detail` (y todos los campos asignados) filtrando los registros `Detail` que coinciden con la clave `OrderNo` del registro `Header`. Este filtrado se hace así:

- Se compara el campo **OrderNo** del registro `Header` con el mismo campo de los registros `Detail` por medio de la función **equal** (el contexto prioritario 883 se estableció en el parámetro **a** para un mayor rendimiento).
- Se usa la función **Logical-and** para aportar solamente los registros `Detail` que contengan el mismo campo `OrderNo` que el registro `Header`.

A través del parámetro **on-true** del componente de filtrado, el elemento **Rows** aporta estos registros filtrados a los elementos Header y Detail del esquema de destino.

En el panel *Resultados* se puede ver el archivo XML de salida. Cada registro de encabezado contiene sus propios datos más los registros detallados asociados que tengan el mismo número de pedido.

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <Order xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation
3    <Header>
4      <RecordType>H</RecordType>
5      <OrderNo>111</OrderNo>
6      <TotalWeight>332.1</TotalWeight>
7      <TotalUnitCost>22537.7</TotalUnitCost>
8      <Currency/>
9      <Shipping-details>Container ship</Shipping-details>
10     <Detail>
11       <RecordType>D</RecordType>
12       <OrderNo>111</OrderNo>
13       <ProductNo>A-1579-227</ProductNo>
14       <UnitWeight>10</UnitWeight>
15       <UnitNo>3</UnitNo>
16       <UnitCost>400</UnitCost>
17       <Unit-description>Microtome</Unit-description>
18     </Detail>
19     <Detail>
20       <RecordType>D</RecordType>
21       <OrderNo>111</OrderNo>
22       <ProductNo>B-152-427</ProductNo>
23       <UnitWeight>7</UnitWeight>
24       <UnitNo>6</UnitNo>
25       <UnitCost>1200</UnitCost>
26       <Unit-description>Miscellaneous</Unit-description>
27     </Detail>
28   </Header>
```

 © 2016-2022 Altova GmbH

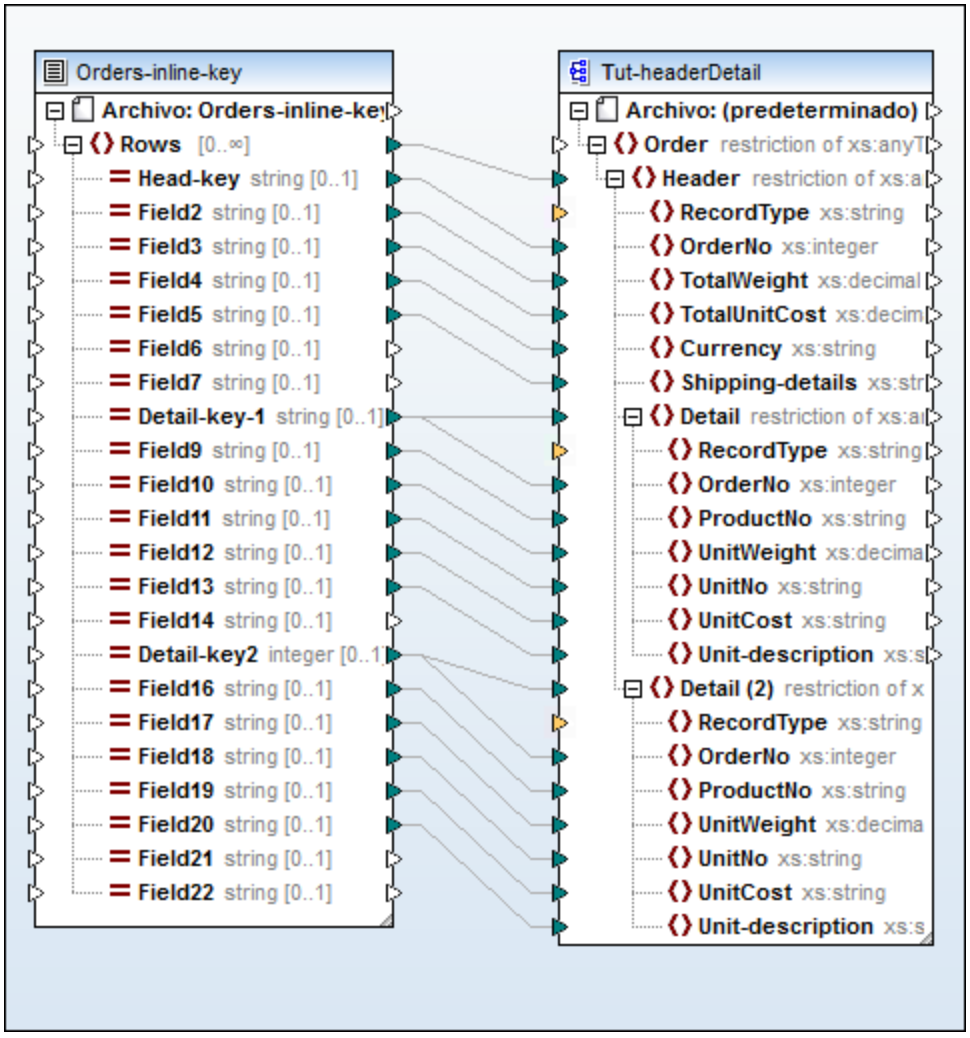
Veamos ahora otro ejemplo que usa un archivo CSV algo diferente: la asignación `<Documentos>\Altova\MapForce2023\MapForceExamples\Tutorial\Head-detail-inline.mfd`. Estas son las características de este nuevo archivo CSV:

- No incluye ningún designador de registro (H o D).
- Incluye un campo de clave común (el primer campo del archivo CSV) tanto para los registros de encabezado (Head-key) como para los detallados (Detail-key). Este campo está asignado al elemento OrderNo del esquema de destino.
- El encabezado y sus correspondientes campos detallados están en la misma línea.

```
111,332.1,22537.7,,Container ship,,,111,A-1579-227,10,3,400,Microtome,,111,B-15
222,978.4,7563.1,,Air freight,,,222,ZZ-AW56-1,10,5,10000,Gas Chromatograph,,
```

Teniendo en cuenta lo anterior, la asignación se diseñó así:

- Los campos clave se asignaron a los correspondientes elementos `OrderNo` del esquema de destino.
- El elemento `Detail` del esquema de destino se duplicó y aparece como **Detail (2)**. Esto permite asignar el segundo conjunto de registros detallados al elemento correcto.
- El resultado de la asignación es el mismo archivo XML que se generó en el primer ejemplo de este apartado.



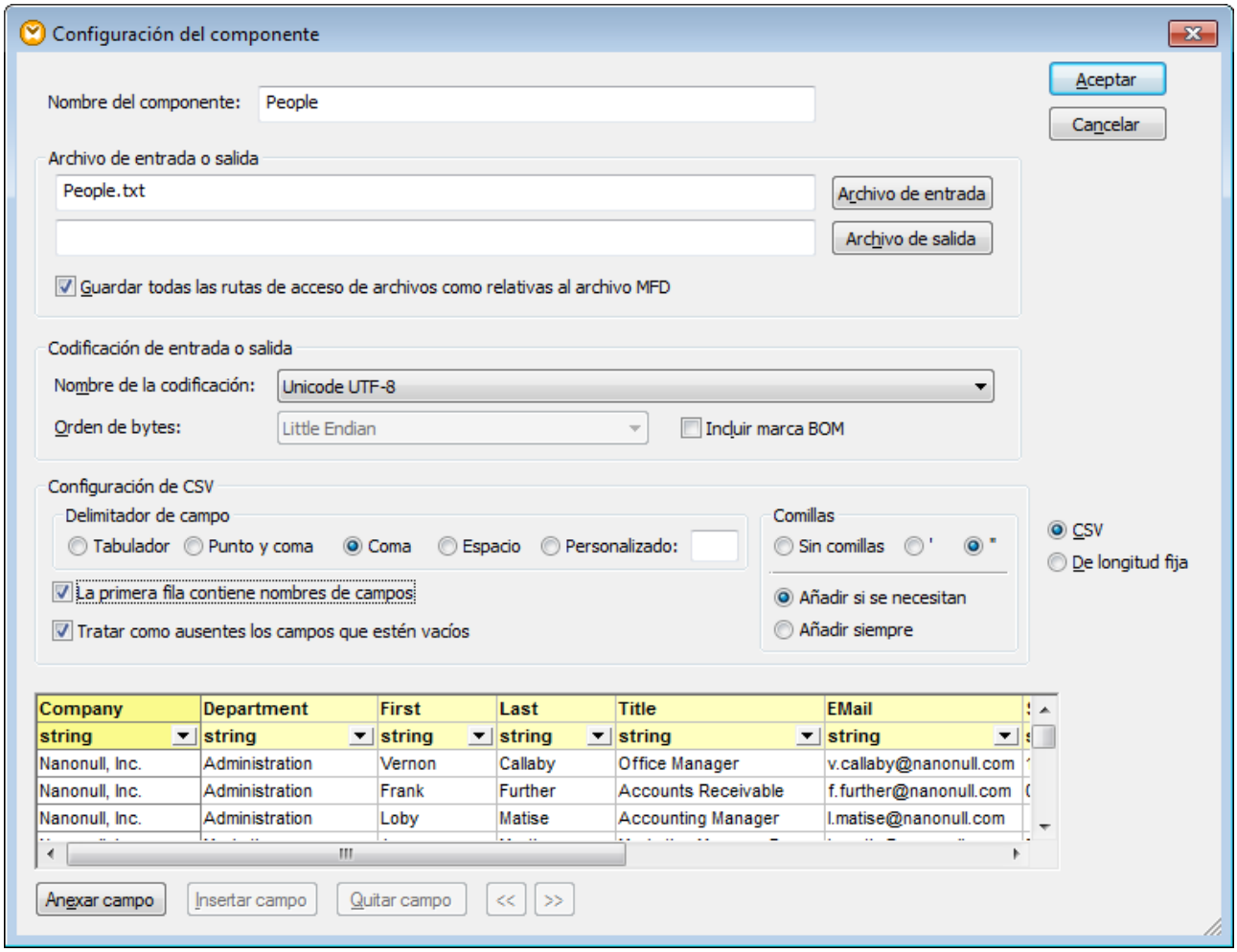
Head-detail-inline.mfd

4.3.4 Opciones de configuración de componentes CSV

Una vez añadido al área de asignación, el componente de texto se puede configurar desde el cuadro de diálogo "Configuración del componente". Este cuadro de diálogo se puede abrir de varias formas distintas:

- Seleccionando el componente y haciendo clic en el comando de menú **Componente | Propiedades**.
- Haciendo doble clic en el título del componente.
- Haciendo clic con el botón derecho en el título del componente y seleccionando **Propiedades** en el menú contextual.

 © 2016-2022 Altova GmbH



Cuadro de diálogo "Configuración del componente" (en modo CSV)

Estas son las opciones de configuración de los componentes CSV:

Nombre del componente	El nombre del componente se genera automáticamente al crearlo, pero se puede volver a cambiar cuando quiera. El nombre del componente puede contener espacios y puntos, pero no barra, barra inversa, dos puntos, comillas dobles, ni espacios antes ni después del nombre. Si cambia el nombre del componente, tenga en cuenta que: • Si quiere implementar la asignación en FlowForce Server, el nombre del componente debe ser único. • Recomendamos que use únicamente caracteres que se puedan usar también en la línea de comandos. Los caracteres especiales propios de cada país pueden estar codificados de forma diferente en Windows y la línea de comandos.

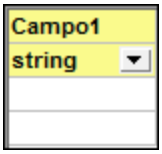
Archivo de entrada	Especifica el archivo donde MapForce debe leer los datos. Este campo es relevante para componentes de origen y se rellena cuando se crea el componente y se le asigna un archivo de texto. El campo puede dejarse vacío si se usa un componente de archivo de texto como destino de la asignación. En un componente de origen MapForce usa el valor de este campo para leer nombres de columna y obtener una vista previa del contenido del archivo de texto. Haga clic en **Archivo de entrada** para seleccionar un archivo nuevo.
Archivo de salida	Especifica el archivo donde MapForce debe escribir datos. Este campo es relevante para componentes de destino. Haga clic en **Archivo de salida** para seleccionar un archivo nuevo.
Guardar todas las rutas de acceso de archivos como relativas al archivo MFD	Si marca esta casilla, MapForce guarda las rutas de acceso de los archivos que aparecen en el cuadro de diálogo "Configuración del componente" como relativas a la ubicación del archivo de diseño de MapForce (.mfd). Esta opción afecta a los archivos de entrada y salida utilizados por el componente de texto (véase también *Usar rutas de acceso relativas en un componente* 82).
Codificación de entrada / salida	Permite configurar estos aspectos del archivo de instancia de salida: • Nombre de la codificación • Orden de bytes • El carácter de la marca de orden de byte (marca BOM). Por defecto todos los componentes nuevos tienen la codificación definida en la opción *Codificación predeterminada para componentes nuevos*. Para acceder a esta opción seleccione **Herramientas | Opciones | Generales**.
Delimitador de campos	Los archivos CSV están delimitados por comas por defecto. Esta opción permite elegir entre tres tipos de delimitadores: *tabulador*, *punto y coma* o *espacio*. También puede introducir un delimitador personal en el campo *Personalizado*.
La primera fila contiene nombres de campos	Seleccione esta opción para indicar a MapForce que debe tratar los valores del primer registro del archivo de texto como encabezados de columna. Los encabezados de columna aparecerán como nombres de elemento en la asignación.
Tratar como ausentes los campos que estén vacíos	Si marca esta casilla, los campos vacíos del archivo de origen no producirán un elemento correspondiente vacío (elemento o atributo) en el archivo de destino. Por ejemplo, el registro CSV `"General outgassing pollutants,,,,"` está compuesto por cuatro campos y los tres últimos están vacíos.

 © 2016-2022 Altova GmbH

	Imaginemos que el destino de la asignación es un archivo XML. Si se desactiva esta opción, los campos vacíos se crearán en el archivo de destino con un valor vacío (p. ej. los elementos `Last`, `Title` y `Email`): 33 <Person> 34 <First>General outgassing pollutants</First> 35 <Last/> 36 <Title/> 37 <Email/> 38 </Person> Si se activa esta opción, los campos vacíos no se crearán en el destino de la asignación: 38 <Person> 39 <First>General outgassing pollutants</First> 40 </Person>
Comillas	Si en el archivo de entrada los valores de campo están entrecomillados, seleccione qué carácter de comillas existe en el archivo de origen. La misma configuración se usará para los archivos de salida. Comillas Sin comillas ' " Añadir si se necesitan Añadir siempre Además, los archivos de salida cuentan con dos opciones de configuración más: **Añadir si se necesitan** — Añade el carácter de comillas seleccionado en campos cuyo texto contiene el delimitador de campo o saltos de línea. **Añadir siempre** — Añade el carácter de comillas seleccionado en todos los campos del archivo CSV generado.
CSV / De longitud fija	Cambia el tipo de componente por CSV o FLF (campo de longitud fija).
Panel de vista previa	La parte inferior del cuadro de diálogo muestra una vista previa de hasta 20 dilas del archivo seleccionado como entrada/salida. Si lo necesita, puede crear la estructura del archivo (o cambiarla) con estos botones:

Anexar campo	Crea un campo nuevo después del último regis CSV.
Insertar campo	Crea un campo nuevo inmediatamente antes d registro CSV que está seleccionado.
Quitar campo	Elimina el campo que está seleccionado.
<<	Desplaza el campo que está seleccionado hac izquierda.
>>	Desplaza el campo que está seleccionado hac derecha.

Para cambiar el nombre de un campo haga clic en el encabezado (por ejemplo, **`Campo1`**) y teclee el nuevo valor. Recuerde que los nombres de campo no se pueden editar si está marcada la casilla *La primera fila contiene nombres de campos*.



Para cambiar el tipo de datos de un campo seleccione el valor pertinente en la lista desplegable. MapForce revisa el tipo de datos y, si los datos de entrada y el formato del campo son discordantes, resalta los datos en rojo.

Los tipos de los campos se basan en los tipos de datos predeterminados de XML Schema. Por ejemplo, el tipo de fecha Date está en el formato **AAAA-MM-DD**.

4.3.5 Ejemplo: asignar archivos de texto de longitud fija a bases de datos

Este ejemplo sirve para explicar una operación de asignación de datos entre un archivo de texto de longitud fija (FLF) y una base de datos Microsoft Access. Los archivos que se utilizan en este ejemplo están en la carpeta `<Documentos>\Altova\MapForce2023\MapForceExamples\Tutorial\`. Tanto el archivo de texto de origen como la base de datos de destino almacenan una lista de empleados. Además, en el archivo de origen los registros están delimitados de forma implícita en función de su tamaño:

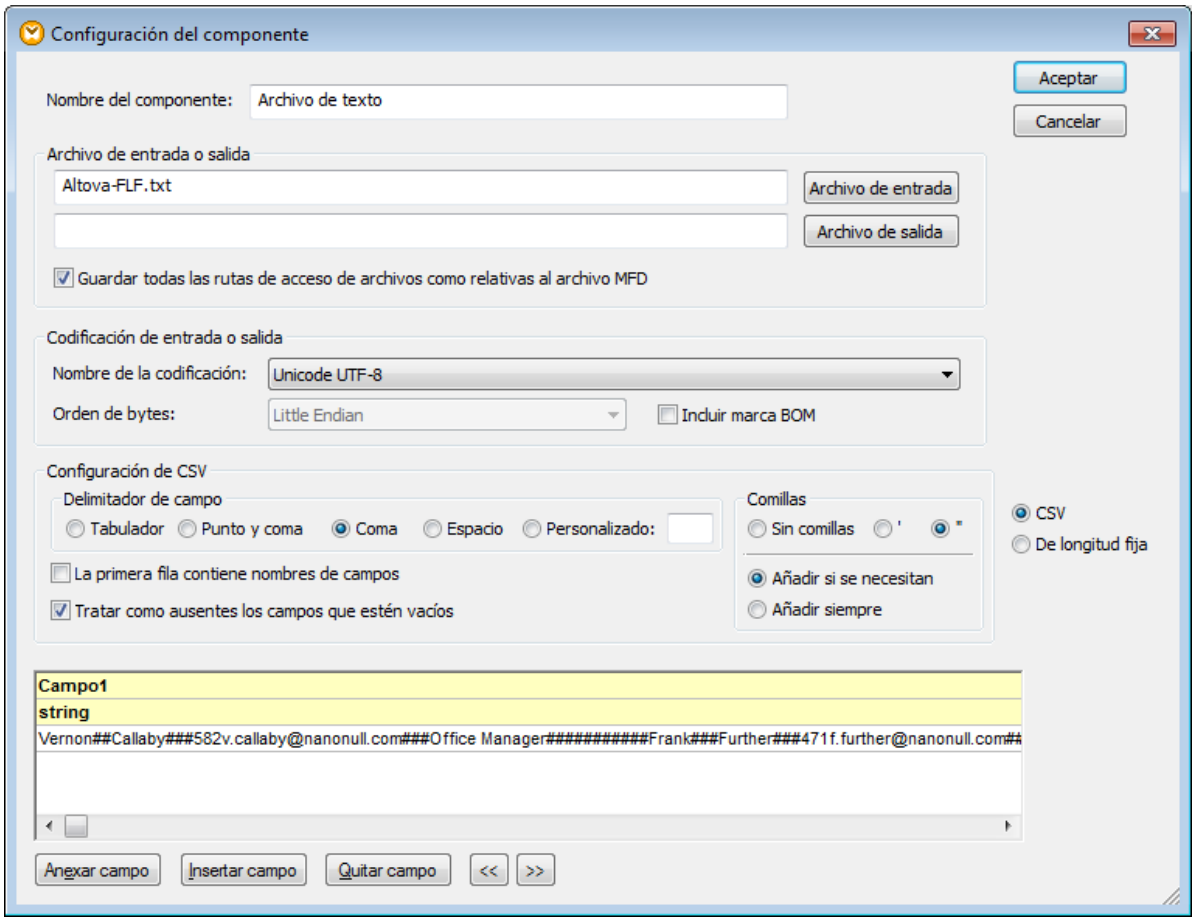
 © 2016-2022 Altova GmbH

Posición y nombre del campo	Tamaño (en caracteres)
Campo 1 (Nombre)	8
Campo 2 (Apellido)	10
Campo 3 (Extensión telefónica)	3
Campo 4 (Correo electrónico)	25
Campo 5 (Posición)	25

El objetivo de la asignación es actualizar la extensión telefónica de cada empleado en la base de datos con la extensión telefónica del archivo de origen, añadiendo además el prefijo 100 a cada extensión.

Paso nº1: insertar y configurar el componente de texto

1. Seleccione la opción de menú **Insertar | Archivo de texto** o haga clic en el botón **Insertar archivo de texto** de la barra de herramientas.
2. Haga clic en el botón **Archivo de entrada** y seleccione el archivo `<Documentos>\Altova\MapForce2023\MapForceExamples\Tutorial\Altova-FLF.txt`.



3. Ahora seleccione el botón de opción *De longitud fija*.

4. Desmarque la casilla *Asumir que delimitadores de registro están presentes*.
5. Las tres filas que aparecen en amarillo se pueden editar y en ellas puede especificar: (i) el nombre del campo, (ii) el tipo de datos y (iii) el tamaño del campo. Introduzca ahora el valor **8** para el tamaño del campo y pulse **Entrar**. Ahora pueden verse más datos en la primera columna, que tiene un ancho de 8 caracteres.
6. Haga clic en **Anexar campo** para añadir un campo nuevo y establezca su longitud en 10 caracteres.
7. Use el mismo método para crear tres campos más con una longitud de 3, 25 y 25 caracteres respectivamente. Ahora cambie el nombre de los 5 campos por: First, Last, Tel.-Ext, Email, Title. Esta sería la vista previa:

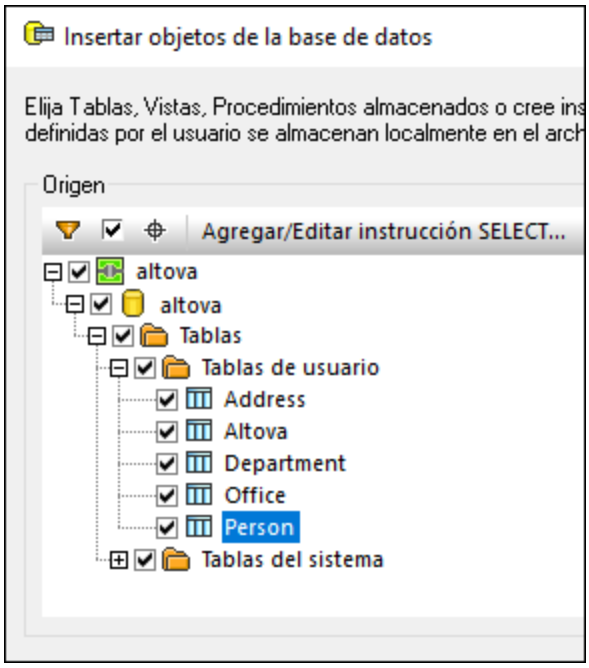
First	Last	Tel.-Ext	Email	Title
string	string	string	string	string
8	10	3	25	25
Vernon	Callaby	582	v.callaby@nanonull.com	Office Manager
Susi	Sanna	753	s.sanna@nanonull.com	Art Director
Fred	Landis	951	f.landis@nanonull.com	Program Manager

8. En el grupo de opciones *Configuración de campo de longitud fija* seleccione la opción *Personalizado* e introduzca una almohadilla (#). Esto le indica a MapForce que debe tratar todos los caracteres # como caracteres de relleno.
9. Haga clic en **Aceptar** para terminar. MapForce emite este aviso:
10. Haga clic en **Cambiar el nombre del componente**. El componente de texto aparecerá en el área de asignación, donde podrá crear conexiones con el componente de destino y otros componentes.

Paso nº2: insertar el componente de base de datos

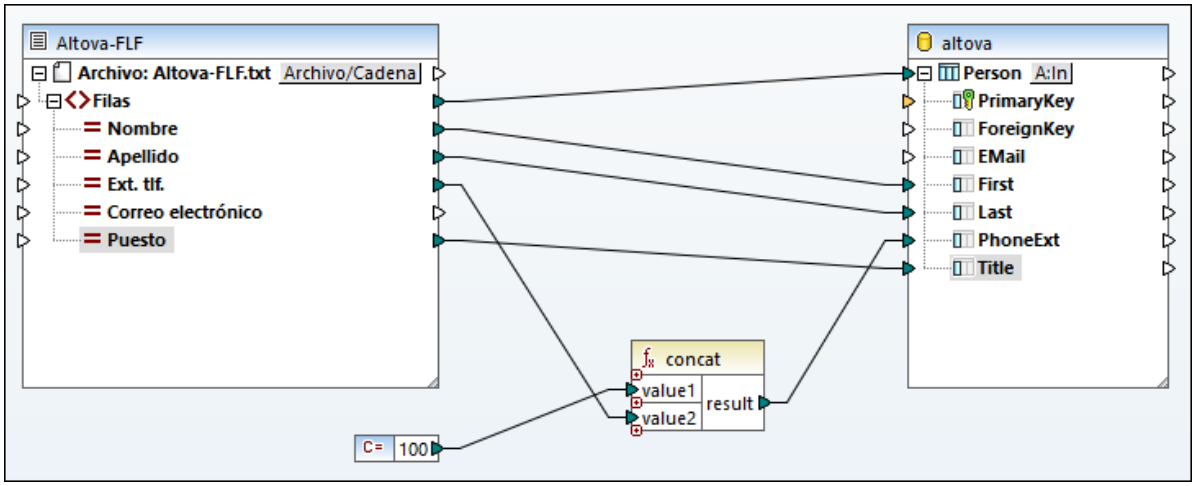
1. Seleccione el comando de menú **Insertar | Base de datos**, seleccione **Microsoft Access** y después haga clic en **Siguiente**.
2. Seleccione la base de datos **altova.mdb** que está en la carpeta `<Documentos>\Altova\MapForce2023\MapForceExamples\Tutorial\` y después haga clic en **Conectarse**.
3. Seleccione la tabla **`Person`** y haga clic en **Aceptar**.

© 2016-2022 Altova GmbH



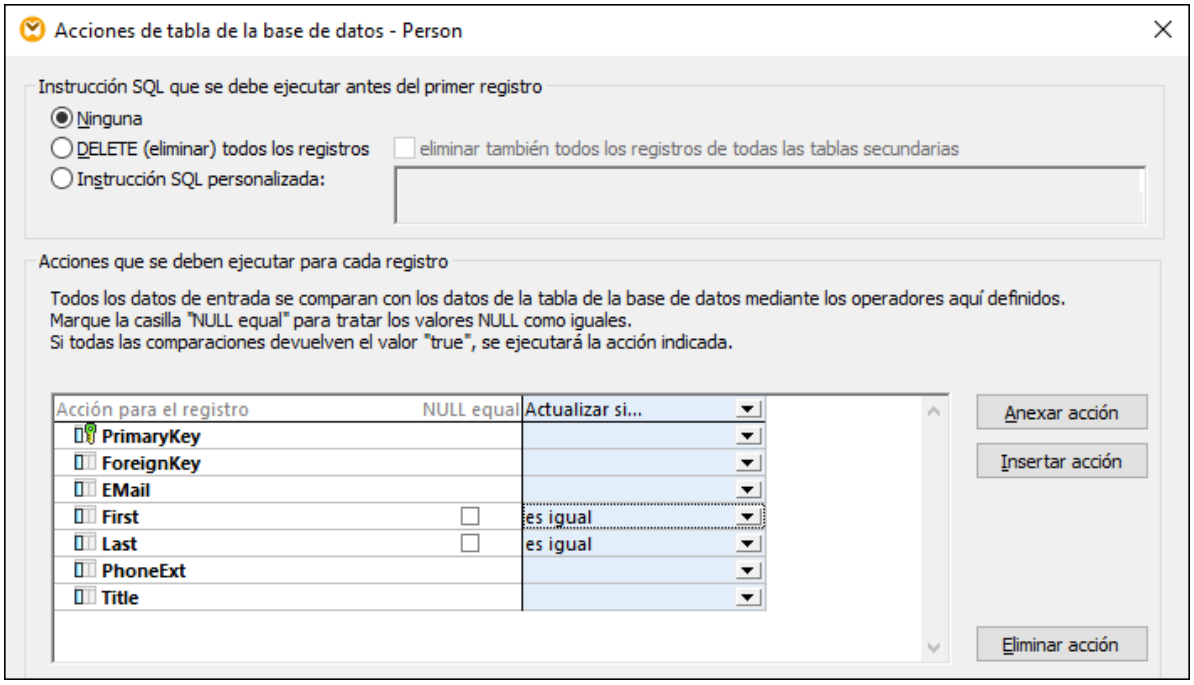
Paso nº3: diseñar la asignación

1. Arrastre la función `core | concat` [708] desde la ventana Bibliotecas hasta el área de asignación.
2. Seleccione el comando de menú **Insertar | Constante**, seleccione el tipo *Número* e introduzca el valor `100`. Esta constante almacena el nuevo prefijo de extensión telefónica.
3. Cree las conexiones que aparecen en esta imagen:



4. En el componente de base de datos haga clic en el botón **Acción de tabla** situado junto al elemento **Person**.
5. En el segundo grupo de acciones (*Acciones que se deben ejecutar para cada registro)* seleccione **Actualizar si...** en el primer cuadro combinado de la segunda columna. Ahora para los campos **First** y **Last** debe seleccionar la acción `es igual`. Esto indica que MapForce solo debe actualizar la tabla **Person** si el nombre y el apellido del archivo de origen es igual al campo correspondiente de la base de datos. Si la condición se cumple, la acción que se debe tomar viene dada por la asignación. En

este caso, la extensión telefónica recibe el prefijo `100` y se copia en el campo **PhoneExt** de la tabla **Person**.



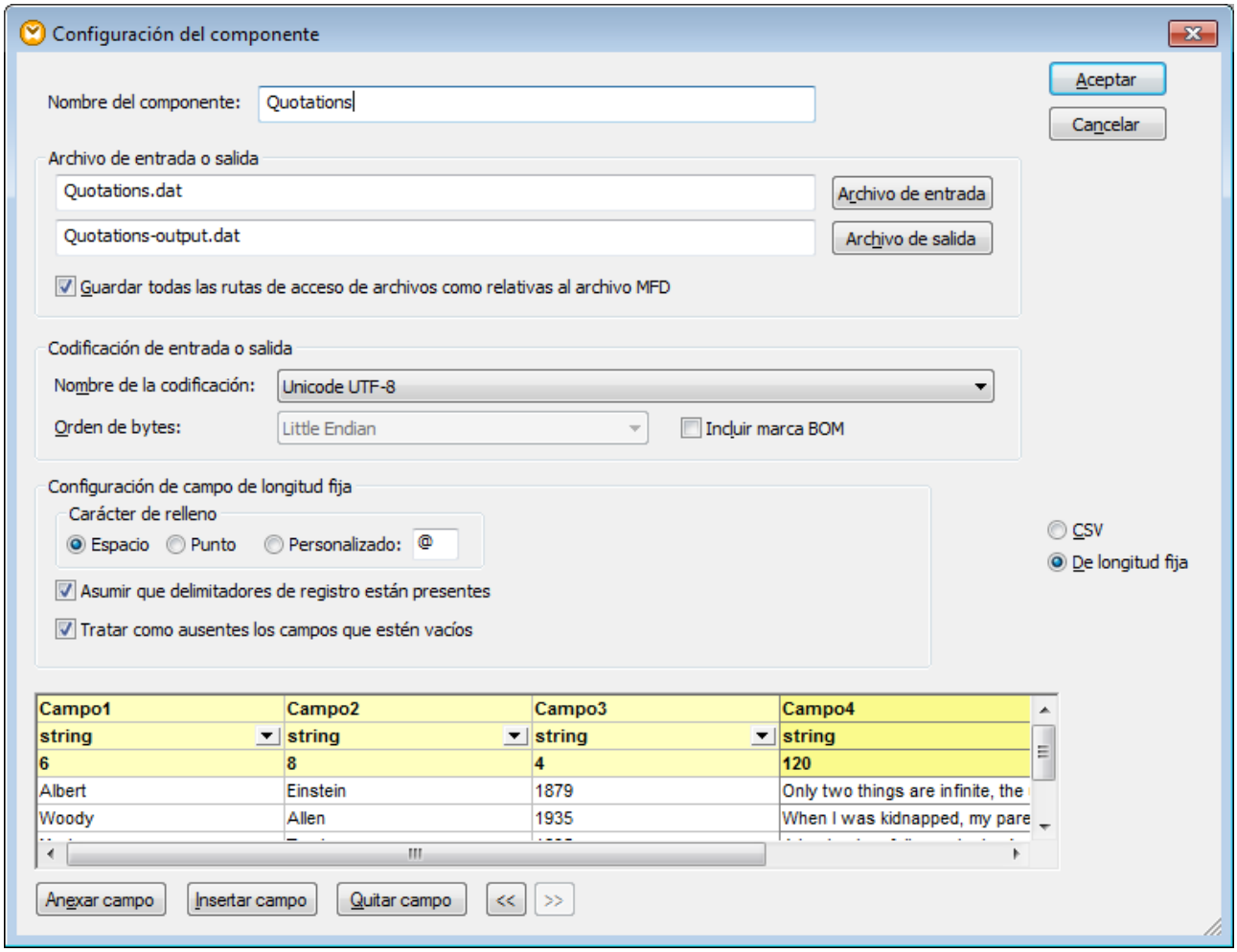
6. Para generar la instrucción SQL (y obtener la vista previa en MapForce) haga clic en el panel *Resultados*. Para ejecutar las instrucciones SQL en la base de datos haga clic en el botón **Ejecutar script SQL** .

4.3.6 Opciones de configuración de componentes FLF

Una vez añadido al área de asignación, el componente de texto se puede configurar desde el cuadro de diálogo "Configuración del componente". Este cuadro de diálogo se puede abrir de varias formas distintas:

- Seleccionando el componente y haciendo clic en el comando de menú **Componente | Propiedades**.
- Haciendo doble clic en el título del componente.
- Haciendo clic con el botón derecho en el título del componente y seleccionando **Propiedades** en el menú contextual.

 © 2016-2022 Altova GmbH



Cuadro de diálogo "Configuración del componente" (en modo de longitud fija)

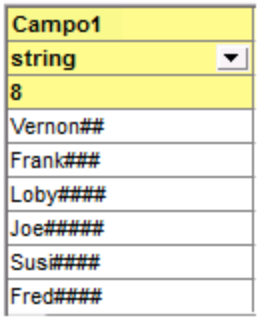
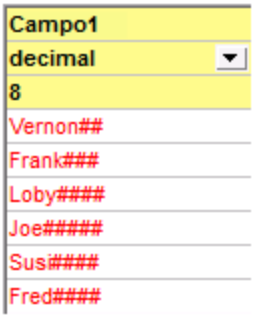
Estas son las opciones de configuración de los componentes FLF:

Nombre del componente	El nombre del componente se genera automáticamente al crearlo, pero se puede volver a cambiar cuando quiera. El nombre del componente puede contener espacios y puntos, pero no barra, barra inversa, dos puntos, comillas dobles, ni espacios antes ni después del nombre. Si cambia el nombre del componente, tenga en cuenta que: • Si quiere implementar la asignación en FlowForce Server, el nombre del componente debe ser único. • Recomendamos que use únicamente caracteres que se puedan usar también en la línea de comandos. Los caracteres especiales propios de cada país pueden estar codificados de forma diferente en Windows y la línea de comandos.

Archivo de entrada	Especifica el archivo donde MapForce debe leer los datos. Este campo es relevante para componentes de origen y se rellena cuando se crea el componente y se le asigna un archivo de texto. El campo puede dejarse vacío si se usa un componente de archivo de texto como destino de la asignación. En un componente de origen MapForce usa el valor de este campo para leer nombres de columna y obtener una vista previa del contenido del archivo de texto. Haga clic en **Archivo de entrada** para seleccionar un archivo nuevo.
Archivo de salida	Especifica el archivo donde MapForce debe escribir datos. Este campo es relevante para componentes de destino. Haga clic en **Archivo de salida** para seleccionar un archivo nuevo.
Guardar todas las rutas de acceso de archivos como relativas al archivo MFD	Si marca esta casilla, MapForce guarda las rutas de acceso de los archivos que aparecen en el cuadro de diálogo "Configuración del componente" como relativas a la ubicación del archivo de diseño de MapForce (.mfd). Esta opción afecta a los archivos de entrada y salida utilizados por el componente de texto (véase también *Usar rutas de acceso relativas en un componente* 82).
Codificación de entrada / salida	Permite configurar estos aspectos del archivo de instancia de salida: • Nombre de la codificación • Orden de bytes • El carácter de la marca de orden de byte (marca BOM). Por defecto todos los componentes nuevos tienen la codificación definida en la opción *Codificación predeterminada para componentes nuevos*. Para acceder a esta opción seleccione **Herramientas | Opciones | Generales**.
Carácter de relleno	Esta opción sirve para determinar qué caracteres deben utilizarse para completar o rellenar el resto del campo (fijo) cuando los datos entrantes son más cortos que las respectivas definiciones del campo. Con la opción *Personalizado* puede definir un carácter de relleno propio. Si los datos entrantes ya contiene un carácter de relleno y ese mismo carácter de relleno se introduce en el campo *Personalizado*, entonces los caracteres de relleno se eliminarán en los datos entrantes.
Asumir que delimitadores de registro están presentes	Esta opción puede ser útil si desea leer datos de un archivo plano de origen que no contiene delimitadores de registro como CR/LF o si desea producir un archivo FLF plano de destino sin delimitadores de registros. Consulte el párrafo Más información sobre la opción 466 *Asumir que delimitadores de registro están presentes* 466 que aparece más adelante.

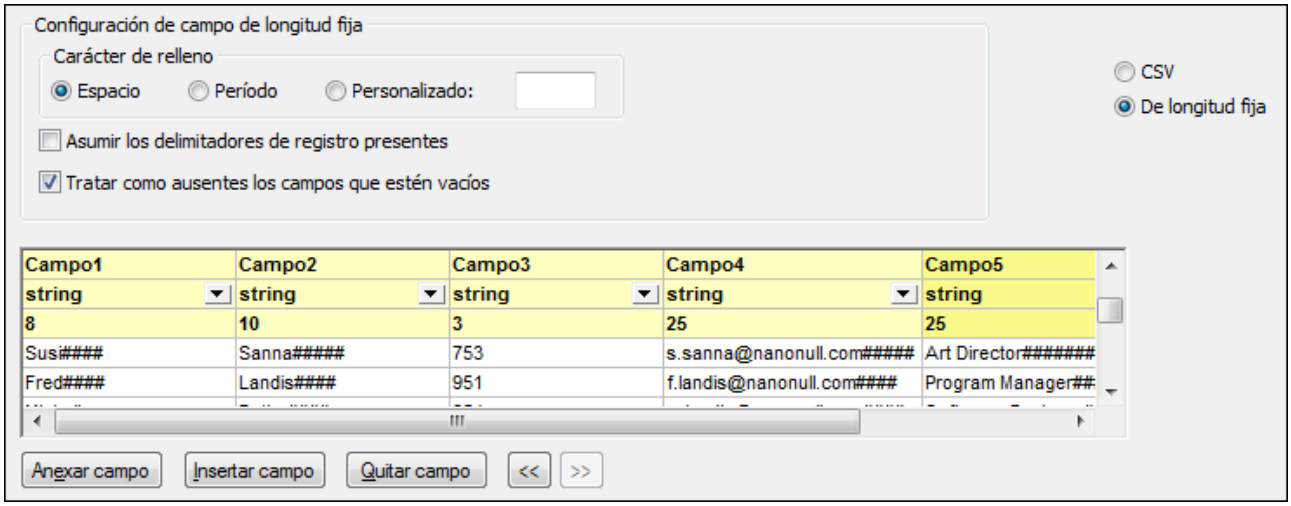
 © 2016-2022 Altova GmbH

Tratar como ausentes los campos que estén vacíos	Si marca esta casilla, los campos vacíos del archivo de origen no producirán un elemento correspondiente vacío (elemento o atributo) en el archivo de destino. Imaginemos que el destino de la asignación es un archivo XML. Si se desactiva esta opción, los campos vacíos se crearán en el archivo de destino con un valor vacío (p. ej. los elementos `Last`, `Title` y `Email`): 33 <Person> 34 <First>General outgassing pollutants</First> 35 <Last/> 36 <Title/> 37 <Email/> 38 </Person> Si se activa esta opción, los campos vacíos no se crearán en el destino de la asignación: 38 <Person> 39 <First>General outgassing pollutants</First> 40 </Person>
CSV / De longitud fija	Cambia el tipo de componente por CSV o FLF (campo de longitud fija).
Panel de vista previa	La parte inferior del cuadro de diálogo muestra una vista previa de hasta 20 dilas del archivo seleccionado como entrada/salida. Si lo necesita, puede crear la estructura del archivo (o cambiarla) con estos botones: **Anexar campo** — Crea un campo nuevo después del último regis **Insertar campo** — Crea un campo nuevo inmediatamente antes d registro que está seleccionado. **Quitar campo** — Elimina el campo que está seleccionado. **<<** — Desplaza el campo que está seleccionado hac izquierda. **>>** — Desplaza el campo que está seleccionado hac derecha. Para cambiar el nombre de un campo haga clic en el encabezado (por ejemplo, **`Campo1`**) y teclee el nuevo valor.

	 Para cambiar el tipo de datos de un campo seleccione el valor pertinente en la lista desplegable. MapForce revisa el tipo de datos y, si los datos de entrada y el formato del campo son discordantes, resalta los datos en rojo.  Para establecer el tamaño del campo en caracteres basta con introducir el tamaño del campo en la tercera fila de la columna.

Más información sobre la opción *Asumir que delimitadores de registro están presentes*

Veamos un ejemplo para comprender mejor el funcionamiento de esta opción. Abra el archivo **Altova-FLF.txt** que está en la carpeta **<Documentos>\Altova\MapForce2023\MapForceExamples\Tutorial**. Observe que el archivo está compuesto por registros con una longitud de 71 caracteres y que no contiene delimitadores. Si tuviéramos que leer datos de este archivo, primero deberíamos dividirlo en registros. Es decir, crear varios campos cuyo tamaño total sea 71 caracteres y después debemos desactivar la casilla *Asumir que delimitadores de registro están presentes*. Para ver un ejemplo con instrucciones paso a paso consulte el apartado Ejemplo: asignar archivos de texto de longitud fija a bases de datos 458.



Si necesita escribir datos de este archivo en un archivo de destino que use la misma estructura y marca la casilla *Asumir que delimitadores de registro están presentes*, MapForce creará un registro nuevo cada 71 caracteres.

```
1   Vernon##Callaby###582v.callaby@nanonull.com###Office Manager###########
2   Frank###Further###471f.further@nanonull.com###Accounts Receivable######
3   Loby####Matise####963l.matise@nanonull.com####Accounting Manager#######
4   Joe#####Firstbread621j.firstbread@nanonull.comMarketing Manager Europe#
5   Susi####Sanna#####753s.sanna@nanonull.com#####Art Director#############
6   Fred####Landis####951f.landis@nanonull.com####Program Manager#########
7   MichelleButler####654m.landis@nanonull.com####Software Engineer########
8   Ted#####Little####852t.little@nanonull.com####Software Engineer########
9   Ann#####Way#######951a.way@nanonull.com#######Technical Writer#########
10  Liz#####Gardner###753l.gardner@nanonull.com###Software Engineer########
11  Paul####Smith#####334p.smith@nanonull.com#####Software Engineer########
12  Alex####Martin####778a.martin@nanonull.com####IT Manager###############
13  George##Hammer####223g.hammer@nanonull.com####Web Developer############
14  Jessica#Bander####241j.band@nanonull.com######Support Engineer#########
15  Lui#####King######345l.king@nanonull.com######Support Engineer#########
16  Steve###Meier#####114s.meier@nanonull.com#####Office Manager###########
17  Theo####Bone######331t.bone@nanonull.com######Accounts Receivable######
18  Max#####Nafta#####122m.nafta@nanonull.com#####PR & Marketing Manager US
19  ValentinBass######716v.bass@nanonull.com######IT Manager###############
20  Carl####Franken###147c.franken@nanonull.com###Support Engineer#########
21  Mark####Redgreen##152m.redgreen@nanonull.com##Support Engineer#########
22
```

El resultado de la asignación cuando está activada la opción Asumir que delimitadores de registro están presentes.

Si no marca la casilla *Asumir que delimitadores de registro están presentes*, entonces el resultado de la asignación será una cadena de gran longitud.

```
1       Vernon##Callaby###582v.callaby@nanonull.com###Office
        Manager###########Frank###Further###471f.further@nanonull.com###Accounts
        Receivable######Loby####Matise####963l.matise@nanonull.com####Accounting
        Manager#######Joe#####Firstbread621j.firstbread@nanonull.comMarketing Manager
        Europe#Susi####Sanna#####753s.sanna@nanonull.com#####Art
        Director#############Fred####Landis####951f.landis@nanonull.com####Program
        Manager##########MichelleButler####654m.landis@nanonull.com####Software
        Engineer########Ted#####Little####852t.little@nanonull.com####Software
        Engineer########Ann#####Way######951a.way@nanonull.com######Technical
        Writer#########Liz#####Gardner###753l.gardner@nanonull.com###Software
        Engineer########Paul####Smith#####334p.smith@nanonull.com#####Software
        Engineer########Alex####Martin####778a.martin@nanonull.com####IT
        Manager###############George##Hammer####223g.hammer@nanonull.com####Web
        Developer############Jessica#Bander####241j.band@nanonull.com######Support
        Engineer#########Lui#####King######345l.king@nanonull.com######Support
        Engineer#########Steve###Meier#####114s.meier@nanonull.com#####Office
        Manager###########Theo####Bone######331t.bone@nanonull.com######Accounts
        Receivable######Max#####Nafta#####122m.nafta@nanonull.com#####PR & Marketing Manager
        USValentinBass######716v.bass@nanonull.com######IT
        Manager###############Carl####Franken###147c.franken@nanonull.com###Support
        Engineer#########Mark####Redgreen##152m.redgreen@nanonull.com##Support
        Engineer#########
```

El resultado de la asignación cuando está desactivada la opción Asumir que delimitadores de registro están presentes.

© 2016-2022 Altova GmbH

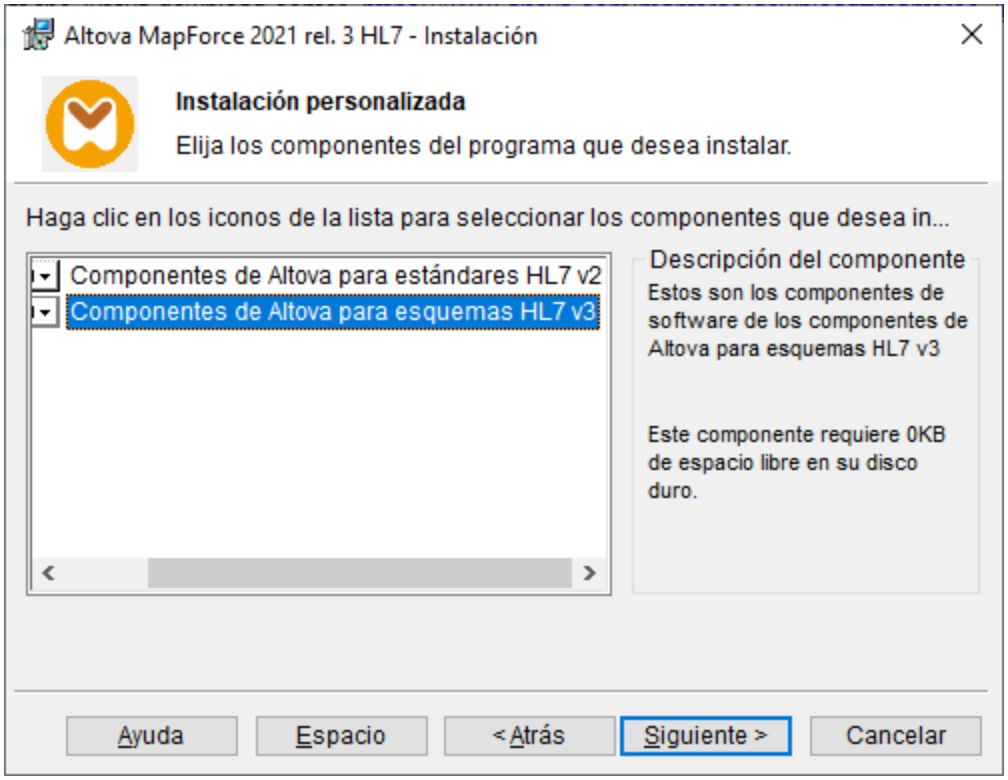
4.4 HL7 versión 3

MapForce permite asignar datos de o a documentos de la versión 3 de HL7. Más concretamente, puede asignar datos de cualquier tipo de componente compatible con su edición de MapForce a archivos de instancia de HL7 3.x y viceversa.

La versión estándar 3 de HL7 se basa en XML, a diferencia de la versión 2. Esto significa que puede añadir componentes de la versión 3 de HL7 a la asignación como añadiría cualquier otro esquema XML. Observe que MapForce no instala ningún esquema de la versión 3 de HL7 por defecto. Sin embargo, sí puede descargar e instalar estos esquemas por separado; para ello siga las instrucciones que se detallan a continuación.

Para instalar esquemas de la versión 3 de HL7:

1. Descargue el paquete "HL7 EDI Configuration Files" de la página "Bibliotecas de MapForce" en el centro de descargas de Altova, https://www.altova.com/mapforce/download#mapforce_libraries. Tenga en cuenta que este paquete incluye tanto los archivos de configuración de la versión 2 de HL7 (que se basan en EDI) como los esquemas XML de la versión 3 de HL7.
2. Haga doble clic en el archivo ejecutable que descargó y siga las instrucciones del asistente de instalación.
3. Cuando la aplicación le pida que seleccione un tipo de configuración, elija **Personalizada** y después haga clic en **Siguiente**.
4. Seleccione que quiere instalar los esquemas de la versión 3 de HL7, por ejemplo:



Después de la instalación, los esquemas XML de la versión 3 de HL7 se pueden encontrar en el directorio: **C:\Program Files\Altova\Common2023\Schemas\hl7v3**.

Nota: si usa una versión de MapForce de 32 bits en un sistema operativo de 64 bits, ajuste la ruta anterior a **C:\Program Files(x86)\Altova\Common2023\Schemas\hl7v3**.

Una vez haya instalado los esquemas de la versión 3 de HL7 puede empezar a diseñar una asignación:

1. Ejecute MapForce.
2. En el menú **Insertar** haga clic en **Archivo o esquema XML**.
3. Navegue hasta el archivo del esquema. Puede seleccionar, por ejemplo, el archivo **PRPA_IN101301UV02.xsd** en el directorio de instalación que mencionamos más arriba.
4. Cuando la aplicación le pida que suministre el archivo de instancia, seleccione el archivo de instancia de la versión 3 de HL7 que corresponde al tipo de mensaje que seleccionó antes, si quiere leer datos del mismo y asignarlo a otro formato. Si lo que pretende es generar el archivo de instancia (y no leer sus datos) haga clic en **Omitir**.

Ha añadido el componente de la versión 3 de HL7 a la asignación y ahora puede trazar las conexiones de la asignación. Para ver una introducción al diseño de asignaciones con MapForce consulte los tutoriales 39.

 © 2016-2022 Altova GmbH

4.5 Archivos binarios

MapForce permite leer datos BLOB (objeto binario grande) a partir de archivos binarios en una asignación y consumirlos sin cambiar la estructura interna de los datos binarios (sin formato). Por ejemplo, puede guardar datos binarios en un campo BLOB de BD, en un campo de tipo `xs:base64Binary` de un archivo XML o enviarlo a un servicio web*.

** Las llamadas a servicios web solo se pueden realizar con MapForce Enterprise Edition.*

También puede crear asignaciones que lean datos binarios desde un origen (como un campo BLOB de BD, un campo de tipo `xs:base64Binary` de un archivo XML o un servicio web) y después escriban datos binarios en disco.

Estos son solo algunos ejemplos de los casos en los que puede venir bien poder leer o escribir archivos binarios:

- Extraer contenido binario cifrado en base 64 de un archivo XML y guardarlo en disco (por ejemplo, como archivo PDF)
- Procesar archivos de imagen almacenados en disco y enviarlos como contenido binario cifrado en base 64 a un servicio web
- Extraer contenido BLOB de una tabla de BD y guardarlo en disco como archivos de imagen (uno por cada fila de la tabla de BD)
- Leer archivos de imagen desde el disco y guardarlos en una tabla de BD como campos de datos BLOB.

Nota: para asignar datos a o desde archivos binarios necesita el lenguaje de transformación BUILT-IN 24. Puede acceder a la vista previa de la asignación en MapForce (y guardar los archivos de salida, si los hay) o ejecutarla con MapForce Server (requiere una licencia propia) en un equipo o una plataforma distintos. No se pueden generar ejecutables en C#, C++ o Java a partir de asignaciones que leen o escriben en archivos binarios.

Añadir archivos binarios a la asignación

En MapForce no existe ningún tipo de componente asociado a archivos binarios, como sí ocurre, por ejemplo, con los archivos XML, JSON o de texto. Sin embargo, para lograr los objetivos de los ejemplos del punto anterior puede usar estas funciones integradas de MapForce:

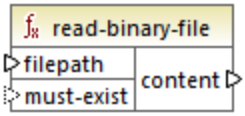
- read-binary-file 744
- write-binary-file 745

Puede encontrar estas funciones en la ventana Bibliotecas (más concretamente, en la biblioteca "lang"). Para usar alguna de estas funciones en una asignación basta con arrastrarlas desde la ventana Biblioteca hasta la asignación o hacer doble clic dentro de una zona vacía de la asignación y empezar a escribir el nombre de la función para después seleccionarlo. Para más información consulte Agregar una función integrada a una asignación de datos 550.s

read-binary-file

Esta función devuelve el contenido del archivo indicado como BLOB (objeto binario grande por sus siglas en inglés) de tipo `xs:base64Binary`. Tenga en cuenta que aunque el tipo de datos se llama "base64Binary", la representación interna es únicamente un BLOB. La codificación en base 64 no tiene lugar hasta que no asigna

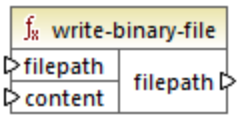
el resultado de la función a un nodo XML de tipo `xs:base64Binary`. También puede asignar el resultado de la función a `xs:hexBinary`, a un blob de BD o a un campo binario de una estructura de Protocol Buffers.



- Para leer un archivo binario en una asignación debe dar su ruta como entrada del argumento **filepath**. Si **filepath** es relativo, entonces MapForce buscará el archivo en el mismo directorio que la asignación. El argumento **must-exist** es opcional; si no se puede abrir el archivo y su parámetro es **true**, la asignación emite un error. Si no se puede abrir el archivo y su parámetro es **false**, la asignación devuelve un binario vacío.

write-binary-file

Esta función escribe contenido binario en la ruta de archivo indicada y devuelve la ruta del archivo escrito. Si quiere que el resultado sera únicamente un archivo binario, conecte el resultado de la función a un componente de salida simple 489. Esta función escribe un archivo cada vez que se usa su resultado en la asignación, por lo que recomendamos que conecte el resultado de la función directamente a un componente de destino, sin usar otros procesamientos intermedios.



Para escribir archivos binarios debe dar su ruta como entrada del argumento **filepath**. Si **filepath** es relativo, entonces MapForce genera el archivo en el mismo directorio que la asignación. El argumento **content** debe estar conectado al contenido binario (por ejemplo, un campo BLOB de una BD).

Si accede a la vista previa de la asignación en MapForce, la función genera archivos temporales por defecto, en lugar de escribir archivos directamente en disco. Para guardar estos archivos temporales en disco haga clic en la pestaña *Resultados* y después haga clic en los botones de la barra de herramientas **Guardar resultado generado** o **Guardar todos los resultados generados**.

Para configurar MapForce para que escriba el resultado directamente en archivos finales en vez de temporales seleccione el comando de menú **Herramientas | Opciones**, haga clic en **Generales** y después marque la opción **Escribir directamente en archivos de salida finales**. Tenga en cuenta que esta opción sobrescribe cualquier otro archivo que tenga el mismo nombre.

La función siempre devuelve el nombre de archivo final (no el temporal), incluso aunque no haya guardado aún el archivo final en disco (que es el caso cuando accede a la vista previa de la asignación y se deshabilita la opción **Escribir directamente en archivos de salida finales**).

> Tenga en cuenta que una asignación no puede volver a leer su propio archivo de salida.

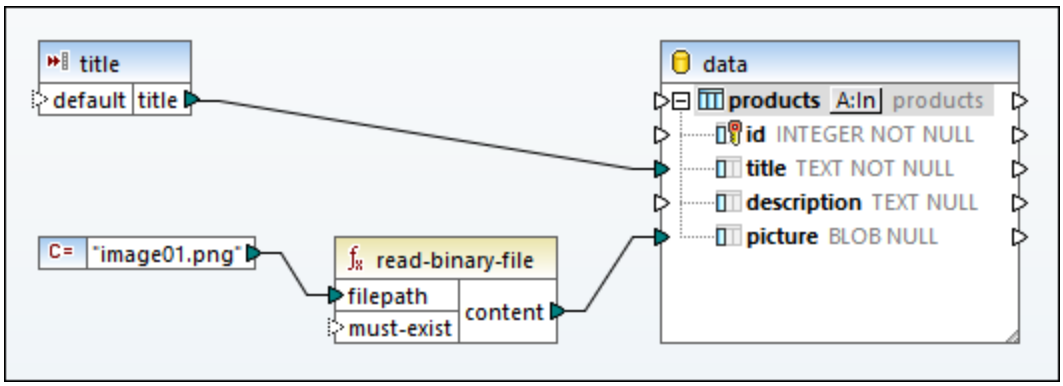
Ejemplos

Consulte Ejemplo: leer archivos binarios 473 y Ejemplo: escribir archivos binarios 474.

 © 2016-2022 Altova GmbH

4.5.1 Ejemplo: leer archivos binarios

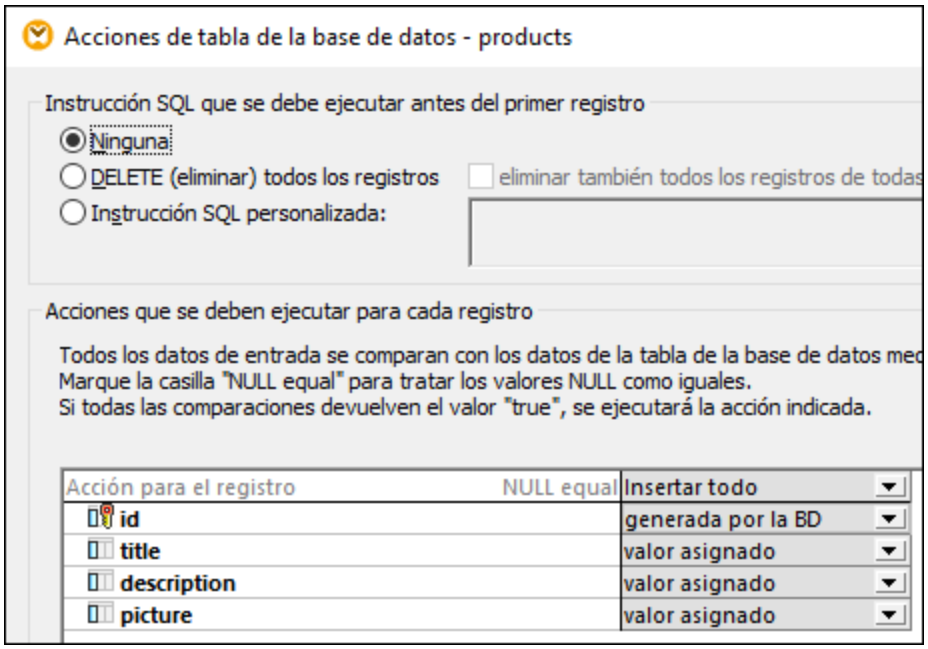
La asignación siguiente lee datos desde un archivo de imagen y los escribe en una tabla de BD. La BD de destino es SQLite. Observe que el tipo de datos del campo de BD **picture** es BLOB.



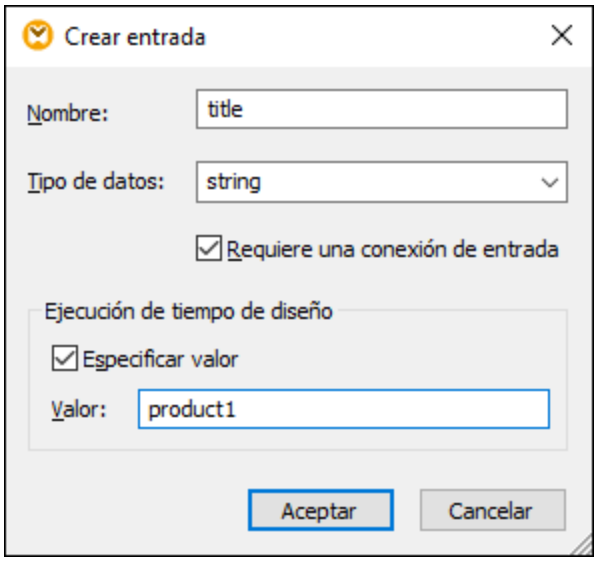
En esta asignación se usó la función `read-binary-file` para extraer contenido binario del archivo. El primer argumento, **filepath**, viene dado por una constante. Tenga en cuenta que, al ser una ruta relativa, MapForce buscará el archivo de imagen en el mismo directorio que la asignación.

La asignación rellena los campos siguientes en la BD de destino:

- **id**: en este ejemplo el componente de BD está configurado de tal manera que **id** es generado por la BD en lugar de por la asignación. Para más información consulte Insertar datos en una tabla 282.



- **title**: este valor viene dado por un componente de entrada simple con el mismo nombre. Observe que al diseñar la asignación se ha definido un valor de ejecución ("product1") para que sea posible acceder a una vista previa de la asignación. Para más información consulte Pasar parámetros a la asignación 477.



- **picture**: este campo recibe el resultado directo de la función `read-binary-file`.

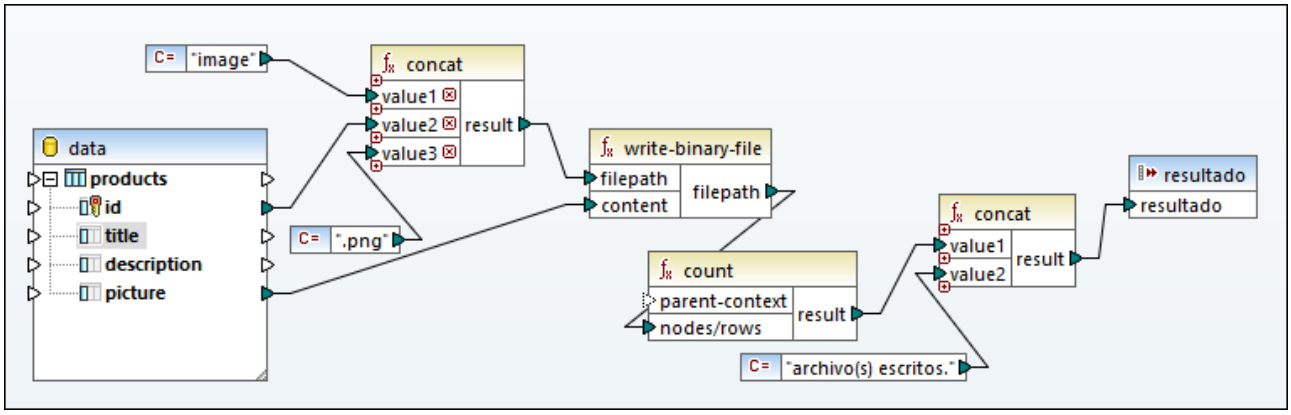
El componente de destino es una BD, por lo que la vista previa de la asignación genera una especie de script SQL que puede revisar, pero no envía cambios a la BD. Para ejecutar el script en la BD debe seleccionar el comando de menú **Resultados | Ejecutar script SQL**.

4.5.2 Ejemplo: escribir archivos binarios

La asignación siguiente lee valores BLOB de una BD SQLite y escribe archivos de imagen en disco. En la BD hay una tabla llamada **products** que tiene estas columnas (o campos):

- **id** (*entero*, el número serial único permanente del registro)
- **title** (*text*, el título del producto)
- **description** (*text*, la descripción del producto)
- **picture** (*blob*, la imagen del producto)

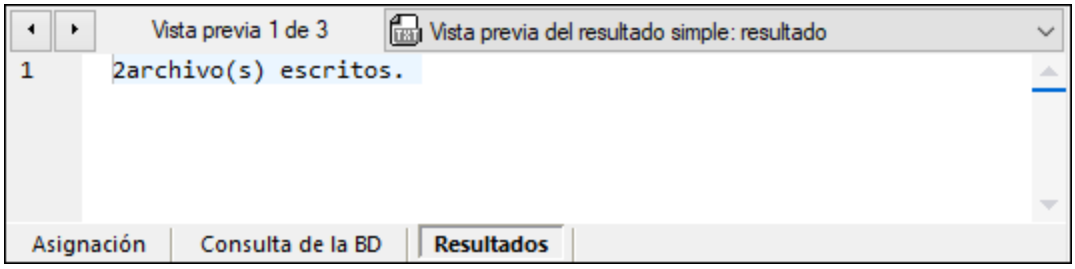
En este ejemplo solo son relevantes los campos **id** y **picture**.



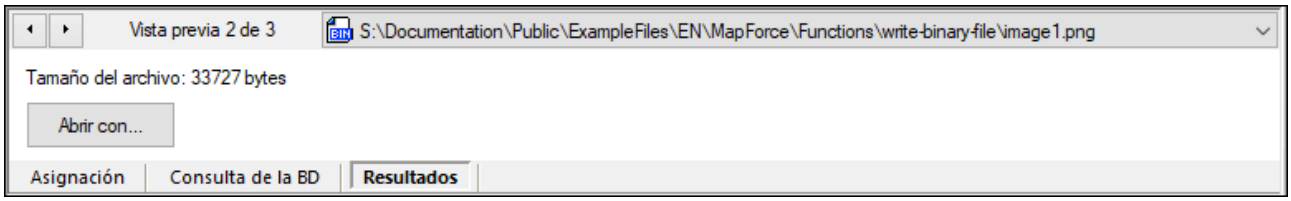
El objetivo de la asignación es extraer todas las imágenes de la tabla **products** y escribirlos en disco como archivos. Como se ve en la imagen, para ello se usa la función `write-binary-file`. El primer argumento, **filepath**, recibe la ruta de archivo de cada imagen. La ruta debe ser única para evitar que sobrescriba otros archivos, por lo que se genera concatenando el **id** de BD único de cada registro con la palabra `image` y la extensión de archivo `.png` (asumiendo que todas las imágenes estén en ese formato). Tenga en cuenta que la ruta es relativa, por lo que los archivos se escriben en el mismo directorio que la asignación.

El segundo argumento, **content**, recibe el contenido binario almacenado en la BD.

La función `count` devuelve el recuento de todos los archivos generados y combina este número con la cadena `archivo(s) escritos`. Así puede saber cuántos archivos ha generado la asignación. En este ejemplo la BD solo contiene dos registros de producto, que es lo que refleja el resultado:



Como hemos explicado anteriormente, esta función genera archivos temporales cuando se ejecuta la asignación para acceder a una vista previa del resultado. Para visualizar cada uno de los archivos, use los botones de flecha para navegar el registro relevante, haga clic en el botón **Abrir con** y seleccione un editor de imagen.



Para guardar todos los archivos haga clic en los botones de la barra de herramientas **Guardar resultado generado** o **Guardar todos los resultados generados**, según el caso.

5 Componentes de transformación

En esta sección explicamos los distintos componentes que se pueden usar para transformar datos o para almacenarlos de forma temporal para seguir procesándolos más adelante. A continuación puede ver una lista de los componentes de transformación:

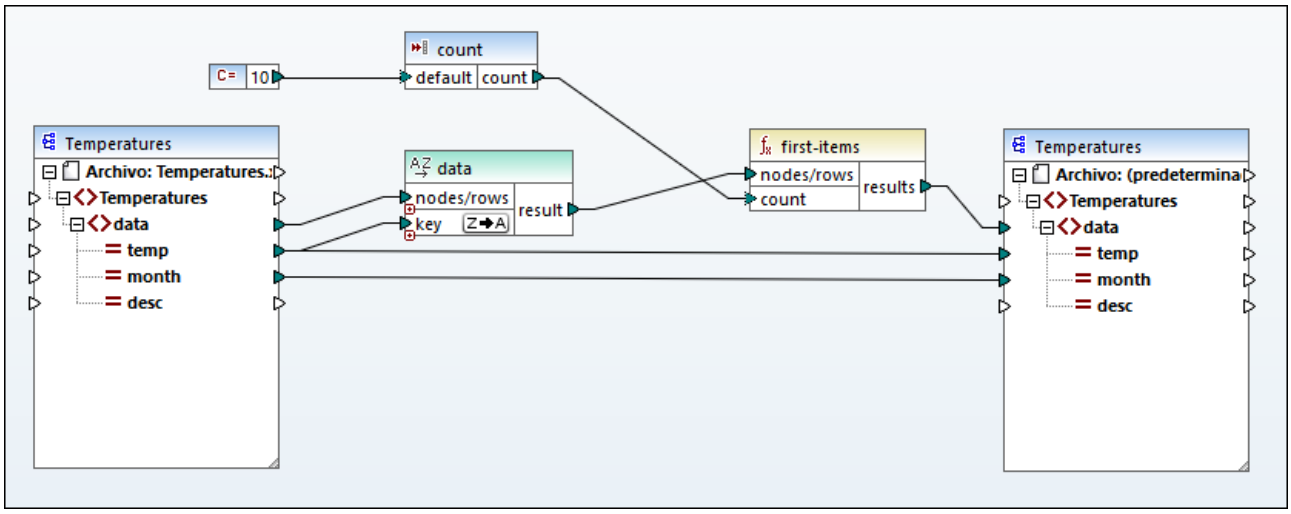
- Entrada simple 477
- Salida simple 488
- Variables 492
- Ordenar datos 505
- Agrupar datos 511
- Combinar datos 517
- Excepciones 529
- Asignaciones de valores 533
- Filtros y condiciones 543

Tenga en cuenta que las funciones también pertenecen a los componentes de transformación. Sin embargo, hemos agrupado esas funciones 549 en una sección propia.

© 2016-2022 Altova GmbH

5.1 Pasar parámetros a la asignación

Si tiene que crear una asignación que tome parámetros como componentes de entrada puede hacerlo añadiendo un componente especial llamado "componente de entrada simple". Este tipo de componente siempre tiene un tipo de datos simple (por ejemplo, cadena, número entero, etc.) en lugar de una estructura de elementos y secuencias. Por ejemplo, en la asignación siguiente encontramos el componente de entrada simple **count**, que sirve para dar como parámetro el número máximo de filas que debe recuperar el archivo XML de entrada (cuyo valor predeterminado es **10**). Es importante recordar que los nodos que se indican como componente de entrada a la función first-items 685 se ordenan con ayuda de un componente de ordenación, por lo que la asignación da como resultado solamente las *N* temperaturas más altas, donde *N* es el valor del parámetro.



FindHighestTemperatures.mfd

Otro uso bastante común de los componentes de entrada de tipo simple es dar un nombre de archivo a la asignación. Esto puede ser útil para asignaciones que lean los archivos de entrada o escriban los archivos de salida de forma dinámica (véase Procesar varios archivos de entrada o salida simultáneamente 892).

Puede usar componentes de entrada simples con cualquier lenguaje de transformación de MapForce:

- BUILT-IN (desde el panel *Resultados* para obtener la vista previa de la transformación de la asignación en MapForce directamente)
- BUILT-IN (cuando se ejecuta un archivo de ejecución de MapForce Server compilado)
- XSLT 1.0, XSLT 2.0, XSLT 3.0
- XQuery
- C++
- C#
- Java

Cuando se trate de asignaciones ejecutadas con MapForce Server o por medio de código generado, los componentes de entrada simples se convierten en parámetros de la línea de comandos. Cuando se trate de asignaciones generadas como transformaciones XSLT, los componentes de entrada simples corresponden a parámetros de hojas de estilos en el archivo XSLT que se genera.

Cada componente de entrada simple (o parámetro) se puede crear como componente opcional u obligatorio (*véase* *Configurar componentes de entrada simples* 479). Si fuera necesario, también podrá crear valores predeterminados para los parámetros de entrada de la asignación (*véase* *Crear un valor de entrada predeterminado* 481). Esto le permitirá ejecutar la asignación de forma segura aunque no aporte explícitamente un valor de parámetro en tiempo de ejecución de la asignación.

Los parámetros de entrada añadidos en el área de asignación principal no se deben confundir con los parámetros de entrada de las funciones definidas por el usuario (*véase* *Funciones definidas por el usuario* 575). En esta tabla puede ver en qué se parecen y en qué se diferencian:

Parámetros de entrada de la asignación	**Parámetros de entrada de funciones definidas por el usuario**
Se añaden con el comando **Función | Insertar componente de entrada**.	Se añaden con el comando **Función | Insertar componente de entrada**.
Pueden tener tipos de datos simples (cadena, entero, etc.).	Puede tener tipos de datos simples y complejos.
Afectan a toda la asignación.	Afectan solo al contexto de la función donde se definen.

Cuando cree una asignación invertida (con el comando de menú **Herramientas | Crear asignación inversa**), los componentes de entrada simples se convertirán en componentes de salida simples.

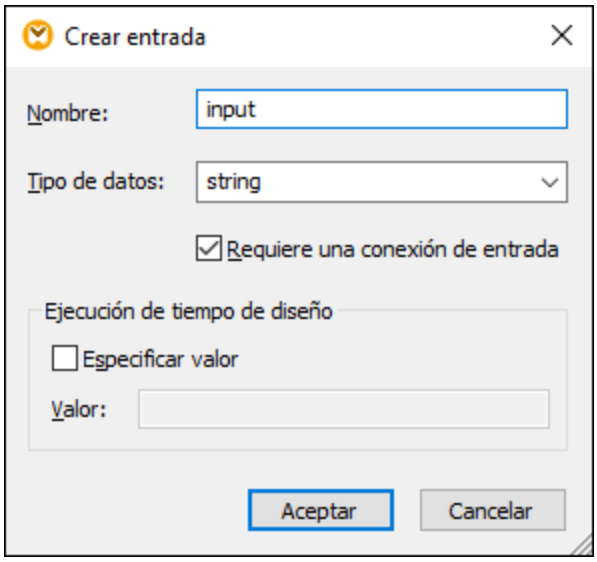
5.1.1 Agregar componentes de entrada simples

Para agregar un componente de entrada simple a la asignación:

1. Asegúrese de que en la ventana de asignación está activa la asignación principal (y no una función definida por el usuario).
2. Elija una opción:

   - En el menú **Función** haga clic en **Insertar componente de entrada**.
   - En el menú **Insertar** haga clic en **Insertar componente de entrada**.
   - Haga clic en el botón de la barra de herramientas **Insertar componente de entrada**.

 © 2016-2022 Altova GmbH



3. Introduzca un nombre y seleccione el tipo de datos que requiere esta entrada. Si la entrada debe tratarse como un parámetro obligatorio de la asignación, marque la casilla Requiere una conexión de entrada. (Para más información sobre las demás opciones de configuración consulte el apartado Configurar componentes de entrada simples 479.)

Nota: el nombre del parámetro solo puede contener letras, cifras y barra baja; el resto de caracteres no están permitidos. Esto permite que las asignaciones funcionen con todos los lenguajes de generación de código.

4. Haga clic en **Aceptar**.

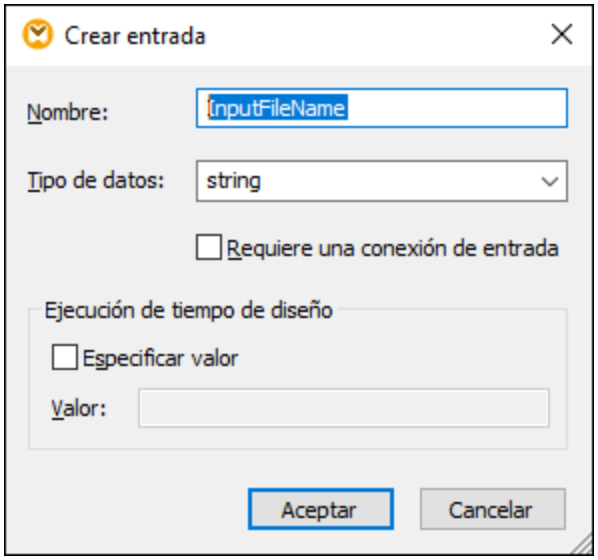
Las demás opciones de configuración se describen en el apartado Configurar componentes de entrada simples 479.

5.1.2 Configurar componentes de entrada simples

Los componentes de entrada simples se pueden configurar en el momento en el que se añaden al área de asignación o más tarde. Esto se hace en el cuadro de diálogo "Crear entrada".

Para abrir el cuadro de diálogo "Crear entrada" tiene tres opciones:

- Seleccione el componente y seleccione el comando de menú **Componente | Propiedades**.
- Haga doble clic en el componente.
- Haga clic con el botón derecho en el componente y después elija **Propiedades** en el menú contextual.



Cuadro de diálogo "Crear entrada"

Estas son las opciones de configuración que ofrece el cuadro de diálogo:

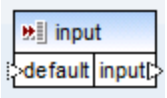
Nombre	Introduzca un nombre descriptivo para el parámetro de entrada que corresponde a este componente. En tiempo de ejecución el valor que se introdujo en este campo pasa a ser el nombre del parámetro que se pasa a la asignación. Por tanto, no está permitido el uso de espacios ni caracteres especiales.
Tipo de datos	Todos los parámetros de entrada se tratan por defecto como tipo de datos de cadena. Si el parámetro debe tener otro tipo de datos, seleccione el valor pertinente en la lista. Cuando la asignación se ejecute, MapForce convertirá el tipo del parámetro de entrada en el tipo de datos que se seleccionó aquí.
Requiere una conexión de entrada	Si marca esta casilla, el parámetro de entrada será obligatorio (es decir, la asignación no se podrá ejecutar a no ser que se aporte un valor de parámetro). Desactive esta casilla si desea especificar un valor predeterminado para el parámetro de entrada (*véase* *Crear un valor de entrada predeterminado* 481).
Especificar valor	Esta casilla solo es relevante cuando se ejecuta la asignación en tiempo de diseño (cuando se hace clic en el panel *Resultados*). Permite introducir directamente en el componente el valor que se debe usar como entrada de la asignación.
Valor	Este campo solo es relevante cuando se ejecuta la asignación en tiempo de diseño (cuando se hace clic en el panel *Resultados*). Para introducir el valor que debe usar MapForce como entrada de la asignación, marque la casilla *Especificar valor* y después escriba el valor en este campo. **Nota:** si marca la casilla *Especificar valor* e introduce un valor en el campo adyacente, ese valor tendrá preferencia frente al valor predeterminado en la vista previa de la asignación (es decir, en el momento de diseñarla). Sin embargo, este valor no afecta al código XSLT, XQuery o de programa, a la

 © 2016-2022 Altova GmbH

	ejecución de MapForce Server o a la implementación en FlowForce Server.

5.1.3 Crear un valor de entrada predeterminado

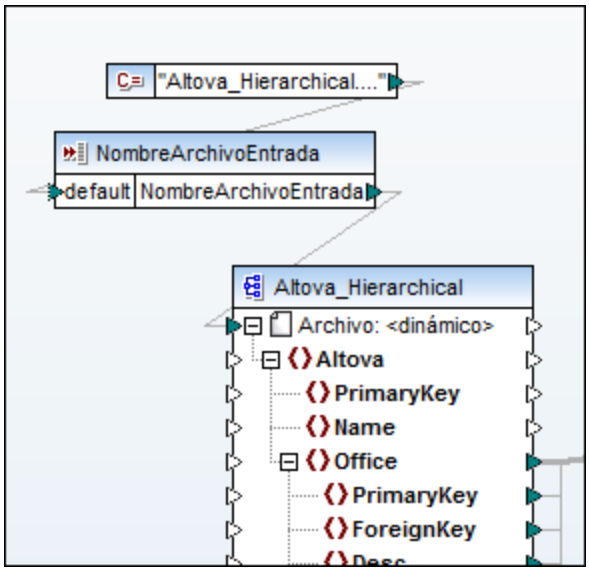
Tras añadir el componente de entrada al área de asignación, debemos fijarnos en el elemento **default** situado en el lado izquierdo del componente.



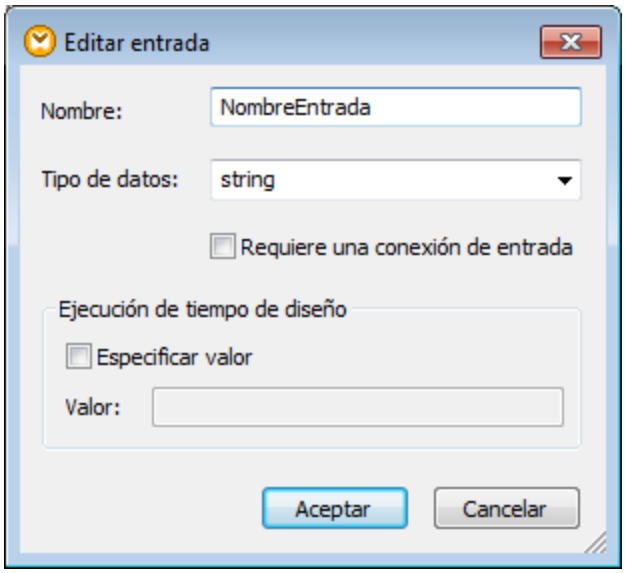
Componente de entrada simples

El elemento **default** permite conectar un valor predeterminado opcional a este componente de entrada. Esto se hace de la siguiente manera:

1. Añada un componente de constante (con el comando de menú **Insertar | Constante**) y después conéctelo con el elemento **default** del componente de entrada.



2. Haga doble clic en el componente de entrada y compruebe que la casilla *Requiere una conexión de entrada* está desactivada. Cuando se crea un valor de entrada predeterminado, esta casilla no es relevante y causa advertencias de validación.



3. Haga clic en **Aceptar** para terminar.

Nota: si marca la casilla *Especificar valor* e introduce un valor en el campo adyacente, ese valor tendrá preferencia frente al valor predeterminado en la vista previa de la asignación (es decir, en el momento de diseñarla). Sin embargo, este valor no afecta al código XSLT, XQuery o de programa, a la ejecución de MapForce Server o a la implementación en FlowForce Server.

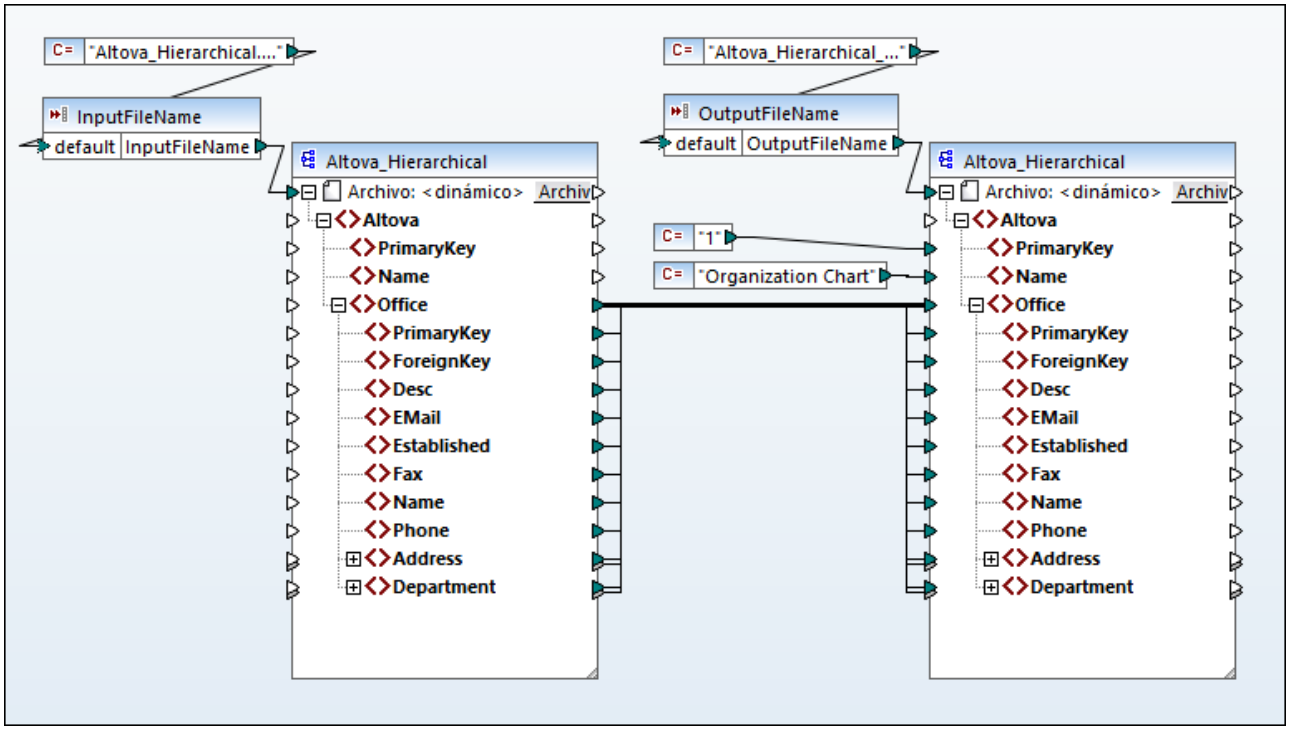
5.1.4 Ejemplo: usar nombres de archivo como parámetros de asignación

En este ejemplo se explica paso a paso cómo ejecutar una asignación que toma parámetros de entrada en tiempo de ejecución. El archivo de diseño de asignación que utilizamos en este ejemplo está en `<Documentos>\Altova\MapForce2023\MapForceExamples\FileNamesAsParameters.mfd`.

Esta asignación lee datos de un archivo XML de origen y los escribe en un archivo XML de destino. Los datos se escriben en el archivo de destino prácticamente sin cambios (los atributos `PrimaryKey` y `Name` se rellenan con valores constantes de la asignación). El objetivo principal de la asignación es permitir que se pueda indicar el nombre de los archivos de entrada y salida como parámetros al ejecutar la asignación.

La asignación utiliza dos componentes de entrada: **InputFileName** y **OutputFileName**. Estos componentes aportan el nombre de archivo de entrada (y el nombre de archivo de salida respectivamente) del archivo XML de origen y de destino. Por este motivo, están conectados con el nodo `Archivo: <dinámico>`. Puede cambiar un componente a este modo haciendo clic en el botón **Archivo/Cadena** (Archivo/Cadena) y seleccionando **Usar nombres de archivo dinámicos dados por la asignación**.

 © 2016-2022 Altova GmbH



FileNamesAsParameters.mfd (MapForce Basic Edition)

Si hace doble clic en la barra del título de uno de los componentes (**InputFileName** u **OutputFileName**), puede ver o editar sus propiedades. Por ejemplo, puede indicar el tipo de datos del parámetro de entrada o cambiar el nombre del parámetro de entrada, como se explica en Configurar componentes de entrada simples 479. En este ejemplo los parámetros de entrada y salida se han configurado así:

- El parámetro **InputFileName** es de tipo "cadena" y tiene un valor predeterminado dado por una constante definida en la misma asignación. Esa constante es de tipo "cadena" y su valor es "Altova_Hierarchical.xml". Por tanto, al ejecutar la asignación, esta intentará leer datos de un archivo llamado "Altova_Hierarchical.xml", siempre que no indique otro valor como parámetro.
- El parámetro **OutputFileName** es de tipo "cadena" y su valor predeterminado también viene dado por una constante definida en la misma asignación. Esa constante también es de tipo "cadena" y se llama "Altova_Hierarchical_output.xml", por lo que al ejecutar la asignación esta genera un archivo XML de salida llamado "Altova_Hierarchical_output.xml", siempre que no indique otro valor como parámetro.

Ambos componentes (**InputFileName** y **OutputFileName**) son componentes de entrada simples, así que podrá suministrarlos como parámetros de entrada a la hora de ejecutar la asignación. A continuación explicamos cómo hacerlo en estos lenguajes de transformación:

- XSLT 2.0 484, con ayuda de RaptorXML Server
- Built-in (archivo de ejecución de MapForce Server) 484, con ayuda de MapForce Server
- Java 485
- C# 486
- C++ 486

XSLT 2.0

Si genera código en XSLT 1.0, XSLT 2.0 o XSLT 3.0, los parámetros de entrada se escriben en el archivo por lotes `DoTransform.bat` que se ejecutará con RaptorXML Server (*véase* *Automatización con RaptorXML Server* 947). Para usar otro archivo de entrada (o de salida), puede pasar los parámetros necesarios en la línea de comandos cuando llame al archivo `DoTransform.bat` o editar este último para incluir los parámetros necesarios.

Siga estos pasos para suministrar un parámetro de entrada personalizado al archivo `DoTransform.bat`:

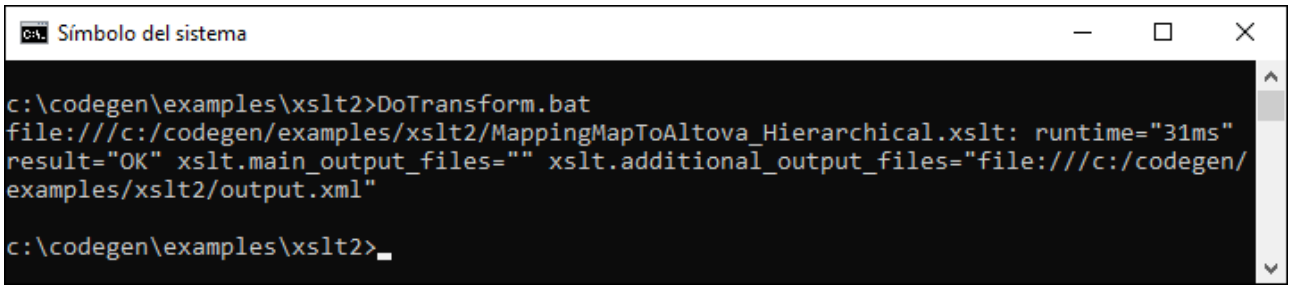
1. Primero genere código XSLT. Por ejemplo, genere 2.0 (**Archivo | Generar código en | XSLT 2.0**) a partir del diseño de asignación `FileNamesAsParameters.mfd`.

2. Copie el archivo `Altova_Hierarchical.xml` del directorio `<Documentos>\Altova\MapForce2023\MapForceExamples\` al directorio donde generó el código XSLT 2.0 (en este ejemplo usamos el directorio de salida `c:\codegen\examples\xslt2\`). Este archivo hará las veces de parámetro personalizado.

3. Edite el archivo por lotes `DoTransform.bat` para incluir el parámetro de entrada personalizado antes o después de `%*` (ver texto resaltado más abajo). Observe que el valor de parámetro va entre comillas simples. Los parámetros de entrada disponibles se enumeran en la sección **rem** (Remark).

```
@echo off

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

Cuando ejecute el archivo por lotes `DoTransform.bat`, RaptorXML Server finaliza la transformación usando `Altova_Hierarchical.xml` como parámetro de entrada.



Archivo de ejecución de MapForce Server

Siga estos pasos para suministrar un parámetro de entrada personalizado a un archivo de ejecución de MapForce Server:

1. Compile el diseño de asignación `FileNamesAsParameters.mfd` en un archivo de ejecución de MapForce Server (*véase* *Compilar asignaciones en archivos de ejecución de MapForce Server* 955).

 © 2016-2022 Altova GmbH

Cuando llegue el momento, guarde el archivo de ejecución `.mfx` en un directorio del equipo (en este ejemplo usamos `c:\codegen\examples\mfx\`).

2. Copie el archivo `Altova_Hierarchical.xml` del directorio `<Documentos>\Altova\MapForce2023\MapForceExamples\` al directorio donde guardó el archivo de ejecución `.mfx`. Este archivo hará las veces de parámetro personalizado para el archivo de ejecución de la asignación.

3. Ejecute MapForce Server con este comando:

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

Observe que, en este comando de MapForce Server, -p=InputFileName y -p=OutputFileName son los parámetros de entrada para la asignación. Como valor de OutputFileName puede usar cualquier nombre de archivo. Sin embargo, el nombre de archivo dado en InputFileName debe existir como archivo físico para que la ejecución pueda realizarse correctamente.

Nota: si recibe el mensaje "MapForceServer.exe no se reconoce como un comando interno o externo, programa o archivo por lotes ejecutable", cambie el directorio actual por el directorio donde está instalado el ejecutable de MapForce Server. Si no quiere tener que cambiar de directorio cada vez que ejecute una asignación, basta con añadir la ruta de acceso del directorio donde está instalado MapForce Server a la variable de entorno PATH del sistema operativo (p. ej. `C:\Archivos de programa (x86)\Altova\MapForceServer2023\bin`) .

Con MapForce Server también puede ejecutar asignaciones llamando a la API del servidor (a la que puede invocar con lenguajes como C++, C#, o Java. Para más información sobre este procedimiento consulte la documentación de MapForce Server (https://www.altova.com/documentation).

Java

Siga estos pasos para suministrar un parámetro de entrada personalizado a una aplicación Java .jar:

1. Genere código Java (**Archivo | Generar código en | Java**) a partir del diseño de asignación `FileNamesAsParameters.mfd`.
2. Compile el código Java en un archivo ejecutable JAR (encontrará instrucciones en este apartado: Ejemplo: generar una aplicación Java con Eclipse y Ant 1089).
3. Copie el archivo `Altova_Hierarchical.xml` del directorio `<Documentos>\Altova\MapForce2023\MapForceExamples\` al directorio donde está el archivo `.jar`. Este archivo hará las veces de parámetro personalizado para la aplicación de asignación Java.
4. En la línea de comandos introduzca `java -jar Mapping.jar /InputFileName "InputFile.xml"`

> Si usa comodines a la hora de pasar parámetros a archivos .jar, debe poner los parámetros comodín entre comillas. Por ejemplo:
>
> `java -jar Mapping.jar /InputFileName "altova-*.xml"`

C#

Para indicar un parámetro de entrada personal a la aplicación línea de comandos en C# generada por MapForce:

1. Abra el ejemplo **FileNamesAsParameters.mfd** que encontrará en el directorio **<Documentos>\Altova\MapForce2023\MapForceExamples**.
2. En el menú **Archivo** haga clic en **Generar código en | C#** y seleccione un directorio de destino (que en este ejemplo es **C:\codegen\examples\cs**).
3. Abra la solución en Visual Studio y pulse **Ctrl + Mayús + B** para generarla.
4. Copie el archivo **Altova_Hierarchical.xml** desde **<Documentos>\Altova\MapForce2023\MapForceExamples** al directorio en el que se genera **Mapping.exe** (en este ejemplo, **C:\codegen\examples\cs\Mapping\bin\Debug**). Como hemos explicado anteriormente, la asignación intentará leer este archivo si no indica otro valor para el parámetro **InputFileName**.
5. Abra una ventana de la línea de comandos y cambie al directorio en el que se encuentra **Mapping.exe**.

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

6. Ejecute la aplicación con este comando:

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

En el comando anterior el parámetro de entrada `/OutputFileName` indica el nombre del archivo de salida que se va a generar.

C++

Para indicar un un parámetro de entrada personal a la aplicación línea de comandos en C++ generada por MapForce:

1. Abra el ejemplo **FileNamesAsParameters.mfd** que encontrará en el directorio **<Documentos>\Altova\MapForce2023\MapForceExamples**.
7. En el menú **Archivo** haga clic en Generar código en **Generar código en | C++** y seleccione un directorio de destino (que en este ejemplo es **C:\codegen\examples\cpp**).
2. Abra la solución en Visual Studio y pulse **Ctrl + Mayús + B** para generarla.
3. Copie el archivo **Altova_Hierarchical.xml** desde **<Documentos>\Altova\MapForce2023\MapForceExamples** al directorio en el que se genera **Mapping.exe** (en este ejemplo **C:\codegen\examples\cpp\Mapping\Debug**). Como hemos explicado anteriormente, la asignación intentará leer este archivo si no indica otro valor para el parámetro **InputFileName**.
8. Abra una ventana de la línea de comandos y cambie al directorio en el que se encuentra **Mapping.exe**.

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

4. Ejecute la aplicación con este comando:

 © 2016-2022 Altova GmbH

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

En el comando anterior el parámetro de entrada `/OutputFileName` indica el nombre del archivo de salida que se va a generar.

5.2 Obtener valores de cadena de una asignación

Cuando necesite devolver un valor de cadena de una asignación, puede usar un componente de salida simple. En el área de asignación los componentes de salida simples desempeñan el papel de componente de destino que tiene un tipo de datos de cadena en lugar de una estructura de elementos y secuencias. Por tanto, puede crear un componente de salida simple en lugar de (o además de) un componente de destino basado en un archivo. Por ejemplo, puede usar un componente de salida simple para probar y obtener una vista previa del resultado de una función (*véase* *Ejemplo: vista previa de resultados de una función* 490). No obstante, debe destacarse que el objetivo principal de los componentes de salida simples es recuperar una cadena cuando se llama a la API de MapForce Server sin necesidad de escribir ningún archivo.

Los componentes de entrada simples no se deben confundir con los parámetros de salida de las funciones definidas por el usuario (*véase* *Funciones definidas por el usuario* 575). En esta tabla puede ver en qué se parecen y en qué se diferencian:

Componentes de salida	**Parámetros de salida de funciones definidas por el usuario**
Se añaden desde el comando de menú **Función | Insertar componente de salida**.	Se añaden desde el comando de menú **Función | Insertar componente de salida**.
Tienen el tipo de datos "string".	Puede tener tipos de datos simples y complejos.
Afectan a toda la asignación.	Afectan solo al contexto de la función donde se definen.

Si lo necesita, puede añadir varios componentes de salida simples a la asignación. También puede usar componentes de salida simples junto con componentes de destino de base de datos y basados en archivos. Si la asignación contiene varios componentes de destino, puede consultar la vista previa de los datos que devuelve cada componente haciendo clic en el botón **Vista previa** ( ) de la barra de título del componente correspondiente y haciendo clic en el panel *Resultados* de la ventana de asignación.

Los componentes de salida simples se pueden usar en estos lenguajes de transformación de MapForce:

Lenguaje	**Funcionamiento**
BUILT-IN (cuando se genera la vista previa de la transformación de la asignación)	Puede consultar la vista previa de componentes de salida igual que los resultados de una asignación basada en archivos (haciendo clic en el panel *Resultados* de la ventana de asignación.
BUILT-IN (cuando se ejecuta el archivo de ejecución de MapForce Server)	Cuando ejecute un archivo de ejecución de MapForce Server compilado (*véase* *Compilar una asignación de MapForce* 955), el resultado de la asignación se devuelve en la secuencia de datos de salida estándar (stdout). De este modo podrá ver los resultados o redirigirlos a un archivo. Por ejemplo, si imaginamos que el nombre del archivo de ejecución de MapForce Server es **MiAsignación.mfx**, puede usar esta sintaxis para redirigir el resultado de la asignación al archivo **resultado.txt** y los errores al archivo **log.txt**:

 © 2016-2022 Altova GmbH

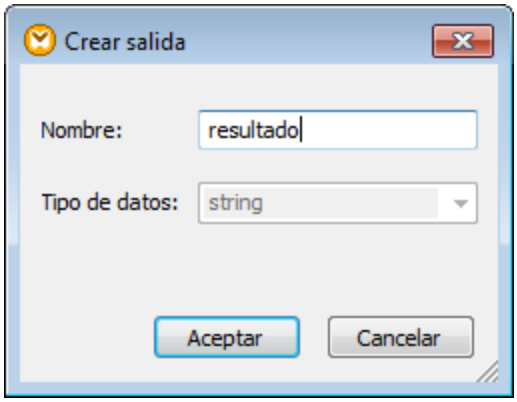
	`MapForceServer.exe run MiAsignación.mfx >resultado.txt 2>log.txt`
XSLT 1.0, XSLT 2.0, XSLT 3.0	En los archivos XSLT que se generan, el componente de salida simple definido en la asignación se convierte en el resultado de la transformación XSLT. Si usa un servidor RaptorXML Server, puede ordenar al servidor que escriba el resultado de la asignación en el archivo que se pasa como valor del parámetro `--output`. Para escribir el resultado en un archivo, añada o edite el parámetro `--output` en el archivo **DoTransform.bat**. Por ejemplo, el archivo **DoTransform.bat** que aparece en el comando siguiente se editó para que se escriba el resultado de la asignación en el archivo **Resultado.txt** (véase el texto resaltado). `RaptorXML xslt --xslt-version=2 --input="MappingMapToResult1.xslt" --output="Resultado.txt" %* "MappingMapToResult1.xslt"` Si no se definió el parámetro `--output`, el resultado de la asignación se escribirá en la secuencia de datos de salida estándar (stdout) cuando se ejecute la asignación.
C++, C#, Java	En el código C++, C# y Java que se genera, el resultado de la asignación se escribe en el resultado estándar de la aplicación generada. Si la asignación contiene varios componentes de destino, la aplicación generada concatena el resultado estándar de cada componente de salida y lo devuelve como resultado estándar unificado.

Cuando cree una asignación invertida (con el comando de menú **Herramientas | Crear asignación inversa**), los componentes de salida simples se convertirán en componentes de entrada simples.

5.2.1 Agregar componentes de salida simples

Para agregar un componente de salida en el área de asignación:

1. Asegúrese de que en la ventana de asignación está activa la asignación principal (y no una función definida por el usuario).
2. En el menú **Función** haga clic en el comando **Insertar componente de salida**.
3. Introduzca el nombre del componente.
4. Haga clic en **Aceptar**.



Cuadro de diálogo "Crear salida"

El nombre del componente se puede cambiar más adelante de varias maneras:

- Seleccionando el componente y haciendo clic en **Componente | Propiedades**.
- Haciendo doble clic en el título del componente.
- Haga clic con el botón derecho en el título del componente y después elija **Propiedades**.

5.2.2 Ejemplo: vista previa de resultados de una función

Este ejemplo sirve para explicar cómo consultar la vista previa de los resultados que devuelven funciones de MapForce con la ayuda de componente de salida simples. Antes de consultar este ejemplo es conveniente conocer el funcionamiento básico de las funciones en general y de las funciones de MapForce en particular. Puede encontrar más información en la sección Funciones 549.

Nuestro objetivo en este ejemplo es añadir varias funciones al área de asignación y aprender a generar la vista previa de los resultados con ayuda de componentes de salida simples. En concreto, este ejemplo utiliza varias funciones simples de la biblioteca principal **core** de MapForce. Aquí resumimos el uso de estas funciones:

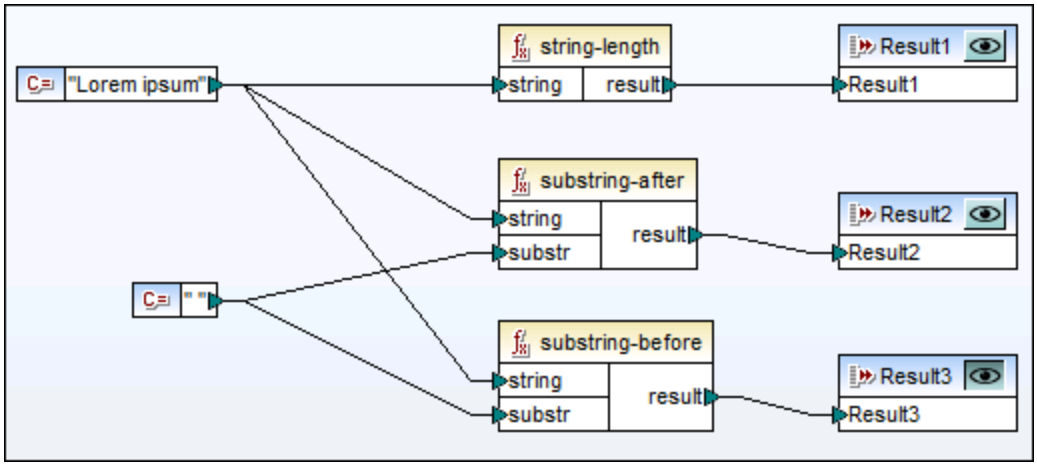
`string-length` 712	Devuelve el número de caracteres de la cadena que se dio como argumento. Por ejemplo, si pasamos a esta función el valor "Lorem ipsum", se obtiene el resultado "11" porque es el número de caracteres que toma el texto "Lorem ipsum".
`substring-after` 713	Devuelve la parte de la cadena que aparece después del separador que se dio como argumento. Por ejemplo, si pasamos a esta función el valor "Lorem ipsum" y el carácter de espaciado (" "), el resultado es "ipsum".
`substring-before` 714	Devuelve la parte de la cadena que aparece antes del separador que se dio como argumento. Por ejemplo, si pasamos a esta función el valor "Lorem ipsum" y el carácter de espaciado (" "), el resultado es "Lorem".

Siga estos pasos para probar todas estas funciones con un valor de texto (p. ej. "Lorem ipsum"):

1. Añada una constante con el valor "Lorem ipsum" al área de asignación (con el comando de menú **Insertar | Constante**). La constante será el parámetro de entrada para cada una de las funciones que vamos a probar.

 © 2016-2022 Altova GmbH

2. Añada las funciones **`string-length`**, **`substring-after`** y **`substring-before`** al área de asignación (arrastrándolas desde la biblioteca **core**, sección **string functions**, hasta el área de asignación).
3. Añada una constante con un carácter de espaciado como valor (" "). Este será el parámetro separador que requieren las funciones **`substring-after`** y **`substring-before`**.
4. Añada tres componentes de salida simples (con el comando de menú **Función | Insertar componente de salida**). En nuestro ejemplo estos componentes se llaman **Resultado1**, **Resultado2** y **Resultado3** respectivamente.
5. Conecte los componentes tal y como aparece en la siguiente imagen.



Probando resultados de funciones con componentes de salida simples

Tal y como puede ver en la imagen anterior, la cadena "Lorem ipsum" hace de parámetro de entrada para las funciones **`string-length`**, **`substring-after`** y **`substring-before`**. Además, las funciones **`substring-after`** y **`substring-before`** toman como segundo parámetro de entrada un valor de espaciado. Los componentes **Resultado1**, **Resultado2** y **Resultado3** pueden utilizarse para generar la vista previa del resultado de cada función.

Para obtener la vista previa de resultados de una función:

- Haga clic en el botón **Vista previa** ( ) en la barra de título del componente correspondiente y después haga clic en el panel *Resultados* de la ventana de asignación.

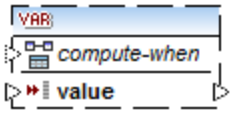
5.3 Variables

Las variables son un tipo especial de componente utilizado para almacenar un resultado de asignación intermedio y poder seguir procesándolo. Las variables pueden ser de tipo simple (p. ej. cadena, entero, valor booleano, etc) o de tipo complejo (estructuras jerárquicas).

Una de las cosas más importantes de las variables es que son secuencias y que pueden utilizarse para crear secuencias. En este sentido el término secuencia significa "una lista de cero o más elementos" (*véase* *Reglas y estrategias de asignación de datos* 871). Esto permite a las variables poder procesar varios elementos durante todo el ciclo de vida de la asignación. No obstante, también puede asignar un valor a una variable una sola vez y conservarlo para el resto de la asignación (*véase* *Cambiar el contexto y ámbito de las variables* 498).

Variables simples

Las variables simples se generan para representar tipos atómicos como cadenas de texto, números y elementos booleanos (imagen siguiente).



Variables complejas

Las variables complejas tienen estructura en forma de árbol. Estas son las estructuras en que pueden basarse las variables complejas.

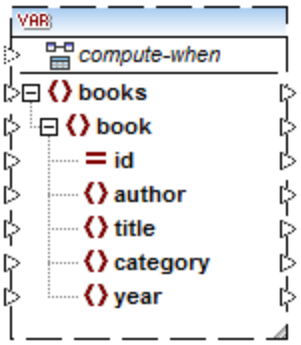
MapForce Basic Edition:
- Estructura XML Schema

MapForce Professional Edition:
- Estructura XML Schema
- Estructura de BD

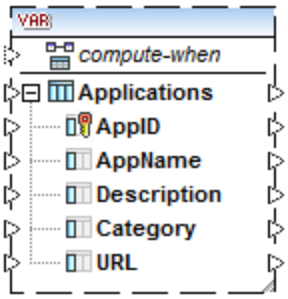
MapForce Enterprise Edition:
- Estructura XML Schema
- Estructura de BD
- Estructura EDI
- Estructura FlexText
- Estructura JSON Schema

Ejemplo 1: variable basada en XML Schema
Puede crear una variable de tipo complejo suministrando un esquema XML que exprese la estructura de la variable. Si el esquema define elementos globalmente, podrá elegir cuál de ellos debe ser el nodo raíz de la estructura de la variable. Recuerde que una variable no tiene ningún archivo XML de instancia asociado sino que sus datos se calculan cuando se ejecuta la asignación.

 © 2016-2022 Altova GmbH



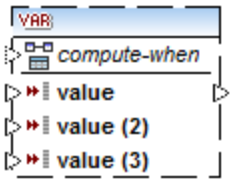
Ejemplo 2: variable basada en una BD
Este ejemplo (*imagen siguiente*) solo es relevante para las ediciones MapForce Professional y Enterprise. Si elige una estructura de BD para la variable, puede elegir una tabla concreta de la base de datos como elemento raíz de la estructura de la variable.



Compute-when
En las imágenes anteriores observará que cada variable tiene un elemento llamado `compute-when`. Conectar este elemento es opcional y sirve para controlar cómo se debe calcular el valor de la variable en la asignación (*véase* *Cambiar el contexto y el ámbito de las variables* 498).

Variables simples con entradas duplicadas

Si es necesario, los elementos de la estructura de una variable se pueden duplicar para que acepten datos de más de una conexión de origen, igual que se hace con componentes estándar (*véase* *Duplicar entradas* 81). Sin embargo, esto no se puede hacer en las variables creadas a partir de tablas de BD. En la imagen siguiente puede ver un ejemplo de variable simple con entradas duplicadas.



Variables encadenadas vs. variables

Hasta cierto punto las variables pueden compararse con los componentes intermedios de una asignación en cadena (*véase* *Asignaciones en cadena* 840). Sin embargo, son mucho más flexibles y prácticas cuando no se necesita generar archivos intermedios en cada fase de la asignación. A continuación se explican las diferencias que existen entre las variables y las asignaciones en cadena.

Asignaciones en cadena	**Variables**
Las asignaciones en cadena se desarrollan en dos fases totalmente independientes. Por ejemplo, imaginemos que una asignación tiene tres componentes llamados A, B y C. La ejecución de la asignación se desarrolla en dos fases: la ejecución de la asignación desde A hasta B y la ejecución de la asignación desde B hasta C.	Mientras se ejecuta la asignación, las variables se evalúan en función de su contexto y su ámbito. Su contexto y ámbito puede modificarse (*véase* *Cambiar el contexto y ámbito de las variables* 498).
Cuando se ejecuta la asignación, los resultados intermedios se almacenan de forma externa en archivos.	Cuando se ejecuta la asignación, los resultados intermedios se almacenan de forma internan. No se producen archivos externos con los resultados de la variable.
Con el botón de vista previa se puede consultar la vista previa del resultado intermedio.	No se puede consultar la vista previa del resultado de una variable porque éste se calcula en tiempo de ejecución.

Nota: no se admite el uso de variables si el lenguaje de transformación elegido es XSLT 1.0.

5.3.1 Agregar variables

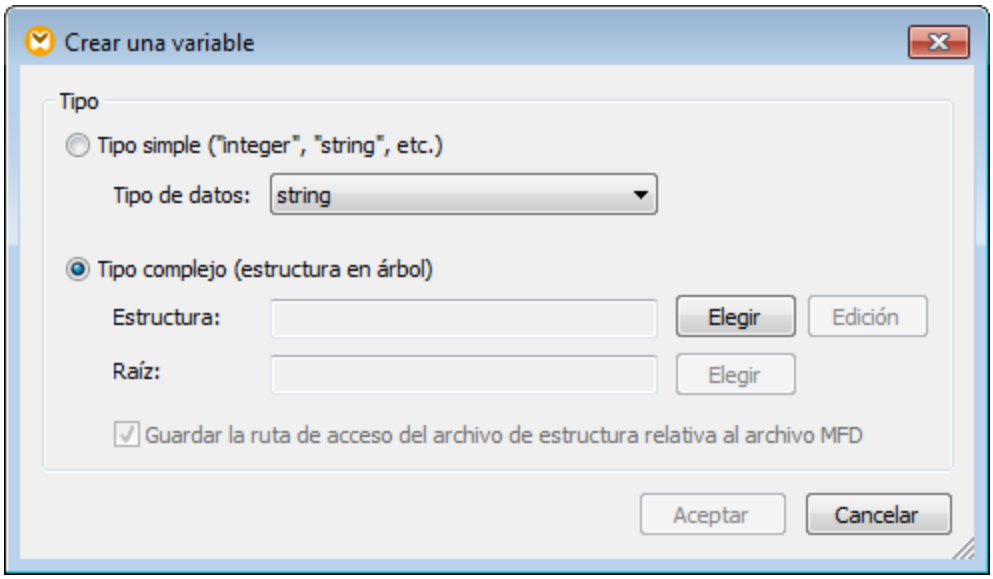
En este apartado explicamos cómo agregar variables a una asignación. La primera opción es agregar variables con comandos de menú o de la barra de herramientas. La segunda opción permite agregar variables desde el menú contextual.

Opción 1: con comandos de menú o de la barra de herramientas

Esta opción sirve para agregar variables con comandos de menú o de la barra de herramientas. Siga estos pasos:

1. En el menú **Insertar** haga clic en el comando **Variable**. También puede hacer clic en el botón (**Variable**) de la barra de herramientas.

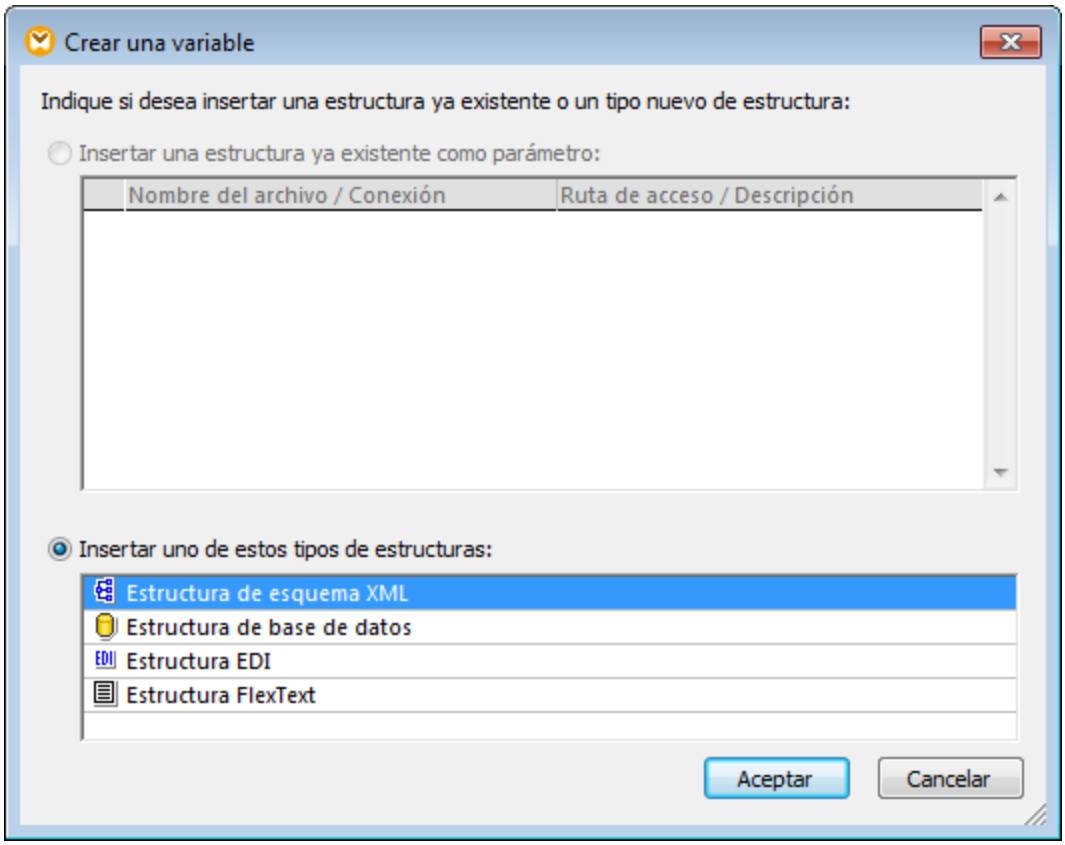
 © 2016-2022 Altova GmbH



2. Seleccione el tipo de variable que quiere insertar (de tipo simple o complejo).

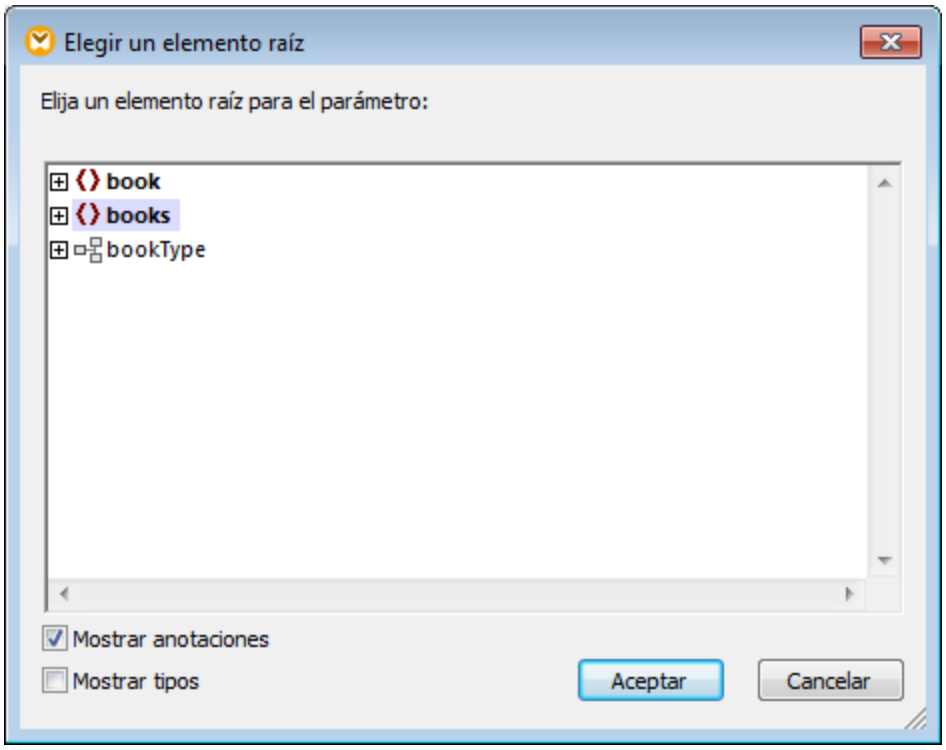
Si selecciona **Tipo complejo** debe seguir también estos pasos:

3. Haga clic en **Elegir** para seleccionar el origen del que se debe tomar la estructura de la variable 492. Las estructuras que ve en la imagen siguiente se aplican solamente a MapForce Enterprise Edition. Consulte la lista de estructuras relevantes para las otras ediciones de MapForce en el apartado anterior 492.



4. Cuando la aplicación se lo pida indique el elemento raíz de la estructura de la variable. Por ejemplo, en los esquemas XML puede seleccionar cualquier elemento o tipo de la fuente seleccionada (*imagen siguiente*).

 © 2016-2022 Altova GmbH

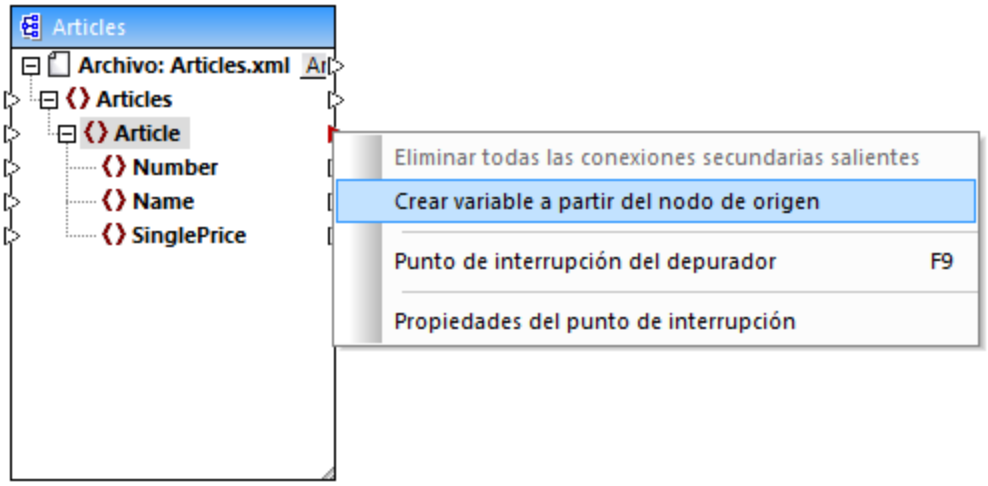


Opción 2: desde el menú contextual

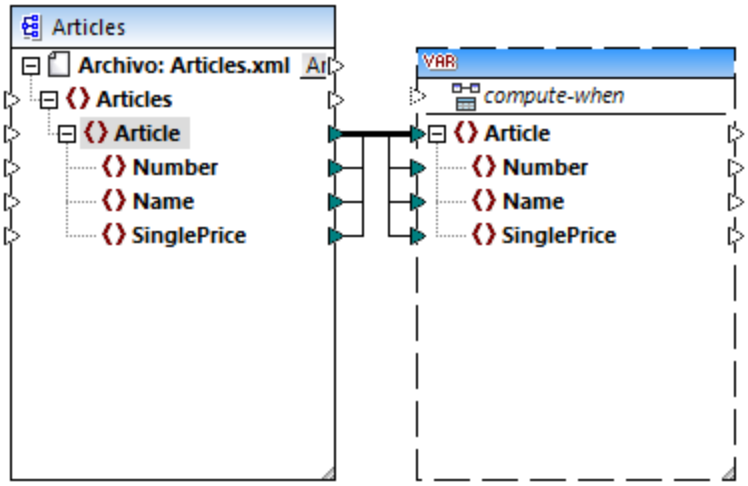
La segunda opción permite crear variables desde el menú contextual. A continuación se enumeran todas las opciones posibles.

Variable a partir de un nodo de origen

Para crear una variable a partir de un nodo de origen, haga clic con el botón derecho en el conector de salida de un componente (en este ejemplo usamos el del elemento `<Article>`) y seleccione **Crear variable a partir del nodo de origen** (*imagen siguiente*).



Se crea una variable compleja con el esquema de origen del componente `Articles`. Todos los elementos se conectan automáticamente con una conexión de copia total 95 (*imagen siguiente*).

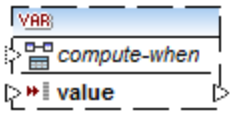


Variable a partir de un nodo de destino
Para crear una variable a partir de un nodo de destino, haga clic con el botón derecho en el conector de entrada de un componente de destino y seleccione **Crear variable a partir del nodo de destino**. Se crea una variable compleja con el mismo esquema del componente de destino. Todos los elementos se conectan automáticamente con una conexión de copia total.

Variable a partir de un filtro:
Para crear una variable usando un filtro, haga clic con el botón derecho en el conector de salida de un componente de filtro (`on-true/on-false`) y seleccione **Crear variable a partir del nodo de origen**. Se crea un componente complejo con el esquema de origen y usa automáticamente el elemento vinculado al filtro de entrada como elemento raíz del componente intermedio.

5.3.2 Cambiar el contexto y ámbito de las variables

Todas las variables tienen un elemento de entrada `compute-when` que sirve para controlar el ámbito de la variable, es decir, en qué momento y con qué frecuencia se calcula el valor de la variable cuando se ejecuta la asignación. En muchos casos este elemento de entrada puede dejarse sin conectar, pero en ocasiones puede ser imprescindible a la hora de reemplazar el contexto predeterminado o para optimizar el rendimiento de la asignación de datos.



El término ***subestructura*** utilizado en esta documentación hace referencia al conjunto de elementos/nodos de un componente de destino y a todos sus descendientes (p. ej. un elemento `<Persona>` con sus elementos secundarios `<Nombre>` y `<Apellido>`.

 © 2016-2022 Altova GmbH

El término ***valor de variable*** sirve para denominar los datos disponibles en el lado saliente del componente de variable.

- En el caso de las variables simples se trata de una secuencia de valores atómicos cuyo tipo de datos está especificado en las propiedades del componente.+
- En el caso de las variables complejas se trata de una secuencia de nodos raíz (del tipo especificado en las propiedades del componente), cada uno de ellos con sus nodos descendientes.

La secuencia de valores atómicos (o nodos) puede contener un elemento e incluso cero. Esto depende de cómo esté conectado el lado entrante de la variable y de con qué elementos principales del componente de origen/destino esté conectado.

Si Compute-when no está conectado (configuración predeterminada)

Si el elemento de entrada `compute-when` no está conectado (a ningún nodo de salida de un componente de origen), entonces el valor de la variable se calcula *cuando se use por primera vez en una subestructura de destino* (ya sea por conexión directa entre el componente de variable y un nodo del componente de destino o por conexión indirecta a través de funciones). El mismo valor de variable se utiliza después para todos los nodos secundarios de destino incluidos dentro de la subestructura.

El valor real de la variable dependerá de las conexiones que existan entre los elementos principales del componente de origen y de destino. Este comportamiento predeterminado es idéntico al de los resultados complejos de funciones definidas por el usuario 578 y de llamadas a función de servicio web. Si el resultado de la variable se conecta a varios nodos de destino que no guardan relación entre sí, el valor de la variable se calcula por separado para cada uno de ellos. Esto puede dar lugar a resultados distintos porque las distintas conexiones principales influyen en el contexto en el que se evalúa el valor de la variable.

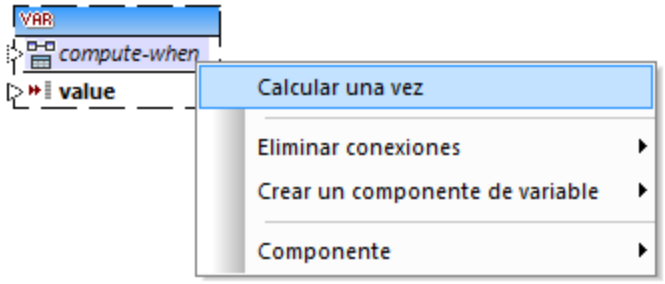
Si Compute-when está conectado

Si se conecta un conector de salida de un componente de origen con el elemento `compute-when`, el valor de la variable se calcula *cuando dicho elemento de origen se use por primera vez en una subestructura de destino*.

En realidad la variable hace las veces de elemento secundario del elemento que está conectado a `compute-when` (*imagen siguiente*). Esto permite vincular la variable con un elemento de origen concreto. Es decir, en tiempo de ejecución la variable se vuelve a evaluar cada vez que se lea un elemento nuevo de la secuencia del componente de origen. Se trata de la regla general que rige las conexiones en MapForce: *por cada elemento de origen, se crea uno de destino*. En el caso del elemento de entrada `compute-when` esto significa que *por cada elemento de origen, se calcula el valor de variable* (*véase* *Reglas y estrategias de asignación de datos* 871).

Compute-once

Si fuera necesario, puede solicitar que el valor de variable se calcule una sola vez antes de cada componente de destino, lo cual convierte a la variable en una constante global para el resto de la asignación. Esto se consigue haciendo clic con el botón derecho en el elemento `compute-when` y seleccionando el comando **Calcular una vez** en el menú contextual (*imagen siguiente*).

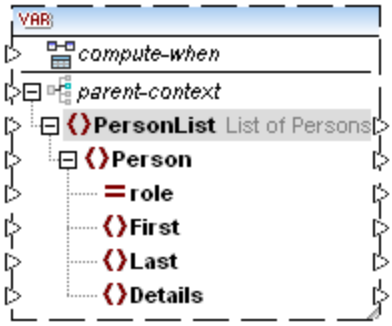


Cuando se cambia el ámbito de una variable por `compute-when=once`, el conector de entrada se elimina del elemento `compute-when` (porque dicha variable solamente se evaluará una vez). En funciones definidas por el usuario la variable `compute-when=once` se evalúa cada vez que se llama a la función y antes de que el resultado de la función se evalúe.

Contexto primario

En ocasiones puede ser necesario agregar un contexto primario. Por ejemplo, si la asignación usa varios filtros y necesita recorrer un nodo primario adicional (*véase* *Reemplazar el contexto de la asignación* 879).

Para agregar un contexto primario a una variable basta con hacer clic con el botón derecho en el nodo raíz (p. ej. "PersonList") y seleccionar el comando b en el menú contextual. Como resultado se añade un nodo nuevo llamado `parent-context` a la jerarquía de la variable.



El contexto primario añade un nodo primario virtual a la jerarquía del componente, lo cual permite recorrer otro nodo más en el mismo componente de origen o en otro distinto.

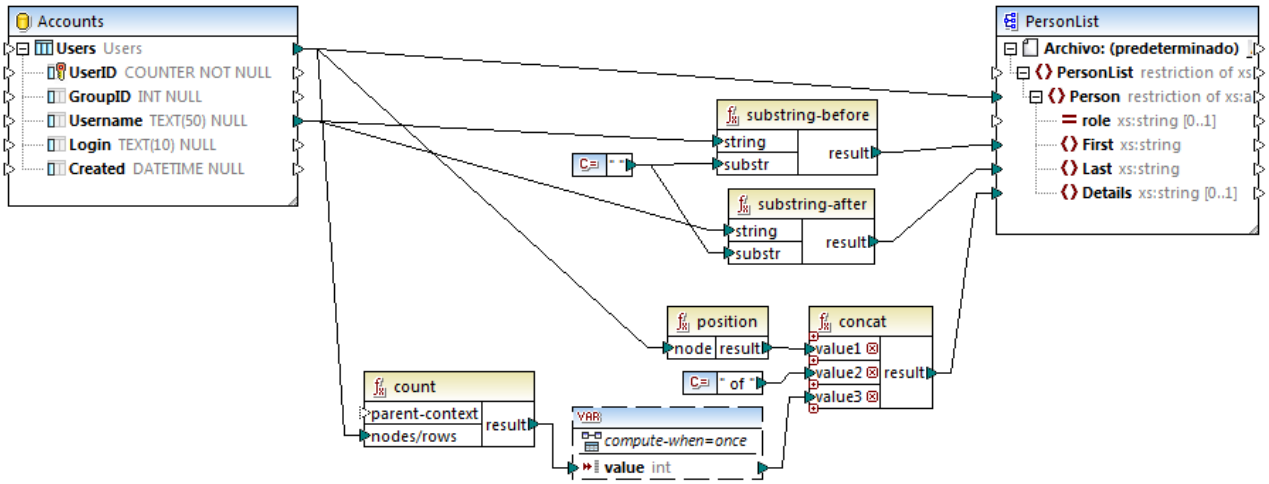
5.3.3 Ejemplo: contar filas de tabla de BD

El diseño de asignación que aparece más abajo corresponde al archivo **DB_UserList.mfd** situado en la carpeta **<Documentos>\Altova\MapForce2023\MapForceExamples**. Esta asignación extrae registros de usuarios de una tabla de base de datos llamada "Users" y los escribe en un archivo XML. La columna de base de datos "Username" contiene el nombre y el apellido de una persona (p. ej. "Vernon Callaby"). Esta asignación tiene varios objetivos:

1. Crear un elemento `Person` nuevo por cada registro de la tabla "Users" en el archivo XML de destino.

 © 2016-2022 Altova GmbH

2. Dividir el valor extraído del campo de base de datos "Username" en dos campos diferentes del archivo XML (`First` y `Last`).
3. Encontrar el número secuencial de cada registro comparado con el número total de registros presentes en la base de datos (p. ej. "Registro 1 de 4") y escribir esta información en el elemento `Details`.



DB_UserList.mfd

Como indica el diseño, para poder conseguir el primer objetivo debemos dibujar una conexión entre la tabla de origen "Users" y el elemento `Person` del archivo XML de destino. Esto permite crear un elemento nuevo `Person` por cada registro de la tabla de origen.

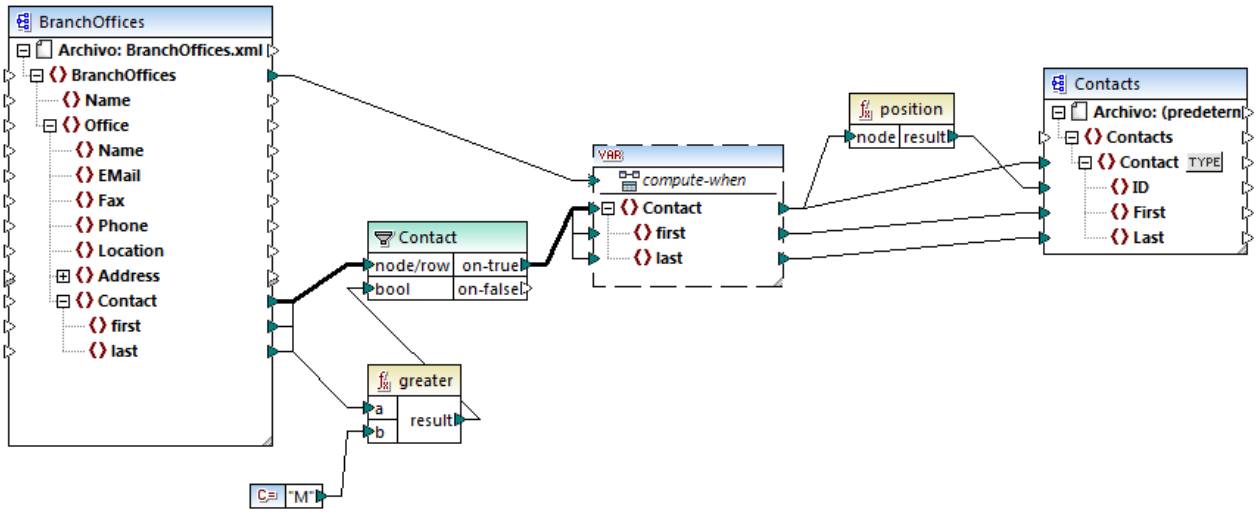
El valor del campo "Username" se pasa a las funciones **`substring-before`** [714] y **`substring-after`** [713] , que se encargan de extraer el texto situado antes y después del carácter de espaciado (" ") respectivamente. Así se consigue el segundo objetivo del diseño.

Y, por último, para conseguir el tercer objetivo es fundamental la función **`count`**. El resultado de esta función se pasa a una variable, que se encarga de almacenar este resultado en la asignación y servirlo a la hora de escribir el elemento `Details` de cada personal en el archivo XML de destino. Recuerde que por razones de eficacia los registros de base de datos se deberían contar una sola vez, así que definimos el ámbito de variable `compute-when=once` (*véase* *Cambiar el contexto y ámbito de las variables* [498]).

5.3.4 Ejemplo: filtrar y numerar nodos

El diseño de asignación que aparece más abajo corresponde al archivo **PositionInFilteredSequence.mfd** situado en la carpeta **<Documentos>\Altova\MapForce2023\MapForceExamples**.

Esta asignación lee un archivo XML que contiene datos de contacto de varias persona, los filtra y los escribe en un archivo XML de destino. El objetivo del diseño es filtrar datos del archivo XML de origen de las personas cuyo apellido empiece a partir de la letra M. Además, los contactos extraídos deben estar numerados en el XML de destino. Este número hará de identificador único en el XML de destino.



PositionInFilteredSequence.mfd

Para conseguir este objetivo se añadieron varios componentes al diseño de asignación:

- Un filtro (*véase* *Filtros y condiciones* 543)
- Una variable compleja (*véase* *Agregar variables* 494)
- Las funciones **greater** 665 y **position** 700 (*véase* *Agregar una función* 550)
- Una constante (para agregar una constante seleccione el comando de menú **Insertar | Constante**).

La variable usa el mismo esquema que el componente de origen. Si hacemos clic con el botón derecho en la variable y seleccionamos Propiedades en el menú contextual, veremos que el nodo raíz que está seleccionado para esta estructura de variable es **BranchOffices/Office/Contact**.

Primero se pasan al filtro los datos del componente de origen. Después el filtro pasa a la variable los registros que cumplan la condición de filtrado. Es decir, el filtro está configurado para obtener los nodos `Contact` cuyo nombre sea igual o mayor que "M". Para ello se usa la función **greater** 665 que compara cada elemento `last` con el valor de constante "M".

La variable tiene la entrada `compute-when` conectada con el elemento raíz del componente de origen (`BranchOffices`). En tiempo de ejecución esto hace que la variable se vuelva a evaluar cada vez que se lea un elemento nuevo de la secuencia del componente de origen. Sin embargo, en esta asignación no influye en modo alguno el conectar o no conectar el elemento `compute-when`. El motivo es que la variable está conectada al elemento de origen `Contact` (de forma indirecta a través del filtro) y se calcularía tantas veces como instancias de `Contact` cumplan la condición de filtrado.

La función **position** 700 devuelve (por cada iteración de la variable) el número de la secuencia actual. Solo ocho contactos cumplen la condición. Por tanto, si consultamos la vista previa del resultado de la asignación, veremos que se escribieron los identificadores 1 - 8 en el elemento `ID` del componente de destino.

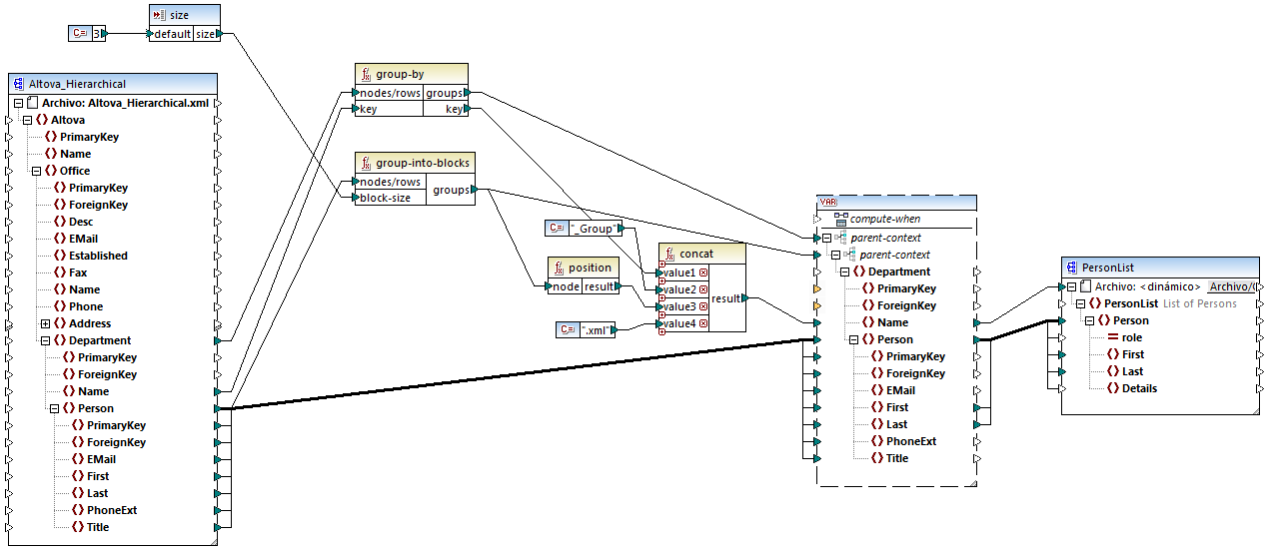
La variable se necesita porque necesitamos numerar todos los registros. Si se hubiera conectado el resultado del filtro al componente de destino directamente, no habría manera de numerar cada instancia de `Contact`. La función de la variable es, por tanto, almacenar temporalmente cada instancia de `Contact` para poder numerarlas antes de escribirlas en el destino.

 © 2016-2022 Altova GmbH

5.3.5 Ejemplo: crear grupos y subgrupos de registros

El diseño de asignación que aparece más abajo corresponde al archivo **DividePersonsByDepartmentIntoGroups.mfd** situado en la carpeta **<Documentos>\Altova\MapForce2023\MapForceExamples**.

Este diseño procesa un archivo XML que contiene registros de empleados de una compañía ficticia. La compañía tiene dos oficinas: "Nanonull, Inc." y "Nanonull Partners, Inc". Cada oficina tiene varios departamentos (p. ej. "IT", "Marketing", etc.) y cada departamento tiene como mínimo un empleado. El objetivo de esta asignación de datos es crear grupos formados por un tres personas como máximo de cada departamento, sin importar la oficina. El tamaño predeterminado de cada grupo es 3, pero esto se puede cambiar. Además cada grupo debe guardarse en un archivo XML distinto, cuyo nombre seguirá el patrón "<Nombre Departamento>_NºGrupo" (p. ej. **Marketing_Group1.xml**, **Marketing_Group2.xml**, etc.).



DividePersonsByDepartmentIntoGroups.mfd

Como puede ver, la asignación cuenta con una variable compleja y varios componentes más (funciones sobre todo). La variable tiene la misma estructura que el elemento `Department` del archivo XML de origen. Si hacemos clic con el botón derecho en la variable para ver sus propiedades, observaremos que utiliza el mismo esquema que el componente de origen y que su elemento raíz es `Department`. Y lo que es más importante, la variable tiene dos elementos `parent-context` anidados, que garantizan que la variable se calcule primero en el contexto de cada departamento y después en el contexto de cada grupo dentro de cada departamento (*véase* *Cambiar el contexto y ámbito de las variables* 498).

En un principio la asignación recorre todos los departamentos para obtener el nombre de cada uno de ellos (el nombre de los departamentos se necesita para crear el nombre de archivo que corresponde a cada grupo). Esto se consigue conectando la función **`group-by`** 689 al elemento de origen `Department` y pasando el nombre del departamento como clave de agrupación.

Después, dentro del contexto de cada departamento, se lleva a cabo otra agrupación: la asignación llama a la función **`group-into-blocks`** 692 para crear los grupos necesarios de empleados. El tamaño de cada grupo viene dado por un componente de entrada simple cuyo valor predeterminado es "3". El valor predeterminado

viene dado por una constante. En este ejemplo, para cambiar el tamaño de cada grupo basta con modificar el valor de la constante según corresponda. Sin embargo, también se puede modificar el componente de entrada **`size`** para que, si la asignación se ejecuta con código generado o con MapForce Server, el tamaño de cada grupo pueda especificarse como parámetro (*véase* *Pasar parámetros a la asignación* 477).

A continuación, el valor de la variable se pasa al componente XML de destino PersonList. El nombre de archivo de cada grupo creado se calcula con ayuda de la función **`concat`** 708, mediante la concatenación de:

1. el nombre de cada departamento,
2. la cadena "_Group",
3. el número que tiene el grupo en la secuencia actual (p. ej. "1" si se trata del primer grupo del departamento) y
4. la cadena ".xml"

El resultado de la concatenación se almacena en el elemento `Name` de la variable y después se pasa como nombre de archivo dinámico al componente de destino. Por tanto, se crea un nombre de archivo nuevo por cada valor recibido. En este ejemplo, la variable calcula 8 grupos en total, así que se crean 8 archivos de salida cuando se ejecuta la asignación. Para más información consulte la sección Procesar varios archivos de entrada o salida simultáneamente 892.

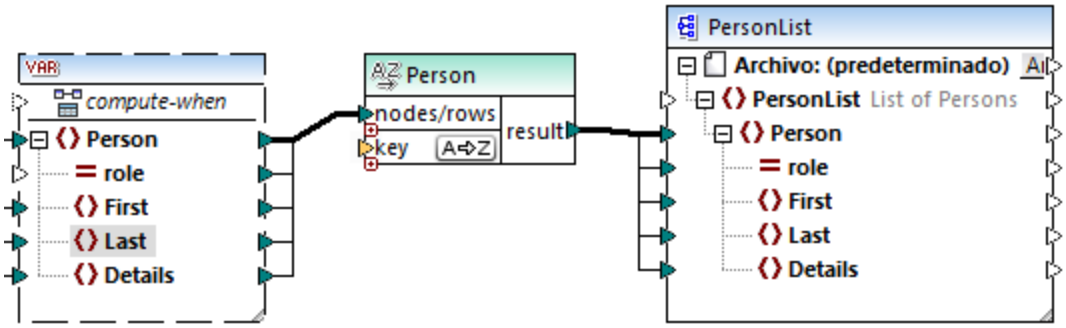
 © 2016-2022 Altova GmbH

5.4 Ordenar datos

Para ordenar datos de entrada siguiendo un criterio de ordenación concreto basta con usar un componente de ordenación. El componente de ordenación es compatible con estos lenguajes de destino: XSLT2, XQuery y el motor de ejecución integrado. Si elige el lenguaje de transformación integrado, podrá usar el componente de ordenación para ordenar datos de tabla de base de datos. Sin embargo, conseguirá un mejor rendimiento con el componente WHERE/ORDER de SQL (*véase* *Filtrar y ordenar datos de base de datos (WHERE/ORDER de SQL* 352) 352).

Para agregar un componente de ordenación a la asignación:

- Haga clic con el botón derecho en una conexión y seleccione **Insertar componente de ordenación: nodos/filas** en el menú contextual. Esto inserta un componente de ordenación y lo conecta automáticamente a los componentes de origen y destino. Por ejemplo, en la imagen siguiente el componente de ordenación se introdujo entre una variable y un componente XML. Lo único que falta por conectar a mano es el criterio de ordenación (el campo por el que desea ordenar los elementos).

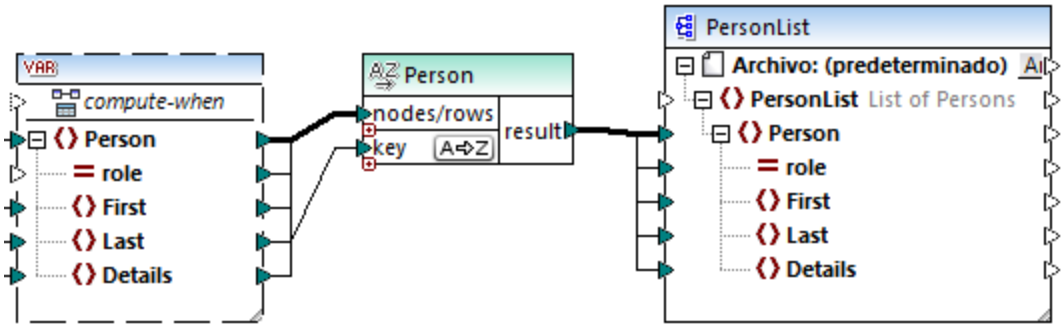


- En el menú **Insertar** seleccione el comando **Componente de ordenación: nodos/filas** (o haga clic en el botón **Insertar componente de ordenación** de la barra de herramientas). Esto inserta el componente de ordenación pero sin conectarlo.

En cuanto conectemos el componente de ordenación con el componente de origen, en la barra de título del componente de ordenación aparecerá el nombre del elemento que está conectado a `nodes/rows`.

Para definir por qué elemento se deben ordenar los nodos/filas:

- Conecte el elemento por el que desea ordenar los nodos/filas al parámetro `key` del componente de ordenación. Por ejemplo, en la imagen siguiente los nodos/filas `Person` se ordenan por el campo `Last`.

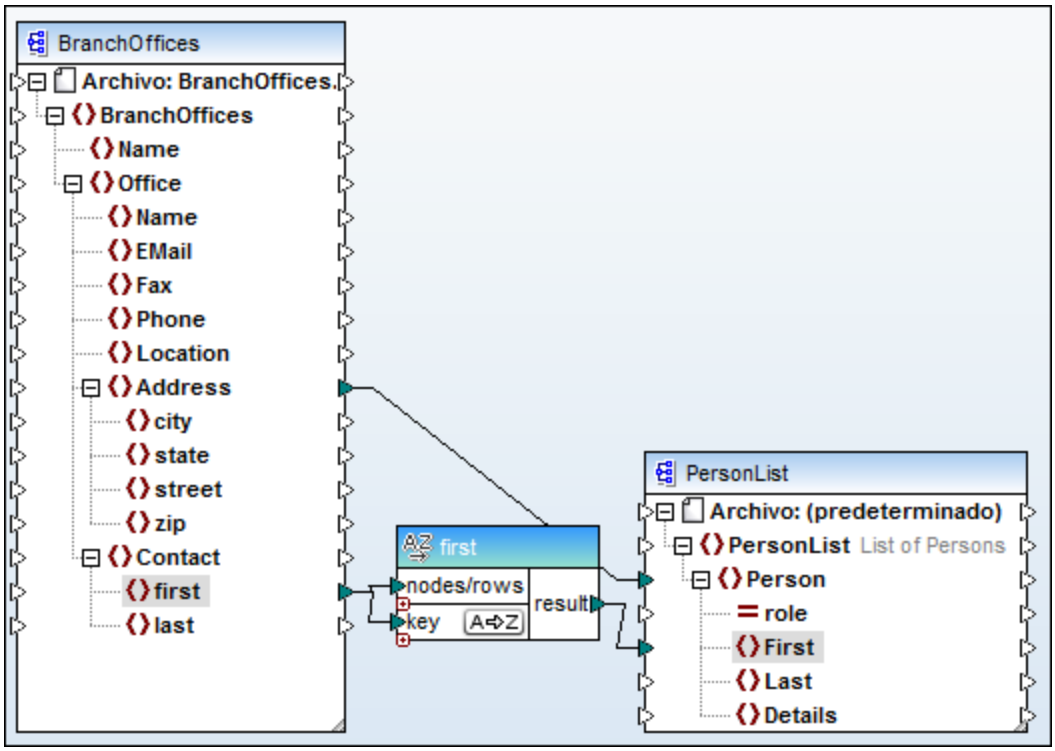


Para cambiar el orden:

- Haga clic en el icono A⇨Z del componente de ordenación. Se convertirá en el icono Z➔A para mostrar que el orden se invirtió.

Para ordenar datos de entrada compuestos por elementos de tipo simple:

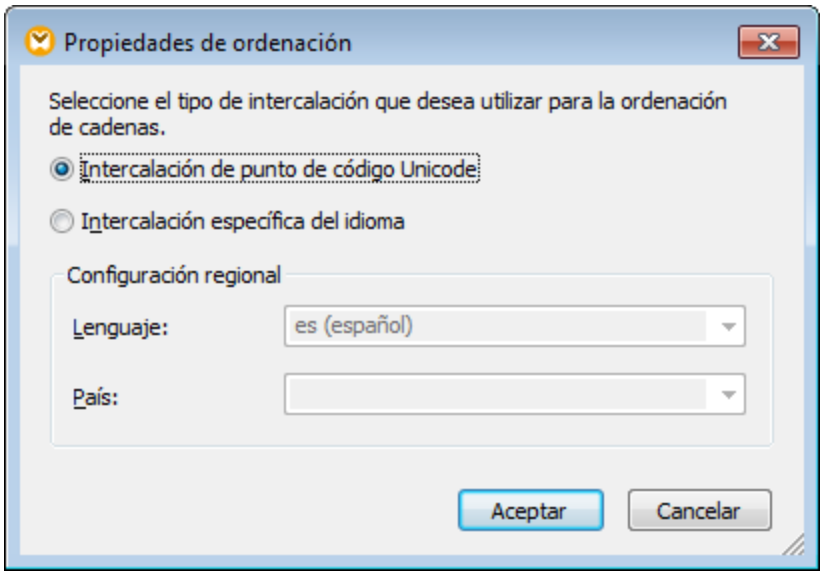
- Conecte el elemento tanto al parámetro `nodes/rows` como al parámetro `key` del componente de ordenación. En la imagen siguiente, por ejemplo, se ordena el elemento de tipo simple `first`.



 © 2016-2022 Altova GmbH

Para ordenar cadenas usando reglas de un idioma concreto:

- Haga doble clic en el título del componente de ordenación para abrir el cuadro de diálogo "Propiedades de ordenación" (*imagen siguiente*).

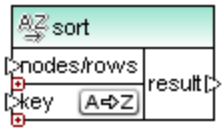


Intercalación de punto de código: esta opción predeterminada compara/ordena las cadenas basándose en valores de punto de código. Los valores de punto de código son enteros que se asignaron a caracteres abstractos del conjunto de caracteres universal adoptado por el consorcio Unicode. Esta opción permite ordenar datos en muchos idiomas y scripts.

Intercalación específica del idioma: esta opción permite definir el idioma y el país que debe utilizarse para la ordenación. Esta opción es compatible con el motor de ejecución integrado. Para XSLT la compatibilidad dependerá del motor que se utilice para ejecutar el código.

5.4.1 Ordenar según varias claves

Cuando se añade un componente de ordenación a la asignación, MapForce crea por defecto un criterio de ordenación llamado `key`.



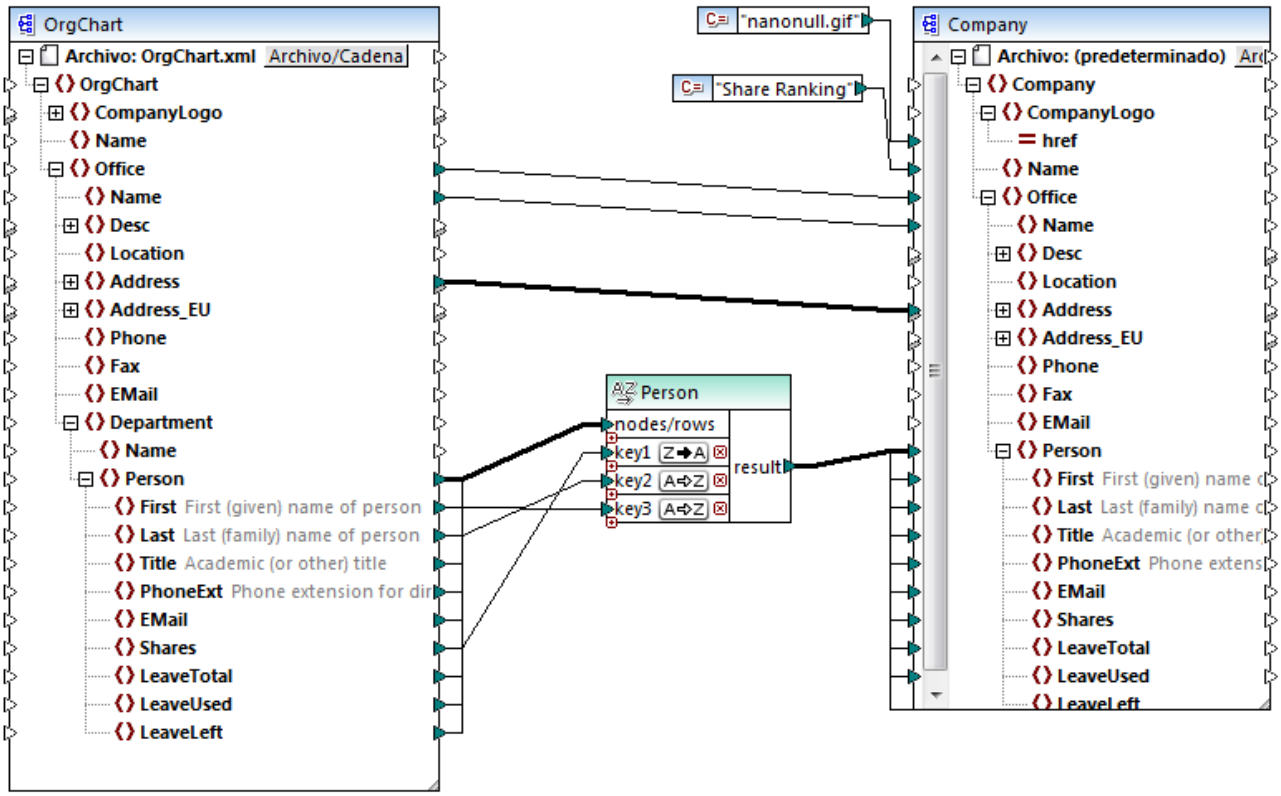
Componente de ordenación

Si desea usar varios criterios de ordenación, deberá realizar varios ajustes en el componente:

- Haga clic en el icono **Agregar criterio** (⊡) para agregar un criterio nuevo (p. ej. `key2`).
- Haga clic en el icono **Eliminar criterio** (⊠) para eliminar un criterio de ordenación.

- Arrastre y coloque una conexión encima del icono para agregar un criterio y al mismo tiempo conectarlo.

El diseño de asignación **<Documentos>\Altova\MapForce2023\MapForceExamples\SortByMultipleKeys.mfd** contiene un componente de ordenación que usa varios criterios.



SortByMultipleKeys.mfd

En esta asignación de datos los registros `Person` se ordenan siguiendo tres criterios:

1. `Shares` (número de acciones en propiedad de cada persona)
2. `Last` (apellido)
3. `First` (nombre)

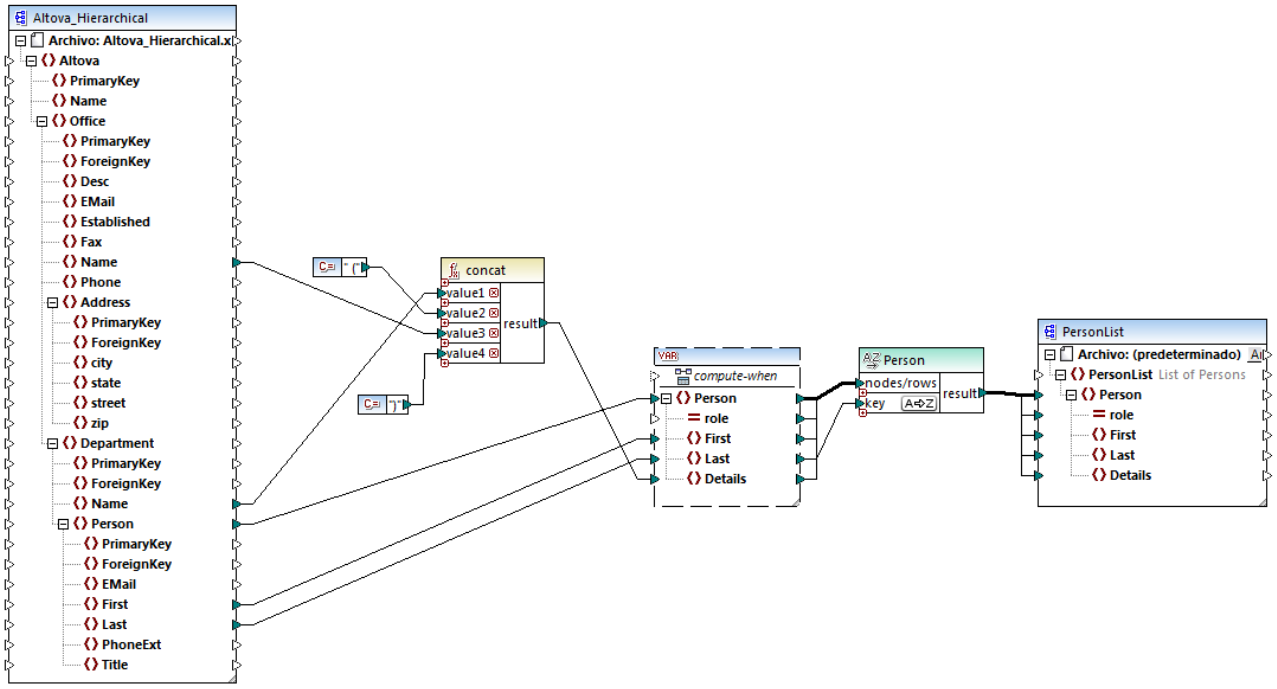
Observe que la posición del criterio de ordenación en el componente determina su prioridad. Por ejemplo, en la imagen puede ver que los registros se orden primero en función del número de acciones. Se trata del criterio de ordenación con mayor prioridad. Si el número de acciones fuera el mismo, las personas se ordenan por apellido. Y, por último, si varios personas tuvieran el mismo apellido, se ordenarán por nombre.

El orden de cada criterio puede ser distinto. Por ejemplo, en la imagen anterior puede ver que el criterio `Shares` indica un orden descendiente (Z-A), mientras que los otros dos criterios siguen un orden ascendente (A-Z).

 © 2016-2022 Altova GmbH

5.4.2 Ordenar con variables

En algunos casos puede ser necesario agregar variables intermedias para conseguir ciertos resultados. En el diseño de asignación que aparece a continuación se extraen registros de un archivo XML y éstos se orden con ayuda de variables intermedias. Se trata del diseño de asignación **<Documentos>\Altova\MapForce2023\MapForceExamples\Altova_Hierarchical_Sort.mfd**.



Altova_Hierarchical_Sort.mfd

Esta asignación lee datos de un archivo XML de origen llamado **Altova_Hierarchical.xml** y los escribe en un archivo XML de destino. Como puede ver, el XML de origen contiene información sobre una compañía ficticia. La compañía se divide en oficinas y éstas en departamentos. A su vez, los departamentos están compuestos por empleados.

El componente XML de destino `PersonList` contiene una lista de registros `Person`. El elemento `Details` tiene la función de almacenar información sobre la oficina y el departamento al que pertenece cada empleado.

El objetivo de esta asignación es extraer todos los empleados del XML de origen y ordenarlos por apellido y por orden alfabético. Además, el nombre de la oficina y del departamento al que pertenece cada empleado debe escribirse en el elemento `Details`.

Estos son los componentes necesarios para conseguir el objetivo de la asignación:

1. La función **`concat`**, que devuelve una cadena con el patrón `Office(Department)`. Toma como entrada el nombre de la oficina, el nombre del departamento y dos constantes que aportan el paréntesis de apertura y de cierre respectivamente (*véase* *Funciones básicas* 550).
2. Una variable intermedia. El papel que desempeña es recopilar todos los datos de cada empleado en el mismo contexto de asignación. La variable consigue que la asignación busque el departamento y la

oficina de cada empleado, en el contexto de cada empleado. Es decir, la variable *recuerda* el nombre de la oficina y del departamento al que pertenece cada empleado. Sin la variable, el contexto sería incorrecto y la asignación produciría resultados no deseados (*véase Reglas y estrategias de asignación de datos* 871). Tenga en cuenta que la variable reproduce la estructura del archivo XML de destino (usa el mismo esquema XML). Esto permite conectar el resultado del componente de ordenación con el componente de destino por medio de una conexión de copia total. Consulte las secciones Usar variables 492 y Conexiones de copia total 95 para obtener más información.

3. Un componente de ordenación que ordena los datos. Observe que el criterio de ordenación del componente está conectado al elemento `Last` de la variable, lo cual permite ordenar todos los registros por apellido.

© 2016-2022 Altova GmbH

5.5 Agrupar datos

Supongamos que una asignación debe agrupar nodos o filas. Esto se puede conseguir con ayuda de las siguientes funciones integradas de MapForce:

- **group-by**
- **group-adjacent**
- **group-into-blocks**
- **group-starting-with**
- **group-ending-with**

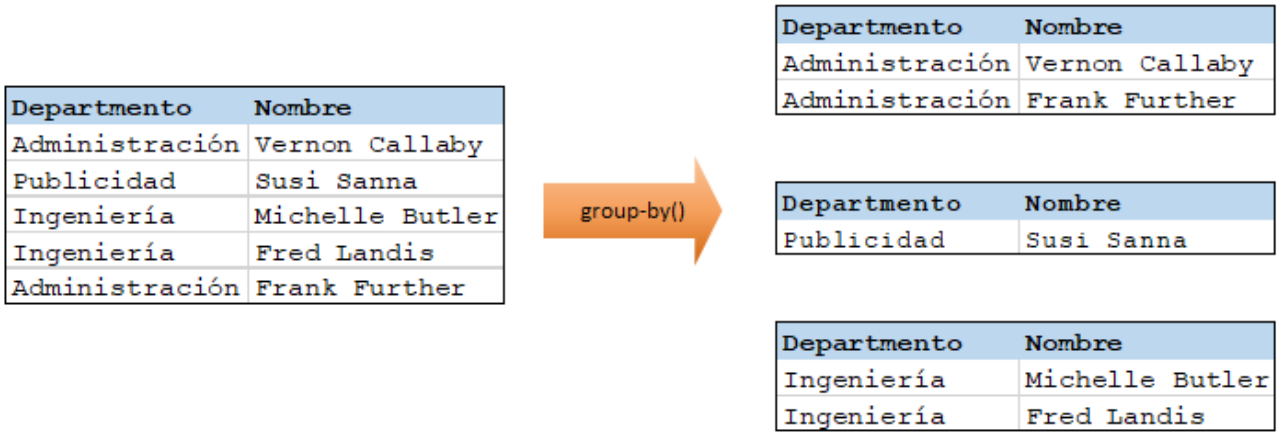
Para utilizar cualquiera de estas funciones en la asignación, arrástrelas desde la ventana Bibliotecas al área de la asignación. Consulte también Agregar una función integrada a una asignación de datos 550.

Nota: las funciones de agrupación están disponibles en estos lenguajes: XSLT 2.0, XSLT 3.0, C++, C#, Java y Built-In.

Las siguientes secciones ofrecen los típicos ejemplos de uso de las funciones de agrupación. Estos ejemplos están acompañados por la siguiente asignación de ejemplo: **<Documentos>\Altova\MapForce2023\MapForceExamples\Tutorial\GroupingFunctions.mfd**. Observe que la asignación de ejemplo contiene varias transformaciones, una por cada función. Como solo se puede previsualizar un resultado cada vez, recuerde hacer clic en el botón **Vista previa** de la función que quiere comprobar antes de hacer clic en la pestaña *Resultados*.

group-by

La función **group-by** crea grupos de registros conforme a las claves de agrupación que se indiquen. Por ejemplo, en el siguiente ejemplo de transformación abstracta la clave de agrupación es "Departmento". Como hay un total de tres departamentos únicos, al aplicar la función **group-by** se crean tres grupos:



Para más información consulte la referencia a la función group-by 689.

group-adjacent

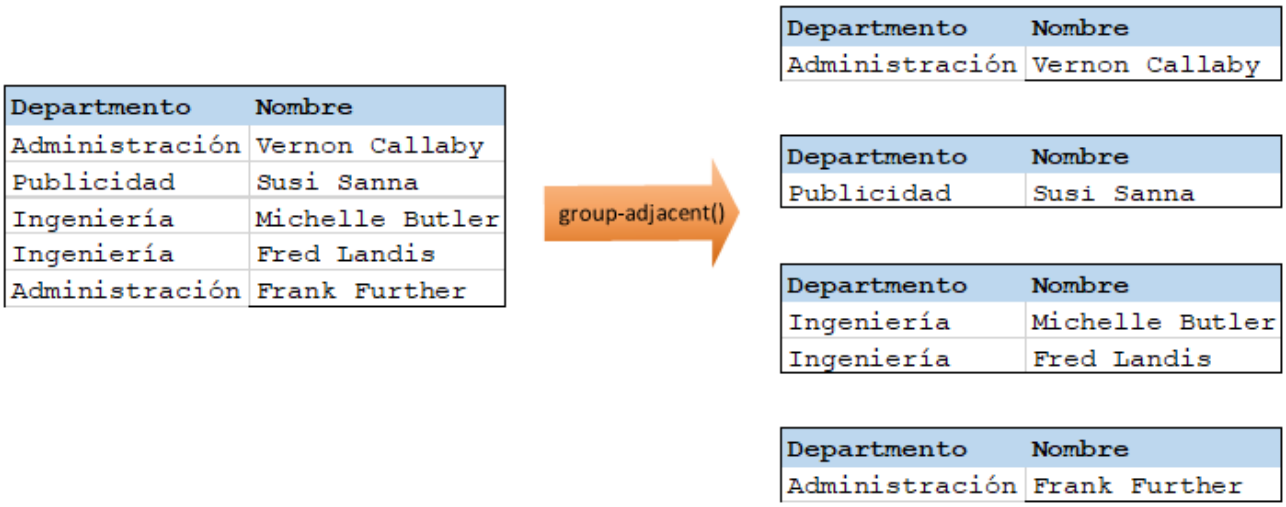
La función **group-adjacent** necesita una clave de agrupación como argumento, de forma parecida a como ocurre con la función `group-by`. Al contrario que esta, sin embargo, la función **group-adjacent** crea un grupo

nuevo siempre que la clave es distinta. Si dos registros adyacentes tienen la misma clave los colocará en el mismo grupo.

Por ejemplo, en la imagen siguiente, que ilustra una transformación abstracta, la clave de agrupación es "Department". El lado izquierdo del diagrama muestra los datos de entrada, mientras que el lado derecho muestra los datos de salida tras la agrupación. Al ejecutar la transformación ocurre lo siguiente:

- Inicialmente, la primera clave, "Administración", crea un grupo nuevo.
- La siguiente clave es distinta, así que se crea un segundo grupo, "Publicidad".
- Como la tercera clave también es diferente, se crea otro grupo, "Ingeniería".
- La cuarta clave es la misma que la tercera, por lo que este registro se coloca en el grupo que ya existe para esa clave.
- Por último, la quinta clave es diferente de la cuarta, lo que hace que se cree un último grupo.

Como se ve en la siguiente imagen, "Michelle Butler" y "Fred Landis" están en el mismo grupo porque tienen la misma clave y son adyacentes. Sin embargo, "Vernon Callaby" y "Frank Further" están en grupos distintos porque no son adyacentes, aunque tengan la misma clave.

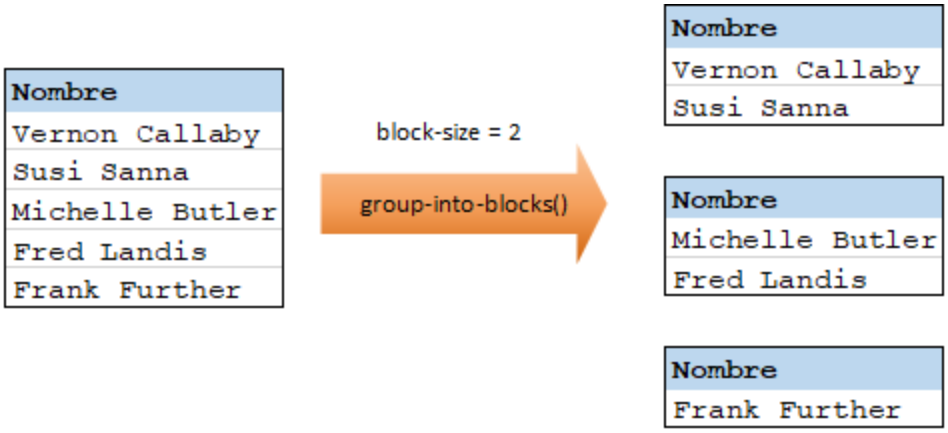


Para más información consulte la referencia a la función group-adjacent 687.

group-into-blocks

La función **`group-into-blocks`** crea grupos iguales que contienen exactamente N elementos, donde N es el valor que se le da al argumento `block-size`. Observe que el último grupo puede contener N elementos o menos en función de la cantidad de elementos de la fuente. En el ejemplo siguiente, `block-size` es 2. Como hay cinco elementos en total, cada grupo contiene exactamente dos elementos, excepto el último, que solo puede contener uno.

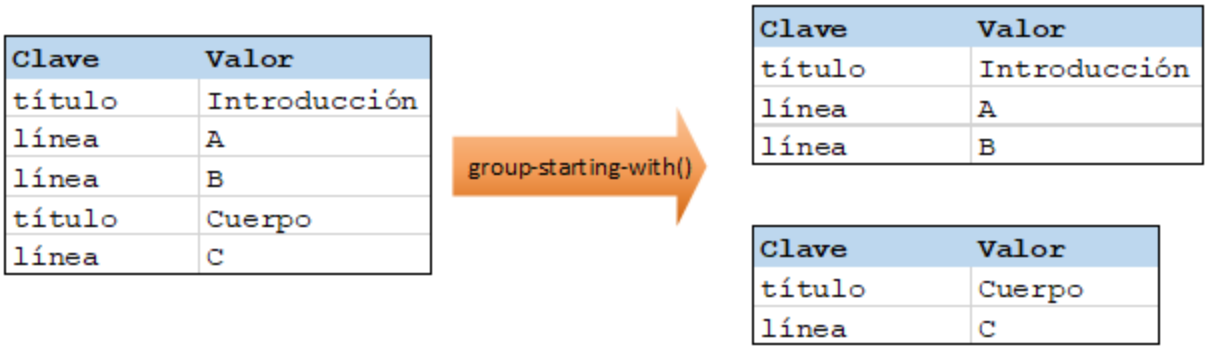
 © 2016-2022 Altova GmbH



Para más información consulte la referencia a la función group-into-blocks 692.

group-starting-with

La función **group-starting-with** toma como argumento una condición booleana. Si se cumple la condición booleana se crea un grupo nuevo que empieza con el registro que cumple esa condición. En el siguiente ejemplo la condición es que la clave "Clave" debe ser "título". Esta condición se cumple en el caso de los registros primero y cuarto, así que se crean dos grupos:



Nota: si existen registros antes del primero que cumple la condición, se crea un grupo más. Por ejemplo, si hubiera más registros "línea" antes del último registro "título", estos se colocarían en ese grupo adicional.

Para más información consulte la referencia a la función group-starting-with 694.

group-ending-with

La función **group-ending** toma como argumento una condición booleana. Si la condición booleana se cumple, se crea un grupo nuevo en el que el último registro es el que cumple esa condición. En el siguiente ejemplo, la condición es que la clave sea "cola". Esta condición se cumple en los registros tercero y quinto, por lo que se crean dos grupos:

Nota: si existen registros después del último que cumple la condición, se crea un grupo más. Por ejemplo, si hubiera más registros "línea" después del último registro "cola", estos se colocarían en ese grupo adicional.

Para más información consulte la referencia a la función group-ending-with 691.

5.5.1 Ejemplo: agrupar registros por clave

Este ejemplo muestra cómo agrupar registros con la ayuda de la función `group-by` e ilustra cómo añadir datos. Este ejemplo viene acompañado por una asignación de muestra que encontrará en la siguiente ruta: **<Documentos>\Altova\MapForce2023\MapForceExamples\GroupTemperaturesByYear.mfd**. Esta asignación lee datos de un archivo XML que contiene un registro de temperaturas mensuales, como se puede ver en el siguiente código de ejemplo:

```
<Temperatures>
   <data temp="-3.6" month="2006-01" />
   <data temp="-0.7" month="2006-02" />
   <data temp="7.5" month="2006-03" />
   <data temp="12.4" month="2006-04" />
   <data temp="16.2" month="2006-05" />
   <data temp="19" month="2006-06" />
   <data temp="22.7" month="2006-07" />
   <data temp="23.2" month="2006-08" />
   <data temp="18.7" month="2006-09" />
   <data temp="11.2" month="2006-10" />
   <data temp="9.1" month="2006-11" />
   <data temp="0.8" month="2006-12" />
   <data temp="-3.2" month="2007-01" />
   <data temp="-0.3" month="2007-02" />
   <data temp="6.5" month="2007-03" />
   <data temp="10.6" month="2007-04" />
   <data temp="19" month="2007-05" />
   <data temp="20.3" month="2007-06" />
   <data temp="22.3" month="2007-07" />
   <data temp="20.7" month="2007-08" />
   <data temp="19.2" month="2007-09" />
   <data temp="12.9" month="2007-10" />
```

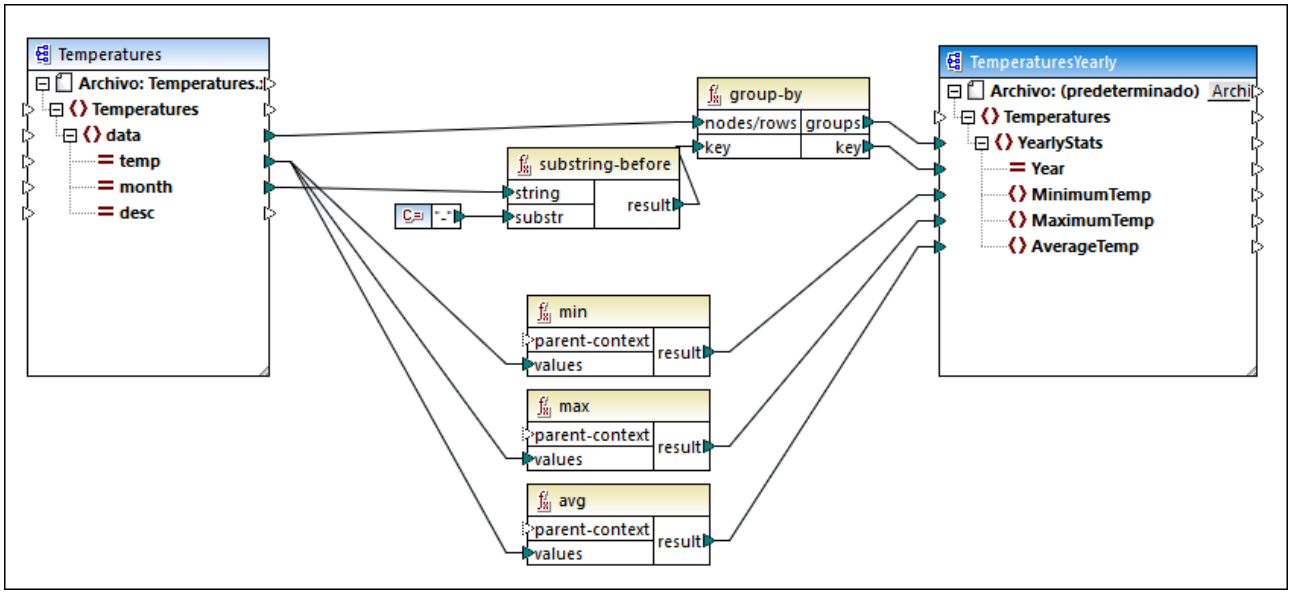
 © 2016-2022 Altova GmbH

```
    <data temp="8.1" month="2007-11" />
    <data temp="1.9" month="2007-12" />
</Temperatures>
```

El requisito empresarial de esta asignación es doble:

1. Agrupar todas las temperaturas de cada año.
2. Averiguar las temperaturas mínima, máxima y media de cada año.

Para conseguir el primer requisito la asignación llama a la función **group-by** 689. Para alcanzar el segundo llama a las funciones de agrupación **min** 637, **max** 636 y **avg** 634.Todas estas funciones son funciones integradas de MapForce y las puede añadir a cualquier aplicación arrastrándolas desde la ventana Bibliotecas (véase Agregar una función integrada a una asignación de datos 550).



GroupTemperaturesByYear.mfd

Para ejecutar una asignación, MapForce empieza por el elemento superior del componente de destino. Este también es el modo recomendado para empezar a leer una. En este ejemplo se creará un elemento **YearlyStats** por cada grupo que devuelva la función **group-by**. Esta función toma como primer argumento todos los elementos **data** del origen y los agrupa en función de las conexiones que existan con la entrada de **key**. El requisito es agrupar las temperaturas por año, por lo que primero debemos obtener el año. Para ello, la función **substring-before** 714 extrae la parte que contiene el año del atributo **month** de cada elemento **data**. Es decir, toma como argumento el valor de **month** y devuelve la parte que encuentre antes de la primera instancia de **substr**. Como se ilustra más arriba, en este ejemplo **substr** equivale al signo menos o guion, por lo que si tenemos el valor "2006-01", la función devolverá "2006".

Por último, los valores **MinimumTemp**, **MaximumTemp** y **AverageTemp** se obtienen conectando estos elementos con las correspondientes funciones agregadas: **min**, **max** y **avg**. Estas tres funciones toman como entrada la secuencia de temperaturas que leen en el componente de origen. Estas funciones no necesitan un argumento **parent-context** porque ya funcionan en el contexto de cada grupo. En otras palabras, la conexión que existe entre **data** y **YearlyStats** da el contexto en el que debe trabajar cada función agregada.

Para obtener una vista previa de los resultados de la asignación, haga clic en la pestaña *Resultados*. Observe que el número de grupos coincide con el número de años obtenido al leer el archivo de origen. Por ejemplo:

```
<Temperatures>
   <YearlyStats Year="2006">
      <MinimumTemp>-3.6</MinimumTemp>
      <MaximumTemp>23.2</MaximumTemp>
      <AverageTemp>11.375</AverageTemp>
   </YearlyStats>
   <YearlyStats Year="2007">
      <MinimumTemp>-3.2</MinimumTemp>
      <MaximumTemp>22.3</MaximumTemp>
      <AverageTemp>11.5</AverageTemp>
   </YearlyStats>
</Temperatures>
```

Nota: para simplificar, el código de ejemplo anterior contiene menos datos que los datos de entrada y salida utilizados en la asignación de ejemplo.

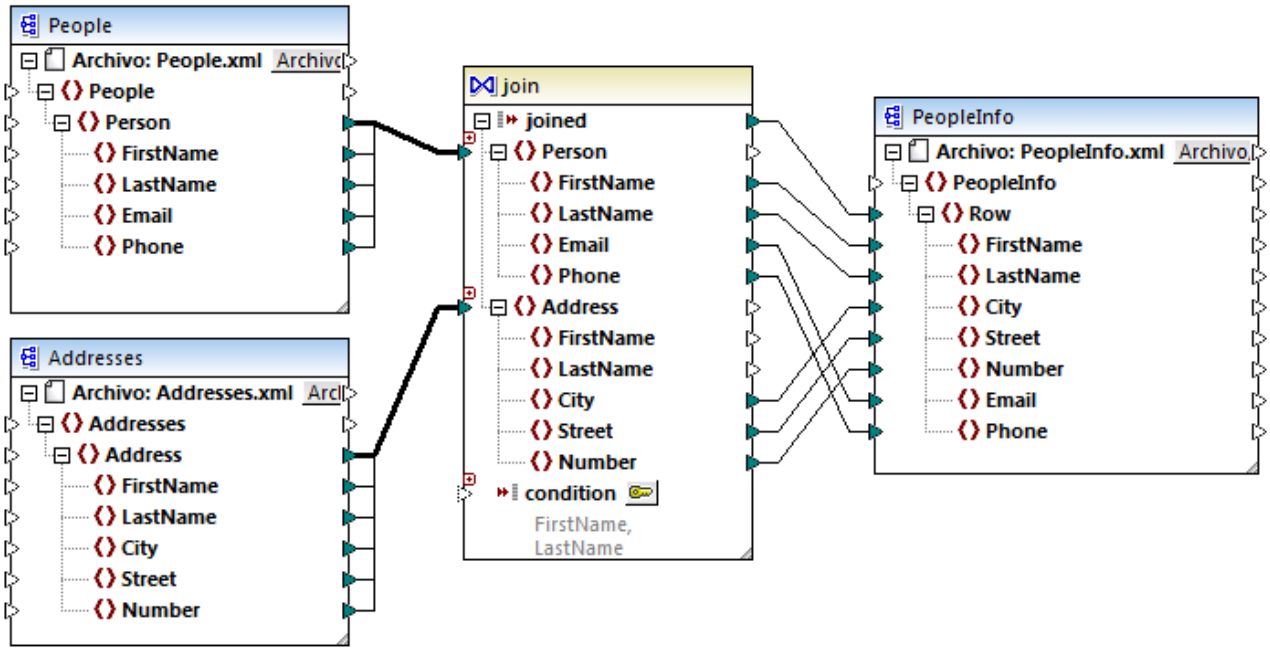
© 2016-2022 Altova GmbH

5.6 Combinar datos

A veces es necesario combinar datos de dos o más estructuras en base a una condición (p. ej. si el campo A de la primera estructura tiene el mismo valor que el campo B de la segunda estructura). Para este tipo de diseños puede usar un componente de combinación.

Un componente de combinación es un componente de MapForce que permite unir varias estructuras en la asignación de acuerdo con las condiciones definidas por el usuario. Devuelve la asociación (conjunto combinado) de elementos que cumplen con la condición. Las combinaciones son de gran ayuda a la hora de combinar datos de dos estructuras que tienen un campo en común (p. ej. una identidad).

Por ejemplo, en la asignación que aparece más abajo, el componente del medio es un componente de combinación. En este diseño de asignación se combinan dos estructuras XML (una lista de personas y una lista de direcciones). El objetivo es recopilar todos los datos personales de cada persona en un archivo XML de destino. Los campos `FirstName` y `LastName` hacen de claves de combinación. Es decir, si `FirstName` y `LastName` (secundarios de `Person`) tienen el mismo valor que `FirstName` y `LastName` (secundarios de `Address`), entonces los datos de la dirección pertenecen a la misma persona y, por tanto, se combinan. Los nodos de la estructura combinada se pueden seguir procesando en un componente de destino (en este caso un archivo XML). La condición de combinación propiamente dicha se define en las propiedades del componente de combinación con el botón **Definir condición de combinación** . Este ejemplo viene con una asignación de muestra y se explica con más detalla en el apartado Ejemplo: combinar estructuras XML 523.



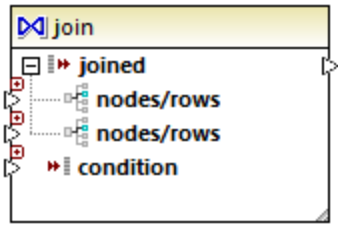
JoinPeopleInfo.mfd

Como puede ver en la imagen, las estructuras de origen y el componente de combinación están conectadas por medio de una conexión de copia total, para reducir la acumulación de elementos en el área de diseño. En general, este tipo de conexiones se crean automáticamente en MapForce si el contexto es relevante (*véase* *Conexiones de copia total* 95).

Las estructuras que se deben combinar pueden ser de componentes distintos (como en el ejemplo anterior) o pueden pertenecer al mismo componente. Las estructuras que se deben combinar también pueden ser de tipos distintos (p. ej. una estructura XML y una tabla de BD). Consulte el apartado Combinar datos de base de datos 335 para aprender cómo funciona la combinación en las bases de datos.

Para agregar un componente de combinación:

1. Active el lenguaje de transformación BUILT-IN (haciendo clic en el botón de la barra de herramientas o con el comando de menú **Resultados | Motor de ejecución integrado**).
2. Haga clic en el comando de menú **Insertar | Combinación** o en el botón **Combinación** de la barra de herramientas. El componente de combinación aparece en la asignación y por defecto acepta datos de dos estructuras, así que tiene dos entradas `nodes/rows`. Si lo necesita, puede agregar entradas nuevas en la combinación con ayuda del botón **Agregar entrada** (*véase Combinar tres o más estructuras* 522).



3. Conecte las estructuras que se deben combinar con las entradas `nodes/rows` del componente de combinación.
4. Añada la condición para la combinación (o varias condiciones). Esto se consigue haciendo clic con el botón derecho en el componente de combinación y seleccionando **Propiedades**. También puede añadir condiciones de combinación desde la asignación directamente: conecte el resultado booleano de una función con la entrada `condition` del componente de combinación. Cuando se están combinando tablas de base de datos, MapForce puede crear condiciones de combinación automáticamente (*véase Agregar condiciones de combinación* 519).

Notas:

- Se admite el uso de componentes de combinación cuando el lenguaje de destino de la asignación es el motor de ejecución integrado (BUILT-IN). No es compatible con la generación de código C#, C++ o Java.
- Cuando una estructura no es una fuente de entrada válida ni compatible para la combinación, MapForce emite sugerencias en la asignación directamente o, cuando se valide la asignación, en la ventana Mensajes (*véase Validar asignaciones* 105).
- No se deberían conectar componentes de combinación con parámetros de entrada ni resultados de funciones definidas por el usuario inline. Si existen dichas conexiones, se producirán errores de validación.
- Cuando conecte componentes de base de datos (tablas o vistas, por ejemplo) a un componente de combinación directamente, en la esquina superior derecha del componente de combinación se habilitará el botón **Modo SQL**. Si se activa, este botón ofrece funciones SQL especiales para trabajar con la operación de combinación (*véase Combinaciones en modo SQL* 336).
- No se puede conectar la salida del nodo `joined` con otro componente de combinación, pero si lo necesita puede conectar un resultado parcial de una combinación con otra.

 © 2016-2022 Altova GmbH

Diferencias entre componentes de combinación y otros tipos de componentes

En algunos casos se puede conseguir el mismo resultado usando variables o filtros complejos en lugar de componentes de combinación (*véase Usar variables* 492 *y Filtros y condiciones* 543). Sin embargo, a diferencia de otros componentes, los de combinación facilitan la comprensión de la asignación porque permiten distinguir a simple vista qué datos se combinan. Además, si se habilita el modo SQL en el componente de combinación, el rendimiento de la asignación aumenta considerablemente cuando se trata de combinaciones de bases de datos (*véase Combinar datos de base de datos* 335).

Agregar un contexto primario

En casos especiales para poder conseguir un resultado concreto se podrá ofrecer explícitamente un contexto de asignación (el denominado *contexto primario*) para los datos que están conectados con el componente de combinación. Para agregar un contexto primario haga clic con el botón derecho en el nodo `joined` del componente de combinación y seleccione **Agregar contexto primario** en el menú contextual. A partir de ese momento, el componente de combinación incluirá una entrada nueva llamada `parent-context` donde podrá conectar el nodo de origen correspondiente. Consulte Reemplazar el contexto de la asignación 879 para obtener más información.

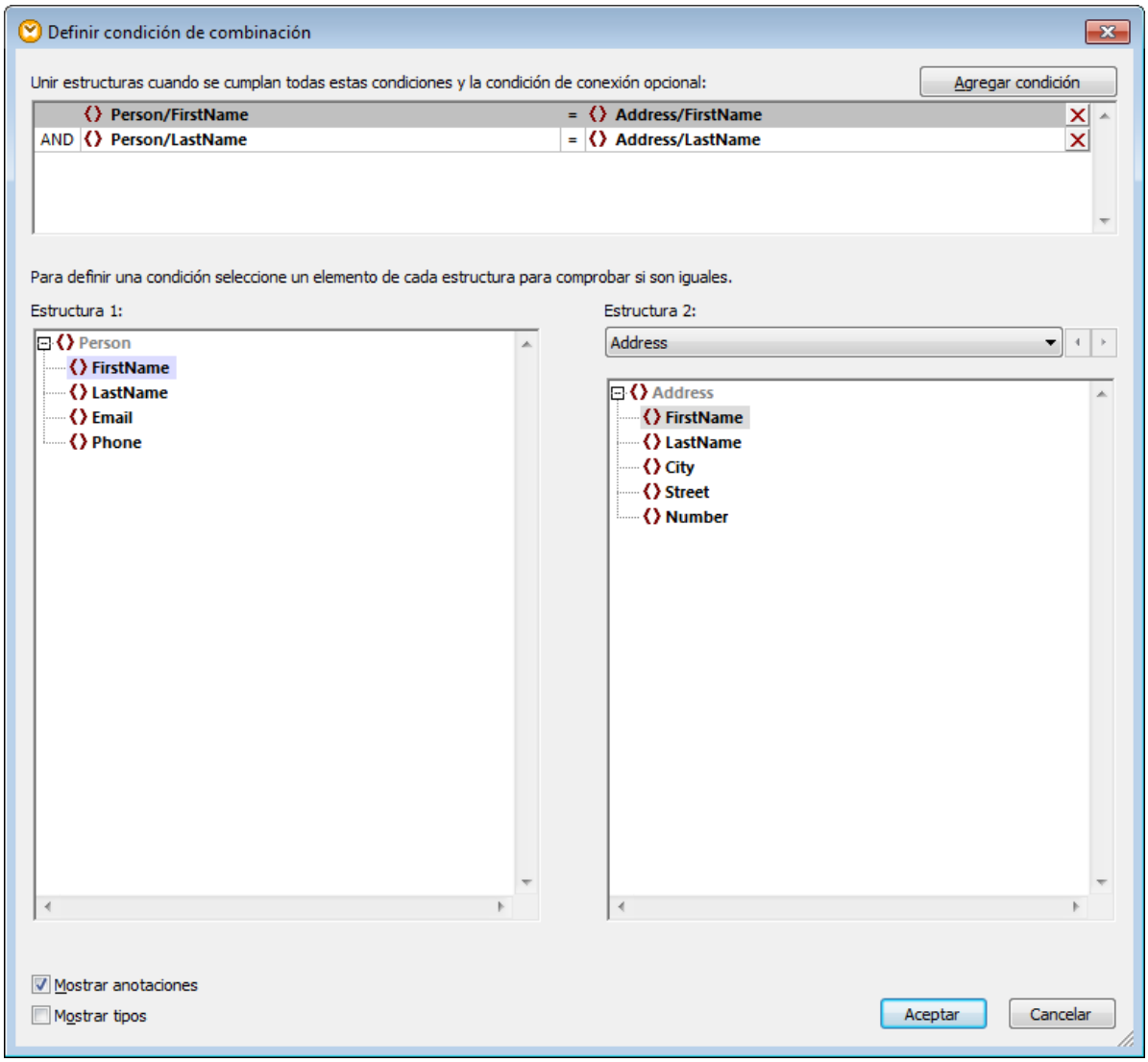
5.6.1 Agregar condiciones de combinación

Una combinación une elementos de dos o más estructuras en base a una condición así que una combinación siempre necesita una condición como mínimo. Hay varias maneras de agregar condiciones de combinación, tal y como explicamos más abajo.

Nota: cuando se combinan tablas de base de datos en modo SQL, MapForce crea las condiciones de combinación automáticamente a partir de las relaciones de clave foránea detectadas entre las tablas. Para que MapForce pueda crear condiciones de combinación automáticas las tablas de la base de datos deben estar en una relación secundario/primario en el componente de MapForce (es decir, una tabla debe ser el primario o secundario de otra tabla en el componente). Consulte el apartado Ejemplo: combinar tablas en modo SQL 344 para obtener más información.

Método 1: agregar una condición de combinación desde las propiedades del componente

1. En la asignación primero debe comprobar que como mínimo hay dos estructuras o tablas de BD conectadas al componente de combinación. (El componente de combinación de este ejemplo forma parte de la asignación **JoinPeopleInfo.mfd**, disponible en la carpeta **<Documentos>\Altova\MapForce2023\MapForceExamples\Tutorial**. Esta asignación se describe más detalladamente en el apartado Ejemplo: combinar estructuras XML 523.
2. Haga clic en el botón **Definir condición de combinación** del componente de combinación o haga clic con el botón derecho en el título del componente y seleccione **Propiedades**.
3. Seleccione un elemento de la estructura de la izquierda y otro de la estructura de la derecha (es decir, cuando la comparación de este par dé true como resultado, las estructuras se combinarán).



Si necesita agregar más condiciones haga clic en **Agregar condición** y después seleccione un nuevo par de elementos. Por ejemplo, en la imagen anterior se definieron dos condiciones de combinación:

1. `FirstName` en Estructura 1 debe ser igual a `FirstName` en Estructura 2 y
2. `LastName` en Estructura 1 debe ser igual a `LastName` en Estructura 2.

Para eliminar una condición de combinación haga clic en el botón **Eliminar** de la condición.

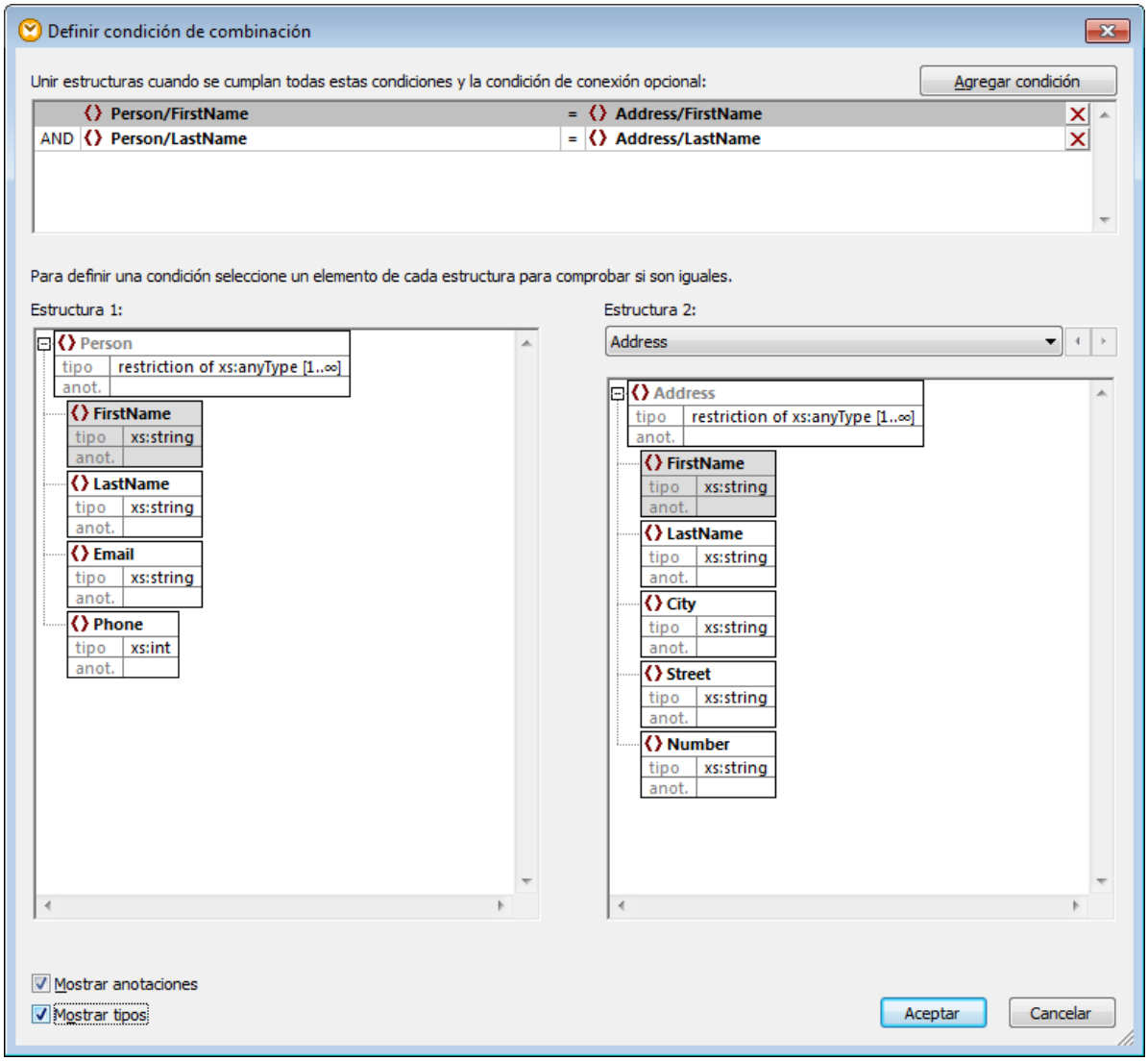
Notas:

- Cuando existen varias condiciones de combinación, todas deben cumplirse o de lo contrario no se combinarán las dos estructuras. En otras palabras, cuando existen varias condiciones, éstas se unen con la operación lógica AND. Esto también incluye condiciones opcionales que se añadieran desde la asignación (ver método 2 más abajo).
- Si hay más de dos estructuras conectadas al componente de combinación, esas estructuras adicionales aparecerán en la lista desplegable situada debajo de *Estructura 2.* Cuando seleccione una de estas estructuras adicionales en la lista desplegable, el panel izquierdo mostrará todas las

 © 2016-2022 Altova GmbH

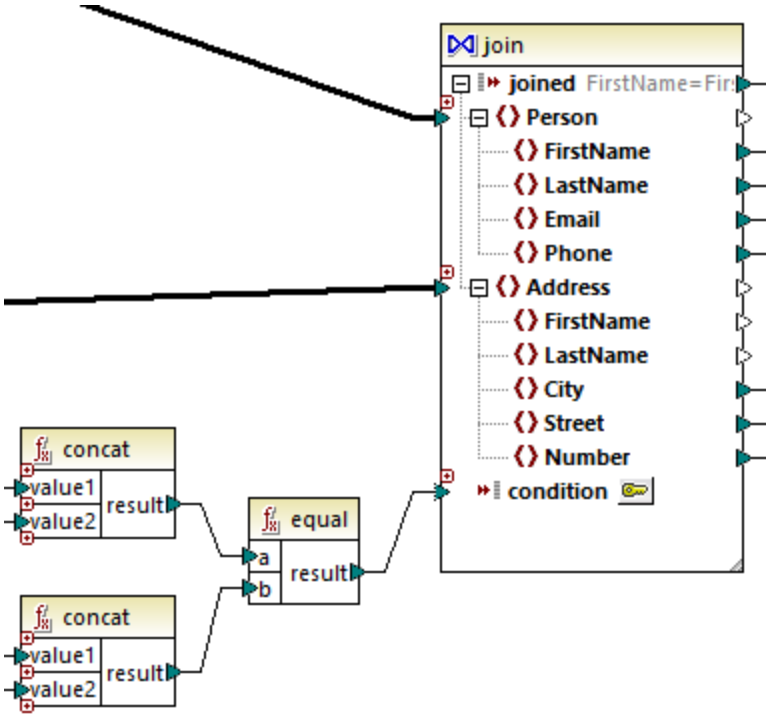
estructuras que aparecen antes de ella en el componente de combinación. Así podrá definir condiciones de combinación entre cualquier estructura. Para ver un ejemplo consulte Ejemplo: crear un informe CSV a partir de varias tablas 339.

- Para ver el tipo de datos de los elementos de cada estructura marque la casilla *Mostrar tipos*. La opción *Mostrar anotaciones* sirve para ver información adicional sobre los elementos (siempre y cuando dicha información exista en el esquema o base de datos subyacente). Si marca ambas casillas, la presentación de los datos cambia ligeramente en el cuadro de diálogo:



Método 2: agregar una condición de combinación desde la asignación

- En la asignación añada componentes que producen un valor booleano y después conecte la salida booleana a la entrada del nodo `condition`. Por ejemplo, la función **`equal`** puede comparar un valor con un nodo de la asignación y aportar el resultado booleano como entrada del nodo `condition` del componente de combinación.



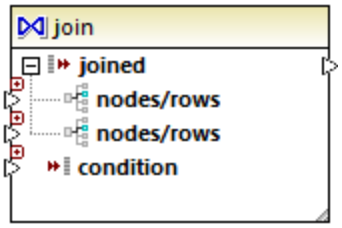
Nota: si no se define ninguna condición desde las propiedades del componente (método 1), debe conectarse el nodo `condition` del componente de combinación (método 2).

Método 3: método mixto

En la misma asignación se pueden definir algunas condiciones de combinación desde las propiedades del componente (método 1) y combinarlas con la condición definida en la asignación (método 2). No obstante, si intenta combinar tablas de BD en modo SQL, las condiciones solamente se podrán definir con el método 1 (*véase* *Combinaciones en modo SQL* 336).

5.6.2 Combinar tres o más estructuras

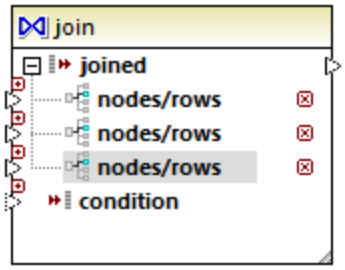
Si se añade un componente de combinación a la asignación con el comando de menú Insertar **| Combinación**, el componente acepta dos estructuras por defecto (es decir, el componente solo contiene dos entradas `nodes/rows`).



Si necesita combinar más de dos estructuras, haga clic en el botón **Agregar entrada** y cree tantos `nodes/rows` como necesite. Si necesita eliminar una entrada `nodes/rows`, haga clic en el botón **Eliminar**

 © 2016-2022 Altova GmbH

entrada ⊠. Tenga en cuenta que un componente de combinación necesita dos estructuras como mínimo (y, por tanto, el botón ⊠ solo está disponible si existen más de dos entradas).



Cuando un componente de combinación tiene varias entradas, las condiciones de combinación deben tener en cuenta cada una de las entradas (*véase* *Agregar condiciones de combinación* 519). Para ver un ejemplo con instrucciones para aprender a combinar varias tablas de base de datos consulte el apartado Ejemplo: crear un informe CSV a partir de varias tablas 339.

5.6.3 Ejemplo: combinar estructuras XML

Con este ejemplo aprenderá a combinar datos de dos estructuras XML de forma condicional y con ayuda de un componente de combinación. La asignación utilizada para el ejemplo está en esta ruta de acceso: **<Documentos>\Altova\MapForce2023\MapForceExamples\Tutorial\JoinPeopleInfo.mfd**.

El objetivo de esta asignación es recopilar en un archivo XML de destino datos personales (nombre, apellido, dirección postal, dirección de correo electrónico y teléfono) procedentes de dos archivos XML de origen.

El primer archivo XML de origen contiene el nombre y apellido de cada persona, así como su dirección de correo electrónico y teléfono (observe que para simplificar se han omitido la declaración XML, los espacios de nombres y algunos registros):

```
<People>
   <Person>
      <FirstName>Marquita</FirstName>
      <LastName>Bailey</LastName>
      <Email>m.bailey@nanonull.com</Email>
      <Phone>555323698</Phone>
   </Person>
   <Person>
      <FirstName>Totie</FirstName>
      <LastName>Rea</LastName>
      <Email>t.rea@nanonull.com</Email>
      <Phone>555598653</Phone>
   </Person>
</People>
```

People.xml

El segundo archivo XML de origen contiene el nombre y apellido de cada persona, así como su dirección postal:

```
<Addresses>
   <Address>
      <FirstName>Marquita</FirstName>
      <LastName>Bailey</LastName>
      <City>Bridgedell</City>
      <Street>Olive Street</Street>
      <Number>4</Number>
   </Address>
   <Address>
      <FirstName>Totie</FirstName>
      <LastName>Rea</LastName>
      <City>Roseford</City>
      <Street>Evergreen Lane</Street>
      <Number>34</Number>
   </Address>
</Addresses>
```

Addresses.xml

El objetivo de la asignación es combinar la información de `<Person>` del primer archivo con la información de `<Address>` del segundo archivo, siempre y cuando coincidan el nombre y el apellido. Concretamente, por cada `<Person>` del primer archivo y por cada `<Address>` del segundo archivo, deben compararse `FirstName` y `LastName`. Si tienen valores idénticos, entonces los registros `<Person>` y `<Address>` correspondientes se refieren a la misma persona y se deben combinar. La estructura XML de destino tendría este aspecto:

```
<PeopleInfo>
   <Row>
      <FirstName>Marquita</FirstName>
      <LastName>Bailey</LastName>
      <City>Bridgedell</City>
      <Street>Olive Street</Street>
      <Number>4</Number>
      <Email>m.bailey@nanonull.com</Email>
      <Phone>555323698</Phone>
   </Row>
   <Row>
      <FirstName>Totie</FirstName>
      <LastName>Rea</LastName>
      <City>Roseford</City>
      <Street>Evergreen Lane</Street>
      <Number>34</Number>
      <Email>t.rea@nanonull.com</Email>
      <Phone>555598653</Phone>
   </Row>
</PeopleInfo>
```

PeopleInfo.xml

Esto se consigue fácilmente añadiendo un componente de combinación a la asignación. Recuerde que puede conseguir el mismo resultado usando otro tipo de componente, pero en este ejemplo usaremos un componente de combinación.

 © 2016-2022 Altova GmbH

Paso nº1: agregar los archivos XML de origen en la asignación

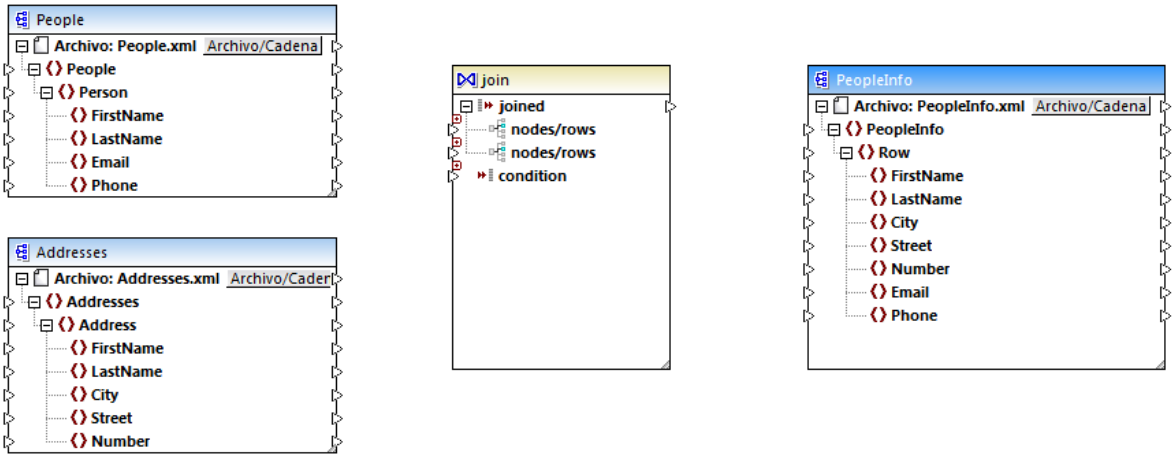
1. En el menú **Insertar** haga clic en el comando **Archivo o esquema XML** y navegue hasta este archivo de origen: **<Documentos>\Altova\MapForce2023\MapForceExamples\Tutorial\People.xml**.
2. Repita el paso anterior e inserte este archivo **<Documentos>\Altova\MapForce2023\MapForceExamples\Tutorial\Addresses.xml**.

Paso nº2: agregar el archivo de esquema de destino en la asignación

1. En el menú **Insertar** haga clic en el comando **Archivo o esquema XML** y navegue hasta este archivo: **<Documentos>\Altova\MapForce2023\MapForceExamples\Tutorial\PeopleInfo.xsd** (el archivo de esquema de destino).
2. Cuando la aplicación solicite un archivo XML de muestra, haga clic en **Omitir**.
3. Cuando la aplicación solicite un elemento raíz, seleccione `PeopleInfo` como elemento raíz.

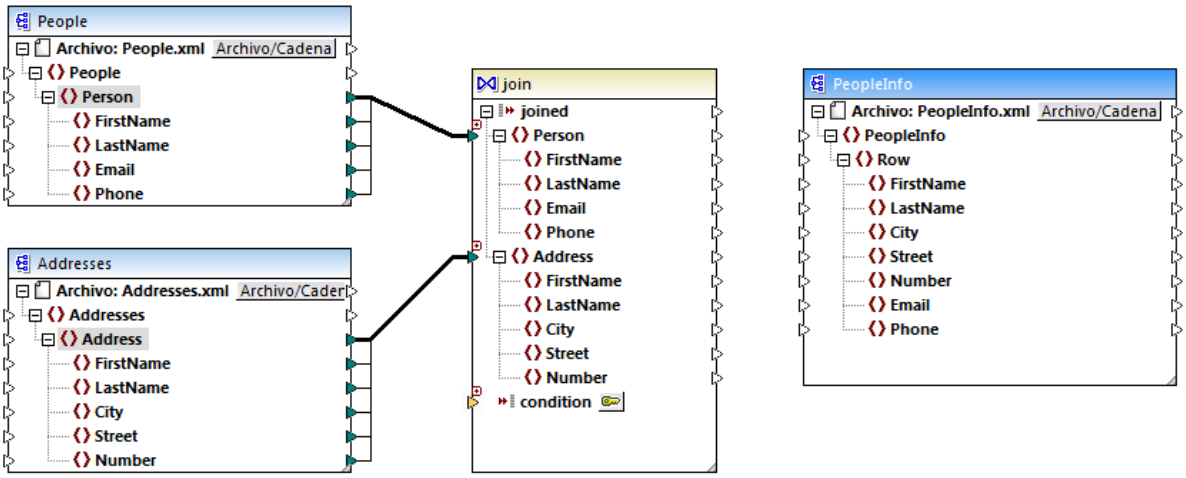
Paso nº3: agregar el componente de combinación

1. En el menú **Insertar** haga clic en el comando **Combinación** (o haga clic en el botón **Combinación** de la barra de herramientas). Llegados a este punto la asignación debería tener este aspecto (deberá arrastrar los componentes y ajustar su tamaño para conseguir este diseño):



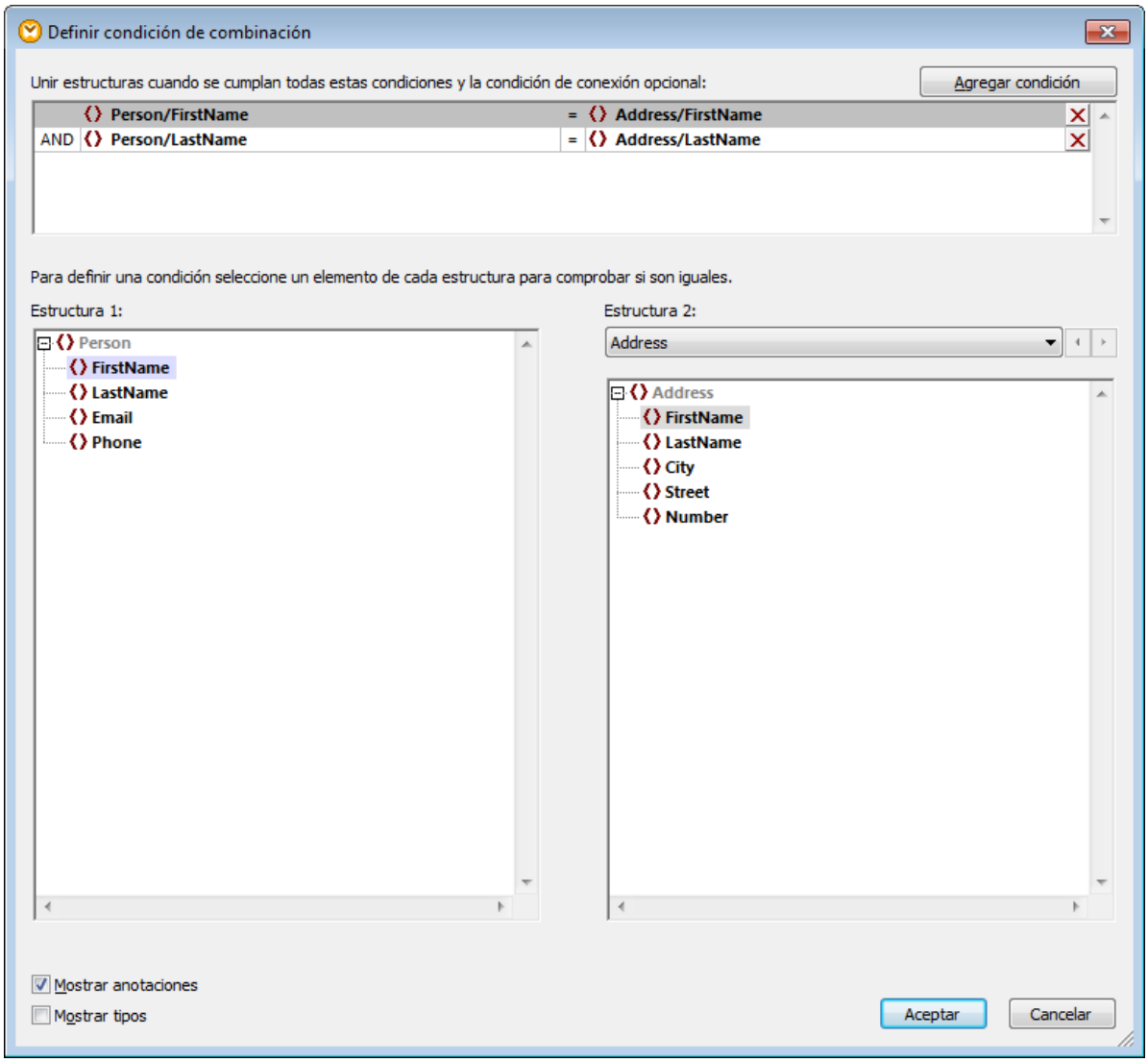
Fíjese en la estructura del componente de combinación. Tiene dos entradas `nodes/rows` para que podamos conectarlo a las dos estructuras que deseamos comparar (las estructuras `Person` y `Address` en este caso).

2. Dibuje una conexión entre `Person` y la primera entrada `nodes/rows` del componente de combinación. Igualmente, conecte `Address` con la otra entrada `nodes/rows`.



3. Como dijimos anteriormente, la combinación solo debería tener lugar si `FirstName` y `LastName` tienen valores idénticos en ambas estructuras. Para definir esta condición haga clic en el botón **Definir condición de combinación** del componente.
4. Seleccione el par de elementos que definen la primera condición de combinación (`FirstName` en Estructura 1 y `FirstName` en Estructura 2).
5. Haga clic en **Agregar condición** y repita el paso anterior con `LastName` esta vez.

© 2016-2022 Altova GmbH



En algunas asignaciones puede ser suficiente una condición formada por una comparación. Sin embargo, en este ejemplo es importante crear dos comparaciones:

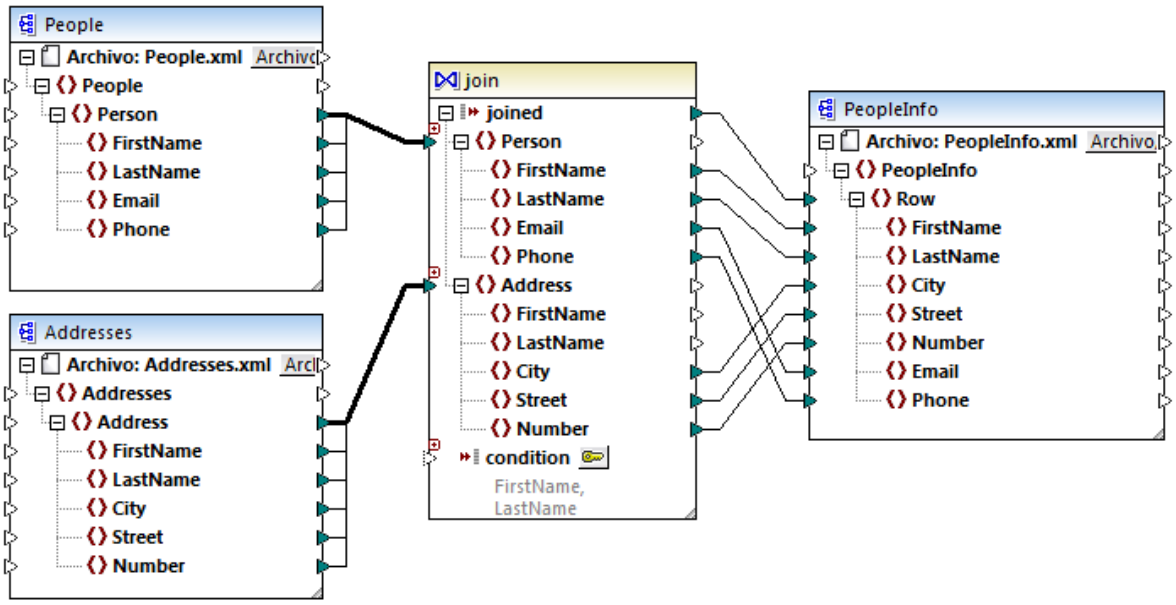
1) `FirstName` en Estructura 1 = `FirstName` en Estructura 2
2) `LastName` en Estructura 1 = `LastName` en Estructura 2.

Cuando se definen varias condiciones, todas ellas deben cumplirse. De lo contrario no tiene lugar la combinación. Por tanto, en este ejemplo, la combinación solo tendrá lugar si las dos comparaciones dan como resultado true. Si, por el contrario, solo se definiera una comparación, podrían combinarse los datos de personas que tienen el mismo nombre pero distinto apellido.

Paso nº4: asignar el componente de combinación al esquema de destino

Ahora que se combinaron las dos estructuras, ya puede definir qué elementos de la estructura combinada se deben asignar al componente de destino. Esto se hace creando conexiones entre los elementos de la estructura combinada y los del componente de destino (*imagen siguiente*). La conexión que une `joined` con

Row tiene este propósito: cuando se cumpla la condición de combinación, se crea un elemento Row nuevo en el componente de destino.



Para consultar una vista previa de los resultados de la asignación abra el panel *Resultados*. Como se esperaba, ahora los registros de las personas (<Row>) incluyen también la dirección postal.

 © 2016-2022 Altova GmbH

5.7 Añadir excepciones

Una excepción es un tipo especial de componente que permite detener el proceso de asignación de datos y emitir un error cuando se cumple una condición devuelta por un filtro. Puede añadir una excepción si su asignación de datos incluye un filtro que comprueba si se cumple una condición true/false (*véase* *Filtros y condiciones* 543). Por ejemplo, puede diseñar la asignación para que emita una excepción si el valor de algún elemento de la asignación es superior a un valor predefinido.

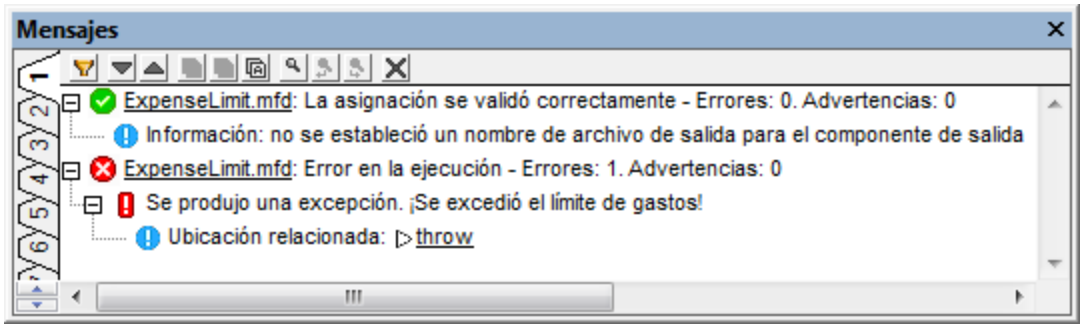
Para añadir una excepción a la asignación:

1. En el menú **Insertar** haga clic en el comando **Excepción** (o haga clic en el icono **Insertar excepción** de la barra de herramientas).
2. Conecte la entrada **throw** de la excepción con la salida **on-true** o con la salida **on-false** de un filtro.
3. Si lo desea, puede conectar la entrada **error-text** de la excepción con otro componente (por lo general una constante) que suministre el texto del error que debe aparecer cuando se emita la excepción.

Nota: las dos salidas del filtro (**on-true** y **on-false**) deben estar conectadas. Concretamente, una de estas dos salidas debe estar conectada con la excepción directamente (sin funciones ni otros componentes de por medio). La otra salida debe estar conectada con el componente de destino, directamente o a través de componentes intermedios.

Cuando la asignación se encuentra con una excepción, aparece una notificación:

- En MapForce aparece un error en la ventana Mensajes, acompañado del texto de la excepción (en este caso "Se excedió el límite de gastos".).



  Si el lenguaje de la asignación es XSLT 2.0 o XQuery, en la ventana Mensajes aparece el error *Error en la ejecución* y se abre el panel correspondiente (*XSLT 2.0* o *XQuery*). La línea del error se resalta en la ventana Mensajes.

- Si ejecuta la asignación con MapForce Server, se devuelve el error "Se produjo una excepción" seguido del texto que se definió en MapForce ("Se excedió el límite de gastos").

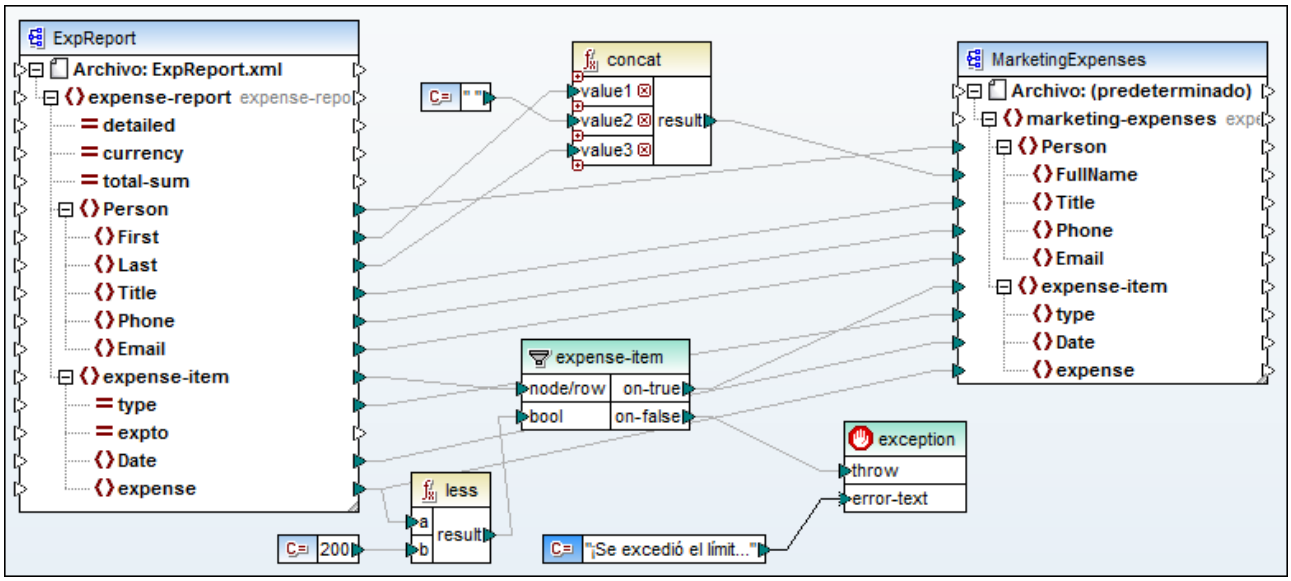
```
c:\codegen\mfx\ExpenseLimit>"C:\Program Files (x86)\Altova\MapForceServer2016\bi
n\MapForceServer.exe" run ExpenseLimit.mfx
Exception was thrown! Expense limit exceeded!
```

- Si ejecuta la asignación a partir del código C#, C++ o Java generado con MapForce, se devuelve el error "EXCEPCIÓN DE USUARIO" seguido del texto que se definió en MapForce ("Se excedió el límite de gastos").

```
C:\codegen\cs\ExpenseLimit\Mapping\bin\Debug>Mapping.exe
Mapping Application
USER EXCEPTION: Expense limit exceeded!
```

5.7.1 Ejemplo: excepción en la condición Greater than

En este ejemplo podemos ver una asignación de datos que emite una excepción cuando se cumple una condición Greater than. Se trata del diseño de asignación de datos **<Documentos>\Altova\MapForce2023\MapForceExamples\ExpenseLimit.mfd**.



Esta asignación de datos emite una excepción si el elemento **expense** de la instancia XML de origen tiene un valor superior a 200. El valor "200" viene dado por una constante. La función `less` se utiliza para comparar los dos valores. Si el valor de **expense** es inferior a 200, entonces se pasa su elemento primario (**expense-item**) al filtro y la asignación no emite ninguna excepción. Si por el contrario el valor de **expense** es superior a 200, se emite una excepción con el texto predefinido "Se excedió el límite de gastos".

Como puede ver en la imagen la excepción se identifica con el icono y se compone de dos elementos: **throw** y **error-text**. El elemento **throw** debe estar conectado con la salida **on-false** o **on-true** de un filtro. El elemento **error-text** está conectado con una constante que suministra el texto de la excepción.

Lo más importante es saber que las dos salidas del filtro deben estar conectadas. De lo contrario no se podrá emitir una excepción. En nuestro ejemplo, la salida **on-false** está conectada con la excepción, mientras que la salida **on-true** está conectada con el componente de destino.

5.7.2 Ejemplo: excepción cuando un nodo no existe

En este ejemplo podemos ver una asignación de datos que emite una excepción cuando no existe un nodo concreto en el esquema XML de origen. Para simplificar el ejemplo, la asignación usa el mismo esquema XML en el componente de origen y de destino.

 © 2016-2022 Altova GmbH

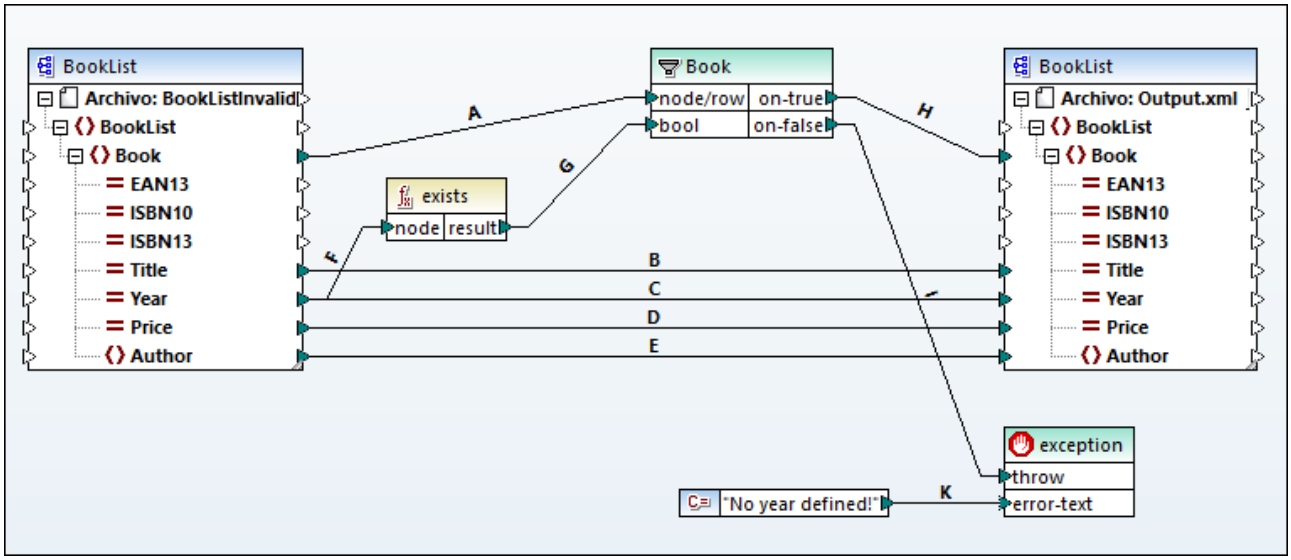
Para añadir el esquema de origen a la asignación:

1. En el menú **Insertar** haga clic en **Archivo o esquema XML...** y navegue hasta el archivo **<Documentos>\Altova\MapForce2023\MapForceExamples\BookList.xsd**.
2. Cuando la aplicación solicite un archivo de instancia, haga clic en **Omitir**.
3. Cuando la aplicación solicite un elemento raíz, seleccione el elemento raíz **BookList**.

Para añadir el esquema de destino siga los mismos pasos. Después, con los comandos correspondientes del menú **Insertar** (o con los botones correspondientes de la barra de herramientas), añada estos componentes:

- Un **Filtro: nodos/filas** (*véase* *Filtros y condiciones* 543)
- Una constante con el texto "No se definió el año"
- Una excepción

Por último, arrastre la función `exists` del panel Bibliotecas hasta el área de asignación y cree las conexiones que aparecen en esta imagen:



Según lo definido en el esquema XML, todos los atributos del elemento Book son opcionales, excepto el título del libro (Title). Por tanto, en una instancia XML válida el atributo Year puede existir o no. El propósito de la asignación es procesar correctamente una instancia XML en la que todos los libros tienen el atributo Year. Si un libro no contiene este atributo, la asignación emitirá una excepción.

Para probar la ejecución de la asignación:

1. Haga doble clic en el título del componente de origen y en el campo *Archivo XML de entrada* navegue hasta este archivo: **<Documentos>\Altova\MapForce2023\MapForceExamples\BookList.xml**.
2. Haga clic en el panel *Resultados* para ejecutar la asignación.

Para probar la excepción:

1. En el mismo directorio cree una copia del archivo **BookList.xml** llamada **BookListInvalid.xml**.
2. Edite el archivo **BookListInvalid.xml** eliminando el atributo Year de algún elemento Book.
3. Haga doble clic en el título del componente de origen y en el campo *Archivo XML de entrada* navegue hasta el archivo **BookListInvalid.xml**.
4. Haga clic en el panel *Resultados* para ejecutar la asignación.

Vemos ahora cómo funciona la asignación.

Por cada libro de la instancia de origen, la **conexión A** crea un libro en la instancia de destino.

Por cada libro del componente de origen, las **conexiones B, C, D y E** se encargan de copiar Title, Year, Price y Author en el componente de destino.

La **conexión F** desencadena la función `exists`, encargada de comprobar si existe o no el atributo "Year". La **conexión G** pasa al filtro el resultado de la función (`true` o `false`). Si el resultado es `true`, significa que el atributo "Year" existe y el libro se pasa al filtro y, en consecuencia, al destino por medio de la **conexión H**.

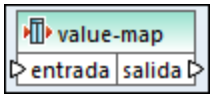
> Observe que el filtro no está conectado directamente con la salida **Year** del componente de origen. Si así fuera, el filtro filtraría el año sin más y la excepción no se emitiría.

La **conexión I** existe porque la excepción debe estar conectada con una salida **on-false** o **on-true** de un filtro. Y, por último, la **conexión K** pasa el texto del error desde la constante hasta el componente de excepción.

© 2016-2022 Altova GmbH

5.8 Usar asignaciones de valores

Con los componentes `value-map` puede reemplazar un valor por otro con la ayuda de una tabla de consulta predefinida. Este tipo de componente solo procesa un valor cada vez, por lo que en la asignación tiene una **entrada** y un **resultado**.



Los componentes `value-map` son útiles si quiere asignar elementos individuales dentro de dos conjuntos para reemplazar elementos. Por ejemplo, puede expresar los días de la semana como números (1, 2, 3, 4, 5, 6 y 7) y asignarlos a los nombres de los días ("lunes", "martes", "miércoles", etc.). También puede asignar los nombres de los meses ("enero", "febrero", "marzo", etc.) a la representación numérica de los mismos (1, 2, 3, 4, etc.). Cuando ejecute la asignación los valores que coincidan serán reemplazados según su tabla personal de consulta. Los valores de estos dos conjuntos pueden ser de distinto tipo, pero cada conjunto debe almacenar valores del mismo tipo de datos.

Los componentes `value-map` sirven para realizar consultas simples en las que cada valor del primer conjunto corresponde a un único valor del segundo conjunto. Si no se encuentra algún valor en la tabla de consulta puede optar por reemplazarlo con el valor que quiera, con un valor vacío o pasarlo tal y como esté. Si necesita buscar o filtrar valores en base a un criterio más complejo, es mejor que use uno de los componentes de filtrado 543.

Es importante subrayar que cuando se genera código o se compila un archivo de ejecución de MapForce Server desde la asignación, la tabla de consulta está incrustada en el código o archivo generado. En consecuencia, solo recomendamos que defina una tabla de consulta directamente en la asignación si los datos de esa tabla no cambian a menudo y si no se trata de una gran cantidad de datos (menos de unas cien entradas). Si los datos de la tabla de consulta cambian a menudo, puede que resulte difícil mantener al día tanto la asignación como el código generado. En este caso sería más sencillo guardar la tabla de consulta en formato texto, XML, BD o incluso Excel.

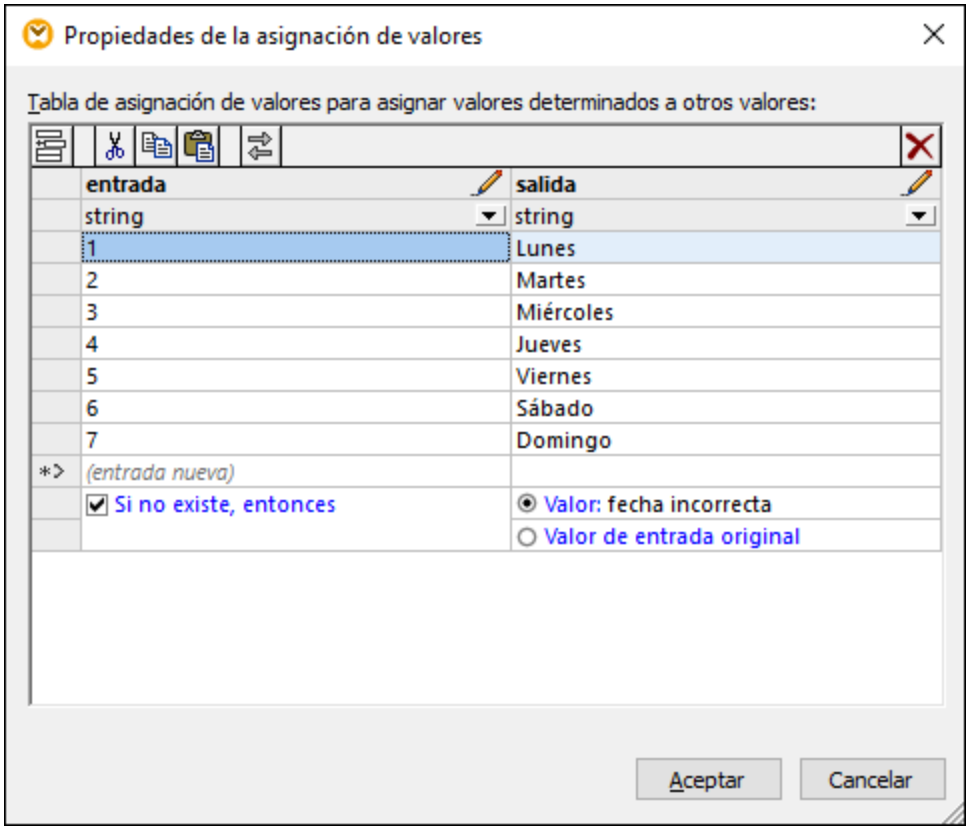
Si la tabla de consulta es enorme ralentizará la ejecución de la asignación. En este caso es mejor utilizar un componente de BD con SQL-Where 352.En estos casos son útiles las bases de datos SQLite, dada su portabilidad. Del lado servidor puede mejorar el rendimiento de las tablas de consulta ejecutando sus asignaciones con MapForce Server o MapForce Server Advanced Edition.

Crear asignaciones de valores

Para añadir un componente de asignación de valores a la asignación, elija una de estas opciones:

- Haga clic en el botón de la barra de herramientas **Insertar asignación de valores** o
- En el menú **Insertar** haga clic en **Asignación de valores**.
- Haga clic con el botón derecho en una conexión y seleccione **Insertar asignación de valores** en el menú contextual.

Así añade un componente de asignación de valores nuevo a la asignación. Ahora puede empezar añadir pares de elementos a la tabla de consulta. Para ello haga doble clic en la barra del título del componente o haga clic con el botón derecho y seleccione **Propiedades** en el menú contextual.



En el momento de ejecutar la asignación, MapForce comprueba cada valor que llega a la **entrada** del componente `value-map`. Si hay un valor que coincida en *la columna izquierda* de la tabla, este reemplaza al valor de entrada original con el valor de *la columna derecha*. De lo contrario puede configurarla para que devuelva:

- Un valor de reemplazo. En el ejemplo anterior el valor de reemplazo es el texto "fecha incorrecta". También puede definir el valor de reemplazo como un valor vacío. Para ello basta con que no introduzca ningún texto en ese campo.
- El valor de entrada original. Esto significa que si no hay ninguna coincidencia en la tabla de consulta el valor que se pasa a la asignación es el valor de entrada original sin modificar.

Si no configura una condición "Si no existe, entonces", entonces el componente `value-map` devuelve un nodo vacío si no encuentra una coincidencia. En este caso no se pasa ningún valor al componente de destino y el resultado contiene campos vacíos. Para evitar esto debe configurar la condición "Si no existe, entonces" o usar la función **`substitute-missing`** 707.

No es lo mismo definir un valor de reemplazo como vacío y no definir la condición "Si no existe, entonces". En el primer caso, el campo se genera en el resultado, pero contiene un valor vacío. En el segundo caso, el campo (o elemento XML) que contiene el valor ni siquiera se crea. Para más información consulte el apartado Ejemplo: reemplazar puestos de trabajo 539.

 © 2016-2022 Altova GmbH

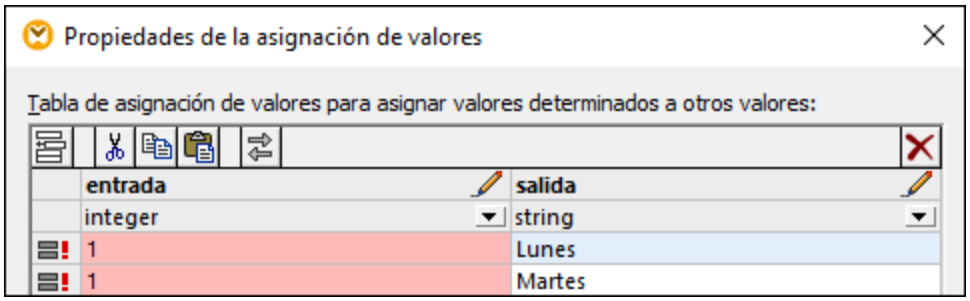
Rellenar una asignación de valores

En una tabla de consulta puede definir tantos pares de valores como necesite. Puede introducirlos de forma manual o copiar los datos de la tabla desde archivos de texto, CSV o Excel. En la mayoría de los casos también se pueden copiar y pegar tablas desde una página HTML usando un navegador web.
También puede pegar datos desde la cuadrícula de la BD en el Consulta de la BD 380. Si copia datos desde archivos de texto los campos deben estar separados por el carácter tabulador. Por lo general MapForce también reconoce texto separado por coma o por punto y coma.

Al crear tablas de consulta, tenga en cuenta lo siguiente:

1. Todos los elementos de la columna izquierda deben ser únicos. De lo contrario no es posible determinar qué elemento en concreto quiere asignar.
2. Los elementos que pertenecen a la misma columna deben ser del mismo tipo de datos. Puede elegir el tipo de datos de la lista desplegable que hay al principio de cada columna en la tabla de consulta. Si necesita convertir tipos booleanos, introduzca el texto "true" o "false". Para ver un ejemplo consulte el apartado Ejemplo: reemplazar días de la semana 537.

Si MapForce encuentra datos no válidos en la tabla de consulta, entonces muestra un mensaje de error y resalta las filas no válidas en color rosa:



Usar una fuente de datos externa para importar datos en el componente de asignación de valores:

1. Seleccione las celdas relevantes en el programa de origen (por ejemplo, Excel). Puede seleccionar una sola columna de datos o dos adyacentes.
2. Copie los datos en el portapapeles con el comando **Copiar** del programa externo.
3. En el componente de asignación de valores haga clic en la fila anterior a la fila en la que quiere pegar los datos.
4. Haga clic en el botón **Pegar tabla desde el portapapeles** del componente de asignación de valores. También pude pulsar **Ctrl+V** o **Mayús+Insertar**.

Nota: el botón **Pegar tabla desde el portapapeles** solo se habilita si copia primero datos de alguna fuente de datos (es decir, si hay datos en el portapapeles).

Si el portapapeles contiene varias columnas, en la tabla de consulta solamente se insertan las dos primeras columnas y el resto de ellas se ignora. Si pega datos desde una única columna encima de otros valores aparece un menú contextual que pregunta si los datos del portapapeles se deben insertar como filas nuevas o si se deben sobrescribir las que ya existen. Por lo tanto, si lo que quiere es sobrescribir los datos de la tabla

de consulta en vez de insertar nuevas filas, debe asegurarse de que el portapapeles debe contener una sola columna y no varias.

Para insertar filas manualmente antes de una fila que ya existe, primero debe hacer clic en la fila en cuestión y después pulsar el botón **Insertar** .

Para mover una fila que ya existe a una posición distinta, arrástrela hasta la nueva posición (hacia arriba o hacia abajo) mientras mantiene pulsado el botón izquierdo del ratón.

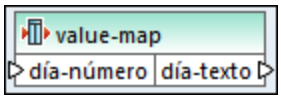
Si quiere copiar o cortar filas para después pegarlas en una posición distinta, primero seleccione la fila en cuestión y después haga clic en el botón **Copiar** (o **Copiar**). También puede copiar o cortar filas que no sean consecutivas. Para ello, mantenga pulsada la tecla **Ctrl** mientras va haciendo clic en las filas que quiere seleccionar. Tenga en cuenta que el texto cortado o copiado siempre contiene valores de ambas columnas, no puede cortar ni copiar valores de una columna solamente.

Para eliminar una fila haga clic en ella y después pulse el botón **Eliminar** .

Para intercambiar una columna por otra haga clic en el botón Intercambiar columnas .

Cambiar el nombre de parámetros de asignación de valores

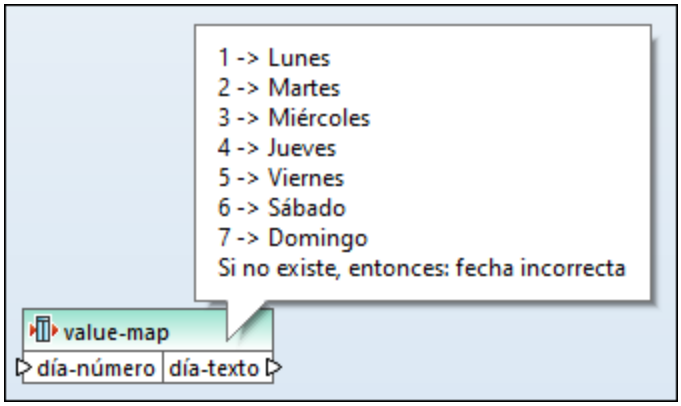
Por defecto, el parámetro de entrada de un componente de asignación de valores se llama "input" (*entrada*) y el parámetro de salida se llama "result" (*resultado*). Para que la asignación resulte más clara puede cambiar el nombre a cualquiera de estos parámetros haciendo clic en el botón **Editar** que hay junto al nombre en cuestión. Este es un ejemplo de un componente de asignación de valores en el que se han cambiado los nombres de los parámetros:



Vista previa de una asignación de valores

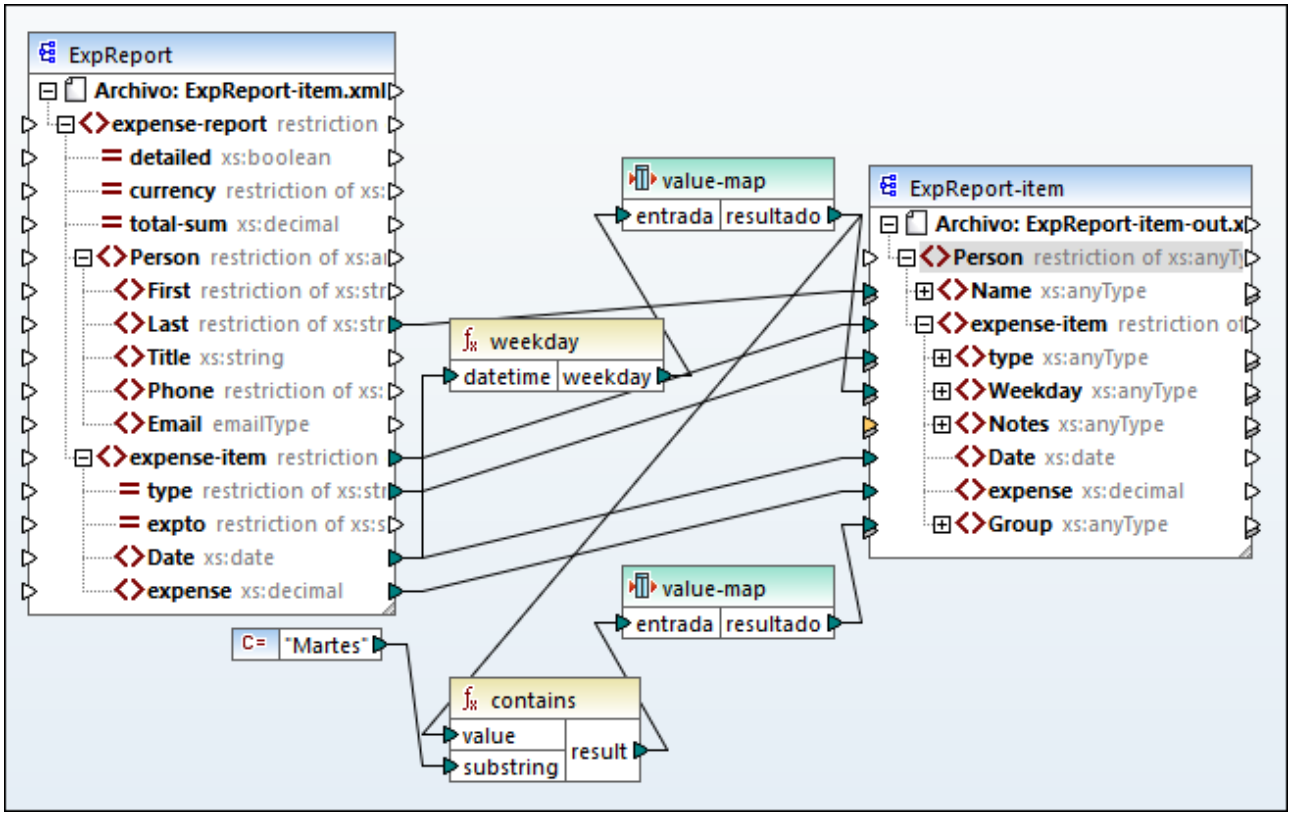
Una vez haya terminado de crear la asignación de valores puede acceder a una vista previa de su implementación directamente desde la asignación. Para ello solo debe pasar el cursor por encima de la barra del título del componente:

 © 2016-2022 Altova GmbH



5.8.1 Pasar datos sin modificarlos

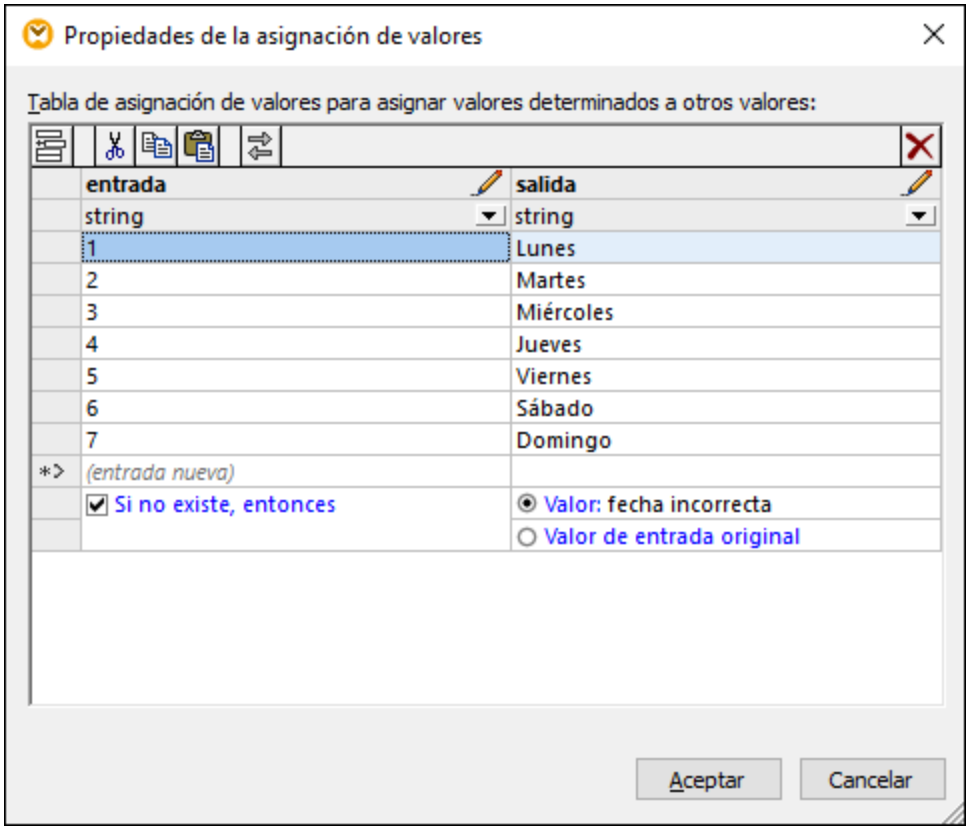
Este ejemplo consiste en una asignación de valores que reemplaza valores enteros con nombres de los días de la semana (1 = lunes, 2 = martes, etc.). El ejemplo viene acompañado de una asignación que puede encontrar en: **<Documentos>\Altova\MapForce2023\MapForceExamples\Tutorial\Expense-valmap.mfd**.



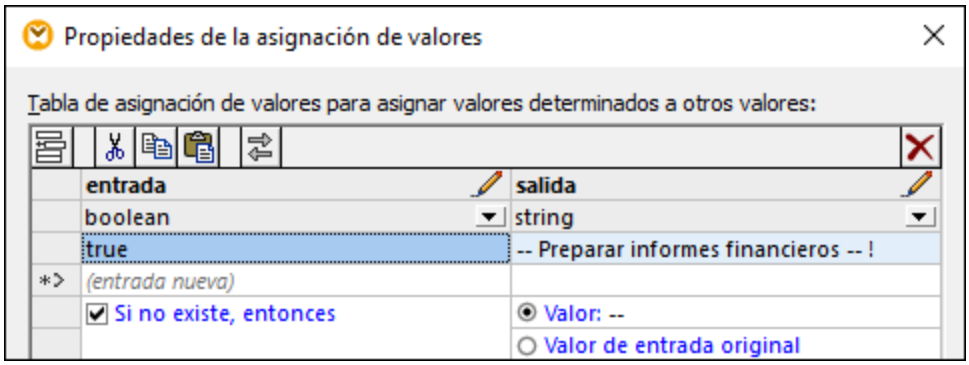
Expense-valmap.mfd

Esta asignación extrae el día de la semana del elemento **Date** en el archivo de origen, convierte el valor numérico en texto y lo escribe en el elemento **Weekday** del componente de destino. En concreto, ocurre lo siguiente:

- La función `weekday` extrae el número del día de la semana del elemento Date en el archivo de origen. El resultado de esta función son números enteros del 1 al 7.
- El primer componente de asignación de valores transforma los números enteros en días de la semana (1 = domingo, 2 = lunes). Si el componente se encuentra con un número entero no válido fuera del rango del 1 al 7, entonces devuelve el texto "fecha incorrecta".



- Si el día de la semana contiene "martes", entonces se escribe el texto "Preparar informes financieros" en el elemento **Notas** del componente de destino. Esto se consigue gracias a la función `contains`, que pasa un valor booleano **true** o **false** al segundo componente de asignación de valores. Ese segundo componente está configurado como se muestra en la imagen:



 © 2016-2022 Altova GmbH

La asignación de valores de la imagen anterior se debe entender así:

- Siempre que se encuentre un valor booleano **true** se convierte en el texto "-- Preparar informes financieros -- ! ". En el resto de casos el texto es solamente "--".

Tenga en cuenta que el tipo de datos de la primera columna está definido como "boolean", lo que garantiza que el valor booleano **true** se reconozca como tal.

5.8.2 Propiedades de las asignaciones de valores

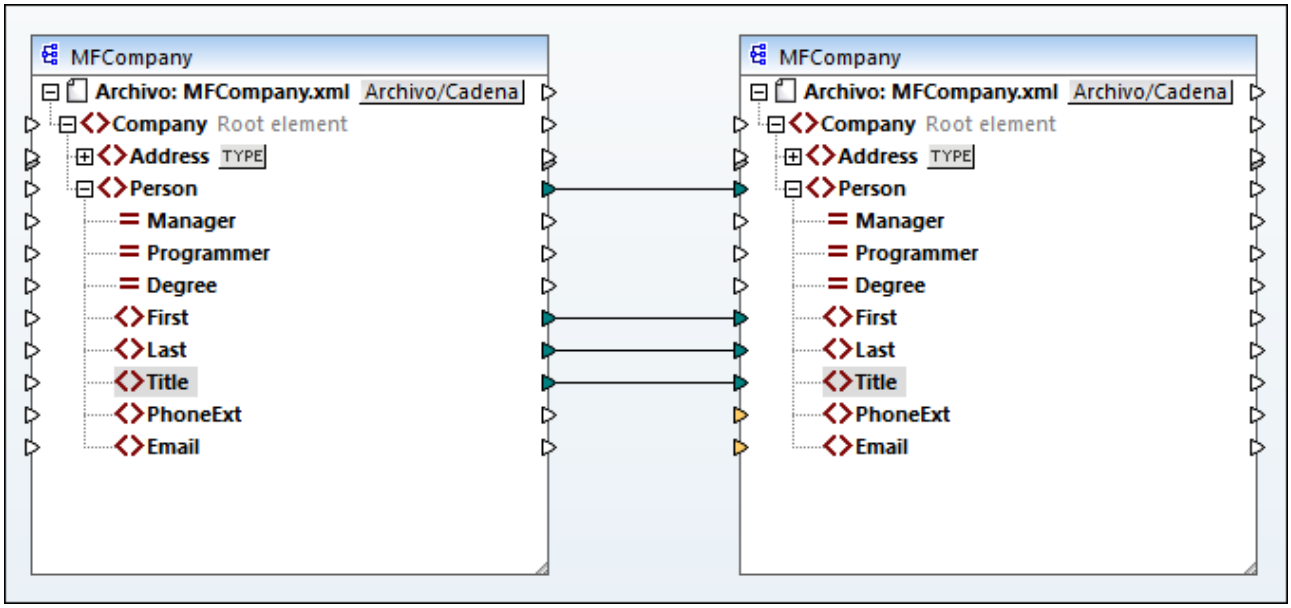
En este ejemplo aprenderá cómo reemplazar valores de elementos específicos en un archivo XML con ayuda de componentes de asignación de valores (es decir, usando una tabla de consulta predefinida).

Puede encontrar el archivo XML necesario para este ejemplo en **<Documentos>\Altova\MapForce2023\MapForceExamples\Tutorial\MFCompany.xml**. Además de otros datos, este archivo almacena información sobre los empleados de la empresa y sus puestos de trabajo, por ejemplo:

```
<Person>
   <First>Michelle</First>
   <Last>Butler</Last>
   <Title>Software Engineer</Title>
</Person>
<Person>
   <First>Lui</First>
   <Last>King</Last>
   <Title>Support Engineer</Title>
</Person>
<Person>
   <First>Steve</First>
   <Last>Meier</Last>
   <Title>Office Manager</Title>
</Person>
```

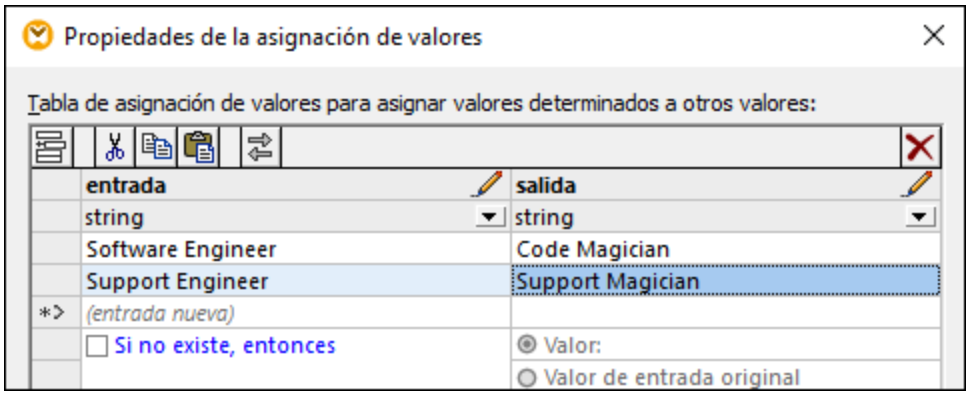
Imagine que necesita reemplazar algunos de esos puestos de trabajo en el archivo XML que hemos mencionado más arriba. En concreto, debe reemplazar "Software Engineer" (Ingeniero de software) por "Code Magician" (Mago del código). También el título "Support Engineer" (Ingeniero de soporte) se debe sustituir por "Support Magician" (Mago del soporte). El resto de puestos de trabajo permanece sin cambios.

Para conseguir este objetivo debe añadir el archivo XML al área de asignación: haga clic en el botón de la barra de herramientas **Insertar archivo o esquema XML** o ejecutando el comando de menú **Insertar archivo o esquema XML**. A continuación, copie y pegue el componente XML en la asignación y cree las conexiones como se muestra en la imagen siguiente. Puede que primero necesite desactivar la opción Conectar automáticamente los secundarios equivalentes para evitar que se creen automáticamente conexiones innecesarias.



La asignación que ha creado hasta ahora solamente copia los elementos **Person** en el archivo XML de destino sin hacer ningún cambio el los elementos **First**, **Last** o **Title**.

Para reemplazar los puestos de trabajo en cuestión debe añadir un componente de asignación de valores. Haga clic con el botón derecho en la conexión entre los dos elementos **Title** y seleccione **Insertar asignación de valores** en el menú contextual. Defina las propiedades de la asignación de valores como en la imagen:



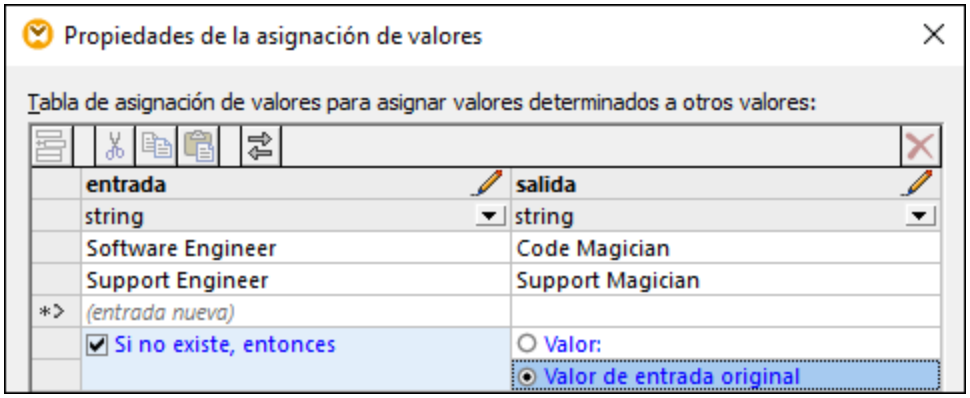
Según la configuración anterior, cada ocurrencia de "Software Engineer" se reemplaza con "Code Magician" y cada ocurrencia de "Support Engineer" se reemplaza con "Support Magician". Tenga en cuenta que no se ha definido la condición **Si no existe, entonces**, por lo que la asignación de valores devuelve un *nodo vacío* si el puesto de trabajo es distinto a "Software Engineer" o "Support Engineer". En consecuencia, si hace clic en la pestaña *Resultados* y accede a la vista previa de la asignación, en algunos de los elementos **Person** faltará el elemento **Title**, por ejemplo:

```
<Person>
   <First>Vernon</First>
```

 © 2016-2022 Altova GmbH

```
   <Last>Callaby</Last>
</Person>
<Person>
   <First>Frank</First>
   <Last>Further</Last>
</Person>
<Person>
   <First>Michelle</First>
   <Last>Butler</Last>
   <Title>Code Magician</Title>
</Person>
```

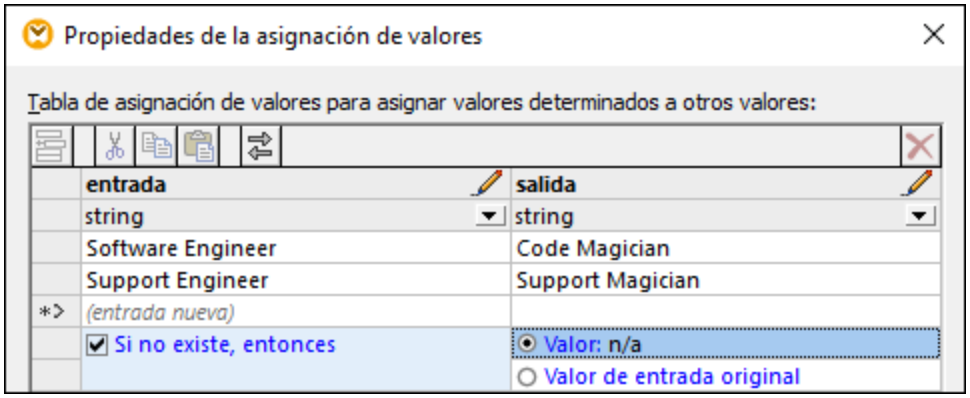
Como acabamos de explicar, si hay nodos vacíos en el resultado generado faltan algunas entradas; por tanto, en el fragmento XML anterior solamente se reemplazó el puesto de Michelle Butler, ya que este estaba en la tabla de consulta. La configuración que ha creado hasta ahora no se ajusta al objetivo inicial. Los ajustes correctos serían:



Con la configuración de la imagen anterior ocurre lo siguiente al ejecutar la asignación:

- Cada ocurrencia de "Software Engineer" se reemplaza con "Code Magician"
- Cada ocurrencia de "Support Engineer" se reemplaza con "Support Magician"
- Si el puesto original no se encuentra en la tabla de consulta, la asignación de valores lo devolverá sin cambios.

Solo a efectos ilustrativos podemos cambiar también los puestos de trabajo que no sean "Software Engineer" o "Support Engineer" por un valor personalizado, por ejemplo "n/a". Para ello debe definir las propiedades de la asignación de valores como en la imagen siguiente:



Si ahora accede a una vista previa de la asignación, en el resultado aparecen todos los puestos de trabajo, pero aquellos que no coincidan con los dos que se han definido aparecen con el valor "n/a", por ejemplo:

```
<Person>
   <First>Vernon</First>
   <Last>Callaby</Last>
   <Title>N/A</Title>
</Person>
<Person>
   <First>Frank</First>
   <Last>Further</Last>
   <Title>N/A</Title>
</Person>
<Person>
   <First>Michelle</First>
   <Last>Butler</Last>
   <Title>Code Magician</Title>
</Person>
```

Aquí termina este ejemplo de asignación de valores. Puede usar este proceso lógico para conseguir el resultado deseado también en otras asignaciones.

© 2016-2022 Altova GmbH

5.9 Filtros y condiciones

Cuando necesite filtrar datos o conseguir un valor de forma condicional, utilice uno de estos componentes:

- Filtro: nodos/filas ( )
- WHERE/ORDER de SQL ( )
- Condición If-Else ( )

Estos componentes se pueden añadir en la asignación desde el menú Insertar o desde la barra de herramientas Insertar componente. A continuación vamos a explicar las diferencias entre estos componentes, su comportamiento y sus requisitos.

Filtrar nodos o filas

Cuando necesite filtrar datos, incluidos nodos XML o filas CSV, utilice un componente **Filtro: nodos/filas**. Este componente permite recuperar un subconjunto de nodos de un conjunto de datos de mayor tamaño, usando una condición true o false. Esta es su estructura en el área de asignación:

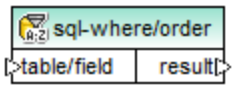
En la estructura de filtro anterior, la condición que está conectada a bool determina si la entrada **node/row** que está conectado se pasa a la salida **on-true** o a la salida **on-false**. Es decir, si la condición se cumple (es true), entonces **node/row** se redirige a la salida **on-true**. Por el contrario, si la condición no se cumple (es false), entonces **node/row** se redirige a la salida **on-false**.

Cuando necesite que la asignación solamente consuma elementos que *cumplan* la condición de filtrado, puede dejar sin conectar la salida **on-false**. Si necesita procesar los elementos que no cumplan la condición de filtrado, entonces conecte la salida **on-false** al nodo de destino donde se deben redirigir dichos elementos. Si quiere añadir una excepción para cuando la condición de filtrado no se cumpla, será obligatorio conectar la salida **on-false** (*véase* *Añadir excepciones* 529).

Para ver un ejemplo con instrucciones paso a paso consulte el apartado Ejemplo: filtrar nodos 545.

Filtrar datos de base de datos

Los componentes **Filtro: nodos/filas** pueden filtrar datos de cualquier otra estructura compatible con MapForce, incluidas las bases de datos. Sin embargo, si desea filtrar datos de una base de datos, lo más recomendado es usar un componente **WHERE/ORDER de SQL**. Este componente está optimizado para trabajar con bases de datos y ofrece un mayor rendimiento que el componente **Filtro: nodos/filas**.



Para más información consulte el apartado Componente WHERE/ORDER de SQL 352.

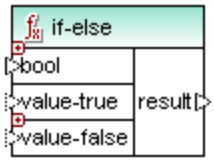
Devolver un valor de forma condicional

Si necesita obtener un solo valor (en lugar de un nodo o de una fila) de forma condicional, lo mejor es usar la **condición If-Else**. Recuerde que estas condiciones no son adecuadas para el filtrado de nodos o filas. A diferencia de los componentes **Filtro: nodos/filas**, las **condiciones If-Else** devuelven un valor de tipo simple (como una cadena o un entero). Por tanto, estas condiciones solamente deben utilizarse cuando sea necesario procesar un valor simple de forma condicional. Por ejemplo, imagine que tiene una lista de las temperaturas medias de cada mes en este formato:

```
<Temperatures>
   <data temp="19.2" month="2010-06" />
   <data temp="22.3" month="2010-07" />
   <data temp="19.5" month="2010-08" />
   <data temp="14.2" month="2010-09" />
   <data temp="7.8" month="2010-10" />
   <data temp="6.9" month="2010-11" />
   <data temp="-1.0" month="2010-12" />
</Temperatures>
```

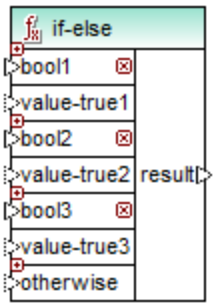
Una **condición If-Else** permite devolver por cada elemento de la lista el valor "alto" si la temperatura supera los 20 grados centígrados y el valor "bajo" si la temperatura es inferior a los 5 grados centígrados.

En la asignación la estructura de la **condición If-Else** sería:



Si la condición que está conectada a **bool** es `true`, entonces el valor que está conectado a **value-true** se devuelve como **result**. Si la condición es `false`, el valor que está conectado a **value-false** se devuelve como **result**. El tipo de datos de **result** se conoce previamente y depende del tipo de datos del valor que está conectado a **value-true** o **value-false**. Lo importante es que siempre sea de tipo simple (cadena, entero, etc.). Las **condiciones If-Else** no admiten el uso de valores de entrada de tipo complejo (como nodos o filas).

Además, las **condiciones If-Else** son extensibles. Esto significa que puede añadir varias condiciones al componente con solo hacer clic en el botón **Agregar** ⊞. Para eliminar una condición añadida previamente, haga clic en el botón **Eliminar** ⊠. Esta característica permite comprobar si se cumplen varias condiciones y devolver un valor distinto para cada condición.



 © 2016-2022 Altova GmbH

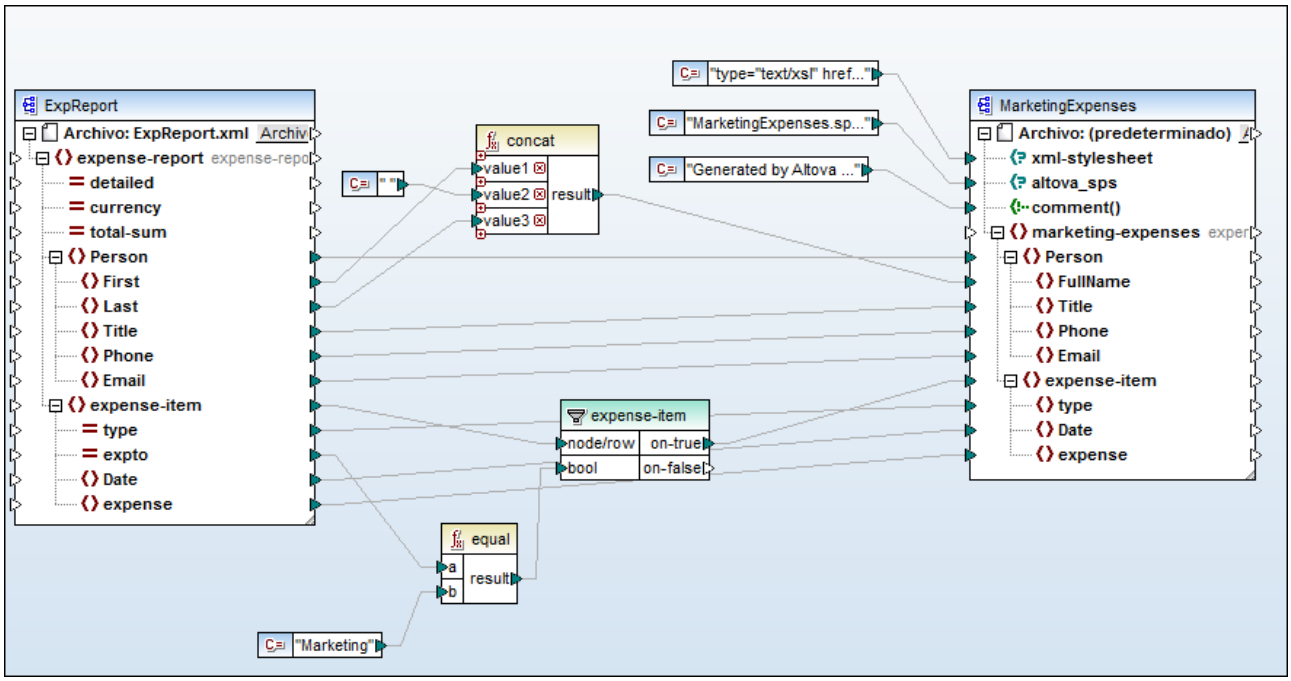
Las **condiciones If-Else** extendidas, se evalúan de arriba a abajo (se comprueba en primer lugar las condiciones situadas más arriba). Si desea devolver un valor aunque no se cumpla ninguna condición (es decir, aunque ninguna condición sea true), conecte el valor a **`otherwise`**.

Para ver un ejemplo con instrucciones paso a paso consulte el apartado Ejemplo: devolver un valor de forma condicional 547.

5.9.1 Ejemplo: filtrar nodos

Este ejemplo explica cómo filtrar nodos basándose en una condición true/false. Para conseguirlo se utiliza un componente **Filtro: nodos/filas** ( ). La técnica que se utiliza en este ejemplo funciona igual para componentes XML y otros tipos como CSV o texto. A la hora de trabajar con bases de datos, sin embargo, se recomienda el uso de componentes WHERE/ORDER de SQL, que permiten un mayor rendimiento (véase Componente WHERE / ORDER de SQL 352).

La asignación de este ejemplo es **<Documentos>\Altova\MapForce2023\MapForceExamples\MarketingExpenses.mfd**.



Como puede verse en la imagen, la asignación lee datos de un XML de origen que contiene un informe de gastos ("ExpReport") y escribe datos en un XML de destino ("MarketingExpenses"). Entre el componente de origen y de destino existen varios componentes más. El más relevante en este ejemplo es el componente de filtrado **expense-item** ( ).

El objetivo de la asignación es filtrar los gastos que pertenecen al departamento de Marketing. Para conseguirlo se añadió un componente de filtrado a la asignación. (Para añadir un filtro haga clic en el menú Insertar y elija el comando **Filtro: nodos/filas**.)

Para identificar qué gastos pertenecen al departamento de Marketing, la asignación busca el valor del atributo "expto" del XML de origen. Este atributo tiene el valor "Marketing" si el gasto pertenece a dicho departamento. Por ejemplo, en el siguiente fragmento de código, puede verse que el primer gasto y el tercero pertenecen a Marketing, el segundo a Development (desarrollo) y el cuarto a Sales (ventas):

```
...
   <expense-item type="Meal" expto="Marketing">
      <Date>2003-01-01</Date>
      <expense>122.11</expense>
   </expense-item>
   <expense-item type="Lodging" expto="Development">
      <Date>2003-01-02</Date>
      <expense>122.12</expense>
   </expense-item>
   <expense-item type="Lodging" expto="Marketing">
      <Date>2003-01-02</Date>
      <expense>299.45</expense>
   </expense-item>
   <expense-item type="Entertainment" expto="Sales">
      <Date>2003-01-02</Date>
      <expense>13.22</expense>
   </expense-item>
...
```

XML de entrada antes de ejecutarse la asignación

En el área de asignación, la entrada **node/row** del filtro está conectado con el nodo **expense-item** del componente de origen. Esto permite al filtro obtener la lista de nodos que debe procesar.

Para agregar la condición en la que se debe basar el filtrado, se añadió la función **equal** de la biblioteca **core** de MapForce (*véase* *Funciones básicas* 550). La función **equal** compara el valor del atributo "type" con una constante cuyo valor es "Marketing". (Para agregar una constante haga clic en el menú **Insertar** y elija el comando **Constante**.)

Como nuestro objetivo es filtrar los elementos que cumplen esta condición, conectamos solamente la salida **on-true** del filtro con el componente de destino.

Cuando consultamos la vista previa del resultado de la asignación (en el panel *Resultados*), MapForce evalúa la condición conectada a la entrada **bool** del filtro en cada nodo expense-item. Si la condición se cumple (es true), el nodo expense-item se pasa al destino. De lo contrario, se pasa por alto. Como resultado se obtienen los gastos que cumplen los criterios:

```
...
   <expense-item>
      <type>Meal</type>
      <Date>2003-01-01</Date>
      <expense>122.11</expense>
   </expense-item>
   <expense-item>
      <type>Lodging</type>
      <Date>2003-01-02</Date>
```

 © 2016-2022 Altova GmbH

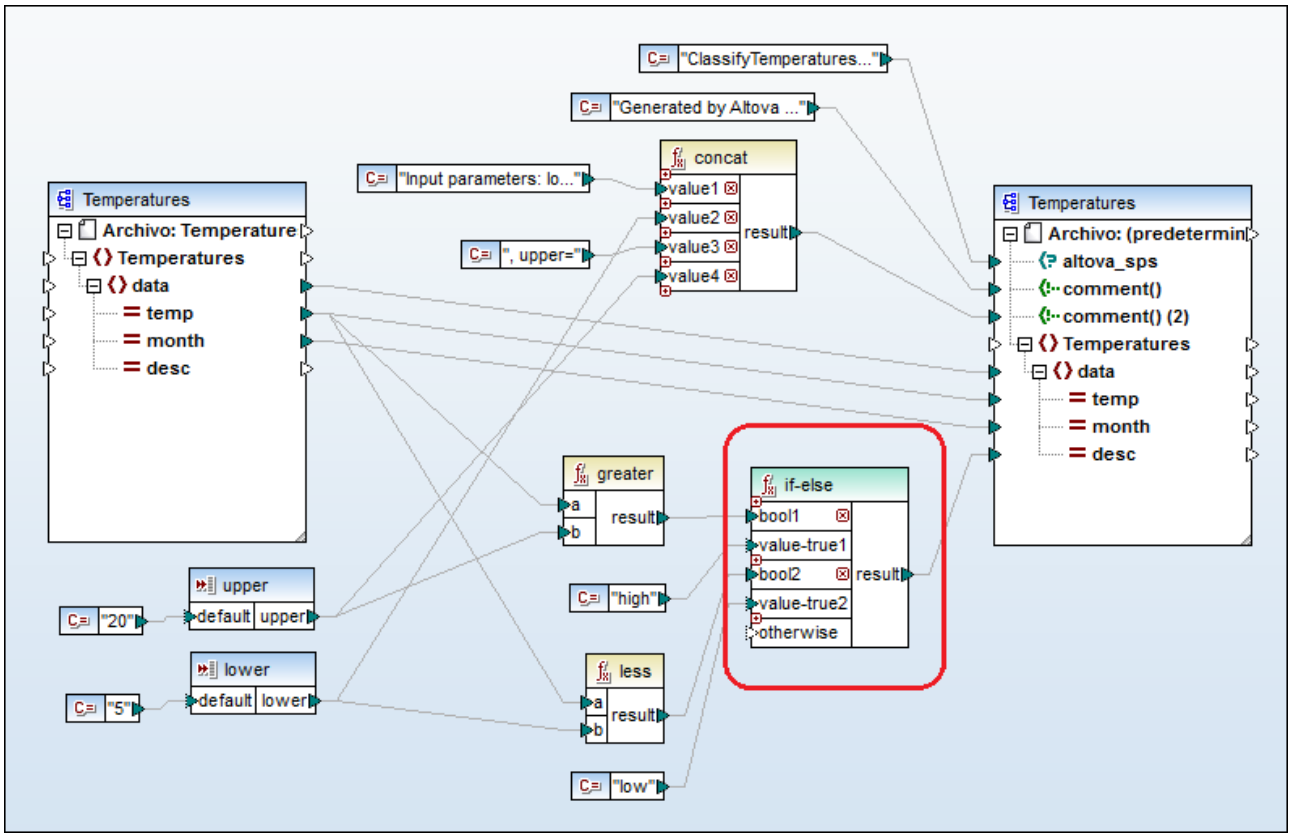
```
      <expense>299.45</expense>
   </expense-item>
...
```

XML de salida después de ejecutarse la asignación

5.9.2 Ejemplo: devolver un valor de forma condicional

Este ejemplo explica cómo obtener un valor simple de un componente con ayuda de una condición true/false. Para conseguirlo se utiliza una **condición If-Else** ( ). Las **condiciones If-Else** no se deben confundir con los componentes de filtrado. Las primeras son adecuadas para procesar valores simples de forma condicional (cadenas, enteros, etc.). Los componentes de filtrado, sin embargo, deben utilizarse para filtrar valores complejos como nodos (*véase* *Ejemplo: filtrar nodos* 545).

La asignación de este ejemplo es **<Documentos>\Altova\MapForce2023\MapForceExamples\ClassifyTemperatures.mfd**.



Esta asignación lee datos de un XML de origen que contiene datos sobre temperaturas ("Temperatures") y escribe datos en un XML de destino que cumple el mismo esquema. Entre el destino y el origen existen varios componentes más. De ellos el más relevante es la condición if-else (marcada en rojo).

El objetivo de la asignación es agregar una breve descripción a cada registro de temperatura en el documento de destino. Concretamente, si la temperatura supera los 20 grados centígrados, la descripción será

"high" (alta). Si, por el contrario, la temperatura no supera los 5 grados centígrados, la descripción será "low" (baja). En el resto de los casos, no se incluirá ninguna descripción.

Para conseguir nuestro objetivo es necesario un procesamiento condicional y, por tanto, se añadió una condición if-else a la asignación. (Para añadir una condición if-else haga clic en el menú **Insertar** y después elija el comando **Condición If-Else**). En esta asignación la condición if-else se amplió (con ayuda del botón ⊡) para que incluyera dos condiciones: **bool1** y **bool2**.

Las condiciones propiamente dichas vienen dadas por las funciones **greater** y **less**, que se añadieron desde la biblioteca **core** de MapForce (*véase* *Trabajar con funciones* 550). Estas funciones evalúan los valores que aportan dos componentes de entrada llamados "upper" y "lower". (Para añadir un componente de entrada haga clic en el menú Insertar y elija el comando Insertar componente de entrada. Consulte el apartado Supplying Parameters to the Mapping 477 para obtener más información.)

Las funciones **greater** y **less** devuelven el valor true o false. El resultado de la función determina qué texto se escribe en la instancia de destino. Es decir, si el valor del atributo "temp" del XML de origen es superior a 20, entonces se pasa el valor de constante "high" a la condición if-else. Si el valor del atributo "temp" del XML de origen es inferior a 5, entonces se pasa el valor de constante "low" a la condición **if-else**. La entrada **otherwise** de la condición se deja sin conectar. Por tanto, si no se cumple ninguna de las condiciones anteriores, no se pasará nada al conector de salida **result**.

Por último, el conector de salida **result** suministra este valor (uno por cada registro de temperatura) al atributo "desc" del componente de destino.

Para consultar una vista previa del resultado de la asignación haga clic en el panel *Resultados*. Observe que el XML de salida resultante incluye ahora el atributo "desc" cuando la temperatura es superior a 20 o inferior a 5.

```
...
   <data temp="-3.6" month="2006-01" desc="low"/>
   <data temp="-0.7" month="2006-02" desc="low"/>
   <data temp="7.5" month="2006-03"/>
   <data temp="12.4" month="2006-04"/>
   <data temp="16.2" month="2006-05"/>
   <data temp="19" month="2006-06"/>
   <data temp="22.7" month="2006-07" desc="high"/>
   <data temp="23.2" month="2006-08" desc="high"/>
...
```

XML de salida después de ejecutarse la asignación

 © 2016-2022 Altova GmbH

6 Funciones

Las funciones suponen un potente mecanismo para transformar datos. En esta sección encontrará instrucciones para trabajar con funciones (tanto sin son funciones integradas de MapForce, como si son funciones definidas por el usuario o importadas de fuentes externas).

- **Funciones integradas de MapForce:** estas funciones vienen predefinidas en MapForce y se pueden usar en las asignaciones para tareas de procesamiento que conlleven cadenas de texto, números, fechas y otros tipos de datos. También puede usarlas para tareas de agrupamiento, suma, numeración automática y muchas tareas más. Para ver todas las funciones disponibles consulte la Referencia de la biblioteca de funciones 631.
- **Funciones de nodo y predeterminadas:** estas funciones son más especializadas que permiten crear y aplicar una lógica de procesamiento personalizada a uno o varios nodos descendientes en un componente de asignación. Estas funciones permiten procesar los datos *antes* de que la función alcance un nodo de una estructura de asignación o *inmediatamente* después de abandonar un nodo. Para más detalles consulte Valores predeterminados y funciones 558.
- **Funciones definidas por el usuario:** estas son funciones de MapForce que puede crear usted mismo usando como base los tipos de componente nativos y las funciones integradas que ya están disponibles en MapForce (véase Funciones definidas por el usuario 575).
- **Funciones personales:** estas son funciones que puede importar de fuentes externas, como bibliotecas XSLT, módulos de bibliotecas XQuery, archivos Java .class, archivos .NET .dll y adaptarlas a MapForce. Tenga en cuenta que para poder reutilizarlas en MapForce, los datos que devuelvan las funciones personalizadas deben ser de tipo simple (como una cadena o un número entero) y los parámetros que tomen también deben ser de tipo simple. Para más información consulte Importar funciones XSLT personales 593, Importar funciones XQuery 1.0 personales 600 y Importar bibliotecas Java y .NET personales 604.

Nota: puede importar bibliotecas externas de funciones directamente (no necesita configurar nada) o configurando un MFF (archivo de función MapForce) que MapForce reconozca. Si usa esta última opción, también puede importar bibliotecas C++ además de clases Java y ensamblajes .NET. Tenga en cuenta que las bibliotecas importadas con archivos .mff cumplen los requisitos mencionados en Referencias manuales a bibliotecas Java, C# y C++ personales 612.

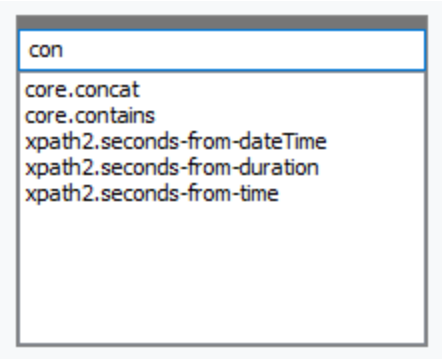
6.1 Fundamentos de las funciones

En las subsecciones siguientes encontrará un resumen de las acciones básicas relacionadas con funciones. Las funciones disponibles en la ventana **Bibliotecas** dependen del lenguaje de transformación seleccionado en ese momento. Para saber más consulte Seleccionar el lenguaje de transformación 24.

Agregar una función

MapForce cuenta con un amplio número de funciones integradas que puede agregar a sus asignaciones. Para más información sobre las funciones integradas disponibles consulte Referencia de la biblioteca de funciones 631. Hay varios métodos para agregar una función a una asignación:

- Busque la función que necesite en la ventana **Bibliotecas** y arrástrela hasta el área de asignación. Para filtrar las funciones por nombre empiece a teclear el nombre de la función en el cuadro de texto que hay en la parte inferior de la ventana.
- Haga doble clic en un área vacía de la asignación y empiece a teclear el nombre de la función (*imagen siguiente*). Para ver una ayuda rápida con información sobre la función vaya a la lista de funciones y haga clic en la función relevante. Para agregar una función a una asignación haga doble clic en la función relevante en el cuadro combinado.



Agregar funciones definidas por el usuario
También puede agregar funciones definidas por el usuario a sus asignaciones, bien como hemos explicado más arriba si (i) esa función definida por el usuario ya se ha creado en la misma asignación o (ii) si ha importado una asignación que contiene una función definida por el usuario como biblioteca local o global.

Agregar una constante

Las constantes permiten proporcionar a una asignación texto y cifras personales. Para agregar una constante a una asignación tiene varias opciones:

- Haga clic con el botón derecho en un área vacía de la asignación y seleccione **Insertar constante** en el menú contextual. Introduzca el valor y seleccione uno de estos tipos de datos: *cadena*, *número*, *demás opciones*.
- Añada una constante con el comando de menú **Insertar | Constante**. Introduzca el valor y seleccione uno de estos tipos de datos: *cadena*, *número*, *demás opciones*.
- Haga clic en el comando de la barra de herramientas **Constante**. Introduzca el valor y seleccione uno de estos tipos de datos: *cadena*, *número*, *demás opciones*.
- Haga doble clic en cualquier parte vacía del área de la asignación. Teclee comillas dobles seguidas del valor de la constante. No es necesario que cierre las comillas. Para añadir una constante numérica, simplemente teclee el número.

 © 2016-2022 Altova GmbH

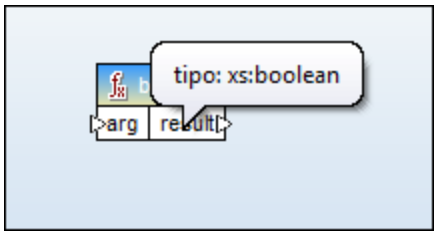
Buscar una función

Para buscar una función en la ventana **Bibliotecas** empiece a teclear el nombre de la función en el campo de texto que hay en la parte inferior de la ventana. Por defecto, MapForce busca el nombre de la función y el texto descriptivo. Si desea excluir la descripción de las funciones de la búsqueda, haga clic en el icono en forma de flecha y deshabilite la opción *Incluir descripción en la búsqueda*. Para cancelar la búsqueda pulse la tecla **Esc** o haga clic en el icono ✖.

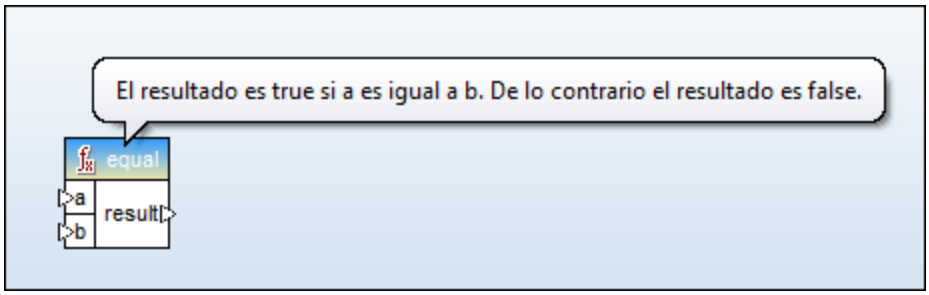
Para buscar todas las instancias de una función dentro de la asignación activa basta con hacer clic con el botón derecho en la función en la ventana **Bibliotecas** y elegir el comando **Buscar todas las llamadas** en el menú contextual. El resultado de la búsqueda aparece en la ventana **Mensajes**.

Ver el tipo y la descripción de una función

Para ver el tipo de datos del argumento de una función, pase el ratón por encima de la parte del argumento de una función (*imagen siguiente*). Asegúrese de que el botón (**Ver información rápida**) está habilitado.

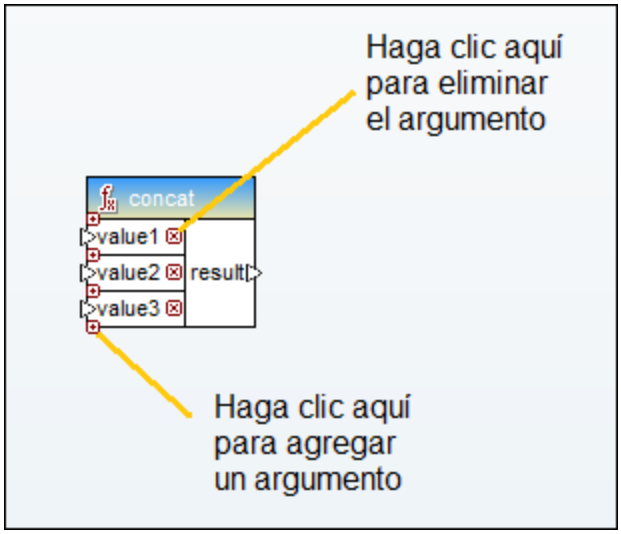


Para ver la descripción de una función mueva el ratón sobre el encabezado de la función (*imagen siguiente*). Asegúrese de que el botón (**Ver información rápida**) está habilitado.



Agregar o eliminar argumentos en una función

En algunas de las funciones integradas de MapForce es posible agregar tantos parámetros como necesite para la asignación. Un ejemplo de ello es la función **concat** 708. Para añadir o eliminar argumentos de una función (para las funciones que lo admiten) haga clic en **Agregar parámetro** (⊕) o **Eliminar parámetro** (⊗) junto al parámetro que quiere añadir o eliminar (*imagen siguiente*). Si mueve una conexión al símbolo ⊕ se añade otro parámetro que se conecta a este.



© 2016-2022 Altova GmbH

6.2 Gestionar bibliotecas de funciones

En MapForce puede importar y usar estos tipos de bibliotecas en una asignación:

- Cualquier archivo de diseño de asignación (*.mfd) que contenga funciones definidas por el usuario. En concreto, asignaciones que contengan funciones definidas por el usuario creadas con MapForce, usando las funciones y los componentes integrados de MapForce como bloques de construcción. Para más información consulte Crear funciones definidas por el usuario 576.
- Archivos XSLT que contengan funciones. En concreto, funciones XSLT escritas fuera de MapForce pero que se pueden importar en MapForce como se describe en Importar funciones XSLT personales 593.
- Archivos XQuery 1.0 personalizados que contengan funciones. En concreto, funciones XQuery escritas fuera de MapForce pero que se pueden importar en MapForce como se describe en Importar funciones XQuery 1.0 personales 600.
- Archivos Java .class y bibliotecas .NET .dll que se pueden importar en MapForce como se describe en Importar bibliotecas Java y .NET personales 604.

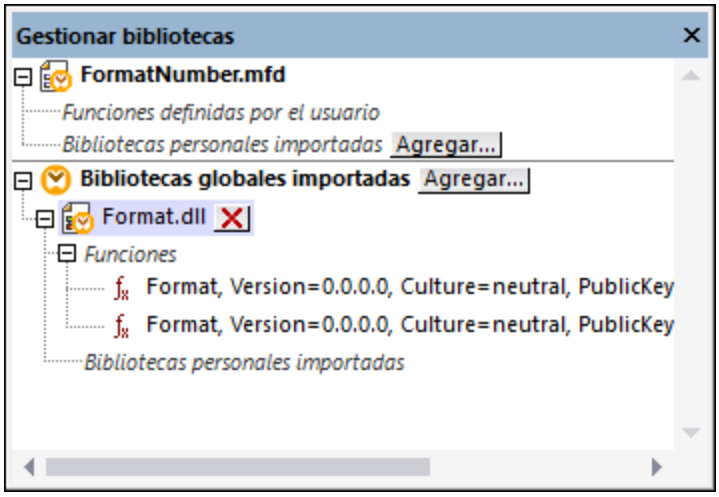
Nota: puede importar bibliotecas externas de funciones directamente (no necesita configurar nada) o configurando un MFF (archivo de función MapForce) que MapForce reconozca. Si usa esta última opción, también puede importar bibliotecas C++ además de clases Java y ensamblajes .NET. Tenga en cuenta que las bibliotecas importadas con archivos .mff cumplen los requisitos mencionados en Referencias manuales a bibliotecas Java, C# y C++ personales 612.

Ventana Gestionar bibliotecas

La ventana Gestionar bibliotecas permite ver y gestionar todas las bibliotecas que usa un archivo de asignación (tanto funciones definidas por el usuario como bibliotecas personales) al mismo tiempo.

Por defecto, la ventana Gestionar bibliotecas no está visible. Para que aparezca tiene dos opciones:

- En el menú **Vista** haga clic en **Gestionar bibliotecas**.
- En la parte inferior de la ventana Bibliotecas haga clic en **Agregar o quitar bibliotecas**.



Puede elegir si quiere ver solamente las funciones definidas por el usuario del archivo de asignación activo o si quiere ver las de todos los archivos de asignación abiertos. Para ver las funciones y bibliotecas importadas

para todas las asignaciones abiertas actualmente haga clic dentro de la ventana y seleccione **Mostrar documentos abiertos** en el menú contextual.

Si en vez de el nombre del archivo de asignación abierto prefiere ver su ruta de acceso haga clic dentro de la ventana y seleccione **Mostrar todas las rutas de acceso** en el menú contextual.

Los datos que aparecen en la ventana Gestionar bibliotecas se organizan como una estructura en árbol:

- Los archivos de asignación abiertos se muestran como entradas de nivel superior. Cada entrada tiene dos ramas: **Funciones definidas por el usuario** y **Bibliotecas personales importadas**.
    - La rama **Funciones definidas por el usuario** muestra las funciones definidas por el usuario que contenga el archivo de asignación.
    - La rama **Bibliotecas personales importadas** muestra las bibliotecas importadas *localmente* en el archivo de asignación actual. El término "bibliotecas" se refiere a otros archivos de asignación (archivos .mfd que contienen funciones definidas por el usuario), a bibliotecas personales externas escritas en XSLT 1.0, XSLT 2.0, XQuery 1.0*, Java*, C#* o a los archivos .mff mencionados con anterioridad. Tenga en cuenta que la estructura **Bibliotecas personales importadas** podría tener varios niveles de profundidad, ya que un archivo de asignación puede importar otros archivos de asignación como bibliotecas.
- La entrada **Bibliotecas globales importadas** incluye todas las bibliotecas que haya importado *globalmente*, a nivel de la aplicación. La estructura de los archivos .mfd también puede tener varios niveles de profundidad por las mismas razones que se explican en el punto anterior.

** Estos lenguajes solo son compatibles con las ediciones MapForce Professional y Enterprise.*

Nota: las bibliotecas XSLT, XQuery, C# y Java pueden tener dependencias propias. Estas dependencias no aparecen en la ventana Bibliotecas.

Comandos del menú contextual

Puede aplicar distintas acciones a los objetos de la ventana Gestionar bibliotecas. Para ello debe hacer clic con el botón derecho en el objeto en cuestión y seleccionar una de estas opciones del menú contextual:

Comando	Descripción	Se aplica a
Abrir	Abre la asignación.	Asignaciones
Agregar	Abre una caja de diálogo en la que puede navegar hasta una biblioteca personal de funciones.	Bibliotecas propias importadas
Buscar función en la ventana Bibliotecas	Pasa a estar activa la ventana Bibliotecas, donde se selecciona la función que se está buscando.	Funciones
Cortar, Copiar, Eliminar	Estos comandos estándar de Windows se pueden usar solamente con las funciones definidas por el usuario. No puede copiar o pegar funciones de archivos XSLT externos o de otros tipos de bibliotecas.	Funciones definidas por el usuario

 © 2016-2022 Altova GmbH

Comando	Descripción	Se aplica a
Pegar	Permite pegar en la biblioteca actual una función definida por el usuario que haya copiado previamente en el portapapeles.	Bibliotecas (Funciones definidas por el usuario)
Opciones	Abre un cuadro de diálogo en el que puede definir o cambiar las opciones de la biblioteca actual.	Bibliotecas
Mostrar todos los documentos abiertos	Si se activa esta opción, la ventana Gestionar bibliotecas muestra todas las asignaciones abiertas, lo que es útil si necesita copiar y pegar funciones de unas asignaciones a otras. De lo contrario, en esta ventana solo aparece la asignación que esté activa.	Siempre
Mostrar todas las rutas de acceso	Si se activa esta opción, la ventana Gestionar bibliotecas muestra la ruta completa de los objetos que contiene. De lo contrario aparecen los nombres, pero no las rutas.	Siempre

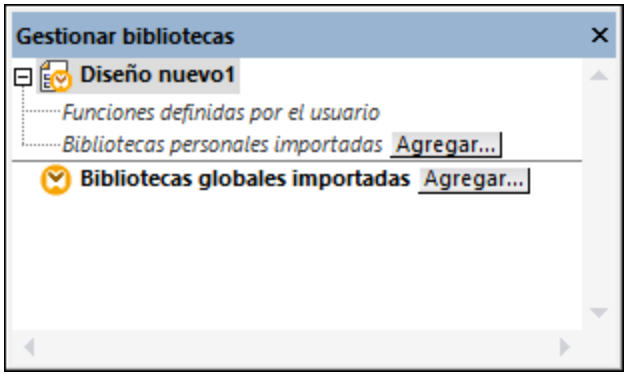
6.2.1 Bibliotecas locales y globales

Puede importar bibliotecas de forma local o global. Globalmente significa que importa la biblioteca a nivel de la aplicación. Si una biblioteca se importa de forma global, puede usar sus funciones para cualquier asignación.

Las importaciones a nivel local se hacen a nivel del archivo de asignación. Por ejemplo, imagine que está trabajando en la asignación **A.mfd** y decide importar todas las funciones definidas por el usuario de la asignación **B.mfd**. En este caso la asignación **B.mfd** se considera como una biblioteca importada localmente en **A.mfd**, por lo que puede usar funciones de **B.mfd** en **A.mfd**. Otro ejemplo de importación a nivel local sería importar funciones de un archivo XSLT en **A.mfd**.

Puede ver y gestionar todas las bibliotecas importadas desde la ventana Bibliotecas. Para importar una biblioteca:

1. Haga clic en el botón **Agregar o quitar bibliotecas**, en la parte inferior de la ventana Bibliotecas 30. Se abre la ventana **Gestionar bibliotecas**.



2. Para importar funciones como biblioteca local (dentro del archivo de asignación actual solamente), haga clic en **Agregar** bajo el nombre de la asignación activa. Para importar funciones como biblioteca global (a nivel de programa), haga clic en **Agregar**, junto a **Bibliotecas globales importadas**. Cuando se importa una biblioteca de forma local, puede hacer que la ruta de acceso al archivo de la biblioteca sea relativa al archivo de asignación. Con las bibliotecas globales la ruta siempre es absoluta.

Nombres de función conflictivos

Es posible que se encuentre en la situación de que un nombre de función esté definido en más de un nivel:

- en la asignación principal
- en una biblioteca importada localmente
- en una biblioteca importada globalmente

En estos casos, MapForce intentará llamar a la función en ese orden para evitar ambigüedades. Es decir, la función definida en la asignación tiene preferencia frente al mismo nombre de función definido en una biblioteca importada localmente. A su vez, la función importada localmente tiene preferencia frente a la función importada a nivel global (si ambas tienen el mismo nombre).

Si existen varias funciones con el mismo nombre, solo se llama a la función "ganadora" según las reglas que acabamos de explicar. Los demás nombres ambiguos se bloquean y aparecen en gris en la ventana Bibliotecas, es decir, no se pueden usar en la asignación.

6.2.2 Rutas relativas de acceso a bibliotecas

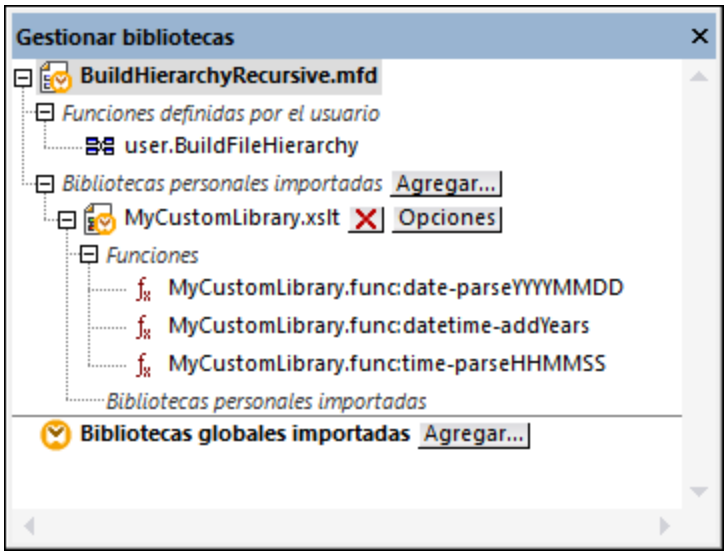
Puede configurar la ruta de cualquier archivo de biblioteca que importe para que sea relativa al archivo de asignación (.mfd), siempre y cuando la biblioteca se haya importado de forma local (no global), como se describe en Bibliotecas locales y globales 555.

> Solo se pueden configurar como relativas las rutas de acceso a bibliotecas que se hayan importado de forma *local* en una asignación. Si se importa una asignación *globalmente*, es decir, a nivel del programa, su ruta será siempre absoluta.

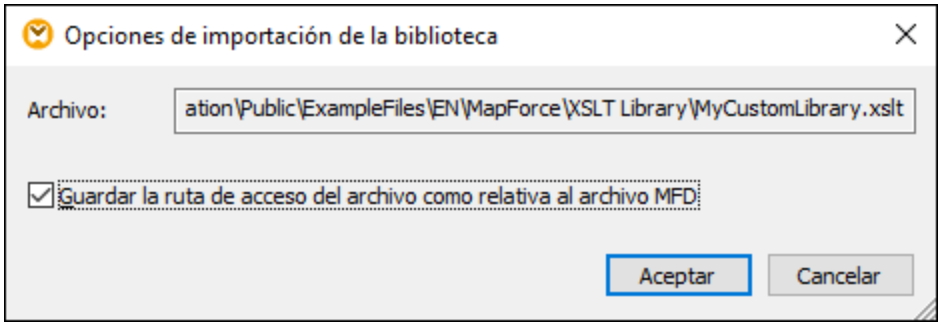
Para convertir la ruta de una biblioteca en relativa al archivo de asignación:

1. Haga clic en **Agregar o quitar bibliotecas**, en la parte inferior de la ventana Bibliotecas. Se abre la ventana Gestionar bibliotecas.

 © 2016-2022 Altova GmbH



2. Haga clic en **Opciones**, junto a la biblioteca en cuestión. (También puede hacer clic con el botón derecho en la biblioteca y seleccionar **Opciones** en el menú contextual.)



3. Marque la casilla *Guardar la ruta de acceso del archivo como relativa al archivo MFD*.

Nota: si esta casilla aparece en gris, asegúrese de que importó la biblioteca de forna local y no global.

Cuando se marca esta casilla, MapForce llevará un registro y actualizará la ruta de cualquier archivo de biblioteca al que se haga referencia cuando guarde la asignación en un directorio nuevo usando el comando de menú **Guardar como**. Si los archivos de biblioteca están en el mismo directorio que la asignación, la referencia de la ruta no se romperá si mueve todo el directorio a una ubicación nueva del disco (*consulte también* *Usar rutas de acceso relativas en un componente* 82).

Tenga en cuenta que la casilla *Guardar la ruta de acceso del archivo como relativa al archivo MFD* indica que las rutas son relativas a la asignación, pero no afecta a las rutas en el código generado. Para saber más sobre las referencias a bibliotecas en el código generado consulte Rutas de acceso según el entorno de ejecución 84.

6.3 Valores predeterminados y funciones de nodo

Los valores predeterminados y las funciones de nodo son especialmente útiles si quiere aplicar la misma lógica de procesamiento a varios elementos descendientes en una estructura. Normalmente se copia y pega la función varias veces en la asignación. Sin embargo, esto puede hacer que la asignación sea poco legible. Los valores predeterminados y las funciones de nodo se pueden aplicar a un solo elemento o a varios a la vez. Los valores predeterminados reemplazan las secuencias vacías. Si la conexión transporta un valor, el valor predeterminado se ignora.

Los valores predeterminados y las funciones de nodo se pueden usar en cualquier componente que tenga una estructura con nodos (como XML, EDI, componentes de combinación, etc.)

Los valores predeterminados y las funciones de nodo solo son compatibles con el lenguaje de transformación BUILT-IN. Es decir, no se pueden ejecutar esas asignaciones desde código generado en C#, C++ o Java, ni desde transformaciones XSLT/XQuery generadas. En el lado servidor puede ejecutar este tipo de asignaciones con MapForce Server Advanced Edition.

Ventajas de los valores predeterminados y las funciones de nodo

Crear funciones de nodo y valores predeterminados es definir una *regla*. Las reglas tienen las siguientes características importantes que además las hacen flexibles y fáciles de usar:

- *Herencia.* Cuando defina una regla en un elemento que tenga descendientes, por defecto la heredarán los descendientes, a no ser que deshabilite esa opción. Si el elemento para el que define la función tiene varios niveles de secundarios anidados, puede escoger entre aplicar esa regla únicamente a los secundarios directos o a todos los secundarios.
- *Filtros.* MapForce aplica reglas de forma condicional basándose en el tipo de datos de cada elemento. Esto permite, por ejemplo, aplicar un valor predeterminado o una función en concreto a todos los elementos de tipo `cadena` y otro valor predeterminado u otra función a todos los elementos de tipo `decimal`. Para más detalles consulte el *Caso 2* de Casos de uso 563. También puede definir filtros más avanzados: Por ejemplo, puede indicar un tipo de datos con el que debe coincidir la función (puede ser una categoría de tipos de datos, como `numérico`) y después filtrar los nodos de ese tipo de datos en función del nombre o tipo de nodo (p. ej. `entero`). Para más detalles consulte el *Caso 5* 568.

Por ejemplo, puede indicar a MapForce que:

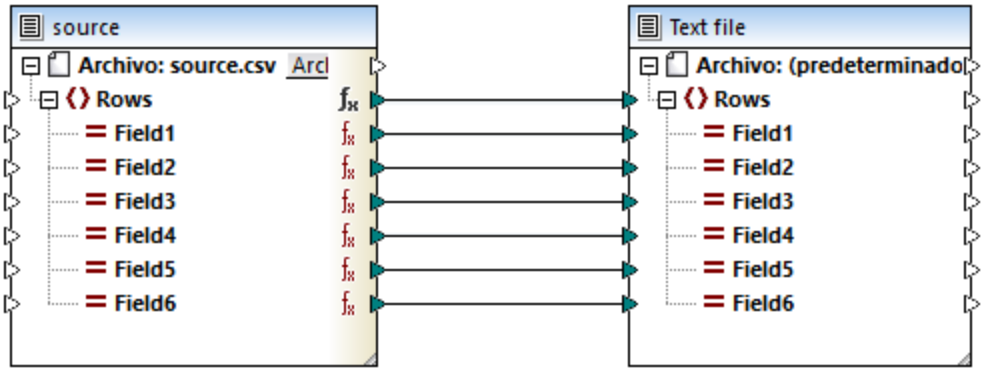
- Cada vez que encuentre un valor null, lo reemplace con otro valor y que haga esto de forma recursiva para todos los elementos descendientes.
- Cada vez que encuentre un valor específico, lo reemplace con otro valor (o una cadena vacía) y que haga esto de forma recursiva para todos los elementos descendientes.
- Reemplace todos los valores null de BD con cadenas de texto vacías o con texto personalizado.
- Anexe un prefijo o sufijo personal a todos los valores que se escriben en un archivo o una BD de destino.

Lado de salida vs. de entrada.

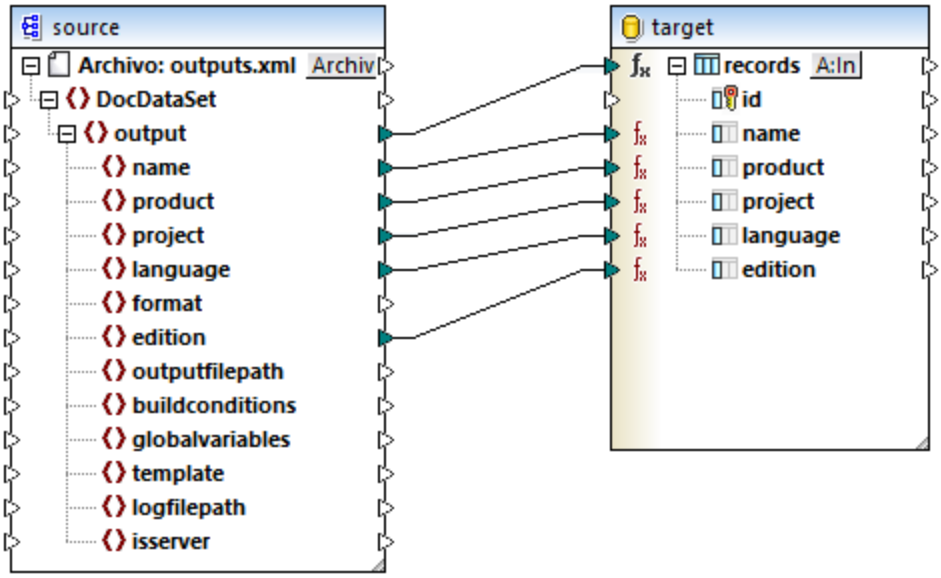
Puede definir las funciones de nodo y los valores predeterminados en el lado de entrada, en el de salida o en ambos lados de un componente, según lo que necesite. En MapForce las asignaciones funcionan como sigue: (i) primero lee datos de un componente de origen (p. ej. un archivo XML), (ii) después puede procesarlos datos (p. ej. con una función) y (iii) por último escribe datos en un componente de destino (p. ej. una BD). SI tiene esto en cuenta puede colocar funciones de nodo y valores predeterminados en varias etapas:

 © 2016-2022 Altova GmbH

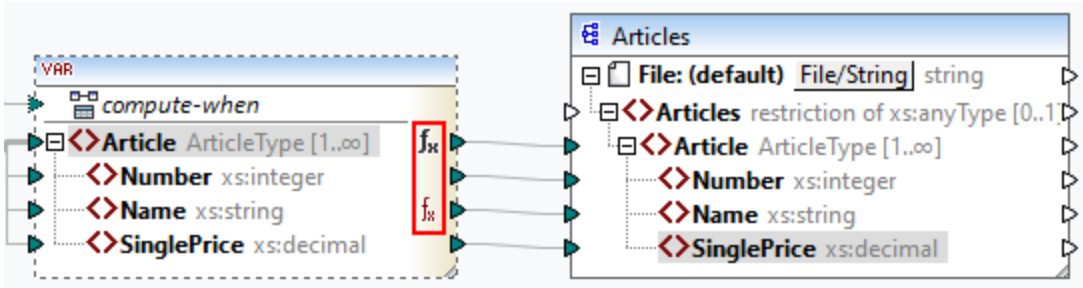
- Inmediatamente *después* de que los datos se lean desde el origen, pero antes de que la asignación los siga procesando, lo que quiere decir que la función o el valor predeterminado se define en el lado de salida del componente de origen (*imagen siguiente*).



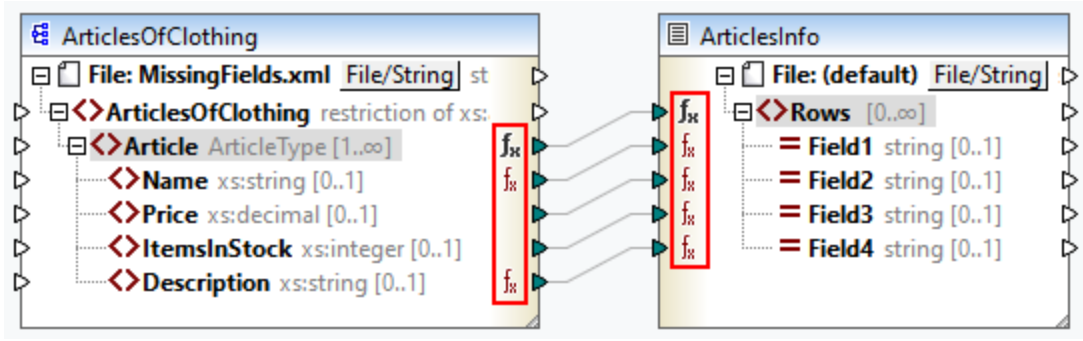
- Inmediatamente *antes* de que se escriban datos en el componente de destino (y después de que haya finalizado todo el procesamiento intermedio), lo cual quiere decir que la función o el valor predeterminado se definen en el lado de entrada del componente de destino (*véase el ejemplo siguiente*).



- En una etapa intermedia del proceso de asignación de datos. Por ejemplo, si la asignación contiene variables de tipo complejo (como una estructura XML), puede reducir todos los valores antes de pasarlos a la estructura XML o inmediatamente después de que se hayan devuelto a la estructura XML (*véase el ejemplo siguiente*).



- En el lado de salida de un componente de origen y en el lado de entrada de un componente de destino. En el ejemplo siguiente se ha definido un valor predeterminado para todos los nodos de tipo `cadena` del componente `ArticlesOfClothing`. En el componente `ArticlesInfo` hemos definido una función de nodo que transforma todos los valores de los nodos de tipo cadena en caracteres en mayúsculas.



Apartados de esta sección

En esta sección explicamos cómo configurar reglas, casos de la vida real en los que los valores predeterminados y las funciones de nodo pueden ser útiles y cómo agregar metadatos de nodo a las funciones de nodo. Estos son los apartados de esta sección:

- Configurar reglas 560
- Casos de uso 563
- Suministrar metadatos del nodo a funciones de nodo 572

6.3.1 Configurar reglas

Puede crear funciones de nodo o valores predeterminados para casi todos los elementos de la asignación de datos. A eso lo llamamos definir una *regla*. Para crear una regla, seleccione el elemento (nodo o campo) para el que quiere definir una regla. Puede tratarse de un solo nodo o de un nodo con nodos secundarios. Cuando defina una regla en un elemento que tenga descendientes, por defecto la heredarán los descendientes, a no ser que deshabilite esa opción.

Aspectos importantes de los valores predeterminados y las funciones de nodo

Estos son los aspectos más importantes que debe tener en cuenta al trabajar con valores predeterminados y funciones de nodo:

 © 2016-2022 Altova GmbH

- Puede crear valores predeterminados y funciones de nodo en la entrada de un componente de destino o en la salida de un componente de origen. Para decidir qué lado le conviene más consulte Entrada vs salida 558.

- Cuando existen varias reglas para el mismo elemento, MapForce aplica la regla que más cercana sea a ese elemento. Para aprender a sobrescribir reglas consulte el *caso 4* en Casos de uso 567.

- Los valores predeterminados y las funciones de nodo requieren que el tipo de conexión entre el origen y el destino esté basada en el origen o en el destino. Las conexiones de copia total no son compatibles. En concreto, las funciones de nodo y los valores predeterminados no se aplican a los descendientes de las conexiones de copia total. Si un nodo principal tiene una conexión de copia total también puede funciones de nodo y valores predeterminados, pero solamente si su nodo ascendiente tiene un valor simple, como un elemento XML con contenido de tipo simple y atributos. Para más información consulte el apartado Tipos de conexión 91.

- Observe que el nodo `Archivo` no admite la creación de valores predeterminados o funciones de nodo

- Dentro de una función de nodo solo se admiten ciertos componentes significativos (como funciones integradas, variables, condiciones if-else). Las estructuras complejas como XML, JSON, EDI o bases de datos no son compatibles. Tampoco es compatible añadir funciones inline definidas por el usuario ni combinar componentes con una función de nodo.

- Una función de nodo puede tener un parámetro de entrada (no más) o ninguno. Ese parámetro de entrada se llama siempre `raw_value`. No elimine nunca el componente de entrada, incluso aunque no necesite ninguna entrada para esa función; de lo contrario aparecerán errores de validación cuando ejecute la asignación de datos. Lo mismo ocurre con la salida de la función. Si necesita restaurar un componente de entrada eliminado por error, ejecute el comando de menú **Función | Insertar componente de entrada**.

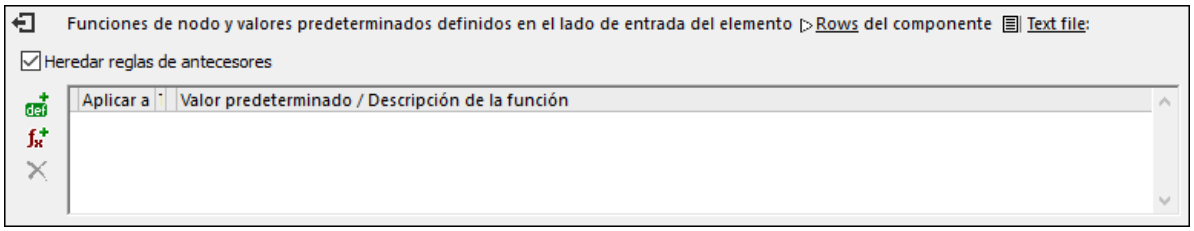
- MapForce permite suministrar metadatos (como un nombre de nodo o una anotación) a las funciones de nodo. Para más detalles consulte Suministrar metadatos del nodo a funciones de nodo 572.

- En el cuadro de diálogo **Funciones de nodo y valores predeterminados** puede elegir de entre varios tipos de datos; algunos de ellos son categorías de tipos, por lo que coincidirán con más tipos. Por ejemplo, el tipo `string` coincide con otros tipos de datos derivados de `string`, como `normalizedString`, `token`, `NCName`, `NMTOKEN`, `IDREF`, `ENTITY`, entre otros. Asimismo, el tipo `decimal` coincidirá con los tipos derivados `integer`, `long`, `short` y otros. La jerarquía de tipos obedece a la recomendación de X3C para el esquema XML.

Crear una regla

Para crear una regla siga estos pasos:

1. Haga clic con el botón derecho en el elemento (nodo) en cuestión y seleccione **Funciones de nodo y predeterminados | Funciones de nodo y predeterminados de entrada (o Funciones de nodo y predeterminados de salida)** en el menú contextual. Elija *Funciones de nodo y predeterminados de entrada* o de *salida* en función del lado del componente en el que quiere crear la función de nodo o el valor predeterminando. También puede hacer clic con el botón derecho en un conector y seleccionar el comando de función de nodo para ese lado. En el panel **Asignaciones** verá la ventana Funciones de nodo, en la que puede definir valores predeterminados y funciones de nodo (*encuadrado en rojo en la imagen siguiente*).



Si el elemento en el que define la regla tiene un elemento primario, es posible que también se hayan definido reglas para este último. Si quiere que herede esas reglas, marque la casilla *Heredar reglas de antecesores* (*imagen siguiente*). Para más información sobre la herencia, consulte el apartado Casos de uso 563.

2. El paso siguiente es decidir si quiere agregar un valor predeterminado (def) o una función(fx). Al hacer clic en la opción que prefiera se crea una regla nueva (una nueva fila en la ventana Funciones de nodo). Para más información sobre cómo configurar reglas consulte *Configurar reglas* más abajo. Si está definiendo una función, podrá ver los parámetros de entrada y de salida en el área de asignación.

Editar/eliminar reglas

Para ver, modificar o eliminar una regla haga clic en el icono fx (negro o rojo) que hay junto al nodo en cuestión. En la ventana Funciones de nodo aparecen todas las reglas que se hayan definido para este nodo. Si agregó una función de nodo, verá el botón Editar en la cuadrícula, con el que puede cambiar la implementación de la función. Si no está el botón Editar, lo más probable es que la función esté definida en un antecesor. En este caso, haga clic en el icono fx que hay junto al elemento para el que se definió la regla.

Para eliminar una regla selecciónela en la cuadrícula de la ventana Funciones de nodo y haga clic en el botón ✕.

Configurar reglas

Cuando seleccione **Funciones de nodo y valores predeterminados | Funciones de nodo y predeterminados de entrada/salida** en el menú contextual (*véase más arriba*) verá la ventana Funciones de nodo, en la que puede configurar funciones de nodo y valores predeterminados. A continuación describimos las opciones disponibles.

- *Aplicar a:* Indica si la regla debe aplicarse al elemento actual o a todos los elementos descendientes independientemente de su profundidad, o únicamente a los descendientes directos. Si el elemento seleccionado en el paso 1 no tiene descendencia, entonces la única opción es *Elemento actual*.

- *Tipo de datos:* Haga clic en el botón Elipsis ... y seleccione un tipo de datos del cuadro de diálogo. La regla (predeterminado o función de nodo) se aplicará únicamente a elementos que tengan este tipo de datos (o uno derivado). Si el elemento seleccionado en el paso 1 no tiene descendencia, entonces la única opción disponible será el tipo de datos del elemento.

- *Valor predeterminado / Descripción de la función:* Si está definiendo un valor predeterminado, teclee aquí el valor predeterminado que quiere establecer para el elemento seleccionado (y todos sus descendientes, si procede). Para establecer una cadena vacía como valor predeterminado, deje este campo vacío. Si está definiendo una función, este campo existe únicamente a efectos informativos. Muestra un resumen de la función.

 © 2016-2022 Altova GmbH

Para volver a la ventana Funciones de nodo haga clic en el botón que hay en la esquina superior izquierda de la ventana o pulse **Escape**.

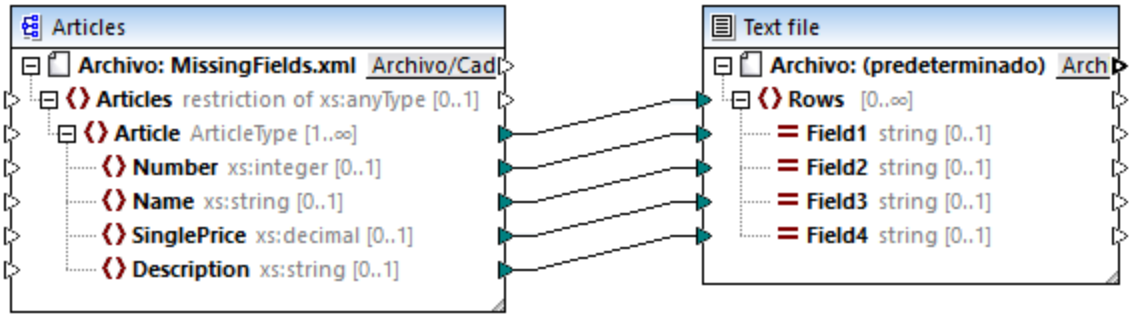
Ayudas visuales

Para que comprenda mejor la diferencia entre reglas definidas y aplicadas, MapForce ofrece las siguientes ayudas visuales en la asignación (*tabla siguiente*).

Icono	Descripción
	Este icono indica que para este elemento existe una regla definida que puede afectar a todos sus descendientes. Haga clic en el icono para modificar o eliminar la regla.
	Este icono indica que el elemento hereda una regla definida a nivel de alguno de sus antecesores.
	Este icono indica que existe una regla definida para este elemento que también se le aplica. Este icono normalmente aparece cuando se define un valor predeterminado para un único nodo.
	Este icono indica que, aunque una regla se pueda aplicar a este elemento, esta está bloqueada a propósito. Este icono solo aparece si la herencia está bloqueada y no hay más reglas definidas para este nodo. Si se aplica una regla de algún ancestro, el icono tiene prioridad.
	Este icono indica que aunque se le pueda aplicar una regla a este elemento, está inactiva. Por ejemplo, este icono puede aparecer para elementos que todavía no están conectados en la asignación.
	Este icono indica que la regla está inactiva actualmente porque no se han encontrado nodos que coincidan o estén conectados al elemento. Con *coincidir* queremos decir: Antes de aplicar una función o un valor predeterminado, MapForce busca el tipo de datos que haya definido; si también indicó un filtro para encontrar un nombre de nodo específico, por ejemplo, MapForce también buscará ese nombre de nodo.

6.3.2 Casos de uso

En este apartado describimos distintos casos de uso en los que los valores predeterminados y las funciones de nodo pueden ser útiles. Para probar estos casos de uso necesita la asignación simple de ejemplo: **`Tutorial\MissingFields.mfd`**. Nuestro punto de partida es la asignación que ve más abajo.



Algunos de los nodos del archivo de origen están vacíos. En este punto la asignación tiene este aspecto:

T-Shirt	25,3	20	Available in black, blue, and red
Shirt	70,3	60	
Pants			Limited stock
Jacket	57,5	40	

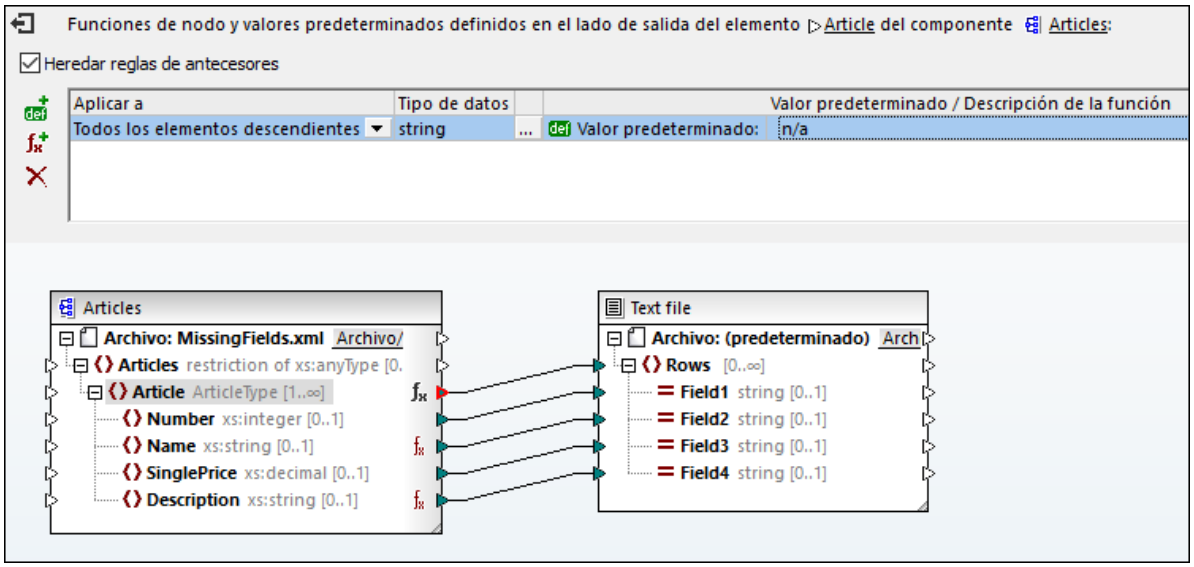
En los casos que describimos más abajo puede ver cómo definir un valor predeterminado para todos los nodos de tipo `cadena`, definir valores predeterminados para diferentes tipos de datos, bloquear reglas para nodos específicos, sobrescribir reglas heredadas, definir una función de nodo y aplicarla condicionalmente a los nodos relevantes, y aplicar valores predeterminados a nodos sin conexión.

Caso 1: definir un valor predeterminado para todos los nodos de tipo cadena

En nuestra asignación, algunos nodos de tipo `cadena` están vacíos. Nuestro objetivo es definir un valor predeterminado `n/a` que reemplace esas cadenas vacías. Siga estos pasos:

1. Haga clic con el botón derecho en el elemento `Article` y seleccione **Funciones de nodo y valores predeterminados | Funciones de nodo y predeterminados de salida** en el menú contextual.
2. Haga clic en la opción def en la ventana Funciones de nodo y teclee `n/a` en el campo Valor predeterminado (*imagen siguiente*).

 © 2016-2022 Altova GmbH



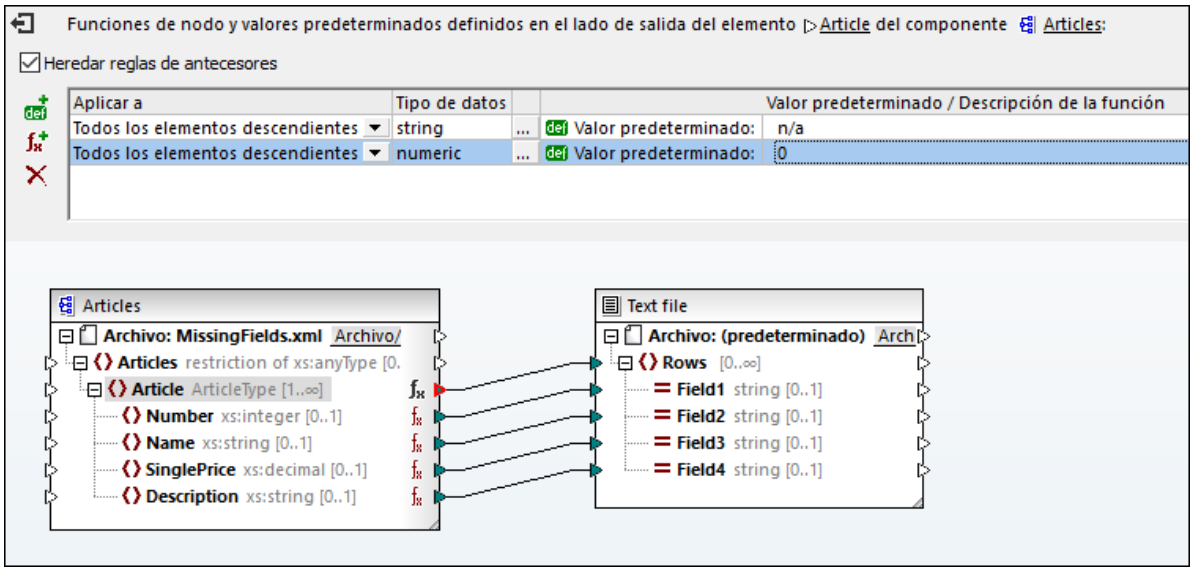
En el componente `ArticlesOfClothing`, los elementos `Name` y `Description` son de tipo `cadena`, por lo que pueden tomar el valor predeterminado. Esto quiere decir que ahora el icono aparece junto a esos nodos. Los valores predeterminados se aplican solamente a los valores vacíos, por lo que solamente dos nodos `Field4` tienen el valor predeterminado `n/a` en el resultado (*texto resaltado en amarillo*).

T-Shirt	25,3	20	Available in black, blue, and red
Shirt	70,3	60	n/a
Pants			Limited stock
Jacket	57,5	40	n/a

Caso 2: definir los valores predeterminados para distintos tipos de datos

Además de definir un valor predeterminado para nodos de un tipo en concreto, también puede definir otro para tipos de datos distintos. Por ejemplo, uno de los artículos tiene valores vacíos en `Price` y en `ItemsInStock`, que son de tipo numérico. Ahora nuestro objetivo es reemplazar esos valores vacíos con `0`. Siga estos pasos:

1. Repita los pasos del *Caso 1*. Eso añade un valor predeterminado a todos los nodos de tipo `cadena` que tengan valores vacíos.
2. Agregue otro valor predeterminado y seleccione `numérico` como tipo de datos.
3. Teclee `0` en el campo *Valor predeterminado* (*imagen siguiente*).



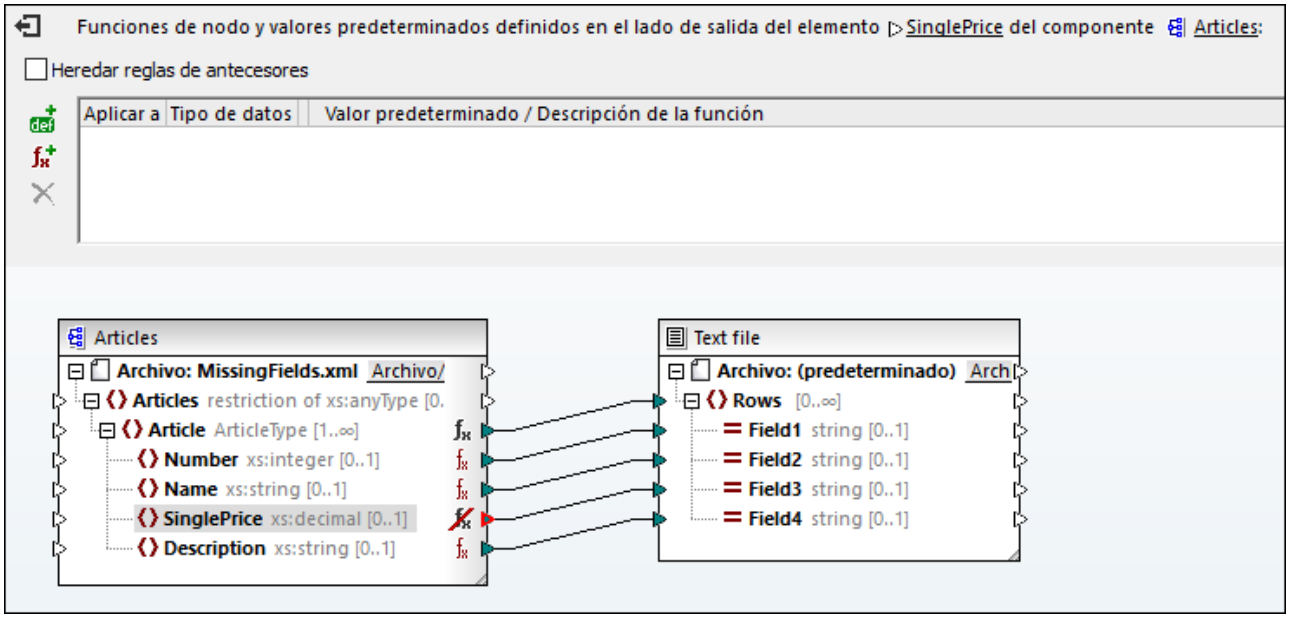
El resultado ahora tiene este aspecto (*observe las partes resaltadas en amarillo*):

T-Shirt	25,3	20	Available in black, blue, and red
Shirt	70,3	60	n/a
Pants	0	0	Limited stock
Jacket	57,5	40	n/a

Nota: según la especificación XML Schema, `entero` se deriva de `decimal` y ambos pertenecen al tipo de datos numérico. En nuestro ejemplo la regla se aplica a los dos elementos, `ItemsInStock` y `Price` si seleccionó `numérico` como tipo de datos en la ventana Funciones de nodo. Si selecciona `decimal` como tipo de datos, la regla también se aplica a los dos elementos. Sin embargo, si selecciona `entero` como tipo de datos, la regla se aplica solamente al elemento `ItemsInStock`.

Caso 3: crear reglas de bloqueo para nodos específicos

En este caso queremos aplicar valores predeterminados a todos los nodos de tipo cadena y numéricos, pero no queremos que ese valor se aplique al elemento `Price`. Para alcanzar ese objetivo reproduzca los pasos del *Caso 2*, haga clic en el elemento `Price` y desmarque la casilla *Heredar reglas de antecesores*. En la asignación siguiente, el elemento `Price` ya no hereda las reglas de su antecesor, `Article`. Por lo tanto, el icono aparece junto al elemento `Price` (*imagen siguiente*).

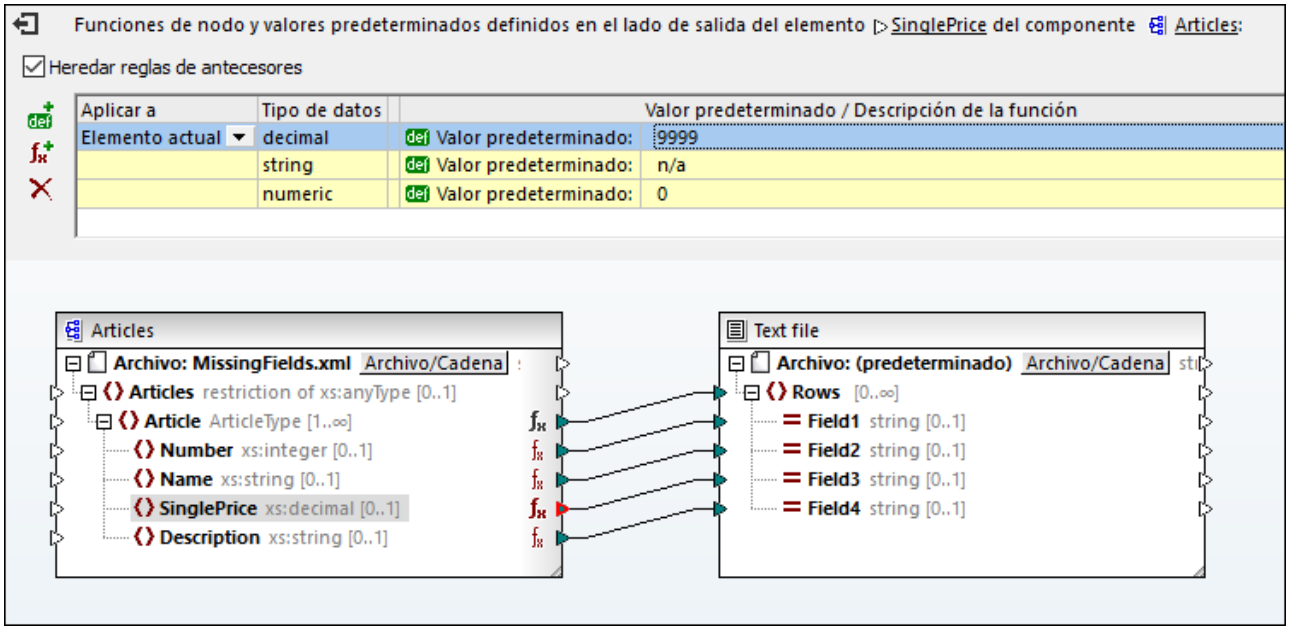


En el resultado siguiente puede ver que el valor vacío del elemento `Price`, para el que hemos bloqueado la regla, se ha asignado a `Field2` (*observe las partes resaltadas*).

T-Shirt	25,3	20	Available in black, blue, and red
Shirt	70,3	60	n/a
Pants		0	Limited stock
Jacket	57,5	40	n/a

Caso 4: sobrescribir las reglas heredadas

En este caso queremos definir valores predeterminados para todos los nodos de tipo cadena y numéricos, salvo para el elemento `Price`, cuyo valor predeterminado debe ser `100`. Para ello repita los pasos del *Caso 2*, haga clic en `Price` en el componente `ArticlesOfClothing`, agregue un valor predeterminado nuevo e introduzca `100`, como se ve en la imagen siguiente. Las reglas heredadas tienen un fondo amarillo.



La regla definida directamente en un nodo tiene prioridad frente a las que se crean a nivel del antecesor. Por tanto, el resultado ahora tiene este aspecto (*observe las partes resaltadas en amarillo*):

T-Shirt	25,3	20	Available in black, blue, and red
Shirt	70,3	60	n/a
Pants	100	0	Limited stock
Jacket	57,5	40	n/a

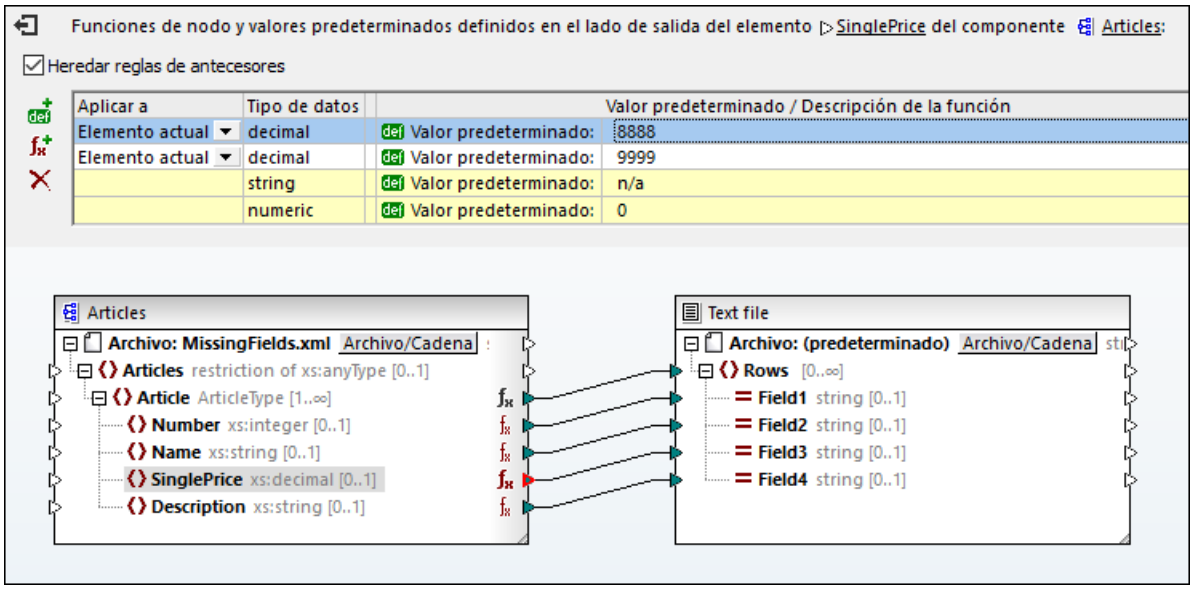
Si definió más de una regla para un mismo nodo, la que esté más arriba en la cuadrícula es la que se aplica al nodo actual. Puede arrastrar esas reglas si quiere cambiar su orden.

Caso 5: Crear una función de nodo y aplicarla condicionalmente

En este caso vamos a explicar cómo crear una función de nodo y aplicarla condicionalmente. Para ello usaremos la función **concat**, con la que añadimos texto a los nodos de tipo cadena. Siga las instrucciones a continuación.

1. Haga clic con el botón derecho en el elemento `Article` y seleccione **Funciones de nodo y valores predeterminados | Funciones de nodo y predeterminados de salida** en el menú contextual.
2. Para agregar una función de nodo haga clic en el icono y cree la función que ve más abajo.

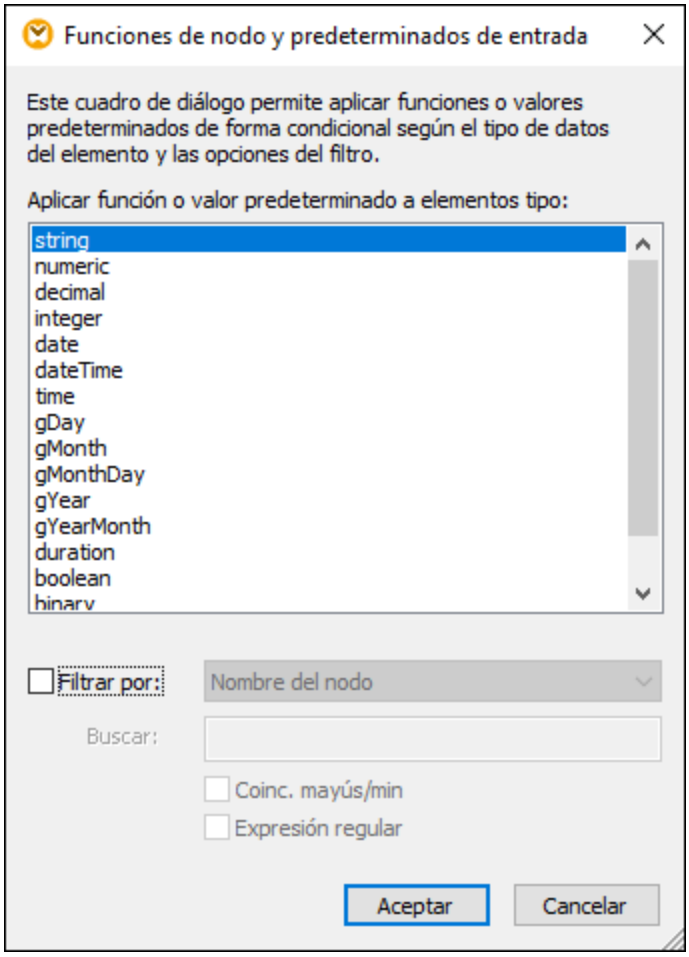
 © 2016-2022 Altova GmbH



En este punto todavía no hemos filtrado los nodos y el resultado tiene el siguiente aspecto (*observe las partes resaltadas*):

T-Shirt (XXS, L-XXL) Shirt (XXS, L-XXL) Pants (XXS, L-XXL) Jacket (XXS, L-XXL)	25,3 70,3 57,5	20 60 40	Available in black, blue, and red (XXS, L-XXL) Limited stock (XXS, L-XXL)

En el resultado puede ver que la regla se ha aplicado a los dos elementos de tipo `cadena`: `Name` y `Description`. Ahora queremos aplicar la regla solamente al elemento `Name`. Para ello vaya a la ventana Funciones de nodo y haga clic en el botón ... en la columna *Tipo de datos y filtro* Se abre el cuadro de diálogo **Funciones de nodo y valores predeterminados**, donde puede cambiar los tipos de datos e indicar las opciones de filtrado. Defina el filtro como se ve en la imagen siguiente.



Ahora el resultado tiene este aspecto:

T-Shirt (XXS, L-XXL) Shirt (XXS, L-XXL) Pants (XXS, L-XXL) Jacket (XXS, L-XXL)	25,3 70,3 57,5	20 60 40	Available in black, blue, and red Limited stock

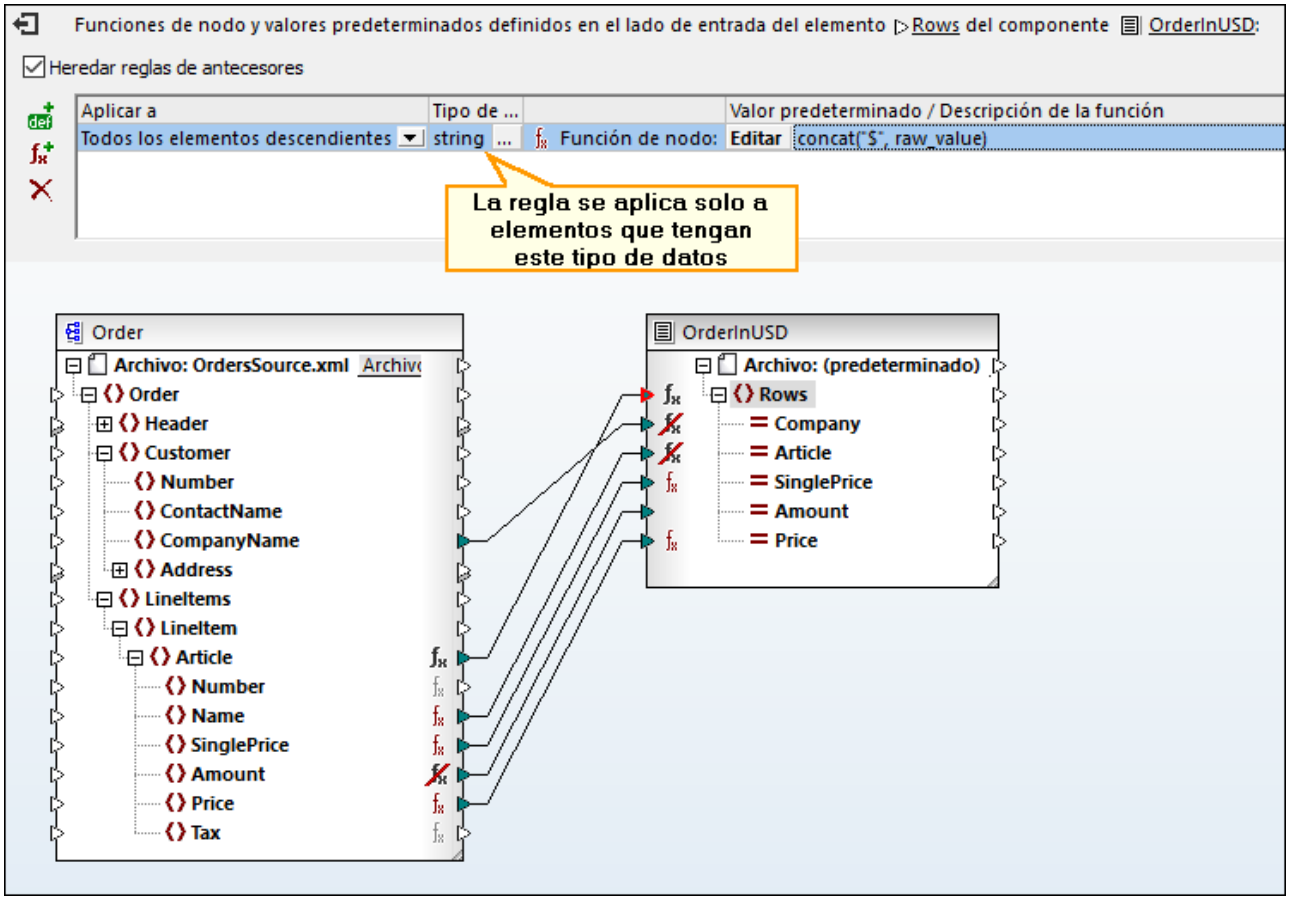
Filtrar por tipo de nodo
Esta opción permite acotar el tipo de datos que seleccionó en el cuadro de diálogo **Funciones de nodo y valores predeterminados**. Por ejemplo, si ha elegido aplicar la función de nodo a nodos numéricos, puede indicar un subtipo (p. ej.: *Filtrar por/Nombre del nodo:* `xs:decimal`).

Filtrar con expresiones regulares
Puede usar expresiones regulares para crear filtros más avanzados (p. ej. Para buscar varios nombres de nodo o de tipos). Para más información consulte Expresiones regulares 627. Tenga en cuenta que:

- Los caracteres ancla `^` y `$` están implícitos, por lo que no es necesario indicarlos en la caja de texto *Buscar*.
- Puede decidir si quiere distinguir entre mayúsculas y minúsculas en la casilla *Coinc. mayús/min*. Por tanto, el carácter flag `i` no se admite.
- Tampoco tiene sentido buscar en varias líneas al filtrar nodos. Por tanto, tampoco se admite el carácter flag `m`.

Caso 6: crear valores predeterminados para nodos sin conexión

MapForce permite aplicar valores predeterminados a nodos sin conexión, pero solamente si el nodo principal y/o algún elemento del mismo nivel están conectados. Aquí puede ver un ejemplo:



En la asignación anterior definimos un valor predeterminado para el nodo `Field4` en el componente `ArticlesInfo` . En este punto `Field4` no tiene ninguna conexión de entrada. Los nodos principales y del mismo nivel están conectados. Si `Field4` sigue sin conexión, el valor predeterminado sobrescribe todos los valores de `Description` en el resultado (*imagen siguiente*). Si conecta `Description` y `Field4`, el valor predeterminado solo se aplica a secuencias vacías.

T-Shirt	25,3	20	n/a
Shirt	70,3	60	n/a
Pants			n/a
Jacket	57,5	40	n/a

Nota: los valores predeterminados no afectan a los nodos sin conexión de los componentes JSON.

6.3.3 Suministrar metadatos del nodo a funciones de nodo

Es posible que existan casos en los que quiera que una función de nodo haga algo basándose en *metadados del nodo*. Por metadatos del nodo entendemos información diversa sobre el nodo en el que se aplica la función (por ejemplo, un nombre de nodo, la longitud de su valor y su precisión), que se almacena en el esquema. En la tabla siguiente puede ver todos los parámetros de metadatos que puede usar en una función de nodo. Recuerde que algunos parámetros de metadatos afectan solamente a ciertos tipo de datos. MapForce le avisa si intenta usar unos metadatos que son incompatibles con el nodo actual.

Parámetros de los metadatos	Descripción
`node_name`	Da el nombre del nodo en cuestión. Estos metadatos se aplican a todos los nodos. En el caso de XML, `node_name` es el nombre del elemento o atributo actual. En el caso de CSV, `node_name` es el campo CSV. En el caso de bases de datos, `node_name` es el nombre de la columna de tabla.
`node_annotation`	Da la anotación de texto que aparece junto al valor del nodo en los resultados. Estos metadatos se aplican a todos los nodos.
`node_minLength`	Da el valor del aspecto `minLength` del tipo de datos del nodo. Se aplica a los nodos XML y de texto con los tipos correspondientes. Para más información consulte la recomendación de XML Schema.
`node_maxLength`	Da el valor del aspecto `maxLength` del tipo de datos del nodo. Se aplica a nodos XML y de texto con los tipos correspondientes. Para más información consulte la recomendación de XML Schema.
`node_totalDigits`	Da el valor del aspecto `totalDigits` del tipo de datos del nodo. Se aplica a los nodos XML con los tipos correspondientes. Para más información sobre el aspecto `totalDigits` consulte la recomendación de XML Schema.
`node_fractionDigits`	Da el valor del aspecto `fractionDigits` del tipo de datos del nodo. Se aplica a nodos XML con los tipos correspondientes. Para más información sobre el aspecto `totalDigits` consulte la recomendación de XML Schema.
`node_length`	Da a longitud del tipo de datos del nodo. Se aplica a los campos de BD con los tipos correspondientes. Para más información consulte la documentación de Microsoft.
`node_precision`	Da la precisión del tipo de datos del nodo. Se aplica a los campos de BD con los tipos correspondientes. Consulte la documentación de Microsoft para obtener más información.
`node_scale`	Da la el número máximo de dígitos a la derecha del punto decimal. Se aplica a los campos de BD con los tipos correspondientes. Consulte la documentación de Microsoft para obtener más información.

 © 2016-2022 Altova GmbH

Nota: el tipo de datos del parámetro de metadatos debe coincidir con el tipo de datos del nodo correspondiente. De lo contrario no se puede ejecutar la función.

Suministrar metadatos a una función de nodo
Para suministrar metadatos a una función de nodo comience por crear una nueva función de nodo 561 y después haga clic en la opción **Agregar detalles del nodo**, que se encuentra bajo la ventana Funciones de nodo. Esta acción abre el cuadro de diálogo **Insertar componente de entrada**, donde puede seleccionar los parámetros de los metadatos. En vez de la opción **Agregar detalles del nodo**, también puede elegir una de estas:

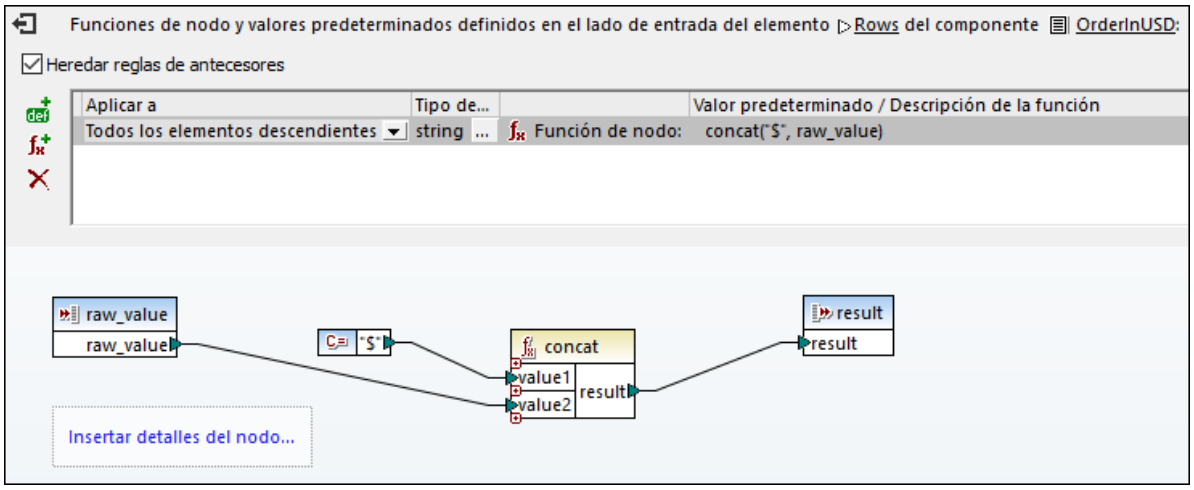
- Haga clic con el botón derecho en un área vacía de la asignación y seleccione **Insertar componente de entrada** en el menú contextual.
- Haga clic en el comando de la barra de herramientas .
- Haga clic en **Insertar componente de entrada** en el menú **Función**.

Metadatos no compatibles con nodos
Si un parámetro de metadatos no es compatible con un nodo, puede indicar a MapForce que devuelva una secuencia vacía o que no aplique la función de nodo (puede encontrar ambas opciones en el cuadro de diálogo **Insertar componente de entrada con nodo específico**). Ahora debe ocuparse de la secuencia vacía; de lo contrario la función de nodo no devolverá ningún valor. Normalmente tendría que usar funciones de secuencia (como `substitute-missing` 707 y `exists` 683) u otros tipos de componente para seguir procesando la secuencia vacía.

Ejemplo

En el ejemplo siguiente puede ver cómo suministrar un texto de anotación a una función de nodo. Para este ejemplo necesita la asignación:
`<Documentos>\Altova\MapForce2023\MapForceExamples\Tutorial\MissingFields.mfd`. En **`MissingFields.xsd`** (en la que se basa **`MissingFields.xml`**) hemos añadido esta anotación para el elemento `Name`: `Article name from catalog Spring 2022`. Nuestro objetivo es pasar esta anotación a una función de nodo y verla en el resultado. Siga las instrucciones a continuación.

1. Cree una función de nodo 561 para el elemento `Name` del componente `ArticlesOfClothing`.
2. Replique la función de más abajo. Para agregar un parámetro `node_annotation` haga clic en **Agregar detalles del nodo** (*encuadrado en rojo más abajo*) y seleccione este parámetro de la lista. Para averiguar cómo agregar y manipular funciones consulte Fundamentos de las funciones 550.



Nota: por defecto, las estructuras anidadas con mucha profundidad no se escanean completamente para ahorrar memoria y mejorar la experiencia del usuario. Si el componente al que está aplicando la función tiene este tipo de estructuras anidadas profundas, puede expandir los nodos relevantes en la asignación para que MapForce los tenga en cuenta. En este caso la aplicación incluirá los nodos expandidos cuando agregue un parámetro de metadatos nuevo y la advertencia desaparecerá. La función de nodo debe estar conectada para que funcione; los elementos expandidos pero no conectados no se tienen en cuenta.

Resultados

A continuación puede ver que la anotación se ha añadido a los valores de los elementos `Name`.

```
T-Shirt (Article name from catalog Spring 2022)   25.5  20    Available in black, blue,
and red
Shirt (Article name from catalog Spring 2022)     70.3  60

Pants (Article name from catalog Spring 2022)                  Limited
stock
Jacket (Article name from catalog Spring 2022)    57.5  40
```

 © 2016-2022 Altova GmbH

6.4 Funciones definidas por el usuario

Las funciones definidas por el usuario son funciones personales que se definen una vez y se pueden reutilizar varias veces dentro de la misma asignación o en varias asignaciones. Las funciones definidas por el usuario son como asignaciones de datos a pequeña escala: Suelen consistir en uno o más parámetros de entrada, algunos componentes intermediarios para procesar datos y una salida que devuelve datos al emisor de la llamada. El emisor de las llamadas es la asignación principal u otra función definida por el usuario.

Ventajas de las funciones definidas por el usuario

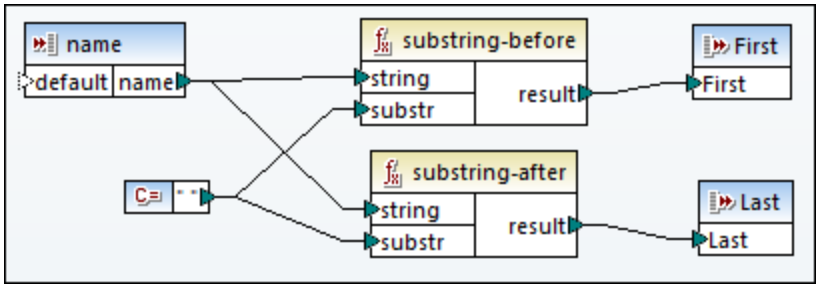
Las funciones definidas por el usuario tienen las ventajas siguientes:

- Se pueden reutilizar dentro de la misma asignación o en asignaciones distintas.
- Pueden hacer una asignación más legible. Por ejemplo, puede dividir partes de la asignación en componentes más pequeños y abstraer los detalles de la implementación. Puede observar este principio en el diagrama siguiente:

- Las funciones definidas por el usuario son funciones flexibles que permiten procesar cadenas, números, fechas y otros datos de forma personalizada y que amplía las funciones integradas de MapForce. Por ejemplo, puede que quiera concatenar o dividir texto de una forma concreta, hacer cálculos, modificar fechas y horas, etc.
- Otro uso común de las funciones definidas por el usuario es para buscar un campo en un archivo XML, una BD u otro formato de datos compatible con su edición de MapForce. Para más detalles consulte Implementación de la búsqueda 589.
- Puede llamar a las funciones definidas por el usuario de forma recursiva (por ejemplo, llamándose a sí misma). Para ello debe definir esa función definida por el usuario como función estándar (no inline) 578. Las funciones definidas por el usuario recursivas 586 permiten, entre otras tareas complejas de las asignaciones, iterar sobre estructuras de datos con una profundidad de *N* secundarios donde *N* no se conoce de antemano (véase el ejemplo Ejemplo: búsqueda recursiva).

Ejemplo

A continuación puede ver un ejemplo de una función definida por el usuario que divide una cadena en dos. Esta función definida por el usuario es parte de una asignación mayor llamada **`MapForceExamples\ContactsFromPO.mfd`**. La función definida por el usuario toma el nombre como parámetro (p.e., `Helen Smith`), aplica las funciones integradas **`substring-before`** y **`substring-after`** y devuelve dos valores: `Helen` y `Smith`.



Apartados de esta sección

En esta sección explicamos cómo trabajar con funciones definidas por el usuario; la sección sección se divide en estas subsecciones:

- Crear funciones definidas por el usuario 576
- Parámetros en funciones definidas por el usuario 581
- Ejemplo: búsqueda recursiva 586
- Implementación de la búsqueda 589

6.4.1 Crear funciones definidas por el usuario

En esta sección explicamos cómo crear, importar, editar, copiar, pegar y eliminar funciones definidas por el usuario.

Crear una función definida por el usuario

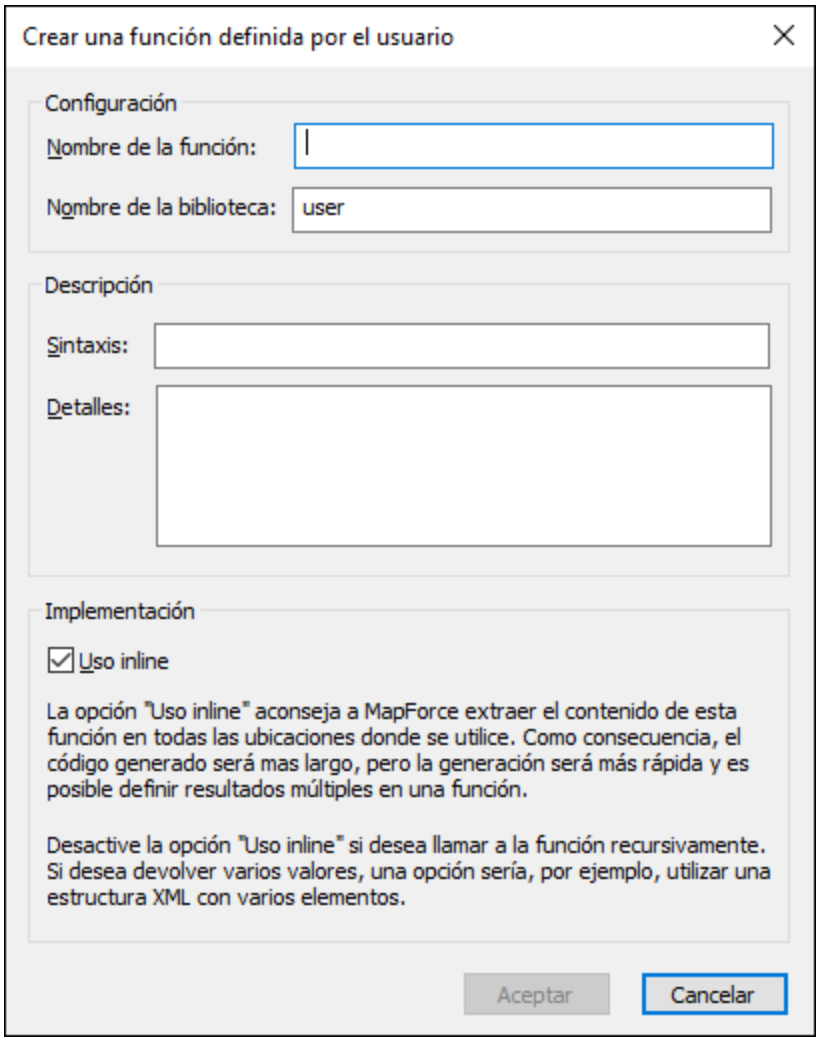
Aquí explicamos cómo crear una función definida por el usuario desde cero y a partir de componentes que ya existen. El requisito mínimo es un componente de salida al que haya datos conectados. En cuanto a los parámetros de entrada, una función puede no tener ninguno o tener uno o más. Los parámetros de entrada o de salida pueden ser de tipo simple (como una cadena de texto o un número entero) o de tipo complejo (una estructura). Para más información sobre parámetros simples y complejos, consulte el apartado Parámetros en funciones definidas por el usuario 581.

Crear una función definida por el usuario desde cero
Para crear una función definida por el usuario desde cero siga estos pasos:

1. Seleccione **Función | Crear una función definida por el usuario**. También puede hacer clic en el botón de la barra de herramientas.

2. Introduzca la información requerida en los campos correspondientes del cuadro de diálogo **Crear una función definida por el usuario** (*tabla siguiente*).

 © 2016-2022 Altova GmbH



A continuación mostramos las opción disponibles.

- *Nombre de la función:* Campo obligatorio. Introduzca un nombre para la función definida por el usuario que quiere crear. Caracteres válidos: caracteres alfanuméricos (a-z, A-Z, 0-9), barra baja (_), guion (-) y dos puntos (:).
- *Nombre de la biblioteca:* Campo obligatorio. Introduzca el nombre de la biblioteca a la que debe pertenecer la función (en la ventana Bibliotecas 30). Si no indica ninguna biblioteca, la función aparecerá en una biblioteca predeterminada llamada `user`.
- *Sintaxis:* Campo opcional. Introduzca en formato texto una breve descripción de la sintaxis de la función (por ejemplo, los parámetros que se esperan). Este texto se mostrará junto a la función en la ventana **Bibliotecas** y no afecta a la implementación de la función.
- *Detalle:* Campo opcional. Introduzca en formato texto una descripción libre de la función. Este texto se mostrará cuando pase el cursor sobre la función en la ventana **Bibliotecas** o en otros contextos.
- *Uso inline:* Marque esta casilla si quiere crear la función como inline. Para más información, consulte el apartado *Funciones definidas por el usuario inline y regulares*.

3. Haga clic en **Aceptar**. La función aparecerá automáticamente en la biblioteca que haya indicado antes. Ahora también puede crear la nueva función en la ventana de asignaciones (esta es una

asignación independiente a la que se hace referencia como la *asignación de la función*). Como toda función necesita un componente de salida, la asignación de la función incluye uno por defecto.

4. Agregue a la asignación de la función todos los componentes requeridos en la definición de la función como haría para cualquier asignación estándar.

Para usar la función definida por el usuario en una asignación arrástrela desde la ventana **Bibliotecas** hasta el área principal de asignación. Consulte también *Llamar e importar funciones definidas por el usuario* más abajo.

Crear una función definida por el usuario a partir de componentes ya existentes
Para crear una función definida por el usuario a partir de componentes ya existentes siga estos pasos:

1. Seleccione varios componentes de la asignación dibujando un rectángulo con el cursor. También puede seleccionar varios componentes haciendo clic en cada uno de ellos mientras mantiene pulsada la tecla **Ctrl**.
2. Seleccione **Función | Crear una función definida por el usuario a partir de la selección**. También puede hacer clic en el botón de la barra de herramientas.
3. Siga los pasos 2-4 que hemos descrito en *Crear una función definida por el usuario desde cero*.

Funciones definidas por el usuario inline y regulares
Existen dos tipos de funciones definidas por el usuario: *inline* y de tipo *estándar*. Puede indicar si una función debe ser inline o de tipo estándar al crearla. Las funciones inline y de tipo estándar se comportan de forma diferente a la hora de generar código, en cuanto a recursividad y al tener varios parámetros de salida. En la tabla siguiente puede ver las diferencias principales entre estos dos tipos de funciones.

Funciones inline (borde discontinuo)	Funciones regulares (borde sólido)
El código de la función definida por el usuario se inserta en todos los lugares en los que se llama a la función, lo que aumenta considerablemente el tamaño del código.	El código de la función definida por el usuario se genera una vez y sus componentes de entrada se le pasan como valores de parámetros. Si se llama varias veces a la función, esta se evalúa cada vez con los valores de parámetro
Las funciones inline pueden tener varios componentes de salida y por tanto devolver varios valores.	Las funciones de tipo estándar solo pueden tener un componente de salida. Para que devuelvan varios valores puede declarar que la salida es de tipo complejo (por ejemplo, estructura XML), lo que permitiría pasar varios valores al usuario.
No se puede llamar a las funciones inline de forma recursiva.	Se puede llamar de forma recursiva a las funciones de tipo estándar.
Las funciones inline no permiten establecer un contexto de prioridad 883 en un parámetro.	Las funciones estándar permiten establecer un contexto de prioridad en un parámetro.

Nota: cambiar el tipo de una función definida por el usuario de inline a estándar y viceversa puede afectar el contexto de la asignación 874, lo que a su vez puede afectar al resultado de la asignación.

 © 2016-2022 Altova GmbH

Importar y llamar a funciones definidas por el usuario

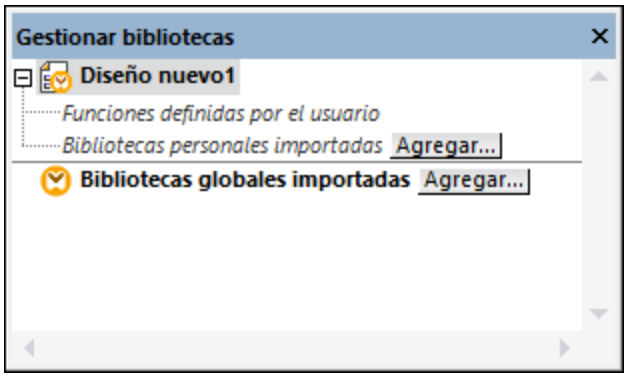
Una vez haya creado una función definida por el usuario puede llamarla desde la asignación en que la creó o desde cualquier otra.

Llamar a una función definida por el usuario desde la misma asignación
Para llamar a una función definida por el usuario desde la misma asignación siga estos pasos:

1. Busque la función en la ventana **Bibliotecas**, bajo la biblioteca que indicó al crear la función. Para ello comience a escribir su nombre en la ventana **Bibliotecas**.
2. Arrastre la función desde la ventana **Bibliotecas** hasta la asignación principal Ahora puede conectarla a todos los parámetros necesarios.

Importar una función definida por el usuario desde otra asignación
Para importar una función definida por el usuario desde otra asignación siga estos pasos:

1. Haga clic en el botón **Agregar o quitar bibliotecas**, en la parte inferior de la ventana Bibliotecas 30. Se abre la ventana **Gestionar bibliotecas**.



2. Para importar funciones como biblioteca local (dentro del archivo de asignación actual solamente), haga clic en **Agregar** bajo el nombre de la asignación activa. Para importar funciones como biblioteca global (a nivel de programa), haga clic en **Agregar**, junto a **Bibliotecas globales importadas**. Cuando se importa una biblioteca de forma local, puede hacer que la ruta de acceso al archivo de la biblioteca sea relativa al archivo de asignación. Con las bibliotecas globales la ruta siempre es absoluta.

3. Navegue hasta el archivo `.mdf` que contiene la función definida por el usuario y haga clic en **Abrir**. Un mensaje le informará de que se añadió una biblioteca nueva a la que puede acceder desde la ventana **Bibliotecas**.

Ahora puede arrastrar cualquiera de las funciones importadas en la asignación actual desde la ventana **Bibliotecas** hasta la asignación para usarlas. Para más información consulte Gestionar bibliotecas de funciones 553.

Asignaciones y credenciales (Enterprise Edition)
Si el archivo `.mdf` que se importa contiene credenciales, estas aparecen como importadas (con un fondo amarillo) en el Gestor de credenciales. Por defecto, las credenciales importadas no se guardan en la asignación, pero si quiere hacerlo basta con crear una copia local y guardar esas credenciales en la asignación principal.

Editar funciones definidas por el usuario

Para edita una función definida por el usuario siga estos pasos:

1. Abra la asignación que contiene la función definida por el usuario.
2. Haga doble clic en la barra del título de la función definida por el usuario en la asignación. La ventana Asignación cambia para mostrar el contenido de la función, en el que puede añadir, editar o eliminar componentes como desee.
3. Para cambiar las propiedades de la función (como el nombre o la descripción), haga clic con el botón derecho en un punto vacío de la asignación y seleccione **Configurar asignación** en el menú contextual. Otra opción es hacer clic en el botón de la barra de herramientas .

También puede editar una función haciendo doble clic en su nombre en la ventana **Bibliotecas**. Sin embargo, solo se pueden abrir de esta forma las funciones que estén en el documento activo en ese momento. Si hace doble clic en una función definida por el usuario que se creó en otra asignación, se abrirá esa asignación en una ventana nueva. Si edita o elimina una función definida por el usuario que se ha importado a varias asignaciones, todas esas asignaciones se verán afectadas por el cambio.

Para volver a la asignación principal haga clic en el botón , que está en la esquina superior izquierda de la ventana de la asignación.

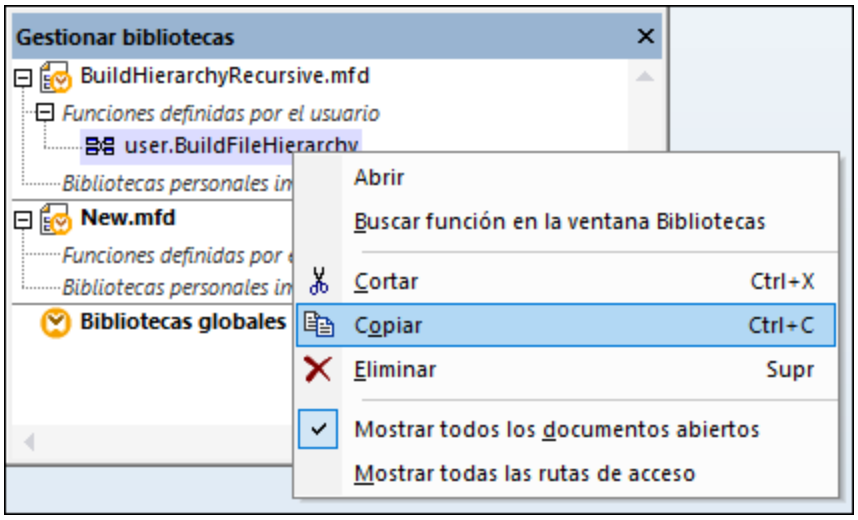
Además, se conserva un historial mientras navega por las pestañas de MapForce, incluidas las funciones definidas por el usuario. Para volver a pestañas visitadas con anterioridad, haga clic en los botones de la barra de herramientas y . Los atajos de teclado correspondientes para estas acciones son respectivamente **Alt+flecha izquierda** y **Alt+flecha derecha**.

Copiar y pegar funciones definidas por el usuario

Para copiar una función definida por el usuario en otra asignación siga estos pasos:

1. Abra la ventana Gestionar bibliotecas 553.
2. Haga clic con el botón derecho en un área vacía de la ventana **Bibliotecas** y seleccione la opción **Mostrar todos los documentos abiertos**.
3. Abra las asignaciones de origen y de destino. Asegúrese de que ambas están guardadas en disco. De lo contrario las rutas de acceso no se pueden resolver correctamente. Consulte también Rutas relativas y las acciones cortar y pegar 82.
4. En la asignación de origen de la ventana **Gestionar bibliotecas** haga clic con el botón derecho en la función definida por el usuario en cuestión y seleccione **Copiar** en el menú contextual (*imagen siguiente*) o pulse **Ctrl+C**. Deje abierta la ventana **Gestionar bibliotecas**.

© 2016-2022 Altova GmbH



5. Cambie la asignación de destino (la ventana **Gestionar bibliotecas** cambiará a su vez), haga clic con el botón derecho en *Funciones definidas por el usuario* y seleccione **Pegar** en el menú contextual.

Eliminar funciones definidas por el usuario

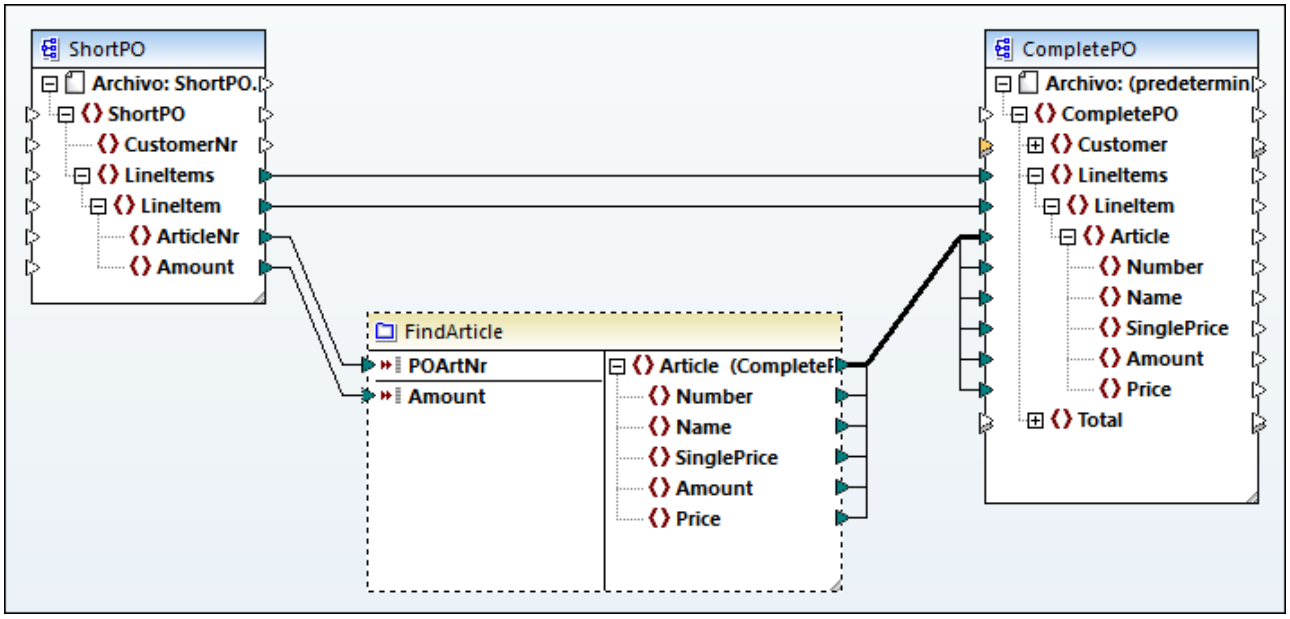
Para eliminar una función definida por el usuario siga estos pasos:

1. En la asignación haga doble clic en la barra del título de la función en cuestión.
2. Haga clic en el botón , en la parte superior derecha de la ventana Asignación.
3. Si la función se usa en la asignación abierta actualmente, MapForce le preguntará si quiere reemplazar todas las instancias con componentes internos. Haga clic en **Sí** si quiere eliminar la función y reemplazar todas las instancias en que se llama a la función con componentes de la función. De esta forma la asignación sigue siendo válida aunque se haya eliminado la función. Sin embargo, si usa la función eliminada en una asignación externa, no será válida. Haga clic en **No** si quiere borrar la función y todos sus componentes de forma permanente. En este caso todas las asignaciones en las que se use esa función dejarán de ser válidas.

6.4.2 Parámetros en funciones definidas por el usuario

Cuando se crear una función definida por el usuario se debe indicar qué parámetros de entrada debe tomar (si los toma) y qué resultados debe devolver. Aunque los parámetros de entrada no siempre son necesarios, es obligatorio en todos los casos incluir un parámetro de salida. Los parámetros de las funciones pueden ser de tipo simple (como una `cadena` o un `entero`) o una estructura de tipo complejo 582. Por ejemplo, la función definida por el usuario `FindArticle` que ve más abajo tiene dos parámetros de entrada y uno de salida:

- `POArtNr` es un parámetro de entrada de tipo `cadena`.
- `Amount` es un parámetro de entrada de tipo `entero`.
- `CompletePO` es un parámetro de salida con una estructura XML compleja.



Orden de los parámetros
Cuando una función definida por el usuario tiene varios parámetros de entrada o salida, puede cambiar el orden en que aparecen cuando se llama a esa función. El orden de los parámetros en la asignación de la función (empezando por arriba) dicta el orden en que aparecen cuando se llama a la función.

Importante

- Los parámetros de entrada y de salida se ordenan según su posición de arriba a abajo. Por tanto, si mueve el parámetro `input3` a la parte superior de la asignación de la función, se convertirá en el primer parámetro de esta función.
- Si dos parámetros tienen la misma posición vertical, entonces tiene precedencia el último.
- En el caso improbable de que varios parámetros tengan la misma posición, entonces se usa automáticamente un ID de componente interno.

Estructuras de tipo complejo

En la lista siguiente puede ver las estructuras en que se puede basar un parámetro de una función definida por el usuario.

MapForce Basic Edition
- Estructura de XML Schema

MapForce Professional Edition
- Estructura de XML Schema
- Estructura de BD

MapForce Enterprise Edition
- Estructura de XML Schema
- Estructura de BD
- Estructura EDI
- Estructura FlexText
- Estructura de JSON Schema

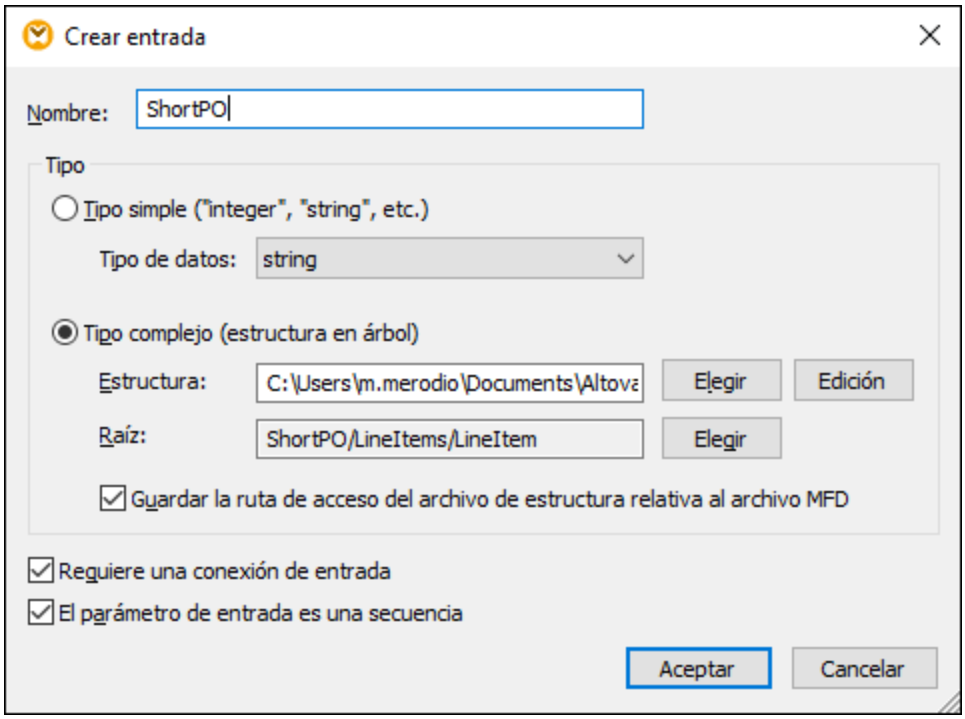
 © 2016-2022 Altova GmbH

Las funciones definidas por el usuario se basan en estructuras de base de datos (en las ediciones Professional y Enterprise)
MapForce permite crear parámetros en funciones definidas por el usuario basadas en bases de datos con una estructura de tablas relacionadas. La estructura de tablas relacionadas representa una estructura en memoria que no tiene conexión con la BD en tiempo de ejecución. Esto también significa que no hay una forma automática de administrar las claves foráneas ni acciones de tabla en los parámetros o las variables.

Agregar parámetros

Para agregar un parámetro de entrada o de salida siga estos pasos:

1. Cree una función definida por el usuario 576 o abra una que ya exista 579.
2. Ejecutar el comando de menú **Función | Insertar componente de entrada** o **Función | Insertar componente de salida** (*imagen siguiente*). También puede hacer clic en (**Insertar componente de entrada**) or (**Insertar componente de salida**) en la barra de herramientas.



3. Elija si los componentes de entrada o de salida deben ser de tipo simple o complejo (*véase el cuadro de diálogo anterior*). Consulte la lista de estructuras complejas disponibles más arriba. Por ejemplo, para crear un parámetro XML de tipo complejo haga clic en **Elegir**, junto a *Estructura*, y navegue hasta el esquema XML que describe la estructura requerida.

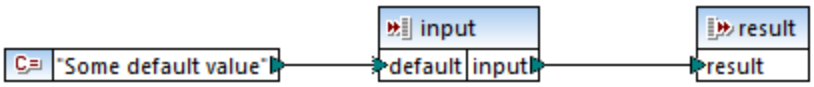
Si la asignación de la función ya incluye esquemas XML, estos pasan a estar disponibles para seleccionarlos como estructuras. De lo contrario puede seleccionar un esquema nuevo para que proporcione estructura al parámetro. Los mismo ocurre con bases de datos y otras estructuras complejas, siempre que sean compatibles con su versión de MapForce. Con las estructuras XML es posible seleccionar el elemento raíz pata la estructura si el esquema XML lo permite. Para indicar el elemento raíz haga clic en **Elegir**, junto a *Raíz*, y seleccione el elemento raíz en el cuadro de diálogo que se abre.

Si la casilla *Guardar ruta de acceso de la estructura como relativas al archivo MFD* está marcada, la ruta absoluta de la estructura pasará a ser relativa a la asignación actual cuando la guarde. Para más información

consulte el apartado relativa y ruta absoluta 82. Las casillas *Requiere una conexión de entrada* y *El parámetro de entrada es una secuencia* se explican en los siguientes apartados.

Requiere una conexión de entrada
Para hacer que un parámetro sea obligatorio en una función definida por el usuario, marque la casilla *Requiere una conexión de entrada* (*véase el cuadro de diálogo de más arriba*). Si desmarca esta casilla, el parámetro se convierte en opcional y aparece con un borde discontinuo en la asignación.

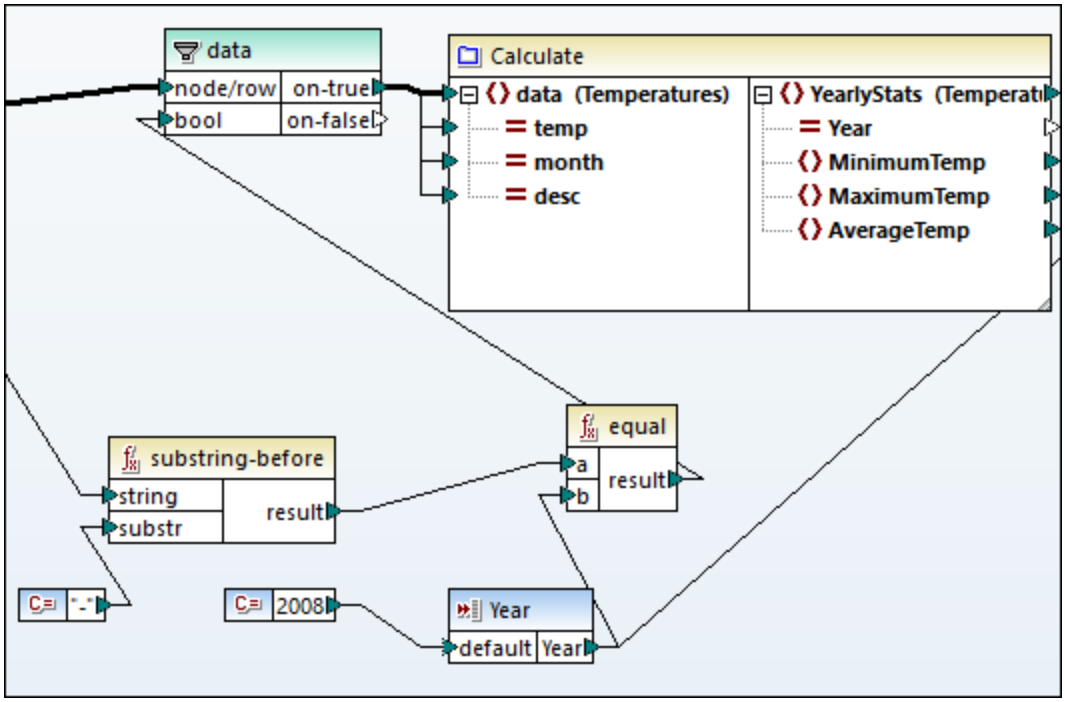
También puede indicar un valor de parámetro predeterminado si lo conecta a la entrada `default` de un parámetro (*ejemplo siguiente*). El valor predeterminado solo se aplica si no hay ningún otro valor. Si el parámetro opcional recibe un valor cuando se llama a la función, ese valor tiene prioridad sobre el valor predeterminado.



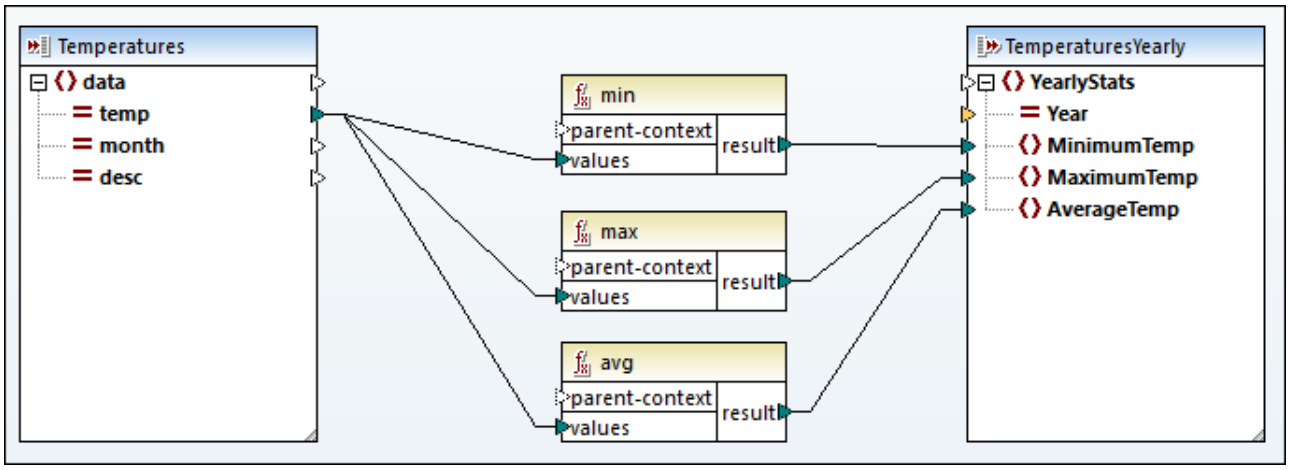
El componente de entrada es una secuencia
Puede decidir si un parámetro de una función se debe tratar como un valor único (opción predeterminada) o como una secuencia. Una secuencia es un rango de cero o más valores. Una secuencia puede ser útil si la función definida por el usuario espera que los datos de entrada sean una secuencia para calcular los valores de esa secuencia, por ejemplo, llamando a funciones como **`avg`**, **`min`**, **`max`**. Para tratar el componente de entrada del parámetro como una secuencia, marque la casilla *El parámetro de entrada es una secuencia*. Recuerde que esta casilla solo se habilita para funciones definidas por el usuario estándar 578.

En la asignación siguiente puede ver islustrado el uso de una secuencia: **`MapForceExamples\InputIsSequence.mfd`**. En el extracto de esta asignación de elementos (*imagen siguiente*) el filtros de `datos` está conectado a la función definida por el usuario `Calculate`. El resultado del filtro es una secuencia de elementos. Por tanto, el componente de entrada de la función se definió como secuencia.

 © 2016-2022 Altova GmbH



A continuación puede ver la implementación de la función `Calculate` que suma todos los valores de la secuencia: Las funciones **`avg`**, **`min`** y **`max`** se ejecutan en la secuencia de entrada. Para ver la estructura interna de la función `Calculate` hace doble clic en el encabezado del componente `Calculate` de la asignación anterior.



Por lo general, los datos de entrada (sean una secuencia o no) determinan cómo de a menudo se llama a la función:

- Cuando los datos de entrada se conectan al parámetro *sequence*, se llama a la función definida por el usuario *una sola vez* y la secuencia completa se pasa a la función definida por el usuario.
- Cuando los datos de entrada se conectan al parámetro *non-sequence*, se llama a la función definida por el usuario *una vez por cada elemento* de la secuencia.

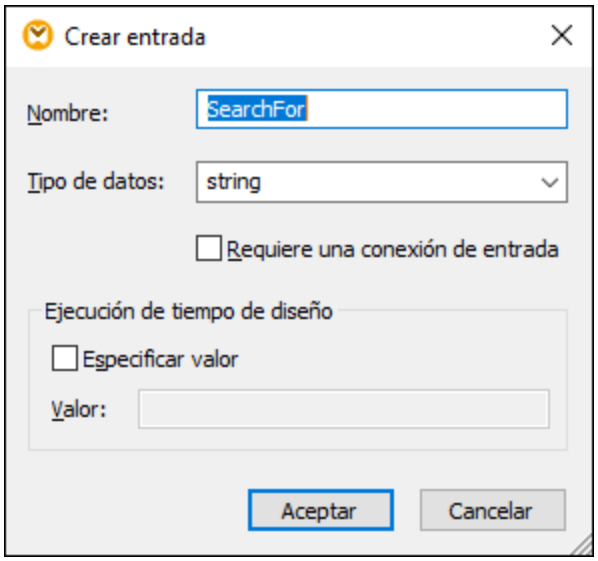
- Si conecta una secuencia vacía a un parámetro que no sea una secuencia, no se llama a la función. Esto puede ocurrir si la estructura de origen tiene elementos opcionales o si una de las condiciones de un filtro no devuelve coincidencias. Para evitarlo, use la función `substitute-missing` 707 antes de la entrada de la función para asegurarse de que la secuencia nunca esté vacía. También puede definir el parámetro como secuencia y agregar controles para la secuencia vacía dentro de la función.

Es posible que también necesite la casilla *El componente de salida es una secuencia* para los parámetros de salida. Cuando una función pasa una secuencia de varios valores a su componente de salida y este no se ha definido como secuencia, la función devuelve solamente el primer elemento de la secuencia.

6.4.3 Ejemplo: búsqueda recursiva

En esta sección explicamos cómo realizar búsquedas recursivas de datos en un archivo XML de origen con ayuda de las funciones definidas por el usuario. Para probar la búsqueda recursiva de la función definida por el usuario necesita esta asignación: **`MapForceExamples\RecursiveDirectoryFilter.mfd`**. En la asignación siguiente la función definida por el usuario `FilterDirectory` recibe datos del archivo de origen `Directory.xml` y del componente de entrada simple `SearchFor` que suministra la extensión `.xml`. Una vez la función ha procesado los datos, se asigna al archivo de destino.

La asignación principal (*imagen siguiente*) describe el diseño general de la asignación. La forma en que la función definida por el usuario procesa los datos se define aparte, en la asignación de la función (consulte más abajo *Implementación de de funciones definidas por el usuario*).



Objetivo
Nuestro objetivo es hacer una lista de los archivos con la extensión **`.xml`** en los resultados mientras conservamos la toda la estructura del directorio. A continuación explicamos en detalle el proceso de asignación.

 © 2016-2022 Altova GmbH

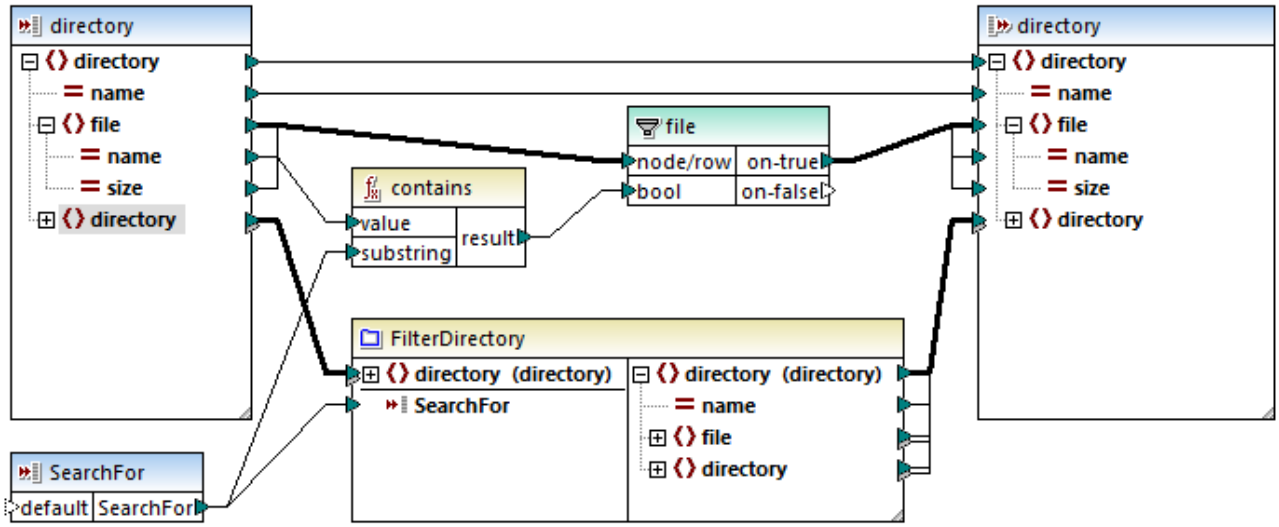
Archivo de origen

Más abajo puede ver un extracto del archivo XML de origen (**`Directory.xml`**) que contiene información sobre los archivos y directorios. Observe que los archivos de esta lista tienen extensiones distintas (e.g., **`.xml`**, **`.dtd`**, **`.sps`**). Según el esquema (**`Directory.xsd`**), el elemento `directory` puede tener descendientes `file` y `directory`. Todos los elementos `directory` son recursivos.

```
<directory name="ExampleSite">
       <file name="blocks.sps" size="7473"/>
       <file name="blocks.xml" size="670"/>
       <file name="block_file.xml" size="992"/>
       <file name="block_schema.xml" size="1170"/>
       <file name="contact.xml" size="453"/>
       <file name="dictionaries.xml" size="206"/>
       <file name="examplesite.dtd" size="230"/>
       <file name="examplesite.spp" size="1270"/>
       <file name="examplesite.sps" size="20968"/>
       ...
       <directory name="output">
              <file name="examplesite1.css" size="3174"/>
              <directory name="images">
                     <file name="blank.gif" size="88"/>
                     <file name="block_file.gif" size="13179"/>
                     <file name="block_schema.gif" size="9211"/>
                     <file name="nav_file.gif" size="60868"/>
                     <file name="nav_schema.gif" size="6002"/>
              </directory>
       </directory>
</directory>
```

Implementación de funciones definidas por el usuario

Para ver la implementación interna de la función definida por el usuario haga doble clic en su encabezado en la asignación principal. La función definida por el usuario es recursiva, es decir, se llama a sí misma. Al estar conectada al elemento recursivo `directory`, se llama a esta función tantas veces como elementos anidados `directory` haya en la instancia XML de origen. Para que admita llamadas recursivas, la función debe ser estándar [578].



La implementación de la función definida por el usuario consiste en dos partes: (i) definir los archivos y (ii) definir el directorio en que se realiza la búsqueda.

Definir archivos
Así es como la función definida por el usuario procesa los archivos: La función **`contains`** comprueba si la primera cadena (el nombre del archivo) contiene la subcadena `.xml` (dada por el componente de entrada simple `SearchFor`). Si la función devuelve `true`, se escribe en el resultado el nombre del archivo con la extensión `.xml`.

Procesar directorios descendientes
Los directorios descendientes del directorio actual se envían como entrada a la función definida por el usuario actual. Por tanto, esa función va comprobando si existen archivos con la extensión **`.xml`** en todos los elementos `directory`.

Resultados

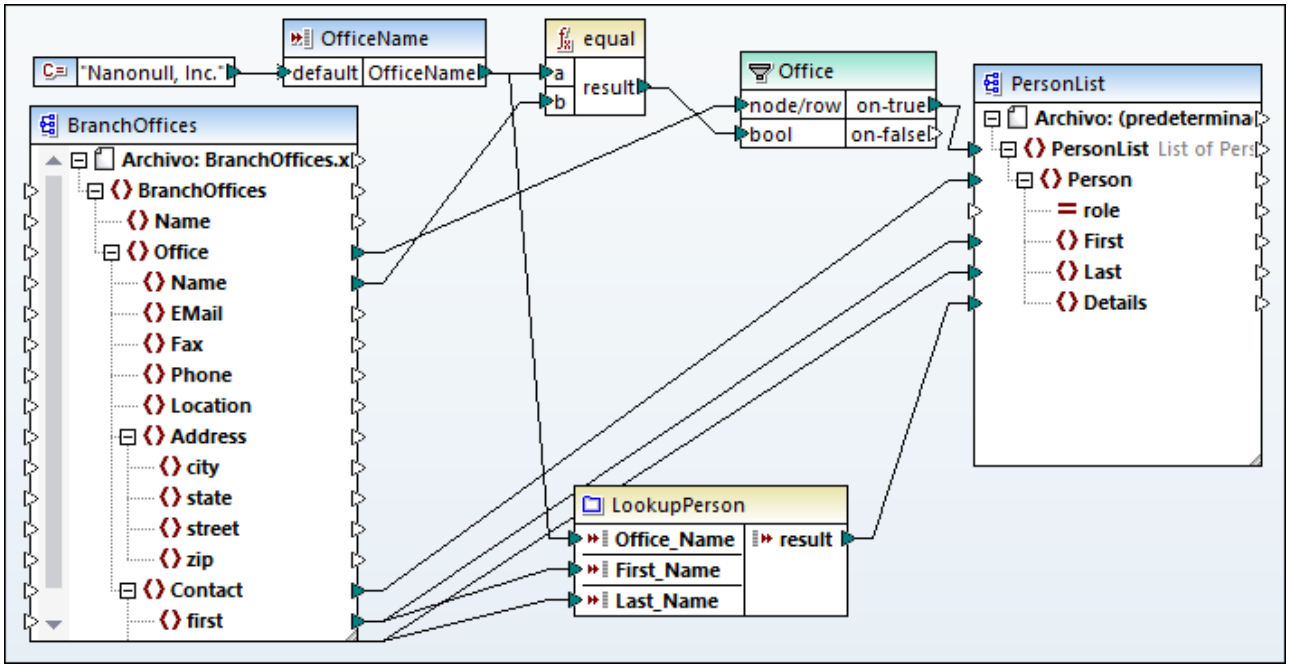
Al hacer clic en el panel **Resultados**, MapForce muestra solamente los archivos que tienen la extensión **`.xml`** (*véase el extracto más abajo*).

```
<directory name="ExampleSite">
        <file name="blocks.xml" size="670"/>
        <file name="block_file.xml" size="992"/>
        <file name="block_schema.xml" size="1170"/>
        <file name="contact.xml" size="453"/>
        ...
        <directory name="output">
                <directory name="images"/>
        </directory>
</directory>
```

 © 2016-2022 Altova GmbH

6.4.4 Implementación de la búsqueda

En este apartado explicamos cómo buscar información sobre los empleados y presentar esa información de forma adecuada. Para probar la implementación de la búsqueda necesitará esta asignación: **`MapForceExamples\PersonListByBranchOffice.mfd`**.



Objetivos
En este caso los objetivos son:

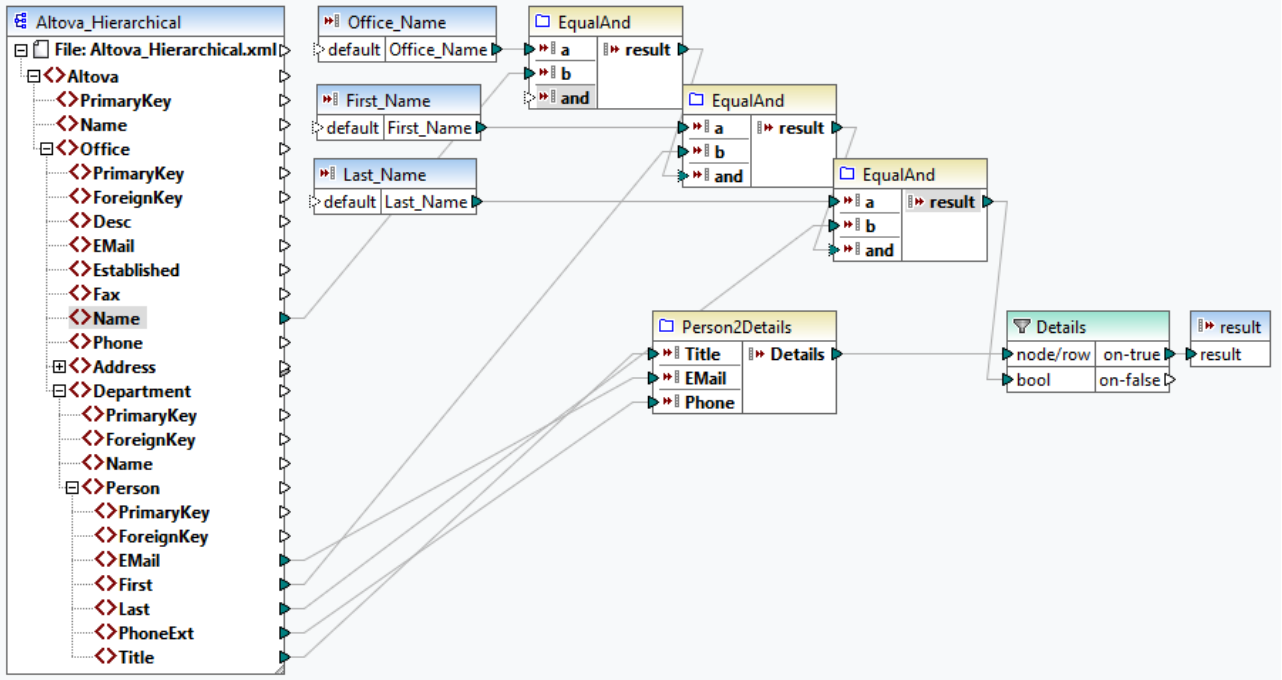
- Buscar información sobre cada uno de los empleados (su extensión de teléfono, dirección de correo electrónico y título) en un archivo XML aparte.
- Presentar estos datos como lista separada por comas y asignarla al elemento `Details` del archivo XML de destino.
- Extraer información sobre los empleados de una sola filial llamada Nanonull, Inc.

Para ello hemos diseñado la asignación de la siguiente forma:

- Para visualizar solamente los empleados de Nanonull, Inc. la asignación usa el filtro `Office`.
- Para buscar información sobre los empleados en un archivo XSLT diferente la asignación llama a la función definida por el usuario `LookupPerson`. La implementación de esta función definida por el usuario se describe más abajo.
- Para procesar la información de los empleados, la función `LookupPerson` llama internamente a otras funciones que obtienen y concatenan la información de los empleados. Todas estas operaciones están dentro de la asignación de la función y no son visibles en la asignación principal. Después, la función `LookupPerson` asigna los datos del empleado al elemento `Details` en `PersonList`.

Implementación de LookupPerson

La funcionalidad de búsqueda la proporciona la función `LookupPerson`, sobre la que encontrará más información más abajo. Para ver la implementación interna de la función definida por el usuario haga doble clic en su encabezado en la asignación principal.



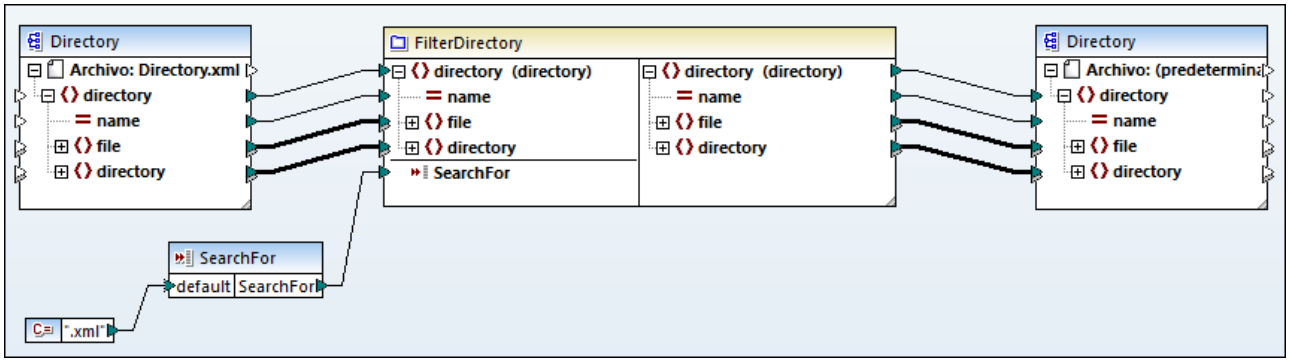
Así se configuró la unción definida por el usuario:

- Los datos se obtienen del archivo XML **`Altova_Hierarchical.xml`**: (i) el nombre de la oficina y el nombre y apellido de los empleados, que se usan para seleccionar solamente empleados de Nanonull, Inc., y (ii) el correo electrónico, el título y la extensión de teléfono, que se concatenan en una sola cadena. A continuación describimos las definiciones de las funciones `EqualAnd` y `Person2Detail`.
- La función definida por el usuario también tiene tres parámetros de entrada que proporcionan los valores de búsqueda `Office_Name`, `First_Name` y `Last_Name`. El valor del parámetro `Office_Name` se obtiene de la entrada `OfficeName` de la asignación principal, y los valores de `First_Name` y `Last_Name` los da el componente `BranchOffices` de la asignación principal.
- El valor de la función `EqaulAnd` (`true` o `false`) se pasa al filtro `Details` cada vez que se procesa la información (título, correo electrónico, teléfono) de un empleado. Cuando el filtro `Details` obtiene el valor `true` se entiende que la búsqueda se ha realizado correctamente y que se pueden pasar los datos obtenidos a la asignación principal. De lo contrario se examina el elemento siguiente en el contexto hasta que termine el bucle.

Implementación de EqualAnd

La función `EqualAnd` (*véase más abajo*) es una función definida por el usuario aparte que está definida dentro de la función definida por el usuario `LookupPerson`. Para ver la estructura interna de `EqualAnd` haga doble clic en su encabezado.

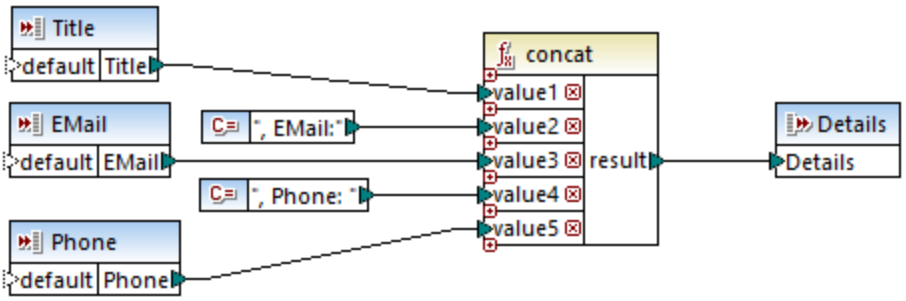
 © 2016-2022 Altova GmbH



`EqualAnd` primero comprueba si `a` es igual a `b`; si el resultado es `true`, este se pasa al primer parámetro de la función **`logical-and`**. Si los dos valores son true en la función **`logical-and`**, el resultado también es true y se pasa a la siguiente función `EqualAnd`. El resultado de la tercera función `EqualAnd` (*véase LookupPerson más arriba*) se pasa al filtro `Details`.

Implementación de Person2Detail

La función definida por el usuario `Person2Details` es otra función que se encuentra dentro de `LookupPerson`. Esta función (*véase más abajo*) concatena tres valores (obtenidos de **`Altova_Hierarchical.xml`**) y dos constantes de texto.



Resultados

Al hacer clic en el panel **Resultados**, MapForce muestra la información recuperada de los empleados de Nanonull Inc (*véase a continuación*).

```
<PersonList>
	<Person>
		<First>Vernon</First>
		<Last>Callaby</Last>
		<Details>Office Manager, EMail:v.callaby@nanonull.com, Phone:
582</Details>
	</Person>
	<Person>
		<First>Frank</First>
		<Last>Further</Last>
		<Details>Accounts Receivable, EMail:f.further@nanonull.com, Phone:
471</Details>
	</Person>
```

```
        ...
</PersonList>
```


© 2016-2022 Altova GmbH

6.5 Funciones personales

En esta sección explicamos cómo importar funciones personales XSLT[593], XQuery[600], Java y .NET[604]. También aprenderá a hacer referencia a bibliotecas C#, C++ y Java manualmente[612].

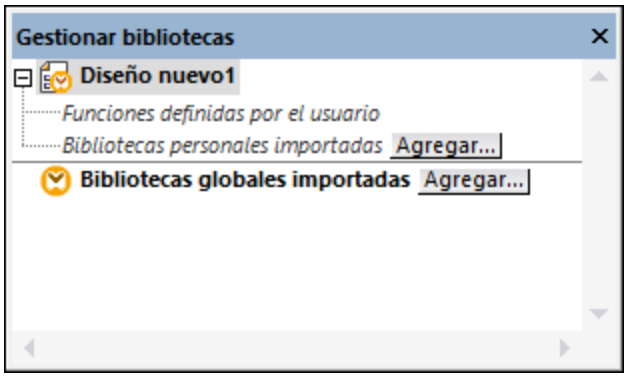
6.5.1 Importar funciones XSLT 1.0/2.0 personales

Puede ampliar las bibliotecas de funciones XSLT 1.0, 2.0 y 3.0 de MapForce con sus propias funciones personales, siempre y cuando estas devuelvan tipos simples.

Solamente se admiten funciones personales que devuelvan tipos simples (p. ej. cadenas).

Para importar funciones de un archivo XSLT:

1. Haga clic en el botón **Agregar o quitar bibliotecas**, en la parte inferior de la ventana Bibliotecas[30]. Se abre la ventana **Gestionar bibliotecas**.



2. Para importar funciones como biblioteca local (dentro del archivo de asignación actual solamente), haga clic en **Agregar** bajo el nombre de la asignación activa. Para importar funciones como biblioteca global (a nivel de programa), haga clic en **Agregar**, junto a **Bibliotecas globales importadas**. Cuando se importa una biblioteca de forma local, puede hacer que la ruta de acceso al archivo de la biblioteca sea relativa al archivo de asignación. Con las bibliotecas globales la ruta siempre es absoluta.

3. Navegue hasta el archivo .xsl que contiene las funciones y haga clic en **Abrir**. Aparece un cuadro de mensaje que le informa de que se ha añadido una biblioteca nueva.

Los archivos XSLT importados aparecen como bibliotecas en la ventana Bibliotecas y muestran todas las plantillas con nombres como si fueran funciones. Si no puede ver la biblioteca importada, compruebe que el lenguaje de transformación[24] seleccionado es XSLT (*véase también* Gestionar bibliotecas de funciones[553]).

Importante:

- Para poder importarlas en MapForce las funciones deben declararse en el archivo XSLT como plantillas con nombre según lo estipulado por la especificación XSLT. También puede importar funciones que aparezcan en un documento XSLT 2.0 con el formato `<xsl:function name="MiFunción">`. Si el archivo XSLT importado importa o incluye otros archivos XSLT, también se importarán dichos archivos y sus funciones.
- Los conectores de entrada asignables de las funciones importadas personales dependen del número de parámetros utilizado en la llamada a plantilla. También se admiten parámetros opcionales.
- No se admiten espacios de nombres.
- Si realiza cambios en archivos XSLT que ya están importados en MapForce, los cambios se detectan automáticamente y puede elegir si los archivos se vuelven a cargar.
- Cuando escriba plantillas con nombre, asegúrese de que las instrucciones XPath utilizadas en la plantilla están enlazadas al espacio de nombres correcto. Para ver los enlaces de espacio de nombres de la asignación genere una vista previa del código XSLT de salida 107.

Tipos de datos en XPath 2.0

Si el documento XML hace referencia a un esquema XML y es válido según este esquema, deberá convertir o construir explícitamente tipos de datos que no se conviertan implícitamente al tipo de datos necesario por medio de una operación.

En el modelo de datos de XPath 2.0 que utiliza el motor XSLT 2.0 de Altova todos los valores de nodos **atomizados** del documento XML tienen asignado el tipo de datos `xs:untypedAtomic`. El tipo `xs:untypedAtomic` es adecuado para las conversiones de tipo implícitas.

Por ejemplo:

- la expresión `xs:untypedAtomic("1") + 1` da como resultado un valor de 2 porque el valor `xdt:untypedAtomic` se convierte **implícitamente** en `xs:double` por medio del operador de suma.
- los operadores aritméticos convierten los operandos implícitamente en `xs:double`.
- los operadores de comparación de valores convierten los operandos en `xs:string` antes de compararlos.

Temas relacionados:

Ejemplo: agregar funciones XSLT 1.0 personales 594
Ejemplo: sumar valores de nodos 597
Implementación del motor XSLT 1.0 1485
Implementación del motor XSLT 2.0 1485

6.5.1.1 Ejemplo: agregar funciones XSLT personales

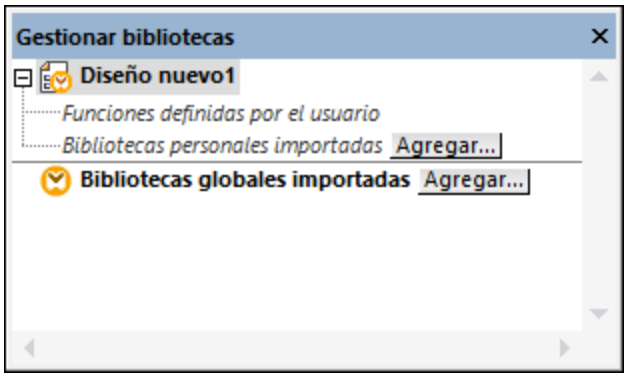
Este ejemplo explica cómo importar funciones XSLT 1.0 personales en MapForce. Los archivos necesarios para seguir este ejemplo están en la carpeta **`C:\Usuarios\<usuario>\Documentos\Altova\MapForce2023\MapForceExamples`**.

- **Name-splitter.xslt:** este archivo XSLT define una plantilla con nombre llamada "**tokenize**" que tiene un solo parámetro llamado "string". La plantilla recorre una cadena de entrada y separa todas las letras mayúsculas de la cadena con un espacio.
- **Name-splitter.xml:** el archivo de instancia XML de origen que se debe procesar.
- **Customers.xsd:** el esquema XML de origen.
- **CompletePO.xsd:** el esquema XML de destino.

 © 2016-2022 Altova GmbH

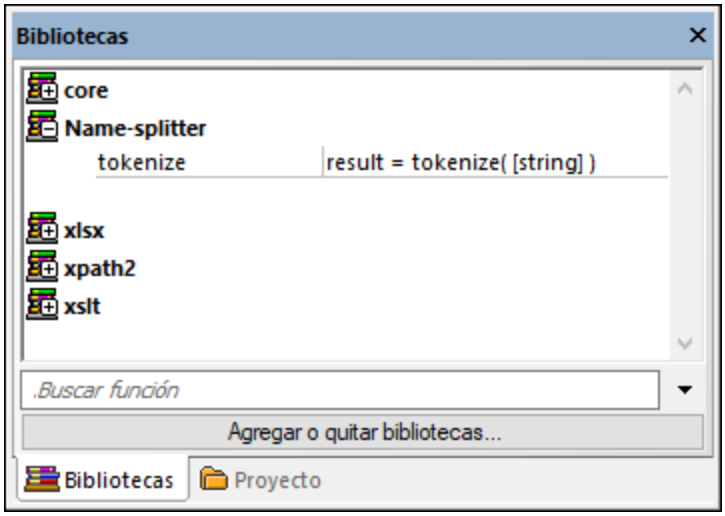
Para agregar una función XSLT personal:

1. Haga clic en el botón **Agregar o quitar bibliotecas**, en la parte inferior de la ventana Bibliotecas 30. Se abre la ventana **Gestionar bibliotecas**.



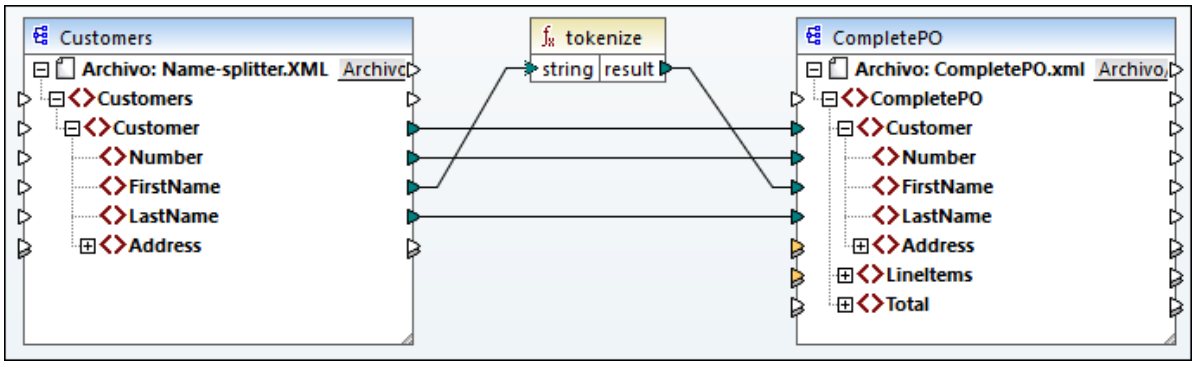
2. Para importar funciones como biblioteca local (dentro del archivo de asignación actual solamente), haga clic en **Agregar** bajo el nombre de la asignación activa. Para importar funciones como biblioteca global (a nivel de programa), haga clic en **Agregar**, junto a **Bibliotecas globales importadas**. Cuando se importa una biblioteca de forma local, puede hacer que la ruta de acceso al archivo de la biblioteca sea relativa al archivo de asignación. Con las bibliotecas globales la ruta siempre es absoluta.

3. Ahora haga clic en el botón **Agregar** y navegue hasta el archivo XSL o XSLT que contiene la plantilla con nombre que desea usar como función (en este caso **Name-splitter.xslt**). Haga clic en **Aceptar**. El nombre del archivo XSLT aparece en la ventana Bibliotecas junto con las funciones definidas como plantillas con nombre (en este ejemplo es **Name-splitter** con la función `tokenize`).



Para usar la función XSLT en la asignación de datos:

1. Arrastre la función **tokenize** hasta el área de asignación y cree las asignaciones de datos que aparecen en esta imagen:



2. Haga clic en el panel *XSLT* para ver el código XSLT de salida.

```
1   <?xml version="1.0" encoding="UTF-8"?>
2   <!-- ... -->
11  <xsl:stylesheet version="2.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform" xmlns:xs="http
12      <xsl:include href="file:///C:/Users/altova/Documents/Altova/MapForce2020/MapForceExamples
13      <xsl:output method="xml" encoding="UTF-8" byte-order-mark="no" indent="yes"/>
14      <xsl:template match="/">
15          <CompletePO>
16              <xsl:attribute name="xsi:noNamespaceSchemaLocation" namespace="http://www.w3.org/
    CompletePO.xsd'"/>
17              <xsl:for-each select="Customers/Customer">
18                  <Customer>
19                      <Number>
20                          <xsl:sequence select="xs:string(xs:integer(fn:string(Number)))"/>
21                      </Number>
22                      <FirstName>
23                          <xsl:call-template name="tokenize">
24                              <xsl:with-param name="string" select="FirstName" as="item()"/>
25                          </xsl:call-template>
26                      </FirstName>
27                      <LastName>
28                          <xsl:sequence select="fn:string(LastName)"/>
29                      </LastName>
30                  </Customer>
31              </xsl:for-each>
32          </CompletePO>
33      </xsl:template>
34  </xsl:stylesheet>
35
```

Nota: en cuanto una plantilla con nombre se utiliza en una asignación de datos, el archivo XSLT que contiene la plantilla con nombre **se incluye** en el código XSLT de salida (**xsl:include href**...) y **se le llama** con el comando **xsl:call-template**.

3. Haga clic en el panel *Resultados* para ver el resultado de la asignación de datos.

 © 2016-2022 Altova GmbH

Para quitar bibliotecas XSLT personales de MapForce:

1. Haga clic en el botón **Agregar o quitar bibliotecas** situado en la parte inferior de la ventana Bibliotecas.
2. Haga clic en la biblioteca XSLT que desea eliminar y después haga clic en **Eliminar biblioteca** ✕.

6.5.1.2 Ejemplo: sumar valores de nodos

Este ejemplo explica cómo procesar varios nodos de un documento XML y asignar el resultado en forma de un solo valor a un documento XML de destino. Concretamente el objetivo de esta asignación de datos es calcular el precio de todos los productos de un archivo XML de origen y escribirlo como un solo valor en un archivo XML de salida. Los archivos utilizados en este ejemplo están en la carpeta **<Documentos>\Altova\MapForce2023\MapForceExamples\Tutorial**:

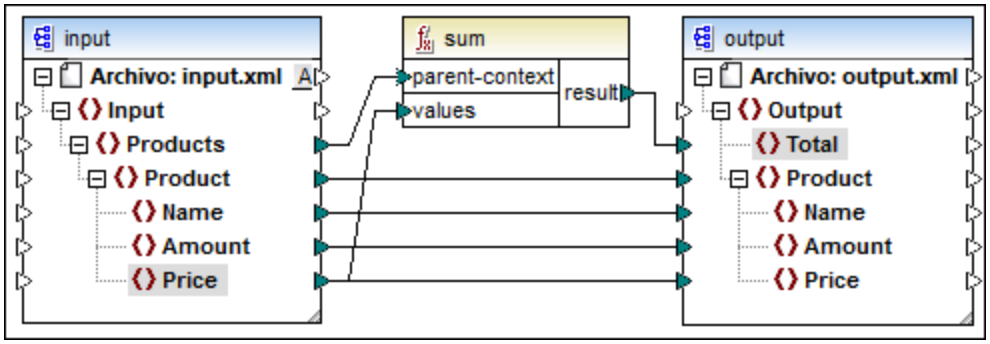
- **Summing-nodes.mfd:** el archivo de asignación de datos
- **input.xml:** el archivo XML de origen
- **input.xsd:** el esquema XML de origen
- **output.xsd:** el esquema XML de destino
- **Summing-nodes.xslt:** una hoja de estilos XSLT personal que contiene una plantilla con nombre que sumará los nodos.

Hay dos maneras de conseguir nuestro objetivo:

- con ayuda de la función de agregado **sum** 639 de la biblioteca **core**.
- importando una hoja de estilos XSLT personal en MapForce.

Solución nº1: usar la función de agregado "sum"

Para usar la función de agregado **sum** a la asignación basta con arrastrarla desde la ventana Bibliotecas hasta el panel de asignación. Recuerde que las funciones que aparecen en la ventana Bibliotecas dependen del lenguaje de transformación que esté seleccionado. Tras colocar la función en el área de asignación deberá crear las conexiones de asignación que aparecen en esta imagen:



Para más información sobre las funciones de agregado de la biblioteca **core** consulte el apartado core | aggregate functions (agregado) 633.

Solución nº2: usar una hoja de estilos XSLT personal

El objetivo del ejemplo es sumar los campos `Price` de los productos del archivo XML de origen (en este caso los productos A y B).

```
<?xml version="1.0" encoding="UTF-8"?>
<Input xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="input.xsd">
   <Products>
      <Product>
         <Name>ProductA</Name>
         <Amount>10</Amount>
         <Price>5</Price>
      </Product>
      <Product>
         <Name>ProductB</Name>
         <Amount>5</Amount>
         <Price>20</Price>
      </Product>
   </Products>
</Input>
```

A continuación puede ver una hoja de estilos XSLT personal que usa la plantilla con nombre "Total" y un solo parámetro `string`. La plantilla recorre el archivo XML de entrada y suma todos los valores que obtiene la expresión `/Product/Price`.

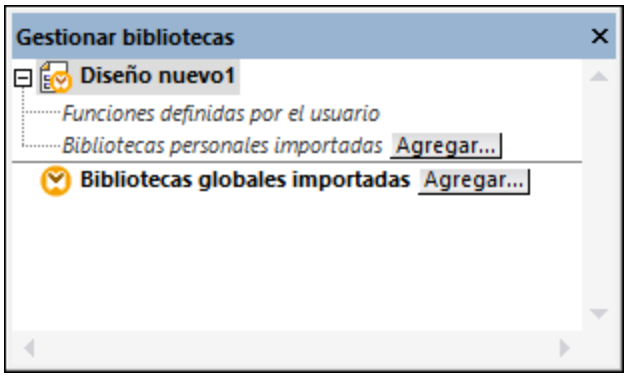
```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
   <xsl:output method="xml" version="1.0" encoding="UTF-8" indent="yes"/>

   <xsl:template match="*">
      <xsl:for-each select=".">
      <xsl:call-template name="Total">
         <xsl:with-param name="string" select="."/>
      </xsl:call-template>
      </xsl:for-each>
   </xsl:template>

   <xsl:template name="Total">
   <xsl:param name="string"/>
      <xsl:value-of select="sum($string/Product/Price)"/>
   </xsl:template>
</xsl:stylesheet>
```

Nota: para sumar los nodos en XSLT 2.0 debe cambiar la declaración de hoja de estilos por `version="2.0"`.

Para importar la hoja de estilos XSLT en MapForce:

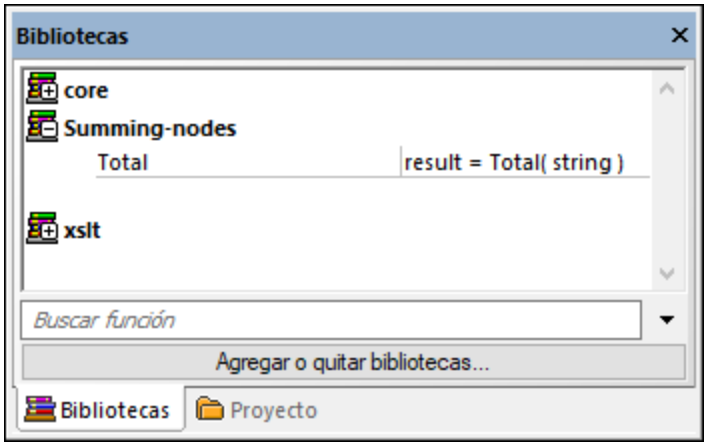
1. Haga clic en el botón **Agregar o quitar bibliotecas**, en la parte inferior de la ventana Bibliotecas 30. Se abre la ventana **Gestionar bibliotecas**.

 © 2016-2022 Altova GmbH

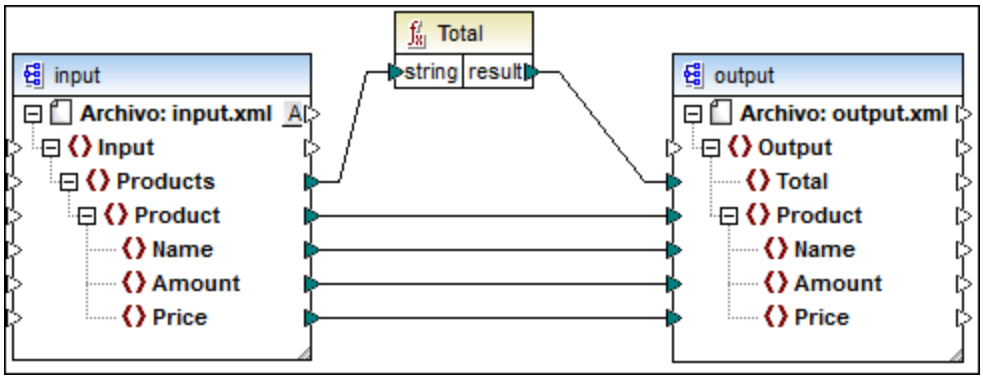


2. Para importar funciones como biblioteca local (dentro del archivo de asignación actual solamente), haga clic en **Agregar** bajo el nombre de la asignación activa. Para importar funciones como biblioteca global (a nivel de programa), haga clic en **Agregar**, junto a **Bibliotecas globales importadas**. Cuando se importa una biblioteca de forma local, puede hacer que la ruta de acceso al archivo de la biblioteca sea relativa al archivo de asignación. Con las bibliotecas globales la ruta siempre es absoluta.

3. En el cuadro de diálogo "Opciones" haga clic en el botón **Agregar**. Navegue hasta el archivo **<Documentos>\Altova\MapForce2023\MapForceExamples\Tutorial\Summing-nodes.xslt**.



4. Arrastre la función `Total` de la biblioteca recién añadida (**Summing-nodes**) hasta el área de asignación y cree las conexiones que aparecen en esta imagen:



Haga clic en el panel *Resultados* para obtener una vista previa del resultado de la asignación. La suma de los dos campos `Price` aparecen ahora en el campo `Total`.

```
<?xml version="1.0" encoding="UTF-8"?>
<Output xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="output.xsd">
   <Total>25</Total>
   <Product>
      <Name>ProductA</Name>
      <Amount>10</Amount>
      <Price>5</Price>
   </Product>
   <Product>
      <Name>ProductB</Name>
      <Amount>5</Amount>
      <Price>20</Price>
   </Product>
</Output>
```

6.5.2 Importar funciones XQuery 1.0 personales

Cuando el lenguaje de transformación seleccionado es XQuery, MapForce muestra las bibliotecas integradas de funciones XQuery en la ventana Bibliotecas. Si quiere, puede ampliar esta biblioteca con sus propias funciones XQuery personales con solo importar sus módulos de biblioteca XQuery 1.0 personales en MapForce.

Para poder importarse en MapForce, el archivo XQuery debe cumplir estos requisitos:

- Debe ser un módulo de biblioteca válido según lo estipulado por la especificación XQuery. Es decir, debe empezar con una declaración de módulo como `module namespace <prefix>="<namespace name"`
- Todas las funciones declaradas en el módulo de biblioteca importado deben devolver tipos de datos atómicos (p. ej. `xs:string`, `xs:boolean`, `xs:integer`, etc.). Los parámetros de las funciones también deben tener tipos atómicos.

Para importar un módulo de biblioteca XQuery:

1. En el menú **Herramientas** haga clic en **Opciones** (o haga clic en el botón **Agregar o quitar bibliotecas** situado en la parte inferior de la ventana Bibliotecas).
2. Haga clic en el botón **Agregar** y navegue hasta el archivo de biblioteca **.xq** o **.xquery**.

Si el módulo de biblioteca que desea importar no es compatible, MapForce emitirá un aviso al respecto. Si el módulo es compatible, aparecerá en la ventana Bibliotecas.

Si no puede ver el módulo de biblioteca XQuery importado, asegúrese de que el lenguaje de transformación que está seleccionado es XQuery (*véase* *Seleccionar un lenguaje de transformación* 24).

 © 2016-2022 Altova GmbH

Temas relacionados:

Implementación del motor XQuery 1487

6.5.2.1 Ejemplo: importar funciones XQuery personales

Este ejemplo explica cómo importar un módulo de biblioteca XQuery de ejemplo en MapForce y llamar a sus funciones desde una asignación. Para este ejemplo importaremos un módulo que consiste en una única función que calcula un impuesto del 20% con importes decimales. En un escenario de producción, un módulo XQuery podría contener varias funciones.

> Todas las funciones declaradas en el módulo XQuery deben devolver tipos atómicos y sus parámetros también deben ser tipos de datos atómicos. De lo contrario el módulo no se puede importar a MapForce.

Puede encontrar el archivo del módulo XQuery de ejemplo en la siguiente ruta relativa a la carpeta personal "Documentos" del equipo en el que está instalado MapForce: **<Documentos>\Altova\MapForce2023\MapForceExamples\Tutorial\module.xq**.

```
xquery version "1.0";

module namespace demo="http://www.altova.com/mapforce/demo";

declare function demo:calculatetax($val as xs:decimal) as xs:decimal {
   $val*0.2
};
```

module.xq

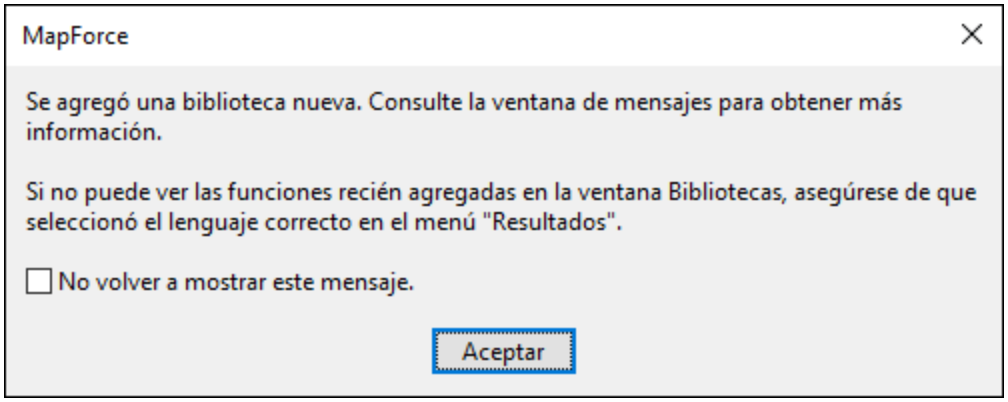
Después de importar el archivo del módulo XQuery en MapForce ya puede llamar a la función `demo:calculatetax` desde una asignación. Observe que estamos calculando el importe con la ayuda de una función XQuery para este ejemplo, pero se puede obtener el mismo resultado usando las funciones integradas de MapForce.

Existe una asignación que llama a la función `demo:calculatetax` en la siguiente ruta: **<Documentos>\Altova\MapForce2023\MapForceExamples\Tutorial\CalculateTax_XQuery.mfd**. La primera vez que abra esta asignación MapForce emitirá un aviso indicando que la asignación contiene uno o más componentes que no están disponibles en XQuery. Esta advertencia es normal y ocurre porque la asignación contiene referencias a funciones de una biblioteca XQuery personal que aún no se ha importado. Para eliminar la advertencia y ejecutar la asignación debe importar el módulo XQuery que falta en MapForce como se explica a continuación.

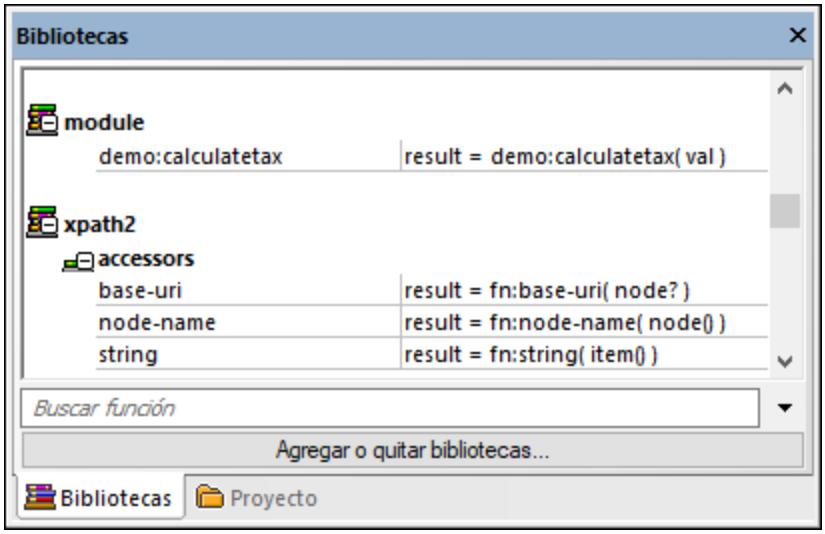
Importar el módulo XQuery en MapForce:

1. Vaya hasta el final de la ventana Bibliotecas y haga clic en **Agregar o quitar bibliotecas**. Se abre el cuadro de diálogo "Opciones".

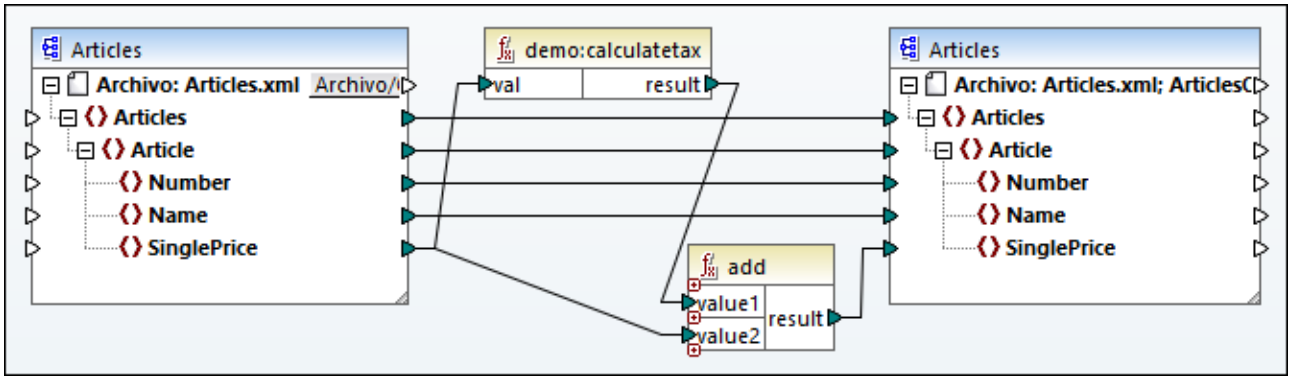
2. Haga clic en **Add** (*Agregar*) y navegue hasta el archivo **<Documentos>\Altova\MapForce2023\MapForceExamples\Tutorial\module.xq**.
3. Haga clic en **Aceptar** para cerrar el cuadro de diálogo. Aparece un cuadro de mensaje que confirma que se ha agregado una biblioteca nueva.



Ahora la biblioteca que ha agregado y la función **`demo:calculatetax`** son visibles en la ventana *Bibliotecas*.



Ahora puede ejecutar la asignación sin que aparezcan advertencias. La función **`demo:calculatetax`** de la imagen siguiente proviene del módulo XQuery que ha importado y se puede añadir a la asignación como cualquier otra función (*véase* *Agregar una función* 550).



CalculateTax_XQuery.mfd

La asignación explicada

La asignación de la imagen anterior, **CalculateTax_XQuery.mfd**, toma como entrada un archivo XML que almacena artículos. Cada artículo tiene un único precio que se expresa con un valor decimal, por ejemplo:

```
<Articles>
   <Article>
      <Number>1</Number>
      <Name>T-Shirt</Name>
      <SinglePrice>25</SinglePrice>
   </Article>
   <Article>
      <Number>2</Number>
      <Name>Socks</Name>
      <SinglePrice>2.30</SinglePrice>
   </Article>
   <Article>
      <Number>3</Number>
      <Name>Pants</Name>
      <SinglePrice>34</SinglePrice>
   </Article>
   <Article>
      <Number>4</Number>
      <Name>Jacket</Name>
      <SinglePrice>57.50</SinglePrice>
   </Article>
</Articles>
```

Articles.xml

La asignación produce un archivo XML que obedece al mismo esquema que el archivo XML de origen. En consecuencia, tanto el componente de entrada como el de salida tienen la misma estructura en la asignación. Tal y como sugieren las conexiones de la asignación, prácticamente todos los elementos están asignados de forma directa del componente de origen al de salida; por ejemplo, por cada elemento `Article` en el componente de origen habrá otro elemento `Article` en el componente de destino. Los valores de todos los

elementos se copian literalmente del XML de origen, excepto **SinglePrice**. El valor de **SinglePrice** se calcula con ayuda de dos funciones:

- La función de ejemplo XQuery **`demo:calculatetax`** calcula el importe del impuesto tomando como entrada el precio original, **SinglePrice**.
- La función integrada de MapForce `add` añade el importe del impuesto al importe del precio original **SinglePrice** y devuelve el importe final.

Conviene tener en cuenta que el tipo de datos del elemento SinglePrice es `xs:decimal`, lo que corresponde al tipo de parámetro de entrada y al tipo de retorno de la función XQuery.

A continuación se muestra el resultado producido por la asignación al hacer clic en la pestaña *Resultados*. Observe que los precios son un 20% más caros que los del XML de origen.

```
1   <?xml version="1.0" encoding="UTF-8"?>
2   <Articles xmlns:xsi="http://www.w3.org/2001/XMLSchema-
3       <Article>
4           <Number>1</Number>
5           <Name>T-Shirt</Name>
6           <SinglePrice>30</SinglePrice>
7       </Article>
8       <Article>
9           <Number>2</Number>
10          <Name>Socks</Name>
11          <SinglePrice>2.76</SinglePrice>
12      </Article>
13      <Article>
14          <Number>3</Number>
15          <Name>Pants</Name>
16          <SinglePrice>40.8</SinglePrice>
17      </Article>
18      <Article>
19          <Number>4</Number>
20          <Name>Jacket</Name>
21          <SinglePrice>69</SinglePrice>
22      </Article>
23  </Articles>
```

6.5.3 Importar bibliotecas Java y .NET personales

En MapForce puede importar archivos de clases Java compilados y archivos de ensamblados .NET (incluidos los ensamblados .NET 4.0). Si las bibliotecas importadas contienen funciones que usan tipos de datos básicos como parámetros y que devuelven tipos simples, dichas funciones aparecerán en la ventana Biblioteca y se podrán usar en los diseños de asignación de datos como cualquier otra función de MapForce. El resultado de las asignaciones de datos que incluyen funciones Java y .NET importadas se puede ver en el panel *Resultados* y las funciones estarán disponibles en el código generado. Para prender más sobre cómo importar bibliotecas personales consulte los ejemplos Ejemplo: importar clase Java personales 608 e Ejemplo: importar ensamblado .NET DLL personal 610.

Tenga en cuenta que:

- para importar funciones Java o .NET personales necesitará los archivos de clases Java compilados (.class) o los archivos de ensamblados .NET. No se pueden importar archivos Java `.jar` ni archivos `.dll` que no sean ensamblados .NET.

- los archivos de ensamblados .NET se pueden importar si el lenguaje de transformación seleccionado es C#. Los ensamblados .NET también pueden estar escritos en lenguajes .NET como C++.NET o VB.NET, siempre y cuando solamente usen como parámetros tipos de datos básicos de System Assembly y tipos devueltos (*véase* *Funciones .NET compatibles* 606).

- si quiere usar funciones .NET personales en la vista previa del resultado (en el panel **Resultados**), esas funciones deben estar compiladas para .NET Framework 4.x o .NET Standard 2.0.

- los archivos de clases Java compilados (`.class`) se pueden importar si el lenguaje de transformación seleccionado es Java. Es necesario tener instalado Java Runtime Environment 7 o superior. Solamente son compatibles ciertos tipos y miembros (*véase* *Funciones Java compatibles* 605).

- si la asignación usa archivos de clases Java o ensamblados .NET importados, no podrá seleccionar el lenguaje de transformación C++.

- si la asignación usa archivos de clases Java o ensamblados .NET importados, no podrá seleccionar el lenguaje de transformación XSLT (deberá escribir una función XSLT personal que haga de adaptador).

- la importación de funciones de ensamblados C++ nativos tiene algunas limitaciones y requiere un procedimiento especial (*véase* *Referencias manuales a bibliotecas Java, C# y C++ personales* 612).

- todas las funciones a las que llame una asignación de MapForce deben devolver el mismo valor cada vez que se llame a la función con los mismos parámetros de entrada. El orden exacto y el número de veces que MapForce llama a una función es indefinido.

- en el caso de los archivos de clases Java importados y de sus paquetes, no es necesario que se añadan a la variable CLASSPATH porque el motor de ejecución integrado y el código Java de salida agregarán automáticamente los paquetes importados a la variable classpath del motor Java o a Ant. Sin embargo, las dependencias de los archivos de clases y paquetes importados no se resolverán automáticamente. Por tanto, si los archivos de clases o paquetes Java dependen de otros archivos de clases, deberá agregar los directorios primarios de todos los paquetes dependientes a la variable de entorno CLASSPATH.

Funciones Java compatibles

Clases de nivel superior, clases de miembro estáticas y clases de miembro no estáticas:

- `new <nombreClase>(<arg1>, <arg2>, ...)`
- `<objeto>.new <clase-miembro>(<arg1>, <arg2>, ...)`

Funciones de miembro y estáticas:

- `<función>(<arg1>, <arg2>, ...)`
- `<objeto>.<método>(<arg1>, ...)`

Conexiones compatibles entre tipos XML Schema y tipos Java:

Tipo XML Schema	Tipo Java
xs:string	String
xs:byte	byte
xs:short	short
xs:int	int
xs:long	long
xs:boolean	boolean
xs:float	float
xs:double	double
xs:decimal	java.math.BigDecimal
xs:integer	java.math.BigInteger

Las conexiones pueden hacerse en ambas direcciones. Los demás tipos Java (incluidos los tipos matriz) no son compatibles. Los métodos que usen dichos parámetros o valores devueltos se omitirán. Se pueden usar tipos de objeto llamando a su constructor o como valor devuelto de un método. Se pueden asignar a otros métodos Java, pero no se puede manipular el objeto en MapForce.

Funciones .NET compatibles

Clases de nivel superior y clases de miembro:

- `new <nombreClase>(<arg1>, <arg2>, ...)`

Funciones de miembro y funciones estáticas:

- `<función>(<arg1>, <arg2>, ...)`
- `<objeto>.<método>(<arg1>, ...)`

Conexiones compatibles entre tipos XML Schema y tipos .NET/C#:

Tipos XML Schema	Tipos .NET	Tipos C#
xs:string	System.String	string
xs:byte	System.SByte	sbyte
xs:short	System.Int16	short
xs:int	System.Int32	int
xs:long	System.Int64	long
xs:unsignedByte	System.Byte	byte
xs:unsignedShort	System.UInt16	ushort

© 2016-2022 Altova GmbH

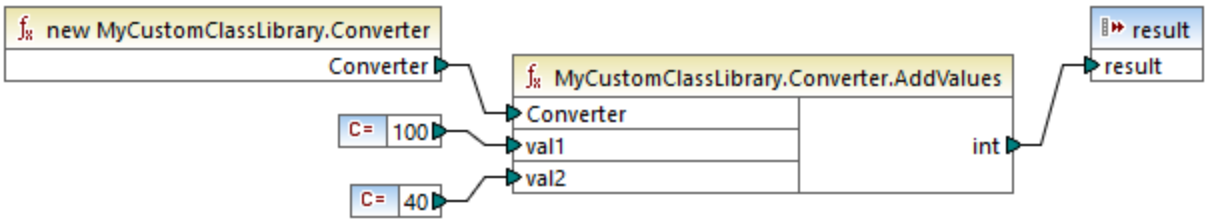
Tipos XML Schema	Tipos .NET	Tipos C#
xs:unsignedInt	System.UInt32	uint
xs:unsignedLong	System.UInt64	ulong
xs:boolean	System.Boolean	bool
xs:float	System.Single	float
xs:double	System.Double	double
xs:decimal	System.Decimal	decimal

Las conexiones pueden hacerse en ambas direcciones. Los demás tipos .NET/C# (incluidos los tipos matriz) no son compatibles. Los métodos que usen dichos parámetros o valores devueltos se omitirán. Se pueden usar tipos de objeto llamando a su constructor o como valor devuelto de un método. Se pueden asignar a otros métodos .NET, pero no se puede manipular el objeto en MapForce.

Tipos de datos: problemas y soluciones

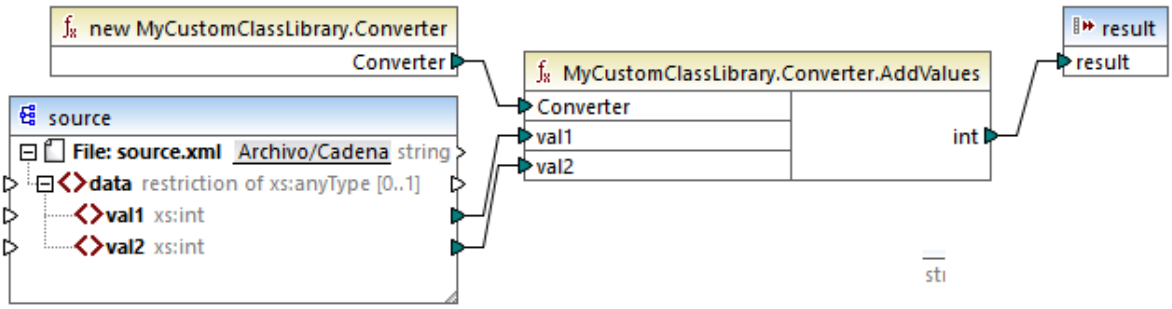
Si una función de una biblioteca personal espera tipos enteros y usted conecta constantes de tipo "Número" a los argumentos de la función, puede ocurrir un error de incompatibilidad parecido a este:

`No match for MyCustomClassLibrary, Version=1.0.0.0, Culture=neutral, PublicKeyToken=null.MyCustomClassLibrary.Converter.AddValues(MyCustomClassLibrary.Converter, xs:decimal, xs:decimal). Check argument types.` Este problema ocurre solamente con las constantes de tipo "Número". A continuación puede ver una asignación de ejemplo que podría producir este error. En la asignación hay dos constantes de tipo "Número" conectadas a los argumentos de la función (de tipo "Entero").



Existen varias soluciones:

1. Cambie el tipo de la constante de "Número" a "Demás opciones". (Para ello haga doble clic en la barra de título de la constante.)
2. En lugar de una constante, use un componente de entrada (por ejemplo, un archivo XML) que pase valores del tipo que espera la función.



3. En su código externo cree una función contenedora que acepte un valor decimal y devuelva un valor entero. Esta solución se puede importar como una biblioteca por separado, por lo que no es necesario cambiar el código fuente original de la función de destino para usar esta solución.

6.5.3.1 Ejemplo: importar clase Java personales

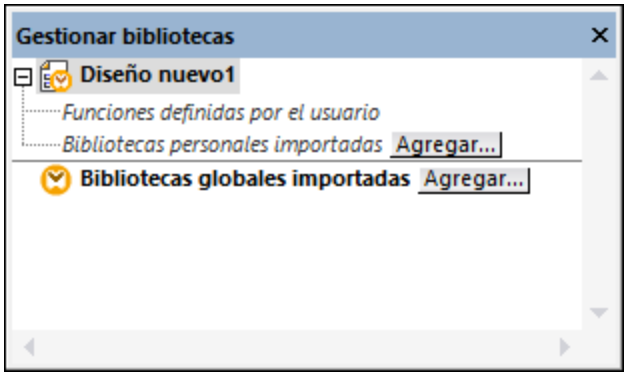
Este ejemplo explica cómo importar un archivo de clases Java personal en MapForce.

Nota: para este ejemplo es necesario tener Java SE 8 Runtime Environment o superior.

Importar Java .class

Para agregar el archivo `.class` de Java como biblioteca de MapForce:

1. Haga clic en el botón **Agregar o quitar bibliotecas**, en la parte inferior de la ventana Bibliotecas 30. Se abre la ventana **Gestionar bibliotecas**.

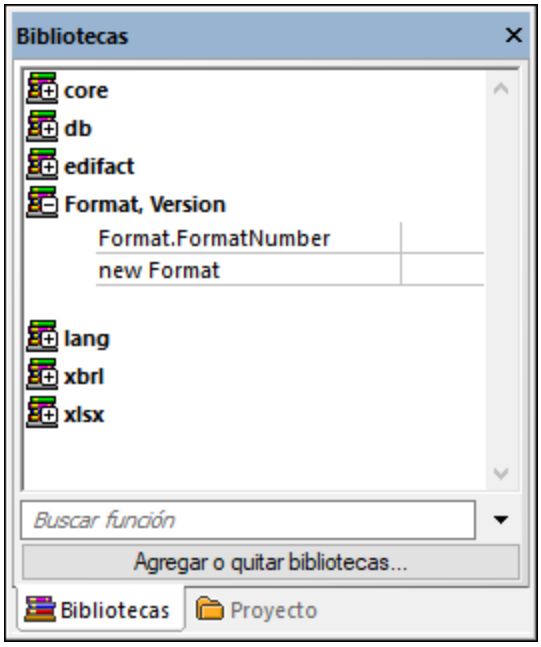


2. Para importar funciones como biblioteca local (dentro del archivo de asignación actual solamente), haga clic en **Agregar** bajo el nombre de la asignación activa. Para importar funciones como biblioteca global (a nivel de programa), haga clic en **Agregar**, junto a **Bibliotecas globales importadas**. Cuando se importa una biblioteca de forma local, puede hacer que la ruta de acceso al archivo de la biblioteca sea relativa al archivo de asignación. Con las bibliotecas globales la ruta siempre es absoluta.

3. Ahora haga clic en el botón **Agregar** y seleccione el archivo **Format.class** de la carpeta **`<Documentos>\Altova\MapForce2023\MapForceExamples\Java\Format\Format.class`**. Aparece un

 © 2016-2022 Altova GmbH

mensaje confirmando que se añadió una biblioteca nueva. La biblioteca importada aparece en la ventana Bibliotecas (*imagen siguiente*).



Si no ve la biblioteca que acaba de importar, asegúrese de que el lenguaje de transformación 24 es Java. Para añadir la función a la asignación basta con arrastrarla desde la ventana Bibliotecas al área de la asignación (*véase también Agregar una función integrada a una asignación de datos* 550).

Resultado de la asignación
Para obtener una vista previa del resultado de la asignación en MapForce:

1. Abra el archivo **`<Documentos>\Altova\MapForce2023\MapForceExamples\Java\FormatNumber.mfd`**.
2. Abra el panel *Resultados* para ver el resultado de la asignación.

```
1   Start date,End date,Region,Amount
2   2008-01-01,2008-01-31,CA,"110.400,00"
3   2008-01-01,2008-01-31,MA,"75.300,00"
4   2008-02-01,2008-02-29,CA,"114.300,00"
5   2008-02-01,2008-02-29,MA,"65.200,00"
6   2008-03-01,2008-03-31,CA,"134.200,00"
7   2008-03-01,2008-03-31,MA,"86.100,00"
8   2008-04-01,2008-04-30,CA,"107.300,00"
9   2008-04-01,2008-04-30,MA,"112.100,00"
10  2008-05-01,2008-05-31,CA,"114.400,00"
11  2008-05-01,2008-05-31,MA,"93.800,00"
```

Asignación en Java
Para ejecutar la asignación en Java:

1. En el menú **Archivo** haga clic en **Generar código en | Java**.
2. Seleccione el directorio de destino donde se debe generar el código y haga clic en **Aceptar**.
3. Importe las bibliotecas generadas a su proyecto de Java y genere la aplicación Java (*véase Ejemplo: generar una aplicación Java con Eclipse y Ant* 1089).

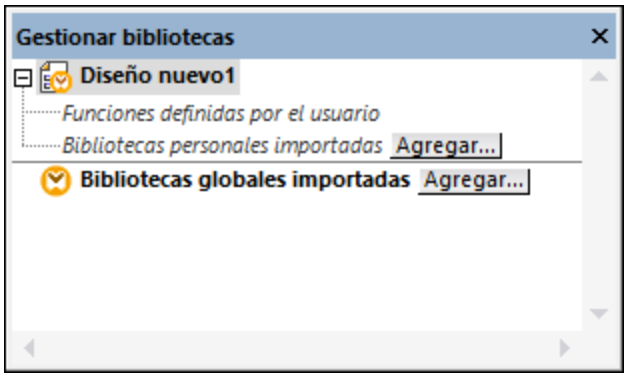
6.5.3.2 Ejemplo: importar ensamblado .NET DLL personal

Este ejemplo explica cómo importar un archivo de ensamblados .NET DLL creado en C#. Puede encontrar el código fuente de este ejemplo en **`<Documentos>\Altova\MapForce2023\MapForceExamples\C#\Format`**. El archivo de ensamblado .dll que se debe importar en ;MapForce está en el directorio **`..\bin\Debug`**. También puede abrir el archivo .sln de la solución en Visual Studio y compilar un .dll nuevo.

Nota: si quiere usar funciones .NET personales en la vista previa del resultado (en el panel **Resultados**), esas funciones deben estar compiladas para .NET Framework 4.x o .NET Standard 2.0.

Importar ensamblados .NET
Para importar el archivo de ensamblados .NET:

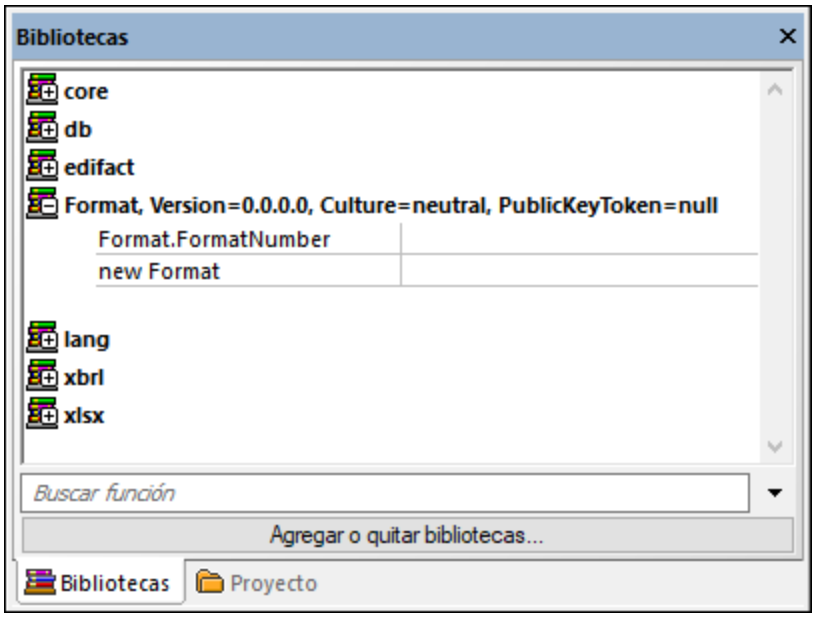
1. Haga clic en el botón **Agregar o quitar bibliotecas**, en la parte inferior de la ventana Bibliotecas 30. Se abre la ventana **Gestionar bibliotecas**.



2. Para importar funciones como biblioteca local (dentro del archivo de asignación actual solamente), haga clic en **Agregar** bajo el nombre de la asignación activa. Para importar funciones como biblioteca global (a nivel de programa), haga clic en **Agregar**, junto a **Bibliotecas globales importadas**. Cuando se importa una biblioteca de forma local, puede hacer que la ruta de acceso al archivo de la biblioteca sea relativa al archivo de asignación. Con las bibliotecas globales la ruta siempre es absoluta.

3. Ahora haga clic en el botón **Agregar** y seleccione el archivo **`Format.dll`** de la carpeta **`...\MapForceExamples\C#\Format\bin\Debug\`**. Aparece un mensaje confirmando que se añadió una biblioteca nueva. La biblioteca importada aparece en la ventana Bibliotecas.

 © 2016-2022 Altova GmbH



Si no ve la biblioteca que acaba de importar, asegúrese de que el lenguaje de transformación 24 es C#. Para añadir la función a la asignación basta con arrastrarla desde la ventana Bibliotecas al área de la asignación (*véase también Agregar una función integrada a una asignación de datos* 550).

Resultado de la asignación

Para obtener una vista previa del resultado de la asignación en MapForce:

1. Abra el archivo **`FormatNumber.mfd`** situado en la carpeta **`...\MapForceExamples\C#`**.
2. Abra el panel *Resultados* para ver el resultado de la asignación.

```
1   Start date,End date,Region,Amount
2   2008-01-01,2008-01-31,CA,"110.400,00"
3   2008-01-01,2008-01-31,MA,"75.300,00"
4   2008-02-01,2008-02-29,CA,"114.300,00"
5   2008-02-01,2008-02-29,MA,"65.200,00"
6   2008-03-01,2008-03-31,CA,"134.200,00"
7   2008-03-01,2008-03-31,MA,"86.100,00"
8   2008-04-01,2008-04-30,CA,"107.300,00"
9   2008-04-01,2008-04-30,MA,"112.100,00"
10  2008-05-01,2008-05-31,CA,"114.400,00"
11  2008-05-01,2008-05-31,MA,"93.800,00"
```

Mapping in C#

Para ejecutar la asignación desde una aplicación C# personal:

1. En el menú **Archivo** haga clic en **Generar código en | C#**.
2. Seleccione el directorio de destino donde se debe generar el código y haga clic en **Aceptar**.
3. Genere la aplicación con Visual Studio y ejecute la aplicación de consola generada (*véase Generar código C#* 1076).

6.5.4 Referencias manuales a bibliotecas Java, C# y C++ personales

En esta sección explicamos cómo hacer referencia a bibliotecas personales en archivos `.mff` (archivo de funciones MapForce). El archivo `.mff` que contiene la referencia se puede importar como biblioteca de MapForce. El archivo de bibliotecas `.mff` es un archivo XML donde se definen a mano las correspondencias entre definiciones de clase del código personal y de MapForce. Tras crear el archivo `.mff` personal podrá importarlo en MapForce como si se tratase de un archivo de clases Java o de ensamblados .NET.

Tenga en cuenta que:

- si quiere usar funciones .NET personales en la vista previa del resultado (en el panel **Resultados**), esas funciones deben estar compiladas para .NET Framework 4.x o .NET Standard 2.0.
- para poder aparecer en la ventana Bibliotecas de MapForce las funciones importadas deben tener tipos devueltos y parámetros de tipo simple. Puede consultar listas de tipos de datos para cada lenguaje en el apartado Correspondencias entre tipos de datos 618.
- cuando importe bibliotecas de funciones de archivos .mff personales, la vista previa de resultados tendrá algunas limitaciones. Para las bibliotecas escritas en C++ la vista previa no estará disponible. En el caso de Java y C#, la vista previa estará disponible cuando la biblioteca use tipos de lenguaje nativos, pero no lo estará si la biblioteca importa las clases generadas de Altova. Sin embargo, recuerde que puede generar código en el lenguaje al que hace referencia su biblioteca. Las funciones personales estarán disponibles en el código generado. Es decir, podrá ejecutar la asignación a partir del código generado.
- el orden exacto en el que la asignación llama a las funciones es indefinido. MapForce puede almacenar en caché los resultados calculados o evaluar las expresiones en cualquier orden. Por tanto, se recomienda utilizar funciones personales que no tengan efectos secundarios.
- es importante distinguir entre funciones definidas por el usuario y bibliotecas de funciones personales. Las funciones definidas por el usuario se crean de forma gráfica dentro de una asignación de datos y no pueden ni deben guardarse en un archivo `*.mff` porque se guardan en el archivo de asignación .mfd donde se crearon. Para más información consulte la sección Funciones definidas por el usuario 576.
- si su versión es anterior a la versión 2010, puede que necesite actualizar los tipos de datos utilizados en sus funciones personales (*véase* *Asignaciones de tipos de datos* 618).

Para aprender a crear y configurar un archivo `.mff` personal consulte el apartado Configurar el archivo .mff 612. Para ver algunos ejemplos consulte estos apartados:

- Ejemplo: hacer referencia a C# en .mff 621
- Ejemplo: hacer referencia a C++ en .mff 622
- Ejemplo: hacer referencia a Java en .mff 625

6.5.4.1 Configurar el archivo .mff

El archivo de funciones de MapForce (`.mff`) es un archivo de configuración en formato XML que permite adaptar funciones de bibliotecas Java, C# y C++ personales a MapForce para que aparezcan en la ventana

 © 2016-2022 Altova GmbH

Bibliotecas. Un archivo `.mff` hace de intermediario entre las bibliotecas personales del usuario y MapForce y debe configurarse para especificar las interfaces con las funciones personales y dónde se puede encontrar la implementación en el código generado.

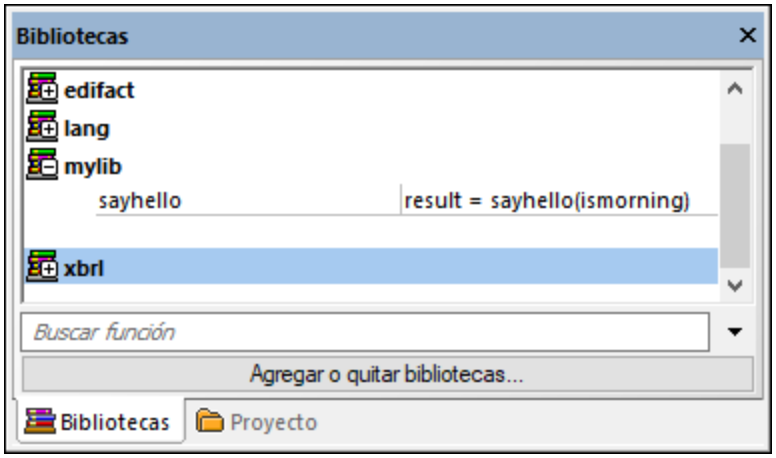
Tenga en cuenta que:

- los archivos de biblioteca `*.mff` deben ser válidos según el archivo de esquema **mff.xsd** de la carpeta **`C:\Archivos de programa\MapForceLibraries\mff.xsd`**. El esquema `mff.xsd` define la configuración de biblioteca personal y es para uso interno. Altova GmbH se reserva el derecho de cambiar el formato de este archivo en futuras versiones del software.
- Solamente se puede definir un archivo C#, C++ o .class Java por archivo `.mff`.

Archivo .mff C# de ejemplo
Este fragmento de código corresponde a un archivo .mff para C++:

```
<?xml version="1.0" encoding="UTF-8"?>
<mapping version="9" library="mylib" xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="mff.xsd">
   <implementations>
      <implementation language="cpp">
         <setting name="namespace" value="mylib"/>
         <setting name="class" value="Greetings"/>
         <setting name="path" value="C:\Libraries\cpp"/>
         <setting name="include" value="Greetings.h"/>
         <setting name="source" value="Greetings.cpp"/>
      </implementation>
   </implementations>
   <group name="greetings">
      <component name="sayhello">
         <sources>
            <datapoint name="ismorning" type="xs:boolean"/>
         </sources>
         <targets>
            <datapoint name="result" type="xs:string"/>
         </targets>
         <implementations>
            <implementation language="cpp">
               <function name="SayHello"/>
            </implementation>
         </implementations>
         <description>
            <short>result = sayhello(ismorning)</short>
            <long>Returns "Good morning" or "Good day", depending on the input
parameter.</long>
         </description>
      </component>
   </group>
</mapping>
```

Biblioteca personal importada
La imagen que aparece a continuación muestra el resultado de importar un archivo `.mff` personal en MapForce. Observe que la biblioteca personal **helloworld** aparece en la ventana Bibliotecas y que contiene la función de cadena "hello".



Configuración

A continuación encontrará las instrucciones para adaptar el archivo `.mff` a los requisitos de su proyecto.

Paso 1. Configurar el nombre de la biblioteca

El nombre de la biblioteca está en la línea del archivo `.mff` que aparece a continuación. El nombre de la biblioteca se escribe en minúsculas.

```
<mapping version="9" library="mylib" xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="mff.xsd">
```

En este ejemplo, la biblioteca aparecerá en la ventana Bibliotecas de MapForce con el nombre "helloworld".

Paso 2. Configurar las implementaciones del lenguaje

El elemento `<implementations>` es un elemento obligatorio que especifica con qué lenguajes debe ser compatible la biblioteca y debe ser un secundario de `<mapping>`. Veamos un ejemplo:

```
<!-- ... -->
<mapping version="9" library="mylib" xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="mff.xsd">
   <implementations>
      <implementation language="cpp">
         <setting name="namespace" value="mylib"/>
         <setting name="class" value="Greetings"/>
         <setting name="path" value="C:\Libraries\cpp"/>
         <setting name="include" value="Greetings.h"/>
         <setting name="source" value="Greetings.cpp"/>
      </implementation>
   </implementations>
<!-- ... -->
```

Las opciones de cada elemento `<implementation>` permiten al código generado llamar a las funciones definidas en Java, C++ o C#. Puede configurar el archivo .mff para que apunte a más de un lenguaje de

 © 2016-2022 Altova GmbH

programación. En este caso, cada lenguaje debe contener un elemento `<implementation>`. Más abajo encontrará información sobre las opciones disponibles para cada lenguaje de programación.

Referencia de la biblioteca Java

```
<!-- ... -->
<implementation language="java">
   <setting name="package" value="com.hello.functions"/>
   <setting name="class" value="Greetings"/>
</implementation>
<!-- ... -->
```

Es importante que el código generado pueda encontrar el archivo **`Greetings.class`**. Por tanto, asegúrese de que la clase está en la variable classpath de Java.

Referencia de la biblioteca C#

```
<!-- ... -->
      <implementation language="cs">
         <setting name="namespace" value="MyLibrary" />
         <setting name="class" value="Greetings" />
         <setting name="reference" value="C:
\Libraries\cs\MyLibrary\bin\debug\MyLibrary.dll" />
      </implementation>
<!-- ... -->
```

Es importante que el espacio de nombres del código se corresponda con el espacio de nombres definido en el archivo `.mff` (en el fragmento de código anterior el espacio de nombres es **`MyLibrary`**). Lo mismo ocurre con el nombre de la clase (en el fragmento de código anterior el nombre de clase es **`Greetings`**). La tercera opción, `reference`, aporta la ruta de acceso del `dll` que debe vincularse al código generado.

Referencia de la biblioteca C++

```
<!-- ... -->
   <implementation language="cpp">
         <setting name="namespace" value="MyLibrary"/>
         <setting name="class" value="Greetings"/>
         <setting name="path" value="C:\Libraries\cpp"/>
         <setting name="include" value="Greetings.h"/>
         <setting name="source" value="Greetings.cpp"/>
   </implementation>
<!-- ... -->
```

En el fragmento de código C++ anterior podemos ver que:

- `namespace` es el espacio de nombres donde se definirá la clase `Greetings`. Debe ser igual que el nombre de la biblioteca.
- `path` es la ruta de acceso donde se encuentran los archivos de inclusión y de origen.
- cuando se genera código para una asignación de datos, los archivos de inclusión y de origen se copian en el directorio **`targetdir/libraryname`** (definido cuando se selecciona el comando de menú **Archivo | Generar código en C++** y se elije el directorio de destino) y también se incluyen en el archivo de proyecto.

Todos los archivos de inclusión se incluirán en el algoritmo que se genera.

Paso 3. Agregar un componente
En la ventana Bibliotecas de MapForce las funciones aparecen anidadas dentro de un grupo de funciones (p. ej. "string functions"). En el archivo `.mff` una función corresponde a un elemento `<component>`. Igualmente, cada `<component>` debe estar anidado dentro de un elemento `<group>`. Por ejemplo:

```
<!-- ... -->
<group name="string functions">
   <component name="sayhello">
   <!-- ... -->
   </component>
</group>
<!-- ... -->
```

El código que aparece a continuación define un ejemplo de función (componente) llamada **`sayhello`**.

```
<!-- ... -->
<component name="sayhello">
   <sources>
      <datapoint name="ismorning" type="xs:boolean"/>
   </sources>
   <targets>
      <datapoint name="result" type="xs:string"/>
   </targets>
   <implementations>
   <!-- ... -->
   </implementations>
   <description>
     <short>result = sayhello(ismorning)</short>
     <long>Returns "Good morning" or "Good day", depending on the input
parameter.</long>
   </description>
</component>
<!-- ... -->
```

Este sería el aspecto del componente anterior en MapForce:

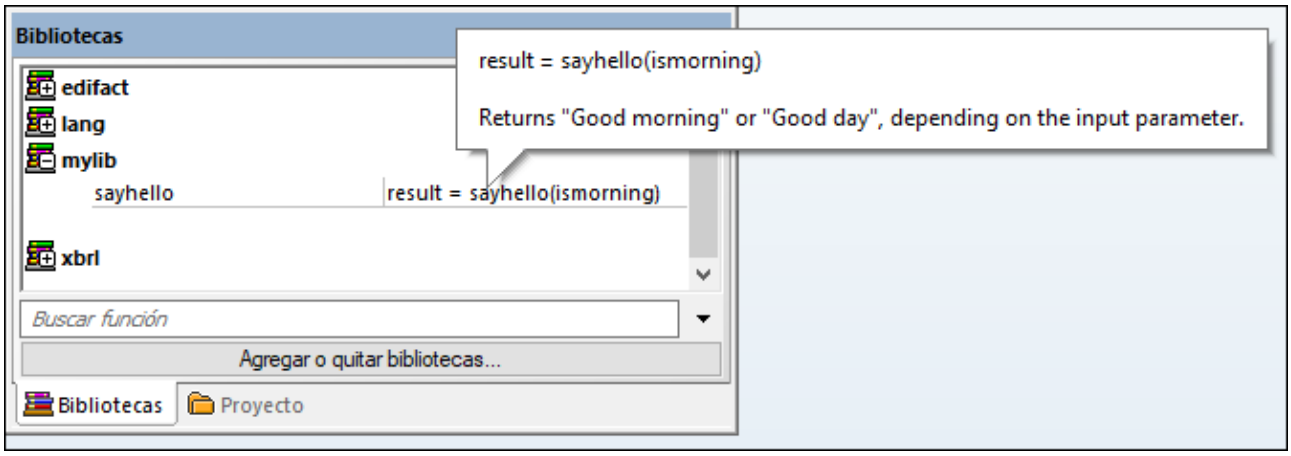
En el fragmento de código, se definió `<datapoint>` como parámetro de entrada o salida de una función (o conector de entrada o salida). El argumento `type` de `<datapoint>` especifica el tipo de datos del parámetro (o el tipo de datos del valor devuelto).

Por cada función se permite un punto de datos de destino como máximo, pero no hay límite en el número de puntos de datos de origen que se pueden definir.

El tipo de datos de cada punto de datos debe ser un tipo XML Schema (p. ej. `xs:string`, `xs:integer`, etc.). Estos tipos de datos deben corresponder a los tipos de datos de los parámetros de la función que definió en la biblioteca Java, C++ o C#. Para ver la correspondencia entre tipos de datos XML Schema y de otros lenguajes consulte el apartado Correspondencias entre tipos de datos 618.

 © 2016-2022 Altova GmbH

En la ventana Bibliotecas de MapForce las funciones van acompañadas de una descripción breve y una más larga. La descripción breve aparece siempre a la derecha del nombre de la función. La descripción más larga aparece cuando se pasa el puntero por encima de la descripción breve.



Paso 4. Definir implementaciones de lenguaje

Llegados a este punto debemos crear una conexión entre la función de la ventana Bibliotecas y la función de las clases Java, C# o C++ personales. Esto se consigue con el elemento `<implementation>`. Tal y como se explica más a arriba, las funciones pueden tener varios elementos `<implementation>`, uno por cada lenguaje de programación compatible. Una función se puede llamar `Hello` en Java o `SayHello` en C++. Por eso es necesario especificar un nombre de función distinto por cada lenguaje de programación. A continuación puede ver una función con tres implementaciones distintas:

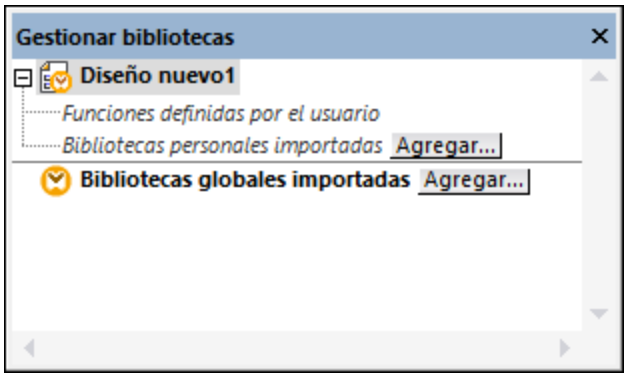
```
<!-- ... -->
<component name="sayhello">
<!-- ... -->
   <implementations>
      <implementation language="cs">
         <function name="HelloFunction"/>
      </implementation>
      <implementation language="java">
         <function name="Hello"/>
      </implementation>
      <implementation language="cpp">
         <function name="SayHello"/>
      </implementation>
   </implementations>
<!-- ... -->
</component>
<!-- ... -->
```

El valor que especifique como nombre de función debe ser idéntico al nombre del método de la clase Java, C# o C++.

6.5.4.2 Importar bibliotecas .mff

Tras crear el archivo `.mff` personal (*véase* *Configurar el archivo .mff* 612), podrá importarlo en MapForce siguiendo estas instrucciones:

1. Haga clic en el botón **Agregar o quitar bibliotecas**, en la parte inferior de la ventana Bibliotecas 30. Se abre la ventana **Gestionar bibliotecas**.



2. Para importar funciones como biblioteca local (dentro del archivo de asignación actual solamente), haga clic en **Agregar** bajo el nombre de la asignación activa. Para importar funciones como biblioteca global (a nivel de programa), haga clic en **Agregar**, junto a **Bibliotecas globales importadas**. Cuando se importa una biblioteca de forma local, puede hacer que la ruta de acceso al archivo de la biblioteca sea relativa al archivo de asignación. Con las bibliotecas globales la ruta siempre es absoluta.

3. Navegue hasta el archivo `.mff` personalizado y haga clic en **Abrir**.

La biblioteca importada aparece en la ventana Bibliotecas si el lenguaje de transformación seleccionado es el lenguaje al que apunta la biblioteca personal recién importada.

Si guarda el archivo `*.mff` en la carpeta **`...\Altova\MapForce2023\MapForceLibraries`** relativa a la carpeta **Archivos de programa** (o **Archivos de programa (x86)**), la biblioteca se cargará automáticamente en la ventana Bibliotecas cuando inicie MapForce. Las bibliotecas y sus funciones se pueden habilitar o deshabilitar con solo agregar o eliminar el correspondiente archivo de biblioteca (`*.mff`).

6.5.4.3 Correspondencias entre tipos de datos

En esta tabla puede ver los tipos de datos que se admiten como tipos devueltos de función y tipos de parámetro de función cuando se crean archivos `.mff` personales que hacen referencia a sus bibliotecas Java, C# y C++ personales. La tabla incluye tanto tipos de datos nativos como no nativos. Si necesita usar tipos de datos no nativos como los tipos date, time y duration de Altova, sus bibliotecas Java y C# personales deberán incluir una referencia a las bibliotecas de Altova. En el caso de C++, deberá importar las bibliotecas de Altova. Para más información sobre cómo generar las bibliotecas de Altova consulte la sección Generador de código 1067.

Tipo XML Schema	Tipo Java	Tipo C#	Tipo C++
`anyAtomicType`	`String`	`string`	`string_type`

 © 2016-2022 Altova GmbH

Tipo XML Schema	Tipo Java	Tipo C#	Tipo C++
anySimpleType	String	string	string_type
anyURI	String	string	string_type
base64Binary	byte[]	byte[]	altova::mapforce::blob
boolean	boolean	bool	bool
byte	int	int	int
date	com.altova.types.DateTime 1189	Altova.Types.DateTime 1172	altova::DateTime 1155
dateTime	com.altova.types.DateTime 1189	Altova.Types.DateTime 1172	altova::DateTime 1155
dayTimeDuration	com.altova.types.Duration 1194	Altova.Types.Duration 1177	altova::Duration 1159
decimal	java.math.BigDecimal	decimal	double
double	double	double	double
duration	com.altova.types.Duration 1194	Altova.Types.Duration 1177	altova::Duration 1159
ENTITIES	String	string	string_type
ENTITY	String	string	string_type
float	double	double	double
gDay	com.altova.types.DateTime 1189	Altova.Types.DateTime 1172	altova::DateTime 1155
gMonth	com.altova.types.DateTime 1189	Altova.Types.DateTime 1172	altova::DateTime 1155
gMonthDay	com.altova.types.DateTime 1189	Altova.Types.DateTime 1172	altova::DateTime 1155
gYear	com.altova.types.DateTime 1189	Altova.Types.DateTime 1172	altova::DateTime 1155
gYearMonth	com.altova.types.DateTime 1189	Altova.Types.DateTime 1172	altova::DateTime 1155
hexBinary	byte[]	byte[]	altova::mapforce::blob
ID	String	string	string_type
IDREF	String	string	string_type
IDREFS	String	string	string_type

Tipo XML Schema	Tipo Java	Tipo C#	Tipo C++
int	int	int	int
integer	java.math.BigInteger	decimal	__int64
language	String	string	string_type
long	long	long	__int64
Name	String	string	string_type
NCName	String	string	string_type
negativeInteger	java.math.BigInteger	decimal	__int64
NMTOKEN	String	string	string_type
NMTOKENS	String	string	string_type
nonNegativeInteger	java.math.BigInteger	decimal	unsigned __int64
nonPositiveInteger	java.math.BigInteger	decimal	__int64
normalizedString	String	string	string_type
NOTATION	String	string	string_type
positiveInteger	java.math.BigInteger	decimal	unsigned __int64
QName	javax.xml.namespace.QNam e	Altova.Types.QName	altova::QName
short	int	int	int
string	String	string	string_type
time	com.altova.types.DateTim e 1189	Altova.Types.DateTime 1172	altova::DateTime 1155
token	String	string	string_type
unsignedByte	long	ulong	unsigned __int64
unsignedInt	long	ulong	unsigned __int64
unsignedLong	java.math.BigInteger	ulong	unsigned __int64
unsignedShort	long	ulong	unsigned __int64
untypedAtomic	String	string	string_type
yearMonthDuration	com.altova.types.Duratio n 1194	Altova.Types.Duration 1177	altova::Duration 1159

© 2016-2022 Altova GmbH

6.5.4.4 Ejemplo: adaptar C# a .mff

Este apartado explica cómo se puede crear una biblioteca C# de muestra y hacer referencia a ella en un archivo de funciones MapForce (`.mff`). El archivo `.mff` se puede importar como biblioteca MapForce. Hacer referencia a una biblioteca C# en un archivo `.mff` es una de las formas que existen de importar bibliotecas C# en MapForce. Una opción más sencilla es importar los ensamblajes .NET directamente (*véase Ejemplo: importar ensamblado .NET DLL personal* 610).

Configuración

Siga estos pasos para hacer referencia a una biblioteca C# en un archivo `.mff`:

Nota: si quiere usar funciones .NET personales en la vista previa del resultado (en el panel **Resultados**), esas funciones deben estar compiladas para .NET Framework 4.x o .NET Standard 2.0.

Paso 1. Crear un proyecto de bibliotecas de clases nuevo en Visual Studio
Cree un proyecto de bibliotecas de clases nuevo en Visual Studio. Observe que la función se definió como `public static`.

```
namespace MyLibrary
{
    public class Greetings
    {
        public static string SayHello(bool isMorning)
        {
            if (isMorning)
                return "Good morning!";
            return "Good Day!";
        }
    }
}
```

Paso 2. Agregar una referencia a Altova.dll
Si necesita tipos XML Schema especiales (como date y duration), deberá agregar una referencia desde el proyecto de Visual Studio a la biblioteca `Altova.dll`. Para obtener esta biblioteca debe generar código C# a partir de una asignación de datos de MapForce sin funciones personales. El archivo `Altova.dll` se guarda en el directorio **`..\Altova\bin\debug`** relativo al directorio donde se generó el código. Para agregar una referencia a `Altova.dll` en Visual Studio haga clic en **Proyecto | Agregar referencia** y navegue hasta el archivo `Altova.dll`. Además, debe añadir esta línea a su código: **`using Altova.Types;`**. Para más información sobre las correspondencias entre tipos XML Schema y tipos C# consulte el apartado Correspondencias entre tipos de datos 618.

Paso 3. Generar un proyecto de Visual Studio
Genere su proyecto de Visual Studio. El archivo `MyLibrary.dll` se genera en el directorio de salida del proyecto.

Paso 4.Crear el archivo .mff y hacer referencia a su biblioteca C#
En un editor XML cree un archivo `.mff` nuevo y valídelo con el esquema **`C:\Program Files\MapForceLibraries\mff.xsd`**. Asegúrese de que todas las referencias bajo **`implementation language="cs"`** apuntan a los miembros y las rutas C# correctos, es decir, los que creó anteriormente. La línea **`function name="SayHello"`** debe hacer referencia al nombre de función exactamente como esté definido en C#. Para más información consulte el apartado Configurar el archivo .mff 612.

```
<?xml version="1.0" encoding="UTF-8"?>
<mapping version="9" library="mylib" xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="mff.xsd">
   <implementations>
      <implementation language="cs">
         <setting name="namespace" value="MyLibrary" />
         <setting name="class" value="Greetings" />
         <setting name="reference" value="C:
\Libraries\cs\MyLibrary\bin\debug\MyLibrary.dll" />
      </implementation>
   </implementations>
   <group name="string functions">
      <component name="sayhello">
         <sources>
            <datapoint name="ismorning" type="xs:boolean"/>
         </sources>
         <targets>
            <datapoint name="result" type="xs:string"/>
         </targets>
         <implementations>
            <implementation language="cs">
               <function name="SayHello"/>
            </implementation>
         </implementations>
         <description>
            <short>result = sayhello(ismorning)</short>
            <long>Returns "Good morning" or "Good day", depending on the input
parameter.</long>
         </description>
      </component>
   </group>
</mapping>
```

Paso 5. Importar .mff como biblioteca
Ahora que el archivo `.mff` hace referencia a su biblioteca personal puede importarlo en MapForce como biblioteca.Para más información véase Importar bibliotecas .mff 618.

6.5.4.5 Ejemplo: adaptar C++ a .mff

Este apartado explica cómo se puede crear una biblioteca C++ y hacer referencia a ella desde un archivo archivo de función de MapForce (`.mff`) para que esa biblioteca aparezca en la ventana Bibliotecas de MapForce.

Configuración

Siga estos pasos para hacer referencia a una biblioteca C++ en un archivo `.mff`:

Paso 1. Crear un archivo de encabezado
Cree un archivo de encabezado (`.h`) para su biblioteca de clases. Por ejemplo, a continuación puede ver un ejemplo de archivo de encabezado llamado **`Greetings.h`**.

 © 2016-2022 Altova GmbH

```
#ifndef MYLIBRARY_GREETINGS_H_INCLUDED
#define MYLIBRARY_GREETINGS_H_INCLUDED

#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000

using namespace altova;

namespace mylib {

class ALTOVA_DECLSPECIFIER Greetings
{
   public:
    static string_type SayHello(bool isMorning);
};

} // namespace mylib

#endif // MYLIBRARY_GREETINGS_H_INCLUDED
```

Observe que la función se declaró como estática y que el espacio de nombres `altova` se importa. Recuerde que debe escribir `ALTOVA_DECLSPECIFIER` delante del nombre de clase para que las clases se compilen correctamente (tanto si usa vínculos dinámicos o estáticos en el código que genere más adelante).

Paso 2. Crear un archivo .cpp

Cree un archivo .cpp que se llame igual que el archivo de encabezado. El archivo .cpp debe estar en el mismo directorio que el archivo .h. Por ejemplo, a continuación puede ver un ejemplo de archivo .cpp llamado **`Greetings.cpp`** que incluye el archivo **`Greetings.h`** que creamos en el primer paso.

```
#include "StdAfx.h"
#include "../Altova/Altova.h"
#include "../Altova/AltovaException.h"
#include "../Altova/SchemaTypes.h"

#include "Greetings.h"

namespace mylib {

   string_type Greetings::SayHello(bool isMorning)
   {
      if( isMorning )
         return _T("Good morning!");
      return _T("Good day!");
   }
}
```

Fíjese en las líneas que importan el archivo **`StdAfx.h`** y varias bibliotecas de Altova. Estas líneas no se deben modificar. Las rutas de acceso de las bibliotecas de Altova son correctas. En el código que se generará estas rutas de acceso dirigirán a los archivos pertinentes. A diferencia de Java y C#, en C++ no es necesario compilar los archivos de origen C++. Se copiarán en el código generado y se compilarán con el resto del código de asignación.

Paso 3. Crea el archivo .mff y hacer referencia a su biblioteca C++
En un editor XML cree un archivo `.mff` nuevo y valídelo con el esquema **`C:\Program Files\MapForceLibraries\mff.xsd`**. Recuerde que el espacio de nombres y los nombres de función y los tipos de datos aquí definidos deben corresponderse con los del código C++ tal y como se explica en el apartado Configurar el archivo .mff 612. Para más información sobre los tipos de datos consulte Correspondencias entre tipos de datos 618.

```
<?xml version="1.0" encoding="UTF-8"?>
<mapping version="9" library="mylib" xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="mff.xsd">
   <implementations>
      <implementation language="cpp">
         <setting name="namespace" value="mylib"/>
         <setting name="class" value="Greetings"/>
         <setting name="path" value="C:\Libraries\cpp"/>
         <setting name="include" value="Greetings.h"/>
         <setting name="source" value="Greetings.cpp"/>
      </implementation>
   </implementations>
   <group name="greetings">
      <component name="sayhello">
         <sources>
            <datapoint name="ismorning" type="xs:boolean"/>
         </sources>
         <targets>
            <datapoint name="result" type="xs:string"/>
         </targets>
         <implementations>
            <implementation language="cpp">
               <function name="SayHello"/>
            </implementation>
         </implementations>
         <description>
            <short>result = sayhello(ismorning)</short>
            <long>Returns "Good morning" or "Good day", depending on the input
parameter.</long>
         </description>
      </component>
   </group>
</mapping>
```

Paso 4. Importar .mff como biblioteca
Ahora que el archivo `.mff` hace referencia a su biblioteca personal puede importarlo en MapForce como biblioteca.Para más información véase Importar bibliotecas .mff 618.

Resolver errores de compilación C++
Recuerde que, para poder ejecutar asignaciones que usan bibliotecas C++ nativas, deberá generar código C++ y ejecutar la asignación desde el código o aplicación C++, como se explica en Generar código C++ 1071. Si recibe un error de compilador en esta línea:

```
#import "msado15.dll" rename("EOF", "EndOfFile")
```

 © 2016-2022 Altova GmbH

modifique las propiedades del proyecto para incluir una referencia al archivo **msado15.dll** en **C:\Program Files\Common Files\System\ADO**.

6.5.4.6 Ejemplo: adaptar Java a .mff

Este apartado explica cómo se puede crear una biblioteca Java y hacer referencia a ella en un archivo de funciones de MapForce (.mff) para que esa biblioteca aparezca en la ventana Bibliotecas de MapForce. Hacer referencia a una biblioteca Java al formato .mff es solamente una de las formas que existen de importar bibliotecas Java en MapForce. Una opción más sencilla es importar los archivos .class de Java directamente (*véase* Ejemplo: importar clase Java personales 608).

Configuración

Siga estos pasos para hacer referencia a una biblioteca C# en un archivo .mff:

Paso 1. Crear un proyecto nuevo
Cree un proyecto de Java nuevo en su entorno de desarrollo (p. ej. Eclipse).

Paso 2. Agregue el paquete com.mylib
Añada al proyecto un paquete nuevo llamado **com.mylib** compuesto por una clase llamada **Greetings**. A continuación puede ver un fragmento de código donde puede ver que la función SayHello se definió como public static.

```
package com.mylib;

public class Greetings {

   public static String SayHello ( boolean isMorning ) {
      if( isMorning )
         return "Good Morning!";
      return "Good Day!";
   }

}
```

Paso 3. Importar com.altova.types
También tiene la opción de importar el paquete com.altova.types si su proyecto necesita ser compatible con tipos especiales como date, time o duration. Para obtener este paquete debe generar código Java a partir de una asignación de datos que no incluya funciones personales: **import com.altova.types.*;**.

Paso 4. Compilar su biblioteca personal
Compile la biblioteca personal en un archivo de clases y añádalo a la variable classpath de Java.

Step 5. Create .mff and reference your Java library
En un editor XML cree un archivo .mff nuevo y valídelo con el esquema **C:\Program Files\MapForceLibraries\mff.xsd**. Asegúrese de que todas las referencias bajo **implementation language="java"** apuntan a los miembros Java que creó antes. Además, la línea **function name="SayHello"** debe hacer referencia al nombre de la función exactamente como esté definido en Java. Para más información consulte el apartado Configurar el archivo .mff 612.

```
<?xml version="1.0" encoding="UTF-8"?>
<mapping version="9" library="custom" xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="mff.xsd">
   <implementations>
      <implementation language="java">
         <setting name="package" value="com.mylib"/>
         <setting name="class" value="Greetings"/>
      </implementation>
   </implementations>
   <group name="greetings">
      <component name="sayhello">
         <sources>
            <datapoint name="ismorning" type="xs:boolean"/>
         </sources>
         <targets>
            <datapoint name="result" type="xs:string"/>
         </targets>
         <implementations>
            <implementation language="java">
               <function name="SayHello"/>
            </implementation>
         </implementations>
         <description>
            <short>result = sayhello(ismorning)</short>
            <long>Returns "Good morning" or "Good day", depending on the input
parameter.</long>
         </description>
      </component>
   </group>
</mapping>
```

Paso 6. Importar .mff como biblioteca

Ahora que el archivo `.mff` hace referencia a su biblioteca personal puede importarlo en MapForce como biblioteca.Para más información véase Importar bibliotecas .mff 618.

 © 2016-2022 Altova GmbH

6.6 Expresiones regulares

Al diseñar una asignación en MapForce puede usar expresiones regulares ("regex") en estos contextos:

- En el parámetro **pattern** de las funciones `match-pattern` 768 y `tokenize-regexp` 720
- Para filtrar los nodos a los que afecta una función de nodo. Para más información consulte Aplicar funciones de nodo y valores predeterminados de forma condicional 568.

La sintaxis de las expresiones regulares en XSLT y XQuery se explican en Appendix F of "XML Schema Part 2: Datatypes Second Edition".

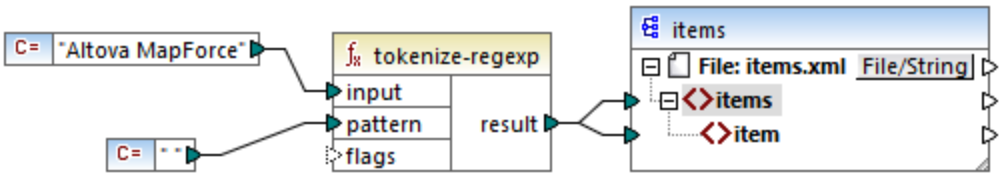
Nota: al generar código C++, C# o Java, las características avanzadas de la sintaxis de la expresión regular pueden variar ligeramente. Consulte la documentación regex de cada lenguaje para más información.

Terminología

Vamos a usar la función `tokenize-regexp` para analizar la sintaxis de las expresiones regulares. Esta función divide texto en una secuencia de cadenas con ayuda de las expresiones regulares. Para ello la función toma estos parámetros de entrada:

input	La cadena de entrada que va a procesar la función. La expresión regular que operará en esta cadena.
pattern	La expresión regular pattern que se aplica.
flags	Este parámetro es opcional y define las demás opciones (flags) que determinan cómo se interpreta una expresión regular; consulte "Flags" más abajo.

En la asignación siguiente la cadena de entrada es "Altova MapForce". El parámetro **pattern** es un carácter de espacio y no se usan expresiones regulares flag.



Lo que ocurre es que se separa el texto cada vez que aparece el carácter espacio, por lo que el resultado de la asignación es:

```
<items>
   <item>Altova</item>
   <item>MapForce</item>
</items>
```

Observe que la función `tokenize-regexp` excluye los caracteres que coinciden del resultado. En otras palabras, el carácter espacio de este ejemplo se omite en el resultado.

El ejemplo anterior es muy básico, por lo que se puede conseguir el mismo resultado sin expresiones regulares, usando la función tokenize[715]. En un caso real el parámetro **pattern** contendría una expresión regular más compleja. La expresión regular puede consistir en:

- Literales
- Clases de caracteres
- Intervalos de caracteres
- Clases negadas
- Metacaracteres
- Cuantificadores

Literales

Use literales para que los caracteres que coincidan sean idénticos a los que indica. Por ejemplo, si la cadena de entrada es `abracadabra` y **pattern** es el literal `br`, el resultado es:

```
<items>
   <item>a</item>
   <item>acada</item>
   <item>a</item>
</items>
```

La explicación es que el literal `br` encontró dos coincidencias en la cadena de entrada `abracadabra`. Una vez omitidos esos caracteres del resultado, queda la secuencia de tres cadenas que se ve en el extracto de código.

Clases de caracteres

Si encierra un conjunto de caracteres entre corchetes (`[` y `]`) se crea una clase de caracteres. Solamente coincide uno de los caracteres dentro de esa clase, por ejemplo:

- El patrón `[aeiou]` busca cualquier vocal en minúsculas.
- El patrón `[mj]ust` busca "must" y "just".

Nota: el patrón distingue entre mayúsculas y minúsculas. Por ejemplo, "a" no encontrará "A". Para que el patrón deje de distinguir entre mayúsculas y minúsculas debe usar el parámetro flag **i**, véase más abajo.

Intervalos de caracteres

Use `[a-z]` para crear el intervalo comprendido entre dos caracteres. Solamente se encontrará uno de los caracteres en cada búsqueda. Por ejemplo, el patrón `[a-z]` busca cualquier carácter de la a a la z que esté en minúsculas.

Clases negadas

Si usa el acento circunflejo (`^`) como primer carácter tras el corchete de apertura, se niega la clase de caracteres. Por ejemplo, el patrón `[^a-z]` busca cualquier carácter que no esté en esa clases de caracteres, incluidas las líneas nuevas.

 © 2016-2022 Altova GmbH

Buscar cualquier carácter.

Use el metacarácter punto (**.**) para buscar un solo carácter, sea cual sea (excepto líneas nuevas). Por ejemplo, **.** busca un solo carácter, sea cual sea.

Cuantificadores

Dentro de una expresión regular, los cuantificadores definen cuántas veces debe aparecer el carácter o la subexpresión anteriores para que la búsqueda obtenga resultados.

?	Busca cero o una coincidencia de la cadena precedente (la cadena es opcional) Por ejemplo, el patrón **mo?** Encuentra "m" y "mo".
+	Busca una o más coincidencias de la cadena precedente. Por ejemplo, el patrón **mo+** Encuentra "mo", "moo", "mooo", etc.
*****	Busca cero o más coincidencias de la cadena precendente.
{min,max}	Busca el número de repeticiones entre *min* y *max*.. P. ej. **mo{1,3}**encuentra "mo", "moo", and "mooo".

Paréntesis

Los paréntesis **(** y **)** se usan para agrupar partes de la expresión regular. Se pueden usar para aplicar cuantificadores a una subexpresión (en vez de a un solo carácter) o con alternancia (véase a continuación).

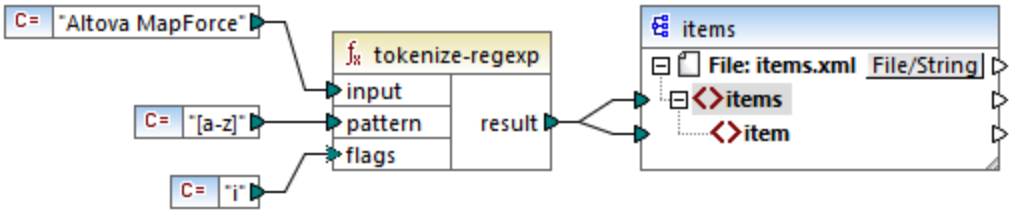
Alternancia

La barra vertical **|** se usa con el significado "o". Se puede usar para buscar cualquiera de las distintas subexpresiones separadas por **|**. Por ejemplo, el patrón **(horse|make) sense** busca tanto "horse sense" como "make sense".

Flags

Estos parámetros opcionales definen cómo se debe interpretar la expresión regular. Para establecer las opciones se usan letras, que pueden estar en cualquier orden y se pueden repetir.

s	Si está presente, el proceso de búsqueda opera en el modo "dot-all". Si la cadena de entrada input contiene "hello" y "world" en dos líneas *diferentes*, la expresión **hello*world** solamente encontrará resultados si se establece la marca flag **s**.
m	Si está presente, el proceso de búsqueda opera en el modo multilínea. En el modo multilínea el acento circunflejo **^** busca el principio de una línea, sea cual sea. Es decir, el inicio de una cadena entera y el primer carácter que aparece después del carácter de línea nueva. El carácter de dólar **$** busca el fin de una línea, sea cual sea. Es decir, el final de una cadena entera y el primer carácter que aparece antes del carácter de línea nueva.

	El carácter de línea nueva es **#x0A**..
i	Si está presente, el proceso de búsqueda no distingue entre mayúsculas y minúsculas. Por ejemplo, la expresión regular `[a-z]` más la marca **i** encontrará todas las letras de la a-z y entre A-Z. 
x	Si está presente, los caracteres de espacio en blanco se quitan de la expresión regular antes de iniciar el proceso de búsqueda. Los caracteres de espacio en blanco son **#x09**, **#x0A**, **#x0D** y **#x20**. **Nota:** los caracteres de espacio en blanco sin expresiones de clases de caracteres no se eliminan (por ejemplo, `[#x20]`).

© 2016-2022 Altova GmbH

6.7 Referencia de la biblioteca de funciones

En esta sección se describen todas las funciones integradas de MapForce que aparecen en la ventana Bibliotecas [30]. Estas funciones se organizan por bibliotecas. Las funciones disponibles en la ventana **Bibliotecas** dependen del lenguaje de transformación que haya escogido para la asignación. Para más información sobre los lenguajes de transformación disponibles consulte este apartado [24].

A continuación puede encontrar información sobre la compatibilidad de los lenguajes de transformación.

core | aggregate functions (agregado)

La lista siguiente es un resumen de la compatibilidad de las funciones principales con los lenguajes de transformación.

core | aggregate functions (agregado)

- **avg**, **max**, **max-string**, **min**, **min-string**: XSLT 2.0, XSLT 3.0, XQuery 1.0, C#, C++, Java, Built-In;
- **count**, **sum**: todos los lenguajes de transformación.

core | conversion functions (conversión)

- **boolean**, string, **number**: todos los lenguajes de transformación;
- **format-date**, **format-dateTime**, **format-time**: XSLT 2.0, XSLT 3.0, C#, C++, Java, Built-In;
- **format-number**: XSLT 1.0, XSLT 2.0, XSLT 3.0, C#, C++, Java, Built-In;
- **parse-date**, **parse-dateTime**, **parse-number**, **parse-time**: C#, C++, Java, Built-In.

core | file path functions (ruta de archivos)
Todas las funciones de ruta de archivos son compatibles con todos los lenguajes de transformación.

core | generator functions (generador)
La función **auto-number** está disponible para todos los lenguajes de transformación.

core | logical functions (lógica)
Las funciones de lógica son compatibles con todos los lenguajes de transformación.

core | math functions (matemáticas)

- **add**, **ceiling**, **divide**, **floor**, **modulus**, **multiply**, **round**, **subtract**: todos los lenguajes de transformación;
- **round-precision**: C#, C++, Java, Built-In.

core | node functions (nodo)

- **is-xsi-nil**, **local-name**, **static-node-annotation**, **static-node-name**: todos los lenguajes de transformación;
- **node-name**, **set-xsi-nil**, **substitute-missing-with-xsi-nil**: XSLT 2.0, XSLT 3.0, XQuery 1.0, C#, C++, Java, Built-In.

core | QName functions (QName)
Las funciones Qname son compatibles con todos los lenguajes de transformación excepto XSLT1.0.

core | sequence functions (secuencia)

- **exists**, **not-exists**, **position**, **substitute-missing**: todos los lenguajes de transformación;
- **distinct-values**, **first-items**, **generate-sequence**, **item-at**, **items-from-till**, **last-items**, **replicate-item**, **replicate-sequence**, **set-empty**, **skip-first-items**: XSLT 2.0, XSLT 3.0, XQuery 1.0, C#, C++, Java, Built-In;
- **group-adjacent**, **group-by**, **group-ending-with**, **group-into-blocks**, **group-starting-with**: XSLT 2.0, XSLT 3.0, C#, C++, Java, Built-In.

core | string functions (cadena)

- **concat**, **contains**, **normalize-space**, **starts-with**, **string-length**, **substring**, **substring-after**, **substring-before**, **translate**: todos los lenguajes de transformación;
- **char-from-code**, **code-from-char**, **tokenize**, **tokenize-by-length**, **tokenize-regexp**: XSLT 2.0, XSLT 3.0, XQuery 1.0, C#, C++, Java, Built-In.

Funciones bson (solo en MapForce Enterprise)

Las funciones BSON son solamente compatibles con Built-In.

Funciones db (ediciones MapForce Professional y Enterprise)

Las funciones `db` son compatibles con C#, C++, Java y Built-In.

Funciones edifact (solo en MapForce Enterprise Edition)

Las funciones `edifact` son compatibles con C#, C++, Java y Built-In.

Funciones lang (ediciones MapForce Professional y Enterprise)

La lista siguiente es un resumen de la compatibilidad de las funciones `lang` con los lenguajes de transformación.

lang | datetime functions (fechaHora)

Las funciones `lang | datetime` son compatibles con C#, C++, Java y Built-In.

lang | datetime functions (fechaHora)

Las funciones**read-binary-file** y **write-binary-file** solo son compatibles con Built-In.

lang | generator functions (generador)

Las funciones **create-guid** son compatibles con C#, C++, Java y Built-In.

lang | logical functions (lógica)

Las funciones `lang | logical` son compatibles con C#, C++, Java y Built-In.

lang | math functions (matemáticas)

Las funciones `lang | math` son compatibles con C#, C++, Java y Built-In.

lang | QName functions (QName)

Las funciones `lang | QName` son compatibles con C#, C++, Java y Built-In.

© 2016-2022 Altova GmbH

como función de MapForce (en lang | QName functions)

- **charset-decode**, **charset-encode**: BUILT-IN
- **match-pattern**: C#, Java, Built-In.
- **capitalize**, **count-substring**, **empty**, **find-substring**, **format-guid-string**, **left**, **left-trim**, **lowercase**, **pad-string-left**, **pad-string-right**, **repeat-string**, **replace**, **reversefind-substring**, **right**, **right-trim**, **string-compare**, **string-compare-ignore-case**, **uppercase**: C#, C++, Java, Built-In.

Funciones functions (solo en MapForce Enterprise)

Las funciones**mime** solo son compatibles con Built-In.

Funciones xbrl (solo en MapForce Enterprise)

Las funciones `xbrl` son compatibles con C#, C++, Java y Built-In.

Funciones xlsx (solo en MapForce Enterprise)

Las funciones `xlsx` son compatibles con XSLT 2.0, XSLT 3.0, C#, Java y Built-In.

Funciones xpath2

Todas las funciones `xpath2` son compatibles con XSLT 2.0, XSLT 3.0 y XQuery 1.0.

Funciones xpath3

Las funciones `xpath3` son solamente compatibles con XSLT 3.0.

Funciones xslt10

La lista siguiente es un resumen de la compatibilidad de las funciones `xslt10` con los lenguajes de transformación.
xslt10 | xpath functions

- **local-name**, **name**, **namespace-uri**: XSLT 1.0, XSLT 2.0 y XSLT 3.0
- **lang**, **last**, **position**: XSLT 1.0

xslt10 | xslt functions

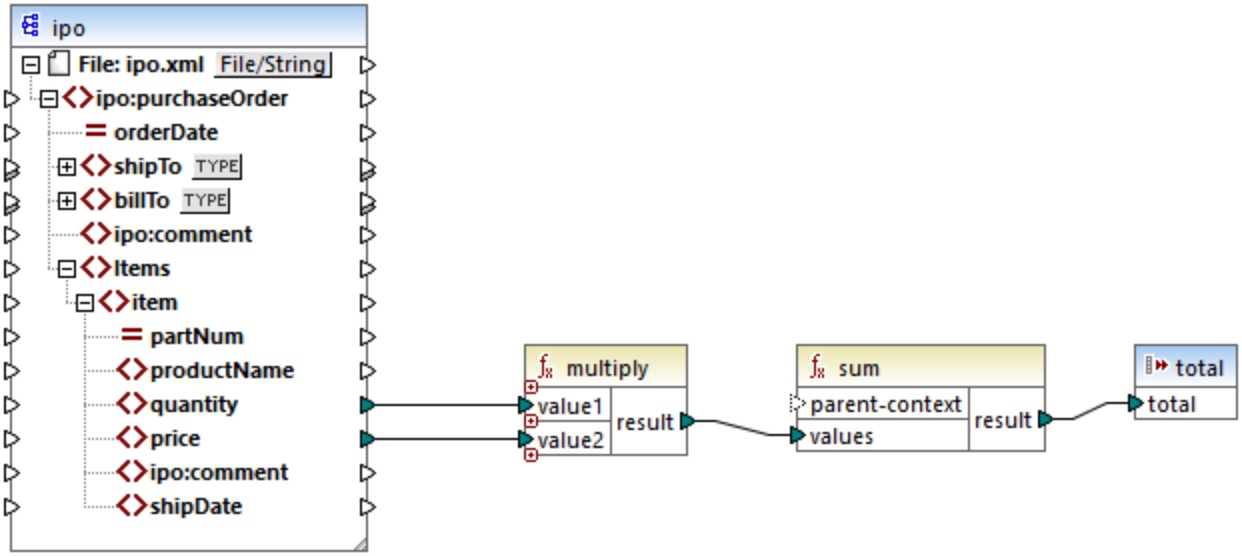
- **generate-id**, **system-property**: XSLT 1.0, XSLT 2.0 y XSLT 3.0
- **current**, **document**, **element-available**, **function-available**, **unparsed-entity-uri**: XSLT 1.0

6.7.1 core | aggregate functions (agregado)

Las funciones de agregado procesan varios valores del mismo tipo para obtener un resultado único, como una suma, un recuento o un promedio. Puede agregar datos en MapForce con ayuda de las funciones de agregado, como `avg`, `count`, `max`, etc.

Estos dos argumentos son comunes a todas las funciones de agregado.

1. **parent-context**. Este argumento es opcional y permite sobrescribir el contexto de asignación predeterminado (y con ello cambiar el alcance de la función o los valores que esta debe recorrer). Para ver un ejemplo, consulte Ejemplo: Cambiar el contexto primario 879.
2. **values**. Este argumento debe estar conectado a un elemento de entrada que suministra los datos que se van a procesar. Por ejemplo, en la asignación siguiente la función `sum` toma como entrada una secuencia de valores numéricos que proviene de un archivo XML de origen. La función `multiply` obtiene el precio multiplicado por la cantidad por cada elemento que hay en el archivo XML de origen y pasa el resultado a la función `sum`. La función `sum` agrega todos los valores de entrada y produce un resultado total que también es el resultado de la asignación. Puede encontrar esta asignación en el directorio **...\MapForceExamples\Tutorial**.

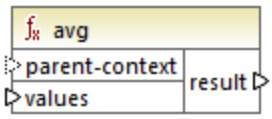


SimpleTotal.mfd

Algunas funciones de agregado, como `min`, `max`, `sum` o `avg`, solo funcionan con valores numéricos. Los datos de entrada de estas funciones se convierten en tipo **decimal** para el procesamiento.

6.7.1.1 avg

Devuelve el valor promedio de todos los valores de la secuencia de entrada. El promedio de un conjunto de valores vacío es un conjunto vacío.



Lenguajes

Built-in, C++, C#, Java, XQuery, XSLT 2.0, XSLT 3.0.

 © 2016-2022 Altova GmbH

Parámetros

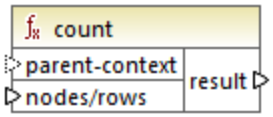
Argumento	Descripción
parent-context	Argumento opcional. Suministra el parent-context. Véase también el apartado Ejemplo: Cambiar el contexto primario 879.
values	Este argumento debe estar conectado a un elemento de entrada que suministra los datos propiamente dichos. Recuerde que el valor de argumento dado debe ser numérico.

Ejemplo

Consulte el apartado Ejemplo: agrupar registros por clave 514.

6.7.1.2 count

Devuelve el número de elementos que componen la secuencia de entrada. El recuento de un conjunto vacío es 0.



Lenguajes

Built-in, C++, C#, Java, XQuery#, XSLT 1.0 y, XSLT 1.0, XSLT 2.0, XSLT 3.0.

Esta función no es totalmente compatible con XSLT1.

Parámetros

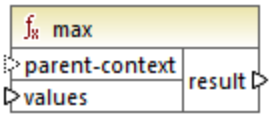
Argumento	Descripción
parent-context	Argumento opcional. Suministra el parent-context. Véase también el apartado Ejemplo: Cambiar el contexto primario 879.
nodos/filas	Este argumento debe estar conectado al elemento de origen que se debe contar.

Ejemplo

Consulte el apartado Ejemplo: Cambiar el contexto primario 879, Ejemplo: contar filas de tabla de BD 500.

6.7.1.3 max

Devuelve el valor máximo de todos los valores numéricos de la secuencia de entrada. El valor máximo de un conjunto de valores vacío es un conjunto vacío.



Lenguajes

Built-in, C++, C#, Java, XQuery, XSLT 2.0, XSLT 3.0.

Parámetros

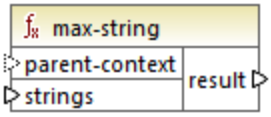
Argumento	Descripción
parent-context	Argumento opcional. Suministra el parent-context. Véase también el apartado Ejemplo: Cambiar el contexto primario 879.
values	Este argumento debe estar conectado a un elemento de entrada que suministra los datos propiamente dichos. Recuerde que el valor de argumento dado debe ser numérico. Para obtener el valor máximo de una secuencia de cadenas, use la función max-string 636 .

Ejemplo

Consulte el apartado Ejemplo: agrupar registros por clave 514.

6.7.1.4 max-string

Devuelve el valor máximo de todos los valores de cadena de la secuencia de entrada. Por ejemplo: `max-string("a", "b", "c")` devuelve `"c"`. Cuando se aplica a un conjunto vacío el resultado es una cadena vacía.



Lenguajes

Built-in, C++, C#, Java, XQuery, XSLT 2.0, XSLT 3.0.

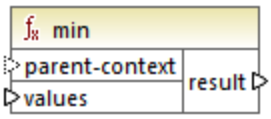
© 2016-2022 Altova GmbH

Parámetros

Argumento	Descripción
parent-context	Argumento opcional. Suministra el parent-context. Véase también el apartado Ejemplo: Cambiar el contexto primario 879.
strings	Este argumento debe estar conectado a un elemento de entrada que suministra los datos propiamente dichos. El valor de argumento dado debe ser una secuencia (cero o varios) de `xs:string`.

6.7.1.5 min

Devuelve el valor mínimo de todos los valores numéricos de la secuencia de entrada. El promedio de un conjunto de valores vacío es un conjunto vacío.



Lenguajes

Built-in, C++, C#, Java, XQuery, XSLT 2.0, XSLT 3.0.

Parámetros

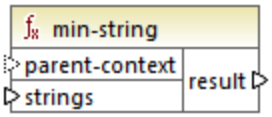
Argumento	Descripción
parent-context	Argumento opcional. Suministra el parent-context. Véase también el apartado Ejemplo: Cambiar el contexto primario 879.
values	Este argumento debe estar conectado a un elemento de entrada que suministra los datos propiamente dichos. Recuerde que el valor de argumento dado debe ser numérico. Para obtener el valor mínimo de un conjunto de cadenas, use la función min-string 638 .

Ejemplo

Consulte el apartado Ejemplo: agrupar registros por clave 514.

6.7.1.6 min-string

Devuelve el valor mínimo de todos los valores de cadena de la secuencia de entrada. Por ejemplo: `min-string("a", "b", "c")` devuelve `"a"`. Cuando se aplica a un conjunto vacío el resultado es una cadena vacía.



Lenguajes

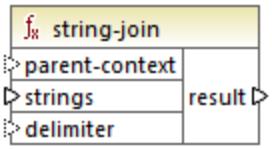
Built-in, C++, C#, Java, XQuery, XSLT 2.0, XSLT 3.0.

Parámetros

Argumento	Descripción
parent-context	Argumento opcional. Suministra el parent-context. Véase también el apartado Ejemplo: Cambiar el contexto primario 879.
strings	Este argumento debe estar conectado a un elemento de entrada que suministra los datos propiamente dichos. El valor de argumento dado debe ser una secuencia (cero o varios) de `xs:string`.

6.7.1.7 string-join

Enlaza todos los valores de la secuencia de entrada y forma una cadena delimitada por la cadena suministrada como delimitador al parámetro delimiter Cuando se aplica a un conjunto vacío el resultado es una cadena vacía.



Lenguajes

Built-in, C++, C#, Java, XQuery, XSLT 2.0, XSLT 3.0.

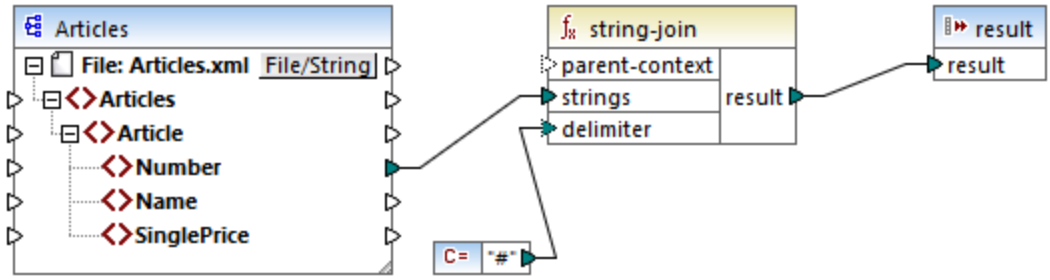
© 2016-2022 Altova GmbH

Parámetros

Argumento	Descripción
parent-context	Argumento opcional. Suministra el parent-context. Véase también el apartado Ejemplo: Cambiar el contexto primario 879.
strings	Este argumento debe estar conectado a un elemento de entrada que suministra los datos propiamente dichos. El valor de argumento dado debe ser una secuencia (cero o varios) de `xs:string`.
delimiter	Argumento opcional. Indica el delimitador que se introduce entre dos cadenas consecutivas.

Ejemplo

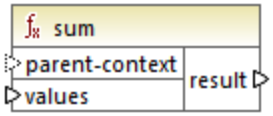
El ejemplo de la siguiente imagen muestra cuatro elementos **Article** con los números: 1, 2, 3, y 4.



La constante indica el carácter "#" como delimitador. El resultado de la asignación es, por tanto, **1#2#3#4**. Si no indica ningún delimitador el resultado sería **1234**.

6.7.1.8 sum

Devuelve la suma aritmética de todos los valores de la secuencia de entrada. La suma de un conjunto vacío es 0.



Lenguajes

Built-in, C++, C#, Java, XQuery#, XSLT 1.0 y, XSLT 1.0, XSLT 2.0, XSLT 3.0.

Parámetros

Argumento	Descripción
parent-context	Argumento opcional. Suministra el parent-context. Véase también el apartado Ejemplo: Cambiar el contexto primario 879.
values	Este argumento debe estar conectado a un elemento de entrada que suministra los datos propiamente dichos. Recuerde que el valor de argumento dado debe ser numérico.

Ejemplo

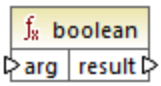
Consulte el apartado Ejemplo: sumar valores de nodo 597.

6.7.2 core | conversion functions (conversión)

MapForce ofrece varias funciones de conversión de tipos en la biblioteca **conversion** para realizar conversiones explícitas de tipos de datos. Tenga en cuenta que, en la mayoría de los casos, MapForce crea conversiones automáticamente y que solamente necesitará usar estas funciones en casos muy concretos. Si los nodos de entrada son de tipos diferentes (p. ej. entero y cadena), puede usar las funciones de conversión para forzar una comparación de cadenas o numérica. Por ejemplo, si en una asignación hay elementos de tipos distintos (como entero y cadena) puede usar la función de conversión number 649 para forzar una comparación numérica.

6.7.2.1 boolean

Convierte un valor numérico de entrada **arg** en un valor booleano. Esta función puede ser útil al trabajar con funciones lógicas (como `equal`, `greater`, etc.) además de con filtros y condiciones if-else 543. Para obtener un valor booleano**false** debe dar una cadena vacía o un valor numérico 0 como argumento. Para obtener un valor booleano **true** debe dar una cadena con contenido o un valor numérico 1 como argumento.



Lenguajes

Built-in, C++, C#, Java, XQuery#, XSLT 1.0 y, XSLT 1.0, XSLT 2.0, XSLT 3.0.

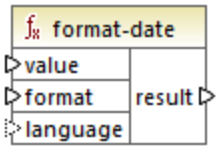
Parámetros

Argumento	Descripción
arg	Argumento obligatorio. Indica el valor que se va a convertir.

© 2016-2022 Altova GmbH

6.7.2.2 format-date

Convierte un valor de entrada `xs:date` en un valor de cadena y le aplica el formato indicado por las opciones especificadas.



Lenguajes

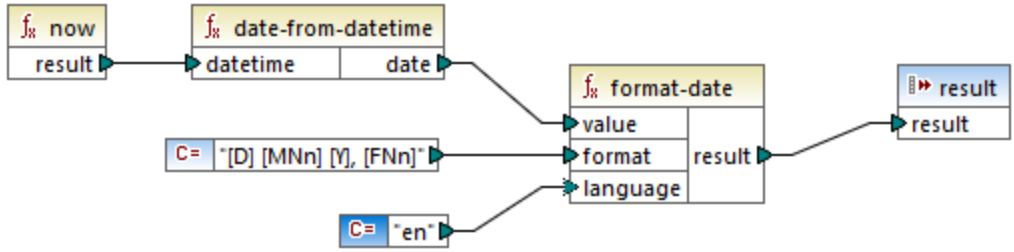
La función está disponible para XSLT 2.0, XSLT 3.0, Java, C#, C++ y el motor de ejecución integrado.

Parámetros

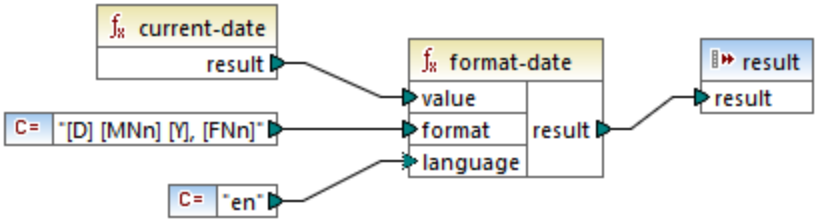
Argumento	Descripción
value	Fecha a la que se debe aplicar formato.
format	Cadena de formato que identifica el formato que se debe aplicar a la fecha. Este argumento se usa igual que el argumento **format** de la format-dateTime 642 .
language	Argumento opcional. Devuelve el nombre del mes y el día de la semana en el idioma seleccionado. Son valores válidos: **de** alemán **en (predet.)** inglés **es** español **fr** francés **ja** japonés

Ejemplo

En la asignación siguiente el resultado es la fecha actual en formato: "25 March 2020, Wednesday". Para traducir este valor al idioma español, defina el valor **es** para el argumento **language**.

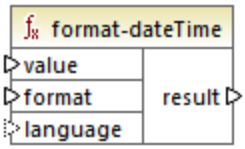


Tenga en cuenta que la asignación anterior se ha diseñado para Java, C#, C++ y el motor de ejecución integrado. Para obtener el mismo resultado en XSLT 2.0 use esta asignación:



6.7.2.3 format-dateTime

Convierte un valor de fecha y hora (`xs:dateTime`) en una cadena de texto. La representación de cadena de la fecha y hora sigue el formato que dicte el valor del argumento **format**



Lenguajes

La función está disponible para XSLT 2.0, XSLT 3.0, Java, C#, C++ y el motor de ejecución integrado.

Parámetros

Argumento	Descripción
value	El valor `xs:dateTime` al que se debe aplicar formato.
format	Cadena de formato que identifica el formato que se debe aplicar al **valor**. Véanse las "Observaciones", más abajo
language	Argumento opcional. Devuelve el nombre del mes y el día de la semana en el idioma seleccionado. Son valores válidos:

 © 2016-2022 Altova GmbH

Argumento	Descripción
	de alemán **en (predet.)** inglés **es** español **fr** francés **ja** japonés

Nota: si el resultado de la función (result) se conecta a un nodo de un tipo que no sea string, el formato se puede perder porque el valor se convierte al tipo de destino. Esta conversión automática se puede deshabilitar desactivando la casilla **Convertir valores en tipos de destino** en el cuadro de diálogo "Configuración" del componente de destino (véase Cambiar la configuración 79 de los componentes.

Observaciones

El argumento **format** está compuesto por una cadena que lleva los llamados marcadores de variable entre corchetes. Los caracteres situados fuera de los corchetes son caracteres literales que se deben copiar en el resultado. Si necesita usar corchetes como caracteres literales en el resultado, escríbalos dos veces.

Cada marcador de variable está compuesto por (i) un especificador de componente que identifica qué componente de fecha u hora se debe mostrar, (ii) un modificador de formato opcional, (iii)otro modificador de presentación opcional y (iv) un modificador de ancho opcional precedido de una coma, si existe.

```
format := (literal | argument)*
argument := [component(format)?(presentation)?(width)?]
width := , min-width ("-" max-width)?
```

Estos son los componentes:

Especificador	Descripción	Presentación predeterminada
Y	año (valor absoluto)	cuatro dígitos (2010)
M	mes del año	1-12
D	día del mes	1-31
d	día del año	1-366
F	día de la semana	nombre del día (dependiendo del idioma)
W	semana del año	1-53
w	semana del mes	1-5
H	hora (24 horas)	0-23

Especificador	Descripción	Presentación predeterminada
h	hora (12 horas)	1-12
P	A.M. o P.M.	alfabética (dependiendo del idioma)
m	minutos de una hora	00-59
s	segundos de un minuto	00-59
f	segundos fraccionarios	numérica, con un decimal
Z	uso horario como diferencia horaria de UTC	+08:00
z	uso horario como diferencia horaria usando GMT	GMT+n

El modificador de formato puede ser uno de estos:

Carácter	Descripción	Ejemplo
1	formato decimal numérico sin ceros iniciales: 1, 2, 3, ...	1, 2, 3
01	formato decimal, con dos dígitos: 01, 02, 03, ...	01, 02, 03
N	nombre del componente, todo en mayúsculas	LUNES, MARTES 1)
n	nombre del componente, todo en minúsculas	lunes, martes 1)
Nn	nombre del componente, primera letra en mayúsculas	Lunes, Martes 1)

Notas:

1. los modificadores **N**, **n** y **Nn** solamente son compatibles con estos componentes: **M**, **d**, **D**.

El modificador del ancho, si es necesario, viene introducido por una coma y seguido de un dígito que exprese el ancho mínimo. Opcionalmente puede añadir un guion seguido de otro dígito para expresar el ancho máximo. Por ejemplo:

- **`[D,2]`** es el día del mes, con ceros iniciales (dos dígitos).
- **`[MNn,3-3]`** es el nombre del mes, escrito con tres caracteres, por ejemplo *Ene*, *Feb*, *Mar*, etc.

Ejemplos

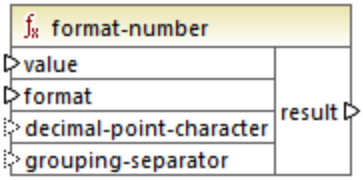
La tabla siguiente muestra algunos ejemplos de formato de valores `xs:dateTime`, obtenido con ayuda de la función **`format-dateTime`**. La columna Valor de la tabla especifica el valor dado al argumento **value**. La columna Formato de la tabla especifica el valor del argumento **format**. La columna Resultado muestra el valor que devuelve la función.

 © 2016-2022 Altova GmbH

Valor	Formato	Resultado
2003-11-03T00:00:00	`[D]/[M]/[Y]`	11/03/2003
2003-11-03T00:00:00	`[Y]-[M,2]-[D,2]`	03/11/2003
2003-11-03T00:00:00	`[Y]-[M,2]-[D,2] [H,2]:[m]:[s]`	2003-11-03 00:00:00
2010-06-02T08:02	`[Y] [MNn] [D01] [F,3-3] [d] [H]:[m]:[s].[f]`	2010 June 02 Wed 153 8:02:12.054
2010-06-02T08:02	`[Y] [MNn] [D01] [F,3-3] [d] [H]:[m]:[s].[f] [z]`	2010 June 02 Wed 153 8:02:12.054 GMT+02:00
2010-06-02T08:02	`[Y] [MNn] [D1] [F] [H]:[m]:[s].[f] [Z]`	2010 June 2 Wednesday 8:02:12.054 +02:00
2010-06-02T08:02	`[Y] [MNn] [D] [F,3-3] [H01]:[m]:[s]`	2010 June 2 Wed 08:02:12

6.7.2.4 format-number

Convierte un número en una cadena y le aplica el formato indicado .en las opciones.



Lenguajes

Built-in, C++, C#, Java#, XSLT 1.0 y, XSLT 1.0, XSLT 2.0, XSLT 3.0.

Parámetros

Argumento	Descripción
value	Argumento obligatorio. Suministra el número al que se debe aplicar formato.
format	Argumento obligatorio. Suministra una cadena de formato que indica de qué forma se debe dar formato al número. Véanse las "Observaciones", más abajo
decimal-point-format	Argumento opcional. Suministra el carácter que se debe usar como carácter de punto decimal. El valor predeterminado es el carácter de punto

Argumento	Descripción
	(.).
grouping-separator	Argumento opcional. Suministra el carácter que se debe usar para separa grupos de números. El valor predeterminado es el carácter de coma (,).

Nota: si el resultado de la función (result) se conecta a un nodo de un tipo que no sea string, el formato se puede perder porque el valor se convierte al tipo de destino. Esta conversión automática se puede deshabilitar desactivando la casilla **Convertir valores en tipos de destino** en el cuadro de diálogo "Configuración" del componente de destino (véase Cambiar la configuración 79 de los componentes.

Observaciones

El argumento **format** toma esta forma:

```
format := subformat (;subformat)?
 subformat := (prefix)? integer (.fraction)? (suffix)?
 prefix := any characters except special characters
 suffix := any characters except special characters
 integer := (#)* (0)* ( allowing ',' to appear)
 fraction := (0)* (#)* (allowing ',' to appear)
```

El primer *subformat* se utiliza para dar formato a los números positivos y el segundo para los números negativos. Si solamente se da un *subformat*, se utiliza el mismo para los números negativos, pero con un signo - antes del *prefix*.

Carácter especial	Predeterminado	Descripción
dígito cero	0	en este punto del resultado siempre aparecerá un dígito
dígito	#	en este punto del resultado siempre aparecerá un dígito, excepto si es un cero inicial o final no significativo
punto decimal	.	separa el entero y la parte de fracción del número
grouping-separator	,	separa grupos de dígitos
signo de porcentaje	%	multiplica el número por 100 y lo muestra como porcentaje
por mil	‰	multiplica el número por 1000 y lo muestra como por mil

Los caracteres utilizados por el carácter de punto decimal y de separador de grupos siempre son "." y "," respectivamente.

Nota: el método de redondeo utilizado para esta función es al alza (p. ej. redondea al alza si la fracción es mayor o igual a 0.5 y redondea a la baja si la fracción es menor que 0.5). Este método de redondeo solamente afecta al código generado y si la opción seleccionada es el motor de ejecución integrado.

 © 2016-2022 Altova GmbH

El XSLT 1.0 el modo de redondeo es indeterminado. En XSLT 2.0 el modo de redondeo es round-half-to-even, es decir se redondea al número par más próximo.

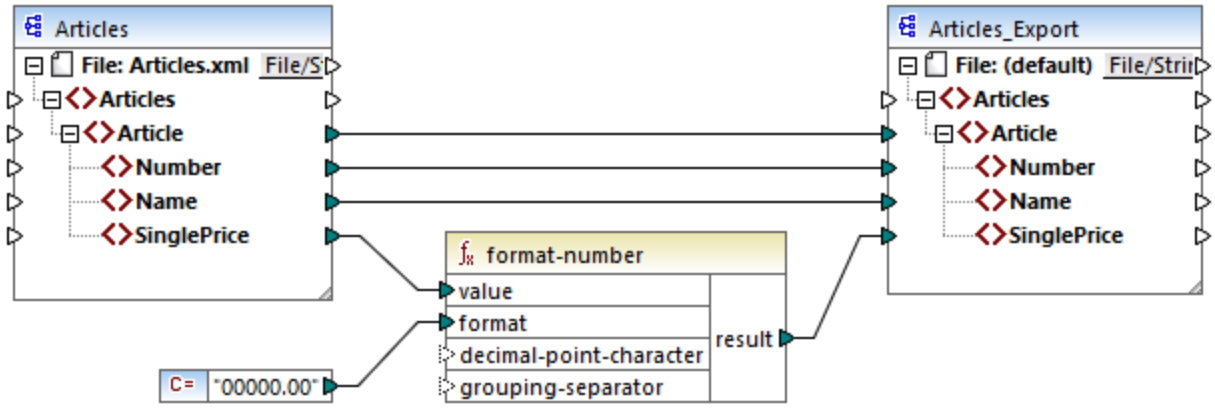
Número	Cadena de formato	Resultado
1234,5	#,##0.00	1.234,50
123,456	#,##0.00	123,46
1000000	#,##0.00	1.000.000,00
-59	#,##0.00	-59,00
1234	###0.0###	1234,0
1234,5	###0.0###	1234,5
.00025	###0.0###	0,0003
.00035	###0.0###	0,0004
0,25	#00%	25%
0,736	#00%	74%
1	#00%	100%
-42	#00%	-4200%
-3,12	#.00;(#.00)	(3.12)
-3,12	#.00;#.00CR	3.12CR

Ejemplo

La asignación siguiente lee datos de un archivo XML de origen y los escribe en un archivo XML de destino. Existen varios elementos **SinglePrice**en el archivo de origen que contienen estos valores decimales: **25**, **2.30**, **34**, **57.50**. La asignación tiene dos objetivos:

1. Añadir ceros a la izquierda de todos los valores de forma que la parte entera ocupe exactamente 5 dígitos
2. Añadir ceros a la derecha de todos los valores de forma que la parte decimal ocupe exactamente 2 dígitos

Para ello se da la cadena de formato `00000.00` como argumento a la función `format-number`.



PreserveFormatting.mfd

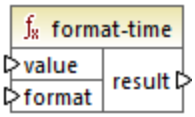
En consecuencia, los valores de destino paran a ser:

```
00025,00
00002,30
00034,00
00057,50
```

Puede encontrar el archivo de diseño de esta asignación en la ruta:
<Documentos>\Altova\MapForce2023\MapForceExamples\PreserveFormatting.mfd.

6.7.2.5 format-time

Convierte un valor `xs:time` de entrada en una cadena.



Lenguajes

La función está disponible para XSLT 2.0, XSLT 3.0, Java, C#, C++ y el motor de ejecución integrado.

Parámetros

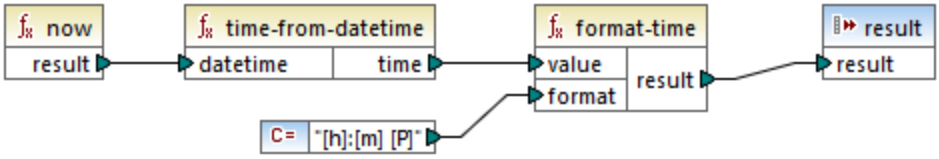
Argumento	Descripción
value	Argumento obligatorio. Suministra el valor `xs:time` para darle formato.
format	Argumento obligatorio. Indica el formato de la cadena. Este argumento se usa igual que el argumento **format** de la format-dateTime 642 .

 © 2016-2022 Altova GmbH

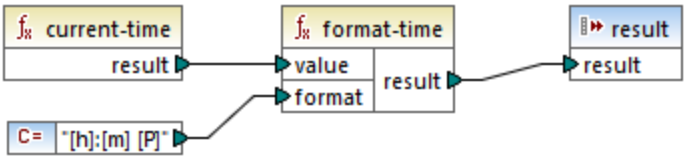
Ejemplo

En la asignación siguiente el resultado es la hora actual en formato `2:15 p.m.` . Para ello se usa la cadena de formato `[h]:[m] [P]`, donde:

- **[h]** es la hora actual en formato 12 horas
- **[m]** es el minuto de la hora actual
- **[P]** indica si es "a.m." o "p.m."

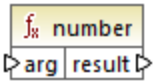


Tenga en cuenta que la asignación anterior se ha diseñado para Java, C#, C++ y el motor de ejecución integrado. Para obtener el mismo resultado en XSLT 2.0 use esta asignación:



6.7.2.6 number

Convierte el valor de **arg** en un número, donde **arg** es una cadena de valor booleano. Si **arg** es una cadena, MapForce intentará analizarlo como número. Por ejemplo, una cadena como `"12.56"` se convierte en el valor decimal `12.56`. Si **arg** tiene el valor booleano **true**, se convierte en el valor numérico **1**. Si **arg** tiene el valor booleano **false** , se convierte en el valor numérico **0**.



Lenguajes

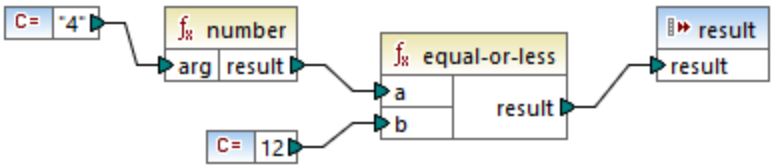
Built-in, C++, C#, Java, XQuery#, XSLT 1.0 y, XSLT 1.0, XSLT 2.0, XSLT 3.0.

Parámetros

Argumento	Descripción
arg	Argumento obligatorio. Indica el valor que se va a convertir.

Ejemplo

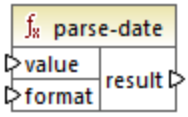
En el ejemplo siguiente la primera constante es de tipo `string` y contiene la cadena "4". La segunda constante contiene la constante numérica 12. Para que los dos valores se puedan comparar como números deben ser del mismo tipo.



Si añade una función **`number`** a la primera constante, la cadena "4" se convierte en el valor numérico 4. Es decir, que el resultado de la comparación es "true". Si no se usara la función **`number`** (es decir, si "4" se conectara directamente a **a**), se llevaría a cabo una comparación de cadenas y el resultado sería "false".

6.7.2.7 parse-date

Convierte una cadena en una fecha. Esta función se basa en la función parse-dateTime 651 e ignora el componente time. El resultado es de tipo `xs:date`.



Lenguajes

Disponible para Java, C#, C++ y el motor de ejecución integrado.

Parámetros

Argumento	Descripción
value	Argumento obligatorio. Indica el valor de la cadena que se va a convertir.
format	Argumento obligatorio. Indica el formato de la cadena. Este argumento se usa igual que el argumento **format** de la parse-dateTime 651 .

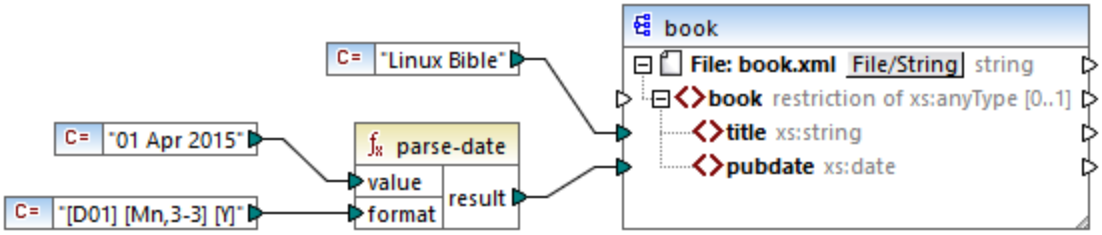
Ejemplo

La asignación siguiente analiza la cadena **`"01 Apr 2015"`**,, la convierte en una fecha y escribe el resultado en un componente de destino (**pubdate**) de tipo `xs:date`. Para ello se usó el formato **`[D01] [MNn,3-3] [Y]`**, donde:

- **`[D01]`** es el día del mes, expresado en dos dígitos
- **`[MNn,3-3]`** es el nombre del mes, con un ancho mínimo y un máximo de 3 caracteres

 © 2016-2022 Altova GmbH

- **[Y]** es el año

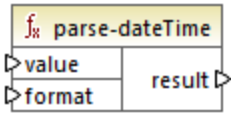


El resultado es el siguiente (sin incluir las declaraciones XML y de espacio de nombres):

```
<book>
 <title>Linux Bible</title>
 <pubdate>2015-04-01</pubdate>
</book>
```

6.7.2.8 parse-dateTime

Convierte un valor de fecha/hora expresado en forma de cadena en un valor de tipo `xs:dateTime`.



Lenguajes

Disponible para Java, C#, C++ y el motor de ejecución integrado.

Parámetros

Argumento	Descripción
value	El valor de cadena que se debe convertir.
format	La máscara de formato que se debe aplicar al argumento **value**.

Observaciones

Una máscara de formato puede estar compuesta por estos componentes:

Componente	Descripción	Presentación predeterminada
Y	año (valor absoluto)	cuatro dígitos (2010)

Componente	Descripción	Presentación predeterminada
M	mes del año	1-12
D	día del mes	1-31
d	día del año	1-366
H	hora (24 horas)	0-23
h	hora (12 horas)	1-12
P	A.M. o P.M.	alfabética (dependiendo del idioma)
m	minutos de una hora	00-59
s	segundos de un minuto	00-59
f	segundos fraccionarios	numérica, con un decimal
Z	uso horario como diferencia horaria de UTC	+08:00
z	uso horario como diferencia horaria usando GMT	GMT+n

Algunos de estos componentes toman modificadores (p. ej. se pueden usar para interpretar una fecha como fecha de un solo dígito o de dos dígitos):

Modificador	Descripción	Ejemplo
1	formato decimal numérico sin ceros iniciales: 1, 2, 3, ...	1, 2, 3
01	formato decimal, con dos dígitos: 01, 02, 03, ...	01, 02, 03
N	nombre del componente, todo en mayúsculas	FEBRERO, MARZO
n	nombre del componente, todo en minúsculas	febrero, marzo
Nn	nombre del componente, primera letra en mayúsculas	Febrero, Marzo

Nota: los modificadores *N*, *n* y *Nn* solo admiten el componente *M* (mes).

El modificador del ancho, si es necesario, viene introducido por una coma y seguido de un dígito que exprese el ancho mínimo. Opcionalmente puede añadir un guion seguido de otro dígito para expresar el ancho máximo. Por ejemplo:

- **`[D,2]`** es el día del mes, con ceros iniciales (dos dígitos).
- **`[MNn,3-3]`** es el nombre del mes, escrito con tres caracteres, por ejemplo *Ene*, *Feb*, *Mar*, etc.

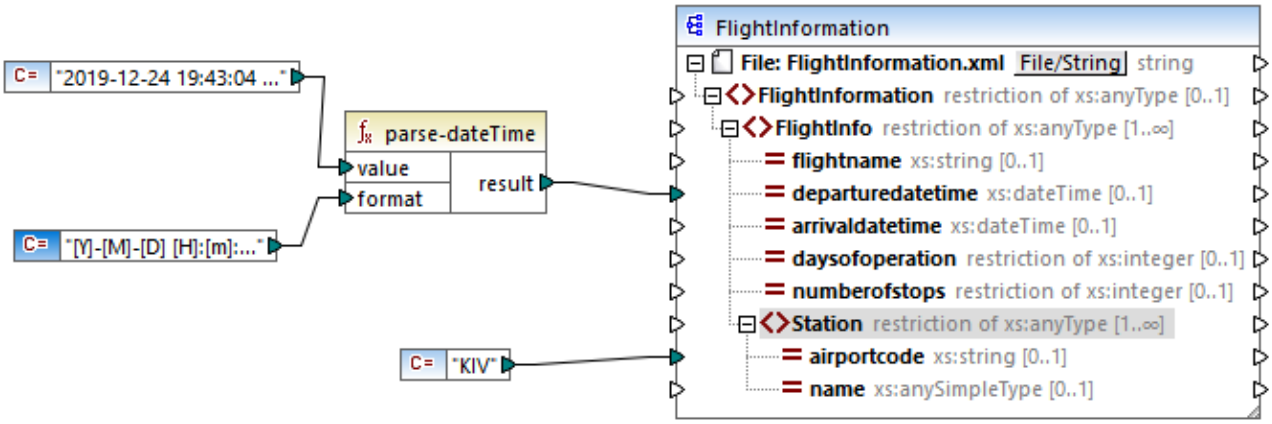
A continuación puede ver algunos ejemplos más:

 © 2016-2022 Altova GmbH

Valor	Formato	Resultado
21-03-2002 16:21:12.492 GMT+02:00	`[D]-[M]-[Y] [H]:[m]:[s].[f] [z]`	2002-03-21T16:21:12.492+02:00
315 2004 +01:00	`[d] [Y] [Z]`	2004-11-10T00:00:00+01:00
1.Diciembre.10 03:2:39 p.m. +01:00	`[D].[MNn].[Y,2-2] [h]:[m]:[s] [P] [Z]`	2010-12-01T15:2:39+01:00
20110620	`[Y,4-4][M,2-2][D,2-2]`	2011-06-20T00:00:00

Ejemplo

En la asignación siguiente, el valor de cadena **`2019-12-24 19:43:04 +02:00`** se convierte en su equivalente `dateTime`; para ello se aplica la máscara de formato **`[Y]-[M]-[D] [H]:[m]:[s] [Z]`**.

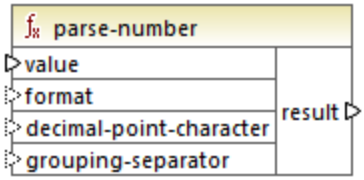


El resultado es el siguiente (sin incluir las declaraciones XML y de espacio de nombres):

```
<FlightInformation>
 <FlightInfo departuredatetime="2019-12-24T19:43:04+02:00">
 <Station airportcode="KIV"/>
 </FlightInfo>
</FlightInformation>
```

6.7.2.9 parse-number

Convierte una cadena de entrada en un número decimal conforme al formato indicado. La función usa este formato para determinar el prefijo, el sufijo y la agrupación de dígitos. La longitud actual del número no se compara con el formato. Si el valor de entrada es más largo o más corto que el indicado en el formato, la comprobación lo ignora.



Lenguajes

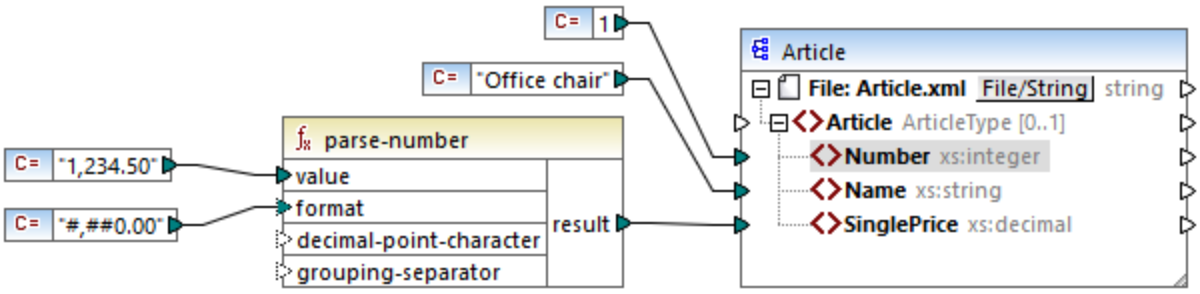
Disponible para Java, C#, C++ y el motor de ejecución integrado.

Parámetros

Argumento	Descripción
value	La cadena que se debe convertir en número.
format	Argumento opcional. Una cadena de formato que identifica el formato actual del número. El formato de la cadena es el mismo que el que se usa en format-number 645. El formato predeterminado es "#,##0.#"
decimal-point-character	Argumento opcional. Suministra el carácter que se debe usar como carácter de punto decimal. El carácter predeterminado es "."
grouping-separator	Argumento opcional. El separador/delimitador utilizado para separar grupos de números. El carácter predeterminado es "."

Ejemplo

Esta asignación convierte el valor de la cadena **`"1,234.50"`** en un equivalente decimal, para lo que usa la máscara de formato **`#,##0.00`**. En esta asignación no es necesario conectar los argumentos **decimal-point-character** y **grouping-separator** porque sus valores predeterminados coinciden con el formato de la cadena de entrada.



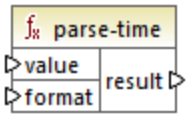
El resultado de la asignación es el siguiente (sin incluir las declaraciones XML y de espacio de nombres):

 © 2016-2022 Altova GmbH

```
<Article>
   <Number>1</Number>
   <Name>Office chair</Name>
   <SinglePrice>1234.5</SinglePrice>
</Article>
```

6.7.2.10 parse-time

Convierte una cadena en un valor `xs:time`. Esta función se basa en la función parse-dateTime 651 e ignora el componente fecha.



Lenguajes

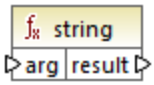
Disponible para Java, C#, C++ y el motor de ejecución integrado.

Parámetros

Argumento	Descripción
value	Argumento obligatorio. Indica el valor de la cadena que se va a convertir.
format	Argumento obligatorio. Indica el formato de la cadena. Este argumento se usa igual que el argumento **format** de la parse-dateTime 651 .

6.7.2.11 string

Convierte un valor de entrada en una cadena. La función también se puede usar para recuperar el contenido de texto de un nodo. Si el nodo de entrada es un tipo complejo XML, todos los descendientes también se generan como una sola cadena.



Lenguajes

Built-in, C++, C#, Java, XQuery#, XSLT 1.0 y, XSLT 1.0, XSLT 2.0, XSLT 3.0.

Parámetros

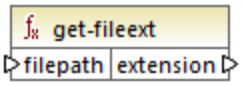
Argumento	Descripción
arg	Argumento obligatorio. Indica el valor que se va a convertir.

6.7.3 core | file path functions (ruta de archivos)

Las funciones de la biblioteca **file path** sirven para acceder a datos de una ruta de acceso (como carpetas, nombres de archivos y extensiones) y manipularlos para después procesarlos en la asignación. Estas funciones están disponibles para todos los lenguajes compatibles con MapForce.

6.7.3.1 get-fileext

Devuelve la extensión de la ruta de archivo, incluido el carácter "."



Lenguajes

Built-in, C++, C#, Java, XQuery#, XSLT 1.0 y, XSLT 1.0, XSLT 2.0, XSLT 3.0.

Parámetros

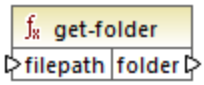
Argumento	Descripción
filepath	Argumento obligatorio. Suministra la ruta de acceso al archivo que se quiere procesar.

Ejemplo

P. ej. c:\data\Sample.mfd devuelve ".mfd"

6.7.3.2 get-folder

Devuelve el nombre de la carpeta de la ruta de archivo, incluida la barra diagonal final o la barra diagonal inversa final.



 © 2016-2022 Altova GmbH

Lenguajes

Built-in, C++, C#, Java, XQuery#, XSLT 1.0 y, XSLT 1.0, XSLT 2.0, XSLT 3.0.

Parámetros

Argumento	Descripción
filepath	Argumento obligatorio. Suministra la ruta de acceso al archivo que se quiere procesar.

Ejemplo

P. ej. c:/data/Sample.mfd devuelve c:/data/Sample.

6.7.3.3 main-mfd-filepath

Devuelve la ruta completa del archivo mfd que contiene la asignación principal. Si el archivo mfd está sin guardar, la función devuelve una cadena vacía.

Lenguajes

Built-in, C++, C#, Java, XQuery#, XSLT 1.0 y, XSLT 1.0, XSLT 2.0, XSLT 3.0.

6.7.3.4 mfd-filepath

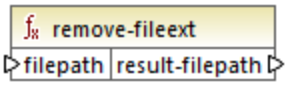
Si se le llama desde la asignación principal, la función devuelve el mismo resultado que la función main-mfd-filepath 657 , es decir, la ruta completa del archivo mfd que contiene la asignación principal. Si el archivo mfd archivo está sin guardar, la función devuelve una cadena vacía. Si se le llama desde dentro de una función definida por el usuario que es *importada* por un archivo mfd, la función mfd-filepath devuelve la ruta completa del archivo mfd *importado* que contiene la definición de la función definida por el usuario.

Lenguajes

Built-in, C++, C#, Java, XQuery#, XSLT 1.0 y, XSLT 1.0, XSLT 2.0, XSLT 3.0.

6.7.3.5 remove-fileext

Borra la extensión de la ruta de archivo, incluido el carácter "."



Lenguajes

Built-in, C++, C#, Java, XQuery#, XSLT 1.0 y, XSLT 1.0, XSLT 2.0, XSLT 3.0.

Parámetros

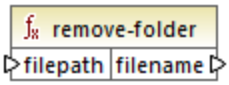
Argumento	Descripción
filepath	Argumento obligatorio. Suministra la ruta de acceso al archivo que se quiere procesar.

Ejemplo

P. ej. c:/data/Sample.mfd devuelve c:/data/Sample.

6.7.3.6 remove-folder

Borra el directorio de la ruta de acceso, incluida la barra diagonal final o la barra diagonal inversa final.



Lenguajes

Built-in, C++, C#, Java, XQuery#, XSLT 1.0 y, XSLT 1.0, XSLT 2.0, XSLT 3.0.

Parámetros

Argumento	Descripción
filepath	Argumento obligatorio. Suministra la ruta de acceso al archivo que se quiere procesar.

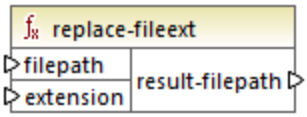
Ejemplo

P. ej. c:\data\Sample.mfd devuelve ".mfd"

© 2016-2022 Altova GmbH

6.7.3.7 replace-fileext

Reemplaza la extensión de la ruta de acceso dada por el parámetro **filepath** con la extensión dada por el parámetro **extension**.



Lenguajes

Built-in, C++, C#, Java, XQuery#, XSLT 1.0 y, XSLT 1.0, XSLT 2.0, XSLT 3.0.

Parámetros

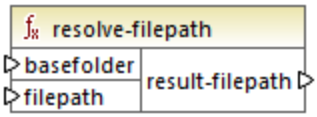
Argumento	Descripción
filepath	Argumento obligatorio. Suministra la ruta de acceso al archivo que se quiere procesar.
extension	Argumento obligatorio. Suministra la estensión nueva que se debe usar.

Ejemplo

P. ej. si el parámetro **filepath** es la ruta de acceso c:/data/Sample.mfd y .mfp es la extensión del parámetro **extension**, la función devuelve `c:\data\Sample.txt`.

6.7.3.8 resolve-filepath

Convierte una ruta de acceso relativa en una carpeta base relativa o absoluta. La función admite "." (directorio actual) y ".." (directorio primario).



Lenguajes

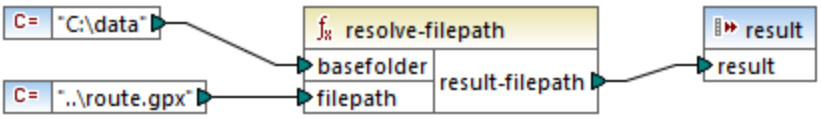
Built-in, C++, C#, Java, XQuery#, XSLT 1.0 y, XSLT 1.0, XSLT 2.0, XSLT 3.0.

Parámetros

Argumento	Descripción
basefolder	Argumento obligatorio. Suministra el directorio base relativo en el que se debe resolver la ruta, que puede ser absoluta o relativa.
filepath	Argumento obligatorio. Suministra la ruta de acceso relativa que se quiere resolver.

Ejemplos

En la imagen siguiente, la ruta relativa **..\route.gpx** se resuelve con respecto al directorio **C:\data**.



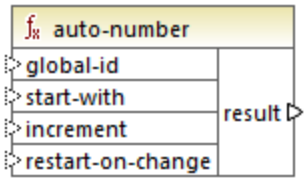
El resultado de la asignación es `C:\route.gpx`.

6.7.4 core | generator functions (generador)

La biblioteca de funciones **core / generator** incluye funciones que generan valores.

6.7.4.1 auto-number

La función auto-number genera enteros en una secuencia (por ejemplo, 1, 2, 3, 4...). Puede usar parámetros para configurar el entero de inicio, el valor incremental y otras opciones.



El orden exacto en el que la asignación llama a las funciones es indefinido. Puede que MapForce necesite guardar en la memoria caché los resultados calculados para volver a utilizarlos o evaluar las expresiones en cualquier orden. Al contrario que otras funciones, la función `auto-number` devuelve un resultado distinto si se la llama varias veces con los mismos parámetros de entrada. Por lo tanto, recomendamos que use esta función con cuidado. En algunos casos se puede conseguir el mismo resultado utilizando la función position 700 en lugar de auto-number.

 © 2016-2022 Altova GmbH

Lenguajes

Built-in, C++, C#, Java, XQuery#, XSLT 1.0 y, XSLT 1.0, XSLT 2.0, XSLT 3.0.

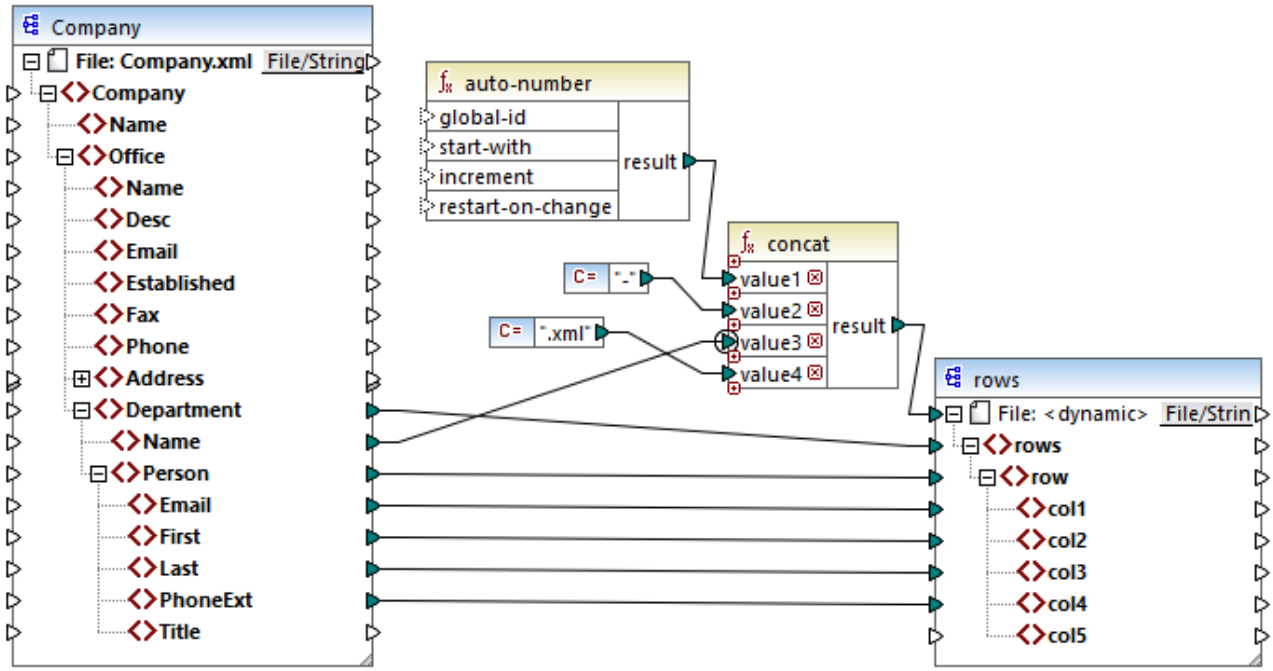
Parámetros

Argumento	Descripción
global-id	Parámetro opcional. Si un diseño de asignación contiene varias funciones `auto-number`, estas generarán secuencias con números duplicados (solapados). Para que todas las funciones `auto-number` sean conscientes unas de otras y no generen secuencias que se solapen, debe conectar una cadena común (por ejemplo, una constante) a la entrada **global-id** de cada función `auto-number`.
start-with	Parámetro opcional. Indica el entero con el que empieza la secuencia que se genera. El valor predeterminado es **1**.
increment	Parámetro opcional. Indica el valor incremental. El valor predeterminado es **1**.
restart-on-change	Parámetro opcional. Restablece el contador a **start-with** cuando cambia el contenido del elemento conectado.

Ejemplo

Esta asignación es una variación de la asignación **ParentContext.mfd** Ejemplo: Cambiar el contexto primario 879.

El objetivo de la asignación siguiente es generar varios archivos XML, uno por cada departamento del archivo XML de origen. Hay varios departamentos con el mismo nombre (esto se debe a que pertenecen a distintas oficinas centrales). Por este motivo, cada nombre de archivo que se genere debe empezar por un número secuencial, por ejemplo **1-Administration.xml**, **2-Marketing.xml**, etc.



Para conseguir el objetivo de la asignación se usó la función **auto-number**. El resultado de esta función se concatena con un guion seguido por el nombre del departamento, seguido por la cadena ".xml" para crear el nombre único del archivo generado. Es importante que se aplique un contexto de prioridad al tercer parámetro de la función **concat** nombre del[883] departamento). Esto sirve para llamar a la función **auto-number** en el contexto de cada departamento y produce los valores secuenciales necesarios. Sin ese contexto de prioridad, la función **auto-number** seguiría generando el número number 1 y, en consecuencia, se generarían también nombres de archivo duplicados.

6.7.5 core | logical functions (lógica)

Las funciones lógicas de la biblioteca core | logical se usan por lo general para comparar datos de entrada y su resultado son los valores booleanos true o false. Estas funciones se suelen utilizar para probar datos antes de pasarlos, por medio de un filtro,[543] a un subconjunto del componente de destino. Casi todas las funciones lógicas tienen esta estructura:

Parámetros de entrada = `a | b` o `value1 | value2`
Parámetro de salida = `result`

El resultado de la evaluación de dos nodos de entrada depende de los valores de entrada así como de los tipos de datos utilizados para la comparación. Por ejemplo, la comparación "less than" de los valores enteros **4** y **12** da como resultado el valor booleano "true" porque 4 es menor que 12. Si las dos cadenas de entrada contienen **"4"** y **"12"**, entonces el análisis léxico da como resultado el valor de salida "false" porque "4" es alfabéticamente mayor que el primer carácter "1" del segundo operando (12).

Si todos los tipos de datos de entrada son del mismo tipo (p. ej. todos los nodos de entrada son de tipo numérico o son cadenas), entonces la comparación se hace para el tipo común. Si los nodos de entrada son

 © 2016-2022 Altova GmbH

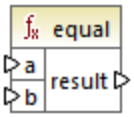
de tipos distintos (p. ej. un `integer` y una `cadena` o una `cadena` y una `fecha`), entonces el tipo de datos utilizado para la comparación es el tipo de datos de entrada más general y menos limitado de los dos.

Antes de iniciarse la comparación de los dos valores, todos los valores de entrada se convierten en el tipo de datos común. Por ejemplo, el tipo de datos "string" es menos limitado que "integer". La comparación del valor entero **4** con la cadena **"12"** convierte el valor entero **4** en la cadena **"4"**, que después se compara con la cadena **"12"**.

Nota: las funciones lógicas no se pueden usar para comprobar la existencia de valores null. Si se da un valor null como argumento de una función lógica, la función devuelve un valor null. Para más información consulte el apartado Valores Nil y Nillable) 131.

6.7.5.1 equal

El resultado es `true` si **a** es igual a **b**. De lo contrario devuelve `false`. La comparación distingue entre mayúsculas y minúsculas.



Ejemplo:

```
a = hi
b = hi
```

En este ejemplo los dos valores son iguales, por lo que el resultado es `true`. Si, por ejemplo, `b` fuera igual que `Hi`, la función devolvería `false`.

Lenguajes

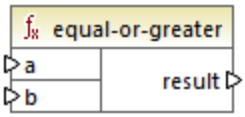
Built-in, C++, C#, Java, XQuery#, XSLT 1.0 y, XSLT 1.0, XSLT 2.0, XSLT 3.0.

Parámetros

Argumento	Descripción
a	Parámetro obligatorio. Suministra el primer valor de la comparación.
b	Parámetro obligatorio. Suministra el segundo valor de la comparación.

6.7.5.2 equal-or-greater

El resultado es **true** si *a* es igual/mayor que *b*. De lo contrario, el resultado es **false**.



Lenguajes

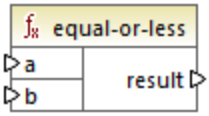
Built-in, C++, C#, Java, XQuery#, XSLT 1.0 y, XSLT 1.0, XSLT 2.0, XSLT 3.0.

Parámetros

Argumento	Descripción
a	Parámetro obligatorio. Suministra el primer valor de la comparación.
b	Parámetro obligatorio. Suministra el segundo valor de la comparación.

6.7.5.3 equal-or-less

El resultado es **true** si *a* es igual/menor que *b*. De lo contrario, el resultado es **false**.



Lenguajes

Built-in, C++, C#, Java, XQuery#, XSLT 1.0 y, XSLT 1.0, XSLT 2.0, XSLT 3.0.

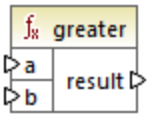
Parámetros

Argumento	Descripción
a	Parámetro obligatorio. Suministra el primer valor de la comparación.
b	Parámetro obligatorio. Suministra el segundo valor de la comparación.

© 2016-2022 Altova GmbH

6.7.5.4 greater

El resultado es **true** si *a* es mayor que *b*. De lo contrario devuelve **false**.



Lenguajes

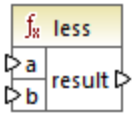
Built-in, C++, C#, Java, XQuery#, XSLT 1.0 y, XSLT 1.0, XSLT 2.0, XSLT 3.0.

Parámetros

Argumento	Descripción
a	Parámetro obligatorio. Suministra el primer valor de la comparación.
b	Parámetro obligatorio. Suministra el segundo valor de la comparación.

6.7.5.5 less

El resultado es **true** si *a* es menor que *b*. De lo contrario devuelve **false**.



Lenguajes

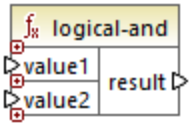
Built-in, C++, C#, Java, XQuery#, XSLT 1.0 y, XSLT 1.0, XSLT 2.0, XSLT 3.0.

Parámetros

Argumento	Descripción
a	Parámetro obligatorio. Suministra el primer valor de la comparación.
b	Parámetro obligatorio. Suministra el segundo valor de la comparación.

6.7.5.6 logical-and

Si los dos valores de entrada son **true**, el resultado es **true**. Si son diferentes, el resultado es **false**. Puede conectar el resultado a otra función `logical-and` de modo que puede conectar varias condiciones con una conjunción lógica Y para comprobar si todas ellas devuelven **true**. Esta función se puede ampliar para que acepte más argumentos, consulte Agregar o eliminar argumentos en una función 551.



Lenguajes

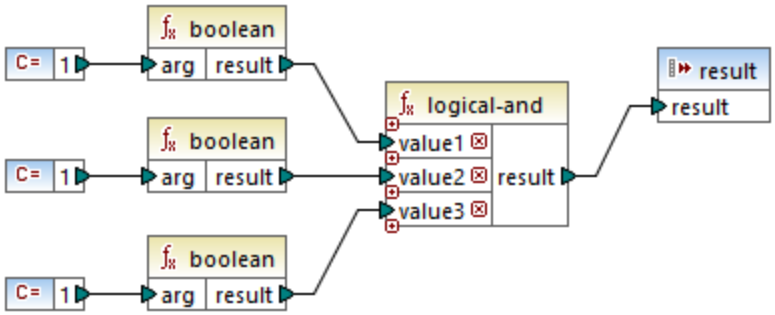
Built-in, C++, C#, Java, XQuery#, XSLT 1.0 y, XSLT 1.0, XSLT 2.0, XSLT 3.0.

Parámetros

Argumento	Descripción
value1	Parámetro obligatorio. Suministra el primer valor de la comparación.
value2	Parámetro obligatorio. Suministra el segundo valor de la comparación.

Ejemplo

La asignación siguiente devuelve **true** porque todos los valores de entrada de la función `logical-and` también son **true**. Si cualquiera de los valores de entrada fuera **false**, entonces el resultado de la asignación también sería **false**.

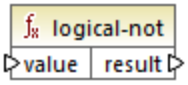


Véase también el apartado Ejemplo: búsqueda y concatenación 589.

6.7.5.7 logical-not

Invierte o le da la vuelta al estado lógico/resultado. Si el *valor* de entrada es **true**, el resultado de la función es **false**. Si el valor de entrada es **true**, el resultado de la función es **false**.

 © 2016-2022 Altova GmbH



Lenguajes

Built-in, C++, C#, Java, XQuery#, XSLT 1.0 y, XSLT 1.0, XSLT 2.0, XSLT 3.0.

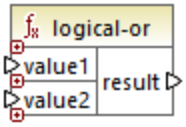
Parámetros

Argumento	Descripción
value	Parámetro obligatorio. Suministra el valor de entrada.

6.7.5.8 logical-or

Esta función exige que ambos valores de entrada sean booleanos. Si uno de los dos valores de entrada (ya sea value1 o value2) de la función logical-or es **true**, el resultado de la función es **true**. Si ambos son **false**, el resultado es **false**.

Esta función se puede ampliar para que acepte más argumentos, consulte Agregar o eliminar argumentos en una función 551.



Lenguajes

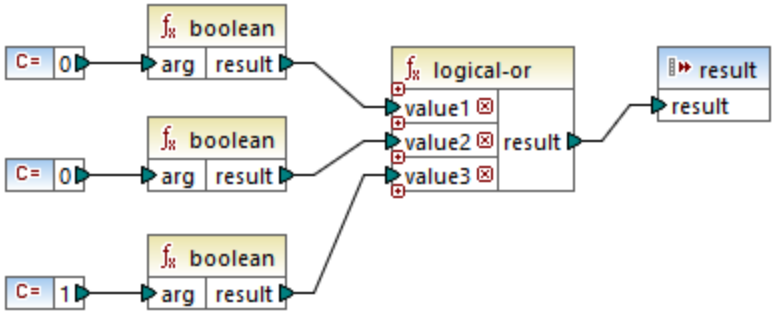
Built-in, C++, C#, Java, XQuery#, XSLT 1.0 y, XSLT 1.0, XSLT 2.0, XSLT 3.0.

Parámetros

Argumento	Descripción
value1	Parámetro obligatorio. Suministra el primer valor de la comparación.
value2	Parámetro obligatorio. Suministra el segundo valor de la comparación.

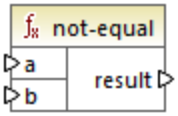
Ejemplo

El resultado de la asignación siguiente es **true** porque al menos uno de los argumentos de la función es **true**.



6.7.5.9 not-equal

El resultado es **true** si *a* no es igual a *b*. De lo contrario devuelve **false**.



Lenguajes

Built-in, C++, C#, Java, XQuery#, XSLT 1.0 y, XSLT 1.0, XSLT 2.0, XSLT 3.0.

Parámetros

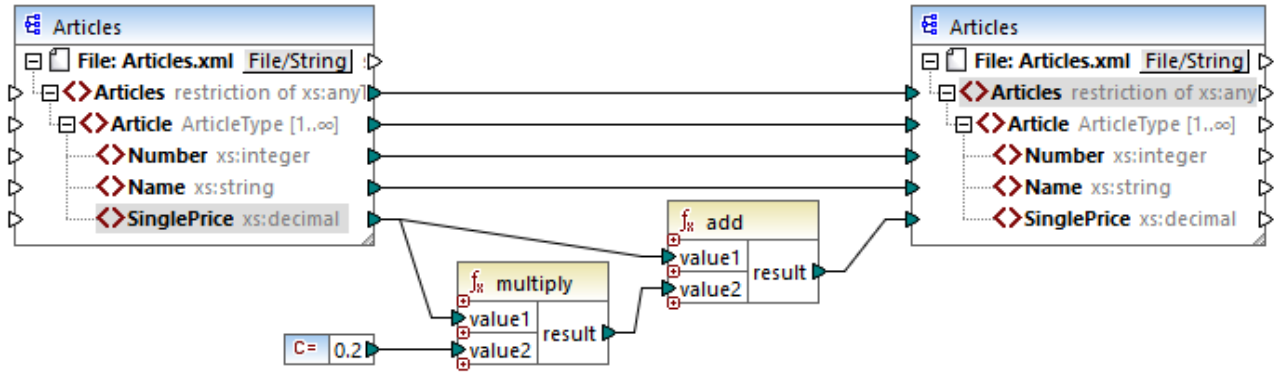
Argumento	Descripción
a	Parámetro obligatorio. Suministra el primer valor de la comparación.
b	Parámetro obligatorio. Suministra el segundo valor de la comparación.

6.7.6 core | math functions (matemáticas)

Las funciones matemáticas de la biblioteca core | math sirven para realizar operaciones matemáticas básicas con los datos. Tenga en cuenta que no se pueden usar para cálculos con datos `duration` ni `datetime`.

La mayoría de las funciones matemáticas toman dos parámetros de entrada (**value1** y **value2**) como operandos. Los valores de entrada se convierten automáticamente en tipo `decimal` para continuar el procesamiento. El resultado de las funciones matemáticas también es de tipo `decimal`

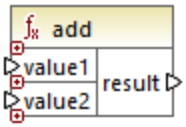
 © 2016-2022 Altova GmbH



En la asignación de esta imagen, por ejemplo, se añade un 20% de impuestos sobre las ventas en los artículos asignados al componente de destino.

6.7.6.1 add

El resultado es el valor decimal resultante de sumar **value1** a **value2**. Esta función se puede ampliar para que acepte más argumentos, consulte Agregar o eliminar argumentos en una función 551.



Lenguajes

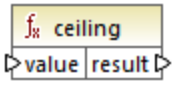
Built-in, C++, C#, Java, XQuery#, XSLT 1.0 y, XSLT 1.0, XSLT 2.0, XSLT 3.0.

Parámetros

Argumento	Descripción
value1	Parámetro obligatorio. Suministra el primer operando
value2	Parámetro obligatorio. Suministra el segundo operando

6.7.6.2 ceiling

El resultado es el entero más pequeño que es mayor o igual que **value**.



Lenguajes

Built-in, C++, C#, Java, XQuery#, XSLT 1.0 y, XSLT 1.0, XSLT 2.0, XSLT 3.0.

Parámetros

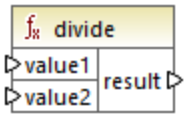
Argumento	Descripción
value	Parámetro obligatorio. Suministra el valor de entrada de la función.

Ejemplo

P. ej. si el resultado de una función de división es **11.2** y a este resultado le aplicamos la función **`ceiling`**, el resultado se convierte en **12** (es decir, el número entero mayor más cercano).

6.7.6.3 divide

El resultado es el valor decimal resultante de dividir **value1** por **value2**. La precisión del resultado depende del lenguaje de destino. Utilice la función round-precision 672 para definir la precisión del resultado.



Lenguajes

Built-in, C++, C#, Java, XQuery#, XSLT 1.0 y, XSLT 1.0, XSLT 2.0, XSLT 3.0.

Parámetros

Argumento	Descripción
value1	Parámetro obligatorio. Suministra el primer operando
value2	Parámetro obligatorio. Suministra el segundo operando

6.7.6.4 floor

El resultado es el entero mayor que sea mayor o igual que **value**.

 © 2016-2022 Altova GmbH

Lenguajes

Built-in, C++, C#, Java, XQuery#, XSLT 1.0 y, XSLT 1.0, XSLT 2.0, XSLT 3.0.

Parámetros

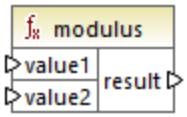
Argumento	Descripción
value	Parámetro obligatorio. Suministra el valor de entrada de la función.

Ejemplo

P. ej. si el resultado de una función de división es **11.7** y a este resultado le aplicamos la función `floor`, el resultado se convierte en **11** (es decir, el número entero menor más cercano).

6.7.6.5 modulus

El resultado es el entero restante después de dividir **value1** por **value2**.



Lenguajes

Built-in, C++, C#, Java, XQuery#, XSLT 1.0 y, XSLT 1.0, XSLT 2.0, XSLT 3.0.

Parámetros

Argumento	Descripción
value1	Parámetro obligatorio. Suministra el primer operando
value2	Parámetro obligatorio. Suministra el segundo operando

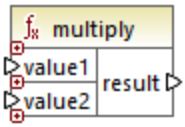
Ejemplo

Si los valores de entrada son **1.5** y **1**, entonces el resultado de la función `modulus` es **0.5**, que es la cifra que queda al dividir `1.5 / 1`.

Si los valores de entrada son **9** y **3**, entonces el resultado es **0** porque al dividir `9 / 3` no queda ningún resto.

6.7.6.6 multiply

El resultado es el valor decimal resultante de multiplicar **value1** por **value2**.



Lenguajes

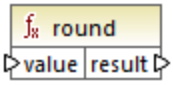
Built-in, C++, C#, Java, XQuery#, XSLT 1.0 y, XSLT 1.0, XSLT 2.0, XSLT 3.0.

Parámetros

Argumento	Descripción
value1	Parámetro obligatorio. Suministra el primer operando
value2	Parámetro obligatorio. Suministra el segundo operando

6.7.6.7 round

Devuelve el valor redondeado al número entero más cercano. Cuando el valor esté justo entre dos enteros, se utilizará el algoritmo Desempate la mitad alejándose del cero. Por ejemplo, el valor 10.5 se redondearía hasta 11 y el valor -10.5 se redondearía hasta -10.



Lenguajes

Built-in, C++, C#, Java, XQuery#, XSLT 1.0 y, XSLT 1.0, XSLT 2.0, XSLT 3.0.

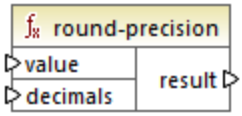
Parámetros

Argumento	Descripción
value	Parámetro obligatorio. Suministra el valor de entrada de la función.

6.7.6.8 round-precision

El resultado es el valor decimal del número redondeado a las cifras decimales definidas por el parámetro **decimals**.

© 2016-2022 Altova GmbH



Lenguajes

Disponible para Java, C#, C++ y el motor de ejecución integrado.

Parámetros

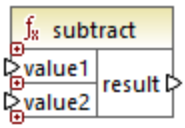
Argumento	Descripción
value	Parámetro obligatorio. Suministra el valor de entrada de la función.
decimals	Parámetro obligatorio. Indica el número de decimales a los que se debe redondear.

Ejemplo

Si redondeamos el valor **2.777777** a dos decimales el resultado es **2.78**. Si redondeamos el valor **0.1234** a 3 decimales el resultado es **0.123**.

6.7.6.9 subtract

El resultado es el valor decimal resultante de restar **value2** a **value1**.



Lenguajes

Built-in, C++, C#, Java, XQuery#, XSLT 1.0 y, XSLT 1.0, XSLT 2.0, XSLT 3.0.

Parámetros

Argumento	Descripción
value1	Parámetro obligatorio. Suministra el primer operando
value2	Parámetro obligatorio. Suministra el segundo operando

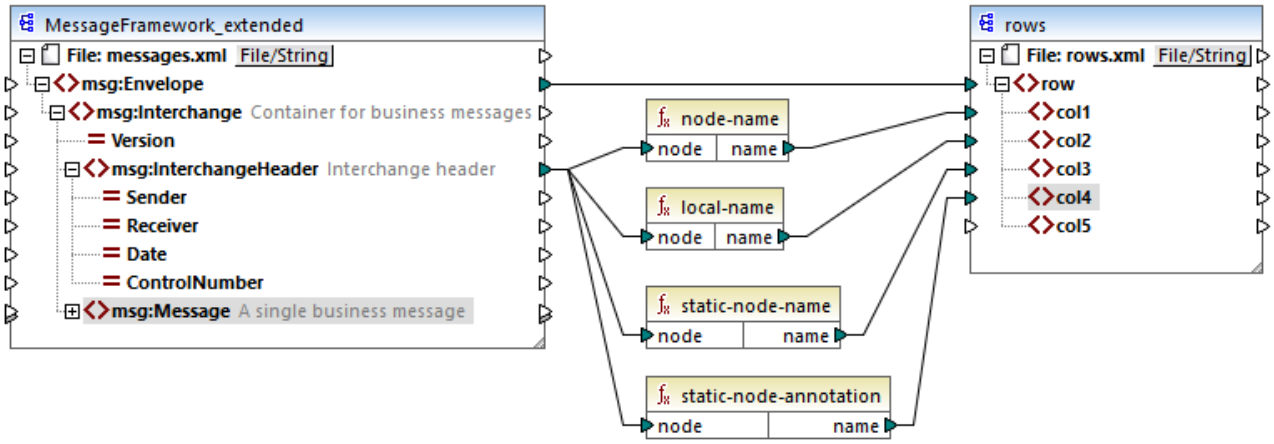
6.7.7 core | node functions (nodo)

Las funciones de la biblioteca **core | node functions** permiten acceder a información sobre los nodos de un componente de asignación (como el nombre o la anotación del nodo) o elementos nillable (véase también Valores Nil y Nillable) 131.

Tenga en cuenta que hay otra manera de acceder a los nombres de los nodos que no necesita funciones de nodo (véase Asignar nombres de nodos) 850.

La asignación siguiente muestra algunas funciones de nodo que obtienen información del nodo **msg:InterchangeHeader** del archivo XML de origen. Más concretamente, esta es la información que se extrae:

1. La función **`node-name`** devuelve el nombre completo del nodo, que incluye el prefijo del nodo.
2. La función **`local-name`** devuelve solamente la parte local.
3. La función **`static-node-name`** es parecida a **`node-name`**, pero también está disponible en XSLT 1.0.
4. La función **`static-node-annotation`** obtiene la anotación del elemento tal y como está definida en el esquema XML.



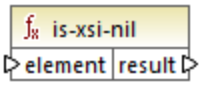
El resultado de la asignación es el siguiente (sin incluir las declaraciones XML y de espacio de nombres):

```
<row>
 <col1>msg:InterchangeHeader</col1>
 <col2>InterchangeHeader</col2>
 <col3>msg:InterchangeHeader</col3>
 <col4>Interchange header</col4>
</row>
```

 © 2016-2022 Altova GmbH

6.7.7.1 is-xsi-nil

Devuelve **true** si el nodo **element** del componente de origen tiene un atributo `xsi:nil` con el valor **true**.



Lenguajes

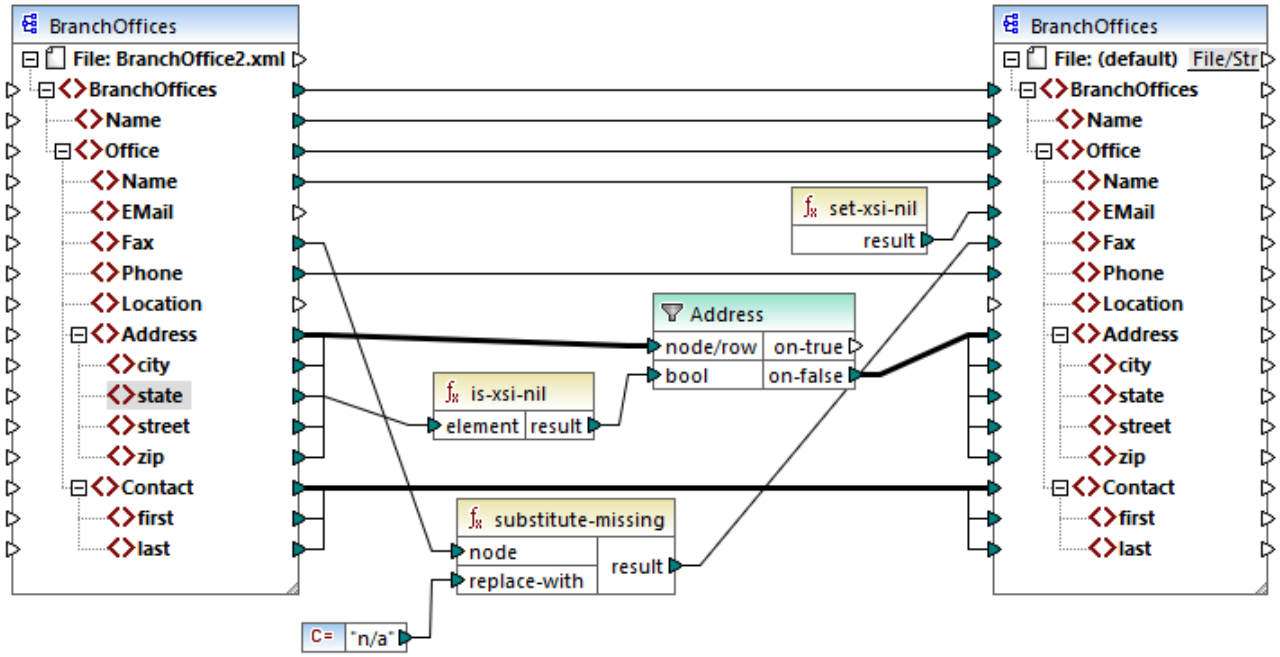
Built-in, C++, C#, Java, XQuery#, XSLT 1.0 y, XSLT 1.0, XSLT 2.0, XSLT 3.0.

Parámetros

Argumento	Descripción
element	Parámetro obligatorio. Debe estar conectado al nodo de origen que se quiere comprobar.

Ejemplo

El diseño de asignación siguiente copia datos de un archivo XML de origen a uno de destino de forma condicional e ilustra también cómo se usan distintas funciones, incluida **`is-xsi-nil`**. Esta asignación se llama **HandlingXsiNil.mfd** y está en la carpeta **<Documentos>\Altova\MapForce2023\MapForceExamples**.



Como se ve en la imagen anterior, la función **is-xsi-nil** comprueba si el atributo `xsi:nil` tiene el atributo "true" para el elemento **state** en el archivo de origen Si, por el contrario, este atributo es "false", el filtro copiará al elemento superior **Address** en el archivo de destino. El archivo XML de origen tiene este aspecto (sin incluir las declaraciones XML y de espacio de nombres):

```
<BranchOffices>
 <Name>Nanonull</Name>
 <Office>
 <Name>Nanonull Research Outpost</Name>
 <EMail>sp@nanonull.com</EMail>
 <Fax xsi:nil="true"/>
 <Phone>+8817 3141 5926</Phone>
 <Address>
 <city>South Pole</city>
 <state xsi:nil="true"/>
 <street xsi:nil="true"/>
 <zip xsi:nil="true"/>
 </Address>
 <Contact>
 <first>Scott</first>
 <last>Amundsen</last>
 </Contact>
 </Office>
</BranchOffices>
```

El resultado de la asignación es que no se copia ningún elemento **Address** en el archivo de destino porque solo hay uno en el archivo de origen y el atributo `xsi:nil` es "true" para el elemento **state**. En consecuencia, el resultado de la asignación es este:

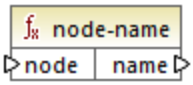
```
<BranchOffices>
 <Name>Nanonull</Name>
 <Office>
 <Name>Nanonull Research Outpost</Name>
 <EMail xsi:nil="true"/>
 <Fax>n/a</Fax>
 <Phone>+8817 3141 5926</Phone>
 <Contact>
 <first>Scott</first>
 <last>Amundsen</last>
 </Contact>
 </Office>
</BranchOffices>
```

6.7.7.2 node-name

Devuelve una cadena que contiene el nombre (QName) del nodo conectado. Si el nodo es un nodo XML **text()**, devuelve un QName vacío. Esta función solamente funciona con los nodos que tienen nombre. Si XSLT 2.0 es el lenguaje de destino (que llama a la función **fn:node-name**), la función devuelve una secuencia vacía para los nodos que no tienen nombre.

 © 2016-2022 Altova GmbH

Nota: Esta función no es compatible con nodos "Introducción de archivos", tablas o campos de BD, XBRL, Excel, JSON o campos de Protocol Buffers.



Lenguajes

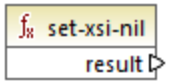
Built-in, C++, C#, Java, XQuery, XSLT 2.0, XSLT 3.0.

Parámetros

Argumento	Descripción
nodo	Parámetro obligatorio. Conecta la entrada con el nodo cuyo nombre se quiere obtener.

6.7.7.3 set-xsi-nil

Establece el nodo de destino en xsi:nil.



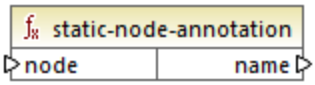
Lenguajes

Built-in, C++, C#, Java, XQuery, XSLT 2.0, XSLT 3.0.

6.7.7.4 static-node-annotation

Devuelve una cadena con la anotación del nodo conectado. El parámetro de entrada node debe ser: (i) un nodo del componente de origen o (ii) una función inline 578 directamente conectada a un parámetro 581, directamente conectado a su vez a un nodo de la asignación que llama a la función.

La conexión debe ser directa. No puede pasar por un filtro ni por una función definida por el usuario no inline. Se trata de una pseudo función, que en tiempo de generación se reemplaza con un texto adquirido del nodo conectado y, por tanto, está disponible en todos los lenguajes.



Lenguajes

Built-in, C++, C#, Java, XQuery#, XSLT 1.0 y, XSLT 1.0, XSLT 2.0, XSLT 3.0.

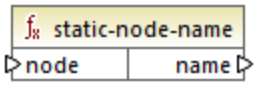
Parámetros

Argumento	Descripción
nodo	Parámetro obligatorio. Conecta el valor de entrada con el nodo cuyo anotación se quiere obtener.

6.7.7.5 static-node-name

Devuelve una cadena que contiene el nombre del nodo conectado. El parámetro de entrada node debe ser: (i) un nodo del componente de origen o (ii) una función inline 578 directamente conectada a un parámetro 581, directamente conectado a su vez a un nodo de la asignación que llama a la función.

La conexión debe ser directa. No puede pasar por un filtro ni por una función definida por el usuario no inline. Se trata de una pseudo función, que en tiempo de generación se reemplaza con un texto adquirido del nodo conectado y, por tanto, está disponible en todos los lenguajes.



Lenguajes

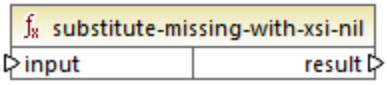
Built-in, C++, C#, Java, XQuery#, XSLT 1.0 y, XSLT 1.0, XSLT 2.0, XSLT 3.0.

Parámetros

Argumento	Descripción
nodo	Parámetro obligatorio. Conecta la entrada con el nodo cuyo nombre se quiere obtener.

6.7.7.6 substitute-missing-with-xsi-nil

Para los nodos de contenido simple esta función reemplaza los valores que faltan o los valores nulos del componente de origen con el atributo `xsi:nil` en el nodo de destino.



© 2016-2022 Altova GmbH

Lenguajes

Built-in, C++, C#, Java, XQuery, XSLT 2.0, XSLT 3.0.

Parámetros

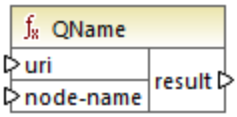
Argumento	Descripción
input	Parámetro obligatorio. Conecta la entrada con el nodo cuyo nombre se quiere obtener.

6.7.8 core | QName functions (QName)

Las funciones QName permiten manipular los nombres completos (QName) en documentos XML.

6.7.8.1 QName

Construye un QName a partir de un URI de espacio de nombres y una parte local. Use esta función para crear un QName en un componente de destino. Los parámetros **uri** y **node-name** se pueden dar con una función constante.



Lenguajes

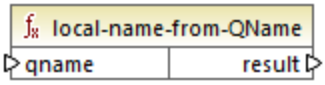
Built-in, C++, C#, Java, XQuery, XSLT 2.0, XSLT 3.0.

Parámetros

Nombre	Descripción
uri	Obligatorio. Suministra el URI.
node-name	Obligatorio. Suministra el nombre del nodo.

6.7.8.2 local-name-from-QName

Extrae la parte del nombre local de un valor de tipo `xs:QName`. Tenga en cuenta que, a diferencia de la función local-name que devuelve el nombre local del nodo, esta función procesa el contenido del elemento conectado a la entrada qnameTenga en, que devuelve el nombre local del *nodo*, esta función procesa el *contenido* del elemento conectado a la entrada **qname**



Lenguajes

Built-in, C++, C#, Java, XQuery, XSLT 2.0, XSLT 3.0.

Parámetros

Nombre	Descripción
qname	Obligatorio. Suministra el valor de entrada de la función, que es de tipo `xs:QName`.

6.7.8.3 namespace-uri-from-QName

El resultado es la parte URI del espacio de nombres del valor QName dado como argumento.

Lenguajes

Built-in, C++, C#, Java, XQuery, XSLT 2.0, XSLT 3.0.

Parámetros

Nombre	Descripción
qname	Obligatorio. Suministra el valor de entrada de la función.

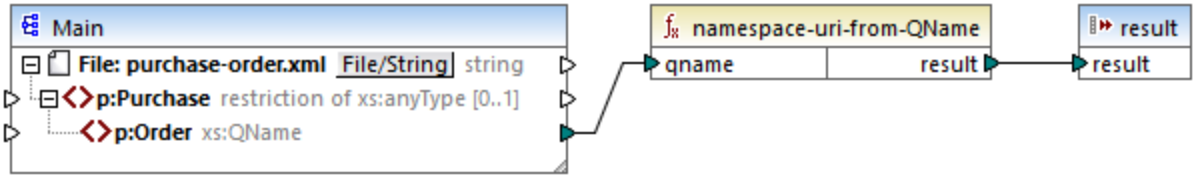
Ejemplo

Este archivo contiene el valor QName **o:name**. Observe que el prefijo "o" está asignado al espacio de nombres **`http://NamespaceTest.com/Order`**.

```
<?xml version="1.0" encoding="utf-8"?>
<p:Purchase xsi:schemaLocation="http://NamespaceTest.com/Purchase Main.xsd"
 xmlns:p="http://NamespaceTest.com/Purchase"
 xmlns:o="http://NamespaceTest.com/Order"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
 <p:Order>o:name</p:Order>
</p:Purchase>
```

 © 2016-2022 Altova GmbH

Esta asignación procesa el valor Qname y obtiene el URI de espacio de nombre:



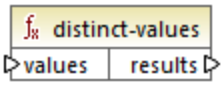
El resultado de esta asignación es `http://NamespaceTest.com/Order`.

6.7.9 core | sequence functions (secuencia)

Las funciones de secuencia[872] permiten procesar los datos de entrada y agrupar su contenido.

6.7.9.1 distinct-values

Procesa la secuencia de valores conectada al componente **valores** y devuelve solamente los valores distintos en forma de secuencia. Esto puede ser útil si necesita eliminar valores duplicados de una secuencia y copiar solamente los elementos únicos en el componente de destino.



Lenguajes

Built-in, C++, C#, Java, XQuery, XSLT 2.0, XSLT 3.0.

Parámetros

Nombre	Descripción
values	Esta entrada debe recibir una conexión desde un elemento de la asignación que suministre una secuencia[872] de cero o más valores. Por ejemplo, la conexión puede provenir de un elemento XML de origen, un campo CSV, un registro de BD, etc.

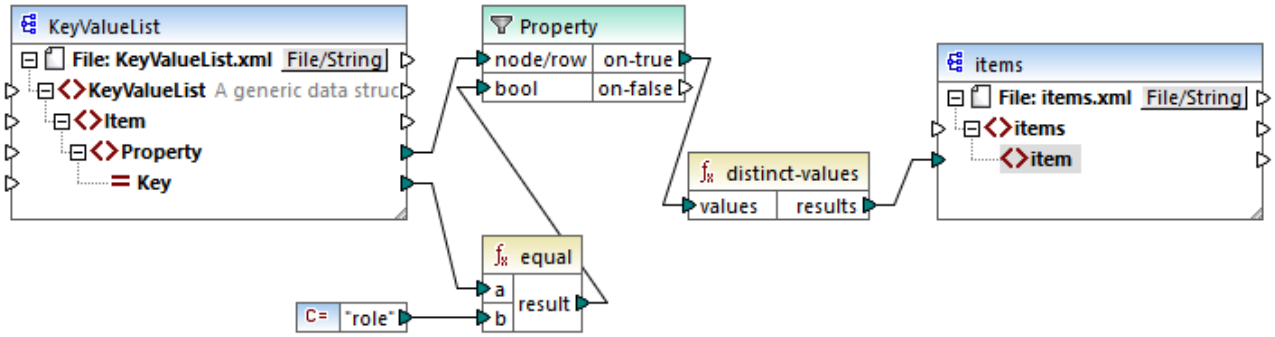
Ejemplo

Este archivo XML contiene información sobre los trabajadores de una empresa de muestra. Algunos trabajadores tienen el mismo puesto, por tanto, el atributo "role" contiene valores duplicados. Por ejemplo, tanto "Loby Matise" como "Susi Sanna" tienen el puesto "Support".

```
<?xml version="1.0" encoding="UTF-8"?>
<KeyValueList xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
```

```
xsi:noNamespaceSchemaLocation="KeyValueList.xsd">
 <Item>
 <Property Key="role">Manager</Property>
 <Property Key="First">Vernon</Property>
 <Property Key="Last">Callaby</Property>
 </Item>
 <Item>
 <Property Key="role">Programmer</Property>
 <Property Key="First">Frank</Property>
 <Property Key="Last">Further</Property>
 </Item>
 <Item>
 <Property Key="role">Support</Property>
 <Property Key="First">Loby</Property>
 <Property Key="Last">Matise</Property>
 </Item>
 <Item>
 <Property Key="role">Support</Property>
 <Property Key="First">Susi</Property>
 <Property Key="Last">Sanna</Property>
 </Item>
</KeyValueList>
```

Imagine que necesita extraer una lista de todos los nombres de puesto únicos del archivo XML. Para ello puede usar una asignación como esta:



En la asignación anterior ocurre lo siguiente:

- Cada elemento **Property** del archivo XML de origen es procesado por un filtro.
- La conexión con el elemento de entrada **bool** del filtro garantiza que solo se pasen al componente de destino elementos **Property** en los que el atributo **Key** es igual a "role". La cadena "role" viene dada por una constante. Tenga en cuenta que el resultado del filtro todavía produce duplicados de momento (porque hay dos propiedades "Support" que cumplen la condición del filtro).
- La secuencia que produce el filtro es procesada por la función `distinct-values`, que excluye los valores duplicados.

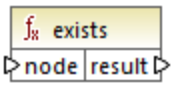
El resultado de la asignación es el siguiente:

 © 2016-2022 Altova GmbH

```
<items>
 <item>Manager</item>
 <item>Programmer</item>
 <item>Support</item>
</items>
```

6.7.9.2 exists

Devuelve **true** si el nodo existe. De lo contrario devuelve **false**. Esta función se suele usar con filtros porque devuelve un valor booleano, lo que permite filtrar 543 por registros que tienen (o que no tienen) un elemento o atributo secundario.



Lenguajes

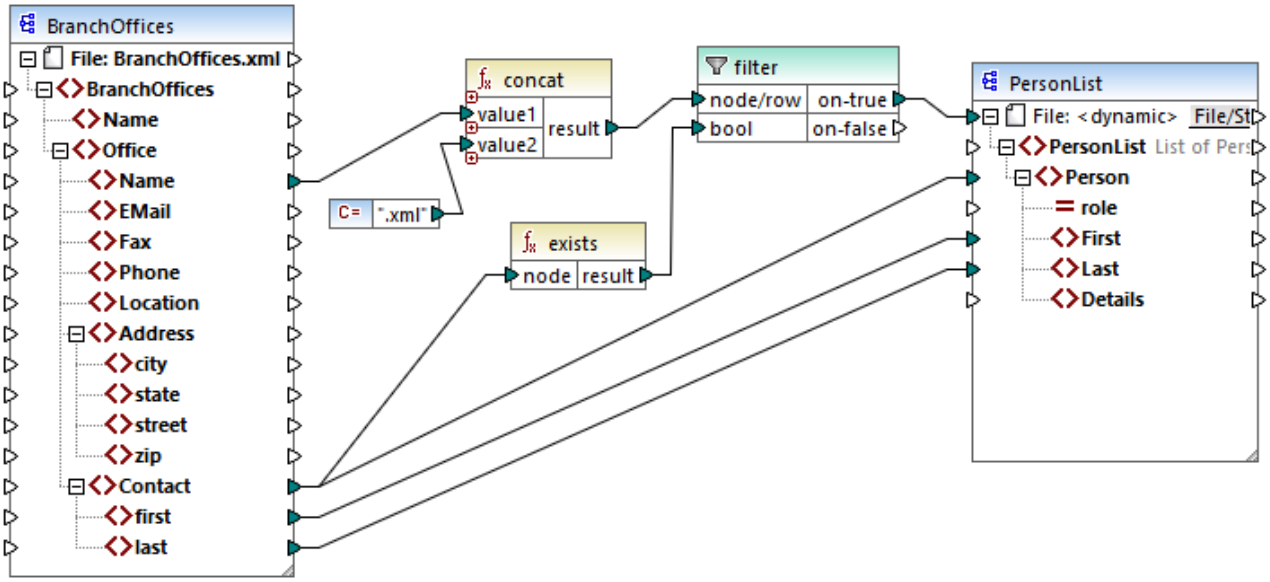
Built-in, C++, C#, Java, XQuery#, XSLT 1.0 y, XSLT 1.0, XSLT 2.0, XSLT 3.0.

Parámetros

Nombre	Descripción
nodo	El nodo cuya existencia se quiere comprobar.

Ejemplos

La asignación siguiente muestra cómo filtrar datos con ayuda de la función `exists`. Esta asignación se llama **PersonListsForAllBranchOffices.mfd** y está en la carpeta **<Documentos>\Altova\MapForce2023\MapForceExamples**.



PersonListsForAllBranchOffices.mfd

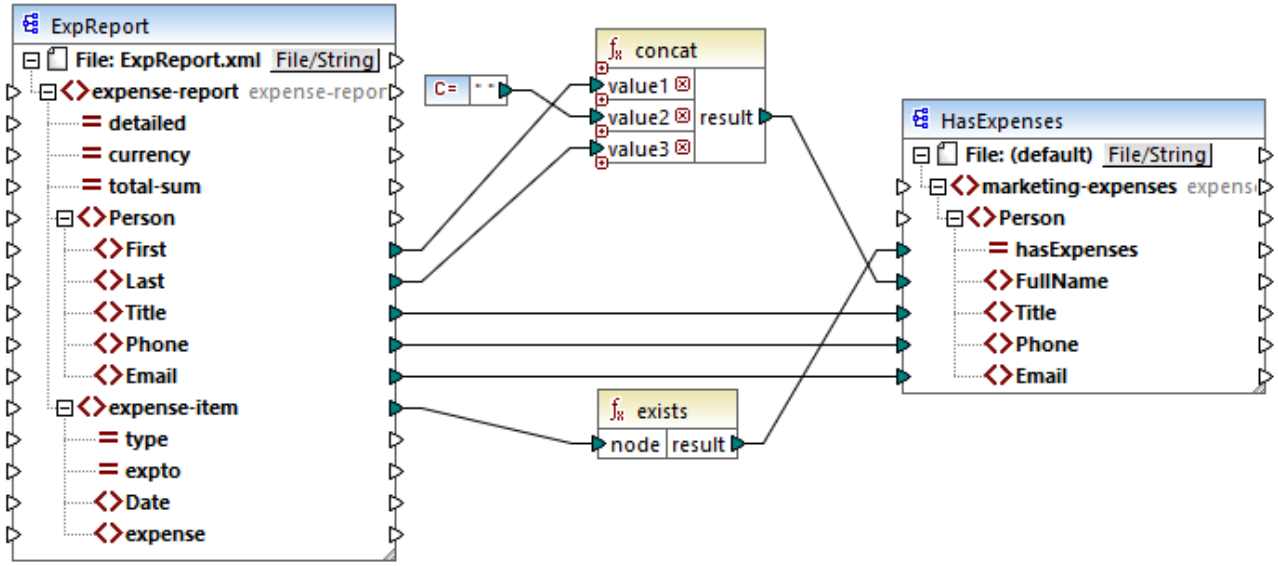
En el archivo de origen **BranchOffices.xml** hay tres elementos **Office**. En concreto, uno de ellos no tienen ningún elemento secundario **Contact**. La asignación tiene varios objetivos:

a) por cada oficina, extraer la lista de contactos de esa oficina
b) por cada oficina, crear un archivo XML con el mismo nombre que esa oficina
C) no generar el archivo XML si la oficina no tiene contactos.

Para ello se ha añadido un filtro a la asignación. El filtro pasa al archivo de destino solamente los elementos **Office** en los que existe al menos un elemento **Contact**. Esta condición booleana viene dada por la función `exists`. Si el resultado de la función es true, entonces el nombre de la oficina se concatena con la cadena de texto `.xml` para producir el nombre del archivo de destino. Para más información sobre cómo generar nombres de archivo para la asignación consulte Procesar varios archivos de entrada o salida simultáneamente 892 .

Otro ejemplo sería esta asignación:
<Documentos>\Altova\MapForce2023\MapForceExamples\HasMarketingExpenses.mfd. Si existe un elemento **expense-item** en el XML de origen, entonces el atributo "hasExpenses" recibe el valor **true** en el esquema o archivo XML de destino.

 © 2016-2022 Altova GmbH

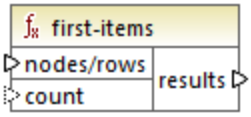


HasMarketingExpenses.mfd

Véase también el apartado Ejemplo: excepción cuando un nodo no existe 530.

6.7.9.3 first-items

Devuelve los *X* últimos nodos de la secuencia nodes/rows, siendo *X* el número dado por el parámetro **count**.



Lenguajes

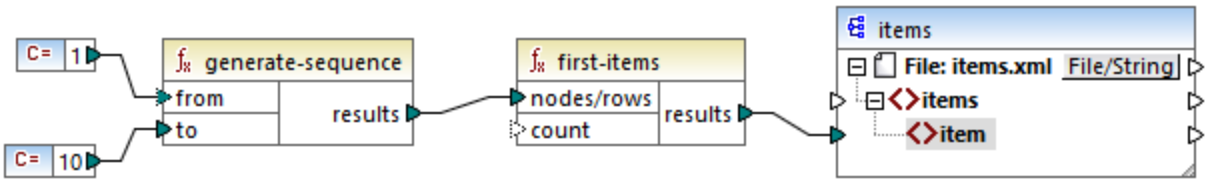
Built-in, C++, C#, Java, XQuery, XSLT 2.0, XSLT 3.0.

Parámetros

Nombre	Descripción
nodos/filas	Esta entrada debe recibir una conexión desde un elemento de la asignación que suministre una secuencia 872 de cero o más valores. Por ejemplo, la conexión puede provenir de un elemento XML de origen, un campo CSV, un registro de BD, etc.
count	Parámetro opcional. Indica cuántos nodos se deben recuperar de la secuencia de entrada. El valor predeterminado es **1**.

Ejemplo

La asignación de ejemplo siguiente genera una secuencia de 10 valores. Esta secuencia es procesada por la función **first-items**, que escribe el resultado en un archivo XML de destino.



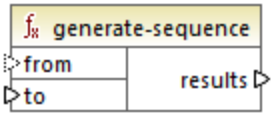
Como el argumento **count** no tiene ningún valor, se aplica el valor predeterminado **1**. El resultado es que en la salida de la asignación solo se genera el primer valor de la secuencia:

```
<items>
 <item>1</item>
</items>
```

Para ver un ejemplo más realista consulte la asignación **FindHighestTemperatures.mfd** de Pasar parámetros a la asignación 477.

6.7.9.4 generate-sequence

Crea una secuencia de enteros usando como límite los parámetros from y to.



Lenguajes

Built-in, C++, C#, Java, XQuery, XSLT 2.0, XSLT 3.0.

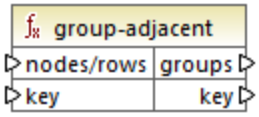
Parámetros

Nombre	Descripción
from	Parámetro opcional. Indica el número entero con el que debe empezar la secuencia (límite inferior). El valor predeterminado es **1**.
to	Parámetro obligatorio. Indica el número entero con el que debe terminar la secuencia (límite superior).

 © 2016-2022 Altova GmbH

6.7.9.5 group-adjacent

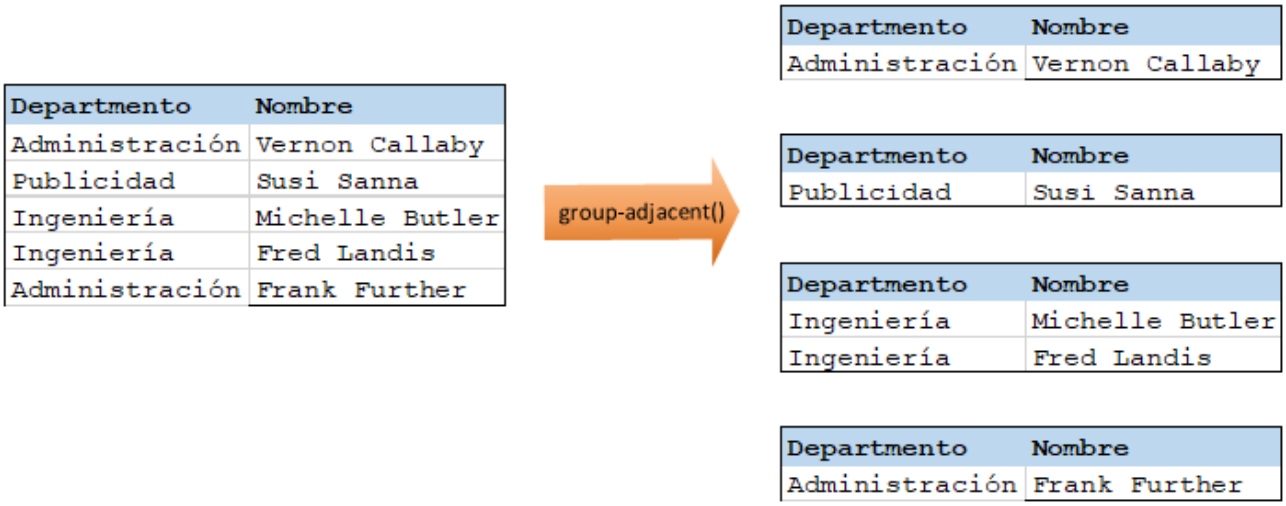
La función **group-adjacent** agrupa los elementos conectados a los datos de entrada de **nodes/rows** con la clave conectada a los datos de entrada de **key**. Observe que si los elementos que comparten la misma clave no son adyacentes, esta clave los coloca en grupos separados. Si varios elementos consecutivos (adyacentes) comparten la misma clave estarán en el mismo grupo.



Por ejemplo, en la imagen siguiente, que ilustra una transformación abstracta, la clave de agrupación es "Department". El lado izquierdo del diagrama muestra los datos de entrada, mientras que el lado derecho muestra los datos de salida tras la agrupación. Al ejecutar la transformación ocurre lo siguiente:

- Inicialmente, la primera clave, "Administración", crea un grupo nuevo.
- La siguiente clave es distinta, así que se crea un segundo grupo, "Publicidad".
- Como la tercera clave también es diferente, se crea otro grupo, "Ingeniería".
- La cuarta clave es la misma que la tercera, por lo que este registro se coloca en el grupo que ya existe para esa clave.
- Por último, la quinta clave es diferente de la cuarta, lo que hace que se cree un último grupo.

Como se ve en la siguiente imagen, "Michelle Butler" y "Fred Landis" están en el mismo grupo porque tienen la misma clave y son adyacentes. Sin embargo, "Vernon Callaby" y "Frank Further" están en grupos distintos porque no son adyacentes, aunque tengan la misma clave.



Lenguajes

La función está disponible para XSLT 2.0, XSLT 3.0, Java, C#, C++ y el motor de ejecución integrado.

Parámetros

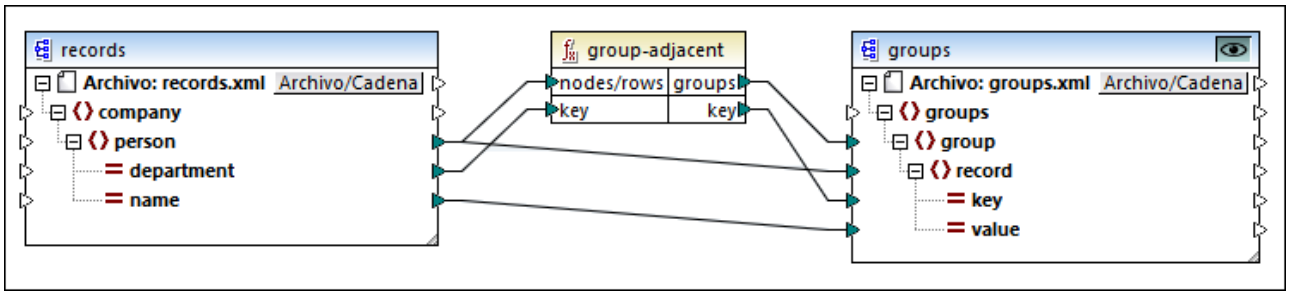
Nombre	Descripción
nodos/filas	Esta entrada debe recibir una conexión desde un elemento de la asignación que suministre una secuencia 872 de cero o más valores. Por ejemplo, la conexión puede provenir de un elemento XML de origen, un campo CSV, un registro de BD, etc.
key	La clave conforme a la cual se agrupan los elementos.

Ejemplo

Imaginemos que sus datos de origen son un archivo XML con el siguiente contenido (tenga en cuenta que, para simplificar, en el código de ejemplo siguiente hemos eliminado el espacio de nombre y las declaraciones XML).

```
<company>
 <person department="Administration" name="Vernon Callaby"/>
 <person department="Marketing" name="Susi Sanna"/>
 <person department="Engineering" name="Michelle Butler"/>
 <person department="Engineering" name="Fred Landis"/>
 <person department="Administration" name="Frank Further"/>
</company>
```

El requisito empresarial es agrupar registros de personas por departamento, siempre que sean adyacentes. Para conseguirlo, la siguiente asignación invoca la función **group-adjacent** y da como clave el elemento **department**.



El resultado de la asignación es el siguiente:

```
<groups>
 <group>
 <record key="Administration" value="Vernon Callaby"/>
 </group>
 <group>
 <record key="Marketing" value="Susi Sanna"/>
 </group>
 <group>
```

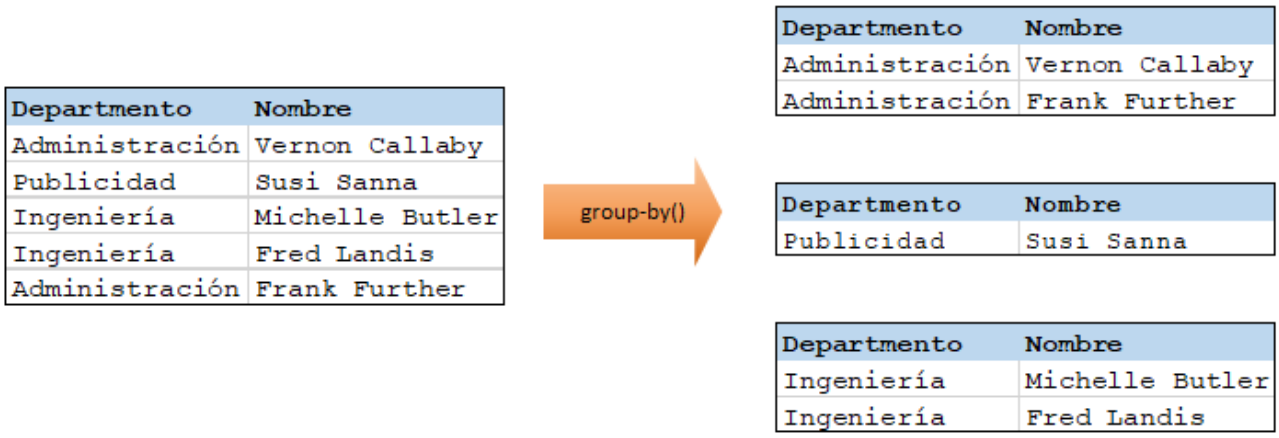

© 2016-2022 Altova GmbH

```
 <record key="Engineering" value="Michelle Butler"/>
 <record key="Engineering" value="Fred Landis"/>
 </group>
 <group>
 <record key="Administration" value="Frank Further"/>
 </group>
</groups>
```

Este ejemplo, junto con otros ejemplos de agrupación, es parte de esta asignación: **<Documentos>\Altova\MapForce2023\MapForceExamples\Tutorial\GroupingFunctions.mfd**. Antes de hacer clic en la pestaña **Resultados** recuerde hacer clic en el botón **Vista previa** de la función que quiere comprobar.

6.7.9.6 group-by

La función **group-by** crea grupos de registros conforme a las claves de agrupación que se indiquen. Por ejemplo, en el siguiente ejemplo de transformación abstracta la clave de agrupación es "Departmento". Como hay un total de tres departamentos únicos, al aplicar la función **group-by** se crean tres grupos:



Lenguajes

La función está disponible para XSLT 2.0, XSLT 3.0, Java, C#, C++ y el motor de ejecución integrado.

Parámetros

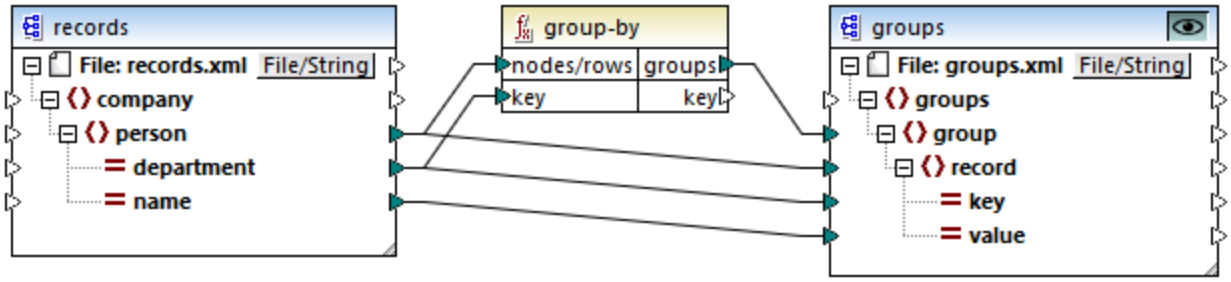
Nombre	Descripción
nodos/filas	Esta entrada debe recibir una conexión desde un elemento de la asignación que suministre una secuencia 872 de cero o más valores. Por ejemplo, la conexión puede provenir de un elemento XML de origen, un campo CSV, un registro de BD, etc.
key	La clave conforme a la cual se agrupan los elementos.

Ejemplo

Imaginemos que sus datos de origen son un archivo XML con el siguiente contenido (tenga en cuenta que, para simplificar, en el código de ejemplo siguiente hemos eliminado el espacio de nombre y las declaraciones XML).

```
<company>
 <person department="Administration" name="Vernon Callaby"/>
 <person department="Marketing" name="Susi Sanna"/>
 <person department="Engineering" name="Michelle Butler"/>
 <person department="Engineering" name="Fred Landis"/>
 <person department="Administration" name="Frank Further"/>
</company>
```

E requisito empresarial es agrupar registros de personas por departamento. Para conseguirlo, la siguiente asignación invoca la función **group-by** y da como clave el elemento **department**.



El resultado de la asignación es el siguiente:

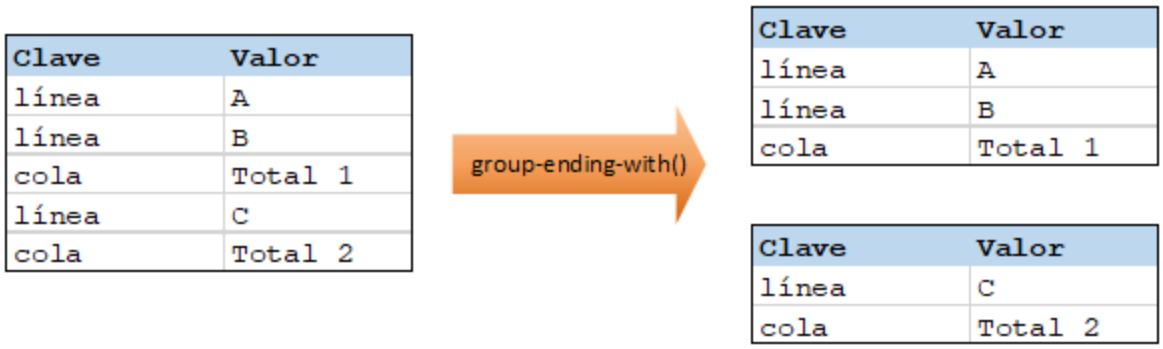
```
<groups>
 <group>
 <record key="Administration" value="Vernon Callaby"/>
 <record key="Administration" value="Frank Further"/>
 </group>
 <group>
 <record key="Marketing" value="Susi Sanna"/>
 </group>
 <group>
 <record key="Engineering" value="Michelle Butler"/>
 <record key="Engineering" value="Fred Landis"/>
 </group>
</groups>
```

Este ejemplo, junto con otros ejemplos de agrupación, es parte de esta asignación: **<Documentos>\Altova\MapForce2023\MapForceExamples\Tutorial\GroupingFunctions.mfd**. Antes de hacer clic en la pestaña **Resultados** recuerde hacer clic en el botón **Vista previa** de la función que quiere comprobar.

© 2016-2022 Altova GmbH

6.7.9.7 group-ending-with

La función **group-ending** toma como argumento una condición booleana. Si la condición booleana se cumple, se crea un grupo nuevo en el que el último registro es el que cumple esa condición. En el siguiente ejemplo, la condición es que la clave sea "cola". Esta condición se cumple en los registros tercero y quinto, por lo que se crean dos grupos:



Nota: si existen registros después del último que cumple la condición, se crea un grupo más. Por ejemplo, si hubiera más registros "línea" después del último registro "cola", estos se colocarían en ese grupo adicional.

Lenguajes

La función está disponible para XSLT 2.0, XSLT 3.0, Java, C#, C++ y el motor de ejecución integrado.

Parámetros

Nombre	Descripción
nodos/filas	Esta entrada debe recibir una conexión desde un elemento de la asignación que suministre una secuencia 872 de cero o más valores. Por ejemplo, la conexión puede provenir de un elemento XML de origen, un campo CSV, un registro de BD, etc.
bool	xs:boolean Indica la condición booleana que inicia un nuevo grupo si se cumple (true).

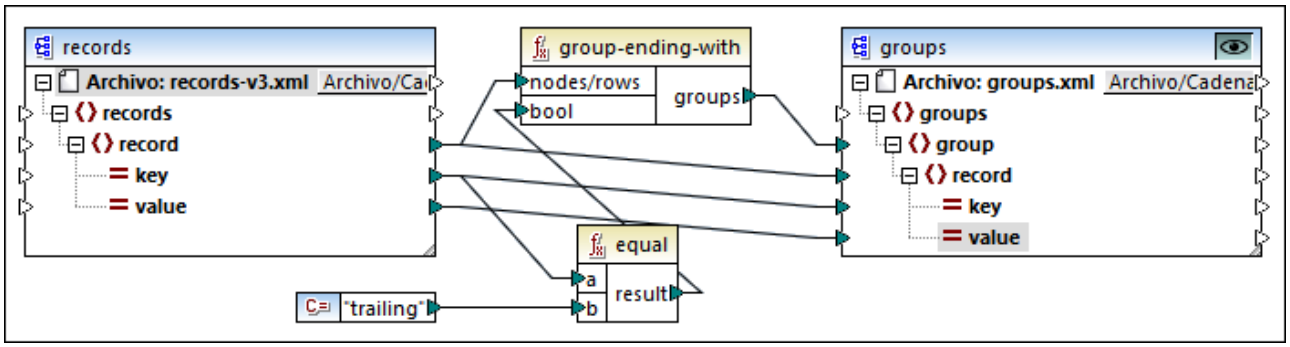
Ejemplo

Imaginemos que sus datos de origen son un archivo XML con el siguiente contenido (tenga en cuenta que, para simplificar, en el código de ejemplo siguiente hemos eliminado el espacio de nombre y las declaraciones XML).

```
<records>
 <record key="line" value="A"/>
 <record key="line" value="B"/>
 <record key="trailing" value="Total 1"/>
 <record key="line" value="C"/>
```

```
 <record key="trailing" value="Total 2"/>
</records>
```

El requisito empresarial es crear un grupo por cada elemento "trailing". Cada uno de los grupos también debe incluir los registros "line" que precedan al registro "trailing". Para ello, la siguiente asignación invoca la función **group-ending-with**. En la siguiente asignación, siempre que el nombre de **key** sea "trailing", el argumento dado a **bool** pasa a ser **true** y se crea un grupo nuevo.



El resultado de la asignación es el siguiente:

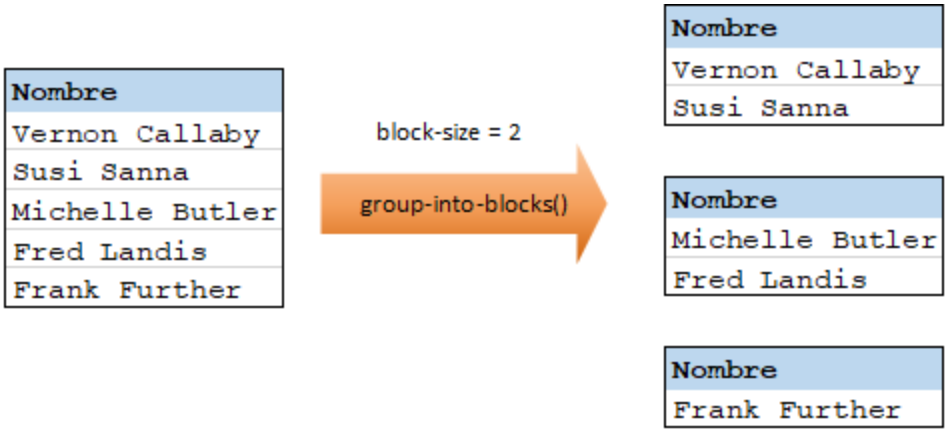
```
<groups>
 <group>
 <record key="line" value="A"/>
 <record key="line" value="B"/>
 <record key="trailing" value="Total 1"/>
 </group>
 <group>
 <record key="line" value="C"/>
 <record key="trailing" value="Total 2"/>
 </group>
</groups>
```

Este ejemplo, junto con otros ejemplos de agrupación, es parte de esta asignación: **<Documentos>\Altova\MapForce2023\MapForceExamples\Tutorial\GroupingFunctions.mfd**. Antes de hacer clic en la pestaña **Resultados** recuerde hacer clic en el botón **Vista previa** de la función que quiere comprobar.

6.7.9.8 group-into-blocks

La función **group-into-blocks** crea grupos iguales que contienen exactamente N elementos, donde N es el valor que se le da al argumento `block-size`. Observe que el último grupo puede contener N elementos o menos en función de la cantidad de elementos de la fuente. En el ejemplo siguiente, `block-size` es 2. Como hay cinco elementos en total, cada grupo contiene exactamente dos elementos, excepto el último, que solo puede contener uno.

 © 2016-2022 Altova GmbH



Lenguajes

La función está disponible para XSLT 2.0, XSLT 3.0, Java, C#, C++ y el motor de ejecución integrado.

Parámetros

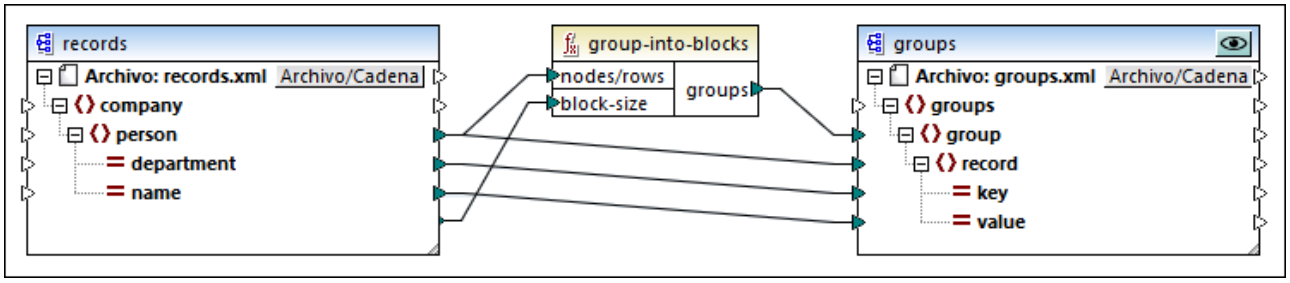
Nombre	Descripción
nodos/filas	Esta entrada debe recibir una conexión desde un elemento de la asignación que suministre una secuencia 872 de cero o más valores. Por ejemplo, la conexión puede provenir de un elemento XML de origen, un campo CSV, un registro de BD, etc.
block-size	Indica el tamaño de cada grupo.

Ejemplo

Imaginemos que sus datos de origen son un archivo XML con el siguiente contenido (tenga en cuenta que, para simplificar, en el código de ejemplo siguiente hemos eliminado el espacio de nombre y las declaraciones XML).

```
<company>
 <person department="Administration" name="Vernon Callaby"/>
 <person department="Marketing" name="Susi Sanna"/>
 <person department="Engineering" name="Michelle Butler"/>
 <person department="Engineering" name="Fred Landis"/>
 <person department="Administration" name="Frank Further"/>
</company>
```

El requisito empresarial es agrupar registros de personas en bloques de dos elementos cada uno. Para ello, la siguiente asignación invoca la función **group-into-blocks** y da como tamaño (**block-size**) el valor entero "2".



El resultado de la asignación es el siguiente:

```
<groups>
 <group>
 <record key="Administration" value="Vernon Callaby"/>
 <record key="Marketing" value="Susi Sanna"/>
 </group>
 <group>
 <record key="Engineering" value="Michelle Butler"/>
 <record key="Engineering" value="Fred Landis"/>
 </group>
 <group>
 <record key="Administration" value="Frank Further"/>
 </group>
</groups>
```

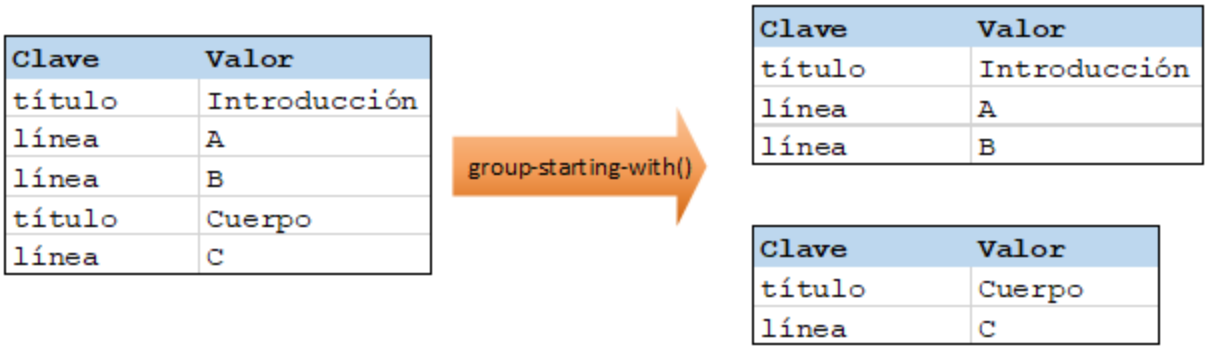
Observe que el último grupo contiene un único elemento, ya que el número total de elementos (5) no se puede dividir entre 2.

Este ejemplo, junto con otros ejemplos de agrupación, es parte de esta asignación: **<Documentos>\Altova\MapForce2023\MapForceExamples\Tutorial\GroupingFunctions.mfd**. Antes de hacer clic en la pestaña **Resultados** recuerde hacer clic en el botón **Vista previa** de la función que quiere comprobar.

6.7.9.9 group-starting-with

La función **group-starting-with** toma como argumento una condición booleana. Si se cumple la condición booleana se crea un grupo nuevo que empieza con el registro que cumple esa condición. En el siguiente ejemplo la condición es que la clave "Clave" debe ser "título". Esta condición se cumple en el caso de los registros primero y cuarto, así que se crean dos grupos:

 © 2016-2022 Altova GmbH



Nota: si existen registros antes del primero que cumple la condición, se crea un grupo más. Por ejemplo, si hubiera más registros "línea" antes del último registro "título", estos se colocarían en ese grupo adicional.

Lenguajes

La función está disponible para XSLT 2.0, XSLT 3.0, Java, C#, C++ y el motor de ejecución integrado.

Parámetros

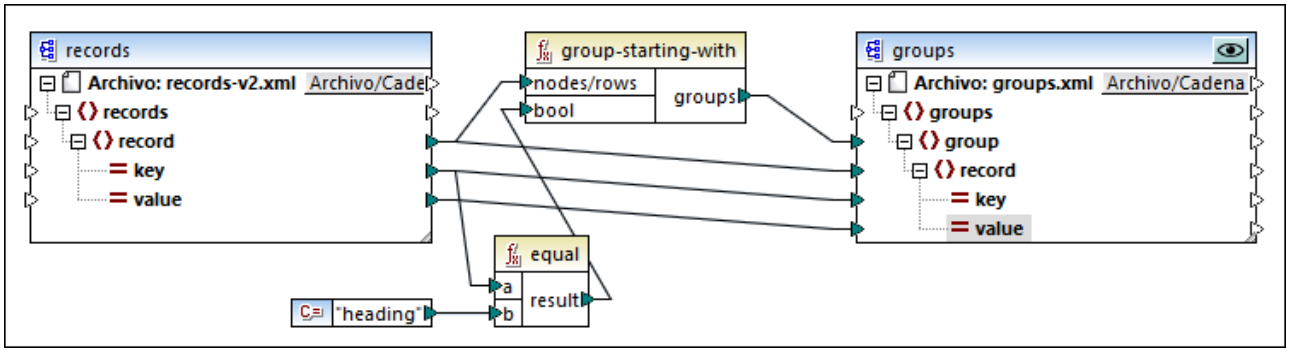
Nombre	Descripción
nodos/filas	Esta entrada debe recibir una conexión desde un elemento de la asignación que suministre una secuencia 872 de cero o más valores. Por ejemplo, la conexión puede provenir de un elemento XML de origen, un campo CSV, un registro de BD, etc.
bool	xs:boolean Indica la condición booleana que inicia un nuevo grupo si se cumple (true).

Ejemplo

Imaginemos que sus datos de origen son un archivo XML con el siguiente contenido (tenga en cuenta que, para simplificar, en el código de ejemplo siguiente hemos eliminado el espacio de nombre y las declaraciones XML).

```
<records>
 <record key="heading" value="Intro"/>
 <record key="line" value="A"/>
 <record key="line" value="B"/>
 <record key="heading" value="Body"/>
 <record key="line" value="C"/>
</records>
```

El requisito empresarial es crear un grupo para cada registro "heading". Cada grupo también debe incluir los registros "line" que siguen al registro "heading". Para ello, la siguiente asignación invoca la función **group-starting-with**. En la siguiente asignación, siempre que el nombre de **key** sea "heading", el argumento dado a **bool** pasa a ser **true** y se crea un grupo nuevo.



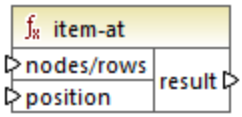
El resultado de la asignación es el siguiente:

```
<groups>
 <group>
 <record key="heading" value="Intro"/>
 <record key="line" value="A"/>
 <record key="line" value="B"/>
 </group>
 <group>
 <record key="heading" value="Body"/>
 <record key="line" value="C"/>
 </group>
</groups>
```

Este ejemplo, junto con otros ejemplos de agrupación, es parte de esta asignación: **<Documentos>\Altova\MapForce2023\MapForceExamples\Tutorial\GroupingFunctions.mfd**. Antes de hacer clic en la pestaña **Resultados** recuerde hacer clic en el botón **Vista previa** de la función que quiere comprobar.

6.7.9.10 item-at

Devuelve un elemento de la secuencia de nodos/filas dados como argumento que se encuentra en la posición indicada por el argumento **position**. El primer elemento está en la posición **1**.



Lenguajes

Built-in, C++, C#, Java, XQuery, XSLT 2.0, XSLT 3.0.

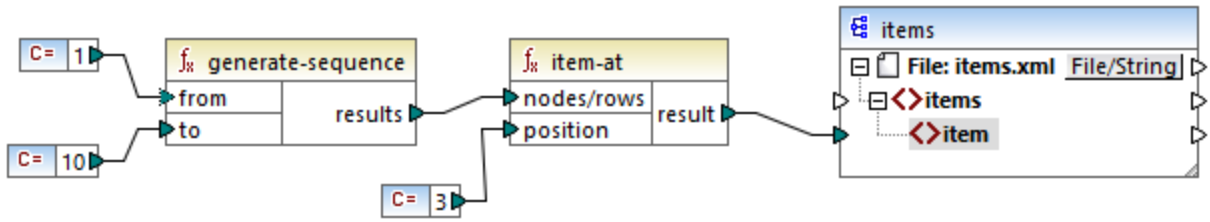
 © 2016-2022 Altova GmbH

Parámetros

Nombre	Descripción
nodos/filas	Esta entrada debe recibir una conexión desde un elemento de la asignación que suministre una secuencia 872 de cero o más valores. Por ejemplo, la conexión puede provenir de un elemento XML de origen, un campo CSV, un registro de BD, etc.
position	Este número entero indica qué elemento de la secuencia se debe devolver.

Ejemplo

La asignación de ejemplo siguiente genera una secuencia de 10 valores. Esta secuencia es procesada por la función **item-at**, que escribe el resultado en un archivo XML de destino.

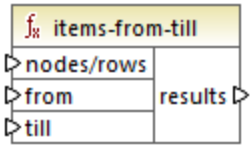


El argumento **position** tiene el valor **3**, por lo que solo el tercer valor de la secuencia pasa al archivo de destino. En consecuencia, la salida de la asignación es la siguiente (sin incluir las declaraciones de esquema XML):

```
<items>
 <item>3</item>
</items>
```

6.7.9.11 items-from-till

Devuelve una secuencia de **nodes/rows** usando los parámetros "from" y "till" como límite de la secuencia. El primer elemento está en la posición **1**.



Lenguajes

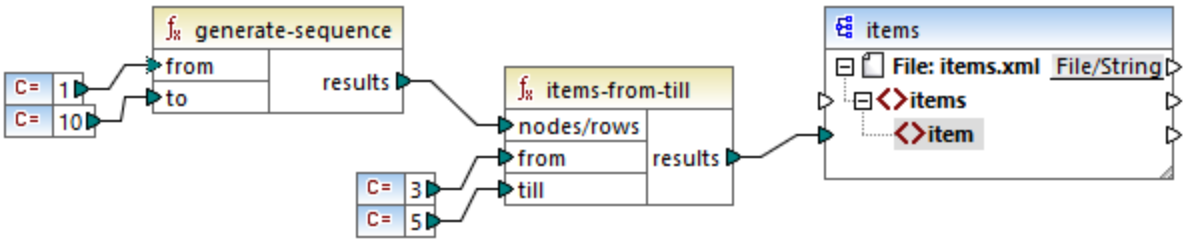
Built-in, C++, C#, Java, XQuery, XSLT 2.0, XSLT 3.0.

Parámetros

Nombre	Descripción
nodos/filas	Esta entrada debe recibir una conexión desde un elemento de la asignación que suministre una secuencia 872 de cero o más valores. Por ejemplo, la conexión puede provenir de un elemento XML de origen, un campo CSV, un registro de BD, etc.
from	Este número entero indica la posición inicial desde la que se deben recuperar los elementos.
till	Este número entero indica hasta qué posición se deben recuperar elementos.

Ejemplo

La asignación de ejemplo siguiente genera una secuencia de 10 valores. Esta secuencia es procesada por la función `items-from-till`, que escribe el resultado en un archivo XML de destino.

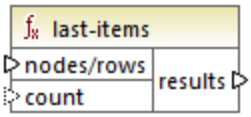


Los argumentos **from** y **till** tienen los valores **3** y **5**, por lo que solo se pasan al componente de destino el subconjunto de valores de **3** a **5**. En consecuencia, la salida de la asignación es la siguiente (sin incluir las declaraciones de esquema XML):

```
<items>
 <item>3</item>
 <item>4</item>
 <item>5</item>
</items>
```

6.7.9.12 last-items

Devuelve los *X* últimos nodos de la secuencia nodos/filas, siendo *X* el número dado por el parámetro **count**. El primer elemento está en la posición 1.



 © 2016-2022 Altova GmbH

Lenguajes

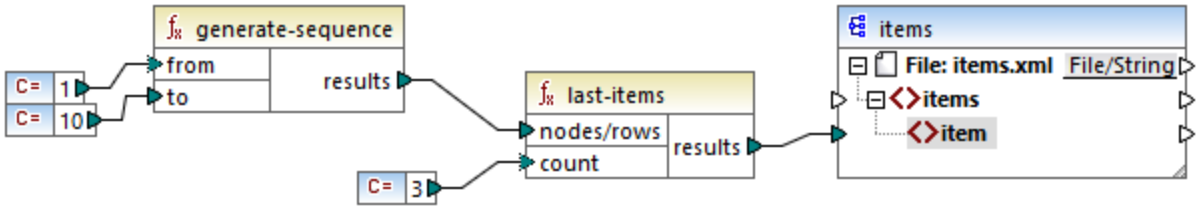
Built-in, C++, C#, Java, XQuery, XSLT 2.0, XSLT 3.0.

Parámetros

Nombre	Descripción
nodos/filas	Esta entrada debe recibir una conexión desde un elemento de la asignación que suministre una secuencia[872] de cero o más valores. Por ejemplo, la conexión puede provenir de un elemento XML de origen, un campo CSV, un registro de BD, etc.
count	Parámetro opcional. Indica cuántos nodos se deben recuperar de la secuencia de entrada. El valor predeterminado es **1**.

Ejemplo

La asignación de ejemplo siguiente genera una secuencia de 10 valores. Esta secuencia es procesada por la función `last-items`, que escribe el resultado en un archivo XML de destino.

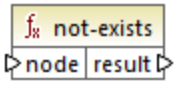


El argumento count es **3**, por lo que solo los tres últimos valores de la secuencia se pasan al archivo de destino. En consecuencia, la salida de la asignación es la siguiente (sin incluir las declaraciones de esquema XML):

```
<items>
 <item>8</item>
 <item>9</item>
 <item>10</item>
</items>
```

6.7.9.13 not-exists

Si el nodo existe, la función devuelve **false**. De lo contrario, devuelve **true**. Esta función es la contraria a la función exists[683] pero funciona de la misma manera.



Lenguajes

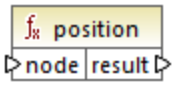
Built-in, C++, C#, Java, XQuery#, XSLT 1.0 y, XSLT 1.0, XSLT 2.0, XSLT 3.0.

Parámetros

Nombre	Descripción
nodo	El nodo cuya existencia se quiere comprobar.

6.7.9.14 position

Devuelve la posición de un nodo dentro de la secuencia de la que forma parte. Esta función se puede usar, por ejemplo, para numerar elementos automáticamente de forma secuencial.



Lenguajes

Built-in, C++, C#, Java, XQuery#, XSLT 1.0 y, XSLT 1.0, XSLT 2.0, XSLT 3.0.

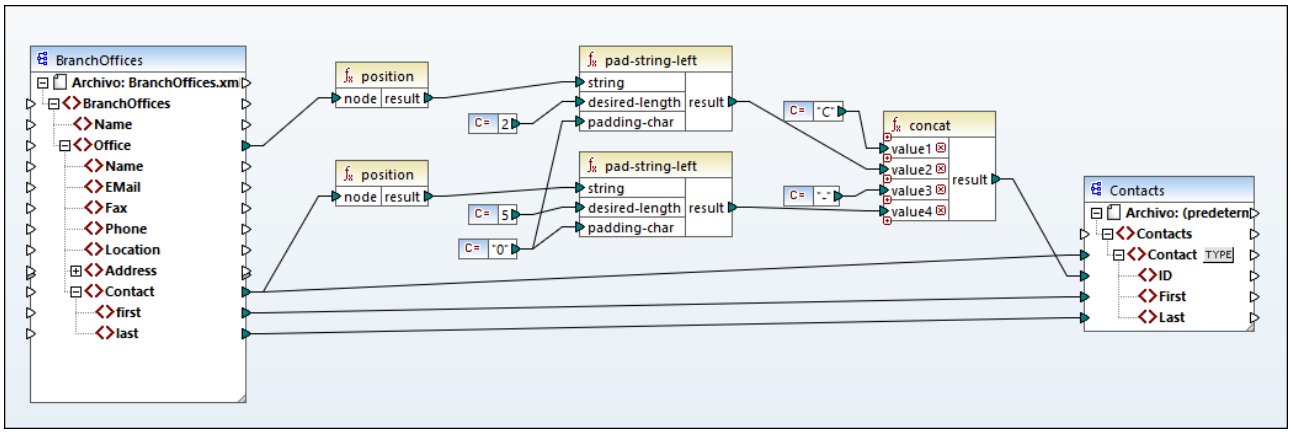
Parámetros

Nombre	Descripción
nodo	Esta entrada debe recibir una conexión desde un elemento de la asignación que suministre una secuencia 872 de cero o más valores. Por ejemplo, la conexión puede provenir de un elemento XML de origen, un campo CSV, un registro de BD, etc.

Ejemplo

La función `position` permite determinar la posición de un nodo de una secuencia o usar una posición concreta para filtrar elementos dependiendo de su posición. Esta asignación viene acompañada por un archivo de diseño de asignación que encontrará en esta ruta:
<Documentos>\Altova\MapForce2023\MapForceExamples\ContactsFromBranchOffices.mfd.

© 2016-2022 Altova GmbH



ContactsFromBranchOffices.mfd

En la asignación anterior, el archivo XML de origen contiene tres oficinas secundarias. Una oficina secundaria puede contener un número cualquiera de elementos secundarios **Contact**. Los objetivos de la asignación son los siguientes:

- Extraer todos los elementos **Contact** del archivo XML de origen y escribirlos en el archivo XML de destino.
- Asignar a cada contacto un número de identificación único (el elemento **ID** del archivo XML de destino).
- El ID de cada contacto debe tener el formato **`CXX-YYYYY`**, donde X identifica el número de oficina e Y el número de contacto. Si el número de oficina ocupa menos que dos caracteres es necesario añadirle ceros a la izquierda. Asimismo, si el número de contactos ocupa menos que cinco caracteres entera ocupe necesario añadirle ceros a la izquierda. En consecuencia, el número identificativo del primer contacto de la primera oficina sería **`C01-00001`**.

Para conseguir los objetivos de la asignación se han usado varias funciones de MapForce, entre las que está la función **`position`**. La función **`position`** obtiene la posición de cada oficina. La siguiente obtiene la posición de cada uno de los contactos dentro de cada oficina.

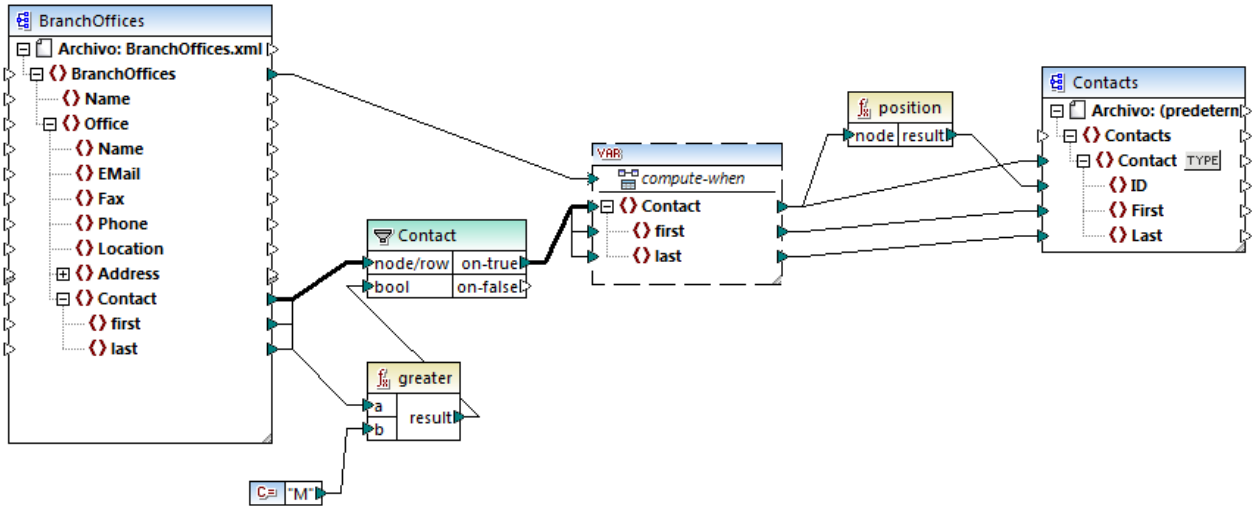
Al usar la función **`position`** es importante considerar el contexto de asignación actual 874. Concretamente, al ejecutar la asignación se establece el contexto inicial desde el elemento raíz del componente de destino al elemento de origen al que está conectado (aunque sea de forma indirecta mediante funciones). En este ejemplo, la función **`position`** de más arriba procesa la secuencia de todas las oficinas y genera inicialmente el valor 1, que corresponde a la primera oficina de la secuencia. La función **`position`** de más abajo genera números secuenciales que corresponden a la posición del contacto en el contexto de esa oficina (1, 2, 3, etc.). Tenga en cuenta que esta secuencia "interna" se reinicia y vuelve a empezar desde el 1 cuando se procesa la oficina siguiente. Las dos funciones **`pad-string-left`** usan ceros de relleno en los números generados, tal y como se explica más arriba. La función `concat` opera en el contexto de cada contacto (debido a la conexión principal del componente de entrada al **Contact** de destino). La función junta todos los valores computados y devuelve el número de identificación único de cada contacto.

A continuación mostramos el resultado que genera la asignación anterior (tenga en cuenta que hemos eliminado algunos de los registros para que ek código se pueda leer mejor):

```
<Contacts>
 <Contact>
 <ID>C01-00001</ID>
```

```
 <First>Vernon</First>
 <Last>Callaby</Last>
 </Contact>
 <Contact>
 <ID>C01-00002</ID>
 <First>Frank</First>
 <Last>Further</Last>
 </Contact>
 <!-- ... -->
 <Contact>
 <ID>C02-00001</ID>
 <First>Steve</First>
 <Last>Meier</Last>
 </Contact>
 <Contact>
 <ID>C02-00002</ID>
 <First>Theo</First>
 <Last>Bone</Last>
 </Contact>
 <!-- ... -->
</Contacts>
```

También puede haber casos en los que tenga que obtener la posición de los elementos resultantes una vez aplicado un filtro 543. Tenga en cuenta que el componente filtro no es una función de secuencia, por lo que no se puede usar directamente junto con la función **position** para buscar la posición de elementos filtrados. Esto se puede hacer de forma indirecta añadiendo un componente 492 de variable a la asignación. Por ejemplo, la asignación siguiente es una versión simplificada de la anterior. El archivo de diseño de esta asignación está en: **<Documentos>\Altova\MapForce2023\MapForceExamples\PositionInFilteredSequence.mfd**.

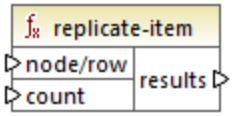


Los resultados de los componentes de variable siempre son secuencias. Por tanto, en la asignación anterior la función **position** recorre la secuencia creada por la variable y devuelve la posición de cada uno de los elementos de esa secuencia. Esta asignación se analiza más detalladamente en Ejemplo: filtrar y numerar nodos 501.

 © 2016-2022 Altova GmbH

6.7.9.15 replicate-item

Repite cada elemento de la secuencia de entrada tantas veces como se indique el argumento **count** Si conecta un único elemento a la secuencia **node/row** de entrada, la función devuelve elementos *N* donde *N* es el valor del argumento **count** Si conecta una secuencia de elementos a la secuencia de entrada **node/row**, la función repite cada elemento individual de la secuencia tantas veces como indique **count** y los procesa uno a uno. Por ejemplo, si count es **2**, entonces la secuencia `1,2,3` produce `1,1,2,2,3,3`. Repite cada elemento de la secuencia de entrada tantas veces como se indique en el argumento **count**.



Lenguajes

Built-in, C++, C#, Java, XQuery, XSLT 2.0, XSLT 3.0.

Parámetros

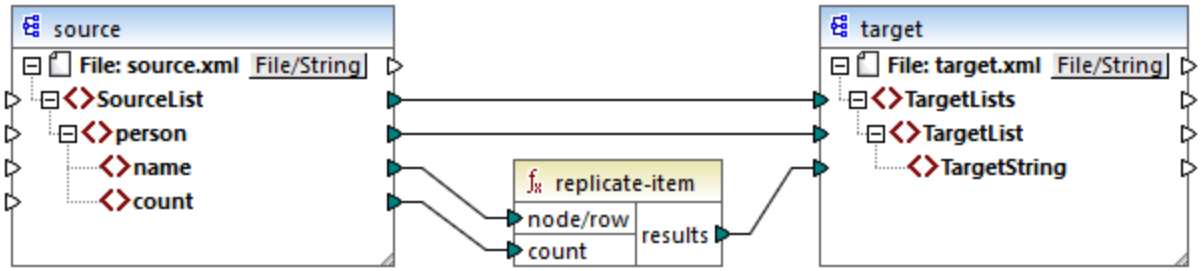
Nombre	Descripción
nodo/fila	Esta entrada debe recibir una conexión desde un elemento de la asignación que suministre una secuencia 872 de cero o más valores. Por ejemplo, la conexión puede provenir de un elemento XML de origen, un campo CSV, un registro de BD, etc.
count	Indica el número de veces que se debe replicar cada elemento o secuencia que estén conectados a **node/row**.

Ejemplo

Por ejemplo, imagine que tiene un archivo XML de origen con esta estructura:

```
<SourceList>
 <person>
 <name>Michelle</name>
 <count>2</count>
 </person>
 <person>
 <name>Ted</name>
 <count>4</count>
 </person>
 <person>
 <name>Ann</name>
 <count>3</count>
 </person>
</SourceList>
```

Con ayuda de la función **replicate-item** puede repetir cada nombre de persona las veces que quiera en el componente de destino. Para conseguirlo conecte el nodo <count> de cada persona a la entrada count de la función **replicate-item**:

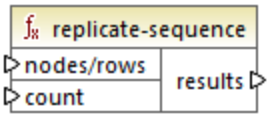


Este sería el resultado:

```
<TargetLists>
 <TargetList>
 <TargetString>Michelle</TargetString>
 <TargetString>Michelle</TargetString>
 </TargetList>
 <TargetList>
 <TargetString>Ted</TargetString>
 <TargetString>Ted</TargetString>
 <TargetString>Ted</TargetString>
 <TargetString>Ted</TargetString>
 </TargetList>
 <TargetList>
 <TargetString>Ann</TargetString>
 <TargetString>Ann</TargetString>
 <TargetString>Ann</TargetString>
 </TargetList>
</TargetLists>
```

6.7.9.16 replicate-sequence

Repite todos los elementos de la secuencia de entrada tantas veces como indique el argumento **count**. Por ejemplo, si count es **2**, entonces la secuencia (`1,2,3`) produce (`1,2,3,1,2,3`).



Lenguajes

Built-in, C++, C#, Java, XQuery, XSLT 2.0, XSLT 3.0.

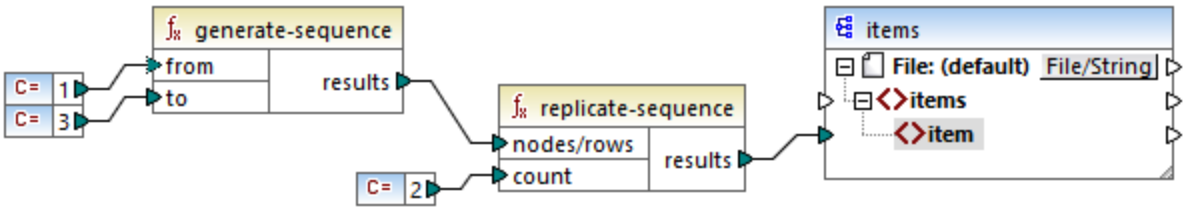
© 2016-2022 Altova GmbH

Parámetros

Nombre	Descripción
nodos/filas	Esta entrada debe recibir una conexión desde un elemento de la asignación que suministre una secuencia 872 de cero o más valores. Por ejemplo, la conexión puede provenir de un elemento XML de origen, un campo CSV, un registro de BD, etc.
count	Indica el número de veces que se debe replicar la secuencia vinculada.

Ejemplo

La asignación de ejemplo siguiente genera la secuencia `1,2,3`. Esta secuencia es procesada por la función `replicate-sequence`, que escribe el resultado en un archivo XML de destino.

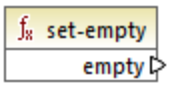


El argumento **count** tiene el valor **2**, por lo que la secuencia se replica dos veces y después pasa al archivo de destino. En consecuencia, la salida de la asignación es la siguiente (sin incluir las declaraciones de esquema XML):

```
<items>
 <item>1</item>
 <item>2</item>
 <item>3</item>
 <item>1</item>
 <item>2</item>
 <item>3</item>
</items>
```

6.7.9.17 set-empty

Devuelve una secuencia vacía.

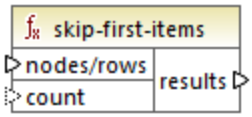


Lenguajes

Built-in, C++, C#, Java, XQuery, XSLT 2.0, XSLT 3.0.

6.7.9.18 skip-first-items

Pasa por alto los X primeros nodos/elementos de la secuencia de entrada (siendo X el número dado por el parámetro **count**) y devuelve el resto de la secuencia.



Lenguajes

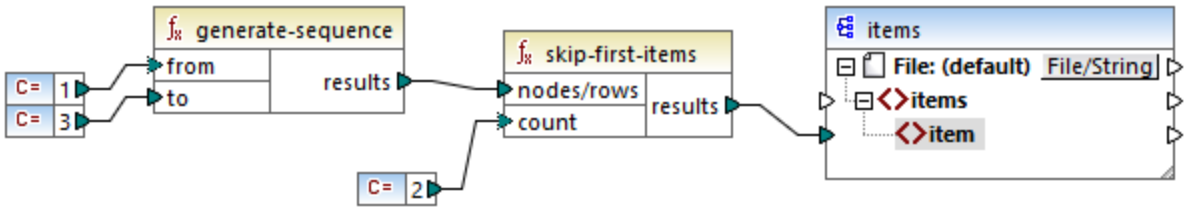
Built-in, C++, C#, Java, XQuery, XSLT 2.0, XSLT 3.0.

Parámetros

Nombre	Descripción
nodos/filas	Esta entrada debe recibir una conexión desde un elemento de la asignación que suministre una secuencia 872 de cero o más valores. Por ejemplo, la conexión puede provenir de un elemento XML de origen, un campo CSV, un registro de BD, etc.
count	Argumento opcional. Indica el número de elementos a skip. El valor predeterminado es **1**.

Ejemplo

La asignación de ejemplo siguiente genera la secuencia `1,2,3`. Esta secuencia es procesada por la función `skip-first-items`, que escribe el resultado en un archivo XML de destino.



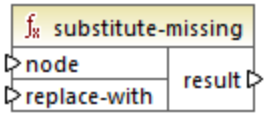
El argumento **count** tiene el valor **2**, por lo que los primeros dos elementos se omiten y el resto se pasan al archivo de destino. En consecuencia, la salida de la asignación es la siguiente (sin incluir las declaraciones de esquema XML):

```
<items>
 <item>3</item>
</items>
```


© 2016-2022 Altova GmbH

6.7.9.19 substitute-missing

Esta función es una combinación de la función exists 683 y la condición if-else. 547. Si el elemento que está conectado a la entrada del **nodo** existe, se copia su contenido en el archivo de destino. De lo contrario utiliza el elemento asignado al parámetro **replace-with**.



Lenguajes

Built-in, C++, C#, Java, XQuery#, XSLT 1.0 y, XSLT 1.0, XSLT 2.0, XSLT 3.0.

Parámetros

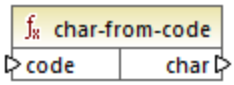
Nombre	Descripción
nodo	Esta entrada debe recibir una conexión desde un elemento de la asignación que suministre una secuencia 872 de cero o más valores. Por ejemplo, la conexión puede provenir de un elemento XML de origen, un campo CSV, un registro de BD, etc.
replace-with	Este elemento de entrada estar conectado a un elemento de la asignación que proporcione el valor de reemplazo.

6.7.10 core | string functions (cadena)

La biblioteca core | string ofrece las funciones de cadena más comunes para manipular diferentes tipos de datos de origen y extraer porciones, comprobar si existen subcadenas o recuperar información de las cadenas.

6.7.10.1 char-from-code

El resultado es el carácter que represente el valor (código) Unicode decimal del parámetro value. **Consejo:** Para encontrar el valor Unicode decimal de un carácter puede usar la función `code-from-char` 708 .



Lenguajes

Built-in, C++, C#, Java, XQuery, XSLT 2.0, XSLT 3.0.

Parámetros

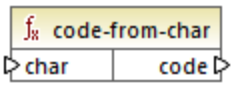
Nombre	Descripción
code	El valor Unicode como número decimal.

Ejemplo

De acuerdo con los gráficos de la página web de Unicode (https://www.unicode.org/charts/), el carácter de exclamación tiene el valor hexadecimal `0021`. El valor correspondiente en formato decimal es **33**. Por tanto, si el argumento de la función `char-from-code` es `33`, el resultado será el carácter `!`. Para ver un ejemplo consulte el apartado Reemplazar caracteres especiales 275.

6.7.10.2 code-from-char

El resultado es el valor Unicode decimal del carácter dado como argumento. Si la cadena dada como argumento tiene varios caracteres, entonces el resultado es el valor Unicode del primer carácter.



Lenguajes

Built-in, C++, C#, Java, XQuery, XSLT 2.0, XSLT 3.0.

Parámetros

Nombre	Descripción
char	El valor string de entrada.

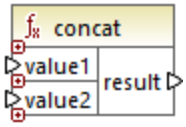
Ejemplo

Si el parámetro de entrada **char** es el carácter `$` (signo del dólar), la función devuelve **36** (que es el valor Unicode decimal de este carácrer).

6.7.10.3 concat

Concatena (anexa) dos o más valores dando lugar a una sola cadena. Todos los valores de entrada se convierten automáticamente en valores de tipo "string". Por defecto, esta función tiene solo dos parámetros, pero puede añadir más. Haga clic en el icono **Agregar parámetro** (⊞) o **Eliminar parámetro** (⊠ para añadir o borrar parámetros.

 © 2016-2022 Altova GmbH



Nota: Todos los componentes de entrada de la función `concat` deben tener un valor. Si falta el valor en alguno de ellos no se llama a la función y se genera un error. Tenga en cuenta que una cadena vacía es un valor de entrada válido; sin embargo, una secuencia vacía (como el resultado de la función **`set-empty`**) no es un valor válido, por lo que en ese caso la función fallará. Para evitar eso puede procesar primero los valores con la función substitute-missing 707 y después usar el resultado como componente de entrada de la función **`concat`**.

Lenguajes

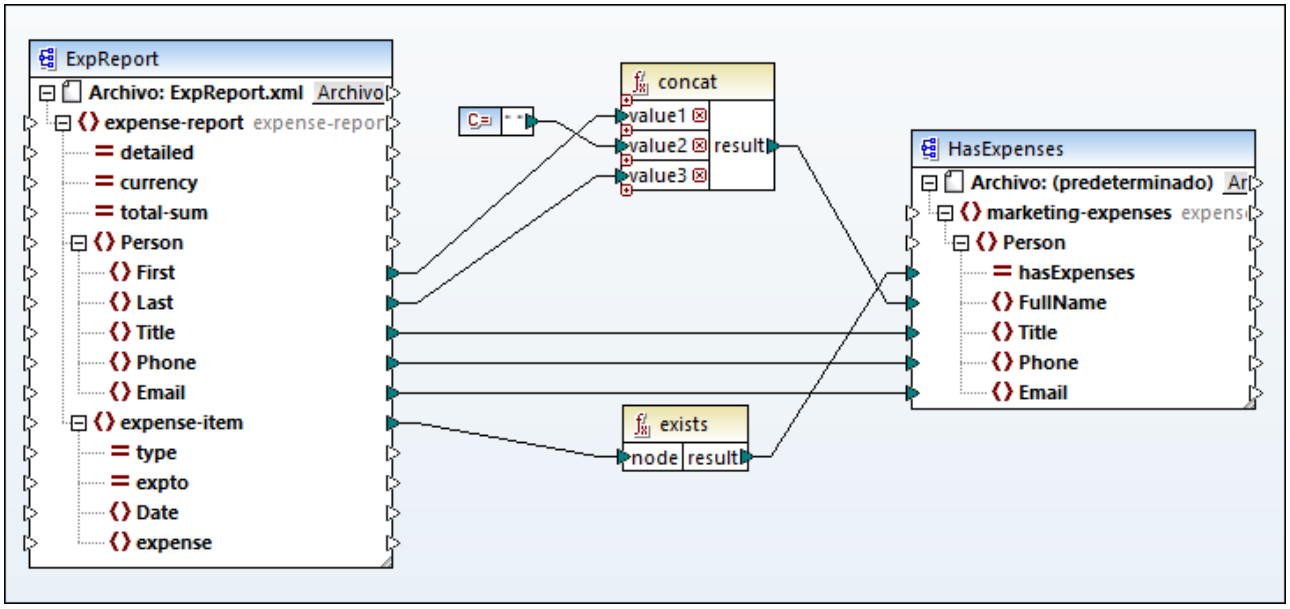
Built-in, C++, C#, Java, XQuery#, XSLT 1.0 y, XSLT 1.0, XSLT 2.0, XSLT 3.0.

Parámetros

Nombre	Descripción
value1	El primer valor de la entrada.
value2	El segundo valor de la entrada.
valueN	El valor de *n* entrada.

Ejemplo

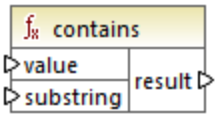
En la asignación siguiente, la función **`concat`** junta el nombre, la constante "" y el apellido. El valor de retorno se escribe en el elemento de destino **FullName**. La asignación de esta función se encuentra en: **<Documentos>\Altova\MapForce2023\MapForceExamples\HasMarketingExpenses.mfd**.



HasMarketingExpenses.mfd

6.7.10.4 contains

El resultado es **true** si la cadena de entrada (string) empieza por la subcadena (susbtr).



Lenguajes

Built-in, C++, C#, Java, XQuery#, XSLT 1.0 y, XSLT 1.0, XSLT 2.0, XSLT 3.0.

Parámetros

Nombre	Descripción
value	El valor de entrada (el "pajar").
substring	La subcadena que se busca (la "aguja").

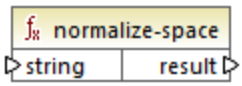
Ejemplo

El resultado es true si la entrada **value** es "category" y **substring** es "cat".

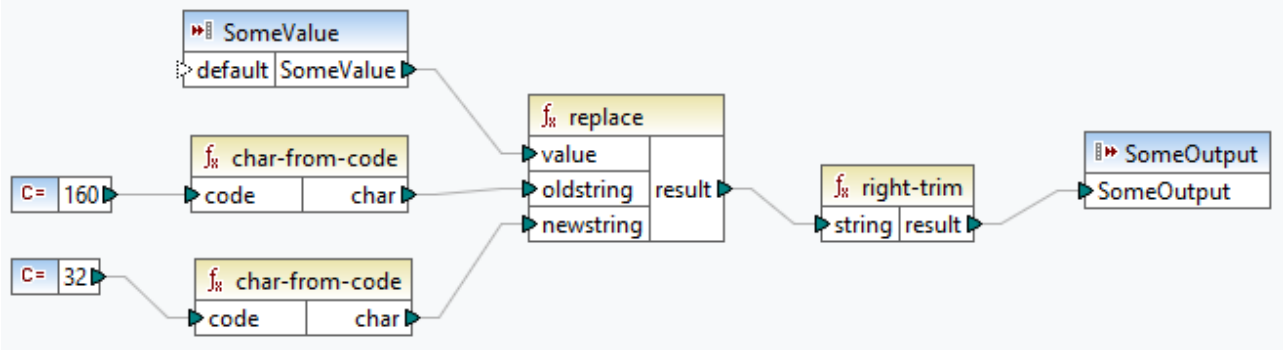
 © 2016-2022 Altova GmbH

6.7.10.5 normalize-space

El resultado es la cadena de entrada normalizada. Es decir, se eliminan los espacios iniciales y finales y cada secuencia de espacios en blancos consecutivos se reemplaza con un solo espacio en blanco. El carácter Unicode para el espacio es (U+0020).



Acerca de los espacios duros
Las funcione **`left-trim`**, **`right-trim`** y **`normalize-space`** no eliminan los espacios duros. Una posible solución es reemplazar esos caracteres, cuya representación decimal es 160, por el carácter espacio, cuya representación decimal es 32. En la asignación siguiente puede ver que, una vez se ha reemplazado el espacio duro, el valor `SomeValue` reducido se asigna al destino.



Si su componente de origen es un archivo de Excel, puede eliminar espacios extra en Excel combinando las funciones TRIM, CLEAN y SUBSTITUTE. Para más detalles consulte *Las diez formas principales de limpiar los datos*.

Lenguajes

Built-in, C++, C#, Java, XQuery, XSLT 1.0 y, XSLT 1.0, XSLT 2.0, XSLT 3.0.

Parámetros

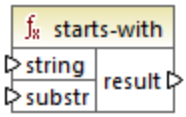
Nombre	Descripción
string	La cadena de entrada que se quiere normalizar.

Ejemplo

Si la cadena de entrada es **`El veloz zorro marrón`**, la función devuelve **`El veloz zorro marrón`**.

6.7.10.6 starts-with

El resultado es **true** si la cadena de entrada (string) empieza por la subcadena (susbtr). De lo contrario, el resultado es **false**



Lenguajes

Built-in, C++, C#, Java, XQuery#, XSLT 1.0 y, XSLT 1.0, XSLT 2.0, XSLT 3.0.

Parámetros

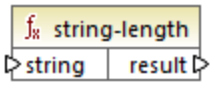
Nombre	Descripción
string	La cadena de entrada.
substr	La subcadena que se debe comprobar.

Ejemplo

Si la cadena de entrada **string** es `category` y la subcadena es `cat`, la función devuelve **true**.

6.7.10.7 string-length

El resultado es el número de caracteres asignados al parámetro de entrada string.



Lenguajes

Built-in, C++, C#, Java, XQuery#, XSLT 1.0 y, XSLT 1.0, XSLT 2.0, XSLT 3.0.

Parámetros

Nombre	Descripción
string	La cadena de entrada.

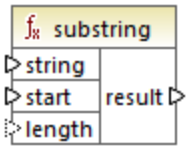
© 2016-2022 Altova GmbH

Ejemplo

Si la cadena de entrada es `car`, la función devuelve **3**. Si la cadena de entrada es **una cadena vacía** , la función devuelve **0**.

6.7.10.8 substring

El resultado es la subcadena (fragmento de cadena) del parámetro de entrada string, siendo **start** la posición del carácter de inicio y **length** la longitud de la subcadena.



Lenguajes

Built-in, C++, C#, Java, XQuery#, XSLT 1.0 y, XSLT 1.0, XSLT 2.0, XSLT 3.0.

Parámetros

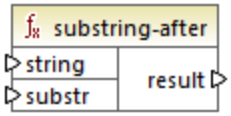
Nombre	Descripción
string	La cadena de entrada.
start	Indica la posición inicial (índice) a partir de la cual se debe obtener la subcadena. El primer índice es **1**.
length	Opcional. Indica el número de caracteres que se obtienen. Si no se indica el parámetro length, el resultado es un fragmento que empieza en la posición de inicio y termina en la posición final de la cadena.

Ejemplo

P. ej., si la cadena de entrada es `MapForce` start es **1** y **length** es **3**, la función devuelve `Map`. P. ej., si la cadena de entrada es `MapForce` start es **4** y no se indica **length**, la función devuelve `Force`.

6.7.10.9 substring-after

El resultado es el fragmento de la cadena string a partir de la primera aparición del parámetro **substr**. Si la cadena del parámetro **substr** no aparece en el parámetro **string**, el resultado de la función es una cadena vacía.



Lenguajes

Built-in, C++, C#, Java, XQuery#, XSLT 1.0 y, XSLT 1.0, XSLT 2.0, XSLT 3.0.

Parámetros

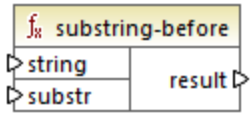
Nombre	Descripción
string	La cadena de entrada.
substr	La subcadena. El resultado de la función son los caracteres que haya después de la primera ocurrencia de **substr**.

Ejemplo

P. ej., si la cadena de entrada es `MapForce`, y **substr** es `Map`, la función devuelve `Force`. P. ej., si la cadena de entrada es `2020/01/04` y **substr** es `/` da como resultado la subcadena `01/04`.

6.7.10.10 substring-before

El resultado es el fragmento de la cadena del parámetro string hasta la primera aparición de los caracteres del parámetro **substr**. Si la cadena del parámetro **substr** no aparece en el parámetro **string**, el resultado de la función es una cadena vacía.



Lenguajes

Built-in, C++, C#, Java, XQuery#, XSLT 1.0 y, XSLT 1.0, XSLT 2.0, XSLT 3.0.

Parámetros

Nombre	Descripción
string	La cadena de entrada.
substr	La subcadena. Los caracteres que hay antes de la primera aparición de **substr** son el resultado de la función.

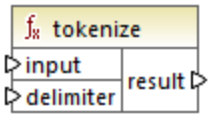
© 2016-2022 Altova GmbH

Ejemplo

P. ej., si la cadena de entrada es **MapForce**, y **substr** es **Force**, la función devuelve **Map** . Si la cadena de entrada es **2020/01/04** y **substr** es **/** da como resultado la subcadena **2020**.

6.7.10.11 tokenize

El resultado es la cadena input dividida en una secuencia formada por caracteres delimitados por el parámetro delimiter.



Lenguajes

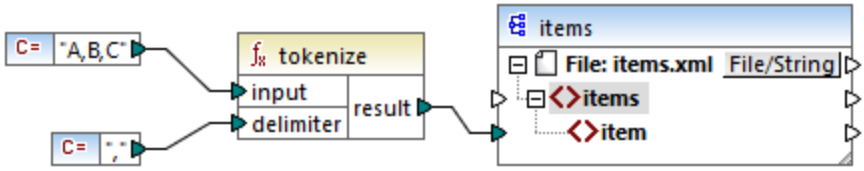
Built-in, C++, C#, Java, XQuery, XSLT 2.0, XSLT 3.0.

Parámetros

Nombre	Descripción
input	La cadena de entrada.
delimiter	El delimitador que se usa.

Ejemplo

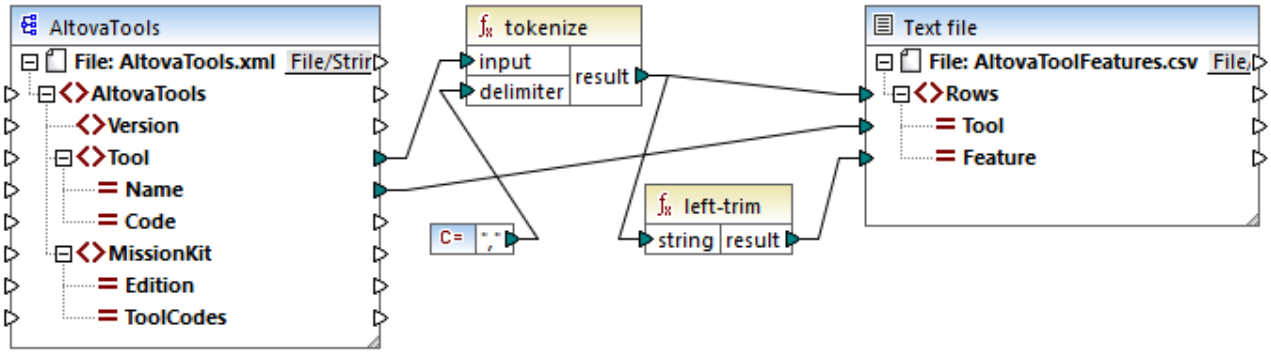
P. ej. la cadena input es **A,B,C** y el delimitador es ",". El resultado será **A**, **B** y **C**.



En la asignación de prueba de más arriba, el resultado de la función es una secuencia de cadenas. Según las reglas generales de asignación, 872 por cada elemento de la secuencia de origen se debe crear un **elemento** nuevo en el componente de destino. En consecuencia, el resultado de la asignación tiene este aspecto:

```
<items>
 <item>A</item>
 <item>B</item>
 <item>C</item>
</items>
```

En el diseño de asignación de datos **tokenizeString1.mfd** de la carpeta ...\MapForceExamples puede ver un ejemplo de uso de la función tokenize-by-length.



tokenizeString1.mfd

A continuación mostramos un extracto del archivo XML de origen. El elemento **Tool** tiene dos atributos: **Name** y **Code**. El elemento **Tool** consiste en un texto delimitado por comas.

```
<?xml version="1.0" encoding="UTF-8"?>
<AltovaTools xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="AltovaTools.xsd">
 <Version>2010</Version>
 <Tool Name="XMLSpy" Code="XS">XML editor, XSLT editor, XSLT debugger, XQuery editor,
XQuery debugger, XML Schema / DTD editor, WSDL editor, SOAP debugger</Tool>
 <Tool Name="MapForce" Code="MF">Data integration, XML mapping, database mapping, text
conversion, EDI translator, Excel mapping, XBRL mapping, Web services</Tool>
 <Tool Name="StyleVision" Code="SV">Stylesheet designer, electronic forms, XSLT design,
XSL:FO design, database reporting, XBRL rendering</Tool>
 <Tool Name="UModel" Code="UM">UML modeling tool, code generation, reverse engineering,
UML, BPMN, SysML, project documentation, XMI interchange</Tool>
 <Tool Name="DatabaseSpy" Code="DS">Multi-database tool, SQL auto-completion, graphical
database design, table browser, content editor, database comparison tool</Tool>
 <!-- ... -->
</AltovaTools>
```

Esta asignación hace lo siguiente:

- La función **tokenize** recibe datos del elemento **Tool** y utiliza el delimitador "**,**" para dividir los datos en secciones (p. ej. la primera sección es "XML editor").
- Como el parámetro result está asignado al elemento Rows del componente de destino, se genera una fila por cada sección. Esto ocurre gracias a la conexión entre el resultado de la función y el elemento **Rows** del componente de destino.
- El parámetro result también está asignado a la función **left-trim**, que elimina los espacios en blanco iniciales de cada sección.
- El resultado del parámetro **left-trim** se asigna al elemento **Feature** del componente de destino.
- El archivo de salida del componente de destino se definió como archivo CSV (**AltovaToolFeatures.csv**) con el delimitador de campos punto y coma (haga doble clic en el componente para ver su configuración).

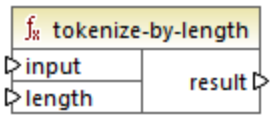
 © 2016-2022 Altova GmbH

El resultado de la asignación de datos es que se crea una fila nueva en el archivo CSV de destino por cada sección que crea la función **tokenize**. Este es un extracto del resultado de la asignación:

```
Tool;Feature
XMLSpy;XML editor
XMLSpy;XSLT editor
XMLSpy;XSLT debugger
XMLSpy;XQuery editor
XMLSpy;XQuery debugger
XMLSpy;XML Schema / DTD editor
XMLSpy;WSDL editor
XMLSpy;SOAP debugger
MapForce;Data integration
MapForce;XML mapping
MapForce;database mapping
MapForce;text conversion
MapForce;EDI translator
MapForce;Excel mapping
```

6.7.10.12 tokenize-by-length

El resultado es la cadena de entrada dividida en una secuencia de cadenas, El parámetro **length** determina el tamaño de las cadenas resultantes.



Lenguajes

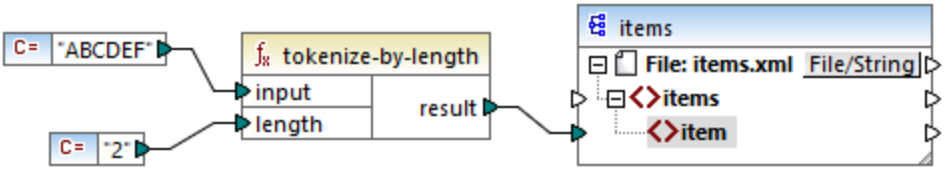
Built-in, C++, C#, Java, XQuery, XSLT 2.0, XSLT 3.0.

Parámetros

Nombre	Descripción
input	La cadena de entrada.
length	Determina la longitud de cada una de las cadenas que forman la secuencia de cadenas generada.

Ejemplo

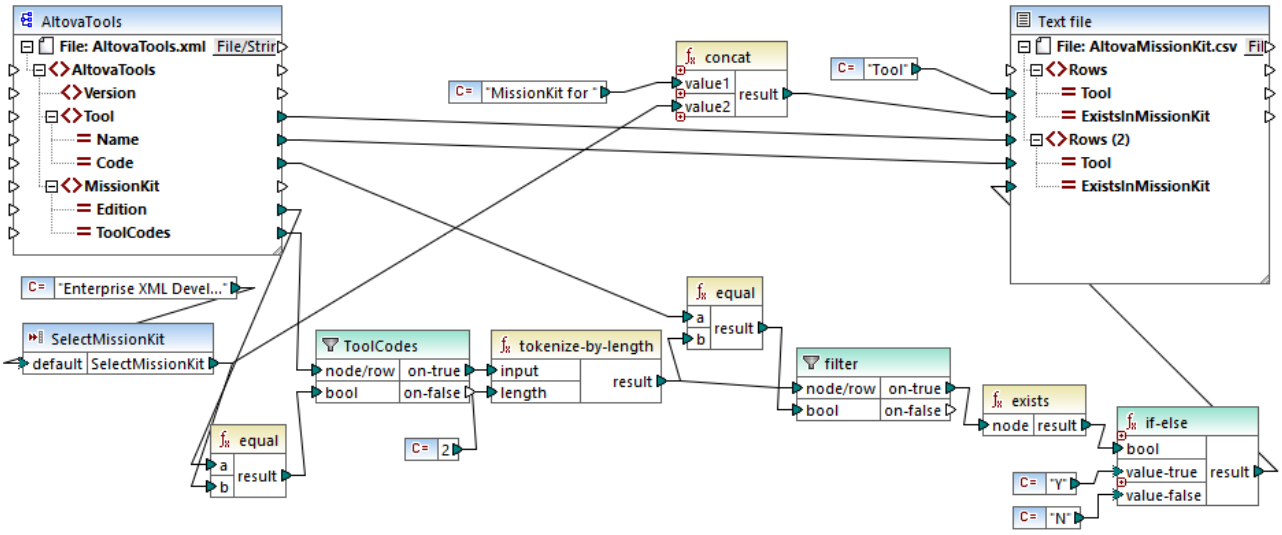
P. ej., si la cadena input es **ABCDEF** y el parámetro length es **2**, entonces la función devuelve una secuencia de tres cadenas: **AB**, **CD** y **EF**.



En la asignación de prueba de más arriba, el resultado de la función es una secuencia de cadenas. Según las reglas generales de asignación, 872 por cada elemento de la secuencia de origen se debe crear un **elemento** nuevo en el componente de destino.. En consecuencia, el resultado de la asignación tiene este aspecto:

```
<items>
 <item>AB</item>
 <item>CD</item>
 <item>EF</item>
</items>
```

En el diseño de asignación de datos **tokenizeString2.mfd** de la carpeta ...\MapForceExamples puede ver un ejemplo de uso de la función tokenize-by-length.



tokenizeString2.mfd

El archivo XML de origen se puede ver más abajo y es el mismo que el del ejemplo anterior. El elemento **MissionKit** tiene dos atributos: **Edition** y **ToolCodes**, pero no contenido del elemento MissionKit. Hemos eliminado la parte del contenido XML no relevante para este ejemplo.

```
<?xml version="1.0" encoding="UTF-8"?>
<AltovaTools xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="AltovaTools.xsd">
 <Version>2010</Version>
 <Tool Name="XMLSpy" Code="XS"><!--...--></Tool>
 <Tool Name="MapForce" Code="MF"><!--...--></Tool>
```

```
 <Tool Name="StyleVision" Code="SV"><!--...--></Tool>
 <Tool Name="UModel" Code="UM"><!--...--></Tool>
 <Tool Name="DatabaseSpy" Code="DS"><!--...--></Tool>
 <Tool Name="DiffDog" Code="DD"><!--...--></Tool>
 <Tool Name="SchemaAgent" Code="SA"><!--...--></Tool>
 <Tool Name="SemanticWorks" Code="SW"><!--...--></Tool>
 <Tool Name="Authentic" Code="AU"><!--...--></Tool>
 <MissionKit Edition="Enterprise Software Architects" ToolCodes="XSMFSVUMDSDDSASW"/>
 <MissionKit Edition="Professional Software Architects" ToolCodes="XSMFSVUMDS"/>
 <MissionKit Edition="Enterprise XML Developers" ToolCodes="XSMFSVDDSASW"/>
 <MissionKit Edition="Professional XML Developers" ToolCodes="XSMFSV"/>
</AltovaTools>
```

El objetivo de la asignación es generar una lista que muestre qué herramientas de Altova forman parte de cada edición de MissionKit.

Características de esta asignación de datos:

- El componente de entrada **SelectMissionKit** recibe su entrada predeterminada de una constante ("Enterprise XML Developers").
- La función **equal** compara el valor de entrada con el valor **Edition** y pasa el resultado al parámetro **bool** del filtro ToolCodes.
- La entrada **node/row** del filtro ToolCodes viene dada por el elemento **ToolCodes** del archivo de origen. El valor para la edición "Enterprise XML Developers" es: **XSMFSVDDSASW**.
- El valor **XSMFSVDDSASW** se envía al parámetro **on-true** y después al parámetro **input** de la función **tokenize-by-length**.
- El valor de entrada ToolCodes es **XSMFSVDDSASW** y se divide en varias secciones de dos caracteres cada una (definidas por el parámetro **length** que es **2**, lo cual da lugar a 6 secciones).
- Cada sección (situada en el parámetro b) de la función equal se compara con el valor **Code** de dos caracteres del archivo de origen (del cual existen 9 entradas en total). El resultado de la comparación (true/false) se envía al parámetro **bool** del filtro. Todas las secciones de la función **tokenize-by-length** se envían al parámetro **node/row** del filtro.
- La función **exists** busca los nodos que existen o que faltan que recibió del parámetro **on-true** del componente de filtrado. Los nodos que existen son aquellos en los que la sección **ToolCodes** y el valor **Code** coinciden. Los nodos que faltan son aquellos donde ninguna sección **ToolCodes** coincide con ningún valor **Code**.
- Los resultados **bool** de la función **exists** se pasan a la función **if-else** que pasa una Y al destino si el nodo existe y una N si no existe.

Resultado de la asignación:

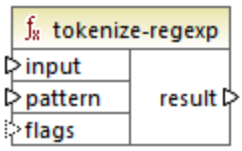
```
Tool;MissionKit for Enterprise XML Developers
XMLSpy;Y
MapForce;Y
StyleVision;Y
UModel;N
DatabaseSpy;N
DiffDog;Y
SchemaAgent;Y
```

```
SemanticWorks;Y
Authentic;N
```

6.7.10.13 tokenize-regexp

El resultado es la cadena de entrada dividida en una secuencia de cadenas, en la que la expresión regular **pattern** define el separador. El parámetro result no genera los separadores.

Nota: al generar código C++, C# o Java, las características avanzadas de la sintaxis de la expresión regular pueden variar ligeramente. Consulte la documentación regex de cada lenguaje para más información.



Lenguajes

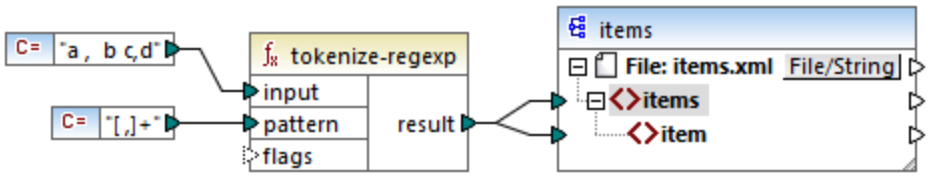
Built-in, C++, C#, Java, XQuery, XSLT 2.0, XSLT 3.0.

Parámetros

Nombre	Descripción
input	La cadena de entrada.
pattern	Indica una expresión regular pattern. Las cadenas que coincidan con esa expresión regular pattern se consideran delimitadores. Para más información consulte Expresiones regulares 627.
flags	Parámetro opcional. Indica expresiones regulares flags 629 . Por ejemplo, la expresión regular flag "i" indica al proceso de la asignación que opere sin tener en cuenta mayúsculas y minúsculas.

Ejemplo

El objetivo de la asignación siguiente es separar la cadena `a , b c,d` en una secuencia de cadenas en la que cada letra es un elemento de la secuencia. Los espacios o las comas redundantes deben eliminarse.



Para ello se pasa la expresión regular `[ ,]+` como parámetro a la función `tokenize-regexp`. Al usar esa expresión regular:

 © 2016-2022 Altova GmbH

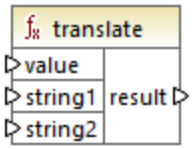
- La expresión regular pattern define una clase de caracteres `[ ,]`, de los cuales se usará un solo carácter (es decir, o bien el espacio, o bien la coma).
- El cuantificador `+` especifica "una o más" apariciones de la clase de caracteres/cadena. Sin este cuantificador, en la secuencia resultante se crearía un elemento por cada espacio o coma, que no es lo que pretendemos.

El resultado de la asignación sería:

```
<items>
 <item>a</item>
 <item>b</item>
 <item>c</item>
 <item>d</item>
</items>
```

6.7.10.14 translate

Reemplaza caracteres uno a uno. Los caracteres de la cadena **string1** (cadena de búsqueda) se reemplazan con los caracteres de la cadena **string2** (cadena de reemplazo) en la cadena de entrada **value**. Si en la cadena **string2** no hay caracteres que correspondan, entonces se elimina el carácter.



Lenguajes

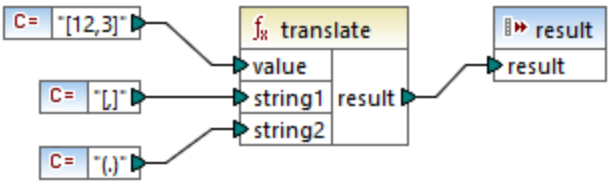
Built-in, C++, C#, Java, XQuery#, XSLT 1.0 y, XSLT 1.0, XSLT 2.0, XSLT 3.0.

Parámetros

Nombre	Descripción
value	La cadena de entrada.
string1	Suministra una lista de caracteres de búsqueda. La posición de cada carácter en la cadena es importante.
string2	Suministra una lista de caracteres de reemplazo. La posición de cada uno de los caracteres de reemplazo debe tener su correspondencia en la cadena **string1**.

Ejemplo

Imagine que quiere convertir la cadena `[12,3]` en `(12.3)`. Es decir, quiere reemplazar los corchetes por paréntesis y la coma por un punto. Para obtener ese resultado puede usar la función `translate`:



En la asignación anterior, la primera constante pasa la cadena de entrada que se debe procesar. La segunda y la tercera constantes pasan una lista de caracteres como **string1** y **string2** respectivamente.

string1	`[,]`
string2	`(.)`

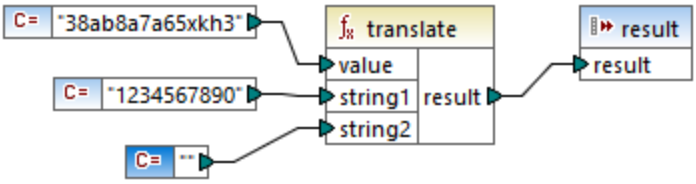
Tenga en cuenta que tanto **string1** como **string2** tienen el mismo número de caracteres. Cada carácter de **string1** se reemplaza con el carácter que esté en esa misma posición en **string2**. Por tanto, en este caso lo que ocurrirá es que:

- Cada carácter `[` será reemplazado por un carácter `(`
- Cada carácter `,` será reemplazado por un carácter `.`
- Cada carácter `]` será reemplazado por un carácter `]`

El resultado de la asignación sería:

```
(12.3)
```

Esta función también se puede usar para eliminar determinados caracteres de una cadena. Para ello, indique con el parámetro **string1** los caracteres que quiere eliminar y deje vacío el parámetro **string2**. Por ejemplo, esta asignación elimina tods los dígitos de la cadena `38ab8a7a65xkh3`.



El resultado de la asignación sería:

```
abaaxkh
```

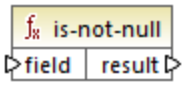
6.7.11 db (base de datos)

La biblioteca **db** contiene funciones que sirven para definir los resultados de la asignación cuando se encuentran campos NULL en bases de datos.

© 2016-2022 Altova GmbH

6.7.11.1 is-not-null

Si el campo es NULL, la función devuelve **false**. De lo contrario devuelve **true**.



Lenguajes

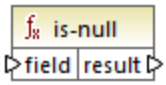
Disponible para Java, C#, C++ y el motor de ejecución integrado.

Parámetros

Nombre	Descripción
field	El campo de la BD.

6.7.11.2 is-null

Si el campo es NULL, la función devuelve **true** . De lo contrario devuelve **false** .



Lenguajes

Disponible para Java, C#, C++ y el motor de ejecución integrado.

Parámetros

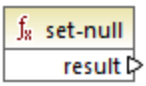
Nombre	Descripción
field	El campo de la BD.

6.7.11.3 set-null

Se utiliza para configurar una columna de la BD como NULL. Esta función también sobrescribe el valor predeterminado de la columna con NULL. Si no se conecta a un campo de BD sino a otro componente, funciona como una secuencia vacía. Nota:

- Si conecta **`set-null`** a otra función, no se llama a la otra función en absoluto. Si conecta **`set-null`** a una función de secuencia (p. ej. **`count`**), se llama a la función con una secuencia vacía.

- Si conecta **`set-null`** a funciones especiales, los filtros y las condiciones IF-Else funcionan normalmente y los campos se configuran como NULL. En los filtros esto corresponde a la entrada node/row.
- Si usa **`set-null`** como entrada para un elemento `simpleType`, no se creará ese elemento en el componente de destino.

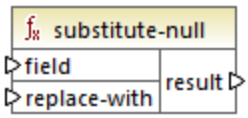


Lenguajes

Disponible para Java, C#, C++ y el motor de ejecución integrado.

6.7.11.4 substitute-null

Sirve para asignar el contenido del campo actual si es que existe. De lo contrario se utiliza el elemento asignado al parámetro **replace-with**.



Lenguajes

Disponible para Java, C#, C++ y el motor de ejecución integrado.

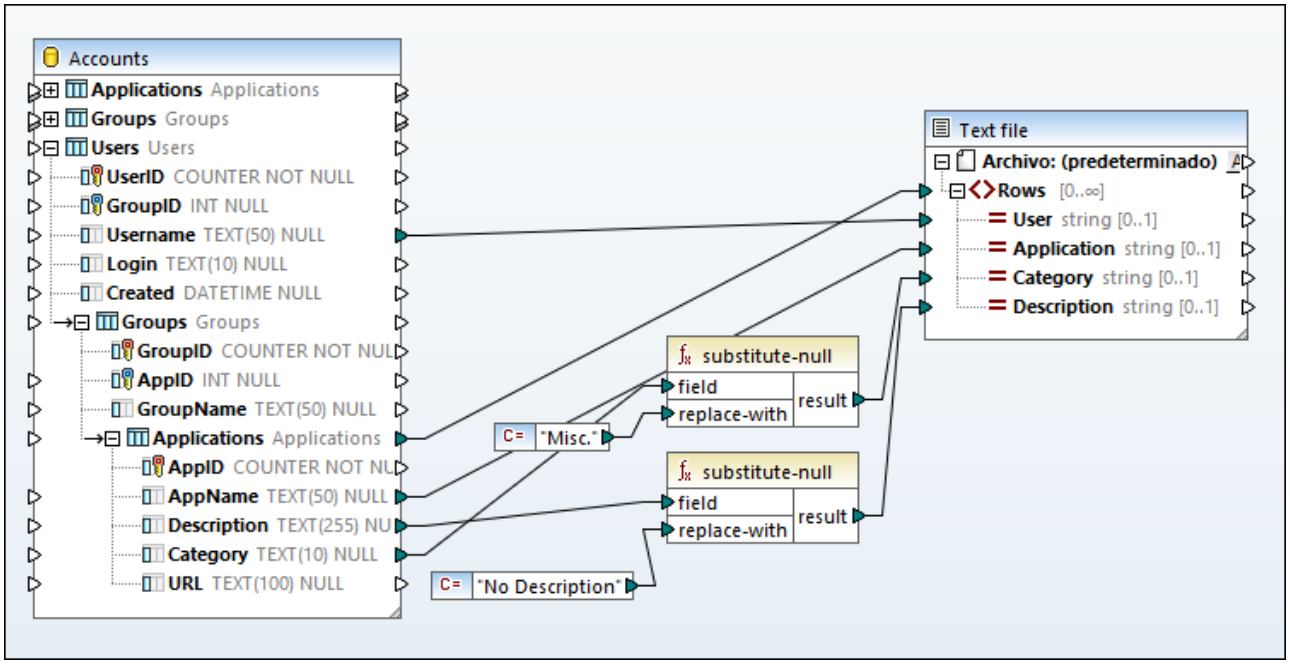
Parámetros

Nombre	Descripción
field	El campo de la BD.
replace-with	El valor de reemplazo.

Ejemplo

La asignación siguiente muestra un ejemplo de la función **`substitute-null`** en uso. Esta asignación se llama **DB_ApplicationList.mfd** y está en la carpeta **<Documentos>\Altova\MapForce2023\MapForceExamples**.

 © 2016-2022 Altova GmbH



La asignación lee datos de una BD Access que contiene la tabla "Applications".

	AppID	AppName	Description	Category	URL
1	1	Altova MapForce	The premier data mapping tool.	IDE	www.altova.com/xmlspy
2	2	Notepad	*[NULL]*	*[NULL]*	*[NULL]*

La primera función comprueba si existe la entrada **Category** en la tabla "Applications". Puesto que **Category** no existe en la aplicación Notepad, se asigna la cadena de texto Misc. al elemento **Category** del archivo de texto.

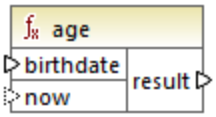
La segunda función comprueba si existe la entrada **Description** y, si no existe, asigna la cadena de texto No description.

6.7.12 lang | datetime functions (fechaHora)

Las funciones date y time de la biblioteca **lang** sirven para manipular fechas, horas y duraciones. A diferencia de las funciones date y time de la biblioteca **core**, estas funciones solo están disponibles para Java, C#, C++ o el motor de ejecución integrado.

6.7.12.1 age

El resultado es la cantidad total de años comprendidos entre la fecha de nacimiento y la fecha actual.



Lenguajes

Disponible para Java, C#, C++ y el motor de ejecución integrado.

Parámetros

Nombre	Tipo	Descripción
birthdate	`xs:date`	Obligatorio. Indica la fecha de nacimiento como valor `xs:date`.
now	`xs:date`	Parámetro opcional. El valor predeterminado es la fecha actual del sistema. Si hay un valor asignado al argumento now, el resultado es el número de años comprendido entre ambas fechas.

6.7.12.2 convert-to-utc

Convierte el parámetro de entrada local time en la hora universal coordinada o GMT. La función tiene en cuenta el componente de uso horario, p. ej. +5:00.

Lenguajes

Disponible para Java, C#, C++ y el motor de ejecución integrado.

Parámetros

Nombre	Tipo	Descripción
time	`xs:dateTime`	Indica el valor `xs:dateTime` que se debe convertir.

Ejemplo

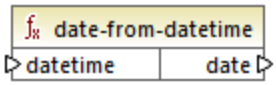
Si el valor de entrada es **`2001-12-17T09:30:02+05:00`**, entonces el resultado de la función es **`2001-12-17T04:30:02`**.

© 2016-2022 Altova GmbH

Si el valor de entrada es **2001-12-17T09:30:02Z**, entonces el resultado de la función es **2001-12-17T09:30:02**. En este caso no hay conversión porque la Z final ya indica la Hora Zulú, que es igual que la hora universal coordinada.

6.7.12.3 date-from-datetime

El resultado es la parte *date* del argumento de entrada `datetime`. La parte *time* del dateTime, empezando por la T del documento de instancia, se fija en 0. Observe que el incremento de uso horario no se cambia.



Lenguajes

Disponible para Java, C#, C++ y el motor de ejecución integrado.

Parámetros

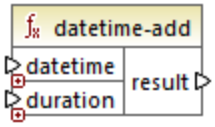
Nombre	Tipo	Descripción
datetime	**xs:dateTime**	Indica el valor `xs:dateTime` que se debe procesar.

Ejemplo

Si el valor de entrada es **2001-12-17T09:30:02+05:00**, entonces el resultado de la función es **2001-12-17T00:00:00+05:00**.

6.7.12.4 datetime-add

El resultado es el valor datetime que se obtiene al añadir un valor duration (segundo argumento) a un valor datetime (primer argumento).



Lenguajes

Disponible para Java, C#, C++ y el motor de ejecución integrado.

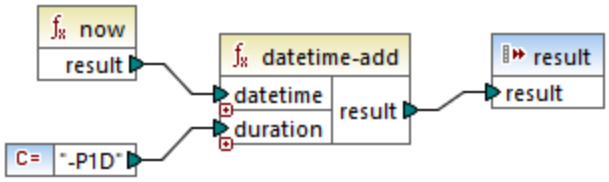
Parámetros

Nombre	Tipo	Descripción
datetime	`xs:dateTime`	Indica el valor `xs:dateTime` que se debe usar como entrada.
duration	`xs:duration`	Indica el valor de `xs: duration`. Un ejemplo del parámetro `duration` es **`P1Y2M3DT04H05M59S`**, donde: • "P" designa el periodo y es obligatorio; • El resto de caracteres representa, en este orden: 1 año, 2 meses, 3 días, T (indica el tiempo), 04 horas, 05 minutos, 59 segundos. Si aparece el caracter menos (-) antes de "P" la duración se entiende como negativa, por ejemplo: **`-P1D`**.

Ejemplo

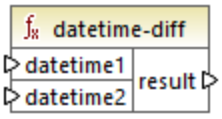
Imagine que el valor de entrada **datetime** es **`2001-12-17T09:30:02+05:00`**. Si **duration** es **`P10D`** (10 días), entonces el resultado de la función es **`2001-12-27T00:00:00+05:00`**.

Para extraer la fecha de ayer de la entrada dateTime, use la función **now** para introducir la fecha/hora actual incluido el uso horario. El período se puede convertir en negativo usando el carácter - antes del designador P (por ejemplo -P1D, menos 1 día).



6.7.12.5 datetime-diff

El resultado es la duración que se obtiene al sustraer **datetime2** (segundo argumento) a **datetime1** (primer argumento). El resultado se puede asignar a un valor string, duration o datetype.



© 2016-2022 Altova GmbH

Lenguajes

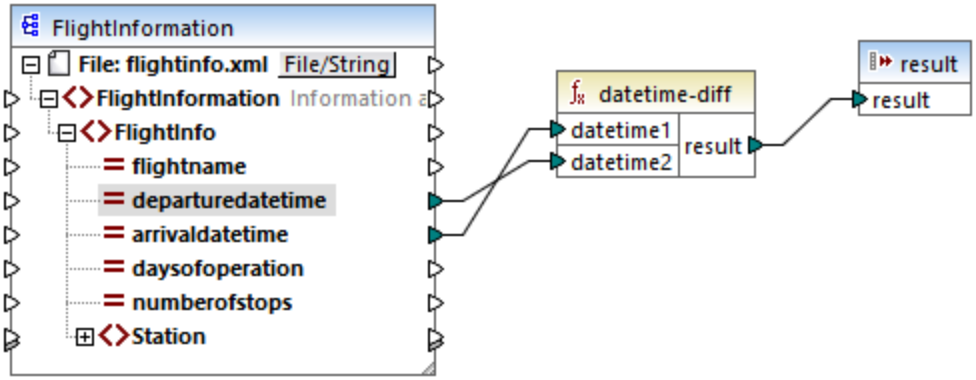
Disponible para Java, C#, C++ y el motor de ejecución integrado.

Parámetros

Nombre	Tipo	Descripción
datetime1	`xs:dateTime`	Indica el primer valor `xs:dateTime`.
datetime2	`xs:duration`	Indica el segundo valor `xs:dateTime`.

Ejemplo

En la asignación siguiente, la función **`datetime-diff`** resta la hora de salida del vuelo **`2001-12-17T09:30:02+05:00`** a la hora de llegada **`2001-12-17T19:30:02+05:00`**. Tenga en cuenta que la hora de salida es el valor mayor, por lo que está conectado a la primera entrada de la función.

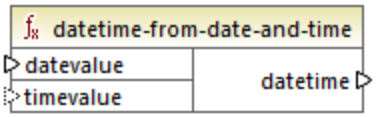


El resultado de la asignación es la diferencia entre las dos horas (que es un periodo de 10 horas).

```
PT10H
```

6.7.12.6 datetime-from-date-and-time

El resultado es un valor `xs:dateTime` creado a partir del argumento `xs:date` (primer argumento) y el argumento `xs:time` (segundo argumento). El resultado se puede asignar a un tipo de datos string o `dateTime`.



Lenguajes

Disponible para Java, C#, C++ y el motor de ejecución integrado.

Parámetros

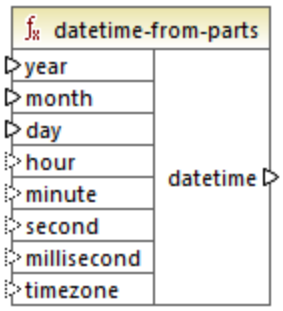
Nombre	Tipo	Descripción
datevalue	`xs:date`	Obtiene el valor de tipo `xs: date`
timevalue	`xs:time`	Obtiene el valor de tipo `xs: time`

Ejemplo

Si el primer argumento es **`2012-06-29`** y el segundo argumento es **`11:59:55`**, la función devuelve **`2012-06-29T11:59:55`**.

6.7.12.7 datetime-from-parts

El resultado es un valor datetime creado uniendo cualquiera de estos argumentos: year, month, day, hour, minute, second, millisecond y timezone. Esta función normaliza automáticamente los parámetros suministrados. Por ejemplo, 32 de enero se convierte automáticamente en 1 de febrero.



Lenguajes

Disponible para Java, C#, C++ y el motor de ejecución integrado.

Parámetros

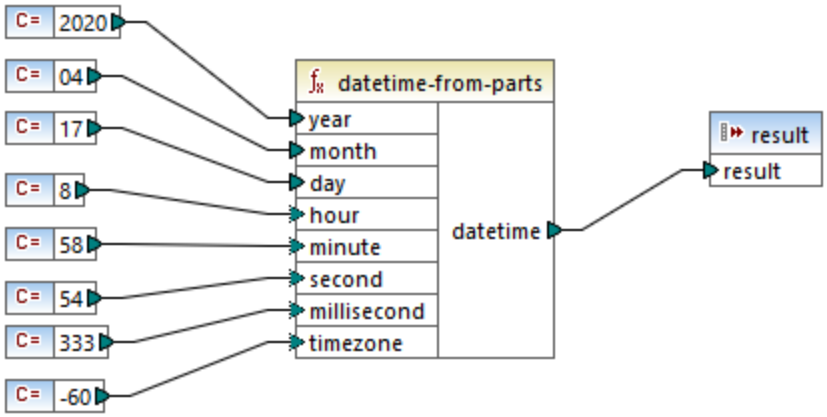
Nombre	Tipo	Descripción
year	`xs:int`	El año.
month	`xs:int`	El mes.
day	`xs:int`	El día del mes.

© 2016-2022 Altova GmbH

Nombre	Tipo	Descripción
hour	`xs:int`	Opcional. La hora.
minute	`xs:int`	Opcional. El minuto.
second	`xs:int`	Opcional. El segundo.
millisecond	`xs:decimal`	Opcional. El milisegundo.
timezone	`xs:int`	Opcional. Suministra el uso horario en minutos. Puede ser un valor negativo.

Ejemplo

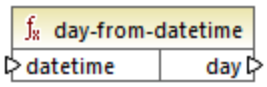
La asignación siguiente genera un valor `xs:dateTime` a partir de partes dadas por constantes.



El resultado de la asignación es `2020-04-17T08:58:54.333-01:00`.

6.7.12.8 day-from-datetime

El resultado es el día del argumento dateTime.



Lenguajes

Disponible para Java, C#, C++ y el motor de ejecución integrado.

Parámetros

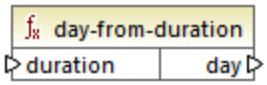
Nombre	Tipo	Descripción
datetime	`xs:dateTime`	Obtiene el valor de la entrada de tipo `xs: dateTime`.

Ejemplo

Si **datetime** es `2001-12-17T10:30:03+01:00`, entonces la función devuelve `17`.

6.7.12.9 day-from-duration

El resultado es el componente day del argumento duration.



Lenguajes

Disponible para Java, C#, C++ y el motor de ejecución integrado.

Parámetros

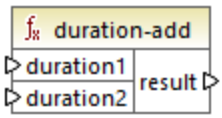
Nombre	Tipo	Descripción
duration	`xs:duration`	Obtiene el valor de la entrada de tipo `xs:duration`.

Ejemplo

Si **duration** es `P1Y2M3DT10H30M`, entonces la función `day-from-datetime` devuelve `3`.

6.7.12.10 duration-add

Devuelve la duración resultante de sumar dos duraciones.



Lenguajes

Disponible para Java, C#, C++ y el motor de ejecución integrado.

© 2016-2022 Altova GmbH

Parámetros

Nombre	Tipo	Descripción
duration1	`xs:duration`	Obtiene el primer valor de la entrada de tipo `xs:duration`.
duration2	`xs:duration`	Obtiene el segundo valor de la entrada de tipo `xs:duration`.

Ejemplo

Si la primera duración es **`P0Y0M3DT03H0M`** (3 días y 3 horas) y la segunda **`P0Y0M3DT01H0M`** (3 días y 1 hora), entonces la función devuelve **`P6DT4H`** (6 días y 4 horas).

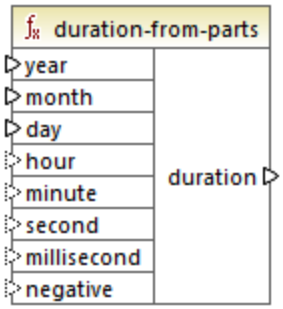
6.7.12.11 duration-from-parts

El resultado es una duración calculada de tipo `xs:duration` al combinar estos argumentos: year, month, day, hour, minute, second, millisecond, negative.

Un ejemplo del parámetro `duration` es **`P1Y2M3DT04H05M59S`**, donde:

- "P" designa el periodo y es obligatorio;
- El resto de caracteres representa, en este orden: 1 año, 2 meses, 3 días, T (indica el tiempo), 04 horas, 05 minutos, 59 segundos.

Si aparece el caracter menos (-) antes de "P" la duración se entiende como negativa, por ejemplo: **`-P1D`**.



Lenguajes

Disponible para Java, C#, C++ y el motor de ejecución integrado.

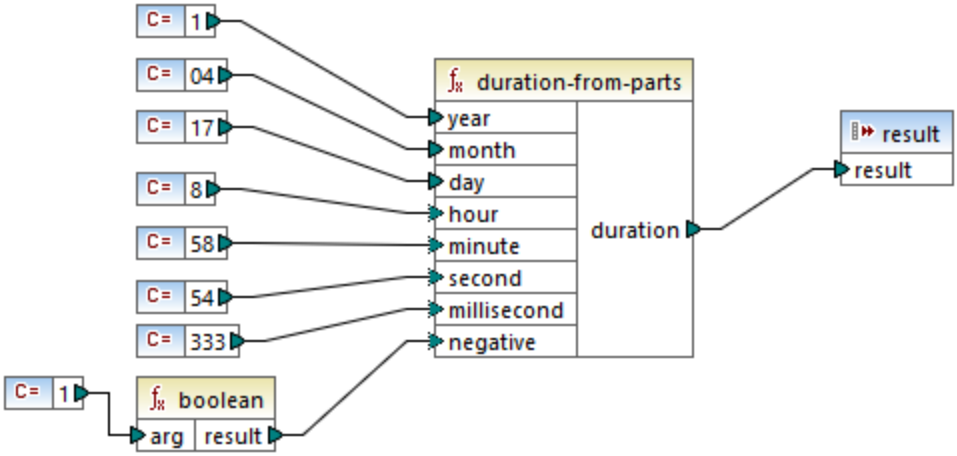
Parámetros

Nombre	Tipo	Descripción
year	`xs:int`	El año.
month	`xs:int`	El mes.

Nombre	Tipo	Descripción
day	`xs:int`	El día del mes.
hour	`xs:int`	Opcional. La hora.
minute	`xs:int`	Opcional. El minuto.
second	`xs:int`	Opcional. El segundo.
millisecond	`xs:decimal`	Opcional. El milisegundo.
negative	`xs:boolean`	Opcional. Debe ser **true** para que resulte en una duración negativa. De lo contrario debe ser **false**.

Ejemplo

Esta asignación genera una duración negativa de 1 año, 4 meses, 17 días, 8 horas, 58 minutos y 54,333 segundos.



El resultado de la asignación es `-P1Y4M17DT8H58M54.333S`.

6.7.12.12 duration-subtract

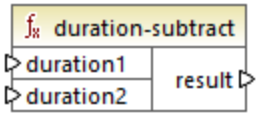
El resultado es la duración que se obtiene al sustraer **duration2** a **duration1**.

Un ejemplo del parámetro `duration` es `P1Y2M3DT04H05M59S`, donde:

- "P" designa el periodo y es obligatorio;
- El resto de caracteres representa, en este orden: 1 año, 2 meses, 3 días, T (indica el tiempo), 04 horas, 05 minutos, 59 segundos.

Si aparece el caracter menos (-) antes de "P" la duración se entiende como negativa, por ejemplo: `-P1D`.

 © 2016-2022 Altova GmbH



Lenguajes

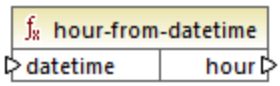
Disponible para Java, C#, C++ y el motor de ejecución integrado.

Ejemplo

Si **duration1** es `P0Y0M0DT05H07M` (5 horas y 7 minutos) y **duration2** es `PT1H` (1 hora), la función devuelve `PT4H7M` (4 horas y 7 minutos).

6.7.12.13 hour-from-datetime

El resultado es la parte date del argumento de entrada datetime.



Lenguajes

Disponible para Java, C#, C++ y el motor de ejecución integrado.

Parámetros

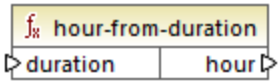
Nombre	Tipo	Descripción
datetime	`xs:dateTime`	Obtiene el valor de la entrada de tipo `xs: dateTime`.

Ejemplo

Si **datetime** es `2001-12-17T09:30:02+05:00`, entonces la función devuelve `9`.

6.7.12.14 hour-from-duration

El resultado es el componente hour del argumento duration.



Lenguajes

Disponible para Java, C#, C++ y el motor de ejecución integrado.

Parámetros

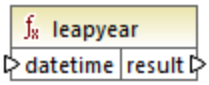
Nombre	Tipo	Descripción
duration	`xs:duration`	Obtiene el valor de la entrada de tipo `xs:duration`.

Ejemplo

Si **duration** es `P0Y0M0DT05H07M`, entonces la función devuelve `5`.

6.7.12.15 leapyear

El resultado es **true** o **false**, dependiendo de si el año del valor `xs:dateTime` es bisiesto.



Lenguajes

Disponible para Java, C#, C++ y el motor de ejecución integrado.

Parámetros

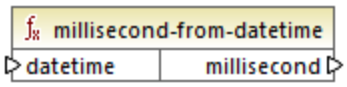
Nombre	Tipo	Descripción
datetime	`xs:dateTime`	Obtiene el valor de la entrada de tipo `xs: dateTime`.

Ejemplo

Si **datetime** es `2020-04-17T09:30:02+02:00`, la función devuelve **true** porque 2020 es bisiesto.

6.7.12.16 millisecond-from-datetime

El resultado es el componente millisecond del argumento datetime.



© 2016-2022 Altova GmbH

Lenguajes

Disponible para Java, C#, C++ y el motor de ejecución integrado.

Parámetros

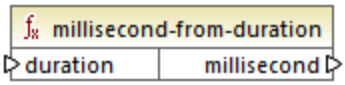
Nombre	Tipo	Descripción
datetime	`xs:dateTime`	Obtiene el valor de la entrada de tipo `xs: dateTime`.

Ejemplo

Si **datetime** es `2001-12-17T09:30:02.544+05:00`, entonces la función devuelve `544`.

6.7.12.17 millisecond-from-duration

El resultado es el componente millisecond del argumento duration.



Lenguajes

Disponible para Java, C#, C++ y el motor de ejecución integrado.

Parámetros

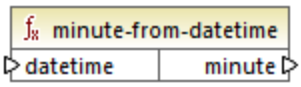
Nombre	Tipo	Descripción
duration	`xs:duration`	Obtiene el valor de la entrada de tipo `xs:duration`.

Ejemplo

Si **duration** es `P0Y0M0DT05H07M02.227S`, entonces la función devuelve `227`.

6.7.12.18 minute-from-datetime

El resultado es el componente minute del argumento dateTime.



Lenguajes

Disponible para Java, C#, C++ y el motor de ejecución integrado.

Parámetros

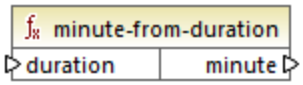
Nombre	Tipo	Descripción
datetime	`xs:dateTime`	Obtiene el valor de la entrada de tipo `xs: dateTime`.

Ejemplo

Si **datetime** es `2001-12-17T09:30:02.544+05:00`, entonces la función devuelve `30`.

6.7.12.19 minute-from-duration

El resultado es el componente minute del argumento duration.



Lenguajes

Disponible para Java, C#, C++ y el motor de ejecución integrado.

Parámetros

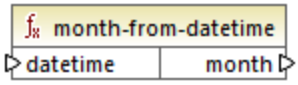
Nombre	Tipo	Descripción
duration	`xs:duration`	Obtiene el valor de la entrada de tipo `xs:duration`.

Ejemplo

Si **duration** es `P0Y0M0DT05H07M02.227S`, entonces la función devuelve `7`.

6.7.12.20 month-from-datetime

El resultado es el componente month del argumento dateTime.



© 2016-2022 Altova GmbH

Lenguajes

Disponible para Java, C#, C++ y el motor de ejecución integrado.

Parámetros

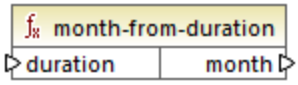
Nombre	Tipo	Descripción
datetime	`xs:dateTime`	Obtiene el valor de la entrada de tipo `xs: dateTime`.

Ejemplo

Si **datetime** es **`2001-12-17T09:30:02.544+05:00`**, entonces la función devuelve **`12`**.

6.7.12.21 month-from-duration

El resultado es el componente month del argumento duration.



Lenguajes

Disponible para Java, C#, C++ y el motor de ejecución integrado.

Parámetros

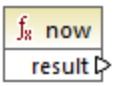
Nombre	Tipo	Descripción
duration	`xs:duration`	Obtiene el valor de la entrada de tipo `xs:duration`.

Ejemplo

Si **duration** es **`P0Y04M0DT05H07M02.227S`**, entonces la función devuelve **`4`**.

6.7.12.22 now

El resultado es la fechaHora actual (incluida la zona horaria) como valor `xs:dateTime`.

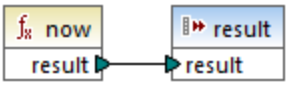


Lenguajes

Disponible para Java, C#, C++ y el motor de ejecución integrado.

Ejemplo

La asignación siguiente devuelve la fecha y hora actuales. El resultado cambia cada vez que se ejecuta la asignación.

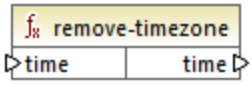


Un ejemplo del resultado sería **`2020-04-17T11:42:34.684+02:00`**.

Para ver un ejemplo de cómo extraer la fecha de ayer consulte la función core | lang | datetime-add 727 .

6.7.12.23 remove-timezone

Elimina el componente de uso horario del parámetro de entrada (de tipo `xs:dateTime`) **time**.



Lenguajes

Disponible para Java, C#, C++ y el motor de ejecución integrado.

Parámetros

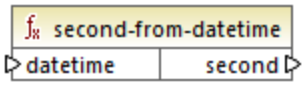
Nombre	Tipo	Descripción
time	**`xs:dateTime`**	Obtiene el valor de la entrada de tipo `xs: dateTime`.

Ejemplo

Si **time** es **`2001-12-17T09:30:02+05:00`**, entonces la función devuelve **`2001-12-17T09:30:02`**.

6.7.12.24 second-from-datetime

El resultado es el componente seconds del argumento dateTime.



 © 2016-2022 Altova GmbH

Lenguajes

Disponible para Java, C#, C++ y el motor de ejecución integrado.

Parámetros

Nombre	Tipo	Descripción
datetime	`xs:dateTime`	Obtiene el valor de la entrada de tipo `xs: dateTime`.

Ejemplo

Si **datetime** es `2001-12-17T09:30:02.544+05:00`, entonces la función devuelve **2**.

6.7.12.25 second-from-duration

El resultado es el componente seconds del argumento duration.

Lenguajes

Disponible para Java, C#, C++ y el motor de ejecución integrado.

Parámetros

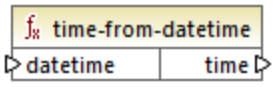
Nombre	Tipo	Descripción
duration	`xs:duration`	Obtiene el valor de la entrada de tipo `xs:duration`.

Ejemplo

Si **duration** es `P0Y04M0DT05H07M02.227S`, entonces la función devuelve **2**.

6.7.12.26 time-from-datetime

Devuelve el componente time del argumento dateTime.



Lenguajes

Disponible para Java, C#, C++ y el motor de ejecución integrado.

Parámetros

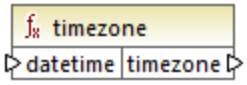
Nombre	Tipo	Descripción
datetime	`xs:dateTime`	Obtiene el valor de la entrada de tipo `xs: dateTime`.

Ejemplo

Si **datetime** es `2001-12-17T09:30:02.544+05:00`, entonces la función devuelve `09:31:02+05:00`.

6.7.12.27 timezone

Devuelve el uso horario en minutos del valor dateTime dado como argumento. El resultado para UTC es 0.



Lenguajes

Disponible para Java, C#, C++ y el motor de ejecución integrado.

Parámetros

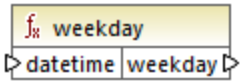
Nombre	Tipo	Descripción
datetime	`xs:dateTime`	Obtiene el valor de la entrada de tipo `xs: dateTime`.

Ejemplo

Si **datetime** es `2001-12-17T09:30:02.544+05:00`, entonces la función devuelve `300`.

6.7.12.28 weekday

Devuelve el día de la semana indicado por el valor dateTime, siendo **Lunes=1** y así sucesivamente hasta **Domingo=7**.



© 2016-2022 Altova GmbH

Lenguajes

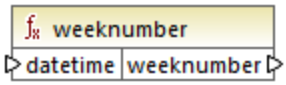
Disponible para Java, C#, C++ y el motor de ejecución integrado.

Parámetros

Nombre	Tipo	Descripción
datetime	`xs:dateTime`	Obtiene el valor de la entrada de tipo `xs: dateTime`.

6.7.12.29 weeknumber

Devuelve el número de la semana del año indicado por el valor dateTime. La función devuelve el valor **1** para la primera semana del año, el valor **2** para la segunda semana y así sucesivamente.



Lenguajes

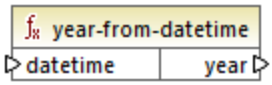
Disponible para Java, C#, C++ y el motor de ejecución integrado.

Parámetros

Nombre	Tipo	Descripción
datetime	`xs:dateTime`	Obtiene el valor de la entrada de tipo `xs: dateTime`.

6.7.12.30 year-from-datetime

El resultado es el componente year del argumento dateTime.



Lenguajes

Disponible para Java, C#, C++ y el motor de ejecución integrado.

Parámetros

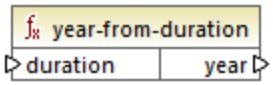
Nombre	Tipo	Descripción
datetime	`xs:dateTime`	Obtiene el valor de la entrada de tipo `xs: dateTime`.

Ejemplo

Si **datetime** es `2001-12-17T09:30:02.544+05:00`, entonces la función devuelve `2001`.

6.7.12.31 year-from-duration

El resultado es el componente year del argumento duration.



Lenguajes

Disponible para Java, C#, C++ y el motor de ejecución integrado.

Parámetros

Nombre	Tipo	Descripción
duration	`xs:duration`	Obtiene el valor de la entrada de tipo `xs:duration`.

Ejemplo

Si **duration** es `P01Y04M0DT05H07M02.227S`, entonces la función devuelve `1`.

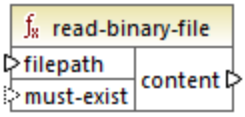
6.7.13 lang | file functions

Las funciones **file** de la biblioteca **lang** sirven para leer contenido binario de archivos o para escribir contenido binario en archivos.

6.7.13.1 read-binary-file

Esta función devuelve el contenido del archivo indicado como BLOB (objeto binario grande por sus siglas en inglés) de tipo `xs:base64Binary`. Tenga en cuenta que aunque el tipo de datos se llama "base64Binary", la representación interna es únicamente un BLOB. La codificación en base 64 no tiene lugar hasta que no asigna el resultado de la función a un nodo XML de tipo `xs:base64Binary`. También puede asignar el resultado de la función a `xs:hexBinary`, a un blob de BD o a un campo binario de una estructura de Protocol Buffers.

 © 2016-2022 Altova GmbH



- Para leer un archivo binario en una asignación debe dar su ruta como entrada del argumento **filepath**. Si **filepath** es relativo, entonces MapForce buscará el archivo en el mismo directorio que la asignación. El argumento **must-exist** es opcional; si no se puede abrir el archivo y su parámetro es **true**, la asignación emite un error. Si no se puede abrir el archivo y su parámetro es **false**, la asignación devuelve un binario vacío.

Lenguajes

Built-in.

Parámetros

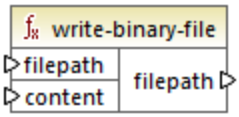
Nombre	Tipo	Descripción
filepath	`xs:string`	La ruta del archivo.
must-exist	`xs:boolean`	Parámetro opcional. Si el archivo no se puede abrir y este parámetro es **true**, la asignación genera un error. Si el archivo no se puede abrir y este parámetro es **false**, la asignación devuelve un binario vacío. El valor predeterminado es **true**.

Ejemplo

Consulte Ejemplo: leer archivos binarios 473.

6.7.13.2 write-binary-file

Esta función escribe contenido binario en la ruta de archivo indicada y devuelve la ruta del archivo escrito. Si quiere que el resultado sera únicamente un archivo binario, conecte el resultado de la función a un componente de salida simple 489. Esta función escribe un archivo cada vez que se usa su resultado en la asignación, por lo que recomendamos que conecte el resultado de la función directamente a un componente de destino, sin usar otros procesamientos intermedios.



Para escribir archivos binarios debe dar su ruta como entrada del argumento **filepath**. Si **filepath** es relativo,

entonces MapForce genera el archivo en el mismo directorio que la asignación. El argumento **content** debe estar conectado al contenido binario (por ejemplo, un campo BLOB de una BD).

Si accede a la vista previa de la asignación en MapForce, la función genera archivos temporales por defecto, en lugar de escribir archivos directamente en disco. Para guardar estos archivos temporales en disco haga clic en la pestaña *Resultados* y después haga clic en los botones de la barra de herramientas **Guardar resultado generado** o **Guardar todos los resultados generados**.

Para configurar MapForce para que escriba el resultado directamente en archivos finales en vez de temporales seleccione el comando de menú **Herramientas | Opciones**, haga clic en **Generales** y después marque la opción **Escribir directamente en archivos de salida finales**. Tenga en cuenta que esta opción sobrescribe cualquier otro archivo que tenga el mismo nombre.

La función siempre devuelve el nombre de archivo final (no el temporal), incluso aunque no haya guardado aún el archivo final en disco (que es el caso cuando accede a la vista previa de la asignación y se deshabilita la opción **Escribir directamente en archivos de salida finales**).

> Tenga en cuenta que una asignación no puede volver a leer su propio archivo de salida.

Lenguajes

Built-in.

Parámetros

Nombre	Tipo	Descripción
filepath	`xs:string`	La ruta del archivo de entrada.
content	`xs:base64Binary`	El contenido binario de tipo `xs:base64Binary`.

Ejemplo

Consulte Ejemplo: escribir archivos binarios 474.

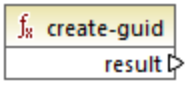
6.7.14 lang | generator functions (generador)

Las funciones generadoras de la biblioteca **lang** generan valores (actualmente **`create-guid`** es la única función de este tipo).

6.7.14.1 create-guid

Crea un identificador global único (GUID) como cadena cifrada en hexadecimal. Esta función se puede usar para generar valores únicos, directamente a partir de la asignación, para campos de BD u otro tipo de complementos. Consulte también la función format-guid-string 765.

 © 2016-2022 Altova GmbH



Lenguajes

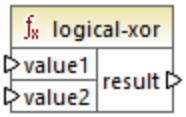
Disponible para Java, C#, C++ y el motor de ejecución integrado.

6.7.15 lang | logical functions (lógica)

Las funciones lógicas de la biblioteca **lang** incluyen funciones que evalúan distintos tipos de valores con lógica booleana.

6.7.15.1 logical-xor

El resultado es **true** si **value1** es distinto de **value2**. De lo contrario, el resultado es **false**.



Lenguajes

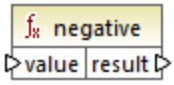
Disponible para Java, C#, C++ y el motor de ejecución integrado.

Parámetros

Nombre	Tipo	Descripción
value1	`xs:boolean`	El primer valor de la entrada.
value2	`xs:boolean`	El segundo valor de la entrada.

6.7.15.2 negative

El resultado es **true** si value es negativo (es decir, inferior a cero). De lo contrario, el resultado es **false**.



Lenguajes

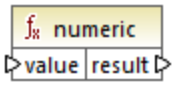
Disponible para Java, C#, C++ y el motor de ejecución integrado.

Parámetros

Nombre	Tipo	Descripción
value	`xs:decimal`	El valor de entrada.

6.7.15.3 numeric

El resultado es **true** si value es un número. De lo contrario, el resultado es **false**. La entrada suele ser una cadena.



Lenguajes

Disponible para Java, C#, C++ y el motor de ejecución integrado.

Parámetros

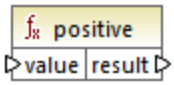
Nombre	Tipo	Descripción
value	`xs:decimal`	El valor de entrada.

Ejemplo

Si el valor de entrada es la cadena `4,33`, la función devuelve **true** . Si el valor de entrada es la cadena `4,33 USD`, la función devuelve **false** .

6.7.15.4 positive

El resultado es **true** si value es positivo De lo contrario, el resultado es **false**.



Lenguajes

Disponible para Java, C#, C++ y el motor de ejecución integrado.

© 2016-2022 Altova GmbH

Parámetros

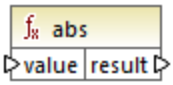
Nombre	Tipo	Descripción
value	`xs:decimal`	El valor de entrada.

6.7.16 lang | math functions (matemáticas)

Las funciones matemáticas de la biblioteca **lang** se pueden usar para ejecutar distintas operaciones matemáticas en la asignación.

6.7.16.1 abs

El resultado es el valor absoluto de la entrada value. Si el argumento no es negativo, devuelve ese argumento. Si el argumento es negativo, devuelve la negación de ese argumento.



Lenguajes

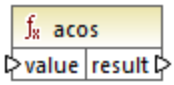
Disponible para Java, C#, C++ y el motor de ejecución integrado.

Parámetros

Nombre	Tipo	Descripción
value	`xs:decimal`	El valor de entrada.

6.7.16.2 acos

El resultado es el arco coseno del parámetro de entrada **value** en el rango de `-pi/2` a `pi/2`.



Lenguajes

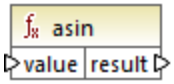
Disponible para Java, C#, C++ y el motor de ejecución integrado.

Parámetros

Nombre	Tipo	Descripción
value	`xs:double`	El valor de entrada.

6.7.16.3 asin

El resultado es el arco seno del parámetro de entrada **value** en el rango de `-pi/2` a `pi/2`.



Lenguajes

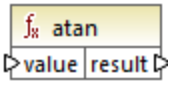
Disponible para Java, C#, C++ y el motor de ejecución integrado.

Parámetros

Nombre	Tipo	Descripción
value	`xs:double`	El valor de entrada.

6.7.16.4 atan

El resultado es el arco tangente del parámetro de entrada **value** en el rango de `-pi/2` a `pi/2`.



Lenguajes

Disponible para Java, C#, C++ y el motor de ejecución integrado.

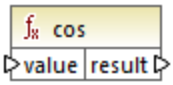
Parámetros

Nombre	Tipo	Descripción
value	`xs:double`	El valor de entrada.

© 2016-2022 Altova GmbH

6.7.16.5 cos

El resultado es el coseno del parámetro de entrada **value**. La unidad del valor es el radián.



Lenguajes

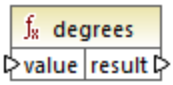
Disponible para Java, C#, C++ y el motor de ejecución integrado.

Parámetros

Nombre	Tipo	Descripción
value	`xs:double`	El valor de entrada.

6.7.16.6 degrees

El resultado de convertir en grados el valor de value dado en radianes.



Lenguajes

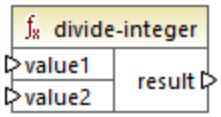
Disponible para Java, C#, C++ y el motor de ejecución integrado.

Parámetros

Nombre	Tipo	Descripción
value	`xs:double`	El valor de entrada.

6.7.16.7 divide-integer

El resultado es el entero resultante de dividir **value1** por **value2**.



Lenguajes

Disponible para Java, C#, C++ y el motor de ejecución integrado.

Parámetros

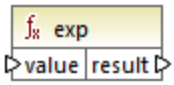
Nombre	Tipo	Descripción
value1	`xs:decimal`	El primer valor de la entrada.
value2	`xs:decimal`	El segundo valor de la entrada.

Ejemplo

Por ejemplo, **15** divide-integer **2**, el entero resultante es **7**.

6.7.16.8 exp

El resultado es **e** (el logaritmo natural base) elevado a la potencia **value**.



Lenguajes

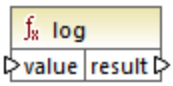
Disponible para Java, C#, C++ y el motor de ejecución integrado.

Parámetros

Nombre	Tipo	Descripción
value	`xs:double`	El valor de entrada.

6.7.16.9 log

El resultado es el logaritmo natural (en base e) de **value**.



Lenguajes

Disponible para Java, C#, C++ y el motor de ejecución integrado.

© 2016-2022 Altova GmbH

Parámetros

Nombre	Tipo	Descripción
value	`xs:double`	El valor de entrada.

6.7.16.10 log10

El resultado es el logaritmo decimal (en base 10) de **value**.

Lenguajes

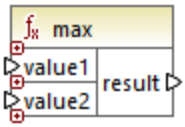
Disponible para Java, C#, C++ y el motor de ejecución integrado.

Parámetros

Nombre	Tipo	Descripción
value	`xs:double`	El valor de entrada.

6.7.16.11 max

El resultado es el valor numéricamente mayor de value1 comparado con value2. Por defecto, esta función tiene solo dos parámetros, pero puede añadir más. Haga clic en el icono **Agregar parámetro** (⊕) o **Eliminar parámetro** (⊗ para añadir o borrar parámetros. Consulte también el apartado Agregar o eliminar argumentos en una función 550.



Lenguajes

Disponible para Java, C#, C++ y el motor de ejecución integrado.

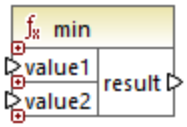
Parámetros

Nombre	Tipo	Descripción
value1	`xs:decimal`	El primer valor de la entrada.

Nombre	Tipo	Descripción
value2	`xs:decimal`	El segundo valor de la entrada.
valueN	`xs:decimal`	El valor de *n* entrada.

6.7.16.12 min

El resultado es el valor numéricamente menor de value1 comparado con value2. Por defecto, esta función tiene solo dos parámetros, pero puede añadir más. Haga clic en el icono **Agregar parámetro** ( ) o **Eliminar parámetro** (para añadir o borrar parámetros. Consulte también el apartado Agregar o eliminar argumentos en una función 550.



Lenguajes

Disponible para Java, C#, C++ y el motor de ejecución integrado.

Parámetros

Nombre	Tipo	Descripción
value1	`xs:decimal`	El primer valor de la entrada.
value2	`xs:decimal`	El segundo valor de la entrada.
valueN	`xs:decimal`	El valor de *n* entrada.

6.7.16.13 pi

El resultado es el valor de π.

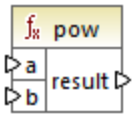
Lenguajes

Disponible para Java, C#, C++ y el motor de ejecución integrado.

© 2016-2022 Altova GmbH

6.7.16.14 pow

El resultado es el valor de **a** elevado a la potencia **b**.



Lenguajes

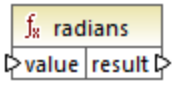
Disponible para Java, C#, C++ y el motor de ejecución integrado.

Parámetros

Nombre	Tipo	Descripción
a	`xs:double`	Da el valor **a** (la base).
b	`xs:double`	Da el valor **b** (la potencia).

6.7.16.15 radians

El resultado de convertir en radianes el valor de value dado en grados.



Lenguajes

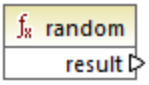
Disponible para Java, C#, C++ y el motor de ejecución integrado.

Parámetros

Nombre	Tipo	Descripción
value	`xs:double`	El valor de entrada.

6.7.16.16 random

El resultado es un valor pseudoaleatorio comprendido entre 0.0 y 1.0. Los valores devueltos se eligen de forma pseudoaleatoria con una distribución (aproximadamente) uniforme de ese rango.

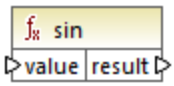


Lenguajes

Disponible para Java, C#, C++ y el motor de ejecución integrado.

6.7.16.17 sin

El resultado es el seno de **value**.



Lenguajes

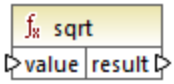
Disponible para Java, C#, C++ y el motor de ejecución integrado.

Parámetros

Nombre	Tipo	Descripción
value	`xs:double`	El valor de entrada.

6.7.16.18 sqrt

El resultado es la raíz cuadrada de **value**.



Lenguajes

Disponible para Java, C#, C++ y el motor de ejecución integrado.

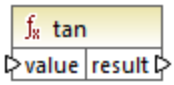
Parámetros

Nombre	Tipo	Descripción
value	`xs:double`	El valor de entrada.

© 2016-2022 Altova GmbH

6.7.16.19 tan

El resultado es la tangente de **value**.



Lenguajes

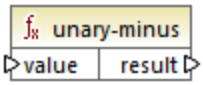
Disponible para Java, C#, C++ y el motor de ejecución integrado.

Parámetros

Nombre	Tipo	Descripción
value	`xs:double`	El valor de entrada.

6.7.16.20 unary-minus

El resultado es la negación del valor con signo de value.



Lenguajes

Disponible para Java, C#, C++ y el motor de ejecución integrado.

Parámetros

Nombre	Tipo	Descripción
value	`xs:decimal`	El valor de entrada.

Ejemplo

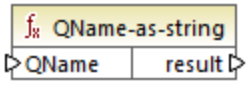
Si el valor de entrada es **3**, la función devuelve **-3**. Si el valor de entrada es **-3**, la función devuelve **3**.

6.7.17 lang | QName functions (QName)

Las funciones de la biblioteca **lang** convierten los valores de nombre cualificado (Qname) en cadenas y viceversa. Al diferencia de las funciones de la biblioteca **core**, estas funciones solo están disponibles para Java, C#, C++ o el motor de ejecución integrado.

6.7.17.1 QName-as-string

El resultado es la representación de cadena única del QName.



Lenguajes

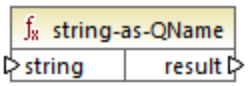
Disponible para Java, C#, C++ y el motor de ejecución integrado.

Parámetros

Nombre	Descripción
QName	El valor `xs:QName` de entrada.

6.7.17.2 string-as-QName

Convierte la representación de cadena de un QName en un QName.



Lenguajes

Disponible para Java, C#, C++ y el motor de ejecución integrado.

Nombre	Descripción
string	El valor `string` de entrada.

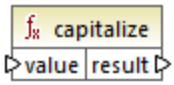
6.7.18 lang | string functions (cadena)

La biblioteca **lang** incluye funciones para trabajar con los lenguajes Java, C# y C++.

© 2016-2022 Altova GmbH

6.7.18.1 capitalize

El resultado es la cadena de entrada **value,** con la primera letra de cada palabra en mayúsculas.



Lenguajes

Disponible para Java, C#, C++ y el motor de ejecución integrado.

Parámetros

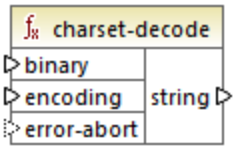
Nombre	Tipo	Descripción
value	`xs:string`	El valor de entrada.

Ejemplo

Si el valor de entrada es `el veloz zorro marrón`, la función devuelve `El Veloz Zorro Marrón`.

6.7.18.2 charset-decode

La función `charset-decode` toma como entrada datos binarios cifrados como texto en Base64. Descifra datos conforme al conjunto de caracteres indicado (por ejemplo, "utf-8") y devuelve el valor de cadena resultante. Si necesita cifrar datos binarios como texto en Base64, use la función charset-encode 761 .



Lenguajes

Disponible en el motor de ejecución integrado.

Parámetros

Nombre	Tipo	Descripción
binary-data	`xs:base64Binary`	Los datos binarios como texto en Base 64.

Nombre	Tipo	Descripción
encoding	`xs:string`	El conjunto de caracteres usado para codificar (por ejemplo, "utf-8").
error-abort	`xs:boolean`	Argumento opcional que indica cómo debe continuar el procesamiento si se encuentran errores. Son valores válidos: • **true**: terminar el procesamiento con una excepción donde está el carácter no válido. • **false**: terminar el procesamiento y reemplazar los caracteres no válidos con el carácter de sustitución . El valor predeterminado es true.

Ejemplo

Imagine que quiere descodificar datos binarios con origen en el siguiente archivo XML. Observe que el elemento **message** contiene datos binarios codificados como texto en Base 64.

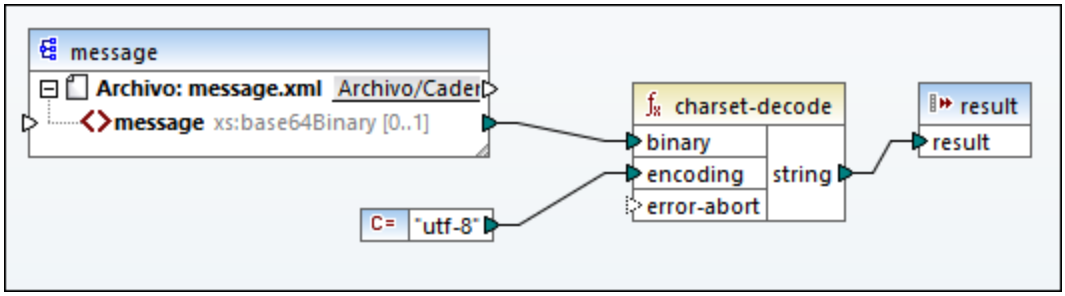
```
<?xml version="1.0" encoding="UTF-8"?>
<message xsi:noNamespaceSchemaLocation="message.xsd"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">TG9yZW0gaXBzdW0=</message>
```

El tipo de datos del elemento **message** es `xs:base64Binary`, como se ve en el esquema:

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
 <xs:element name="message" type="xs:base64Binary"/>
</xs:schema>
```

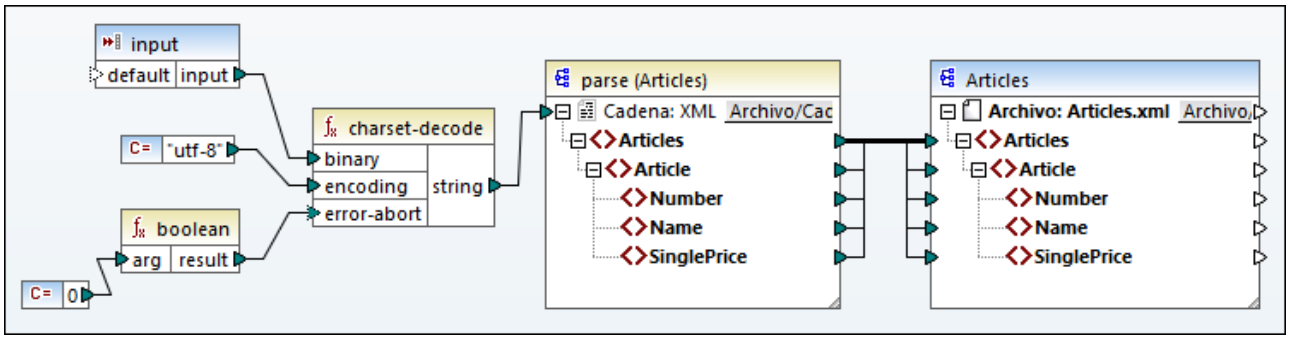
Este es el aspecto de una asignación que descodifica el mensaje anterior:

© 2016-2022 Altova GmbH



La asignación de este mensaje tiene como salida el texto "Lorem ipsum".

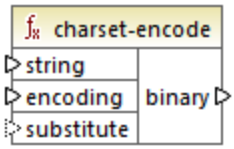
Una asignación también puede procesar texto o archivos XML codificados como datos en Base64 con ayuda de un componente de serialización de MapForce. Por ejemplo, la asignación siguiente tiene un parámetro de entrada que espera datos de texto en Base64. Si asumimos que los datos en Base64 se crearon a partir de un archivo XML, como se muestra en el ejemplo charset-encode 761 , se puede recrear el archivo XML original como se aprecia en esta asignación:



En esta asignación el argumento **error-abort** toma el valor **false**, que se produce con ayuda de la función integrada **boolean**. Esto garantiza que el procesamiento siga aunque se encuentren caracteres no válidos. La cadena de texto que resulta de la función se pasa a un componente de análisis XML que la convierte en un archivo XML. Observe que para que sea posible el análisis XML debe tener el archivo de esquema XSD. Para más información consulte Redistribuir y serializar cadenas de texto 903.

6.7.18.3 charset-encode

La función **charset- encode** toma como cadena de entrada datos que cifra como texto en Base64. Los datos se cifran en el conjunto de caracteres especificado (por ejemplo, "utf-8") y se devuelven como un tipo `xs:base64Binary`. Si necesita descifrar datos binarios previamente cifrados como texto en Base64, use la función charset-decode. charset-decode 759 .



Lenguajes

Disponible en el motor de ejecución integrado.

Parámetros

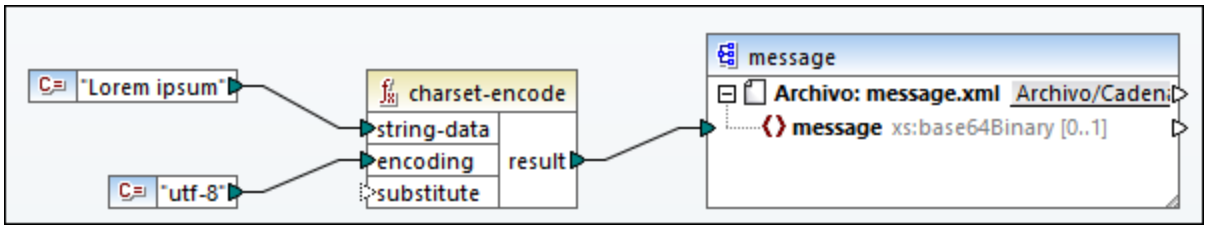
Nombre	Tipo	Descripción
string-data	`xs:string`	El datos de cadena que se debe cifrar.
encoding	`xs:string`	El conjunto de caracteres usado para codificar (por ejemplo, "utf-8").
substitute	`xs:string`	Argumento opcional que indica qué carácter de reemplazo usar si se encuentran caracteres no válidos. Este argumento se aplica si está usando un cifrado que no sea Unicode. Para cifrados Unicode, el carácter de reemplazo es ⁇.

Ejemplo

Imagine que quiere cifrar el texto "Lorem ipsum" como datos en Base64 usando el conjunto de caracteres UTF-8 y escribir esos datos en un archivo XML de destino. El archivo XML de destino tiene un elemento **message** de tipo `xs:base64Binary`, como se ve en el esquema:

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
 <xs:element name="message" type="xs:base64Binary"/>
</xs:schema>
```

Este es el aspecto de una asignación que cifra en Base64:

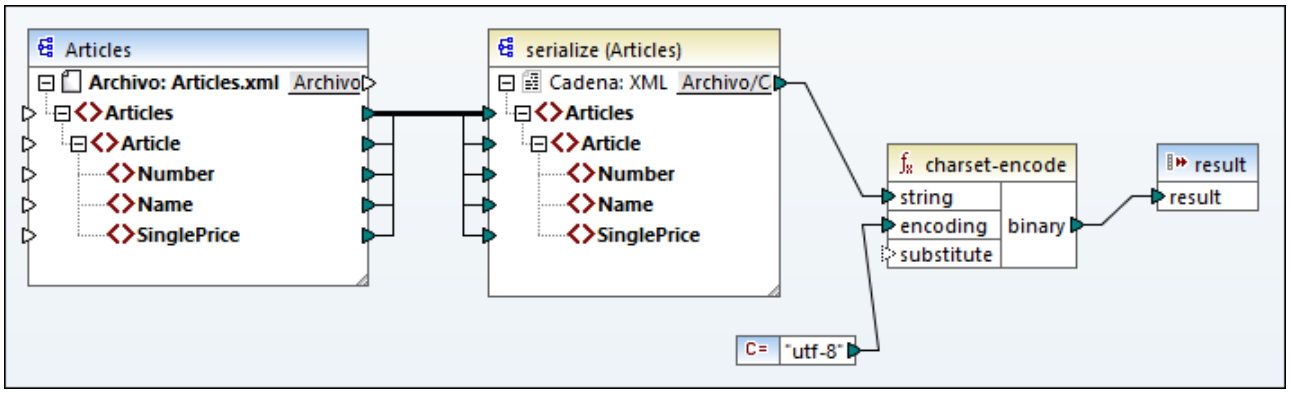


Esta asignación produce resultados XML como los del extracto de código siguiente (se han omitido las referencias al esquema y la declaración XML):

```
<message>TG9yZW0gaXBzdW0=</message>
```

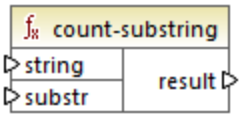

© 2016-2022 Altova GmbH

También puede cifrar texto o archivos XML como Base64 con ayuda de un componente de serialización de MapForce. Por ejemplo, la asignación de la imagen siguiente serializa un XML de origen en una cadena de texto. La cadena resultante se pasa como argumento a la función **charset-encode**. último, el resultado de la función se devuelve como resultado de la asignación, con ayuda de un componente simple de salida (véase el apartado Obtener valores de cadena de una asignación) 488. Para más información consulte el apartado Redistribuir y serializar cadenas de texto 903.



6.7.18.4 count-substring

El resultado es el número de veces que la cadena **substr** aparece en el parámetro **string**.



Lenguajes

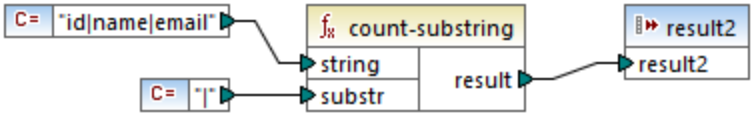
Disponible para Java, C#, C++ y el motor de ejecución integrado.

Parámetros

Nombre	Tipo	Descripción
string	`xs:string`	La cadena de entrada.
substr	`xs:string`	La subcadena que se debe comprobar.

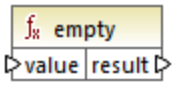
Ejemplo

Esta asignación devuelve **2**. El número de veces que el separador aparece dentro de la cadena de entrada `id|name|email`.



6.7.18.5 empty

El resultado es **true** si la cadena de entrada value está vacía. De lo contrario, el resultado es **false**.



Lenguajes

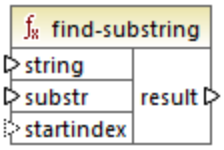
Disponible para Java, C#, C++ y el motor de ejecución integrado.

Parámetros

Nombre	Tipo	Descripción
value	`xs:string`	El valor de entrada.

6.7.18.6 find-substring

Devuelve la posición de la primera aparición de **substr** dentro de **string**, a partir de la posición startindex. Por defecto, la búsqueda empieza por el primer carácter, que tiene la posición 1, pero también puede indicar un índice inicial distinto. Si no se encuentra la cadena **substr**, el resultado es **0**.



Lenguajes

Disponible para Java, C#, C++ y el motor de ejecución integrado.

Parámetros

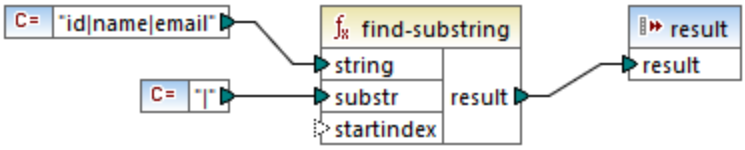
Nombre	Tipo	Descripción
string	`xs:string`	La cadena de entrada.

© 2016-2022 Altova GmbH

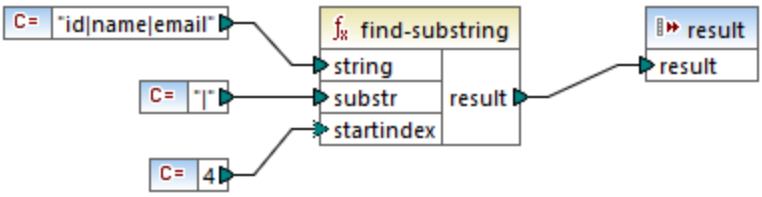
Nombre	Tipo	Descripción
substr	`xs:string`	La subcadena que se debe buscar.
startindex	`xs:int`	Opcional. Indica la posición (índice) inicial de la búsqueda. Si no se indica este parámetro, la búsqueda empieza en la posición 1.

Ejemplo

El resultado de la asignación siguiente es **3**, que es la posición de la primera aparición de la barra vertical en la cadena de entrada `id|name|email`.

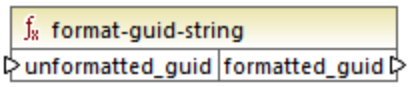


Si indica que el índice inicial es **4**, entonces la función empieza a buscar por el cuarto carácter. Por lo tanto, el resultado de la asignación siguiente sería **8**, que es la primera aparición de la barra vertical a partir del cuarto carácter.



6.7.18.7 format-guid-string

El resultado es una cadena GUID de formato correcto formatted_guid, se puede usar en campos de BD. Consulte la documentación de la create-guid 746 .



Lenguajes

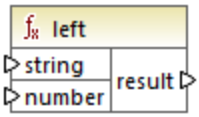
Disponible para Java, C#, C++ y el motor de ejecución integrado.

Parámetros

Nombre	Tipo	Descripción
unformatted_guid	`xs:string`	La cadena de entrada en formato HEX a la que se le quiere dar formato.

6.7.18.8 left

El resultado es una cadena que contiene los primeros caracteres **number** de string.



Lenguajes

Disponible para Java, C#, C++ y el motor de ejecución integrado.

Parámetros

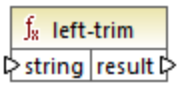
Nombre	Tipo	Descripción
string	`xs:string`	La cadena de entrada.
number	`xs:int`	Indica cuántos caracteres se devuelven empezando por el principio de la cadena.

Ejemplo

Si la cadena de entrada es `Esto es una frase` y el número es **4**, la función devuelve `Esto`.

6.7.18.9 left-trim

El resultado es la cadena de entrada string después de quitarle todos los espacios en blanco iniciales.

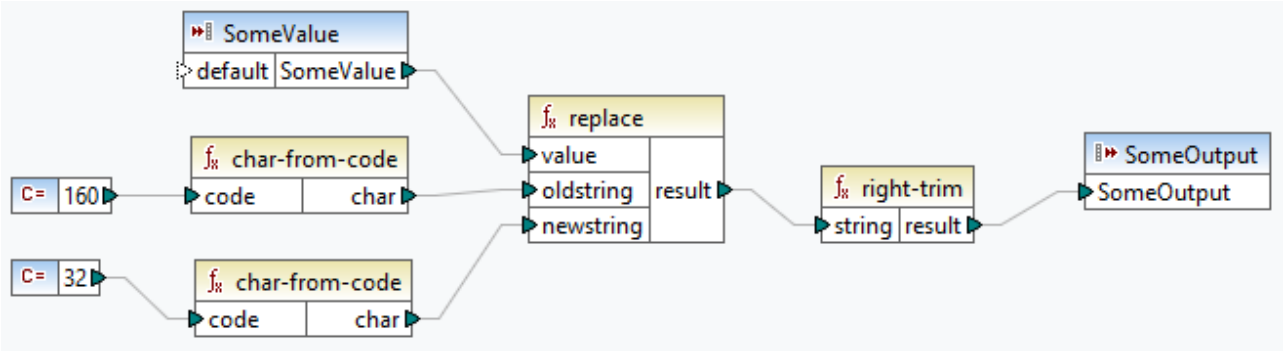


Acerca de los espacios duros

Las funcione `left-trim`, `right-trim` y `normalize-space` no eliminan los espacios duros. Una posible solución es reemplazar esos caracteres, cuya representación decimal es 160, por el carácter espacio, cuya representación decimal es 32. En la asignación siguiente puede ver que, una vez se ha reemplazado el

© 2016-2022 Altova GmbH

espacio duro, el valor `SomeValue` reducido se asigna al destino.



Si su componente de origen es un archivo de Excel, puede eliminar espacios extra en Excel combinando las funciones TRIM, CLEAN y SUBSTITUTE. Para más detalles consulte *Las diez formas principales de limpiar los datos*.

Lenguajes

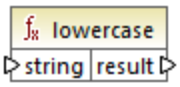
Disponible para Java, C#, C++ y el motor de ejecución integrado.

Parámetros

Nombre	Tipo	Descripción
string	`xs:string`	La cadena de entrada.

6.7.18.10 lowercase

El resultado es la cadena de entrada **string** en minúsculas. Para los caracteres Unicode se usan los caracteres correspondientes en minúsculas.



Lenguajes

Disponible para Java, C#, C++ y el motor de ejecución integrado.

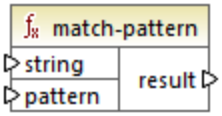
Parámetros

Nombre	Tipo	Descripción
string	`xs:string`	La cadena de entrada.

6.7.18.11 match-pattern

El resultado es **true** si la cadena de entrada string coincide con la expresión regular definida por el parámetro **pattern**. De lo contrario, el resultado es **false**. Para ver un ejemplo consulte el apartado Expresiones regulares 627.

Nota: al generar código C++, C# o Java, las características avanzadas de la sintaxis de la expresión regular pueden variar ligeramente. Consulte la documentación regex de cada lenguaje para más información.



Lenguajes

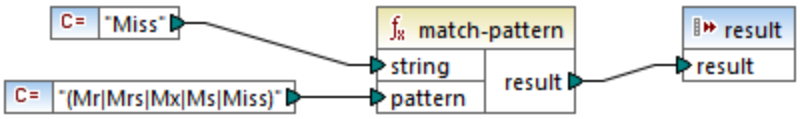
Disponible para Java, C#, C++ y el motor de ejecución integrado.

Parámetros

Nombre	Tipo	Descripción
string	`xs:string`	La cadena de entrada.
pattern	`xs:string`	La expresión regular que debe coincidir.

Ejemplo

Esta asignación valida títulos extranjeros de personas. Concretamente, el resultado de la asignación será **true** para estos títulos: Mr, Mrs, Mx, Ms, Miss.

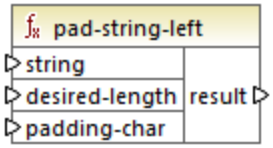


Si la cadena de entrada es distinta a esos títulos, el resultado de la asignación será **false**.

6.7.18.12 pad-string-left

Devuelve una cadena que se rellena a la izquierda con determinado carácter y hasta determinada longitud. La longitud que debe tener la cadena y el carácter de relleno se dan en forma de argumento.

 © 2016-2022 Altova GmbH



Lenguajes

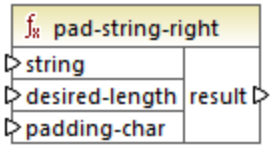
Disponible para Java, C#, C++ y el motor de ejecución integrado.

Parámetros

Nombre	Tipo	Descripción
string	`xs:string`	Especifica la cadena de entrada.
desired-length	`xs:int`	Define la longitud que debe tener la cadena después de aplicar el relleno.
padding-char	`xs:string`	Define el carácter que se debe usar como carácter de relleno.

6.7.18.13 pad-string-right

Devuelve una cadena que se rellena a la derecha con determinado carácter y hasta determinada longitud. La longitud que debe tener la cadena y el carácter de relleno se dan en forma de argumento.



Lenguajes

Disponible para Java, C#, C++ y el motor de ejecución integrado.

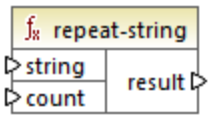
Parámetros

Nombre	Tipo	Descripción
string	`xs:string`	Especifica la cadena de entrada.
desired-length	`xs:int`	Define la longitud que debe tener la cadena después de aplicar el relleno.

Nombre	Tipo	Descripción
padding-char	`xs:string`	Define el carácter que se debe usar como carácter de relleno.

6.7.18.14 repeat-string

Repite la cadena dada tantas veces como indique el argumento count. El argumento **count** determina número de veces que se debe repetir la cadena.



Lenguajes

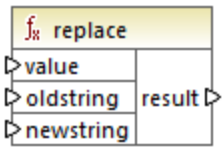
Disponible para Java, C#, C++ y el motor de ejecución integrado.

Parámetros

Nombre	Tipo	Descripción
string	`xs:string`	La cadena de entrada.
count	`xs:int`	El número de veces que se debe repetir la cadena.

6.7.18.15 replace

El resultado es una cadena nueva formada por la cadena de entrada **value**, después de sustituir cada instancia de **oldstring**por **newstring**.



Lenguajes

Disponible para Java, C#, C++ y el motor de ejecución integrado.

© 2016-2022 Altova GmbH

Parámetros

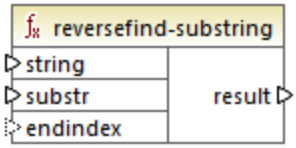
Nombre	Tipo	Descripción
value	`xs:string`	El valor de entrada.
oldstring	`xs:string`	La cadena antigua que se va a reemplazar.
newstring	`xs:string`	La cadena nueva que reemplaza a la antigua.

Ejemplo

Consulte el apartado Reemplazar caracteres especiales 275.

6.7.18.16 reversefind-substring

Devuelve la posición de la primera aparición de **substr** dentro de **string**, a partir de la posición endindex, es decir, de derecha a izquierda. El primer carácter tiene la posición 1. Si no se encuentra la cadena **substr**, el resultado es **0**.



Lenguajes

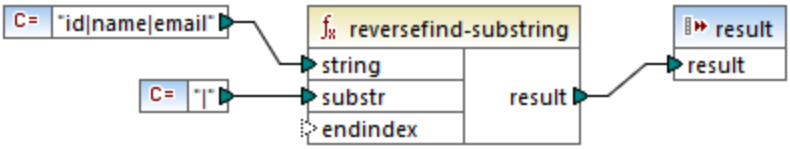
Disponible para Java, C#, C++ y el motor de ejecución integrado.

Parámetros

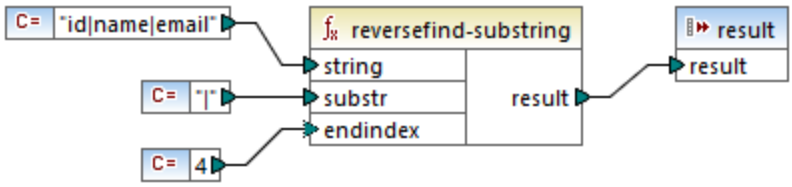
Nombre	Tipo	Descripción
string	`xs:string`	La cadena de entrada.
substr	`xs:string`	La subcadena que se debe buscar.
endindex	`xs:int`	Opcional. Indica la posición final de la búsqueda. Si no se indica este parámetro, la búsqueda finaliza después del último carácter en **string**.

Ejemplo

El resultado de la asignación siguiente es **8**, que es la posición de la última aparición de la barra vertical en la cadena de entrada `id|name|email`.

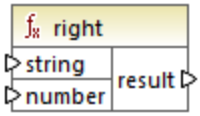


Si indica que el índice final es **4**, entonces la función busca solamente hasta el cuarto carácter. En ese caso, el resultado de la asignación sería **3**.



6.7.18.17 right

El resultado es una cadena que contiene los últimos caracteres **number** de la cadena de entrada.



Lenguajes

Disponible para Java, C#, C++ y el motor de ejecución integrado.

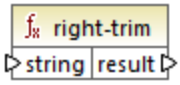
Ejemplo

Si la cadena de entrada es `El veloz zorro marrón` y el número es **3**, la función devuelve `zorro`.

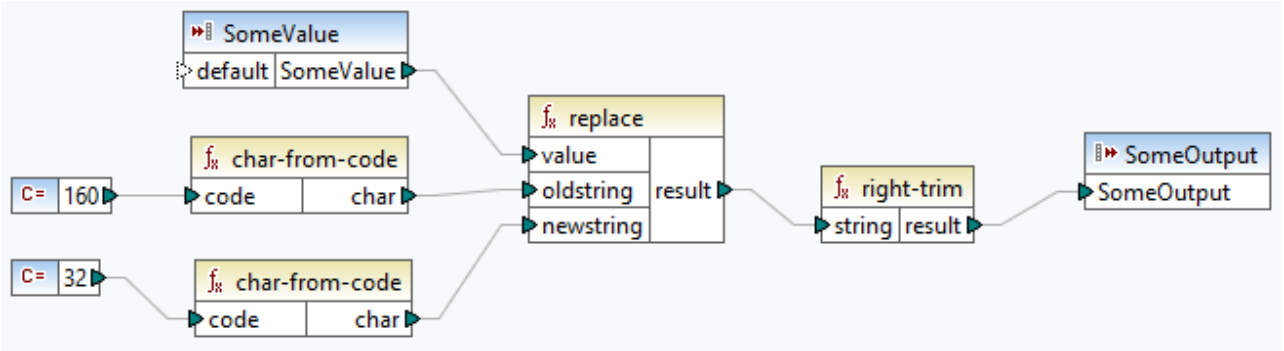
6.7.18.18 right-trim

El resultado es la cadena de entrada string después de eliminar todos los caracteres de espacio en blanco finales.

© 2016-2022 Altova GmbH



Acerca de los espacios duros
Las funcione **left-trim**, **right-trim** y **normalize-space** no eliminan los espacios duros. Una posible solución es reemplazar esos caracteres, cuya representación decimal es 160, por el carácter espacio, cuya representación decimal es 32. En la asignación siguiente puede ver que, una vez se ha reemplazado el espacio duro, el valor `SomeValue` reducido se asigna al destino.



Si su componente de origen es un archivo de Excel, puede eliminar espacios extra en Excel combinando las funciones TRIM, CLEAN y SUBSTITUTE. Para más detalles consulte *Las diez formas principales de limpiar los datos*.

Lenguajes

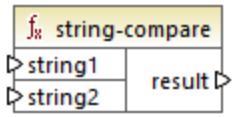
Disponible para Java, C#, C++ y el motor de ejecución integrado.

Parámetros

Nombre	Tipo	Descripción
string	`xs:string`	La cadena de entrada.

6.7.18.19 string-compare

Devuelve el resultado de comparar carácter por carácter dos cadenas de entrada: `string1` y `string2`. La comparación se basa en códigos ASCII. Estas dos cadenas son de tipo `xs:string`. La función distingue entre mayúsculas y minúsculas. Si las cadenas son iguales, entonces el resultado es 0. Si `string1` es menor que `string2`, entonces el resultado es -1. Si `string1` es mayor que `string2`, entonces el resultado es 1.



Ejemplo:

```
string1: hi
string2: Hit
```

Compara cadenas de texto carácter por carácter. La comparación se detiene si detecta que el primer carácter de `string1` y el primer carácter de `string2` son distintos. El resultado se basa en la comparación del primer carácter de cada una de las cadenas. Como `h` está representado por un número código ASCII mayor (104 en el sistema decimal) que `H` (72 en el sistema decimal), `string1` es mayor que `string2` y el resultado de la comparación de cadenas es 1. Si el primer carácter de `string1` y el de `string2` fueran iguales, la función analizaría el segundo carácter y así sucesivamente.

Para ver una comparación simple de cadenas con resultado binario consulte `core | logical functions | equal` 663.

Lenguajes

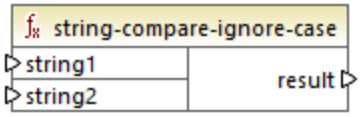
Disponible para Java, C#, C++ y el motor de ejecución integrado.

Parámetros

Nombre	Tipo	Descripción
`string1`	`xs:string`	La primera cadena de entrada.
`string2`	`xs:string`	La segunda cadena de entrada.

6.7.18.20 string-compare-ignore-case

Devuelve el resultado de comparar carácter por carácter dos cadenas de entrada: `string1` y `string2`. Estas dos cadenas son de tipo `xs:string`. Esta función no distingue entre mayúsculas y minúsculas. La comparación se basa en códigos ASCII. Si las cadenas son iguales, entonces el resultado es 0. Si `string1` es menor que `string2`, entonces el resultado es -1. Si `string1` es mayor que `string2`, entonces el resultado es 1.



Ejemplo:

```
string1: hi
string2: Hit
```

Compara cadenas de texto carácter por carácter. Aunque `h` está representada por un número de ASCII mayor que `H`, la función trata a estos dos caracteres por igual. El segundo carácter es el mismo en ambas cadenas. Sin embargo, `string2` tiene un tercer carácter, mientras que `string1` carece de él. El valor del tercer carácter de `string1` está vacío. El valor `t` de `string2` es mayor que el valor vacío de `string1`. Por tanto, `string1` es menor que `string2` y el resultado es -1.

 © 2016-2022 Altova GmbH

Lenguajes

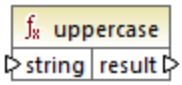
Disponible para Java, C#, C++ y el motor de ejecución integrado.

Parámetros

Nombre	Tipo	Descripción
`string1`	`xs:string`	La primera cadena de entrada.
`string2`	`xs:string`	La segunda cadena de entrada.

6.7.18.21 uppercase

El resultado es la cadena de entrada **string** en mayúsculas. Para los caracteres Unicode se usan los caracteres correspondientes en mayúsculas.



Lenguajes

Disponible para Java, C#, C++ y el motor de ejecución integrado.

Parámetros

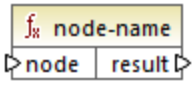
Nombre	Tipo	Descripción
string	`xs:string`	La cadena de entrada.

6.7.19 xpath2 | accessors (descriptores de acceso)

Las funciones de la biblioteca **xpath2 | accessors** obtienen información sobre nodos o elementos XML. Estas funciones están disponibles cuando se seleccionan los lenguajes XSLT2 o XQuery.

6.7.19.1 base-uri

La función **`base-uri`** toma un argumento de nodo como entrada y devuelve el URI del recurso XML que contiene el nodo El salida es de tipo `xs: string`.



Lenguajes

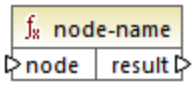
XQuery, XSLT 2.0, XSLT 3.0.

Parámetros

Nombre	Tipo	Descripción
nodo	`mf:node`	El nodo de entrada.

6.7.19.2 node-name

La función **`node-name`** toma un nodo como argumento de entrada y devuelve su QName. Cuando el Qname se representa como una cadena, toma la forma de `prefix:localname` si el nodo tiene un prefijo, o `localname` si no lo tiene. Para obtener el URI del espacio de nombres de un nodo, use la función **`namespace-uri-from-QName`** 680 .



Lenguajes

XQuery, XSLT 2.0, XSLT 3.0.

Parámetros

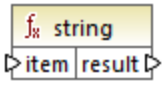
Nombre	Tipo	Descripción
nodo	`mf:node`	El nodo de entrada.

6.7.19.3 string

La función **`string`** funciona como el constructor `xs:string`: convierte su argumento en `xs:string`.

Cuando el argumento de entrada es un valor de un tipo atómico (por ejemplo `xs:decimal`), este valor atómico se convierte en un valor de tipo `xs:string`. Si el argumento de entrada es un nodo, se extrae el valor de cadena del nodo. (El valor de cadena de un nodo es una concatenación de los valores de los descendientes del nodo.)

 © 2016-2022 Altova GmbH



Lenguajes

XQuery, XSLT 2.0, XSLT 3.0.

Parámetros

Nombre	Tipo	Descripción
item	`mf:item`	El valor de entrada.

6.7.20 xpath2 | anyURI functions

La biblioteca secundaria **xpath2 | anyURI** contiene la función **`resolve-uri`**. Esta función está disponible cuando se seleccionan los lenguajes XSLT2 o XQuery.

6.7.20.1 resolve-uri

La función **`resolve-uri`** toma un URI relativo como su primer argumento y lo resuelve a partir del URI de base del segundo argumento. El resultado es del tipo de datos `xs: string`. La implementación de la función trata las dos entradas como cadenas y no se comprueba si los recursos que identifican esos URIs existen realmente.

Lenguajes

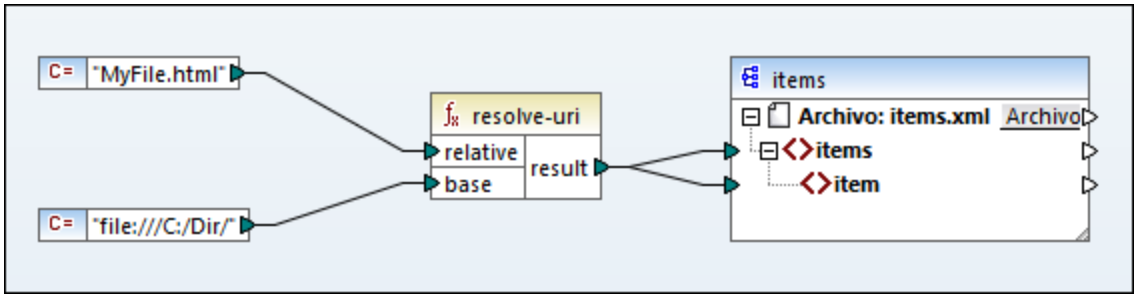
XQuery, XSLT 2.0, XSLT 3.0.

Parámetros

Nombre	Tipo	Descripción
relative	`xs:string`	El URI relativo que se quiere resolver a partir de la base.
base	`xs:string`	El URI de base.

Ejemplo

En la asignación siguiente, el primer argumento indica el URI relativo **`MyFile.html`** y el segundo el URI de base **`file:///C:/Dir/`**. El URI resuelto será una concatenación de ambos, es decir **`file:///C:/Dir/MyFile.html`**.

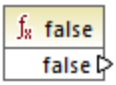


6.7.21 xpath2 | boolean functions (booleanas)

Las funciones booleanas de la biblioteca xpath2 | boolean son las funciones `true` y `false`. Estas funciones no toman ningún argumento y devuelven los valores booleanos de constante `true` y `false` respectivamente Estas funciones se pueden utilizar cuando se necesite un valor booleano de constante.

6.7.21.1 false

Devuelve el valor booleano **false**.

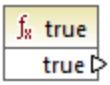


Lenguajes

XQuery, XSLT 2.0, XSLT 3.0.

6.7.21.2 true

Devuelve el valor booleano **true**.



Lenguajes

XQuery, XSLT 2.0, XSLT 3.0.

© 2016-2022 Altova GmbH

6.7.22 xpath2 | constructors (constructores)

Las funciones constructoras de la biblioteca de funciones xpath2 construyen ciertos tipos de datos a partir del texto de entrada. La tabla siguiente enumera las funciones constructoras disponibles.

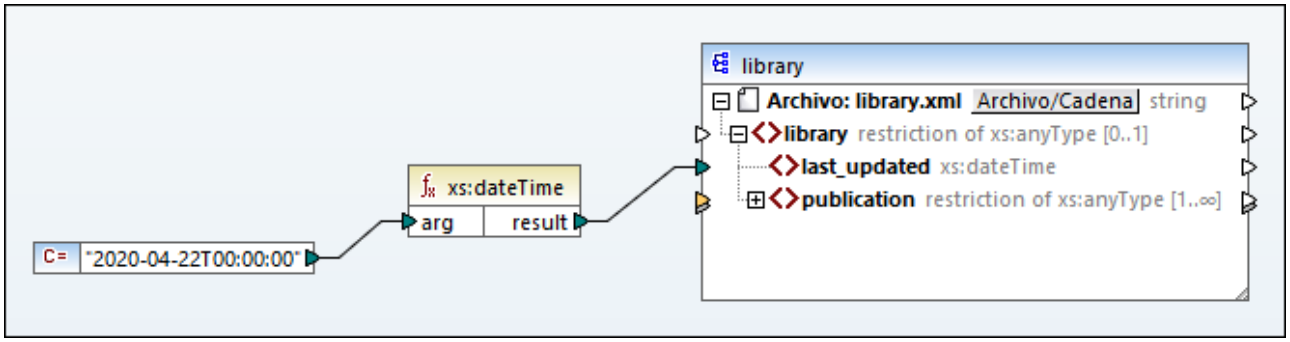
xs:ENTITY	**xs:double**	**xs:nonPositiveInteger**
xs:ID	**xs:duration**	**xs:normalizedString**
xs:IDREF	**xs:float**	**xs:positiveInteger**
xs:NCName	**xs:gDay**	**xs:short**
xs:NMTOKEN	**xs:gMonth**	**xs:string**
xs:Name	**xs:gMonthDay**	**xs:time**
xs:QName	**xs:gYear**	**xs:token**
xs:anyURI	**xs:gYearMonth**	**xs:unsignedByte**
xs:base64Binary	**xs:hexBinary**	**xs:unsignedInt**
xs:boolean	**xs:int**	**xs:unsignedLong**
xs:byte	**xs:integer**	**xs:unsignedShort**
xs:date	**xs:language**	**xs:untypedAtomic**
xs:dateTime	**xs:long**	**xs:yearMonthDuration**
xs:dayTimeDuration	**xs:negativeInteger**	
xs:decimal	**xs:nonNegativeInteger**	

Lenguajes

XQuery, XSLT 2.0, XSLT 3.0.

Ejemplo

Por lo general, el formato léxico del texto de entrada debe ser el formato esperado del tipo de datos que se debe construir. De lo contrario, la transformación no se puede realizar. Por ejemplo, si desea construir un tipo de datos `xs:dateTime`, utilice la función constructor **`xs:dateTime`**. El texto de entrada debe tener el formato léxico del tipo de datos `xs:dateTime`, es decir: **`AAAA-MM-DD:mm:ss`** (imagen siguiente).



En la imagen anterior, observe que se utilizó la constante **"2020-04-28T00:00:00"** como argumento de entrada de la función. La entrada también podría obtenerse de un elemento del documento de origen. La función **xs:dateTime** devuelve el texto de entrada **2020-04-28T00:00:00**, que es del tipo de datos `xs:dateTime`.

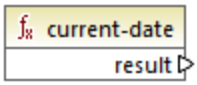
Al pasar el puntero del ratón encima del conector de entrada o de salida aparece el tipo de datos esperado para el elemento de la asignación (incluido el tipo de datos de los argumentos de la función).

6.7.23 xpath2 | context functions (contexto)

Las funciones de contexto de la biblioteca **xpath2 | context** ofrecen la hora y la fecha, la intercalación predeterminada utilizada por el procesador y el tamaño de la secuencia actual y la posición del nodo actual.

6.7.23.1 current-date

Devuelve la fecha actual (`xs:date`) del reloj del sistema.

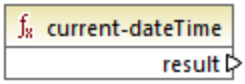


Lenguajes

XQuery, XSLT 2.0, XSLT 3.0.

6.7.23.2 current-dateTime

Devuelve la fecha y hora actual (`xs: dateTime`) del reloj del sistema.



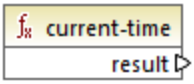
© 2016-2022 Altova GmbH

Lenguajes

XQuery, XSLT 2.0, XSLT 3.0.

6.7.23.3 current-time

Devuelve la hora actual (`xs:time`) del reloj del sistema.



Lenguajes

XQuery, XSLT 2.0, XSLT 3.0.

6.7.23.4 default-collation

La función **`default-collation`** no tiene argumentos y devuelve la intercalación predeterminada, es decir, la intercalación utilizada cuando no se especifica una intercalación para una función que la necesita.

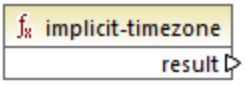
Las comparaciones (incluidas las funciones **`max-string`** y **`min-string`**) se basan en esta intercalación.

Lenguajes

XQuery, XSLT 2.0, XSLT 3.0.

6.7.23.5 implicit-timezone

Devuelve el valor de la propiedad "implicit timezone" del contexto de evaluación.

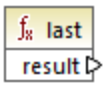


Lenguajes

XQuery, XSLT 2.0, XSLT 3.0.

6.7.23.6 last

Devuelve el número de elementos de la secuencia que se está procesando. Es importante recordar que la secuencia de los elementos la determina el contexto de asignación actual 874, como se describe en el ejemplo siguiente.



Lenguajes

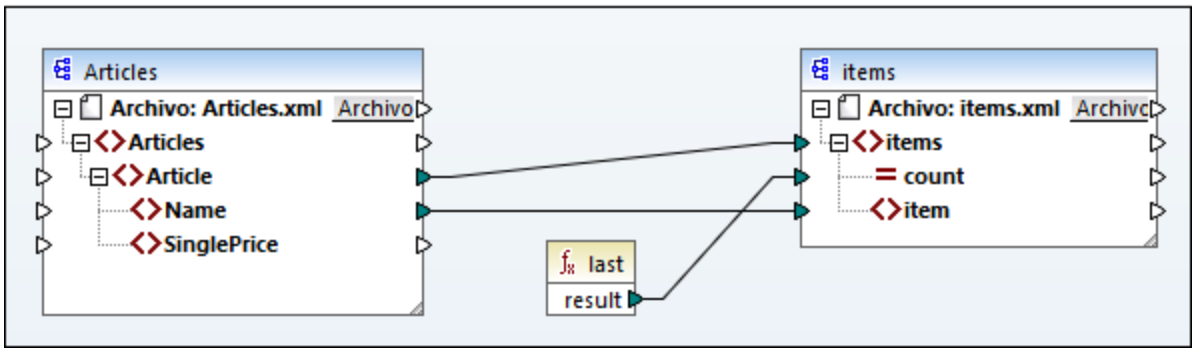
XQuery, XSLT 2.0, XSLT 3.0.

Ejemplo

Imagine que tiene que trabajar con este archivo XML de origen:

```
<Articles>
 <Article>
 <Name>T-Shirt</Name>
 <SinglePrice>25</SinglePrice>
 </Article>
 <Article>
 <Name>Socks</Name>
 <SinglePrice>2.30</SinglePrice>
 </Article>
 <Article>
 <Name>Jacket</Name>
 <SinglePrice>57.50</SinglePrice>
 </Article>
</Articles>
```

El objetivo es copiar datos en un archivo XML que tiene un esquema distinto. El recuento de todos los elementos también se debe guardar en el archivo XML de destino. Para ello puede usar una asignación como esta:



 © 2016-2022 Altova GmbH

En el ejemplo anterior la función **last** devuelve la posición del último nodo del contexto primario actual y rellena el atributo **count** con el valor **3**.

```
<items count="3">
 <item>T-Shirt</item>
 <item>Pants</item>
 <item>Jacket</item>
</items>
```

Tenga en cuenta que el valor **3** es la posición del último elemento (y el recuento del total de elementos) en el contexto de la asignación creado por la conexión entre los elementos **Article** y **items**. Los elementos se copian en el archivo de destino aunque no exista esa conexión, pero en ese caso la función **last** devuelve el valor incorrecto **1** porque no tiene un contexto primario 879 que recorrer. (Más concretamente, usaría el contexto implícito predeterminado creado entre los elementos raíz de ambos componentes, que produce una secuencia de 1 elemento y no de 3, como se esperaba).

Se recomienda usar la función **count** 635 de la biblioteca **core** en lugar de la función **last** porque la primera tiene un argumento **parent-context** que permite alterar el contexto de la asignación de forma explícita.

6.7.24 xpath2 | durations, date, time functions (duración, fecha y hora)

Las funciones de duración, fecha y hora de la biblioteca **xpath2** permiten ajustar las fechas y horas al uso horario, extraer componentes de los datos fecha/hora y sustraer una unidad fecha/hora a otra.

Ajustar fechas y horas

Para ajustar los valores de hora y fecha de un uso horario siga estos pasos:

- **adjust-date-to-timezone**
- **adjust-date-to-timezone** (con argumento timezone)
- **adjust-dateTime-to-timezone**
- **adjust-dateTime-to-timezone** (con argumento timezone)
- **adjust-time-to-timezone**
- **adjust-time-to-timezone** (con argumento timezone)

Cada una de estas funciones toma un valor `date`, `time` o `dateTime` como primer argumento y ajusta los datos de entrada añadiendo, eliminando o modificando el componente de uso horario en función del valor del segundo argumento, si lo hay.

Si el primer argumento no contiene uso horario (por ejemplo, la fecha `2020-01` o la hora `14:00:00`) hay tres posibilidades.

- Si está presente el argumento **timezone**, el resultado contendrá el uso horario especificado en el segundo argumento. Se añade el uso horario en el segundo argumento.
- Si falta el argumento **timezone**, el resultado contendrá el uso horario implícito, es decir, el del sistema. Se añade el uso horario del sistema.
- Si el argumento **timezone** está vacío, está vacío, el resultado no contendrá el uso horario.

Si el primer argumento contiene un uso horario (por ejemplo, la fecha `2020-01-01+01:00` o la hora `14:00:00+01:00`) hay tres posibilidades.

- Si está presente el argumento **timezone**, el resultado contendrá el uso horario especificado en el segundo argumento. El uso horario original es reemplazado por el del segundo argumento.
- Si falta el argumento **timezone**, el resultado contendrá el uso horario implícito, es decir, el del sistema. El uso horario original es reemplazado por el del sistema.
- Si el argumento **timezone**, está vacío, el resultado no contendrá el uso horario.

Extraer componentes de fechas y horas

Estas son las funciones para extraer valores numéricos como horas, minutos, días, meses, etc. de valores de fecha y hora:

- **day-from-date**
- **day-from-dateTime**
- **hours-from-dateTime**
- **hours-from-time**
- **minutes-from-dateTime**
- **minutes-from-time**
- **month-from-date**
- **month-from-dateTime**
- **seconds-from-dateTime**
- **seconds-from-time**
- **timezone-from-date**
- **timezone-from-dateTime**
- **imezone-from-time**
- **year-from-date**
- **year-from-dateTime**

Cada una de estas funciones extrae un componente en concreto de los valores `xs:date`, `xs:time`, `xs:dateTime` y `xs:duration`. El resultado puede ser `xs:integer` o `xs:decimal`.

Funciones From

Estas son las funciones para extraer componentes de hora de duraciones:

- **days-from-duration**
- **hours-from-duration**
- **minutes-from-duration**
- **months-from-duration**
- **seconds-from-duration**
- **years-from-duration**

La duración se puede indicar como `xs:yearMonthDuration` (para extraer años o meses) o como `xs:dayTimeDuration` (para extraer días, horas, minutos y segundos). Todas las funciones devuelven un resultado de tipo `xs:integer` excepto la función **seconds-from-duration**, que devuelve .`xs:decimal`.

Funciones Subtract

Estas son las funciones para sustraer valores de fecha y hora:

- **subtract-dateTimes**

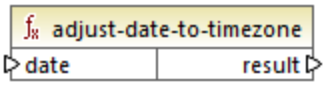
 © 2016-2022 Altova GmbH

- **subtract-dates**
- **subtract-times**

Cada una de estas funciones permite sustraer un valor de hora a otro y obtener un valor de duración como resultado.

6.7.24.1 adjust-date-to-timezone

Ajusta un valor `xs:date` a la zona horaria implícita en el contexto de evaluación (el huso horario del sistema).



Lenguajes

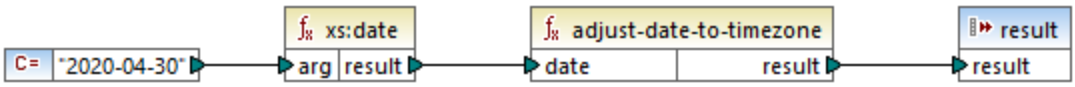
XQuery, XSLT 2.0.

Parámetros

Nombre	Tipo	Descripción
date	**xs:date**	El valor de entrada de tipo `xs:date`.

Ejemplo

Esta asignación genera un `xs:date` a partir de una cadena y lo pasa como argumento a la función **adjust-date-to-timezone**.

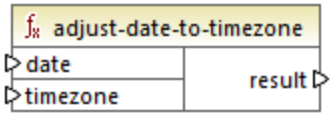


Asignación XSLT 2.0

Si la asignación se ejecuta en un equipo en el que el huso horario es +02:00, la función ajustará el valor de la fecha a este huso. En consecuencia, el resultado de la asignación será **2020-04-30+02:00**.

6.7.24.2 adjust-date-to-timezone

Ajusta un valor `xs:date` a un huso horario específico o a ninguno. Si el argumento **timezone** es una secuencia vacía, la función devuelve un elemento `xs:date` sin huso horario. De lo contrario lo devuelve con huso horario.



Lenguajes

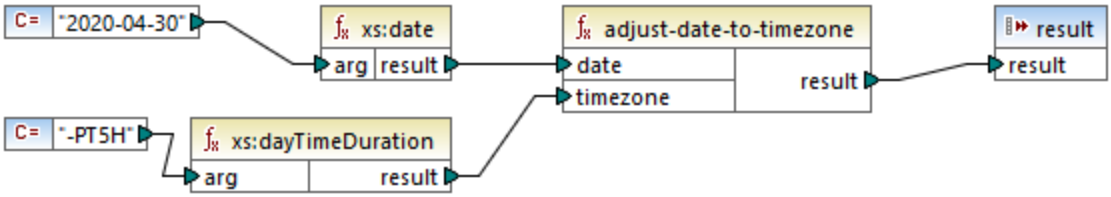
XQuery, XSLT 2.0.

Parámetros

Nombre	Tipo	Descripción
date	`xs:date`	El valor de entrada de tipo `xs:date`.
timezone	`xs:dayTimeDuration`	El huso horario expresado como un valor `xs:dayTimeDuration`. El valor puede ser negativo. Por ejemplo, un huso horario de -5 horas se puede expresar como **`-PT5H`**.

Ejemplo

Esta asignación genera los dos parámetros de la función **`adjust-date-to-timezone`** a partir de cadenas usando las funciones XPath 2 constructor 779 correspondientes. El objetivo de la asignación es ajustar el huso horario a -5 horas. Este huso horario se puede expresar como **`-PT5H`**.



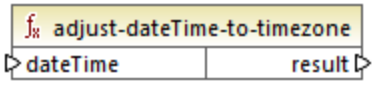
Asignación XSLT 2.0

La función ajusta el valor de la fecha al huso horario dado como argumento. En consecuencia, el resultado de la asignación es **`2020-04-30-05:00`**.

6.7.24.3 adjust-dateTime-to-timezone

Ajusta un valor `xs:dateTime` al huso horario implícito en el contexto de evaluación (el huso horario del sistema).

© 2016-2022 Altova GmbH



Lenguajes

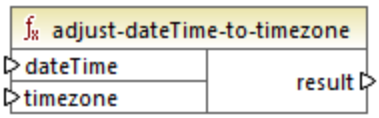
XQuery, XSLT 2.0.

Parámetros

Nombre	Tipo	Descripción
dateTime	`xs:dateTime`	El valor de entrada de tipo `xs:dateTime`.

6.7.24.4 adjust-dateTime-to-timezone

Ajusta un valor `xs:dateTime` a un huso horario específico o a ninguno. Si el argumento **timezone** es una secuencia vacía, la función devuelve un elemento `xs:dateTime` sin huso horario. De lo contrario lo devuelve con huso horario.



Lenguajes

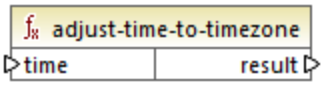
XQuery, XSLT 2.0.

Parámetros

Nombre	Tipo	Descripción
dateTime	`xs:dateTime`	El valor de entrada de tipo `xs:dateTime`.
timezone	`xs:dayTimeDuration`	El huso horario expresado como un valor `xs:dayTimeDuration`. El valor puede ser negativo. Por ejemplo, un huso horario de -5 horas se puede expresar como `-PT5H`.

6.7.24.5 adjust-time-to-timezone

Ajusta un valor `xs:time` al huso horario implícito en el contexto de evaluación (el huso horario del sistema).



Lenguajes

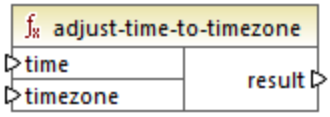
XQuery, XSLT 2.0.

Parámetros

Nombre	Tipo	Descripción
time	`xs:time`	El valor de entrada de tipo `xs:time`.

6.7.24.6 adjust-time-to-timezone

Ajusta un valor `xs:time` a un huso horario específico o a ninguno. Si el argumento **timezone** es una secuencia vacía, la función devuelve un elemento `xs:time` sin huso horario. De lo contrario lo devuelve con huso horario.



Lenguajes

XQuery, XSLT 2.0.

Parámetros

Nombre	Tipo	Descripción
time	`xs:time`	El valor de entrada de tipo `xs:time`.
timezone	`xs:dayTimeDuration`	El huso horario expresado como un valor `xs:dayTimeDuration`. El valor puede ser negativo. Por ejemplo, un huso horario de -5

© 2016-2022 Altova GmbH

Nombre	Tipo	Descripción
		horas se puede expresar como **-PT5H**.

6.7.24.7 day-from-date

Devuelve un elemento `xs:integer` que representa la parte del día del valor `xs:date` dado como argumento.

Lenguajes

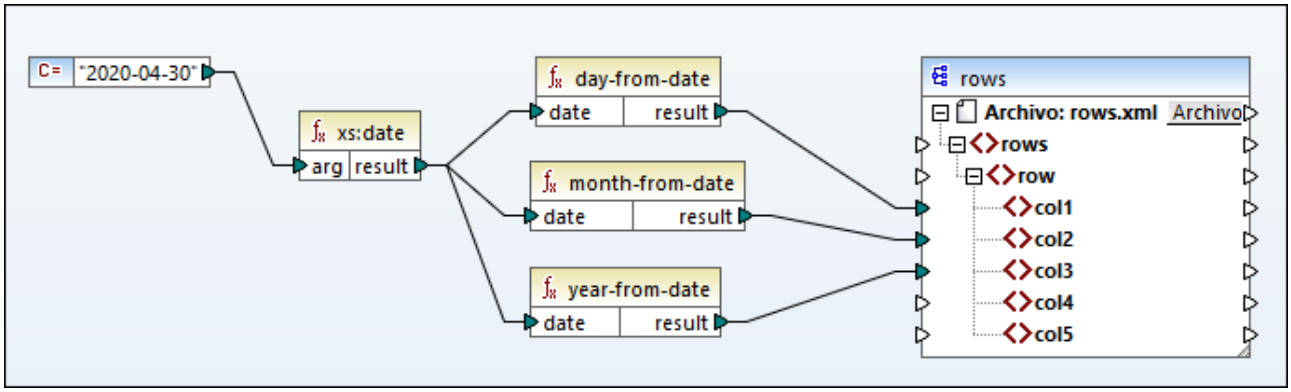
XQuery, XSLT 2.0.

Parámetros

Nombre	Tipo	Descripción
date	`xs:date`	El valor de entrada de tipo `xs:date`.

Ejemplo

Esta asignación convierte una cadena en `xs:date` con la función constructora **`xs:date`**. Las funciones **`day-from-date`**, **`month-from-date`** y **`year-from-date`** extraen las respectivas partes de la fecha y las escriben en un elemento aparte en el archivo XML de destino.



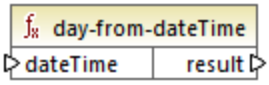
Asignación XQuery 1.0

El resultado de la asignación es:

```
<rows>
   <row>
      <col1>30</col1>
      <col2>4</col2>
      <col3>2020</col3>
   </row>
</rows>
```

6.7.24.8 day-from-dateTime

Devuelve un elemento `xs:integer` que representa la parte del día del valor `xs:dateTime` dado como argumento.



Lenguajes

XQuery, XSLT 2.0.

Parámetros

Nombre	Tipo	Descripción
dateTime	`xs:dateTime`	El valor de entrada de tipo `xs:dateTime`.

6.7.24.9 days-from-duration

Devuelve un elemento `xs:integer` que representa el componente "días" de la representación canónica del valor de duración dado como argumento.

Lenguajes

XQuery, XSLT 2.0.

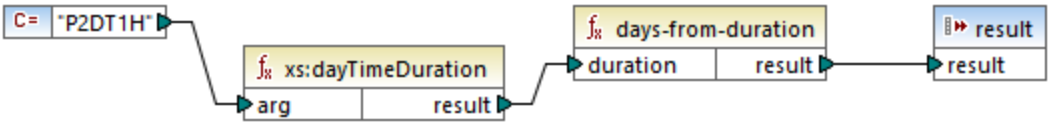
Parámetros

Nombre	Tipo	Descripción
duration	`xs:duration`	El valor de entrada de tipo `xs:duration`.

© 2016-2022 Altova GmbH

Ejemplo

Esta asignación genera el `xs:dayTimeDuration` de **P2DT1H** (2 días y 1 hora) y lo para como entrada a la función **days-from-duration**. El resultado es **2**.



Asignación XSLT 2.0

Nota: si la duración es **P1DT24H** (2 días y 1 hora), la función devuelve **2**, no **1** porque la representación canónica de **P1DT24H** es **P2D** (2 días).

6.7.24.10 hours-from-dateTime

Devuelve un elemento `xs:integer` que representa la parte de las horas del valor `xs:dateTime` dado como argumento.

Lenguajes

XQuery, XSLT 2.0.

Parámetros

Nombre	Tipo	Descripción
dateTime	`xs:dateTime`	El valor de entrada de tipo `xs:dateTime`.

6.7.24.11 hours-from-duration

Devuelve un elemento `xs:integer` que representa el componente horas de la la representación canónica del valor de duración dado como argumento.

Lenguajes

XQuery, XSLT 2.0.

Parámetros

Nombre	Tipo	Descripción
duration	`xs:duration`	El valor de entrada de tipo `xs:duration`.

Ejemplo

Si la duración es **PT1H60M** (1 hora y 60 minutos), la función devuelve **2**, no **1** porque la representación canónica **PT1H60M** es **PT2H** (2 días).

6.7.24.12 hours-from-time

Devuelve un elemento `xs:integer` que representa la parte de las horas del valor `xs:time` dado como argumento.

Lenguajes

XQuery, XSLT 2.0.

Parámetros

Nombre	Tipo	Descripción
time	**xs:time**	El valor de entrada de tipo `xs:time`.

6.7.24.13 minutes-from-dateTime

Devuelve un elemento `xs:integer` que representa la parte de los minutos del valor `xs:dateTime` dado como argumento.

Lenguajes

XQuery, XSLT 2.0.

Parámetros

Nombre	Tipo	Descripción
dateTime	**xs:dateTime**	El valor de entrada de tipo `xs:dateTime`.

6.7.24.14 minutes-from-duration

Devuelve un elemento `xs:integer` que representa el componente "minutos" de la la representación canónica del valor de duración dado como argumento.

Lenguajes

XQuery, XSLT 2.0.

© 2016-2022 Altova GmbH

Parámetros

Nombre	Tipo	Descripción
duration	`xs:duration`	El valor de entrada de tipo `xs:duration`.

Ejemplo

Si la duración es **`PT1M60S`** (1 minuto y 60 segundos), la función devuelve **2**, no **1** porque la representación canónica **`PT1M60S`** es **`PT2M`** (2 minutos).

6.7.24.15 minutes-from-time

Devuelve un elemento `xs:integer` que representa la parte de los minutos del valor `xs:time` dado como argumento.

Lenguajes

XQuery, XSLT 2.0.

Parámetros

Nombre	Tipo	Descripción
time	`xs:time`	El valor de entrada de tipo `xs:time`.

6.7.24.16 month-from-date

Devuelve un elemento `xs:integer` que representa la parte del mes del valor `xs:date` dado como argumento.

Lenguajes

XQuery, XSLT 2.0.

Parámetros

Nombre	Tipo	Descripción
date	`xs:date`	El valor de entrada de tipo `xs:date`.

6.7.24.17 month-from-dateTime

Devuelve un elemento `xs:integer` que representa la parte del mes del valor `xs:dateTime` dado como argumento.

Lenguajes

XQuery, XSLT 2.0.

Parámetros

Nombre	Tipo	Descripción
dateTime	`xs:dateTime`	El valor de entrada de tipo `xs:dateTime`.

6.7.24.18 months-from-duration

Devuelve un elemento `xs:integer` que representa el componente "meses" de la representación canónica del valor de duración dado como argumento.

Lenguajes

XQuery, XSLT 2.0.

Parámetros

Nombre	Tipo	Descripción
duration	`xs:duration`	El valor de entrada de tipo `xs:duration`.

6.7.24.19 seconds-from-dateTime

Devuelve un elemento `xs:integer` que representa la parte de los segundos del valor localizado de dateTime.

Lenguajes

XQuery, XSLT 2.0.

Parámetros

Nombre	Tipo	Descripción
dateTime	`xs:dateTime`	

© 2016-2022 Altova GmbH

6.7.24.20 seconds-from-duration

Devuelve un elemento `xs:integer` que representa el componente "segundos" de la la representación canónica del valor de duración dado como argumento.

Lenguajes

XQuery, XSLT 2.0.

Parámetros

Nombre	Tipo	Descripción
duration	`xs:duration`	El valor de entrada de tipo `xs:duration`.

6.7.24.21 seconds-from-time

Devuelve un elemento `xs:integer` que representa la parte del mes del valor `xs:time` dado como argumento.

Lenguajes

XQuery, XSLT 2.0.

Parámetros

Nombre	Tipo	Descripción
time	`xs:time`	El valor de entrada de tipo `xs:time`.

6.7.24.22 subtract-dateTimes

Devuelve el elemento `xs:dayTimeDuration` que corresponde a la diferencia entre el valor normalizado de **dateTime1** y el valor normalizado de **dateTime2**.

Lenguajes

XQuery, XSLT 2.0.

Parámetros

Nombre	Tipo	Descripción
dateTime1	`xs:dateTime`	El primer valor de entrada.

Nombre	Tipo	Descripción
dateTime2	`xs:dateTime`	El segundo valor de entrada.

6.7.24.23 subtract-dates

Devuelve el elemento `xs:dayTimeDuration` que corresponde a la diferencia entre el valor normalizado de **date1** y el valor normalizado de **date2**.

Lenguajes

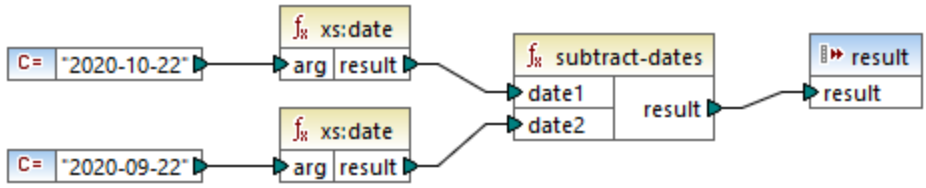
XQuery, XSLT 2.0.

Parámetros

Nombre	Tipo	Descripción
date1	`xs:date`	El primer valor de entrada.
date2	`xs:date`	El segundo valor de entrada.

Ejemplo

Esta asignación sustrae dos fechas (2020-10-22 menos 2020-09-22). El resultado es el valor **`P30D`** de tipo `xs:dayTimeDuration`., que representa una duración de 30 días.



6.7.24.24 subtract-times

Devuelve el elemento `xs:dayTimeDuration` que corresponde a la diferencia entre el valor normalizado de **time1** y el valor normalizado de **time2**.

Lenguajes

XQuery, XSLT 2.0.

Parámetros

Nombre	Tipo	Descripción
time1	`xs:time`	El primer valor de entrada.

© 2016-2022 Altova GmbH

Nombre	Tipo	Descripción
time2	`xs:time`	El segundo valor de entrada.

6.7.24.25 timezone-from-date

Devuelve el componente de huso horario de la fecha dada como argumento. El resultado es un componente `xs:dayTimeDuration` que indica una desviación de UTC; su valor puede ir de +14:00 horas a -14:00 horas, ambas incluidas.

Lenguajes

XQuery, XSLT 2.0.

Parámetros

Nombre	Tipo	Descripción
date	`xs:date`	El valor de entrada de tipo type `xs:date`.

6.7.24.26 timezone-from-dateTime

Devuelve el componente de huso horario del valor `xs:dateTime` dado como argumento. El resultado es un componente `xs:dayTimeDuration` que indica una desviación de UTC; su valor puede ir de +14:00 horas a -14:00 horas, ambas incluidas.

Lenguajes

XQuery, XSLT 2.0.

Parámetros

Nombre	Tipo	Descripción
dateTime	`xs:dateTime`	El valor de entrada de tipo `xs:dateTime`.

6.7.24.27 timezone-from-time

Devuelve el componente de huso horario del valor `xs:time` dado como argumento. El resultado es un componente `xs:dayTimeDuration` que indica una desviación de UTC; su valor puede ir de +14:00 horas a -14:00 horas, ambas incluidas.

Lenguajes

XQuery, XSLT 2.0.

Parámetros

Nombre	Tipo	Descripción
time	`xs:time`	El valor de entrada de tipo `xs:time`.

6.7.24.28 year-from-date

Devuelve un elemento `xs:integer` que representa la parte del año del valor `xs:date` dado como argumento.

Lenguajes

XQuery, XSLT 2.0.

Parámetros

Nombre	Tipo	Descripción
date	`xs:date`	El valor de entrada de tipo `xs:date`.

6.7.24.29 year-from-dateTime

Devuelve un elemento `xs:integer` que representa la parte del año del valor `xs:dateTime` dado como argumento.

Lenguajes

XQuery, XSLT 2.0.

Parámetros

Nombre	Tipo	Descripción
dateTime	`xs:dateTime`	El valor de entrada de tipo `xs:dateTime`.

© 2016-2022 Altova GmbH

6.7.24.30 years-from-duration

Devuelve un elemento `xs:integer` que representa el componente "años" de la representación léxica canónica del valor de duración dado como argumento.

Lenguajes

XQuery, XSLT 2.0.

Parámetros

Nombre	Tipo	Descripción
duration	`xs:duration`	El valor de entrada de tipo `xs:duration`.

6.7.25 xpath2 | node functions (nodo)

Las funciones de nodo de la biblioteca **xpath2** ofrecen información sobre los nodos (elementos) de un componente de asignación.

La función **`lang`** toma como argumento una cadena que identifica un código de idioma (como "en"). La función devuelve **true** o **false** dependiendo de si el nodo de contexto tiene un atributo `xml:lang` con el valor especificado por el argumento de la función.

Las funciones **`local-name`**, **`name`** y **`namespace-uri`** devuelven el nombre local, el nombre y el URI de espacio de nombres respectivamente del nodo de entrada. Por ejemplo, en el caso del nodo **altova:Products**, el nombre local es **`Products`**, el nombre es **`altova:Products`** y el URI de espacio de nombres es el URI del espacio de nombres al que está enlazado el prefijo altova: (consulte el ejemplo de la función **`local-name`** 801). Cada una de estas tres funciones tiene dos variantes:

- Sin argumento: la función se aplica al nodo de contexto (para ver un ejemplo del nodo de contexto consulte el ejemplo de la función **`lang`** 799).
- Con argumento (que debe ser un nodo): la función se aplica al nodo especificado.

La función **`number`** toma un nodo como entrada, atomiza el nodo (es decir, extrae su contenido), convierte el valor en un decimal y devuelve el valor convertido. Esta función tiene dos variantes:

- Sin argumento: la función se aplica al nodo de contexto (para ver un ejemplo del nodo de contexto consulte el ejemplo de la función **`lang`** 799).
- Con argumento (que debe ser un nodo): la función se aplica al nodo especificado.

6.7.25.1 lang

Devuelve **true** si el nodo de contexto tiene un atributo `xml:lang` con un valor que coincide exactamente con el argumento **testlang** o es un subconjunto del mismo. De lo contrario devuelve **false**.

Lenguajes

XQuery, XSLT 2.0.

Parámetros

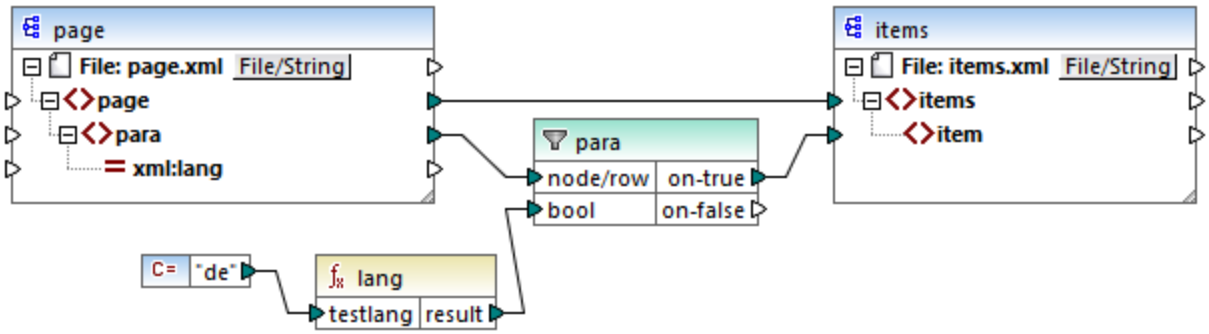
Nombre	Tipo	Descripción
testlang	`xs:string`	El código del idioma que se comprueba, por ejemplo "en".

Ejemplo

Este ejemplo XML contiene elementos **para** con distintos valores para el atributo `xml:lang`.

```
<page>
   <para xml:lang="en">Good day!</para>
   <para xml:lang="fr">Bonjour!</para>
   <para xml:lang="de-AT">Grüss Gott!</para>
   <para xml:lang="de-DE">Guten Tag!</para>
   <para xml:lang="de-CH">Grüezi!</para>
</page>
```

La asignación siguiente filtra solamente los párrafos en alemán, independientemente de la variante del país, gracias a la función **`lang`**.



Asignación XSLT 2.0

En la asignación anterior, por cada **para** de origen se crea un **item** en el componente de destino, de forma condicional. La condición la da un filtro que pasa al componente de destino solo los nodos en los que la función **`lang`** devuelve **true**. Es decir, solo los nodos cuyo atributo `xml:lang` es "de" (o un subconjunto de "de") cumplen la condición del filtro. En consecuencia, el resultado de la asignación es:

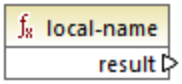
 © 2016-2022 Altova GmbH

```
<items>
   <item>Grüss Gott!</item>
   <item>Guten Tag!</item>
   <item>Grüezi!</item>
</items>
```

Observe que la función **lang** opera en el contexto de cada **para** debido a la conexión primaria entre **para** and **item**, véase también Contexto de la asignación 874.

6.7.25.2 local-name

Devuelve la parte local del nombre del nodo de contexto como `xs:string`. Es una variante sin parámetros de la función **local-name** donde el nodo de contexto lo determinan las conexiones de la asignación. Para indicar un nodo de forma explícita use la función **local-name** 801 que toma un nodo de entrada como parámetro.

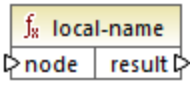


Lenguajes

XQuery, XSLT 2.0.

6.7.25.3 local-name

Devuelve la parte local del nombre del nodo como `xs:string`.



Lenguajes

XQuery, XSLT 2.0.

Parámetros

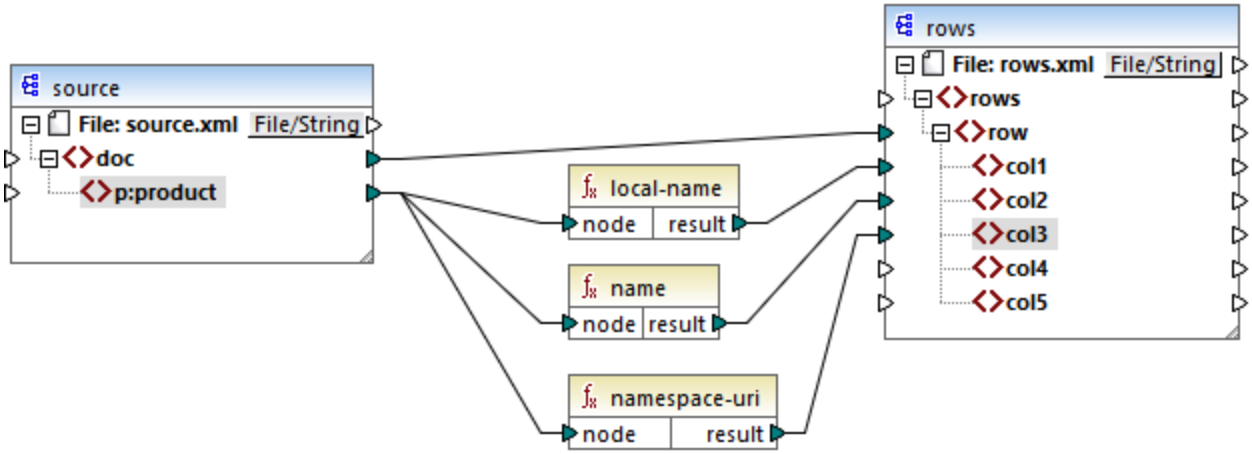
Nombre	Tipo	Descripción
node	**node()**	El nodo de entrada.

Ejemplo

En este archivo XML el nombre del elemento `p:product` es un nombre cualificado prefijado (QName). El prefijo "p" está asignado al espacio de nombres "http://mycompany.com".

```
<?xml version="1.0" encoding="UTF-8"?>
<doc xmlns:p="http://mycompany.com" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:noNamespaceSchemaLocation="source.xsd">
   <p:product/>
</doc>
```

Esta asignación extrae el nombre local, el nombre y el URI de espacio de nombres del nodo y escribe esos valores en un archivo de destino:



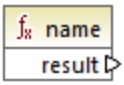
Asignación XSLT 2.0

A continuación puede ver el resultado de la asignación. Cada elemento **col** muestra el resultado de las funciones **`local-name`**, **`name`** y **`namespace-uri`**, respectivamente.

```
<rows>
   <row>
      <col1>product</col1>
      <col2>p:product</col2>
      <col3>http://mycompany.com</col3>
   </row>
</rows>
```

6.7.25.4 name

Devuelve el nombre del nodo de contexto. Es una variante sin parámetros de la función **`name`** donde el nodo de contexto lo determinan las conexiones de la asignación. Para indicar un nodo de forma explícita use la función **`name`** 803 que toma un nodo de entrada como parámetro.

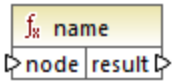


Lenguajes

XQuery, XSLT 2.0.

6.7.25.5 name

Devuelve el nombre de un nodo.



Lenguajes

XQuery, XSLT 2.0.

Parámetros

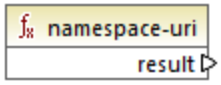
Nombre	Tipo	Descripción
node	`node()`	El nodo de entrada.

Ejemplo

Consulte el ejemplo de la función `local-name` 801.

6.7.25.6 namespace-uri

Devuelve el URI de espacio de nombres del QName del nodo de contexto, como `xs:string`. Es una variante sin parámetros de la función `namespace-uri` donde el nodo de contexto lo determinan las conexiones de la asignación. Para indicar un nodo de forma explícita use la función `namespace-uri` 804 que toma un nodo de entrada como parámetro.



Lenguajes

XQuery, XSLT 2.0.

6.7.25.7 namespace-uri

Devuelve el URI de espacio de nombres del QName del nodo como `xs:string`.

Lenguajes

XQuery, XSLT 2.0.

Parámetros

Nombre	Tipo	Descripción
node	**node()**	El nodo de entrada.

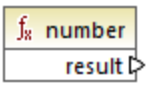
Ejemplo

Consulte el ejemplo de la función **local-name** 801.

6.7.25.8 number

Devuelve el valor del nodo de contexto convertido en `xs:double`. Es una variante sin parámetros de la función **number** donde el nodo de contexto lo determinan las conexiones de la asignación. Para indicar un nodo de forma explícita use la función **number** 804 que toma un nodo de entrada como parámetro.

Los únicos tipos que se pueden convertir en números son booleanos, cadenas numéricas y otros tipos numéricos. Los valores de entrada no numéricos (como cadenas no numéricas) resultan en NaN (Not a Number; *no es un número*).



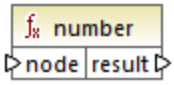
Lenguajes

XQuery, XSLT 2.0.

6.7.25.9 number

Devuelve el valor del nodo convertido en `xs:double`. Los únicos tipos que se pueden convertir en números son booleanos, cadenas numéricas y otros tipos numéricos. Los valores de entrada no numéricos (como cadenas no numéricas) resultan en NaN (Not a Number; *no es un número*).

© 2016-2022 Altova GmbH



Lenguajes

XQuery, XSLT 2.0.

Parámetros

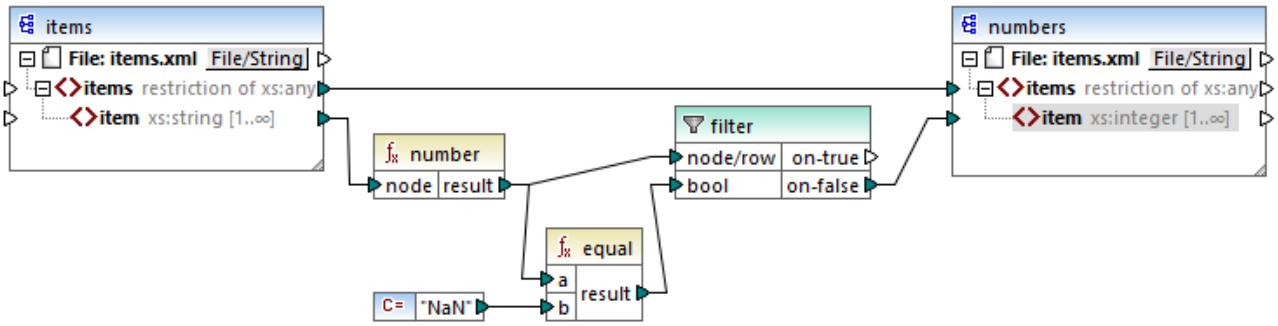
Nombre	Tipo	Descripción
node	`mf:atomic`	El nodo de entrada.

Ejemplo

Este XML contiene elementos de tipo `string`:

```
<items>
   <item>1</item>
   <item>2</item>
   <item>Jingle Bells</item>
</items>
```

La ilustración siguiente pretende convertir todas estas cadenas en valores numéricos y escribirlas en un archivo XML de destino. Observe que el tipo de datos del elemento **item** en el componente XML de destino es `xs:integer`, mientras que el elemento **item** de origen es de tipo `xs:string`. Si la conversión no se realiza correctamente, ese elemento se debe omitir y no se copiará en el archivo de destino.



Asignación XSLT 2.0

Para alcanzar el objetivo de la asignación se usó un filtro. La función **`equal`** comprueba si el resultado de la conversión es "NaN". Si no es así, indica que la conversión se realizó correctamente, por lo que el elemento sí se copia en el archivo de destino. El resultado de la asignación es el siguiente:

```
<items>
```

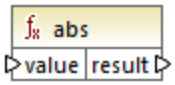
```
   <item>1</item>
   <item>2</item>
</items>
```

6.7.26 xpath2 | numeric functions (numéricas)

La biblioteca de funciones **xpath2 | numeric** ofrece las funciones numéricas **abs** y **round-half-to-even**.

6.7.26.1 abs

Devuelve el valor absoluto del argumento. Por ejemplo, si el argumento es **-2** o **2**, la función devuelve **2**.



Lenguajes

XQuery, XSLT 2.0.

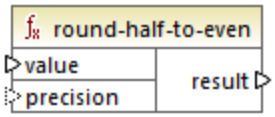
Parámetros

Nombre	Tipo	Descripción
value	`xs:decimal`	El valor de entrada.

6.7.26.2 round-half-to-even

La función **round-half-to-even** redondea el número indicado (primer argumento) con el grado de precisión (número de decimales) indicado en el segundo argumento. Por ejemplo, si el primer argumento es **2,141567** y el segundo es **3**, el número del primer argumento se redondea con tres decimales, por lo que el resultado es **2,141**. Si no se indica el segundo argumento el número se redondea sin decimales.

El elemento "even" de la función hace referencia al redondeo a un número par si el dígito indicado está entre dos valores. Por ejemplo, `round-half-to-even(3.475, 2)` devolvería **3,48**.



Lenguajes

XQuery, XSLT 2.0.

© 2016-2022 Altova GmbH

Parámetros

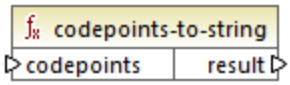
Nombre	Tipo	Descripción
value	`xs:decimal`	Obligatorio. Indica el valor de entrada que se redondea.
precision	`xs:integer`	Opcional. Indica el número de decimales del redondeo. El valor predeterminado es **0**.

6.7.27 xpath2 | string functions (cadena)

La biblioteca de funciones **xpath2 | string** permiten procesar cadenas (compararlas, convertirlas en mayúsculas o minúsculas, extraer subcadenas, etc.).

6.7.27.1 codepoints-to-string

Crea una cadena a partir de una secuencia de puntos de código Unicode. Esta función es opuesta a la función `string-to-codepoints` 815.



Lenguajes

XQuery, XSLT 2.0.

Parámetros

Nombre	Tipo	Descripción
codepoints	`ZeroOrMore xs:integer`	Esta entrada debe estar conectada a una secuencia de elementos de tipo integer, donde cada uno de ellos indica un punto de código Unicode.

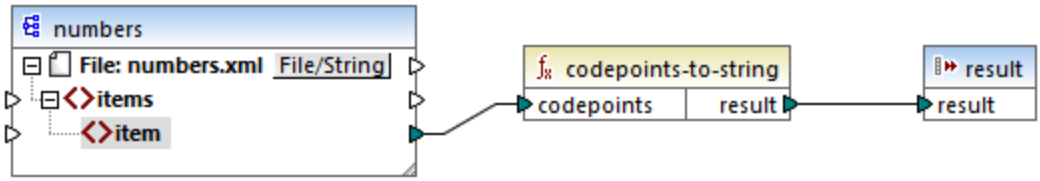
Ejemplo

Este XML contiene varios elementos **item** que almacenan valores de puntos de código Unicode.

```
<items>
   <item>77</item>
   <item>97</item>
```

```
    <item>112</item>
    <item>70</item>
    <item>111</item>
    <item>114</item>
    <item>99</item>
    <item>101</item>
</items>
```

Esta asignación suministra la secuencia de elementos como argumento a la función **codepoint-to-string**.



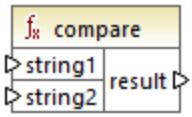
Asignación XSLT 2.0

El resultado de la función es **MapForce**.

6.7.27.2 compare

La función **compare** toma como argumentos dos cadenas y las compara. Si **string1** es alfabéticamente menor que **string2** (por ejemplo, las cadenas son "A" y "B") la función devuelve **-1**. Si son iguales ("A" y "A") la función devuelve **0**. Si **string1** es mayor que **string2** (por ejemplo "B" y "A"), entonces la función devuelve **1**.

Esta variante de las funciones usa el elemento collation predeterminado, que es Unicode. Existe otra variante 809 de esta función en la que puede usar el elemento collation como argumento.



Lenguajes

XQuery, XSLT 2.0.

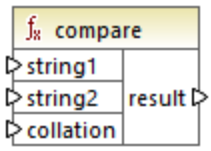
Parámetros

Nombre	Tipo	Descripción
string1	xs:string	La primera cadena de entrada.
string2	xs:string	La segunda cadena de entrada.

© 2016-2022 Altova GmbH

6.7.27.3 compare

La función **compare** toma como argumentos dos cadenas y las compara usando el elemento collation dado como argumento. Si **string1** está más abajo en el alfabeto que **string2** (por ejemplo, las dos cadenas son "A" y "B"), entonces la función devuelve **-1**. Si las dos cadenas son iguales ("A" y "A") la función devuelve **0**. Si **string1** es mayor que **string2** (por ejemplo "B" y "A"), entonces la función devuelve **1**.



Lenguajes

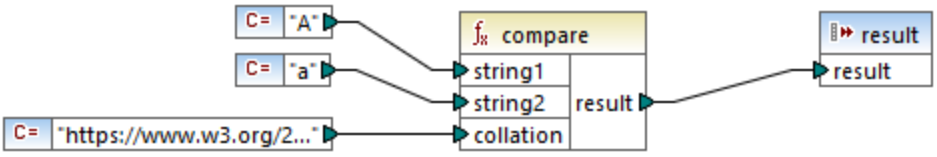
XQuery, XSLT 2.0.

Parámetros

Nombre	Tipo	Descripción
string1	`xs:string`	La primera cadena de entrada.
string2	`xs:string`	La primera cadena de entrada.
collation	`xs:string`	Indica qué cotejo usar para la comparación de cadenas. Puede provenir de la entrada de la función **default-collation** 781 o puede ser un cotejo como `http://www.w3.org/2005/xpath-functions/collation/html-ascii-case-insensitive`.

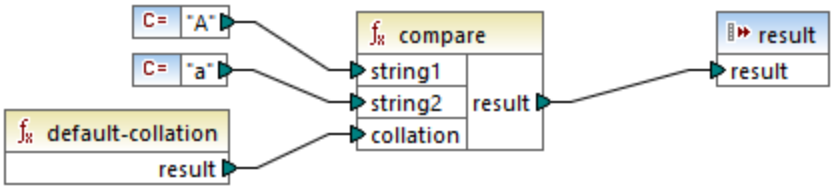
Ejemplo

Esta asignación compara las cadenas "A" y "a" usando el elemento collation que no distingue entre mayúsculas y minúsculas `http://www.w3.org/2005/xpath-functions/collation/html-ascii-case-insensitive`, que viene dado por una constante.



Asignación XSLT 2.0

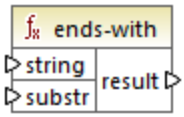
El resultado de la asignación anterior es **0** porque las dos se consideran iguales. Sin embargo, si reemplazamos el elemento collation con el que nos da la función `default-collation`, que usa los puntos de código Unicode predeterminados, el resultado se convierte en **-1** porque "A" es alfabéticamente menos que "a".



6.7.27.4 ends-with

Devuelve **true** si **string** termina en **substr**, de lo contrario devuelve **false**. El valor devuelto es de tipo `xs:boolean`.

Esta variante de la función usa el elemento collation predeterminado, que es Unicode. Existe otra variante 810 de esta función en la que puede usar el elemento collation como argumento.



Lenguajes

XQuery, XSLT 2.0.

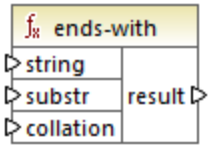
Parámetros

Nombre	Tipo	Descripción
string	`xs:string`	La cadena de entrada (el "pajar").
substr	`xs:string`	La subcadena (la "aguja").

6.7.27.5 ends-with

Devuelve **true** si **string** termina en **substr**, de lo contrario devuelve **false**. El valor devuelto es de tipo `xs:boolean`.

 © 2016-2022 Altova GmbH



Lenguajes

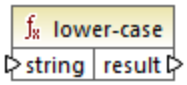
XQuery, XSLT 2.0.

Parámetros

Nombre	Tipo	Descripción
string	`xs:string`	La cadena de entrada (el "pajar").
substr	`xs:string`	La subcadena (la "aguja").
collation	`xs:string`	Indica qué cotejo usar para la comparación de cadenas. Puede provenir de la entrada de la función `default-collation` 781 o puede ser un cotejo como `http://www.w3.org/2005/xpath-functions/collation/html-ascii-case-insensitive`.

6.7.27.6 lower-case

Devuelve el valor de **string** después de pasar cada uno de sus caracteres a su versión en minúsculas.



Lenguajes

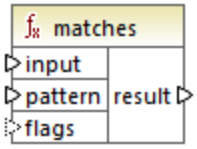
XQuery, XSLT 2.0.

Parámetros

Nombre	Tipo	Descripción
string	`xs:string`	El valor de entrada.

6.7.27.7 matches

La función **matches** comprueba si una cadena dada (el primer argumento) coincide con una expresión regular 8segundo argumento). La sintaxis de la expresión regular debe ser la definida por el aspecto `pattern` del esquema XML. La función devuelve **true** si la cadena coincide con la expresión regular; de lo contrario devuelve **false**.



Lenguajes

XQuery, XSLT 2.0.

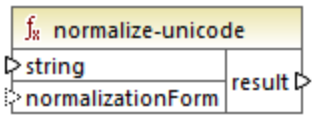
Parámetros

Nombre	Tipo	Descripción
input	`xs:string`	La cadena de entrada.
pattern	`xs:string`	La expresión regular, véase Expresiones regulares 627.
flags	`xs:string`	Argumento opcional que influye en las coincidencias. Este argumento puede dar cualquier combinación de los elementos flag: `i`, `m`, `s`, `x`. Se pueden usar muchos flags, por ejemplo, `imx`. Si no se indica ninguno, se usan los valores predeterminados de los cuatro flags, que son: `i` Ignorar mayúsculas y minúsculas. El valor predeterminado es no ignorarlas. `m` Usar modo multilínea, en el que se considera que la cadena de entrada tiene varias líneas, cada una separada por un carácter de línea nueva (`x0a`). Los metacaracteres **^** u **$** indican el principio y final de cada línea. El modo predeterminado es de cadena, donde las cadenas empiezan y terminan con los caracteres **^** and **$**. `s` Usar el modo *dot-all*. El modo predeterminado es no usarlo, donde el metacarácter **.** encuentra todos los caracteres excepto el de línea nueva (`x0a`). En el modo *dot-all* el punto también encuentra la línea nueva. `x` Ignora los espacios en blanco. El modo predeterminado es no ignorarlos.

© 2016-2022 Altova GmbH

6.7.27.8 normalize-unicode

Devuelve el valor de **string** normalizado conforme a las reglas de normalización de la forma normal indicada (segundo argumento). Para más información sobre la normalización Unicode consulte §2.2 en https://www.w3.org/TR/charmod-norm/.



Lenguajes

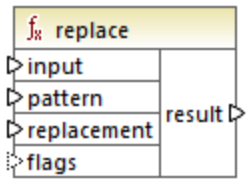
XQuery, XSLT 2.0.

Parámetros

Nombre	Tipo	Descripción
string	`xs:string`	El valor de cadena que se quiere normalizar.
normalizationForm	`xs:string`	El argumento opcional que da la forma normal. La predeterminada es la Unicode Normalization Form C (NFC). También son compatibles las formas normales NFC, NFD, NFKC y NFKD.

6.7.27.9 replace

Esta función toma como argumentos una cadena de entrada, una expresión regular y una cadena de reemplazo y reemplaza todas las coincidencias de la expresión regular en la cadena de entrada con la cadena de reemplazo. Si la expresión regular coincide con dos cadenas que se superponen en la cadena de entrada, solo se reemplaza la primera.



Lenguajes

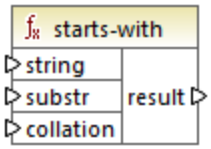
XQuery, XSLT 2.0.

Parámetros

Nombre	Tipo	Descripción
input	`xs:string`	La cadena de entrada.
pattern	`xs:string`	La expresión regular, véase Expresiones regulares 627.
replacement	`xs:string`	La cadena de reemplazo.
flags	`xs:string`	Argumento opcional que influye en las coincidencias. Se usa igual que el argumento **flags** de la función `matches` 812.

6.7.27.10 starts-with

Devuelve **true** si **string** empieza por **substr**; de lo contrario devuelve **false**. El valor devuelto es de tipo `xs:boolean`. La comparación de cadenas tiene lugar conforme al elemento collation que se indique.



Lenguajes

XQuery, XSLT 2.0.

Parámetros

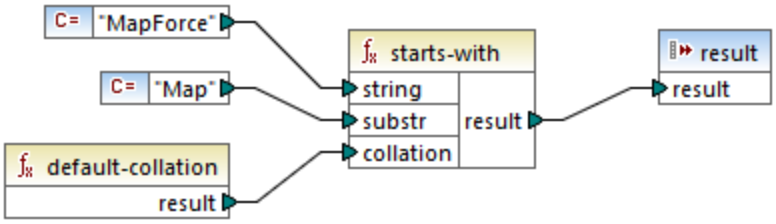
Nombre	Tipo	Descripción
string	`xs:string`	La cadena de entrada (el "pajar").
substr	`xs:string`	La subcadena (la "aguja").
collation	`xs:string`	Indica qué cotejo usar para la comparación de cadenas. Puede provenir de la entrada de la función `default-collation` 781 o puede ser un cotejo como `http://www.w3.org/2005/xpath`

© 2016-2022 Altova GmbH

Nombre	Tipo	Descripción
		`-functions/collation/html-ascii-case-insensitive`.

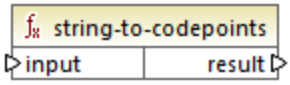
Ejemplo

The following mapping returns the value **`true`**, because the input string "MapForce" begins with the substring "Map", assuming that the default Unicode collation is used.



6.7.27.11 string-to-codepoints

Devuelve la secuencia de de puntos de código Unicode (valores enteros) que constituye la cadena dada como argumento. Esta función es opuesta a la función **`codepoints-to-string`** 807.



Lenguajes

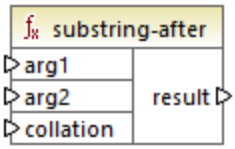
XQuery, XSLT 2.0.

Parámetros

Nombre	Tipo	Descripción
input	`xs:string`	La cadena de entrada.

6.7.27.12 substring-after

Devuelve la parte de **arg1** que aparece después de la cadena **arg2**.



Lenguajes

XQuery, XSLT 2.0.

Parámetros

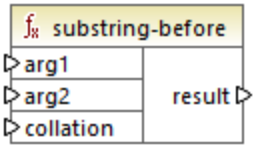
Nombre	Tipo	Descripción
arg1	`xs:string`	haystack
arg2	`xs:string`	haystack
collation	`xs:string`	Indica qué cotejo usar para la comparación de cadenas. Puede provenir de la entrada de la función `default-collation` 781 o puede ser un cotejo como `http://www.w3.org/2005/xpath-functions/collation/html-ascii-case-insensitive`.

Ejemplo

Si **arg1** es "MapForce", **arg2** es "Map" y **collation** es `default-collation` 781, la función devuelve "Force".

6.7.27.13 substring-before

Devuelve la parte de **arg1** que aparece antes de la cadena **arg2**.



© 2016-2022 Altova GmbH

Lenguajes

XQuery, XSLT 2.0.

Parámetros

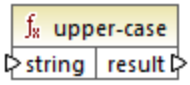
Nombre	Tipo	Descripción
arg1	`xs:string`	La cadena de entrada (el "pajar").
arg2	`xs:string`	La subcadena (la "aguja").
collation	`xs:string`	Indica qué cotejo usar para la comparación de cadenas. Puede provenir de la entrada de la función `default-collation` 781 o puede ser un cotejo como `http://www.w3.org/2005/xpath-functions/collation/html-ascii-case-insensitive`.

Ejemplo

Si **arg1** es "MapForce", **arg2** es "Force" y **collation** es `default-collation` 781, la función devuelve "Map".

6.7.27.14 upper-case

Devuelve el valor de **string** después de pasar cada uno de sus caracteres a su versión en mayúsculas.



Lenguajes

XQuery, XSLT 2.0.

Parámetros

Nombre	Tipo	Descripción
string	`xs:string`	La cadena de entrada.

6.7.28 xpath3 | external information functions

Las funciones de información externas de la biblioteca **xpath3** permiten obtener información sobre el entorno de ejecución XSLT o recuperar datos de recursos externos.

6.7.28.1 available-environment-variables

Devuelve una lista de nombres de variables de entorno que se pueden pasar a la función **environment-variable** como una secuencia (puede que vacía) de cadenas.

Lenguajes

XSLT 3.0.

6.7.28.2 environment-variable

Devuelve el valor de una variable de entorno del sistema, si existe. El tipo de retorno es `xs: string`.

Lenguajes

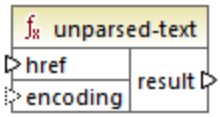
XSLT 3.0.

Parámetros

Nombre	Tipo	Descripción
name	`xs:string`	El nombre de la variable de entorno.

6.7.28.3 unparsed-text

Lee un recurso externo (por ejemplo un archivo) y devuelve una representación en cadena de ese recurso.



Lenguajes

XSLT 3.0.

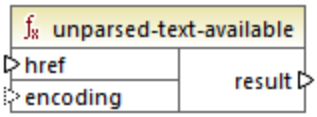
© 2016-2022 Altova GmbH

Parámetros

Nombre	Tipo	Descripción
href	`xs:string`	Una cadena en forma de referencia URI.
encoding	`xs:string`	Argumento opcional. Indica el nombre del tipo de cifrado, por ejemplo "UTF-8", "UTF-16". Si el cifrado no se puede determinar automáticamente se asume que es UTF-8.

6.7.28.4 unparsed-text-available

Determina si una llamada a **`unparsed-text`** con argumentos particulares se realiza correctamente. El tipo de retorno es `xs:boolean`.



Lenguajes

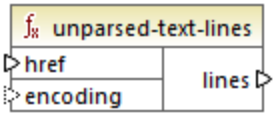
XSLT 3.0.

Parámetros

Nombre	Tipo	Descripción
href	`xs:string`	Una cadena en forma de referencia URI.
encoding	`xs:string`	Argumento opcional. Indica el nombre del tipo de cifrado, por ejemplo "UTF-8", "UTF-16". Si el cifrado no se puede determinar automáticamente se asume que es UTF-8.

6.7.28.5 unparsed-text-lines

Lee un recurso externo (por ejemplo un archivo) y devuelve su contenido como una secuencia de cadenas, una por cada línea de texto en la representación de las cadenas del recurso.



Lenguajes

XSLT 3.0.

Parámetros

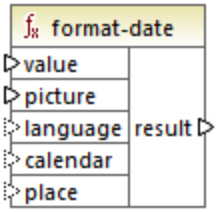
Nombre	Tipo	Descripción
href	`xs:string`	Una cadena en forma de referencia URI.
encoding	`xs:string`	Argumento opcional. Indica el nombre del tipo de cifrado, por ejemplo "UTF-8", "UTF-16". Si el cifrado no se puede determinar automáticamente se asume que es UTF-8.

6.7.29 xpath3 | formatting functions

Las funciones de formato de la biblioteca **xpath3** se usan para dar formato a los valores date, time e integer.

6.7.29.1 format-date

Devuelve una cadena que contiene un valor `xs:date` con formato para su visualización.



Lenguajes

XSLT 3.0.

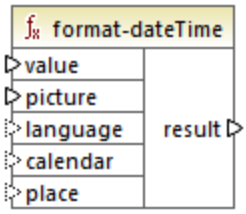
© 2016-2022 Altova GmbH

Parámetros

Nombre	Tipo	Descripción
value	`xs:date`	El valor de entrada `xs:date` al que se debe aplicar formato. Parámetro obligatorio.
picture	`xs:string`	Parámetro obligatorio. Consulte el apartado 9.8.4.1 de la recomendación "XPath and XQuery Functions and Operators 3.1" W3C Recommendation (https://www.w3.org/TR/xpath-functions-31) para obtener más información.
language	`xs:string`	Parámetro opcional. Consulte el apartado 9.8.4.8 de la recomendación "XPath and XQuery Functions and Operators 3.1" W3C Recommendation (https://www.w3.org/TR/xpath-functions-31) para obtener más información.
calendar	`xs:string`	Véase el punto anterior.
place	`xs:string`	Véase el punto anterior.

6.7.29.2 format-dateTime

Devuelve una cadena que contiene un valor `xs: dateTime` con formato para su visualización.



Lenguajes

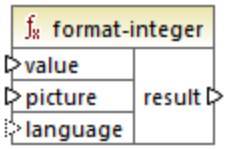
XSLT 3.0.

Parámetros

Nombre	Tipo	Descripción
value	`xs:dateTime`	El valor de entrada `xs:dateTime` al que se debe aplicar formato.
picture	`xs:string`	Parámetro obligatorio. Consulte el apartado 9.8.4.1 de la recomendación "XPath and XQuery Functions and Operators 3.1" W3C Recommendation (https://www.w3.org/TR/xpath-functions-31) para obtener más información.
language	`xs:string`	Parámetro opcional. Consulte el apartado 9.8.4.8 de la recomendación "XPath and XQuery Functions and Operators 3.1" W3C Recommendation (https://www.w3.org/TR/xpath-functions-31) para obtener más información.
calendar	`xs:string`	Véase el punto anterior.
place	`xs:string`	Véase el punto anterior.

6.7.29.3 format-integer

Da formato a un número entero conforme a una cadena de imagen dada y usando las convenciones del lenguaje natural indicado.



Lenguajes

XSLT 3.0.

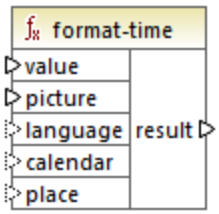
© 2016-2022 Altova GmbH

Parámetros

Nombre	Tipo	Descripción
value	`xs:integer`	El valor del número entero de entrada al que se debe aplicar formato.
picture	`xs:string`	Parámetro obligatorio. Consulte el apartado 4.6.1 de la recomendación "XPath and XQuery Functions and Operators 3.1" W3C Recommendation (https://www.w3.org/TR/xpath-functions-31) para obtener más información.
language	`xs:string`	Parámetro opcional. Indica el lenguaje natural según el cual se aplica formato al valor. Si se indica, este valor debe ser una cadena vacía o cualquier valor permitido para el atributo `xml:lang` de acuerdo con la recomendación "Extensible Markup Language (XML) 1.0 W3C Recommendation (https://www.w3.org/TR/xml) para obtener más información.

6.7.29.4 format-time

Devuelve una cadena que contiene un valor `xs: time` con formato para su visualización.



Lenguajes

XSLT 3.0.

Parámetros

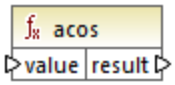
Nombre	Tipo	Descripción
value	`xs:time`	El valor de entrada `xs: time` al que se debe aplicar formato.
picture	`xs:string`	Parámetro obligatorio. Consulte el apartado 9.8.4.1 de la recomendación "XPath and XQuery Functions and Operators 3.1" W3C Recommendation (https://www.w3.org/TR/xpath-functions-31) para obtener más información.
language	`xs:string`	Parámetro opcional. Consulte el apartado 9.8.4.8 de la recomendación "XPath and XQuery Functions and Operators 3.1" W3C Recommendation (https://www.w3.org/TR/xpath-functions-31) para obtener más información.
calendar	`xs:string`	Véase el punto anterior.
place	`xs:string`	Véase el punto anterior.

6.7.30 xpath3 | math functions

Las funciones matemáticas de la biblioteca **xpath3** se usan para realizar cálculos trigonométricos, entre otros.

6.7.30.1 acos

El resultado es el arco coseno de un ángulo en el rango de **0** a **pi**.



Lenguajes

XSLT 3.0.

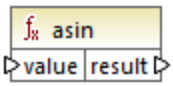
© 2016-2022 Altova GmbH

Parámetros

Nombre	Tipo	Descripción
value	`xs:double`	El valor de entrada.

6.7.30.2 asin

El resultado es el arcoseno de un ángulo en el rango de **-pi/2** a **pi/2**.



Lenguajes

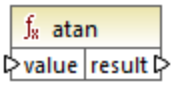
XSLT 3.0.

Parámetros

Nombre	Tipo	Descripción
value	`xs:double`	El valor de entrada.

6.7.30.3 atan

El resultado es el arco tangente de un ángulo en el rango de **-pi/2** a **pi/2**.



Lenguajes

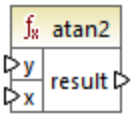
XSLT 3.0.

Parámetros

Nombre	Tipo	Descripción
value	`xs:double`	El valor de entrada.

6.7.30.4 atan2

Devuelve el ángulo en radianes subtendido en el origen por el punto de un plano con coordenadas (x, y) y el eje x positivo.



Lenguajes

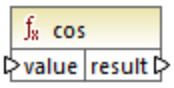
XSLT 3.0.

Parámetros

Nombre	Tipo	Descripción
y	`xs:double`	La coordenada x.
x	`xs:double`	La coordenada y.

6.7.30.5 cos

El resultado es el coseno del ángulo del parámetro de entrada. La unidad del valor es el radián.



Lenguajes

XSLT 3.0.

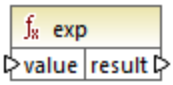
Parámetros

Nombre	Tipo	Descripción
value	`xs:double`	El valor de entrada.

© 2016-2022 Altova GmbH

6.7.30.6 exp

El resultado es *e* (el logaritmo natural base) elevado a la potencia de la entrada value.



Lenguajes

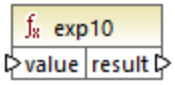
XSLT 3.0.

Parámetros

Nombre	Tipo	Descripción
value	`xs:double`	El valor de entrada.

6.7.30.7 exp10

El resultado es 10 elevado a la potencia de la entrada value.



Lenguajes

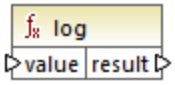
XSLT 3.0.

Parámetros

Nombre	Tipo	Descripción
value	`xs:double`	El valor de entrada.

6.7.30.8 log

El resultado es el logaritmo natural (en base e) de un valor.



Lenguajes

XSLT 3.0.

Parámetros

Nombre	Tipo	Descripción
value	`xs:double`	El valor de entrada.

6.7.30.9 log10

El resultado es el logaritmo decimal (en base 10) de un valor.

Lenguajes

XSLT 3.0.

Parámetros

Nombre	Tipo	Descripción
value	`xs:double`	El valor de entrada.

6.7.30.10 pi

Devuelve una aproximación a la constante matemática **pi**.

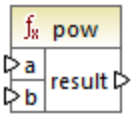
Lenguajes

XSLT 3.0.

© 2016-2022 Altova GmbH

6.7.30.11 pow

El resultado es el valor de **a** elevado a la potencia **b**.



Lenguajes

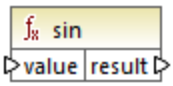
XSLT 3.0.

Parámetros

Nombre	Tipo	Descripción
a	`xs:double`	El valor de entrada **a**.
b	`xs:double`	El valor de entrada **b**.

6.7.30.12 sin

El resultado es el seno del ángulo del parámetro de entrada. La unidad del valor es el radián.



Lenguajes

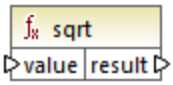
XSLT 3.0.

Parámetros

Nombre	Tipo	Descripción
value	`xs:double`	El valor de entrada.

6.7.30.13 sqrt

Devuelve la raíz cuadrada no negativa del argumento.



Lenguajes

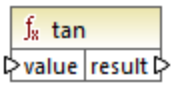
XSLT 3.0.

Parámetros

Nombre	Tipo	Descripción
value	`xs:double`	El valor de entrada.

6.7.30.14 tan

El resultado es la tangente del ángulo del parámetro de entrada. La unidad del valor es el radián.



Lenguajes

XSLT 3.0.

Parámetros

Nombre	Tipo	Descripción
value	`xs:double`	El valor de entrada.

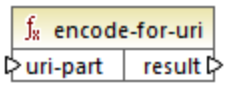
6.7.31 xpath3 | URI functions

Las funciones de URI de la biblioteca **xpath3** cifran, convierten y aplican secuencias de escape a los valores que se quieren usar en URIs.

© 2016-2022 Altova GmbH

6.7.31.1 encode-for-uri

Cifra caracteres reservados en una cadena que se pretende usar en el segmento de la ruta de un URI. Para más información sobre esta función consulte el apartado 6.2 de la recomendación "XPath and XQuery Functions and Operators 3.1" W3C Recommendation (https://www.w3.org/TR/xpath-functions-31) para obtener más información.



Lenguajes

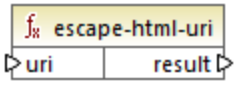
XSLT 3.0.

Parámetros

Nombre	Tipo	Descripción
uri-part	`xs:string`	El valor del URI de entrada al que se debe aplicar el cifrado.

6.7.31.2 escape-html-uri

Aplica secuencias de escape a un URI de igual forma que los agentes de usuario HTML administran los valores de atributo que se espera que contengan los URIs. Para más información sobre esta función consulte el apartado 6.4 de la recomendación "XPath and XQuery Functions and Operators 3.1" W3C Recommendation (https://www.w3.org/TR/xpath-functions-31) para obtener más información.



Lenguajes

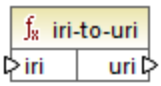
XSLT 3.0.

Parámetros

Nombre	Tipo	Descripción
uri	`xs:string`	El valor del URI de entrada al que se deben aplicar las secuencias de escape.

6.7.31.3 iri-to-uri

Convierte una cadena que contiene un IRI (Identificador de recursos internacionalizado) en un URI (Identificador de recursos uniforme). Para más información sobre esta función consulte el apartado 6.3 de la recomendación "XPath and XQuery Functions and Operators 3.1" W3C Recommendation (https://www.w3.org/TR/xpath-functions-31) para obtener más información.



Lenguajes

XSLT 3.0.

Parámetros

Nombre	Tipo	Descripción
iri	`xs:string`	El valor de IRI de entrada.

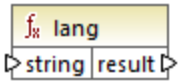
6.7.32 xslt | xpath functions

Las funciones de la biblioteca xslt | xpath son funciones de conjunto de nodos XPath 1.0. Estas funciones toman un nodo o conjunto de nodos como contexto y devuelve información sobre ellos. Las funciones `last` y `position` operan en el contexto de asignación actual 874, que determinan las conexiones de la asignación.

Nota: en la biblioteca de funciones **core** puede encontrar más funciones XPath 1.0.

6.7.32.1 lang

Devuelve **true** si el nodo de contexto tiene un atributo `xml:lang` con un valor que o coincide totalmente con el argumento **string** o es un subconjunto de esta. De lo contrario, la función devuelve **false**.



Lenguajes

XSLT 1.0.

© 2016-2022 Altova GmbH

Parámetros

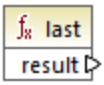
Nombre	Tipo	Descripción
string	`xs:string`	El código del idioma que se comprueba, por ejemplo "en".

Ejemplo

Consulte el ejemplo de la función `lang` 799 de la biblioteca **xpath2**.

6.7.32.2 last

Devuelve el número de la posición del último nodo en la lista de nodos que se procesa.



Lenguajes

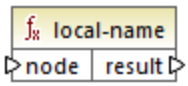
XSLT 1.0.

Ejemplo

Consulte el ejemplo de la función `last` 782 de la biblioteca **xpath2**.

6.7.32.3 local-name

Devuelve la parte local del nombre del nodo indicado como argumento.



Lenguajes

XSLT 1.0, XSLT 2.0.

Parámetros

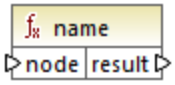
Nombre	Tipo	Descripción
node	`node()`	El nodo de entrada.

Ejemplo

Consulte el ejemplo de la función `local-name` 801 de la biblioteca **xpath2**.

6.7.32.4 name

Devuelve el nombre del nodo dado como argumento.



Lenguajes

XSLT 1.0, XSLT 2.0.

Parámetros

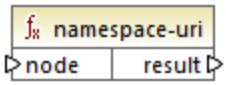
Nombre	Tipo	Descripción
node	`node()`	El nodo de entrada.

Ejemplo

Consulte el ejemplo de la función `local-name` 801 de la biblioteca **xpath2**

6.7.32.5 namespace-uri

Devuelve el URI de espacio de nombres del nodo dado como argumento.



Lenguajes

XSLT 1.0, XSLT 2.0.

Parámetros

Nombre	Tipo	Descripción
node	`node()`	El nodo de entrada.

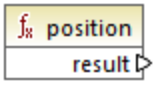
© 2016-2022 Altova GmbH

Ejemplo

Consulte el ejemplo de la función `local-name` 801 de la biblioteca **xpath2**

6.7.32.6 position

Devuelve la posición del nodo actual en el conjunto de nodos que se está procesando.



Lenguajes

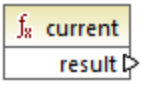
XSLT 1.0.

6.7.33 xslt | xslt functions

Las funciones de esta biblioteca con funciones XSLT 1.0.

6.7.33.1 current

Esta función no toma argumentos y devuelve el nodo actual.

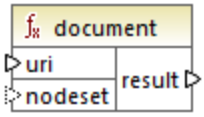


Lenguajes

XSLT 1.0.

6.7.33.2 document

Accede a los nodos desde un documento externo. El resultado de la función se envía a un nodo del documento de destino.



Lenguajes

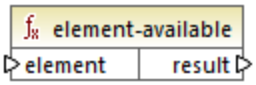
XSLT 1.0.

Parámetros

Nombre	Tipo	Descripción
uri	`xs:string`	Obligatorio. Indica la ruta al documento de destino. El documento XML debe ser válido y poder analizarse.
nodeset	`nodo()`	Opcional. Indica un nodo, el URI de base que se usa para resolver el URI indicado como primer argumento si es relativo.

6.7.33.3 element-available

Esta función comprueba si un elemento es compatible con el procesador XSLT (el elemento se da como único argumento de cadena de la función). La cadena del argumento se evalúa como QName Por tanto, los elementos XSLT deben tener un prefijo `xsl:` y los elementos XML Schema deben tener un prefijo `xs:` porque estos los prefijos declarados para estos espacios de nombres en el XSLT que se genera para la asignación. La función devuelve un valor booleano.



Lenguajes

XSLT 1.0.

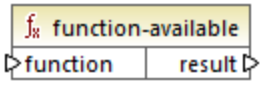
Parámetros

Nombre	Tipo	Descripción
element	`xs:string`	El nombre del elemento.

6.7.33.4 function-available

Esta función es similar a la función **`element-available`** y comprueba si el nombre de la función dado como argumento es compatible con el procesador XSLT. La cadena de entrada se evalúa como QName. La función devuelve un valor booleano.

© 2016-2022 Altova GmbH



Lenguajes

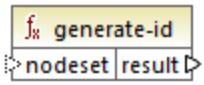
XSLT 1.0.

Parámetros

Nombre	Tipo	Descripción
Función	`xs:string`	El nombre de la función.

6.7.33.5 generate-id

Esta función genera una cadena única que identifica el primer nodo del conjunto de nodos identificado por el argumento de entrada opcional. Si no tiene ningún argumento, la función genera el identificador en el nodo de contexto. El resultado se puede dirigir después a cualquier nodo del documento de salida.



Lenguajes

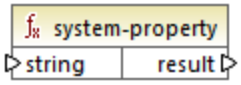
XSLT 1.0, XSLT 2.0, XSLT 3.0.

Parámetros

Nombre	Tipo	Descripción
nodeset	`nodo()`	Argumento opcional que indica el nodo de entrada.

6.7.33.6 system-property

Esta función devuelve las propiedades del procesador XSLT (el sistema). Para los procesadores XSLT hay tres propiedades de sistema obligatorias, todas situadas en el espacio de nombres XSLT. Se trata de `xsl:version`, `xsl:vendor` y `xsl:vendor-url`. La cadena de entrada se evalúa como QName y debe tener el prefijo `xsl:` porque este es el prefijo asociado con el espacio de nombres XSLT de la hoja de estilos XSLT subyacente.



Lenguajes

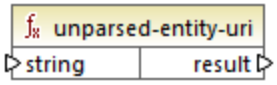
XSLT 1.0, XSLT 2.0, XSLT 3.0.

Parámetros

Nombre	Tipo	Descripción
string	`xs:string`	Indica el nombre de la propiedad, que puede ser cualquiera de estos: `xsl:version`, `xsl:vendor`, `xsl:vendor-url`.

6.7.33.7 unparsed-entity-uri

Si usa una DTD puede declarar en ella una entidad sin analizar. Esta entidad sin analizar (por ejemplo, una imagen) tendrá un URI que ubica la entidad sin analizar. La cadena de entrada de la función debe coincidir con el nombre de la entidad sin analizar que se declaró en la DTD. La función devuelve el URI de la entidad sin analizar, que puede dirigirse a cualquier nodo del documento de destino (al nodo **href** por ejemplo).



Lenguajes

XSLT 1.0.

Parámetros

Nombre	Tipo	Descripción
string	`xs:string`	El nombre de la entidad sin analizar cuyo URI se quiere obtener.

© 2016-2022 Altova GmbH

7 Asignaciones avanzadas

Sitio web de Altova: Herramienta de integración de datos

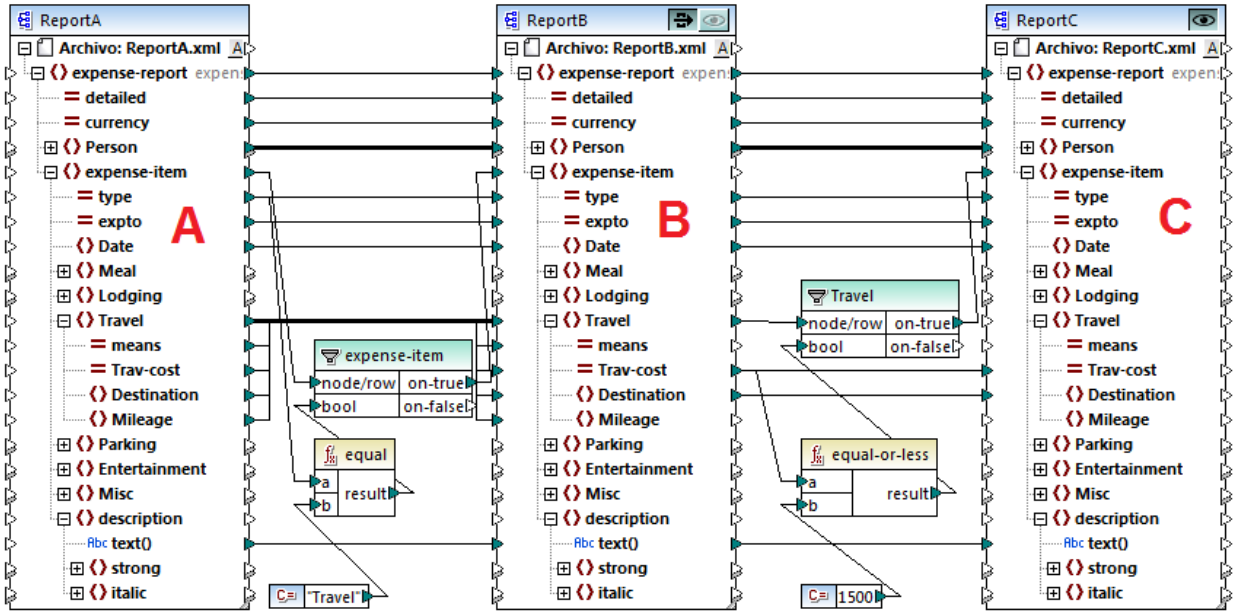
En esta sección explicamos los distintos tipos de asignaciones avanzadas y se incluyen los apartados siguientes:

- Asignaciones encadenadas 840
- Asignar nombres de nodos 850
- Reglas y estrategias de asignación 871
- Procesar varios archivos de entrada o salida simultáneamente 892
- Redistribuir y serializar cadenas de texto 903
- Paneles de resultados de StyleVision 911
- Generar la documentación de la asignación 915

7.1 Asignaciones en cadena

En MapForce puede crear asignaciones de datos formadas por varios componentes en cadena. Las asignaciones en cadena son asignaciones donde como mínimo un componente hace de origen y de destino a la vez. Este tipo de componentes crea un resultado que se usa después como entrada del siguiente paso de la asignación en cadena. A estos componentes se les denomina *componente intermedio*.

Por ejemplo, la asignación que aparece a continuación muestra un informe de gastos (en formato XML) que se procesa en dos etapas. La parte de la asignación que va de A a B filtra los gastos marcados como "Travel" (viaje). La parte que va de B a C filtra los gastos de viaje por un importe inferior a 1500. El componente B es el componente intermedio porque tiene conexiones de entrada y salida. Puede encontrar esta asignación en esta ruta de acceso:
<Documentos>\Altova\MapForce2023\MapForceExamples\Tutorial\ChainedReports.mfd.



ChainedReports.mfd

Las asignaciones en cadena presentan una característica llamada *paso a través.* Se trata de una función de vista previa que permite consultar los resultados que produce cada etapa de la asignación en cadena (en el panel *Resultados*). Por ejemplo, en la asignación del ejemplo puede consultar y guardar la vista previa del resultado XML que produce la etapa A - B y del resultado XML que produce la etapa B - C.

Nota: la característica *paso a través* solamente está disponible para componentes basados en archivos (como componentes XML, CSV y de texto). Los componentes de base de datos pueden ser componentes intermedios pero no disponen del botón **Paso a través**. El componente intermedio siempre se regenera desde cero cuando se crea la vista previa o se genera código. Esto no sería posible con bases de datos porque antes de regenerar la base de datos sería necesario eliminarla.

 © 2016-2022 Altova GmbH

Si la asignación se ejecuta con MapForce Server o con código generado, entonces se ejecuta toda la cadena de asignación. La asignación genera los archivos de salida necesarios en cada etapa de la cadena y el resultado de una etapa se reenvía como entrada de la siguiente etapa.

También se pueden generar nombres de archivo dinámicos a partir de los componentes intermedios. Es decir, se pueden crear conexiones con el elemento `Archivo:` de los componentes intermedios, siempre y cuando éstos estén configurados correctamente (*véase* *Procesar varios archivos de entrada o salida simultáneamente* 892).

Botón Vista previa

Tanto el componente B como el componente C tienen un botón de vista previa. Este botón genera en MapForce una vista previa del resultado intermedio de B, así como el resultado final de la asignación en cadena. Haga clic en el botón **Vista previa** del componente correspondiente y después haga clic en el panel *Resultados* para ver el resultado de la asignación.

Los componentes intermedios que tengan activo el botón **Paso a través** no pueden verse en la vista previa porque su botón **Vista previa** se deshabilita automáticamente (ya que no tiene sentido obtener una vista previa de los datos y pasarlos al mismo tiempo). Para ver el resultado de los componentes intermedios, primero es necesario desactivar el botón **Paso a través** y después hacer clic en el botón **Vista previa**.

Botón Paso a través

El componente intermedio B tiene un botón más en la barra de título que se llama **Paso a través**.

Si el botón **Paso a través** está **activo** , MapForce asigna dirige todos los datos a la ventana de vista previa de una vez, del componente A al B y después al C. Se crearán dos archivos de resultados:

- el resultado de asignar el componente A al componente intermedio B.
- el resultado de asignar el componente intermedio B al componente de destino C.

Si el botón **Paso a través** está **inactivo** , MapForce solamente ejecutará partes de la cadena de asignación. Los datos que se generan dependen de qué botones **Vista previa** están activos:

- Si está activo el botón **Vista previa** del componente B, se genera el resultado de asignar el componente A al componente B. La cadena de asignación se detiene en el componente B. El componente C no se utiliza para nada en la vista previa.
- Si está activo el botón **Vista previa** del componente C, se genera el resultado de asignar el componente intermedio B al componente C. Como el paso a través está inactivo, se interrumpe el encadenamiento automático para el componente B. Solamente se ejecuta la parte derecha de la cadena de asignación. Es decir, el componente A no se utiliza.

> Cuando el botón Paso a través está inactivo, es importante que en los campos *Archivo XML de entrada* y *Archivo XML de salida* del componente intermedio se especifiquen nombres de archivo idénticos. Esto garantiza que el archivo de salida que se genera durante la vista previa de la etapa A - B sirva de entrada para la vista previa de la etapa B - C. Además, en el código generado o en la ejecución con MapForce Server, esto garantiza que la cadena de asignación no se rompa.

Como ya dijimos anteriormente, si la asignación se ejecuta con MapForce Server o con el código generado, se da salida a todos los componentes. Cuando así sea, el estado del botón **Paso a través** del componente B no

se tiene en cuenta (tampoco se tiene en cuenta qué componente de visa previa está seleccionado). Por ejemplo, a partir de la asignación de la imagen se generarían dos archivos de resultados:

1. el archivo de salida que se obtiene de la asignación de A a B y
2. el archivo de salida que se obtiene de la asignación de B a C.

Los apartados Ejemplo: paso a través activo 842 y Ejemplo: paso a través inactivo 846 explican con más detalle cómo se transfieren los datos de origen cuando el botón **Paso a través** está activo e inactivo, respectivamente.

7.1.1 Ejemplo: paso a través activo

La asignación utilizada en este ejemplo se llama **ChainedReports.mfd** y está en la carpeta **<Documentos>\Altova\MapForce2023\MapForceExamples\Tutorial**. Esta asignación procesa un archivo XML llamado **ReportA.xml** que contiene gastos de viaje (ver más abajo). Para mayor claridad se omitieron la declaración de espacio de nombres y algunos elementos `expense-item`:

```
<?xml version="1.0" encoding="UTF-8"?>
<expense-report currency="USD" detailed="true">
   <Person>
      <First>Fred</First>
      <Last>Landis</Last>
      <Title>Project Manager</Title>
      <Phone>123-456-78</Phone>
      <Email>f.landis@nanonull.com</Email>
   </Person>
   <expense-item type="Travel" expto="Development">
      <Date>2003-01-02</Date>
      <Travel Trav-cost="337.88">
         <Destination/>
      </Travel>
      <description>Biz jet</description>
   </expense-item>
   <expense-item type="Lodging" expto="Sales">
      <Date>2003-01-01</Date>
      <Lodging Lodge-cost="121.2">
         <Location/>
      </Lodging>
      <description>Motel mania</description>
   </expense-item>
   <expense-item type="Travel" expto="Marketing">
      <Date>2003-02-02</Date>
      <Travel Trav-cost="2000">
         <Destination/>
      </Travel>
      <description>Hong Kong</description>
   </expense-item>
</expense-report>
```

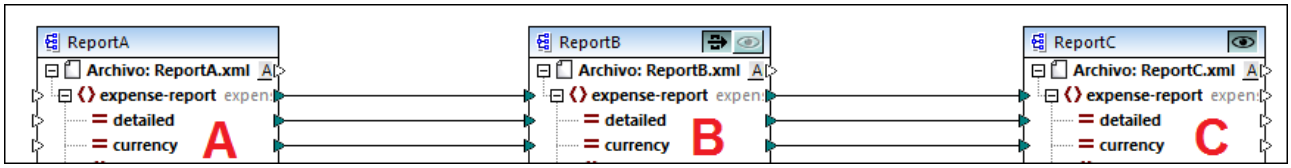
Archivo ReportA.xml

 © 2016-2022 Altova GmbH

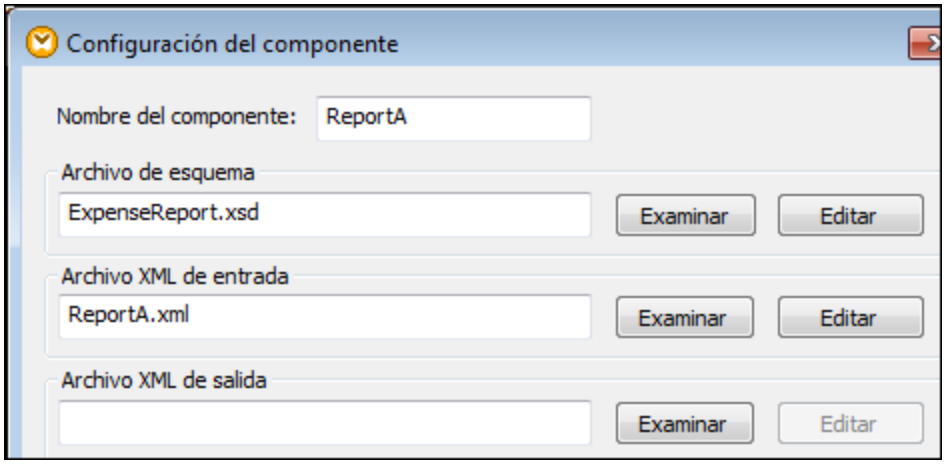
El objetivo de la asignación es producir dos informes a partir del archivo XML:

- El informe **ReportB.xml**, que debe contener los gastos de viaje que correspondan al tipo "Travel".
- El informe **ReportC.xml**, que debe contener los gastos de viaje que correspondan al tipo "Travel" y que no superen un valor de 1500.

Para conseguirlo el componente intermedio de la asignación (el componente B) tiene activado el botón **Paso a través** (*ver más abajo*). Esto hace que la asignación se ejecute en dos etapas: de A a B y de B a C. El resultado que genera el componente intermedio se usará como entrada para la asignación entre B y C.

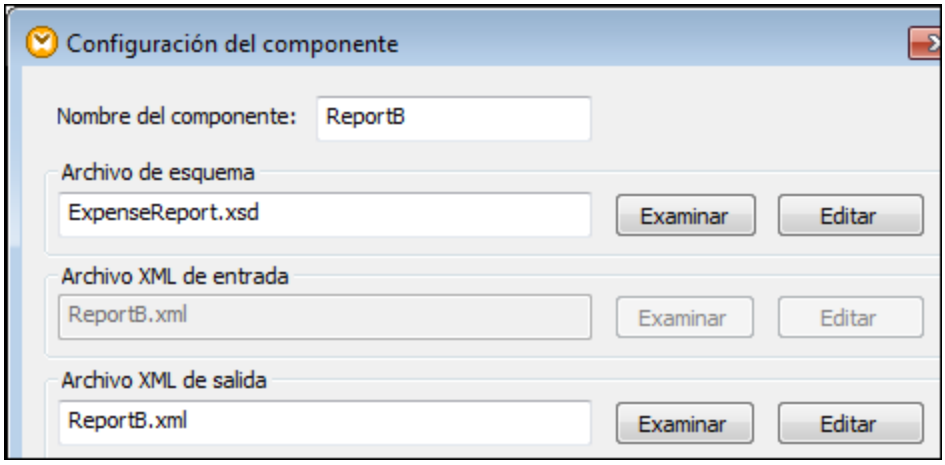


El nombre de los archivos de salida que se generan en cada etapa de la cadena de asignación viene dado por la configuración de cada componente. Para abrir la configuración de un componente haga clic con el botón derecho en el componente y seleccione **Propiedades** en el menú contextual. En nuestro ejemplo el primer componente está configurado para leer datos de un archivo XML llamado **ReportA.xml**. Como este es el componente de origen, el archivo de salida XML no es relevante y este campo se dejó vacío en la configuración del componente.



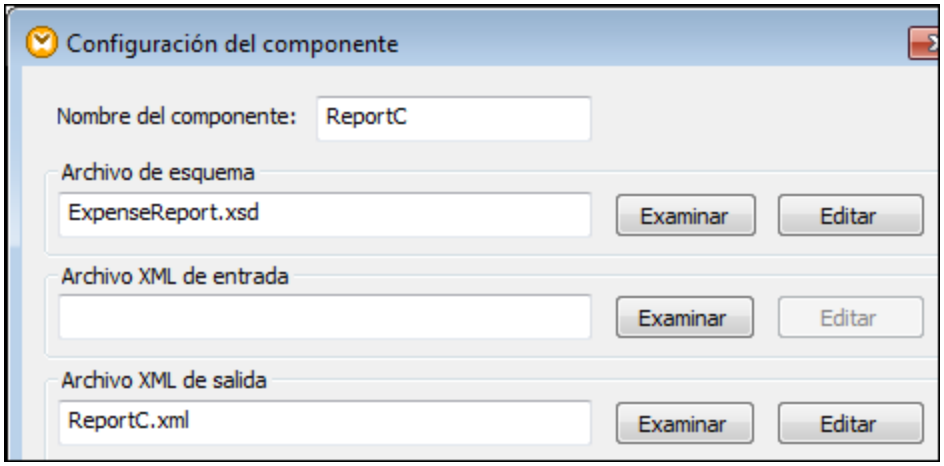
Configuración del componente de origen

Como puede verse en la siguiente imagen, el segundo componente (**ReportB**) está configurado para generar un archivo de salida llamado **ReportB.xml**. Observe que el campo *Archivo XML de entrada* está atenuado. Esto se debe a que cuando el paso a través está activo (como en nuestro ejemplo) el campo *Archivo XML de entrada* del componente intermedio se desactiva automáticamente. Para poder ejecutar la asignación no se necesita un nombre de archivo de entrada porque el resultado que se crea en esta etapa de la cadena se almacena en un archivo temporal y se vuelve a utilizar en las etapas siguientes. Además, si se define un archivo XML de salida (*imagen siguiente*), éste se usará para el nombre de archivo del archivo de salida intermedio. Si no se define el valor del campo *Archivo XML de salida*, se usará automáticamente un nombre de archivo predeterminado.



Configuración del componente intermedio

Por último, el tercer componente está configurado para generar un archivo de salida llamado **ReportC.xml**. El valor del campo *Archivo XML de entrada* no es relevante porque se trata del componente de destino.



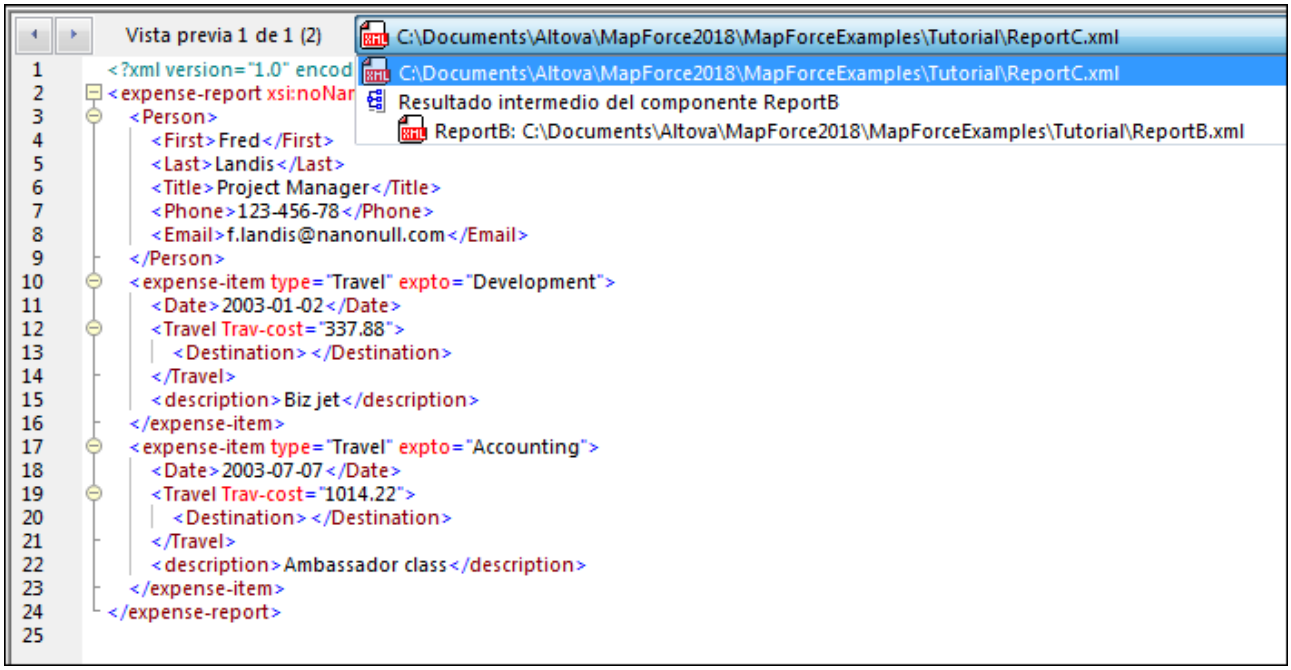
Configuración del componente de destino

Si consultamos la vista previa abriendo el panel *Resultados*, observaremos que se generaron dos archivos de salida:

1. **ReportB.xml**, que representa el resultado de la asignación de A a B.
2. **ReportC.xml**, que representa el resultado de la asignación de B a C.

Para seleccionar cuál de los dos archivos de salida se presentan en el panel *Resultados* haga clic en los botones de flecha o seleccione una entrada en la lista desplegable.

 © 2016-2022 Altova GmbH



Archivos de salida generados

Cuando MapForce ejecuta la asignación, la configuración de la opción *Escribir directamente en archivos de salida finales* (que se configura desde **Herramientas | Opciones | Generales**) determina si los archivos intermedios se deben guardar como archivos temporales o físicos. Recuerde que esto solo es válido cuando se consulta una vista previa en MapForce. Si la asignación se ejecutara con MapForce o por medio de código, se producirían archivos en cada etapa de la cadena de asignación.

Si tiene StyleVision instalado y asignó un archivo SPS (StyleVision Power Stylesheet) al componente de destino (como en nuestro ejemplo), entonces podrá ver (y guardar) el resultado final de la asignación como HTML, RTF. Para generar y consultar este resultado en MapForce basta con abrir el panel correspondiente.

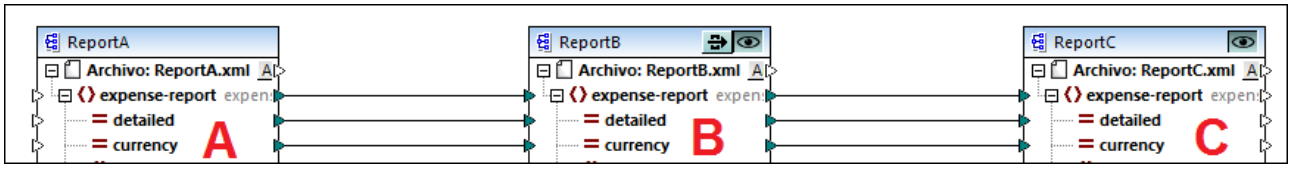
Resultado HTML generado

Recuerde que solamente se presenta el resultado del componente de destino final de la cadena de asignación. Para ver el resultado de StyleVision de componentes intermedios deberá desactivar el botón de paso a través y generar una vista previa del componente intermedio (*véase* *Ejemplo: paso a través inactivo* 846).

7.1.2 Ejemplo: paso a través inactivo

La asignación utilizada en este ejemplo se llama **ChainedReports.mfd** y está en la carpeta **<Documentos>\Altova\MapForce2023\MapForceExamples\Tutorial**. Este ejemplo muestra el resultado que genera la asignación cuando el botón **Paso a través** del componente intermedio está desactivado.

 © 2016-2022 Altova GmbH

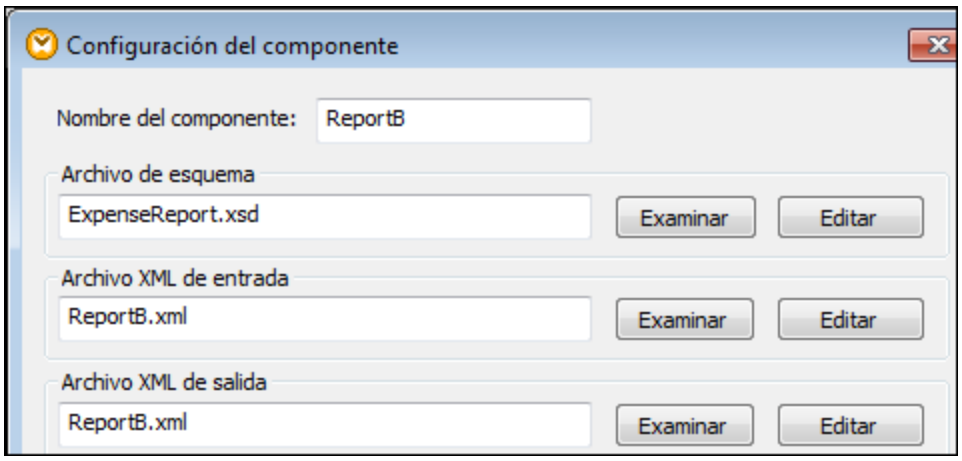


Como explicamos anteriormente en el apartado Ejemplo: paso a través activo 842, el objetivo de la asignación es producir dos informes distintos. En el ejemplo anterior, el botón **Paso a través** estaba activo y los informes generados podían verse en el panel *Resultados*. Sin embargo, si quiere consultar la vista previa de un solo informe (de **ReportB.xml** o de **ReportC.xml**), entonces debe desactivar el botón **Paso a través**. Desactivar el botón de paso a través puede ser de utilidad si su objetivo es:

- Consultar la vista previa del resultado que genera la asignación A - B solamente, ignorando la etapa B - C de la asignación.
- Consultar la vista previa del resultado que genera la asignación B - C solamente, ignorando la etapa A - B de la asignación.

Una vez desactivado el botón de paso a través, podrá elegir si consulta la vista previa de **ReportB** o de **ReportC** (observe que ambos tienen activo el botón **Vista previa**).

Además, desactivar el botón de paso a través permite elegir qué archivo de entrada debe leer el componente intermedio. En la mayoría de los casos, el archivo de entrada debería ser el mismo archivo que se definió en el campo *Archivo XML de salida* (como en este ejemplo).



Configuración del componente intermedio

El que el componente intermedio tenga el mismo archivo de entrada que de salida es importante si tiene pensado generar código a partir de la asignación o ejecutar la asignación con MapForce Server. Como se dijo anteriormente, en estos entornos se generan todos los resultados que crean los componentes de la cadena de asignación. Por tanto, lo más razonable es que el componente intermedio reciba un archivo (en este caso **ReportB.xml**), lo procese y lo reenvíe a la siguiente asignación, en lugar de buscar un nombre de archivo distinto. Tenga en cuenta que si en el componente intermedio el archivo de entrada y el de salida tienen nombres distintos (con el botón de paso a través inactivo), pueden producirse errores tipo *El sistema no encontró el archivo especificado* en el código generado o en la ejecución con MapForce Server.

Si hace clic en el botón **Vista previa** del tercer componente (**ReportC**) e intenta generar la vista previa de resultados en MapForce, se producirá un error de ejecución. Se trata de un error esperado porque, según la configuración, se espera como entrada un archivo llamado **ReportB.xml** pero la asignación todavía no ha producido dicho archivo (porque el botón de paso a través no está activo y solo se ejecuta la etapa B - C de la asignación). Para solucionar este problema:

1. Haga clic en el botón de vista previa del componente intermedio.
2. Haga clic en el panel *Resultados* para generar una vista previa de los resultados.
3. Guarde el archivo de salida resultante como **ReportB.xml** en la carpeta donde está la asignación (**<Documentos>\Altova\MapForce2023\MapForceExamples\Tutorial**).

Si ahora vuelve a hacer clic en el botón de vista previa del tercer componente (**ReportC**), el error ya no se produce.

Desactivar el botón de paso a través también permite consultar la vista previa de resultados generados con StyleVision creados a partir de todos los componentes que tengan un archivo SPS (StyleVision Power StyleSheet) asociado. También podrá ver la versión HTML del informe intermedio (además del informe final):

 © 2016-2022 Altova GmbH

Resultado HTML del componente intermedio

7.2 Asignar nombres de nodos

En la mayoría de las asignaciones de datos de MapForce el objetivo es leer *valores* de un componente de origen y escribir *valores* en un componente de destino. Sin embargo, en algunos casos el objetivo no es solo acceder a los valores de los nodos de origen, sino al nombre de los nodos. Por ejemplo, puede tratarse de una asignación que lea el nombre de los elementos o atributos (y no sus valores) de un archivo XML de origen y convertir esos nombres en valores de elemento o atributo en el XML de destino.

Imaginemos que tenemos un archivo XML que contiene una lista de productos. Cada producto tiene este formato:

```
<product>
   <id>1</id>
   <color>red</color>
   <size>10</size>
</product>
```

Nuestro objetivo es convertir información sobre cada producto en pares nombre/valor. Por ejemplo:

```
<product>
   <attribute name="id" value="1" />
   <attribute name="color" value="red" />
   <attribute name="size" value="10" />
</product>
```

En este caso, necesitaremos acceder al nombre de los nodos de forma *dinámica* para realizar conversiones de datos.

Nota: la transformación del ejemplo también puede conseguirse con las funciones **node-name** 676 y **static-node-name** 678 de la biblioteca **core**. No obstante, en nuestro ejemplo necesitamos conocer exactamente los nombres de elemento del componente de origen y necesitamos conectar cada uno de los elementos manualmente con el componente de destino. Además, puede que las funciones mencionadas no sean suficiente para filtrar o agrupar nodos por nombre o para manipular el tipo de datos de los nodos.

No solo puede acceder al nombre de los nodos de forma dinámica cuando necesite leer nombres de nodo, sino también cuando necesite escribirlos. En una asignación normal y corriente el nombre de los atributos o elementos del componente de destino siempre se conoce antes de que se ejecute la asignación. Estos nombres de atributo o elemento vienen del esquema subyacente del componente. Sin embargo, con los nombres de nodo dinámicos podrá crear atributos o elementos nuevos cuyo nombre no se conoce antes de que se ejecute la asignación. Concretamente, el nombre del atributo o elemento viene dado por la asignación propiamente dicha (por el componente de origen).

Para poder acceder a elementos o atributos secundarios de forma dinámica el nodo debe tener elementos o atributos secundarios y el nodo principal no puede ser el nodo XML raíz.

Puede trabajar con nombres de nodo dinámicos si en la asignación de datos participan componentes de este tipo:

 © 2016-2022 Altova GmbH

- XML
- CSV/FLF*

* Solo compatible con MapForce Professional o Enterprise Edition.

Nota: si se trata de componentes CSV/FLF el acceso dinámico implica poder acceder a los *campos* en lugar de a los *nodos* porque las estructuras CSV/FLF no tienen nodos.

Cuando el destino de la asignación es un archivo CSV o FLF, los campos deben estar definidos en la configuración del componente (y el nombre, el orden y el número de los campos de destino no se puede cambiar). Al contrario de los archivos XML, el formato de los archivos de texto es fijo así que solamente se puede manipular el valor de los campos (no se puede modificar el nombre, el número ni el orden de los campos).

Puede trabajar con nombres de nodo dinámicos si selecciona estos lenguajes de transformación: motor integrado BUILT-IN*, XSLT2, XQuery*, C#*, C++*, Java*.

* Estos lenguajes solo son compatibles con MapForce Professional o Enterprise Edition.

Para más información consulte el apartado Obtener acceso al nombre de los nodos 851 o el ejemplo del apartado Ejemplo: asignar nombres de elemento a valores de atributo 863.

7.2.1 Obtener acceso al nombre de los nodos

Cuando un nodo de un componente XML (o un campo de un componente CSV/FLF) tiene nodos secundarios, podemos obtener tanto el nombre como el valor de cada nodo secundario en la asignación directamente. Esta técnica se denomina *nombres de nodo dinámicos.* El adjetivo *dinámicos* hace referencia al hecho de que el procesamiento se realiza *sobre la marcha,* durante la ejecución de la asignación, y no se basa en la información estática del esquema que se conoce antes de que se ejecute la asignación. En este apartado explicamos cómo habilitar el acceso dinámico a nombres de nodo y cómo utilizarlo.

Cuando se leen datos de un componente de origen, los *nombres de nodo dinámicos* permiten:

- Obtener una lista de nodos o atributos secundarios de un nodo en forma de secuencia. En MapForce una *secuencia* es una lista de cero o más elementos que se pueden conectar con un componente de destino y que crea tantos elementos en el componente de destino como elementos existen en el componente de origen. Por ejemplo, si un nodo tiene cinco atributos en el componente de origen, puede crear cinco elementos nuevos en el componente de destino (cada uno de ellos corresponderá a un atributo).
- Leer no solo los valores de los nodos secundarios (como hacen las asignaciones normales y corrientes) sino también sus nombres.

Cuando se escriben datos en un componente de destino, los *nombres de nodo dinámicos* permiten:

- Crear nodos nuevos usando nombres que vienen dados por la asignación (nombres *dinámicos*), en lugar de los nombres que vienen dados por la configuración del componente (nombres *estáticos*).

A continuación veremos un ejemplo basándonos en el esquema XML **<Documentos>\Altova\MapForce2023\MapForceExamples\Tutorial\Products.xsd**. Este esquema viene acompañado de un documento de instancia llamado **Products.xml**. Para agregar tanto el esquema como el archivo de instancia al área de asignación seleccione **Insertar | Archivo o esquema XML** y navegue hasta el archivo **<Documentos>\Altova\MapForce2023\MapForceExamples\Tutorial\Products.xml**. Cuando la aplicación solicite un elemento raíz, seleccione `products`.

Para habilitar los nombres de nodo dinámicos para el nodo `product` haga clic con el botón derecho en ese nodo y seleccione uno de estos comandos en el menú contextual:

- **Mostrar atributos con nombre dinámico** si desea tener acceso a los atributos del nodo o
- **Mostrar elementos secundarios con nombre dinámico** si desea tener acceso a los elementos secundarios del nodo.

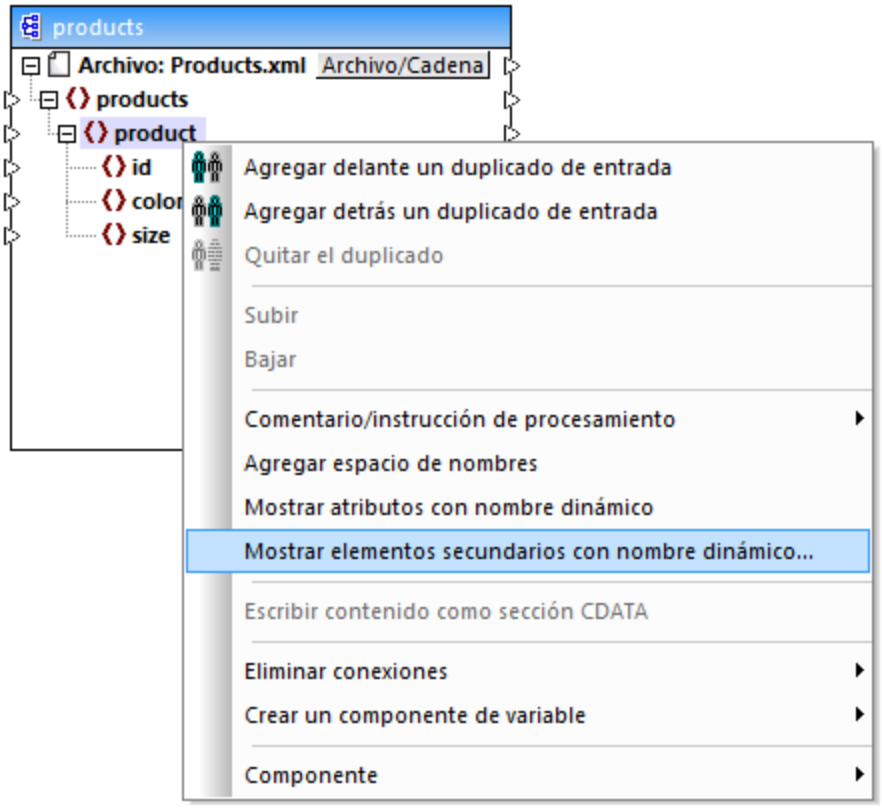


Fig. 1: habilitar nombres de nodo dinámicos (para elementos secundarios)

Nota: los comandos que aparecen en la imagen anterior solamente están disponibles cuando se trata de nodos con nodos secundarios. Además, estos comandos no están disponibles cuando se trata del nodo raíz.

Cuando se cambia un nodo al modo dinámico, aparece un cuadro de diálogo como el de la imagen siguiente. Siguiendo con nuestro ejemplo, ahora configuramos las opciones que puede ver en la imagen (estas opciones se definen en detalle en el apartado Obtener acceso a determinado tipo de nodos 858).

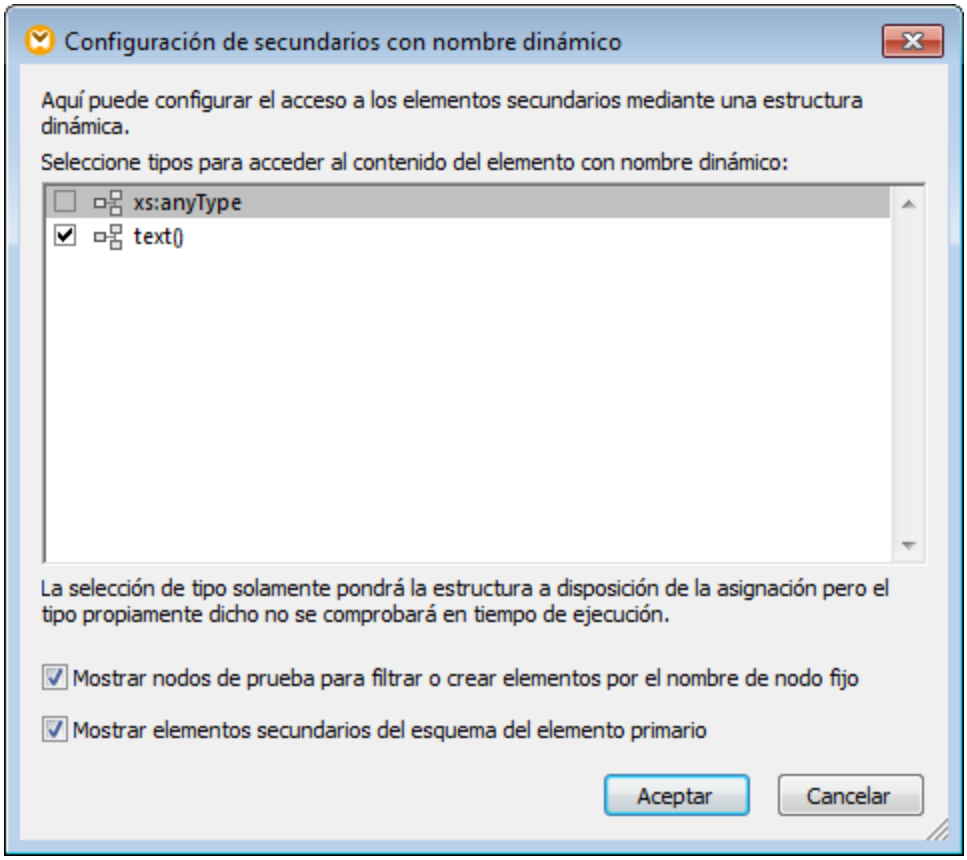


Fig. 2: cuadro de diálogo "Configuración de secundarios con nombre dinámico"

A continuación explicamos lo que hace el componente cuando se habilitan los nombres de nodo dinámicos para el nodo `product`. Observe que el aspecto del componente cambia significativamente.

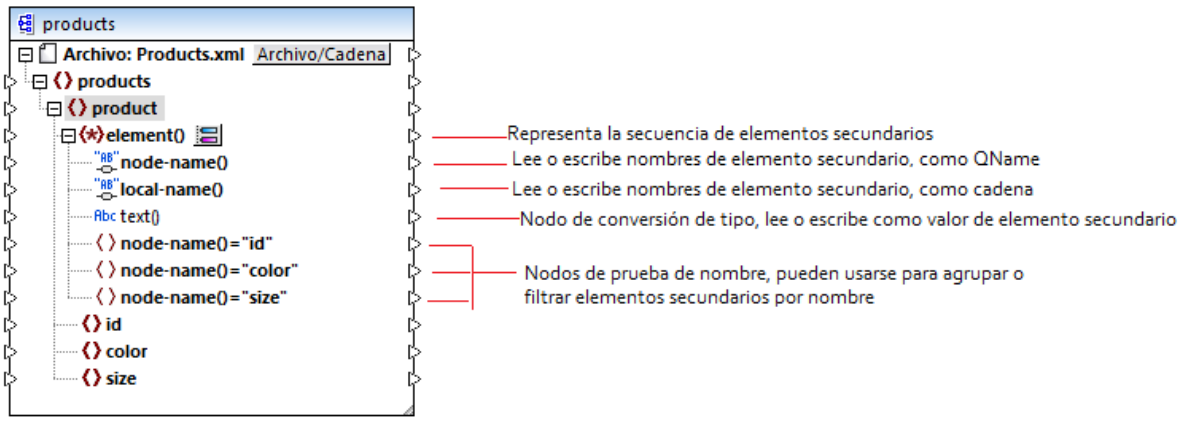


Fig. 3: nombres de nodo dinámicos habilitados (para elementos secundarios)

Para restaurar el modo estándar del componente haga clic con el botón derecho en el nodo `product` y desactive el comando **Mostrar elementos secundarios con nombre dinámico** en el menú contextual.

En la imagen siguiente podemos ver el aspecto que tiene el componente cuando se habilita el acceso dinámico a los atributos de un nodo (haciendo clic con el botón secundario en el elemento `product` y seleccionando el comando **Mostrar atributos con nombre dinámico** en el menú contextual).

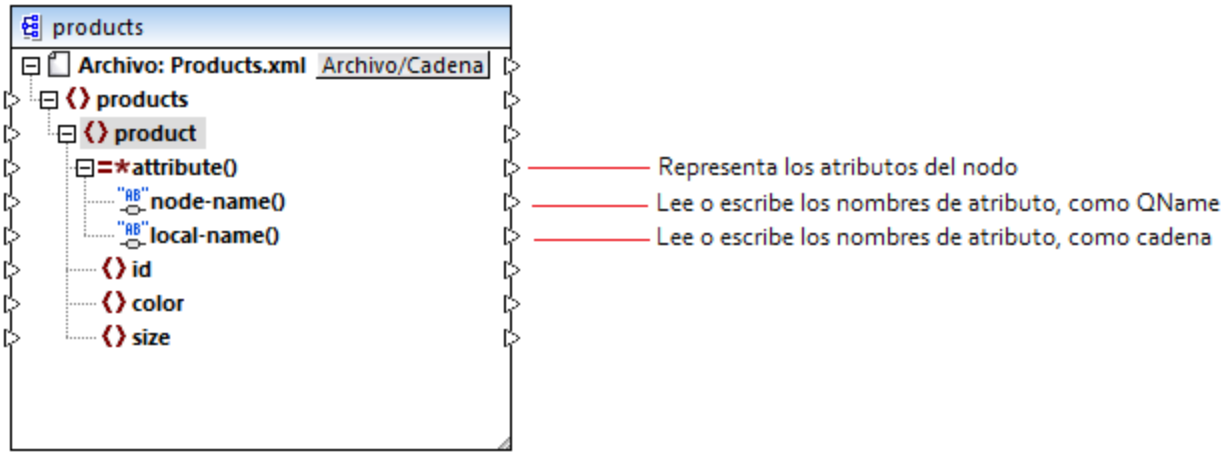


Fig. 4: nombres de nodo dinámicos habilitados (para atributos)

Para restaurar el modo estándar del componente haga clic con el botón derecho en el nodo `product` y desactive el comando **Mostrar atributos con nombre dinámico** en el menú contextual.

Como puede ver en las dos imágenes anteriores, el aspecto del componente cambia cuando se habilita el modo *nombres de nodo dinámicos* en uno de los nodos (en nuestro ejemplo se trata del nodo `product`). El nuevo aspecto del componente ofrece varias opciones nuevas porque permite:

- Leer o escribir una lista con todos los elementos secundarios o atributos de un nodo (que vienen dados por el elemento `element()` o `attribute()` respectivamente).
- Leer o escribir el nombre de cada elemento secundario o atributo (que viene dado por el elemento `node-name()` o `local-name()` respectivamente).
- Si se trata de elementos, leer o escribir el valor de cada elemento secundario, en el tipo de datos que corresponda. Este valor viene dado por el nodo de conversión de tipo (el elemento `text()`). Recuerde que solamente los elementos pueden tener nodos de conversión de tipo. Los atributos siempre tienen el tipo "string".
- Agrupar o filtrar elementos secundarios por nombre (*véase* *Ejemplo: agrupar y filtrar nodos según su nombre* 866).

A continuación explicamos con qué tipos de nodos puede trabajar cuando use el modo *nombres de nodo dinámicos.*

element()

Este nodo diferentes comportamientos, dependiendo de si está en el componente de origen o en el componente de destino. Si está en el componente de origen, aporta los elementos secundarios del nodo en forma de secuencia. En la Fig. 3 (ver más arriba) el elemento `element()` ofrece una lista (secuencia) con todos los elementos secundarios de `product`. Por ejemplo, la secuencia que se crea a partir de este XML contendría tres elementos (porque `product` tiene tres elementos secundarios):

 © 2016-2022 Altova GmbH

```
<product>
   <id>1</id>
   <color>red</color>
   <size>10</size>
</product>
```

Observe que el nombre y el tipo de cada elemento de la secuencia es el que nos da el nodo `node-name()` y el nodo de conversión de tipo respectivamente. Imagine que necesita pasar datos de un archivo XML a otro XML de destino de la siguiente manera:

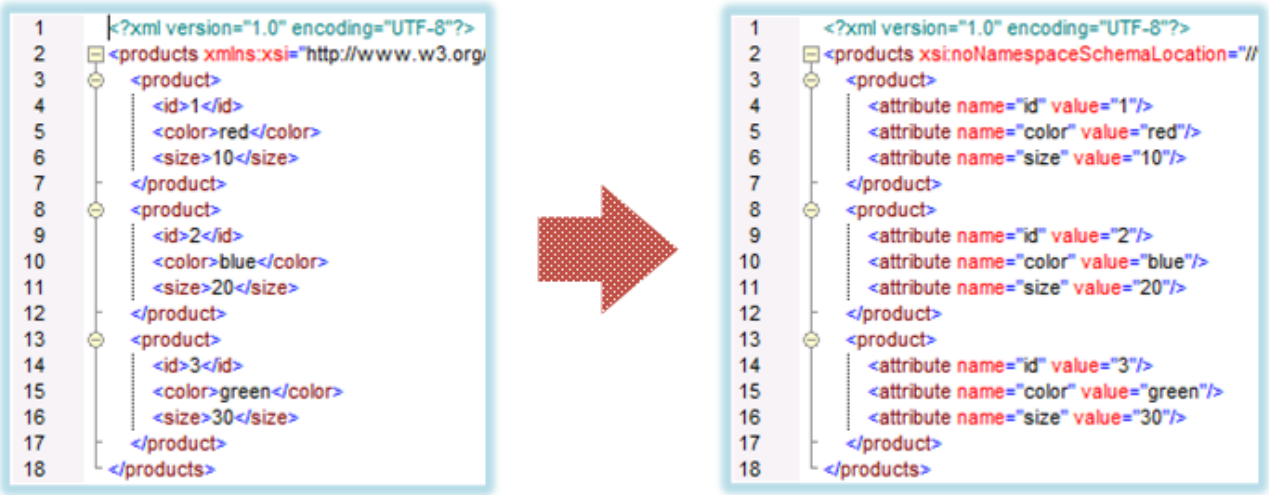


Fig. 6: asignación entre nombres de elemento XML y valores de atributo

Para conseguir este objetivo debemos diseñar esta asignación de datos:

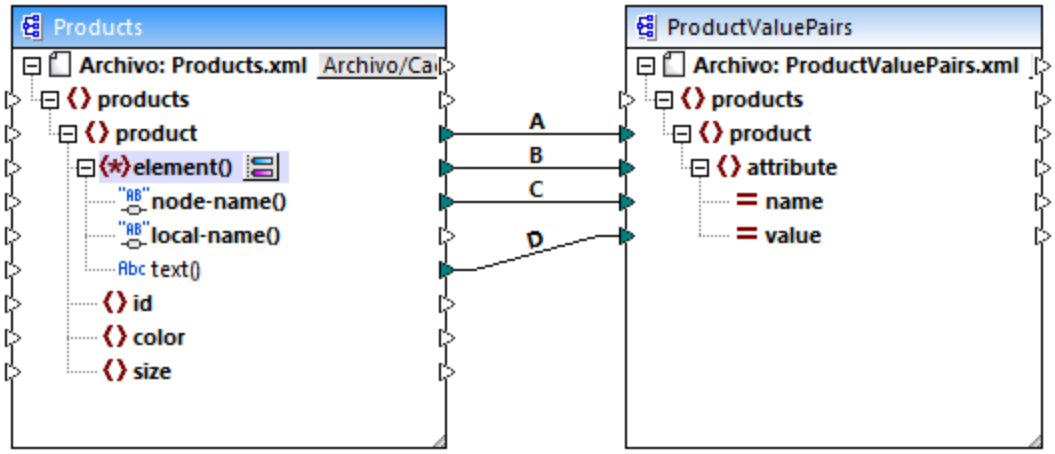


Fig. 7: asignar nombres de elemento XML a valores de atributo

El papel que `element()` desempeña aquí es aportar la secuencia de elementos secundarios de `product`, mientras que `node-name()` y `text()` aportan el nombre y el valor real de cada elemento de la secuencia. Esta

asignación se describe más detalladamente en el apartado Ejemplo: asignar nombres de elemento a valores de atributo 863.

En el componente de destino `element()` no crea nada por sí mismo, lo cual constituye una excepción de la regla básica de asignación (por cada elemento del origen, crear un elemento de destino). Los elementos propiamente dichos los crean los nodos de conversión de tipo (usando el valor de `node-name()`) y por los nodos de prueba de nombre (usando su propio nombre).

attribute()

Como puede verse en la figura nº4, este elemento permite acceder a todos los atributos del nodo en tiempo de ejecución de la asignación. En un componente de origen aporta, en forma de secuencia, los atributos del nodo de origen que está conectado. Por ejemplo, en este archivo XML, la secuencia incluiría dos elementos (porque `product` tiene dos atributos):

```
<product id="1" color="red" />
```

Observe que el nodo `attribute()` solo aporta el valor de cada atributo en la secuencia, siempre como tipo string. El nombre de cada atributo viene dado por el nodo `node-name()`.

En un componente de destino este nodo procesa una secuencia conectada y crea un valor de atributo por cada elemento de la secuencia. El nombre de atributo viene dado por `node-name()`. Por ejemplo, imagine que necesita pasar datos de un archivo XML a otro archivo XML de la siguiente manera:

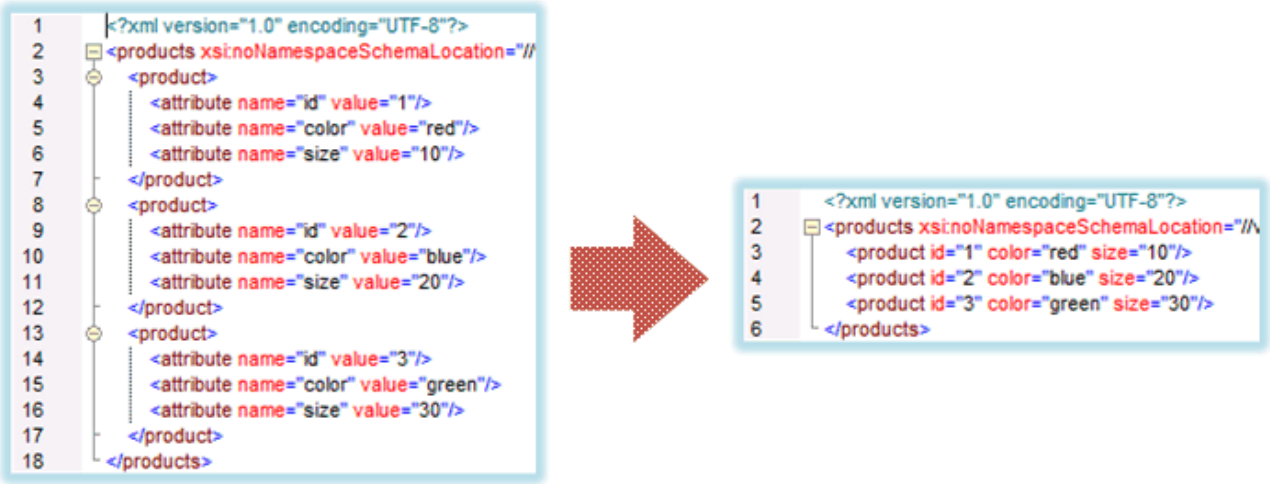


Fig. 8: asignar valores a nombres de atributo

Para conseguir este objetivo debemos diseñar esta asignación de datos:

 © 2016-2022 Altova GmbH

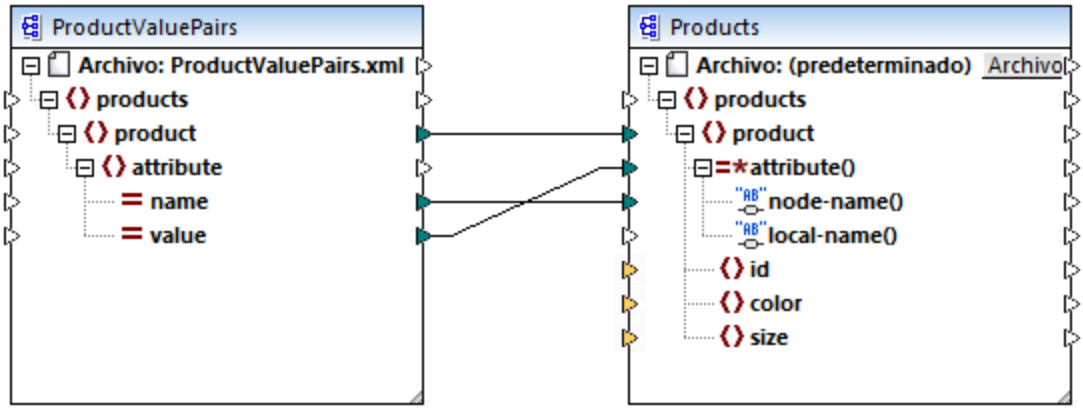


Fig. 9: asignar valores de atributo a nombres de atributo

Nota: esta transformación puede conseguirse sin habilitar el acceso dinámico a los atributos del nodo, pero aquí usamos el acceso dinámico para explicar cómo funciona `attribute()` en un componente de destino.

Si quiere reconstruir esta asignación, tenga en cuenta que usa los mismos componentes XML que la asignación **ConvertProducts.mfd** de la carpeta **<Documentos>\Altova\MapForce2023\MapForceExamples\Tutorial**. La única diferencia es que el destino ahora es el origen y el origen ahora es el destino. Como datos de entrada del componente de origen deberá utilizar una instancia XML que contenga valores de atributo, por ejemplo:

```
<?xml version="1.0" encoding="UTF-8"?>
<products>
   <product>
      <attribute name="id" value="1"/>
      <attribute name="color" value="red"/>
      <attribute name="size" value="big"/>
   </product>
</products>
```

Nota: para simplificar el ejemplo se omitió la declaración de espacio de nombres y de esquema en el fragmento de código anterior.

node-name()

En un componente de origen `node-name()` aporta el nombre de cada elemento secundario de `element()` o el nombre de cada atributo de `attribute()` respectivamente. Por defecto el nombre que aporta `node-name()` es de tipo `xs:QName`. Para obtener el nombre como tipo string debe utilizarse el nodo `local-name()` (*véase la figura nº3*) o utilice la función QName-as-string 758.

En un componente de destino `node-name()` escribe el nombre de cada elemento o atributo que exista en `element()` o `attribute()`.

local-name()

Este nodo funciona igual que el nodo `node-name()`, pero su tipo es `xs:string` en lugar de `xs:QName`.

Nodo de conversión de tipo

En un componente de origen el nodo de conversión de tipo aporta el valor de cada elemento secundario que incluye `element()`. El nombre y la estructura de este nodo dependerá del tipo qeu esté seleccionado en el cuadro de diálogo "Configuración de secundarios con nombre dinámico" (figura nº2).

Para cambiar el tipo del nodo haga clic en el botón **Cambiar selección** y seleccione un tipo de la lista de tipos disponibles, incluido el comodín de esquema (`xs:any`). Consulte Obtener acceso a determinado tipo de nodos 858 para obtener más información.

En un componente de destino el nodo de conversión de tipo escribe el valor de cada elemento secundario que incluye `element()`, en el tipo de datos correspondiente. El tipo de datos deseado se selecciona haciendo clic en el botón **Cambiar selección**.

Nodos de prueba de nombre

En un componente de origen los nodos de prueba de nombre permiten agrupar o filtrar elementos secundarios de una instancia de origen por nombre. Por ejemplo, si necesita filtrar elementos secundarios por nombre para que la asignación acceda a los datos de instancia usando el tipo correcto (*véase* *Obtener acceso a determinado tipo de nodos* 858). Para ver un ejemplo consulte Ejemplo: agrupar y filtrar nodos según su nombre 866.

En general los nodos de prueba de nombre funcionan casi igual que los nodos de elemento normales para leer y escribir valores y subestructuras. Sin embargo, como la semántica de la asignación es distinta cuando está habilitado el acceso dinámico, existen algunas restricciones. Por ejemplo, no se puede concatenar el valor de dos nodos de prueba de nombre.

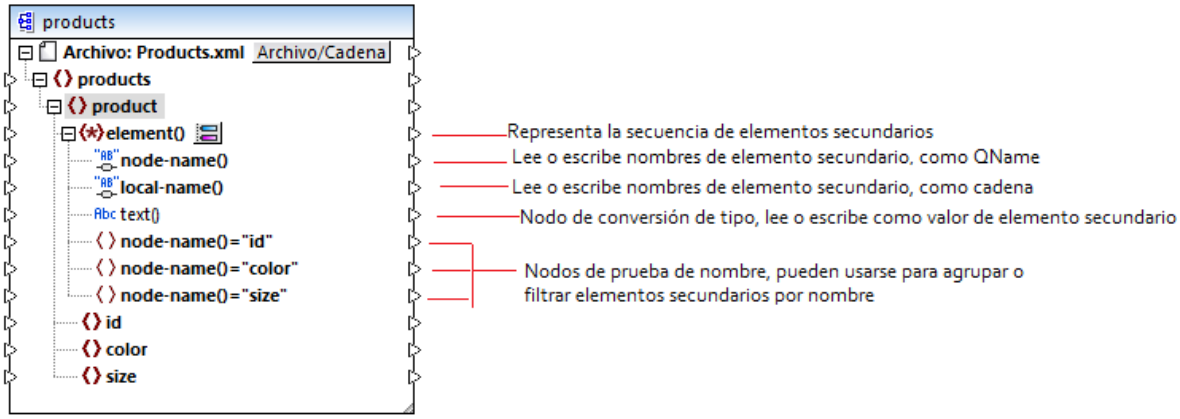
En un componente de destino los nodos de prueba de nombre crean tantos elementos en el resultado como elementos existen en la secuencia de origen conectada. Su nombre reemplaza el valor que esté asignado a `node-name()`.

Si es necesario, puede ocultar los nodos de prueba de nombre en el componente. Esto se hace haciendo clic con el botón **Cambiar selección** del nodo `element()` y desactivando la casilla *Mostrar nodos de prueba....* en el cuadro de diálogo "Configuración de secundarios con nombre dinámico" (figura nº2).

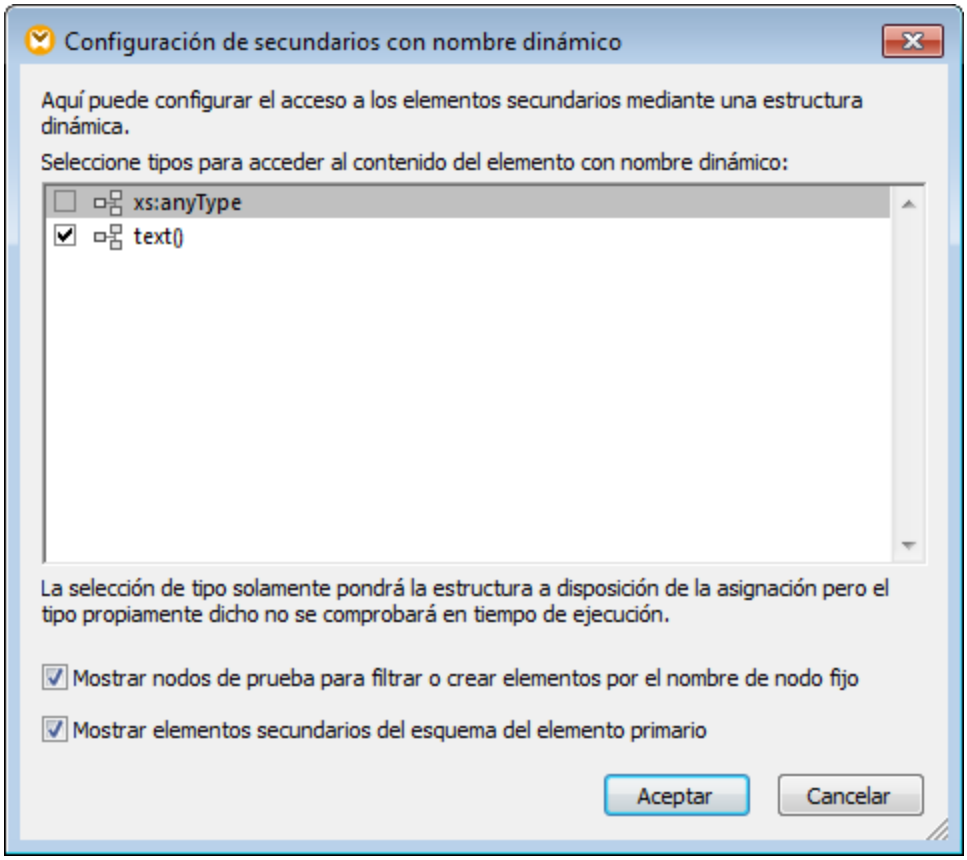
7.2.2 Obtener acceso a determinado tipo de nodos

Como decíamos en el apartado anterior, podemos acceder a todos los elementos secundarios de un nodo haciendo clic con el botón derecho en el nodo y seleccionando el comando **Mostrar elementos secundarios con nombre dinámico** en el menú contextual. En tiempo de ejecución esto permitirá acceder al nombre de cada elemento secundario a través del nodo `node-name()`, mientras que el valor estará disponible a través de un nodo de conversión de tipo especial. En la imagen siguiente podemos ver que el nodo de conversión de tipo es el nodo `text()`.

 © 2016-2022 Altova GmbH



Es importante señalar que el tipo de datos de cada elemento secundario no se conoce antes de que se ejecute la asignación. Además, el tipo de datos de cada elemento secundario puede ser distinto. Por ejemplo, un nodo `product` del archivo de instancia XML puede tener un elemento secundario `id` de tipo `xs:integer` y un elemento secundario `size` de tipo `xs:string`. Para poder acceder al contenido del nodo de un tipo específico el cuadro de diálogo (*imagen siguiente*) se abre cada vez que se habilita el acceso dinámico a los elementos secundarios de un nodo. Este cuadro de diálogo se puede abrir en cualquier momento con el botón **Cambiar selección** situado junto al nodo `element()`.



Cuadro de diálogo "Configuración de secundarios con nombre dinámico"

Para acceder al contenido de cada elemento en tiempo de ejecución tenemos varias opciones:

1. Acceder al contenido como cadena: marque la casilla **text()** del cuadro de diálogo. En este caso, se crea un nodo `text()` en el componente cuando se cierra el cuadro de diálogo. Esta opción es la más adecuada si el contenido es de tipo simple (`xs:int`, `xs:string`, etc.) y se describe en el apartado Ejemplo: asignar nombres de elemento a valores de atributo 863. Recuerde que el nodo **text()** solamente aparece si un nodo secundario del nodo actual puede contener texto.
2. Acceder al contenido como cierto tipo complejo permitido por el esquema. Cuando el esquema permite para el nodo seleccionado tipos complejos personalizados definidos globalmente, éstos también estarán disponibles en el cuadro de diálogo y podremos marcar sus casillas. En la imagen anterior, por ejemplo, no hay tipos complejos definidos globalmente por el esquema así que no hay ninguno en el cuadro de diálogo.
3. Acceder al contenido como cualquier tipo. Esto puede ser muy práctico en asignaciones de datos complejas y se consigue marcando la casilla de **xs:anyType**.

Debe tener en cuenta que en tiempo de ejecución MapForce no dispone de información (a través del nodo de conversión de tipo) sobre el tipo real del nodo de instancia. Por tanto, su asignación debe acceder al contenido del nodo usando el tipo correcto. Por ejemplo, si se espera que el nodo de una instancia XML de origen tenga nodos secundarios de varios tipos complejos, deberá hacer lo siguiente:
a) Asignar al nodo de conversión de tipo el tipo complejo que debe corresponderle.

 © 2016-2022 Altova GmbH

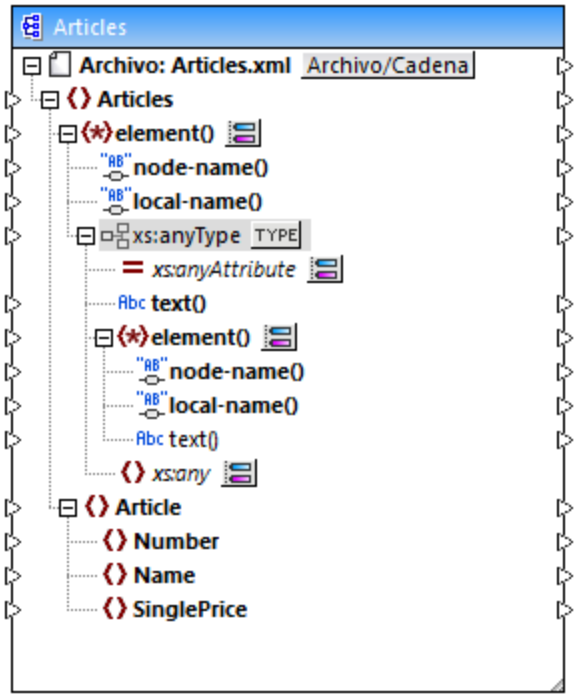
b) Agregar un filtro para leer de la instancia solamente el tipo complejo que debe corresponder.

Esta técnica se describe con más detalle en Ejemplo: agrupar y filtrar nodos según su nombre 866.

Acceder a estructuras más profundas

En asignaciones más complejas es posible acceder a nodos de niveles más profundos del esquema que el nivel de secundarios inmediatos de un nodo. En las asignaciones más sencillas como las del apartado Ejemplo: asignar nombres de elemento a valores de atributo 863 esta técnica no es necesaria porque la asignación solo accede a los secundarios inmediatos de un nodo XML. Sin embargo, cuando lo necesite también podrá acceder de forma dinámica a estructuras más profundas (p. ej. a *nietos*, *bisnietos*, etc.) si sigue estos pasos:

1. Cree una asignación nueva.
2. Seleccione el comando de menú **Insertar | Archivo o esquema XML** y navegue hasta el archivo XML de instancia (p. ej. **Articles.xml** de la carpeta **<Documentos>\Altova\MapForce2023\MapForceExamples\Tutorial**).
3. Haga clic con el botón derecho en el nodo `Articles` y seleccione **Mostrar elementos secundarios con nombre dinámico** en el menú contextual.
4. Marque la casilla **xs:anyType** en el cuadro de diálogo "Configuración de secundarios con nombre dinámico".
5. Haga clic con el botón derecho en el nodo `xs:anyType` y seleccione otra vez **Mostrar elementos secundarios con nombre dinámico** en el menú contextual.
6. Marque la casilla **text()** en el cuadro de diálogo "Configuración de secundarios con nombre dinámico".



En la imagen puede ver que el componente tiene dos nodos `element()`. El segundo de ellos ofrece acceso dinámico a los nietos del nodo `<Articles>` de la instancia **Articles.xml**.

```
<?xml version="1.0" encoding="UTF-8"?>
<Articles xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="Articles.xsd">
   <Article>
      <Number>1</Number>
      <Name>T-Shirt</Name>
      <SinglePrice>25</SinglePrice>
   </Article>
   <Article>
      <Number>2</Number>
      <Name>Socks</Name>
      <SinglePrice>2.30</SinglePrice>
   </Article>
   <Article>
      <Number>3</Number>
      <Name>Pants</Name>
      <SinglePrice>34</SinglePrice>
   </Article>
   <Article>
      <Number>4</Number>
      <Name>Jacket</Name>
      <SinglePrice>57.50</SinglePrice>
   </Article>
</Articles>
```

Articles.xml

Por ejemplo, para obtener los nombres de elementos *nietos* (`Number`, `Name`, `SinglePrice`) debemos dibujar una conexión entre el nodo `local-name()` del segundo nodo `element()` y un elemento de destino. Igualmente para obtener los valores de elementos *nietos* (`1`, `T-Shirt`, `25`), debemos dibujar una conexión desde el nodo `text()`.

Aunque en este ejemplo no es relevante, MapForce ofrece la posibilidad de seguir habilitando nombres de nodo dinámico para los siguientes nodos `xs:anyType` y así poder alcanzar niveles aún más profundos de la estructura.

Debe tener en cuenta que:

- El botón TYPE sirve para seleccionar cualquier tipo derivado del esquema actual y mostrarlo en un nodo independiente. Esto puede ser práctico a la hora de crear asignaciones entre tipos de esquema derivados (*véase* *Tipos XML Schema derivados* 129).
- El botón **Cambiar selección** situado junto a un nodo `element()` abre el cuadro de diálogo "Configuración de secundarios con nombre dinámico".
- El botón **Cambiar selección** situado junto a `xs:anyAttribute` permite seleccionar cualquier atributo definido globalmente en el esquema. Igualmente el botón **Cambiar selección** situado junto al elemento `xs:any` permite seleccionar cualquier elemento definido globalmente en el esquema. Se trata del mismo funcionamiento de las asignaciones de comodines de esquema (*véase* *Comodines: xs:any / xs:anyAttribute* 137). Si usa esta opción, asegúrese de que el esquema permite que el atributo o elemento exista de verdad.

 © 2016-2022 Altova GmbH

7.2.3 Ejemplo: asignar nombres de elemento a valores de atributo

Este ejemplo demuestra cómo asignar nombres de elemento de un documento XML a valores de atributo de un documento XML de destino. El ejemplo viene acompañado del diseño de asignación **<Documentos>\Altova\MapForce2023\MapForceExamples\Tutorial\ConvertProducts.mfd**.

Para comprender cómo funciona el ejemplo primero debemos imaginar que tenemos un archivo XML que contiene una lista de productos. Cada producto tiene este formato:

```
<product>
   <id>1</id>
   <color>red</color>
   <size>10</size>
</product>
```

Nuestro objetivo es convertir la información disponible sobre cada producto en pares de nombre/valor. Por ejemplo:

```
<product>
   <attribute name="id" value="1" />
   <attribute name="color" value="red" />
   <attribute name="size" value="10" />
</product>
```

Para conseguirlo el diseño de asignación del ejemplo utiliza la característica de MapForce conocida como "acceso dinámico a nombres de nodo". El término *dinámico* hace referencia al hecho de que, cuando se ejecute la asignación, se podrán leer los nombres de nodo (no solo los valores) y estos nombres se podrán usar como valores. A continuación explicamos cómo diseñar esta asignación.

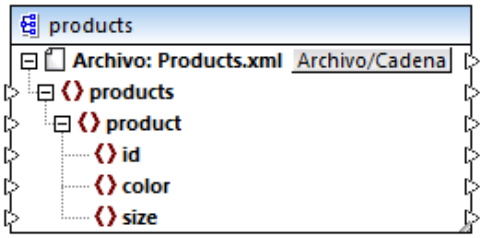
Paso nº1: agregar el componente XML de origen a la asignación

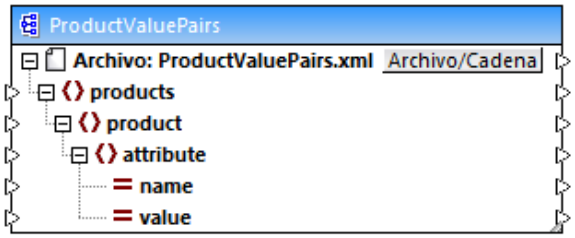
- En el menú **Insertar** haga clic en el comando **Archivo o esquema XML** y navegue hasta el archivo **<Documentos>\Altova\MapForce2023\MapForceExamples\Tutorial\Products.xml**. Este archivo XML apunta al esquema **Products.xsd** que está situado en la misma carpeta.

Paso nº2: agregar el componente XML de destino a la asignación

- En el menú **Insertar** haga clic en el comando **Archivo o esquema XML** y navegue hasta el archivo de esquema **<Documentos>\Altova\MapForce2023\MapForceExamples\Tutorial\ProductValuePairs.xsd**. Cuando la aplicación solicite un archivo de instancia, haga clic en **Omitir**. Cuando solicite un elemento raíz, seleccione `products`.

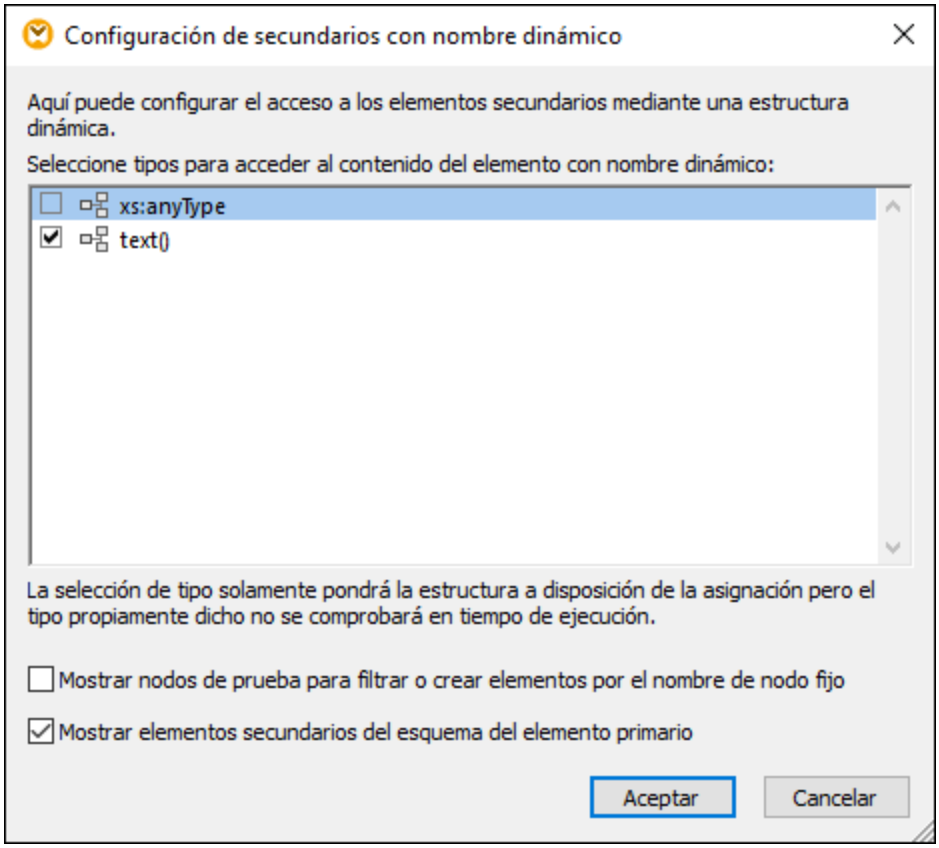
Llegados a este punto el diseño tendrá este aspecto:





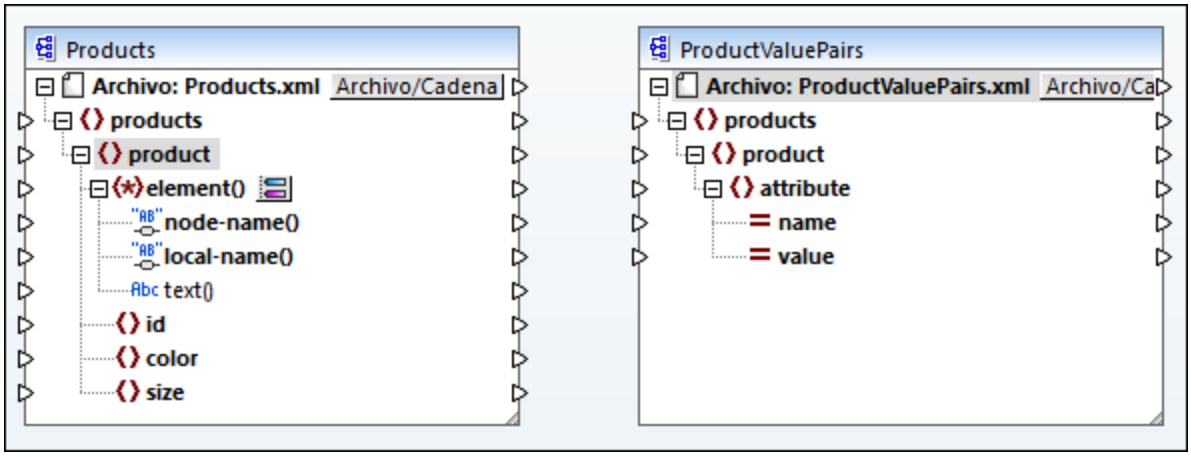
Paso nº3: habilitar el acceso dinámico a los nodos secundarios

1. Haga clic con el botón derecho en el nodo `product` del componente de origen y seleccione **Mostrar elementos secundarios con nombre dinámico** en el menú contextual.
2. En el cuadro de diálogo que aparece seleccione el tipo **text()** y deje las demás opciones como están.



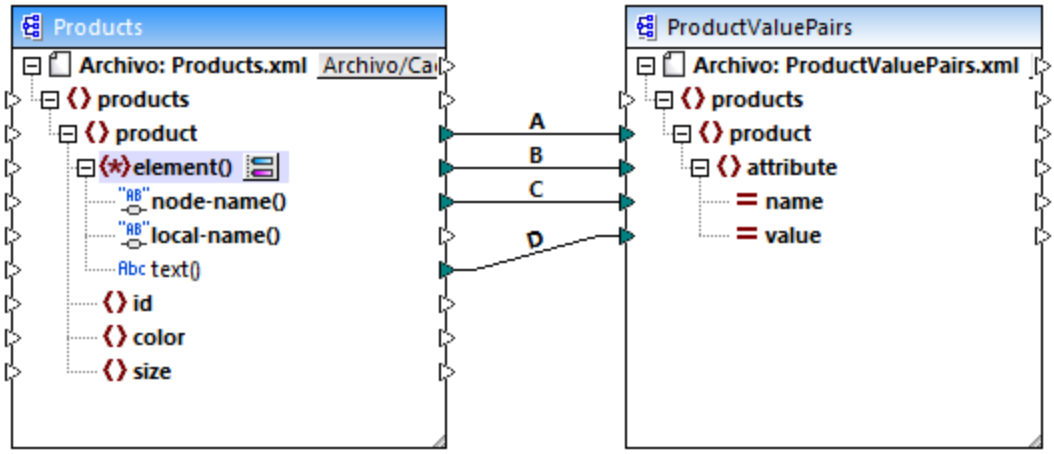
Observe que esto añade un nodo llamado `text()` al componente de origen. Este nodo se encargará de suministrar el contenido de los elementos secundarios a la asignación (en este caso, el valor de "id", "color" y "size").

 © 2016-2022 Altova GmbH



Paso nº4: dibujar las conexiones de asignación de datos

Por último, debemos dibujar las conexiones de asignación de datos A, B, C y D que pueden verse en la imagen siguiente (si quiere, haga doble clic en cada línea de conexión, empezando por la primera, e introduzca el texto "A", "B", "C" y "D" respectivamente en el cuadro *Descripción*.



ConvertProducts.mfd

En el diseño de asignación anterior, la conexión A crea en el componente de destino un producto por cada producto del componente de origen. Se trata de una conexión estándar de MapForce que no se ocupa de los nombres de los nodos. Sin embargo, la conexión B crea en el componente de destino un elemento nuevo llamado `attribute` por cada elemento secundario de `product` del componente de origen.

La conexión B es una conexión crucial en la asignación pues se encarga de transferir una *secuencia* de elementos secundarios de `product` desde el componente de origen al componente de destino. No transfiere los nombres ni valores propiamente dichos. Por tanto, debe entenderse de la siguiente forma: si el **element()** de origen tiene X secundarios, entonces crea X instancias de dicho elemento en el componente de destino. En este caso concreto, `product` tiene tres secundarios en el componente de

origen (`id`, `color` y `size`). Esto significa que cada `product` del componente de destino tendrá tres secundarios llamados `attribute`.

Aunque ahora no es el caso, la misma regla puede utilizarse para asignar elementos secundarios de **attribute()**: si el elemento **attribute()** de origen tiene X atributos secundarios, la conexión creará X instancias de dicho elemento en el componente de destino.

Después, la conexión C copia el nombre real de cada elemento secundario de `product` en el componente de destino (literalmente "id", "color" y "size").

Por último, la conexión D copia el valor de cada elemento secundario de `product` (como tipo `string`) en el componente de destino.

Para consultar una vista previa de los resultados de la asignación abrimos el panel *Resultados* y consultamos el XML que se genera. Tal y como esperábamos, el resultado contiene varios productos cuyos datos están almacenados como pares nombre/valor.

```
<?xml version="1.0" encoding="UTF-8"?>
<products xsi:noNamespaceSchemaLocation="ProductValuePairs.xsd"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <product>
      <attribute name="id" value="1"/>
      <attribute name="color" value="red"/>
      <attribute name="size" value="10"/>
   </product>
   <product>
      <attribute name="id" value="2"/>
      <attribute name="color" value="blue"/>
      <attribute name="size" value="20"/>
   </product>
   <product>
      <attribute name="id" value="3"/>
      <attribute name="color" value="green"/>
      <attribute name="size" value="30"/>
   </product>
</products>
```

Resultado de la asignación

7.2.4 Ejemplo: agrupar y filtrar nodos según su nombre

Este ejemplo demuestra cómo diseñar una asignación que lee pares de clave/valor de una lista de propiedades XML (o XML plist) y los escribe en un archivo CSV. Las listas de propiedades XML permiten almacenar información de objetos OS X e iOS en formato XML (véase https://developer.apple.com/library/mac/documentation/Cocoa/Conceptual/PropertyLists/UnderstandXMLPlist/UnderstandXMLPlist.html.) Este ejemplo viene acompañado de un diseño de asignación: **<Documentos>\Altova\MapForce2023\MapForceExamples\Tutorial\ReadPropertyList.mfd**.

 © 2016-2022 Altova GmbH

A continuación podemos ver un fragmento de código del archivo XML de origen.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist SYSTEM "https://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
   <dict>
      <key>First Name</key>
      <string>William</string>
      <key>Last Name</key>
      <string>Shakespeare</string>
      <key>Birthdate</key>
      <integer>1564</integer>
      <key>Profession</key>
      <string>Playwright</string>
      <key>Lines</key>
      <array>
         <string>It is a tale told by an idiot,</string>
         <string>Full of sound and fury, signifying nothing.</string>
      </array>
   </dict>
</plist>
```

El objetivo de la asignación es crear una línea nueva en el archivo CSV a partir de ciertos pares de clave/valor del nodo `<dict>` del archivo de lista de propiedades. Concretamente, la asignación solamente filtrará los pares **`<key>`** - **`<string>`** y el resto de pares clave/valor (p. ej. **`<key>`** - **`<integer>`**) se pasarán por alto. En el archivo CSV la nueva línea debe almacenar el nombre de la propiedad, separado del valor de la propiedad por una coma. En otras palabras, el resultado de la asignación debe tener este aspecto:

```
First Name,William
Last Name,Shakespeare
Profession,Playwright
```

Para conseguir el objetivo de la asignación será necesario tener acceso dinámico a los nodos secundarios del nodo `dict`. Además, necesitaremos la función group-starting-with 694 para agrupar los pares de clave/valor recuperados del archivo XML. Y, por último, deberemos usar un filtro para filtrar los nodos cuyo nombre sea "string".

A continuación explicamos cómo diseñar esta asignación paso a paso.

Paso nº1: agregar el componente XML de origen a la asignación

1. Elija el lenguaje de transformación integrado (*véase Seleccionar un lenguaje de transformación* 24).
2. En el menú **Insertar** haga clic en el comando **Archivo o esquema XML** y navegue hasta el archivo **<Documentos>\Altova\MapForce2023\MapForceExamples\Tutorial\plist.xml**. Este archivo XML apunta al esquema **plist.xsd** que está situado en la misma carpeta.

Paso nº2: agregar el componente CSV de destino a la asignación

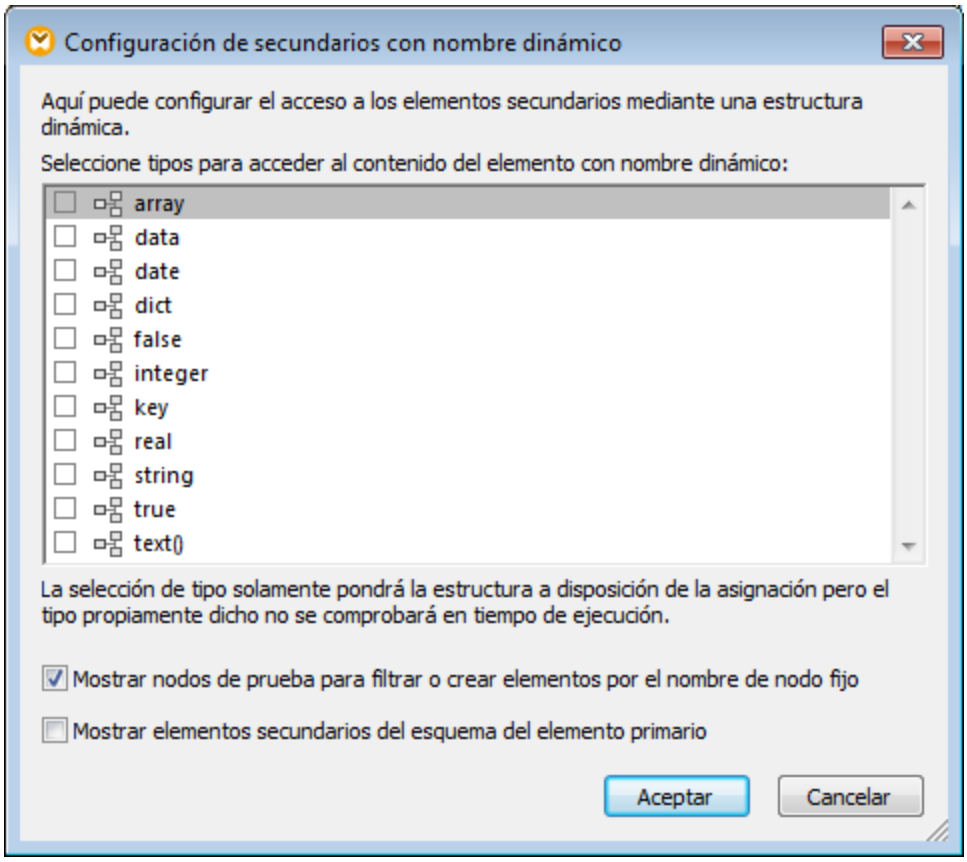
1. En el menú **Insertar** haga clic en el comando **Archivo de texto**. Ahora seleccione la opción *Utilizar procesamiento básico para archivos CSV...* en el cuadro de diálogo.
2. Añada un campo CSV más al componente con el botón **Anexar campo**.

3. Haga doble clic en el nombre de cada campo e introduzca **`Key`** para el nombre del primer campo y **`Value`** para el nombre del segundo. El campo **`Key`** almacenará el nombre de la propiedad, mientras que el campo **`Value`** almacenará su valor. Consulte la sección Archivos CSV y archivos de texto 445 para obtener más información.

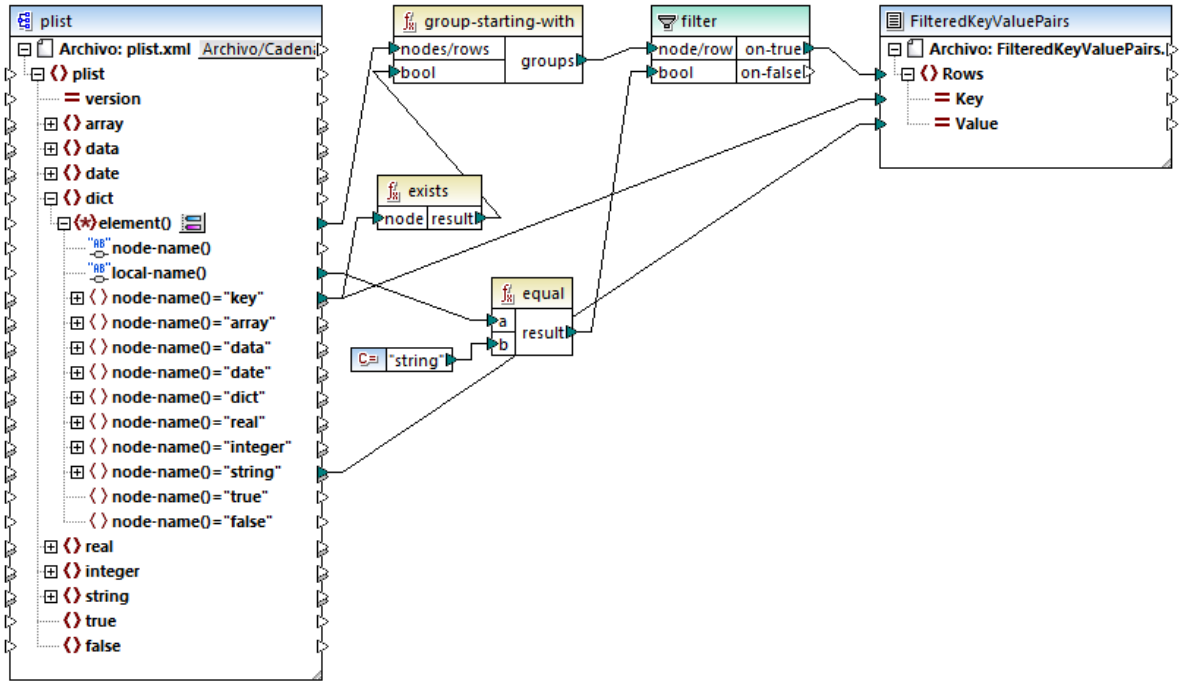
Paso nº3: agregar el filtro y funciones

1. Arrastre las funciones **`equal`** 663, **`exists`** 683 y **`group-starting-with`** 694 desde la ventana Bibliotecas hasta el área de asignación (*véase Funciones básicas* 549).
2. Para agregar el filtro haga clic en el menú **Insertar** y seleccione el comando **Filtro: nodos/filas** (*véase Filtros y condiciones* 543).
3. En el menú **Insertar** seleccione el comando **Constante** e introduzca el texto "string".
4. En el componente de origen haga clic con el botón derecho en el nodo `dict` y seleccione **Mostrar elementos secundarios con nombre dinámico** en el menú contextual. En el cuadro de diálogo que aparece compruebe que la casilla *Mostrar nodos de prueba para filtrar o crear elementos...* está marcada.

© 2016-2022 Altova GmbH



5. Dibuje las conexiones que aparecen en la imagen siguiente.



ReadPropertyList.mfd

Funcionamiento de la asignación

El elemento `element()` del componente de origen suministra todos los secundarios del nodo `dict` en forma de secuencia a la función **`group-starting-with`**. La función **`group-starting-with`** crea un grupo nuevo cada vez que se detecta un nodo llamado `key`. La función **`exists`** comprueba si se cumple esta condición y devuelve el valor binario true/false a la función de agrupación.

Por cada grupo el filtro comprueba si el nombre del nodo actual equivale a "string" (con ayuda de la función **`equal`**). El nombre propiamente dicho se lee de `local-name()`, que suministra el nombre del nodo en forma de cadena.

Las conexiones con el componente de destino se encargan de lo siguiente:

- Cuando la condición de filtrado se cumple, entonces se crea una fila nueva en el componente CSV de destino.
- `Key` (nombre de la propiedad) se toma del valor del elemento `key` del componente de origen.
- `Value` (valor de la propiedad) se toma del nodo de prueba `string`.

 © 2016-2022 Altova GmbH

7.3 Reglas y estrategias de asignación

Por lo general, MapForce asigna los datos de manera intuitiva, pero puede que encuentre situaciones en las que los resultados contengan demasiados o demasiados pocos elementos. En este apartado le explicaremos cómo evitar situaciones en las que la asignación genera un resultado no deseado debido a conexiones o un contexto de asignación incorrectos.

Reglas de asignación

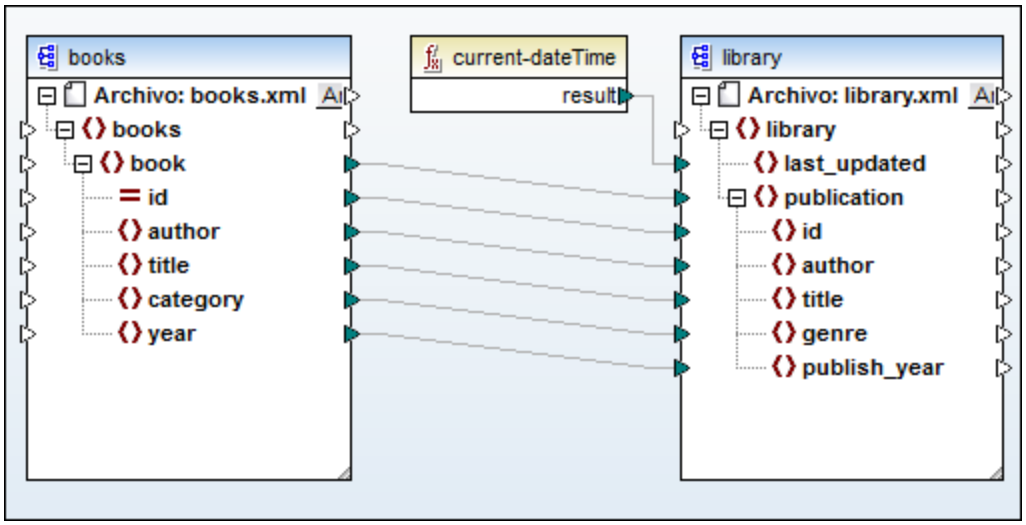
Para que una asignación sea válida, esta debe incluir al menos un componente de entrada y uno de salida. Un componente de entrada es aquel que lee datos, normalmente de un archivo o de una base de datos. Un componente de destino es el que escribe datos, normalmente en un archivo o en una base de datos. Si intenta guardar una asignación que no cumpla estos requisitos aparece un error en la ventana Mensajes: "Una asignación necesita tener al menos dos estructuras conectadas, por ejemplo un esquema o una estructura de BD".

Para crear una asignación de datos debe trazar las conexiones entre los elementos de los componentes de entrada y de salida.

Todas las conexiones de asignación que trace conforman el algoritmo de asignación. En el momento de la ejecución, MapForce evalúa el algoritmo y procesa los datos basándose en él. La integridad y la eficiencia del algoritmo de asignación depende principalmente de estas conexiones. También puede adaptar algunas opciones a nivel de asignación 115, de componente 79 o incluso de conexión 91, pero básicamente son las conexiones de la asignación las que determinan cómo se procesan los datos.

Tenga en cuenta estas reglas al crear las conexiones:

1. Al trazar una conexión *desde* un elemento de entrada, la asignación lee datos asociados a él en el archivo o la BD de entrada. Los datos pueden aparecer ninguna vez, una o varias veces (es decir, pueden ser una secuencia, como se describe a continuación). Por ejemplo, si la asignación lee datos de un archivo XML que contiene libros, el archivo XML de entrada puede contener ninguno, uno o varios elementos **book**. En la asignación siguiente, observe que el elemento **book** aparece solamente una vez en el componente de la asignación, aunque puede que el archivo (de instancia) de origen contenga varios elementos **book** o ninguno.



2. Al trazar una conexión a un elemento de destino, la asignación genera datos de instancia de esa clase. Si el elemento de entrada contiene contenido simple (por ejemplo, una cadena o un número entero) y el elemento de destino acepta contenido simple, entonces MapForce copia el contenido al elemento de destino y, si lo necesita, convierte el tipo de datos. Se pueden generar ningún valor, uno o varios en función de los datos de entrada (*véase el punto siguiente*).
3. Para cada elemento (de instancia) en el origen se crea un elemento (de instancia) en el destino. **Esta es la regla general de asignación en MapForce**. Si tomamos como ejemplo la asignación anterior, si el XML de origen contiene tres elementos **`book`**, entonces se crean tres elementos **`publication`** en el lado de destino. Tenga en cuenta que también existen algunos casos especiales (*véase el apartado Secuencias* 872).
4. Cada conexión crea un *contexto actual de asignación*. El contexto determina qué datos están disponibles *en el momento actual* para el *nodo de destino actual*. Por lo tanto, el contexto determina qué elementos de origen se copian de en el componente de destino. Al trazar u omitir una conexión puede cambiar el contexto actual sin querer, lo que afectaría al resultado de la asignación. Por ejemplo, puede que la asignación en la que está trabajando haga varias llamadas innecesarias a una BD o a un servicio web dentro de la misma ejecución de la asignación. Este concepto se describe con más detalle en Contexto de la asignación 874.

7.3.1 Secuencias

Como hemos mencionado anteriormente, la regla general de asignación es "por cada elemento de origen se crea uno de destino". Aquí, "elemento" puede significar:

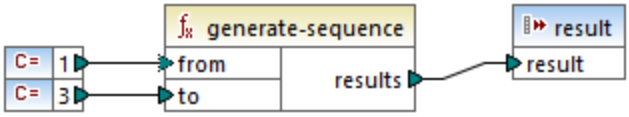
- una única instancia del archivo o la BD de entrada
- una secuencia de cero a varios nodos de instancia del archivo o la BD de entrada

Durante la ejecución de la asignación, si una secuencia alcanza un elemento de destino se crea un bucle que genera tantos nodos de destino como nodos de origen haya. Sin embargo, hay algunas excepciones a esta regla:

- Si el elemento de destino es un elemento XML raíz, entonces solo se crea una vez. Si conecta a él una secuencia, el resultado puede no ser un esquema válido. Si también se conectan a él atributos del elemento raíz, entonces la serialización XML fallará en tiempo de ejecución. Esto quiere decir que debe evitar conectar secuencias a elementos XML raíz.

 © 2016-2022 Altova GmbH

- Si el elemento de destino acepta solamente un valor, se crea una sola vez. Algunos ejemplos de elementos que aceptan solamente un valor: atributos XML, campos de BD, componentes simples de salida. Por ejemplo, la asignación siguiente genera una secuencia de tres números enteros (1, 2, 3) con ayuda de la función **generate-sequence**. Sin embargo, el resultado contendrá solamente un número entero porque el destino es un componente simple de destino que acepta solamente un valor. Los otros dos valores se ignoran.



- Si el esquema de origen indica que un elemento específico ocurre una sola vez pero el archivo de instancia contiene varios, MapForce extrae el primer elemento del origen (que, conforme al esquema, debe existir) y crea un solo elemento en el componente de destino. Para deshabilitar este comportamiento, desmarque la casilla *Optimi&zación de procesamiento de datos de entrada basada en minOccurs/maxOccurs* en la configuración del componente (*véase también* *Configuración de componentes XML* 125).

Si la secuencia está vacía no se genera nada en el lado de destino. Por ejemplo, si el componente de destino es un documento XML y la secuencia de origen está vacía, entonces no se crearía ningún elemento XML en el componente de destino.

Las funciones funcionan de forma parecida: si obtienen una secuencia como entrada, entonces se les llama tantas veces como (y generan tantos resultados como) elementos haya en la secuencia.

> Si una función obtiene una secuencia vacía como entrada, también devuelve un resultado vacío, por lo que no genera ninguna salida.

Sin embargo, hay algunas categorías o funciones que, por motivos de diseño, devuelven un valor incluso aunque obtengan una secuencia vacía como entrada:

- **exists**, **not-exists**, **substitute-missing**
- **is-null**, **is-not-null**, **substitute-null** (estas tres funciones son alias de las tres anteriores)
- funciones agregadas (**sum**, **count**, etc.)
- funciones definidas por el usuario que aceptan secuencias y son funciones regulares (no inline)

Si necesita reemplazar algún valor vacío, añada la función **substitute-missing** a la asignación y reemplace el valor vacío con otro valor. También puede obtener el mismo resultado usando Valores predeterminados y funciones 558.

Las funciones pueden tener distintos elementos de entrada. Si se conecta una secuencia a cada elemento de entrada, esto genera un producto cartesiano de todos los elementos de entrada, que no suele ser el resultado deseado..Para evitarlo, conecte solamente una secuencia a una función con varios parámetros: todos los demás parámetros se deben conectar a elementos "únicos" de elementos de nivel superior u otros componentes.

7.3.2 Contexto de la asignación

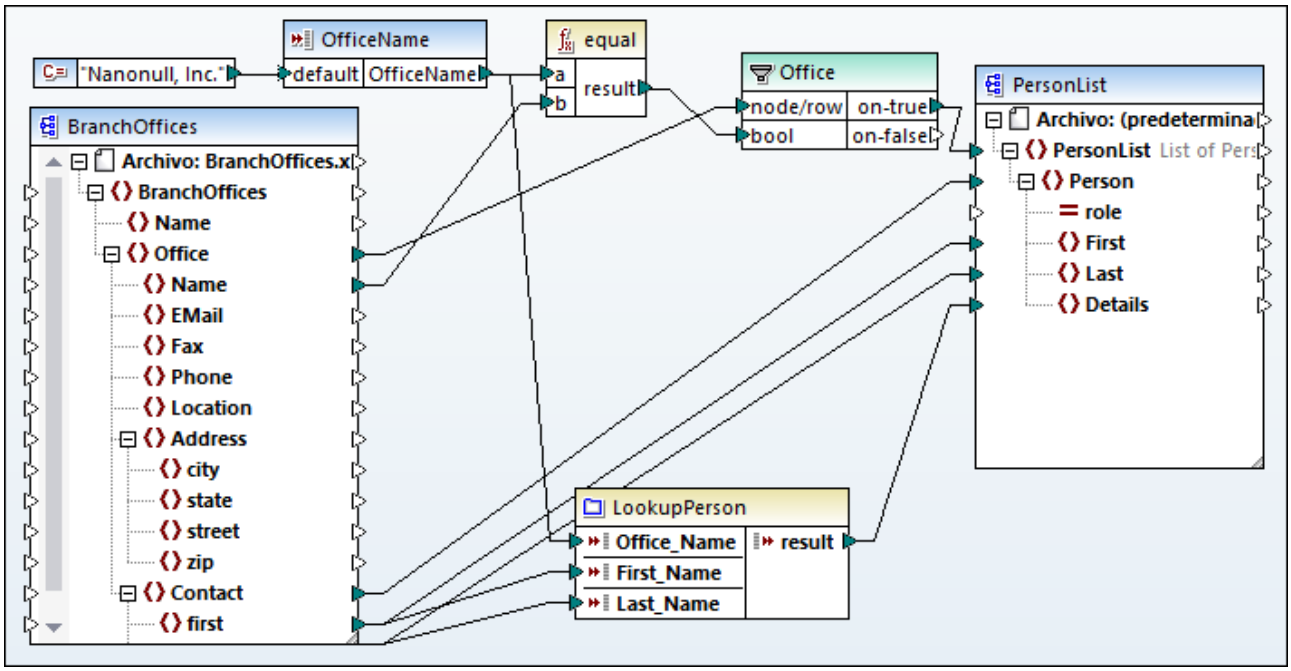
Los elementos de una asignación son estructuras jerárquicas que pueden contener muchos niveles de profundidad. Por otro lado, una asignación puede tener varios elementos de entrada y componentes, además de otros componentes intermedios como funciones, filtros, pares valor-asignación, etc. Esto añade complejidad al algoritmo de la asignación, especialmente si se conectan varios componentes no relacionados. Para poder ejecutar una asignación por partes, es decir, un paso cada vez, se debe estableces un contexto actual para cada conexión.

También se podría decir que se establecen varios "contextos actuales" para la duración de la ejecución de la asignación, ya que el contexto actual va cambiando según se van procesando las conexiones.

MapForce siempre establece el contexto actual empezando por el (*nodo*) *elemento raíz de destino*. Ese es el punto en el que empieza la ejecución de la asignación. La conexión con el elemento raíz de destino se traza de vuelta a todos los elementos de origen que están directa o indirectamente conectados a él, también mediante funciones u otros componentes intermedios. Todos los elementos de origen y los resultados producidos por las funciones se añaden al contexto actual.

Una vez se ha terminado de procesar el nodo de destino, MapForce empieza a avanzar por la estructura jerárquica. Más concretamente, procesa todos los *elementos asignados* del componente de destino de arriba a abajo. Por cada elemento nuevo se establece un contexto nuevo que inicialmente contiene todos los elementos del contexto matriz.Por lo tanto, todos los elementos asignados del mismo nivel que haya en un componente de destino son independientes unos de otros pero tienen acceso a todos los datos de origen de los elementos correspondientes de nivel superior.

Ahora vamos a ver cómo funciona en la práctica lo que acabamos de explicar; para ello usaremos la asignación de ejemplo **PersonListByBranchOffice.mfd**, que puede encontrar en el directorio **<Documentos>\Altova\MapForce2023\MapForceExamples**.



En la asignación anterior, tanto el componente de origen como el de destino son archivos XML. El archivo XML de origen contiene dos elementos **Office**.

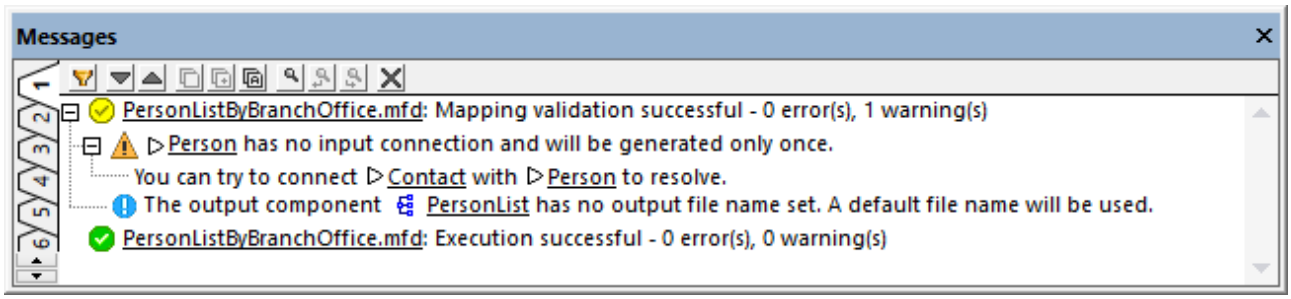
Como hemos mencionado previamente, la ejecución de la asignación siempre empieza por el nodo raíz de destino (**PersonList** en este ejemplo). Al trazar de vuelta la conexión (con el filtro y la función) a un elemento de origen, se puede concluir que el elemento de origen es **Office**. (La otra ruta de conexión lleva a un parámetro de entrada cuyo propósito se explica más abajo).

Si hubiera habido una conexión directa entre **Office** y **PersonList**, entonces, conforme a la regla general de asignación, la asignación habría creado tantos elementos de instancia `PersonList` como elementos **Office** hay en el archivo de origen. Sin embargo, esto no es lo que ocurre aquí porque hay un filtro entre medias. El filtro solo pasa al componente de destino datos que cumplen con la condición booleana que hay conectada al elemento de entrada `bool` del filtro. La función `equal` devuelve `true` si el nombre de la oficina es igual a "Nanonull, Inc.". Esta condición se cumple una sola vez porque solo existe un nombre así en el archivo XML de origen.

En consecuencia, la conexión entre **Office** y **PersonList** define una única oficina como el contexto para todo el documento de destino. Esto significa que todos los elementos secundarios de **Personlist** tienen acceso a los datos de la oficina "Nanonull, Inc." y que no existe ninguna otra oficina en el contexto actual.

La siguiente conexión es entre **Contact** y **Person**. Según la regla general de asignación se crea un elemento **Person** de destino por cada elemento **Contact** de origen. En cada iteración esta conexión establece un contexto actual nuevo; por lo tanto, las conexiones secundarias (**first** a **First**, **last** a **Last**) suministran datos del elemento de origen al de destino en el contexto de cada elemento **Person**.

Si omite la conexión entre **Contact** y **Person**, entonces la asignación crearía solamente un elemento **Person** con varios nodos **First**, **Last** y **Details**. En estos casos, MapForce emite un mensaje de advertencia y una sugerencia en la ventana Mensajes, por ejemplo:



Finalmente, la asignación incluye una función definida por el usuario, `LookupPerson`. La función definida por el usuario también se ejecuta en el contexto de cada elemento **Person** debido a la conexión primaria entre **Contact** y **Person**. Más concretamente, cada vez que un elemento **Person** nuevo se crea en el lado de destino, se llama a la función para que rellene el elemento **Details** de esa persona. Esta función toma tres parámetros de entrada. El primero (**OfficeName**) está definido para que lea datos del parámetro de entrada de la asignación. Los datos de origen para este parámetro también los podría suministrar el elemento de origen **Name** sin que eso cambiara el resultado de la asignación. En ambos casos el valor de origen es el mismo y se toma de un contexto de nivel superior. A nivel interno, la función de búsqueda concatena los valores recibidos como argumentos y produce un valor único. Para más información sobre cómo funciona la función `LookupPerson` consulte Ejemplo: búsqueda y concatenación 589.

7.3.2.1 Bases de datos

Para mejorar la eficiencia y reducir el uso de recursos físicos o de red, lo normal es querer evitar llamar varias veces a una misma base de datos en la misma asignación sin que esto sea necesario. Puede que sigan existiendo situaciones en las que simplemente no podrá evitar que se llame varias veces a una misma base de datos debido a la naturaleza de esa asignación, pero tenga en cuenta las siguientes consideraciones:

- Si solo necesita una llamada a la BD, evite colocar el componente de BD en un contexto de nivel superior que requiera llamar a la BD varias veces. Esto podría ocurrir, por ejemplo, si añade un componente de BD dentro de una función definida por el usuario que recibe una secuencia de valores como entrada y a la que, por tanto, se llama una vez por cada elemento de esa secuencia (*véase también* *Funciones definidas por el usuario* 876). Las variables suelen ser útiles para obtener datos dentro del mismo contexto antes de pasarlo al componente de destino.
- Si necesita sumar valores de una BD (por ejemplo, para contar el número de registros con la función `count`), se recomienda que conecte el resultado de la función agregada a una variable en la que **compute-when=once**. Esto evita llamadas repetitivas a la BD, como se describe en Ejemplo: contar filas de tabla de BD 500.
- Intente extraer todos los datos de BD en una llamada (por ejemplo, una instrucción SQL-SELECT o un procedimiento almacenado) en vez de añadir varias veces el mismo componente de BD a la asignación.
- Si necesita extraer datos de varias tablas o vistas de la misma base de datos, es recomendable que use un componente Join (en modo SQL) o una instrucción SQL-SELECT. Esta última es lo más conveniente si prefiere escribir la instrucción SQL-SELECT usted mismo. Si necesita combinar datos de BD con otros datos que no sean de BD o con datos de distintas BD, es preferible usar uniones no SQL.
- Para optimizar la ejecución de uniones no SQL en asignaciones con muchos datos, ejecute las asignaciones con MapForce Server Advanced Edition.
- Si necesita filtrar datos de una BD, es más eficiente usar un componente SQL-WHERE en vez de un filtro estándar, ya que en la gramática de la BD correspondiente el componente anterior está optimizado para que funciones específicamente con BD.

7.3.2.2 Funciones definidas por el usuario

Las funciones definidas por el usuario son funciones personalizadas incrustadas en la asignación, donde se definen los componente de entrada y de salida, así como la lógica de procesamiento. Cada función definida por el usuario puede contener los mismos tipos de componentes que una asignación principal, incluidos servicios web y bases de datos.

> Por defecto, si una función definida por el usuario contiene una BD o un componente de servicio web y si los datos de entrada a esa función son una secuencia de varios valores, entonces cada valor de entrada llamará a la función, lo que resultará en una llamada a la BD o al servicio web.

Este comportamiento puede ser aceptable para asignaciones en las que realmente necesita que se llame a la función definida por el usuario *tantas veces como valores de entrada haya*, para las que no existe otra alternativa.

Si no quiere que ocurra eso, puede configurar la función definida por el usuario para que solo se la llame una vez, incluso aunque el componente de entrada sea una secuencia de valores. Tiene sentido usar esta opción

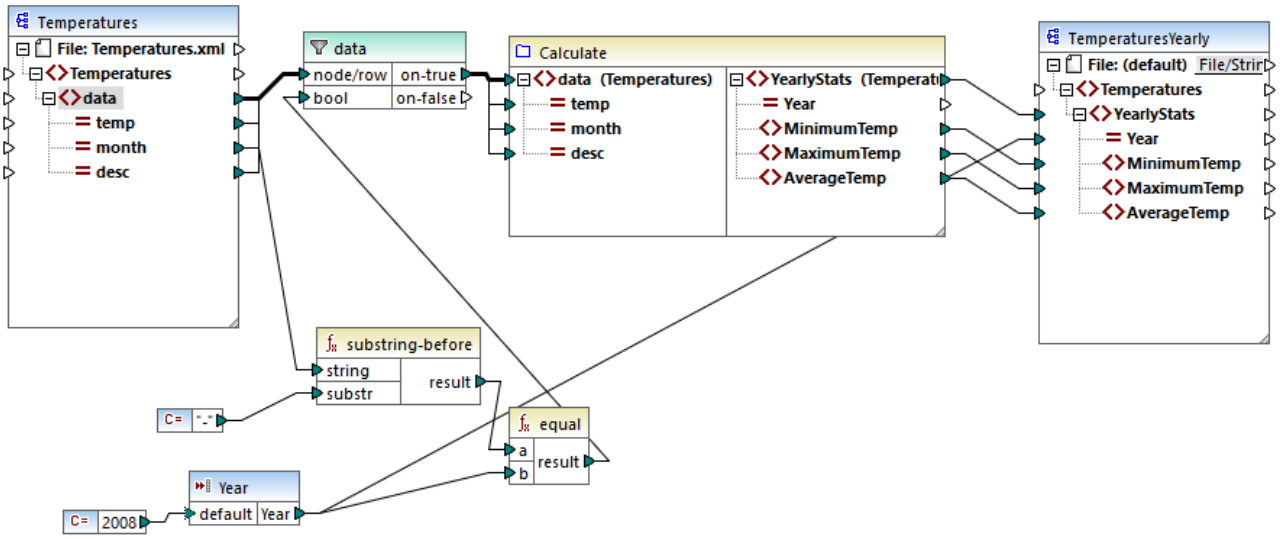
 © 2016-2022 Altova GmbH

en el caso de aquellas funciones definidas por el usuario que operan en un conjunto de valores antes de devolver un resultado (como las funciones que calculan valores medios o totales).

Se pueden configurar funciones definidas por el usuario para que acepten varios valores de entrada en la misma llamada a la función si esta es de tipo "regular" y no "inline". (*Para más detalles consulte* *Funciones definidas por el usuario* 575). Para indicar en funciones regulares que el parámetro de entrada es una secuencia marque la casilla *El parámetro de entrada es una secuencia*. Esta casilla se hace visible en la configuración del componente después de hacer doble clic en la barra de título de un parámetro de entrada. La casilla afecta la cantidad de llamadas que se hacen a la función:

- Cuando los datos de entrada están conectados al parámetro **sequence** se llama *solo una vez* a la función definida por el usuario y se le pasa la secuencia completa.
- Cuando los datos de entrada están conectados a un parámetro **non-sequence**, se llama a la función definida por el usuario *una vez por cada elemento de la secuencia*.

Para ver un ejemplo abra esta asignación: **<Documentos>\Altova\MapForce2023\MapForceExamples\\InputIsSequence.mfd**.



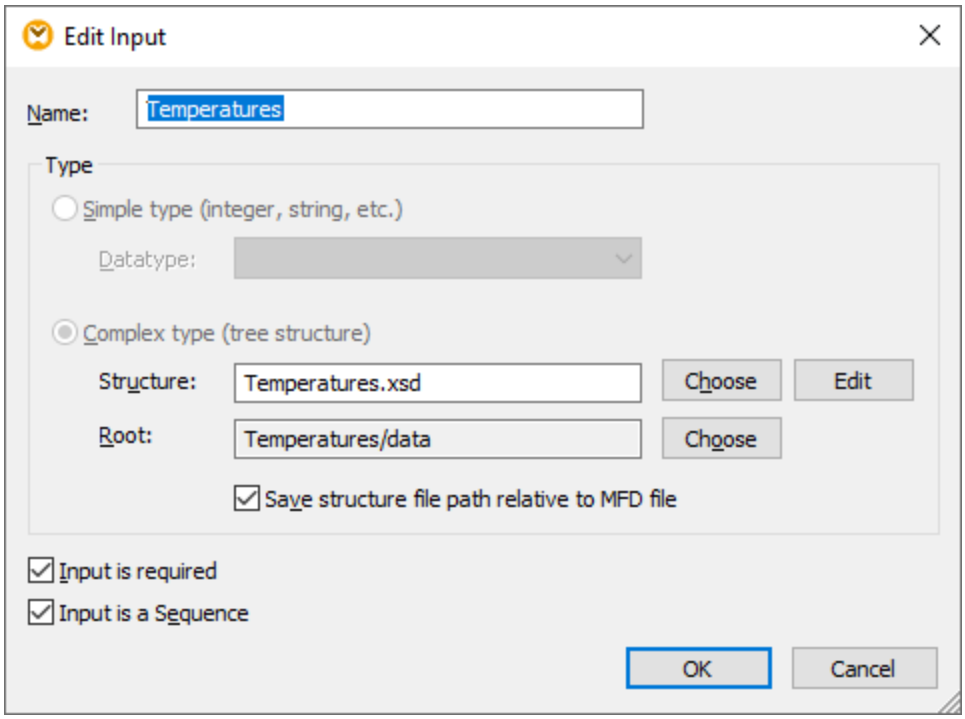
La asignación anterior ilustra un caso típico de una función definida por el usuario que opera en un conjunto de valores y, por tanto, necesita todos los valores de entrada en una sola llamada. Más concretamente, la función definida por el usuario `Calculate` devuelve las temperaturas mínima, máxima y media, y toma sus datos de entrada de un archivo XML de origen. El resultado esperado de esta asignación es:

```
<Temperatures>
    <YearlyStats Year="2008">
        <MinimumTemp>-0.5</MinimumTemp>
        <MaximumTemp>24</MaximumTemp>
        <AverageTemp>11.6</AverageTemp>
    </YearlyStats>
</Temperatures>
```

Como es habitual, la ejecución de la asignación empieza por el primer elemento del componente de destino (**YearlyStats** en este ejemplo). Para rellenar este nodo, la asignación intenta obtener datos de entrada de la

función definida por el usuario, lo que a su vez desencadena el filtro. El papel del filtro en esta asignación es pasar a la función definida por el usuario solamente temperaturas del año 2008.

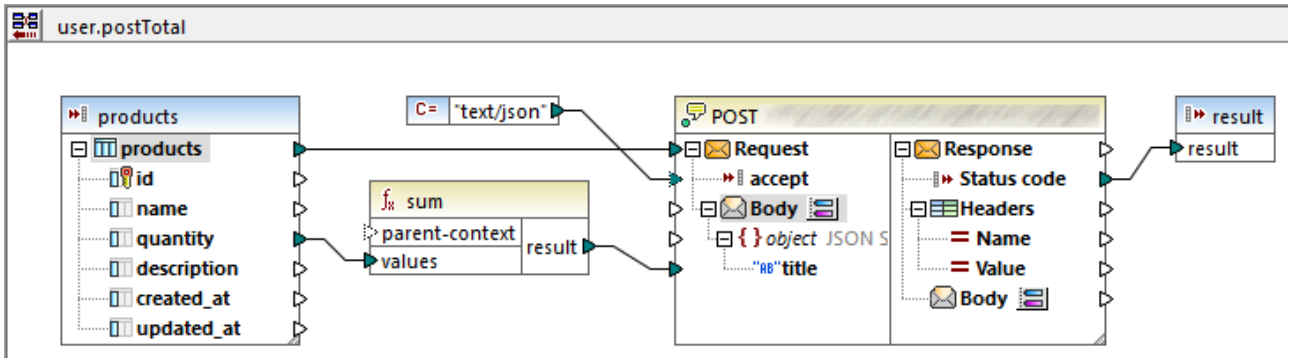
La casilla *El parámetro de entrada es una secuencia* se marcó para el parámetro de entrada de la función definida por el usuario (para ver esta casilla, haga clic en la barra del título de la función `Calculate` para introducir la asignación de la función y después haga doble clic en la barra del título del parámetro de entrada). Como hemos mencionado antes, marcar la casilla *El parámetro de entrada es una secuencia* hace que se suministre la secuencia completa de valores como parámetro de entrada de la función y que se llame a la función una sola vez.



Si la casilla *El parámetro de entrada es una secuencia* no se hubiera marcado, se habría llamado a la función definida por el usuario una vez por cada valor en el componente de origen. Como resultado, se habría calculado el valor máximo, mínimo y medio para cada valor individualmente y se habría producido un resultado incorrecto.

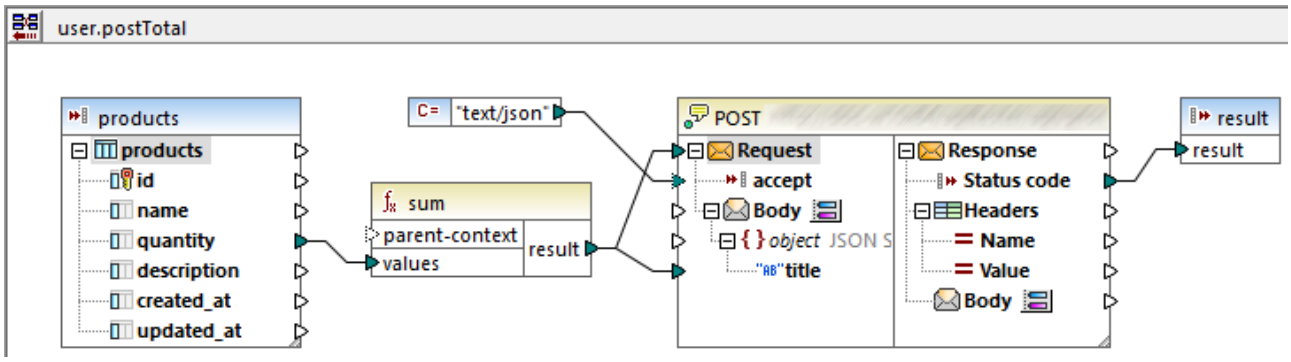
Al aplicar la misma lógica en funciones definidas por el usuario más complejas que incluyen llamadas a BD o a servicios web, es posible optimizar la ejecución y evitar llamadas innecesarias a la BD o al servicio web. Sin embargo, tenga en cuenta que la casilla *El parámetro de entrada es una secuencia* no controla qué le ocurre a la secuencia de valores *después* de que entre en la función. En otras palabras, nada le impide conectar la secuencia de valores de entrada al componente de entrada de un servicio web y llamarlo varias veces. Considere este ejemplo:

 © 2016-2022 Altova GmbH



La función definida por el usuario de la imagen anterior recibe una secuencia de valores de la asignación externa. Más concretamente, los datos suministrados al parámetro de origen provienen de una BD. Se ha marcado la casilla *El parámetro de entrada es una secuencia* para el parámetro de origen, por lo que se suministra a la función toda la secuencia en una sola llamada. La función debe sumar varios valores **quantity** y enviar el resultado en un servicio web. Se espera exactamente una llamada al servicio web. Sin embargo, cuando se ejecute la asignación se harán varias llamadas al servicio web. El motivo es que el componente de entrada **Request** del servicio web recibe una *secuencia de valores* y no un único valor.

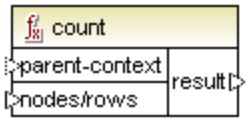
Para solucionar este problema, conecte el componente de entrada **Request** del servicio web al resultado de la función `sum`. La función produce un solo valor, por lo que también se llama al servicio web una sola vez:



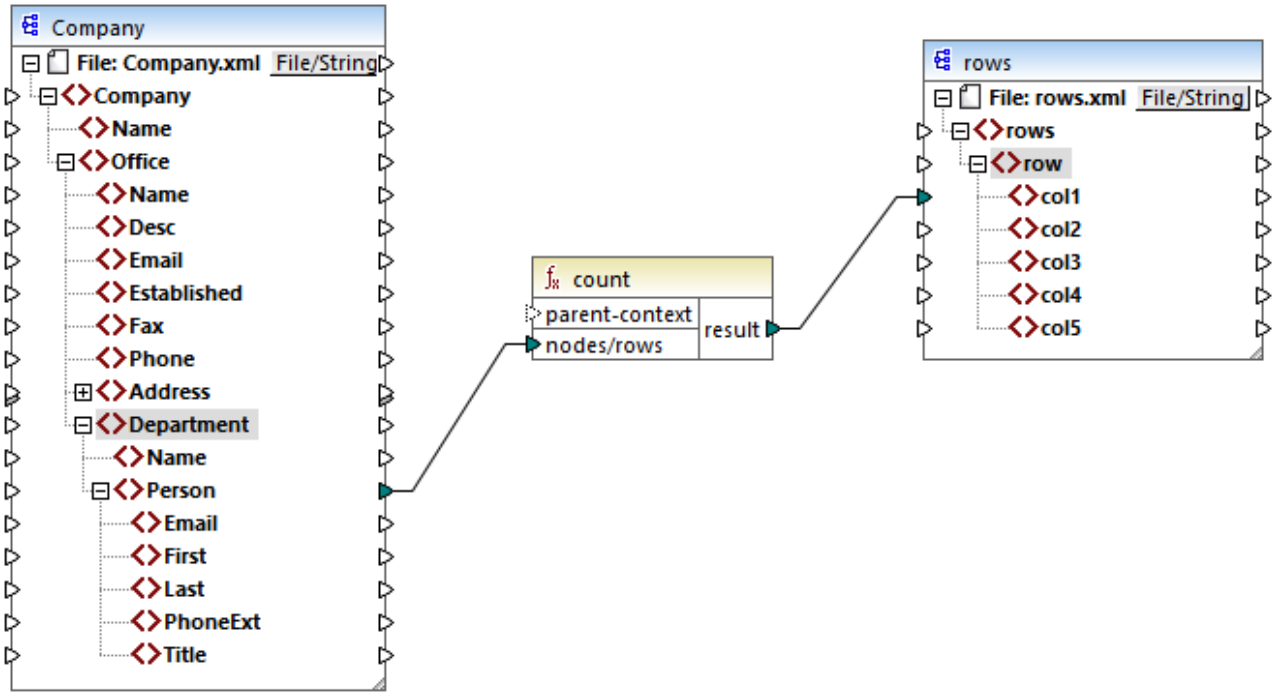
Normalmente, las funciones agregadas como **`sum`**, **`count`**, etc. producen un único valor. Sin embargo, si hay una conexión de nivel superior que lo permita, puede que también produzcan una secuencia de valores, como se describe en más detalle en Ejemplo: cambiar el contexto matriz 879.

7.3.2.3 Ejemplo: cambiar el contexto matriz

Algunos componentes de asignación tienen un elemento **parent-context** opcional. Con ayuda de este elemento puede influir en el contexto de la asignación en el que debe operar ese componente y, por tanto, cambiar el resultado de la asignación. Los componentes que tienen un elemento **parent-context** opcional son: funciones agregadas, variables y componentes `Join`.



Para ver un ejemplo de la utilidad de cambiar el contexto matriz, abra esta asignación: **<Documentos>\Altova\MapForce2023\MapForceExamples\Tutorial\ParentContext.mfd**.



En el XML de origen de la asignación anterior existe un único nodo Company que contiene dos nodos **Office**. Cada nodo **Office** contiene varios nodos **Department**, cada uno de los cuales contiene varios nodos **Person**. Si abre el archivo XML en un editor XML puede ver que la distribución de personas por oficina y departamento es esta:

Oficina	Departmento	Número de personas
Nanonull, Inc.	Administración	3
	Marketing	2
	Ingeniería	6
	IT & Soporte técnico	4
Nanonull Partners, Inc.	Administración	2
	Marketing	1
	IT & Soporte técnico	3

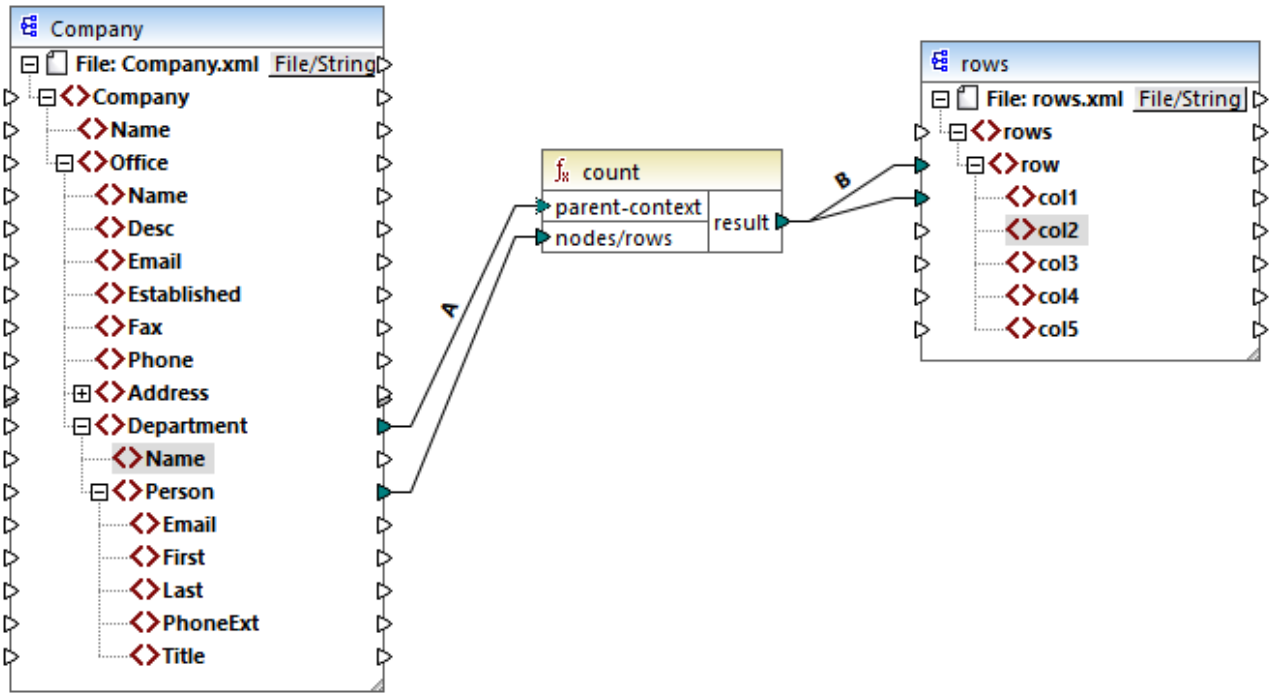
 © 2016-2022 Altova GmbH

La asignación cuenta todas las personas de todos los departamentos. Para ello usa la función `count` de la biblioteca `core`. Si hace clic en la pestaña Resultados para previsualizar la asignación verá que produce un único valor, **21**, que corresponde al número total de personas del archivo XML.

La asignación funciona así:

- Como es habitual, la asignación empieza por el nodo de nivel superior del componente de destino (**rows** en este ejemplo). No hay ninguna conexión entrante para **rows**. Como resultado, el contexto implícito de asignación está establecido entre **Company** (elemento de nivel superior del componente de origen) y **rows** (elemento superior del componente de destino).
- El resultado de la función es un valor único porque solo hay un elemento **Company** en el archivo de origen.
- Para rellenar el elemento de destino **col1**, MapForce ejecuta la función `count` en el contexto matriz implícito que se menciona más arriba, por lo que contará todos los nodos **Person** de todas las oficinas de todos los departamentos.

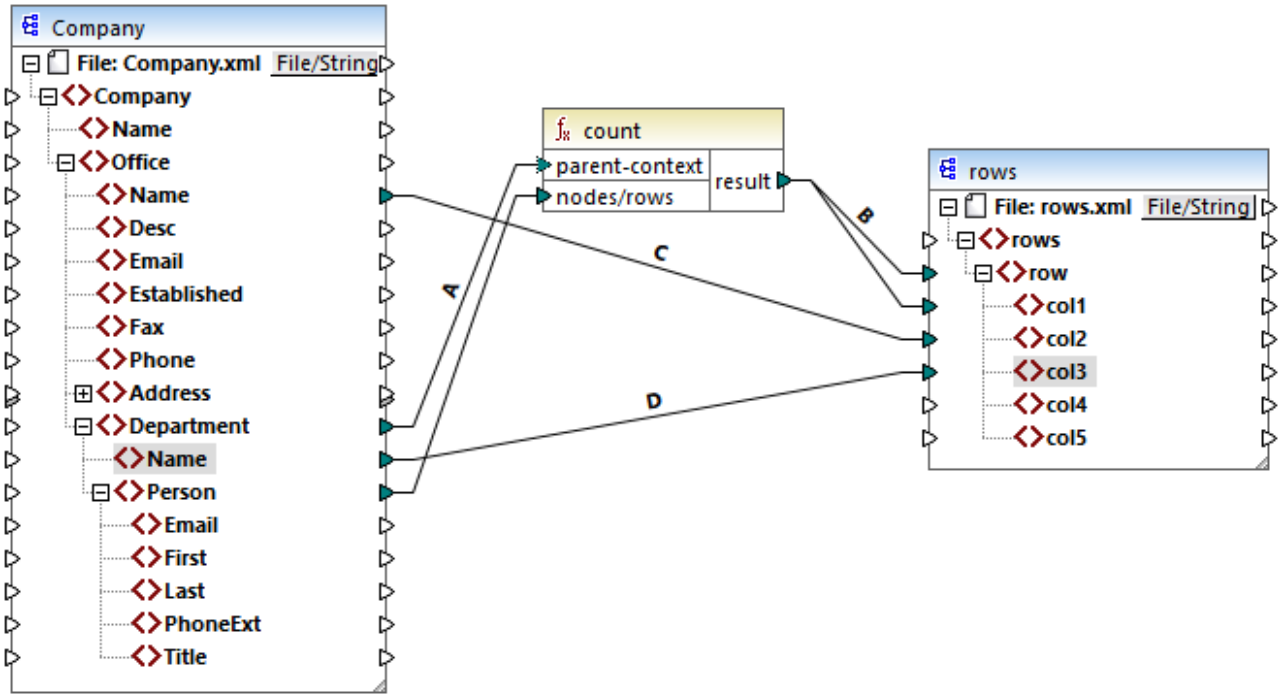
El argumento **parent-context** de la función permite cambiar el contexto de la asignación. Esto permite, por ejemplo, contar el número de personas de cada departamento. Para ello, trace dos o más conexiones como las de la imagen siguiente:



En la asignación anterior la conexión A cambia el contexto matriz de la función `count` a **Department**. Como resultado, la función contará el número de personas de cada departamento. Es importante recordar que la función ahora devolverá una *secuencia* de resultados en lugar de un solo resultado debido a que en el origen existen varios departamentos. Esta es la razón por la que existe la conexión B: por cada elemento de la secuencia resultante crea una fila nueva en el archivo de destino. El resultado de la asignación ahora cambia en consecuencia (tenga en cuenta que los números corresponden exactamente al número de personas de cada departamento):

```
<rows>
    <row>
        <col1>3</col1>
    </row>
    <row>
        <col1>2</col1>
    </row>
    <row>
        <col1>6</col1>
    </row>
    <row>
        <col1>4</col1>
    </row>
    <row>
        <col1>2</col1>
    </row>
    <row>
        <col1>1</col1>
    </row>
    <row>
        <col1>3</col1>
    </row>
</rows>
```

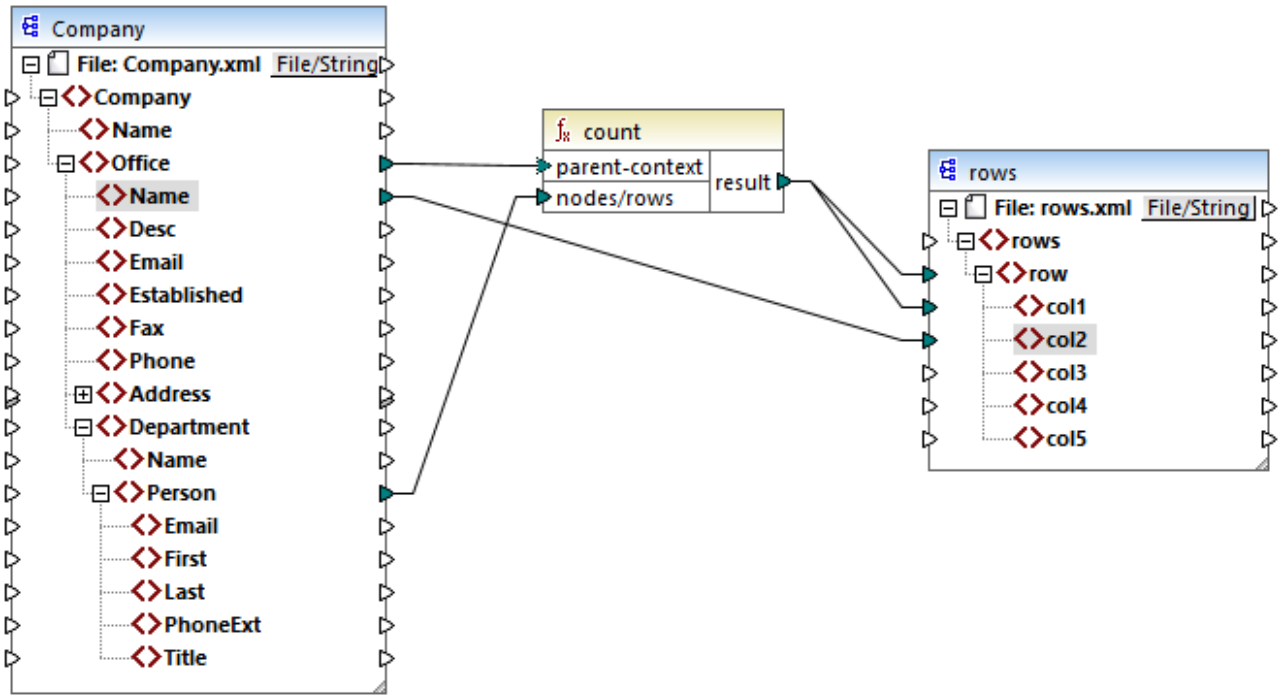
Si tenemos en cuenta que la asignación actual crea una fila por cada departamento, puede optar por copiar también el nombre de la oficina y el del departamento en el archivo de destino; para ello debe trazar las conexiones C y D:



 © 2016-2022 Altova GmbH

De este modo, el resultado no solo mostrará el número de personas, sino también los nombres de oficina y departamento correspondientes.

Si quiere contar el nombre de personas de cada oficina, conecte el contexto matriz de la función `count` al elemento **Office** del componente de destino.



Con las conexiones de la imagen anterior, la función `count` devuelve un resultado por cada oficina. En el archivo de origen existen dos oficinas, por lo que la función ahora devolverá dos secuencias. En consecuencia, en el resultado existen dos filas de las que cada una corresponde al número de personas de esa oficina.

```
<rows>
    <row>
        <col1>15</col1>
        <col2>Nanonull, Inc.</col2>
    </row>
    <row>
        <col1>6</col1>
        <col2>Nanonull Partners, Inc.</col2>
    </row>
</rows>
```

7.3.3 Contexto de prioridad

El contexto de prioridad es una forma de influir en el orden en que se evalúan los parámetros de entrada de una función. Puede que sea necesario establecer un contexto de prioridad si su asignación combina datos de dos orígenes no relacionados.

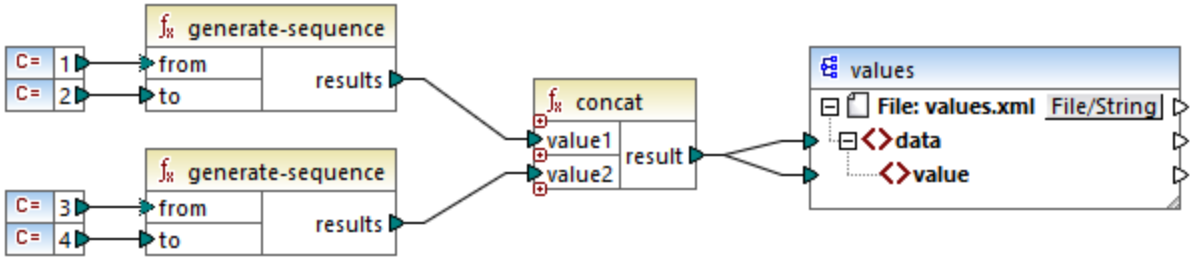
Para entender cómo funciona el contexto de prioridad, recuerde que cuando se ejecuta una asignación la conexión a un elemento de entrada puede levar una secuencia de varios valores. Para las secuencias con dos parámetros de entrada esto significa que MapForce debe crear dos bucles, uno de los cuales se debe procesar primero. El bucle que se procesa primero es el bucle "exterior". Por ejemplo, la función `equal` recibe dos parámetros: *a* y *b*. Si ambos obtienen una secuencia de valores, entonces MapForce procesa así:

- Por cada ocurrencia de *a*
  - Por cada ocurrencia de *b*
    - ¿Es *a* igual a *b*?

Como puede ver en el ejemplo anterior, cada ocurrencia de *b* se evalúa en el contexto de cada ocurrencia de *a*. El contexto de prioridad permite alterar el proceso lógico para que cada ocurrencia de *a* se evalúe en el contexto de cada ocurrencia de *b*. En otras palabras, permite cambiar el bucle interno por el externo:

- Por cada ocurrencia de *b*
  - Por cada ocurrencia de *a*
    - ¿Es *a* igual a *b*?

Ahora examinemos una asignación en la que el contexto de prioridad afecta al resultado de la asignación. En la asignación siguiente la función `concat` tiene dos parámetros de entrada. Cada parámetro de entrada es una secuencia que se genera con ayuda de la función `generate-sequence`. La primera secuencia es "1,2" y la segunda es "3,4".



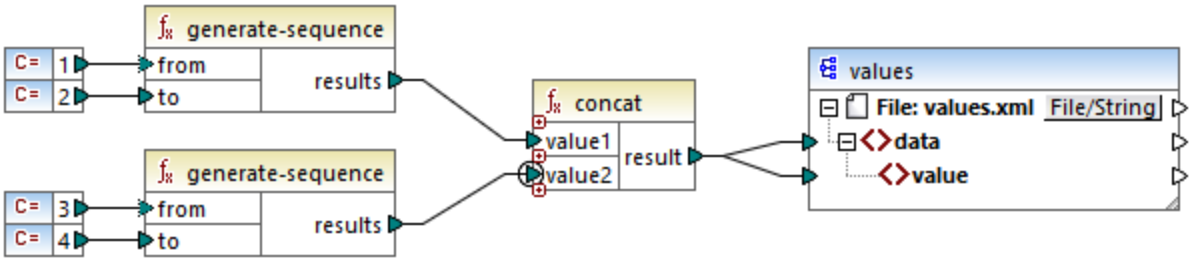
Primero, ejecutemos la asignación sin un contexto de prioridad. La función `concat` empieza por evaluar la secuencia de nivel superior, de forma que combine valores en este orden:

- 1 con 3
- 1 con 4
- 2 con 3
- 2 con 4

Esto también se refleja en el resultado de la asignación:

```
<data>
    <value>13</value>
    <value>14</value>
    <value>23</value>
    <value>24</value>
</data>
```

Si hace clic con el botón derecho en el segundo parámetro de entrada y selecciona **Contexto de prioridad** en el menú contextual, este se convierte en el contexto de prioridad, En la imagen siguiente se ve el contexto de prioridad de entrada rodeado por un círculo.



Esta vez se evalúa el segundo parámetro de entrada primero. La función `concat` sigue concatenando los mismos valores, pero esta vez procesa primero la secuencia "3,4" y el resultado cambia a:

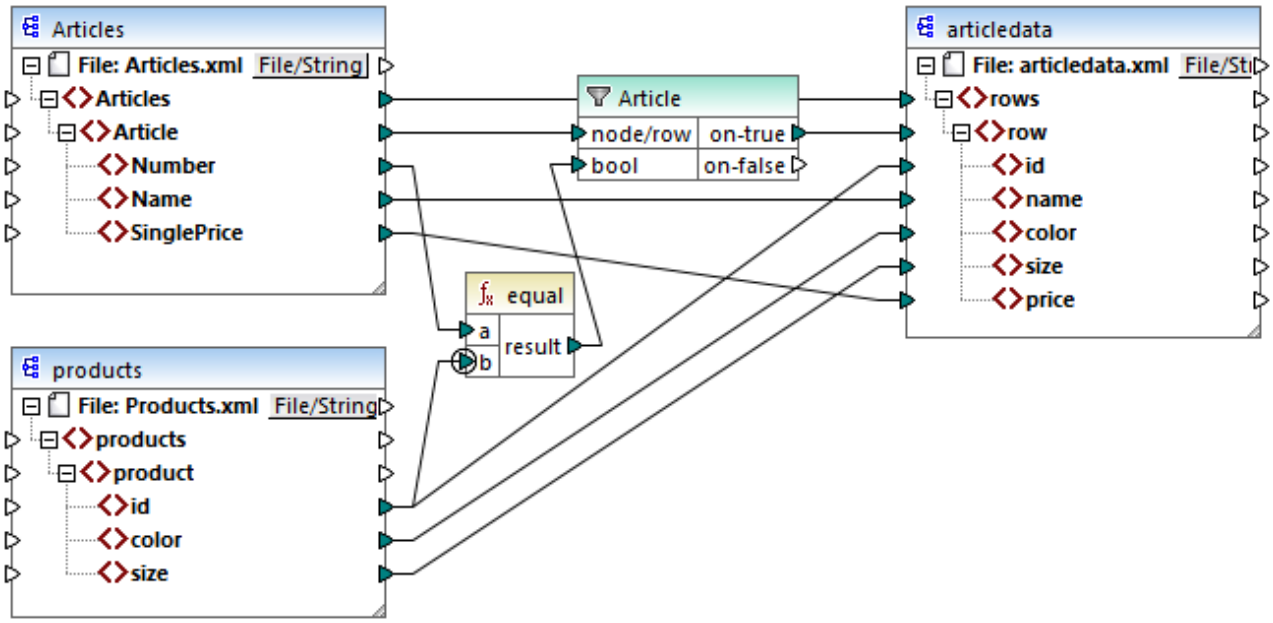
```
<data>
    <value>13</value>
    <value>23</value>
    <value>14</value>
    <value>24</value>
</data>
```

Hasta ahora solo hemos visto la parte teórica del contexto de prioridad. Para ver un ejemplo práctico consulte Example: Filter with priority context 885.

7.3.3.1 Ejemplo: filtrar con el contexto de prioridad

Cuando se conecta una función a un filtro, el contexto de prioridad no solo afecta a la función en sí, sino también a la evaluación del filtro. La asignación de la imagen siguiente muestra un caso típico en el que es necesario usar el contexto de prioridad para obtener el resultado correcto. Puede encontrar esta asignación en la ruta: **<Documentos>\Altova\MapForce2023\MapForceExamples\Tutorial\FilterWithPriority.mfd**.

Nota: esta asignación usa componentes XML, pero otros tipos de componentes de MapForce obedecen a la misma lógica, incluidos EDI, JSON, etc. Para las bases de datos se recomienda filtrar con componentes SQL WHERE 352 en lugar de filtros estándar.



El objetivo de esta asignación es copiar datos de personas de **Articles.xml** en un archivo XML nuevo con un esquema distinto, **articledata.xml**. Al mismo tiempo, la asignación debería buscar los detalles de cada artículo en el archivo **Products.xml** y combinarlos con el registro de archivo correspondiente. Observe que cada registro de **Articles.xml** tiene un elemento **Numbers** y cada registro en **Products.xml** tiene un elemento **id**. Si estos dos valores son iguales, entonces todos los otros valores () se deben copiar a la misma fila en el destino.

Para conseguir este objetivo se ha añadido un filtro. Cada filtro requiere una condición booleana como entrada; solamente aquellos nodos/filas que cumplan con esa condición se copian en el destino, para lo que se ha incluido una función `equal` a la asignación. La función `equal` comprueba si el número de artículo y el ID del producto son iguales en ambos orígenes. El resultado se suministra como entrada para el filtro. Si es **true**, entonces el elemento **Article** se copia en el destino.

Observe que se ha definido un contexto de prioridad en el segundo parámetro de entrada de la segunda función `equal`. En esta asignación no incluir el contexto de prioridad en la asignación resultaría en un resultado incorrecto.

Asignación inicial: sin contexto de prioridad

Esta es la lógica de la asignación sin contexto de prioridad:

- Conforme a la regla general de asignación, por cada elemento **Article** que cumpla con la condición del filtro se crea una fila nueva en el destino. La conexión entre Article y row (mediante la función y el filtro) se encarga de ello.
- El filtro comprueba la condición de cada artículo. Para ello, recorre todos los productos y trae varios productos al contexto actual.
- Para rellenar el **id** en el lado de destino, MapForce sigue la regla general (por cada elemento de origen se crea uno de destino). Sin embargo, como hemos explicado más arriba, todos los productos de **Products.xml** están en el contexto actual. No hay ninguna conexión entre **product** y otro lugar del

 © 2016-2022 Altova GmbH

destino de forma que se lea el **id** de un producto en concreto. En consecuencia, se crearán varios elementos id por cada **Article** en el destino. Lo mismo ocurre con **color** y **size**.

En resumen: los elementos de **Products.xml** tienen el contexto del filtro (que debe recorrer todos los productos); por tanto, los valores **id**, **color** y **size** se copiarán en cada fila de destino tantas veces como productos haya en el archivo de origen, lo que generará un resultado incorrecto, como este:

```
<rows>
    <row>
        <id>1</id>
        <id>2</id>
        <id>3</id>
        <name>T-Shirt</name>
        <color>red</color>
        <color>blue</color>
        <color>green</color>
        <size>10</size>
        <size>20</size>
        <size>30</size>
        <price>25</price>
    </row>
</rows>
```

Solución A: use contexto de prioridad

El problema anterior se soluciona añadiendo un contexto de prioridad a la función que computa la condición booleana del filtro.

En concreto, si el segundo parámetro de entrada de la función `equal` se designa como contexto de prioridad, la secuencia proveniente de **Products.xml** se prioriza. Esto se traduce en la siguiente lógica de asignación:

- Por cada producto se rellena la entrada **b** de la función `equal` (en otras palabras, se prioriza **b**). En este punto, los detalles del producto actual están en contexto.
- Por cada artículo se rellena la entrada **a** de la función `equal` y se comprueba si la condición del filtro es **true**. Si lo es, entonces los detalles del artículo también se añaden al contexto actual.
- A continuación, se copian los detalles del artículo y del producto desde el contexto actual a los elementos correspondientes del destino.

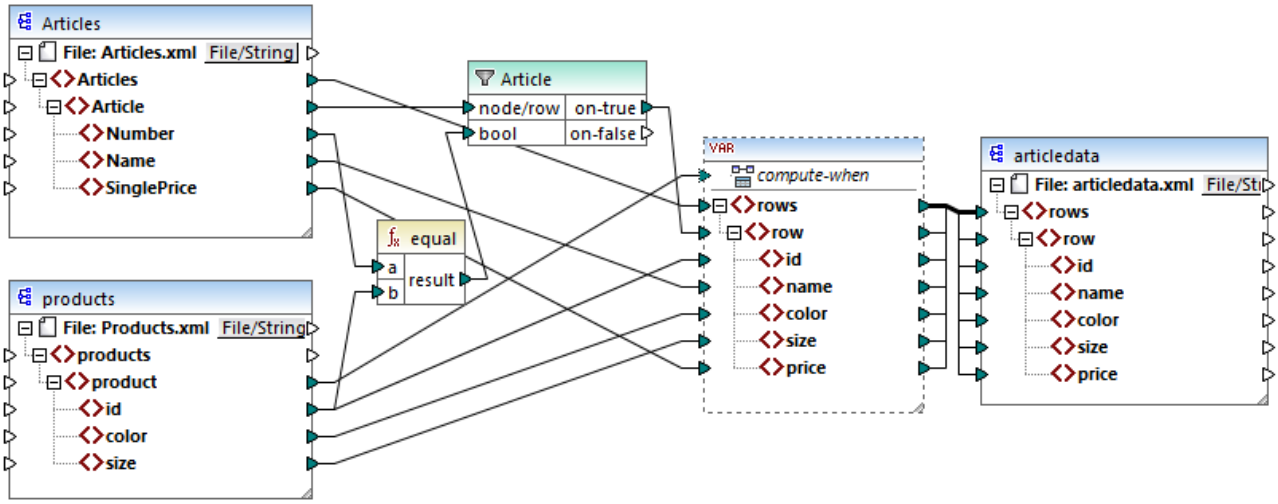
La lógica de asignación anterior produce un resultado correcto, por ejemplo:

```
<rows>
    <row>
        <id>1</id>
        <name>T-Shirt</name>
        <color>red</color>
        <size>10</size>
        <price>25</price>
    </row>
</rows>
```

Solución B: usar una variable

Una solución alternativa es llevar al mismo contexto a cada artículo y producto que cumpla con la condición del filtro, con ayuda de una variable intermedia. Las variables se pueden usar en escenarios como este porque permiten almacenar datos de forma temporal en la asignación, lo que permite cambiar el contexto según lo necesite.

Para escenarios como este puede añadir a la asignación una variable que tiene el mismo esquema que el componente de destino. En el menú **Insertar**, haga clic en **Insertar**, después en **Variable** y proporcione el esquema **articledata.xml** como estructura cuando la aplicación se lo pida.



En la asignación anterior ocurre lo siguiente:

- El contexto de prioridad ya no se usa. En su lugar hay una variable que tiene la misma estructura que el componente de destino.
- Como es habitual, la ejecución de la asignación empieza en el nodo raíz de destino. Antes de rellenar el destino, la asignación recopila datos en la variable.
- La variable se computa en el contexto de cada producto. Esto ocurre porque existe una conexión desde **product** hasta la entrada **compute-when** de la variable.
- La condición del filtro se comprueba en el contexto de cada producto. La estructura de la variable solo se rellena y se pasa al destino si se cumple esta condición.

7.3.4 Multiple target components

Una asignación puede tener varios componentes de origen y de destino. Cuando hay varios componentes de destino solo puede previsualizar el resultado de un componente cada vez en MapForce, que es el que indique con el botón **Vista previa** . En otros entornos de ejecución (MapForce Server o código generado), todos los componentes de destino se ejecutan de forma secuencial y se produce el resultado correspondiente de cada componente.

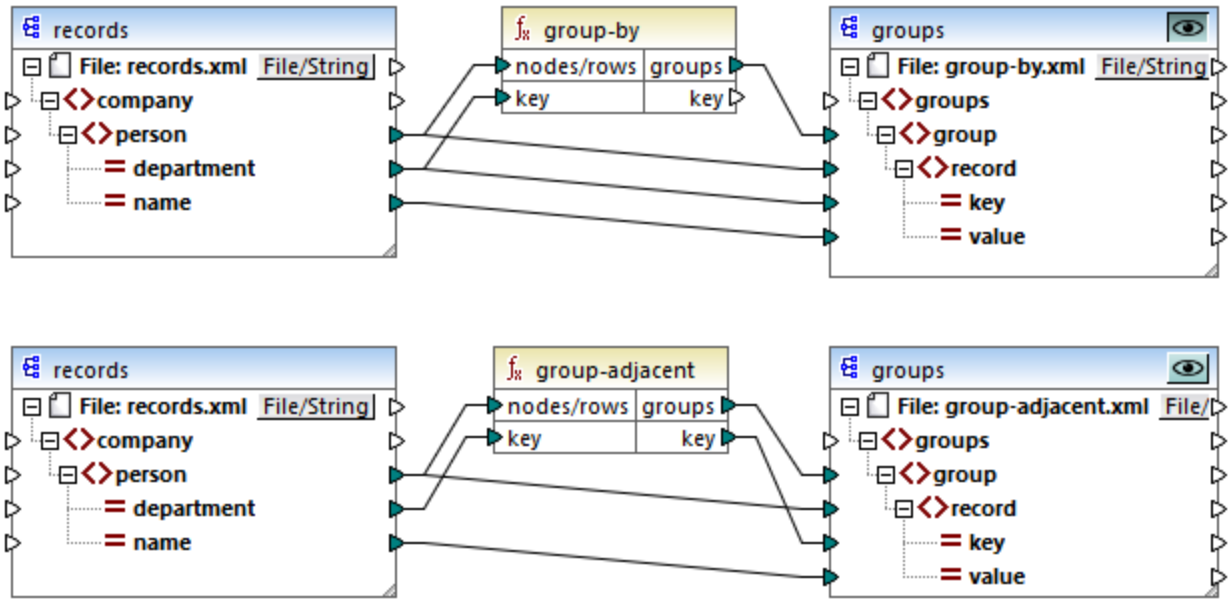
Por defecto, los componentes de destino se procesan de arriba a abajo y de izquierda a derecha. Si lo necesita, puede modificar este orden cambiando la posición de los componentes de destino en la ventana de

 © 2016-2022 Altova GmbH

asignación. El punto de referencia es la esquina superior izquierda de cada uno de los componentes. Tenga en cuenta lo siguiente:

- Si dos componentes tienen la misma posición vertical, entonces tiene precedencia el último.
- Si dos componentes tienen la misma posición horizontal, entonces tiene precedencia el de nivel superior.
- En el caso improbable de que varios componentes tengan la misma posición, entonces se usa automáticamente un ID de componente interno que garantiza un orden bien definido pero que no se puede modificar.

Para ver un ejemplo abra la asignación de ejemplo: **<Documentos>\Altova\MapForce2023\MapForceExamples\Tutorial\GroupingFunctions.mfd**. Esta asignación consiste en varios componentes de origen y varios de destino; en la imagen se ve solamente un fragmento.



Conforme a las reglas, la orden de procesamiento predeterminada de esta asignación en MapForce Server y en código generado es de arriba a abajo. Puede comprobar que es el caso al generar código XSLT 2.0, por ejemplo:

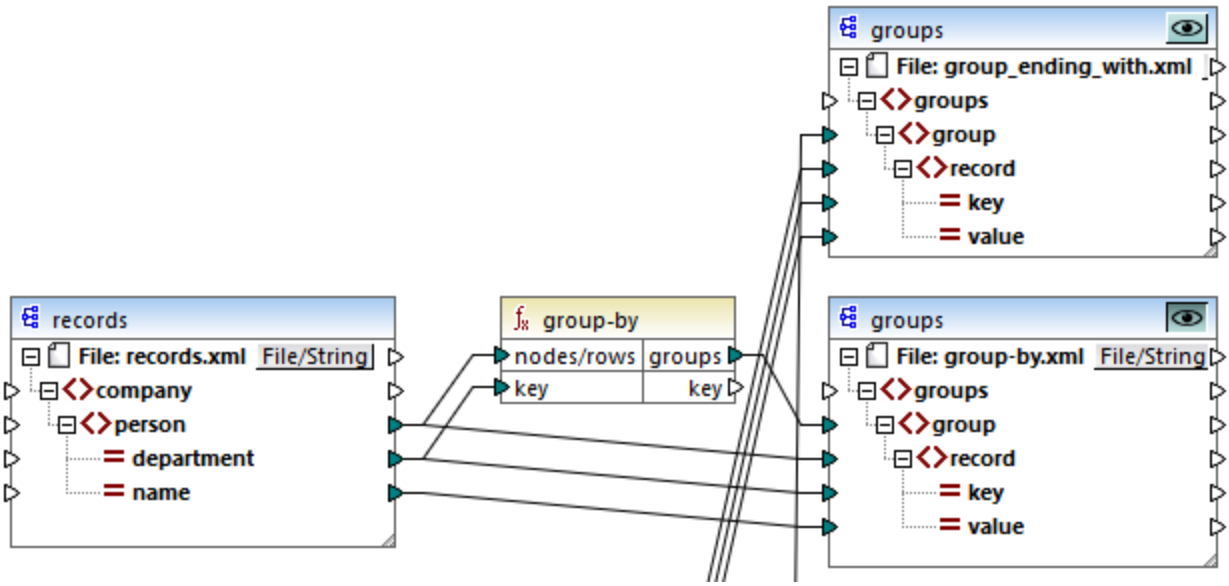
1. En el menú **Archivo**, haga clic en **Generar código en | XSLT 2.0**.
2. Cuando la aplicación lo pida, seleccione un directorio de destino para el código generado.

Una vez lo haya generado, el directorio de destino incluye varios archivos XSLT y un archivo **DoTransform.bat**. Este último se puede ejecutar con RaptorXML Server (que requiere una licencia aparte). El archivo **DoTransform.bat** procesa componentes en el mismo orden en el que se definieron en la asignación, de arriba a abajo. Esto se puede comprobar en el parámetro `--output` de cada transformación.

```
RaptorXML xslt --xslt-version=2 --input="records.xml" --output="group-by.xml" --xml-
validation-error-as-warning=true %* "MappingMapTogroups.xslt"
IF ERRORLEVEL 1 EXIT/B %ERRORLEVEL%
RaptorXML xslt --xslt-version=2 --input="records.xml" --output="group-adjacent.xml" --
```

```
xml-validation-error-as-warning=true %* "MappingMapTogroups2.xslt"
IF ERRORLEVEL 1 EXIT/B %ERRORLEVEL%
RaptorXML xslt --xslt-version=2 --input="records.xml" --output="group-into-blocks.xml" --
xml-validation-error-as-warning=true %* "MappingMapTogroups3.xslt"
IF ERRORLEVEL 1 EXIT/B %ERRORLEVEL%
RaptorXML xslt --xslt-version=2 --input="records-v2.xml" --output="group-starting-
with.xml" --xml-validation-error-as-warning=true %* "MappingMapTogroups4.xslt"
IF ERRORLEVEL 1 EXIT/B %ERRORLEVEL%
RaptorXML xslt --xslt-version=2 --input="records-v3.xml" --output="group_ending_with.xml"
--xml-validation-error-as-warning=true %* "MappingMapTogroups5.xslt"
IF ERRORLEVEL 1 EXIT/B %ERRORLEVEL%
```

La última transformación produce un archivo de salida llamado **group-ending-with.xml**. Ahora vamos a mover este componente de destino en la asignación a la parte de más arriba:



Si ahora vuelve a generar código XSLT 2.0, el orden de procesamiento también cambia:

```
RaptorXML xslt --xslt-version=2 --input="records-v3.xml" --output="group_ending_with.xml"
--xml-validation-error-as-warning=true %* "MappingMapTogroups.xslt"
IF ERRORLEVEL 1 EXIT/B %ERRORLEVEL%
RaptorXML xslt --xslt-version=2 --input="records.xml" --output="group-by.xml" --xml-
validation-error-as-warning=true %* "MappingMapTogroups2.xslt"
IF ERRORLEVEL 1 EXIT/B %ERRORLEVEL%
RaptorXML xslt --xslt-version=2 --input="records.xml" --output="group-adjacent.xml" --
xml-validation-error-as-warning=true %* "MappingMapTogroups3.xslt"
IF ERRORLEVEL 1 EXIT/B %ERRORLEVEL%
RaptorXML xslt --xslt-version=2 --input="records.xml" --output="group-into-blocks.xml" --
xml-validation-error-as-warning=true %* "MappingMapTogroups4.xslt"
IF ERRORLEVEL 1 EXIT/B %ERRORLEVEL%
RaptorXML xslt --xslt-version=2 --input="records-v2.xml" --output="group-starting-
with.xml" --xml-validation-error-as-warning=true %* "MappingMapTogroups5.xslt"
IF ERRORLEVEL 1 EXIT/B %ERRORLEVEL%
```


© 2016-2022 Altova GmbH

En el fragmento de código anterior, la primera llamada ahora produce **group-ending-with.xml**.

Puede cambiar el orden de procesamiento de forma parecida en otros lenguajes de programación y en los archivos de ejecución compilados de MapForce Server (.mfx).

Asignaciones encadenadas

Para las asignaciones encadenadas se sigue la misma secuencia de procesamiento que se describe más arriba. Sin embargo, el grupo de asignaciones encadenadas se entiende como una unidad. Cambiar la posición del componente intermedio o el componente final de una única asignación encadenada no afecta a la secuencia de procesamiento. La posición de los componentes finales de destino de cada grupo determinan qué se procesa primero solamente si existen varias "cadenas" o varios componentes de destino en una asignación.

- Si dos componentes de destino finales tienen la misma posición vertical, entonces tiene precedencia el último.
- Si dos componentes de destino finales tienen la misma posición horizontal, entonces tiene precedencia el de nivel superior.
- En el caso improbable de que varios componentes tengan exactamente la misma posición, entonces se usa automáticamente un ID de componente interno que garantiza un orden bien definido pero que no se puede modificar.

7.4 Procesar varios archivos de entrada o salida simultáneamente

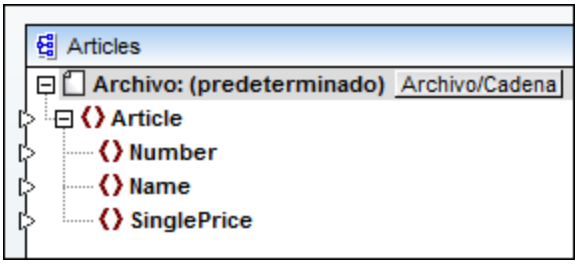
MapForce puede configurarse para procesar varios archivos simultáneamente (p. ej. todos los archivos de un directorio) cuando se ejecute la asignación. Gracias a esta característica podrá:

- Suministrar a la asignación una lista de archivos de entrada que se deben procesar.
- Generar como resultado de la asignación una lista de archivos en lugar de un solo archivo de salida.
- Generar una aplicación de asignación de datos donde el nombre de los archivos de entrada y salida se definen en tiempo de ejecución.
- Pasar un grupo de archivos a otro formato.
- Dividir un archivo (o base de datos) de gran tamaño en varios archivos más pequeños.
- Agrupar varios archivos diferentes en un solo archivo (o cargarlos en una base de datos).

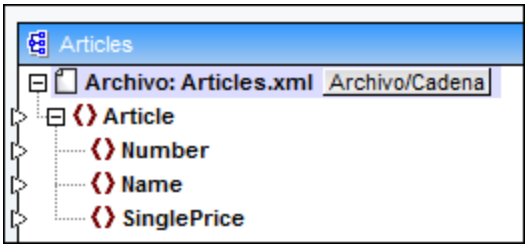
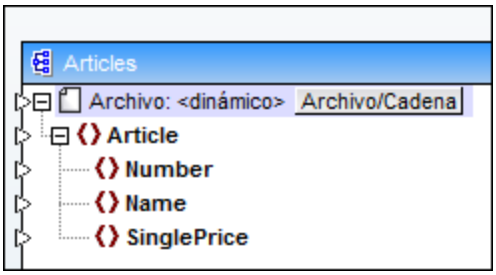
Hay dos maneras de configurar componentes de MapForce para que procese varios archivos a la vez:

- Suministrando la ruta de acceso de los archivos de entrada/salida pertinentes por medio de caracteres de comodín (en lugar de dar un nombre de archivo fijo) en la configuración del componente (*véase* *Cambiar configuración de los componentes* 79). Es decir, puede introducir los comodines * y ? en el cuadro de diálogo "Configuración del componente" para que MapForce resuelva la ruta de acceso correspondiente cuando se ejecute la asignación.
- Conectando el nodo raíz de un componente con una secuencia que suministra la ruta de acceso de forma dinámica (p. ej. el resultado de la función `replace-fileext`). Cuando se ejecute la asignación, MapForce leerá de forma dinámica todos los archivos de entrada o generará todos los archivos de salida de forma dinámica.

Dependiendo de cual sea el objetivo del proyecto, podrá usar una de estas técnicas o ambas en la misma asignación. Sin embargo, no tiene sentido utilizar ambas técnicas al mismo tiempo en el mismo componente. Para indicar qué técnica desea usar para cada componente haga clic en el botón Archivo o Archivo/Cadena disponible junto al nodo raíz de cada componente. Estos son los comportamientos que puede especificar con estos botones:

Usar nombres de archivo de la configuración del componente	Si el componente debe procesar un archivo de instancia o varios, esta opción ordena el procesamiento de los nombres de archivo definidos en el cuadro de diálogo "Configuración del componente". Si selecciona esta opción, el nodo raíz no tiene un conector de entrada porque no es relevante. 

 © 2016-2022 Altova GmbH

	Si todavía no ha especificado archivos de entrada/salida en el cuadro de diálogo "Configuración del componente", el nombre del nodo raíz será **`Archivo: (predeterminado)`**. De lo contrario el nodo raíz muestra el nombre del archivo de entrada seguido de un punto y coma más el nombre del archivo de salida. Si el nombre de la entrada es idéntico al archivo de salida, entonces aparece como nombre del nodo raíz.  Esta opción y la opción **Usar nombres de archivo dinámicos dados por la asignación** se excluyen mutuamente.
Usar nombres de archivo dinámicos dados por la asignación	Esta opción ordena el procesamiento de los nombres de archivo definidos en el área de asignación al conectar valores con el nodo raíz del componente. Si selecciona esta opción, el nodo raíz recibe un conector de entrada con el que se pueden conectar valores que suministren de forma dinámica los nombres de archivo que se deben procesar durante la ejecución de la asignación. Si también definió nombres de archivo en el cuadro de diálogo "Configuración del componente", dichos valores se omitirán. Cuando se selecciona esta opción, el nombre del nodo raíz aparece como **`Archivo: <dinámico>`**.  Esta opción y la opción **Usar nombres de archivo de la configuración del componente** se excluyen mutuamente.

Redistribuir cadenas en XML, **Redistribuir cadenas en JSON**, **Redistribuir cadenas en CSV**, **Redistribuir cadenas en FLF**, **Redistribuir cadenas en EDI**	Si se activa, esta opción permite al componente aceptar un valor de cadena como entrada del nodo raíz y convertirlo en una estructura XML, JSON, CSV, FLF o EDI respectivamente. Consulte el apartado Analizar y serializar cadenas de texto 903 para obtener más información.
Serializar XML en cadenas, **Serializar JSON en cadenas**, **Serializar CSV en cadenas**, **Serializar FLF en cadenas**, **Serializar EDI en cadenas**	Si se activa, esta opción permite al componente aceptar una estructura como entrada y convertirla en cadenas. La estructura de entrada puede ser XML, JSON, CSV, FLF o EDI respectivamente. Consulte el apartado Analizar y serializar cadenas de texto 903 para obtener más información.

Estos son los componentes donde se pueden definir varios archivos de entrada/salida:

- Archivos XML
- Archivos de texto (archivos CSV*, FLF* y FlexText**)
- Documentos EDI**
- Hojas de cálculo Excel**
- Documentos XBRL**
- Archivos JSON**
- Archivos Protocol Buffers**

** Con MapForce Professional Edition*
*** Con MapForce Enterprise Edition*

En esta tabla puede ver la compatibilidad de cada lenguaje de MapForce con archivos de entrada/salida dinámicos y con comodines:

Lenguaje de destino	**Nombre de archivo de entrada dinámico**	**Nombre de archivo de entrada admite comodines**	**Nombre de archivo de salida dinámico**
XSLT 1.0	*	Incompatible con XSLT 1.0	Incompatible con XSLT 1.0
XSLT 2.0	*	*(1)	*
XSLT 3.0	*	*(1)	*
XQuery	*	*(1)	Incompatible con XQuery
C++	*	*	*
C#	*	*	*
Java	*	*	*
BUILT-IN (motor integrado)	*	*	*

Leyenda:

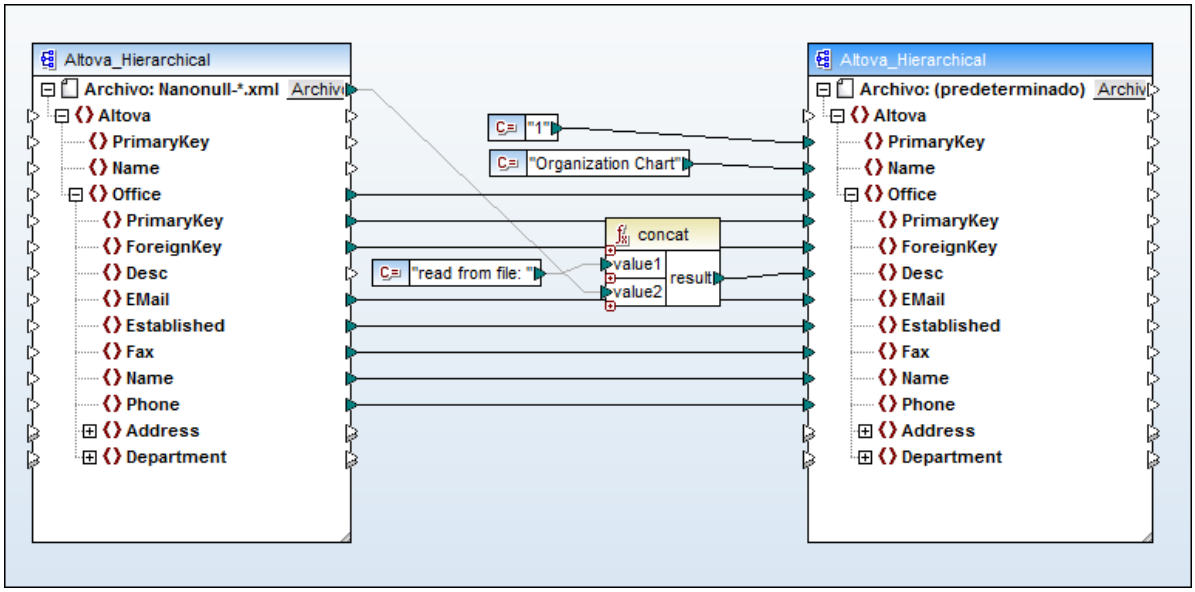
 © 2016-2022 Altova GmbH

*	Compatible
(1)	XSLT 2.0, XSLT 3.0 y XQuery utilizan la función **`fn:collection`**. La implementación en los motores XSLT 2.0, XSLT 3.0 y XQuery de Altova resuelve comodines. Los demás motores pueden comportarse de modo distinto.

7.4.1 Asignar varios archivos de entrada a un solo archivo de salida

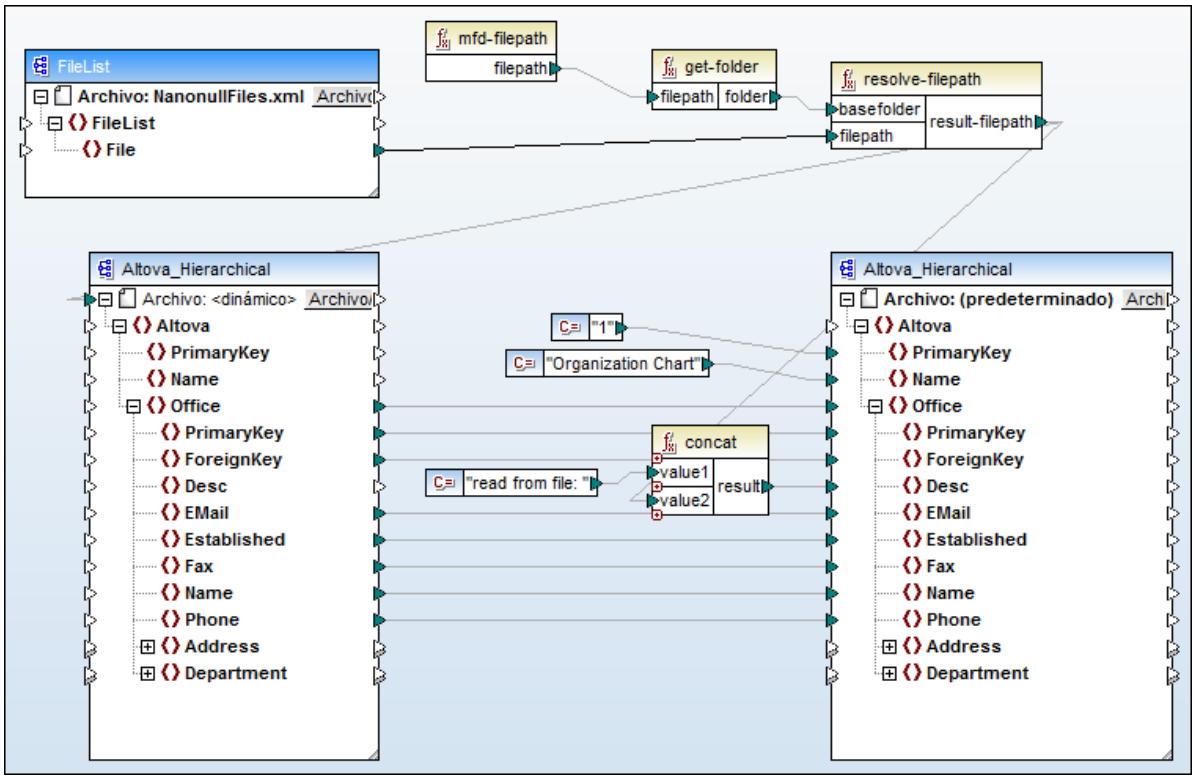
Hay dos maneras de procesar varios archivos de entrada a la vez:

- Introduciendo una ruta de acceso con comodines (* o ?) como archivo de entrada en el cuadro de diálogo "Configuración del componente". Se procesarán todos los archivos que coincidan con la ruta de acceso indicada. En la imagen siguiente, por ejemplo, en el campo *Archivo XML de entrada* se usa el comodín * para indicar que todos los archivos cuyo nombre empiece por `Nanonull-` serán los archivos de entrada de la asignación. En este ejemplo se combinan varios archivos de entrada para crear **un solo archivo de salida**. Como se puede apreciar en la imagen, no hay ningún conector dinámico con el componente de destino y el componente de origen tiene acceso a varios archivos gracias al comodín *. Observe también que el nombre del nodo raíz del componente de destino es **`Archivo: <predeterminado>`**. Es decir, no se definió ninguna ruta de acceso de archivo de salida en el cuadro de diálogo "Configuración del componente". Por tanto, al documento de destino se anexan varios archivos de origen.



MergeMultipleFiles.mfd (MapForce Basic Edition)

- Asignando una **secuencia** de cadenas al nodo **`Archivo`** del componente de origen. Cada cadena de la secuencia representa un nombre de archivo. Las cadenas también pueden contener comodines (que se resuelven automáticamente). Los componentes que pueden suministrar secuencias de nombres de archivos son los archivos XML y los campos de texto de BD.



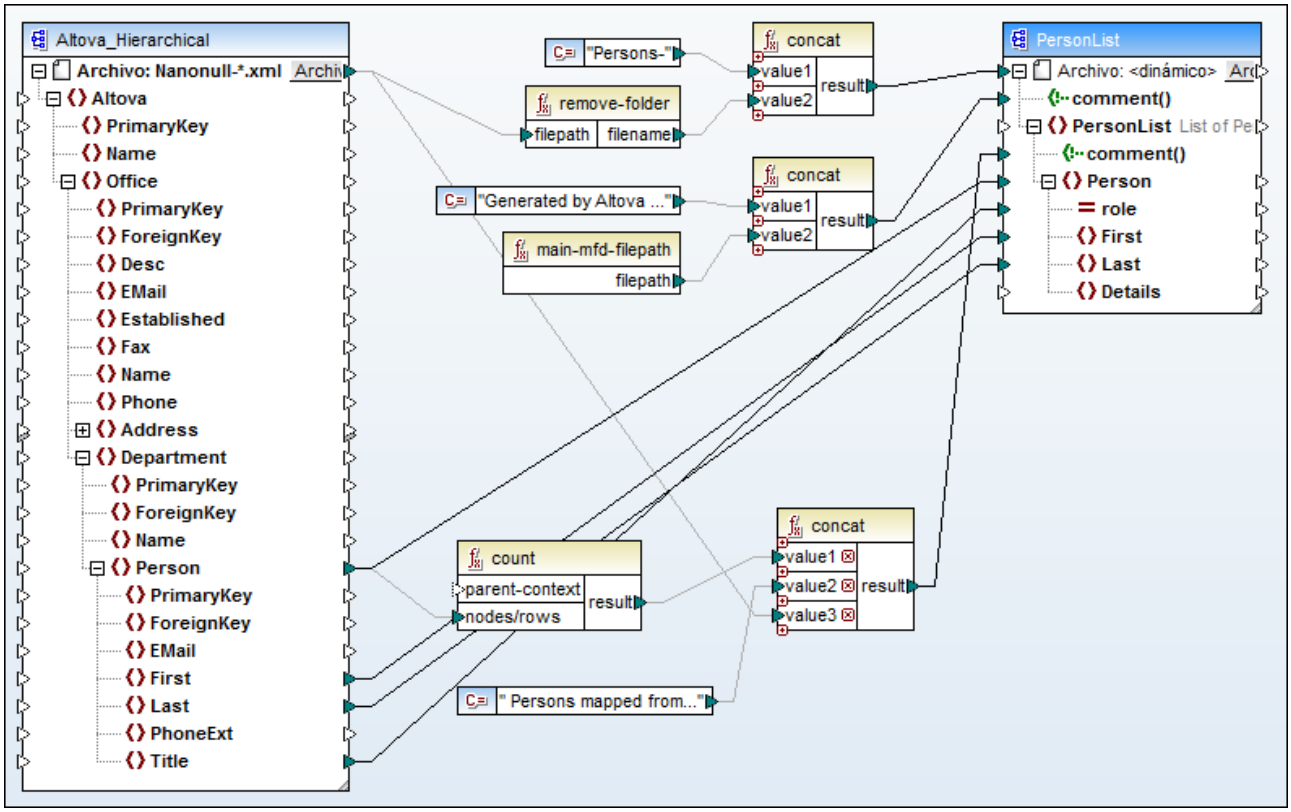
MergeMultipleFiles_List.mfd (MapForce Basic Edition)

7.4.2 Asignar varios archivos de entrada a varios archivos de salida

Para asignar varios archivos a varios archivos de destino es necesario generar nombres de archivo de salida únicos. En algunos casos, el nombre de los archivos de salida se puede derivar de cadenas existentes en los datos de entrada. En otros casos, lo más práctico es derivar el nombre de los archivos de salida del nombre del archivo de entrada (p. ej. cambiando la extensión de archivo).

En la asignación que aparece a continuación el nombre del archivo de salida se deriva del nombre del archivo de entrada: se añadió el prefijo **`Persons-`** con ayuda de la función **`concat`**.

 © 2016-2022 Altova GmbH



MultipleInputToMultipleOutputFiles.mfd (MapForce Basic Edition)

Nota: recomendamos conectar simplemente los nodos raíz de entrada y salida directamente, sin usar funciones de procesamiento. Esto sobrescribiría los archivos de entrada cuando se ejecute la asignación. Puede cambiar los nombres de los archivos de salida con ayuda de funciones como la función **`concat`**.

Con el comando de menú **Archivo | Configurar asignación** puede definir la configuración de rutas de acceso global que debe utilizar la asignación (véase Cambiar configuración de la asignación 115).

7.4.3 Especificar nombres de archivo como parámetros de asignación

Para indicar nombres de archivo personalizados como parámetros de entrada de la asignación:

1. Añada un componente de entrada simple a la asignación (**Función | Insertar componente de entrada**). Para más información consulte el apartado Agregar componentes de entrada simples 478.
2. Haga clic en el botón Archivo o Archivo/Cadena del componente de origen y seleccione **Usar nombres de archivo dinámicos dados por la asignación** en el menú contextual.
3. Conecte el componente de entrada simple **input** con el nodo raíz del componente de origen.

Para ver un ejemplo más detallado consulte el apartado Ejemplo: usar nombres de archivo como parámetros de asignación 482.

7.4.4 Vista previa de varios archivos de salida

Haga clic en el panel *Resultados* para ver el resultado de la asignación. Si la asignación produce varios archivos de salida, cada archivo contará con un panel numerado en el panel *Resultados*. Haga clic en los botones en forma de flecha para navegar por los distintos archivos de salida.

Vista previa 1 de 2 C:\Users\Documents\Altova\MapForce2017\MapForceExamples\Persons-Nanonull-Branch.xml

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <!--Generated by Altova MapForce (http://www.altova.com/mapforce) using
   C:\Users\Documents\Altova\MapForce2017\MapForceExamples\MultipleInputToMultipleOutputFiles.mfd-->
3  <PersonList xsi:noNamespaceSchemaLocation="PersonList.xsd" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
4    <!--6 Persons mapped from input file C:\Users\Documents\Altova\MapForce2017\MapForceExamples\Nanonull-Branch.xml-->
5    <Person role="Office Manager">
6      <First>Steve</First>
7      <Last>Meier</Last>
8    </Person>
9    <Person role="Accounts Receivable">
10     <First>Theo</First>
11     <Last>Bone</Last>
12   </Person>
13   <Person role="PR &amp; Marketing Manager US">
14     <First>Max</First>
15     <Last>Nafta</Last>
16   </Person>
17   <Person role="IT Manager">
18     <First>Valentin</First>
19     <Last>Bass</Last>
20   </Person>
21   <Person role="Support Engineer">
22     <First>Carl</First>
23     <Last>Franken</Last>
24   </Person>
25   <Person role="Support Engineer">
26     <First>Mark</First>
27     <Last>Redgreen</Last>
28   </Person>
29 </PersonList>
30
```

MultipleInputToMultipleOutputFiles.mfd

Hay dos maneras de guardar los archivos de salida generados:

- En el menú **Resultados** haga clic en el comando **Guardar todos los archivos de salida** ( ).
- Haga clic en el botón **Guardar todos** ( ) de la barra de herramientas.

7.4.5 Ejemplo: dividir un archivo XML en varios archivos

Este ejemplo demuestra cómo se pueden generar varios archivos XML de forma dinámica a partir de un solo archivo XML de origen. Este ejemplo es el archivo de muestra **<Documentos>\Altova\MapForce2023\MapForceExamples\Tutorial\Tut-ExpReport-dyn.mfd**.

 © 2016-2022 Altova GmbH

El archivo XML de origen (situado en la misma carpeta que la asignación) está compuesto por un informe de gastos de alguien llamado *Fred Landis* y contiene cinco conceptos de gastos de diferentes tipos. El objetivo de esta asignación de datos es generar un archivo XML por cada concepto de gasto.

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

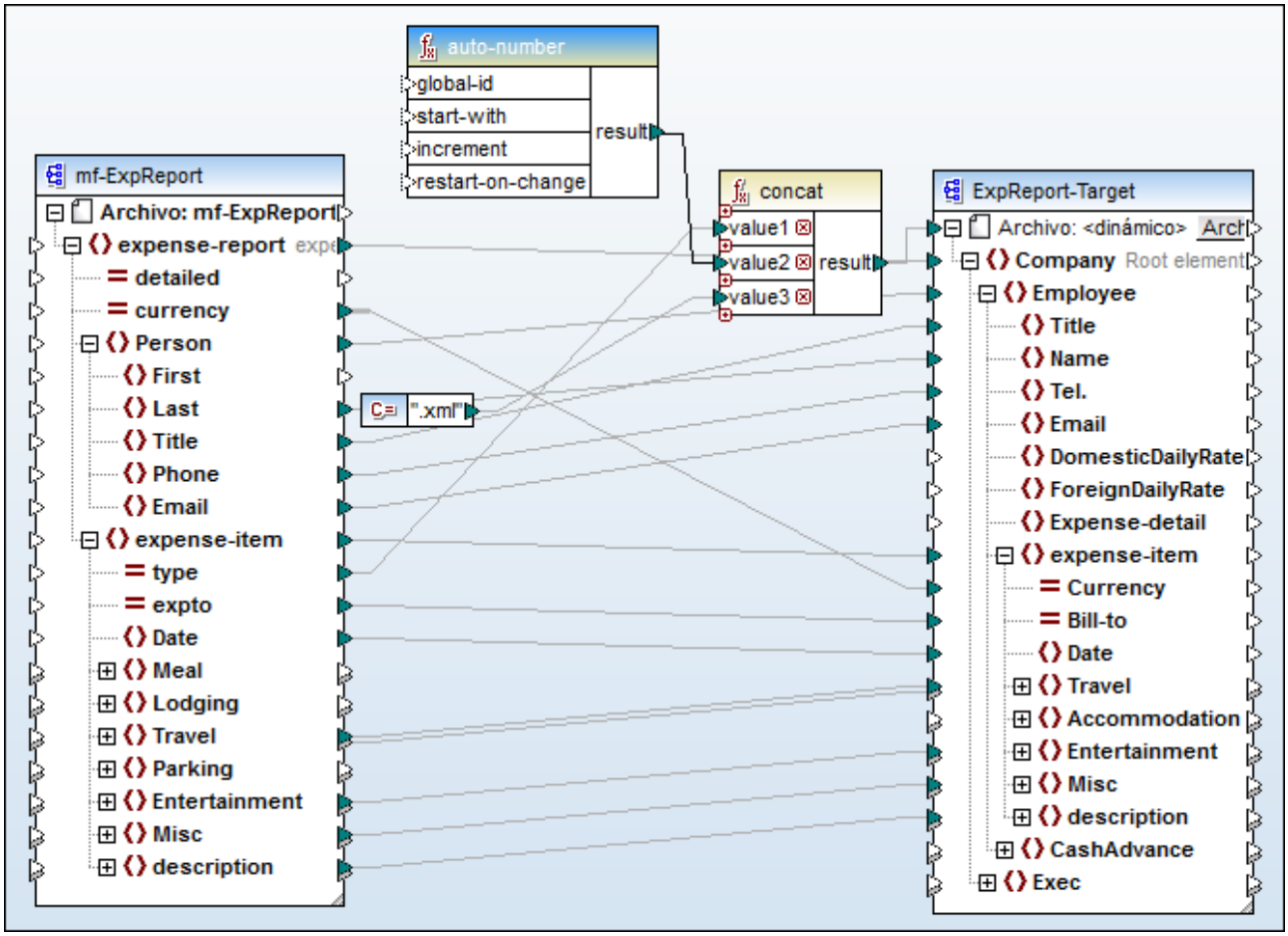
expense-item (5)

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

mf-ExpReport.xml (en la vista Cuadrícula de Altova XMLSpy)

Como el atributo `type` define el tipo de gasto, este será el elemento que usaremos para dividir el archivo de origen. Para conseguir nuestro objetivo debemos seguir estos pasos:

1. Insertamos una función **`concat`** (se puede arrastrar desde la biblioteca de funciones **core | string functions** de la ventana Bibliotecas).
2. Insertamos una constante (clic en el comando **Insertar | Constante**) e introducimos el valor ".xml".
3. Insertamos la función **`auto-number`** (se puede arrastrar desde la biblioteca de funciones **core | generator functions** de la ventana Bibliotecas).
4. Hacemos clic en el botón Archivo o Archivo/Cadena del componente de destino y seleccionamos **Usar nombres de archivo dinámicos dados por la asignación**.
5. Para terminar creamos las conexiones que se ven en la imagen siguiente y hacemos clic en el panel *Resultados* para ver el resultado de la asignación.



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

Tenga en cuenta que los archivos de salida resultantes tendrán nombres dinámicos construidos de la siguiente manera:

- El atributo `type` aporta la primera parte del nombre de archivo (p. ej. "Travel").
- La función **`auto-number`** aporta el número secuencial del archivo (p. ej. "Travel1", "Travel2" y así sucesivamente).
- La constante aporta la extensión del archivo, que es ".xml". Por tanto, el nombre del primer archivo de salida es "Travel1.xml".

7.4.6 Ejemplo: dividir una tabla de BD en varios archivos XML

Este ejemplo demuestra cómo se pueden generar varios archivos XML de forma dinámica a partir de cada registro de una tabla de BD. Este ejemplo es el archivo de muestra **<Documentos>\Altova\MapForce2023\MapForceExamples\Tutorial\PersonDB-dyn.mfd**.

El archivo de BD de origen (situado en la misma carpeta que la asignación) contiene una tabla llamada Person que tiene 21 registros. El objetivo de esta asignación es generar un archivo XML de salida por cada registro de la tabla Person.

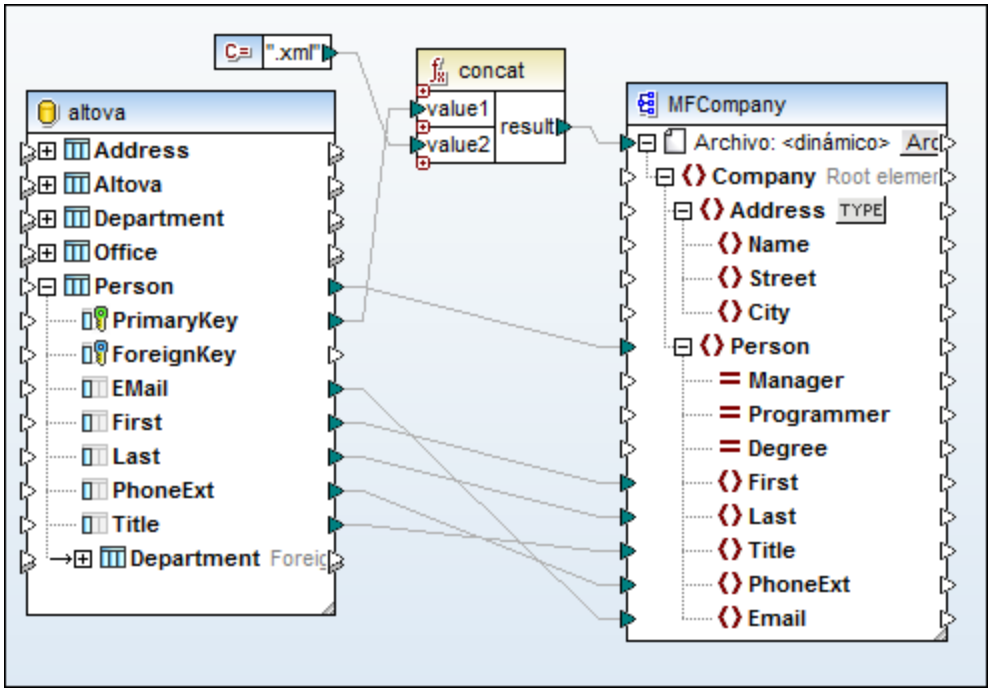
 © 2016-2022 Altova GmbH

Person

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

Como el campo "PrimaryKey" identifica a todas las personas de la tabla con una clave única, lo utilizaremos para dividir la base de datos de origen en varios archivos. Para conseguir nuestro objetivo debemos seguir estos pasos:

1. Insertamos una función `concat` (se puede arrastrar desde la biblioteca de funciones **core | string functions** de la ventana Bibliotecas).
2. Insertamos una constante (clic en el comando **Insertar | Constante**) e introducimos el valor ".xml".
3. Hacemos clic en el botón Archivo o Archivo/Cadena del componente de destino y seleccionamos **Usar nombres de archivo dinámicos dados por la asignación**.
4. Para terminar creamos las conexiones que se ven en la imagen siguiente y hacemos clic en el panel *Resultados* para ver el resultado de la asignación.



PersonDB-dyn.mfd (MapForce Professional Edition)

Tenga en cuenta que los archivos de salida resultantes tendrán nombres dinámicos construidos de la siguiente manera:

- El campo **PrimaryKey** aporta la primera parte del nombre de archivo (p. ej. "1").
- La constante aporta la extensión de archivo (".xml"). Por tanto, el nombre del primer archivo de salida es "1.xml".

© 2016-2022 Altova GmbH

7.5 Redistribuir y serializar cadenas de texto

La redistribución y serialización de cadenas de texto es una técnica avanzada de asignación de datos que le permitirá configurar un componente para redistribuir datos de una cadena o serializar datos en una cadena. Esta técnica es una manera alternativa de leer datos (o escribir datos) en archivos. Los componentes de MapForce que redistribuyen cadenas o serializan datos en cadenas pueden ser muy prácticos en determinados casos, por ejemplo:

- Si necesita insertar estructuras XML en campos de BD.
- Si necesita convertir fragmentos XML almacenados en campos de BD en archivos XML independientes.
- Si tiene datos heredados en formato texto (p. ej. contenido de longitud fija en un solo campo de BD) y desea pasarlos a una estructura basada en campos

La redistribución y serialización de cadenas de texto es compatible con este tipo de componentes de MapForce:

- Componentes de texto (texto CSV, texto de longitud fija)
- Archivos de esquema XML

La redistribución y serialización de cadenas de texto es compatible con estos lenguajes de destino de MapForce:

Lenguaje	Lectura	Escritura
BUILT-IN (vista previa de la transformación)	Sí	Sí
BUILT-IN (ejecución de archivo de ejecución de MapForce Server)	Sí	Sí

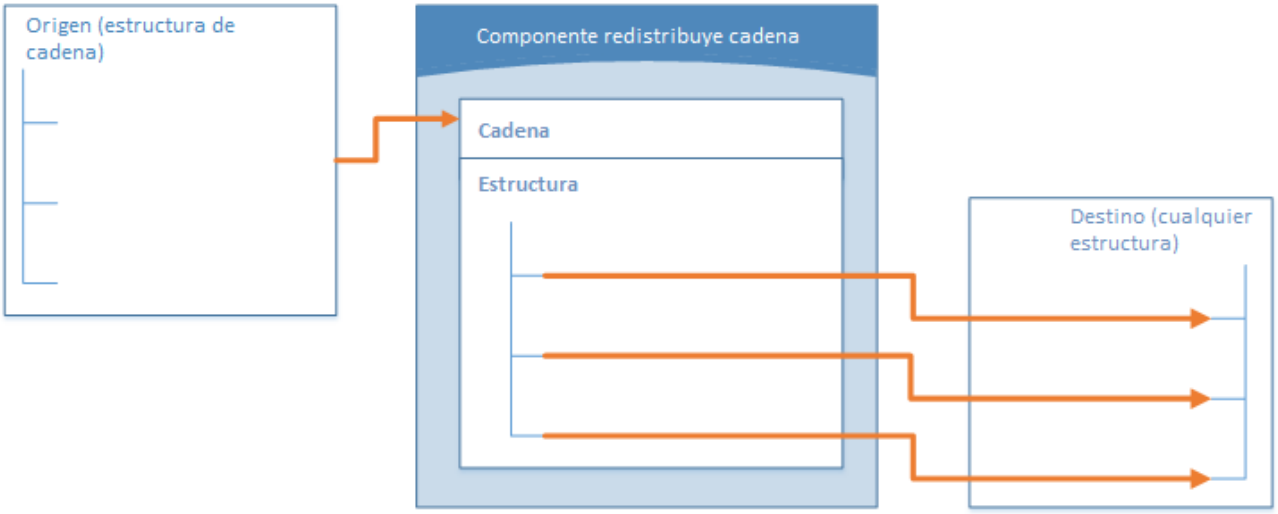
Esta sección incluye varios apartados:

- Información sobre el componente de análisis/serialización 903
- Ejemplo: serializar datos a una cadena (de XML a BD) 906

7.5.1 Información sobre el componente de análisis/serialización

En MapForce un componente de redistribución/serialización es un componente híbrido, que no es componente de origen ni de destino. Debido al papel que desempeñan en el diseño de asignación, este tipo de componentes debe colocarse entre componentes de origen y de destino.

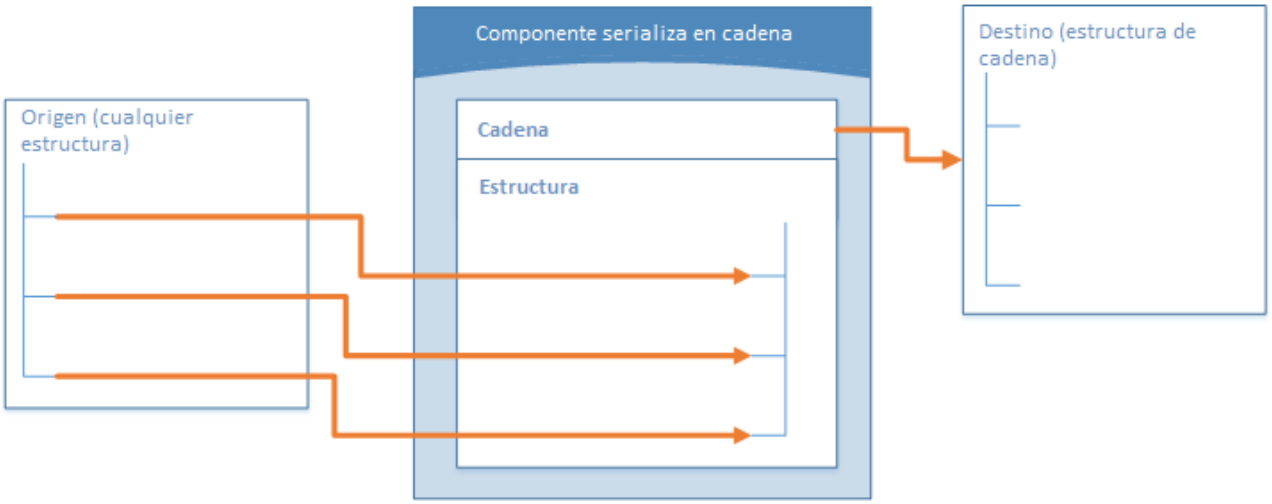
Puede utilizar un componente de redistribución/serialización de cadenas cuando necesite pasar a otro formato una cadena de texto que tiene una estructura (p. ej. datos XML guardados como cadena de texto en una BD). La redistribución de la cadena de origen a través del componente de redistribución/serialización implica que la cadena de origen se convierte en una estructura de MapForce y, por tanto, será posible acceder a los elementos y atributos del XML de origen guardado como cadena.



Componente de redistribución de cadena genérico

En el diagrama anterior puede verse la estructura típica de un componente de MapForce que redistribuye una cadena. Observe que el componente de redistribución/serialización de cadenas está situado entre el componente de origen y el de destino. Lo que hace este componente es aceptar como entrada una estructura de cadena (por medio de un conector de MapForce que está conectado su nodo de nivel superior **Cadena**). La estructura de salida puede ser cualquier destino de datos compatible con MapForce.

Cuando se serializan datos de un componente a una cadena ocurre lo contrario. Toda la estructura del componente de MapForce se convierte en una estructura de cadena que se puede manipular. Por ejemplo, puede escribir un archivo XML (o fragmento XML) en un campo de BD o en una sola celda de una hoja de cálculo Excel.



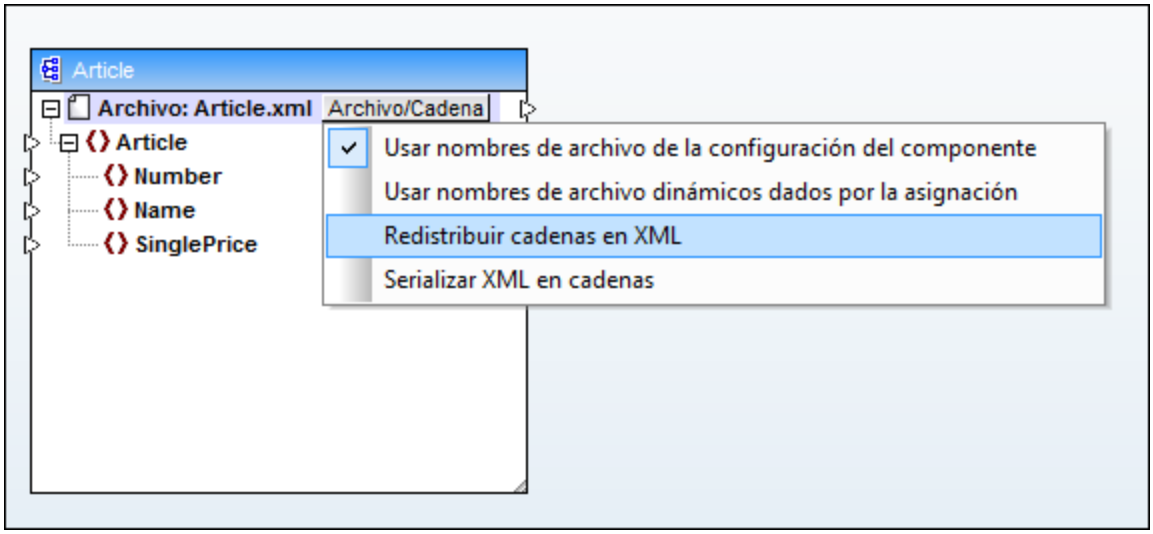
Componente de serialización de cadena genérico

En el diagrama anterior puede verse un componente de serialización de cadena genérico. Lo que hace este componente es aceptar como entrada cualquier origen de datos compatible con MapForce (por medio de

 © 2016-2022 Altova GmbH

conectores estándar de MapForce). La estructura de salida es una cadena que se puede pasar dibujando un conector entre el nodo de nivel superior **Cadena** del componente y un elemento del componente de destino (p. ej. una celda de una hoja de cálculo). Puede ver un ejemplo en Ejemplo: serializar datos a una cadena (de XML a BD) 906.

Para configurar un componente para la redistribución o serialización de cadenas basta con hacer clic en el botón Archivo/Cadena situado junto al nodo raíz de un componente y seleccionar la opción correspondiente.



Cambiar el modo del componente

Nota: un componente de redistribución/serialización de cadenas no puede leer datos de una cadena y escribirlos en otra simultáneamente. Por tanto, el nodo raíz puede tener un conector de entrada o uno de salida (pero no ambos). MapForce genera un error si se intenta usar el mismo componente para ambas operaciones.

Cuando un componente se configura para la redistribución o serialización de cadenas, su aspecto cambia:

- El título del componente aparece con el prefijo **parse** (redistribución) o **serialize** (serialización).
- El color de fondo de la barra de título es amarillo (igual que la barra de título de las funciones).
- El nodo de nivel superior incluye el prefijo `Cadena:` y aparece junto al icono .
- Si el componente redistribuye una cadena, el conector de salida del nodo raíz no es relevante y, por tanto, no está disponible.
- Si el componente serializa datos en una cadena, el conector de entrada del nodo raíz no es relevante y, por tanto, no está disponible.

Cuando un componente está en modo de redistribución/serialización de cadenas, su configuración se puede cambiar como si se tratase de un componente basado en un archivo (*véase* *Cambiar configuración de los componentes* 79). Recuerde que no todas las opciones de configuración están disponibles si el componente está en modo de redistribución/serialización de cadenas.

7.5.2 Ejemplo: serializar datos a una cadena (de XML a BD)

Este ejemplo explica paso a paso cómo crear un diseño de asignación que serializa datos en una cadena de texto. El ejemplo viene acompañado de un archivo de muestra. Si desea consultar el archivo de muestra antes de empezar con el ejemplo, puede hacerlo en esta ubicación:
<Documentos>\Altova\MapForce2023\MapForceExamples\Tutorial\SerializeToString.mfd.

Imaginemos que tenemos un archivo XML (y su correspondiente esquema) compuesto por varios elementos `<Person>`. Cada elemento `<Person>` describe el nombre de una persona, su apellido, su puesto, su extensión de teléfono y su dirección de correo electrónico:

```
<Person>
      <First>Joe</First>
      <Last>Firstbread</Last>
      <Title>Marketing Manager Europe</Title>
      <PhoneExt>621</PhoneExt>
      <Email>j.firstbread@nanonull.com</Email>
</Person>
```

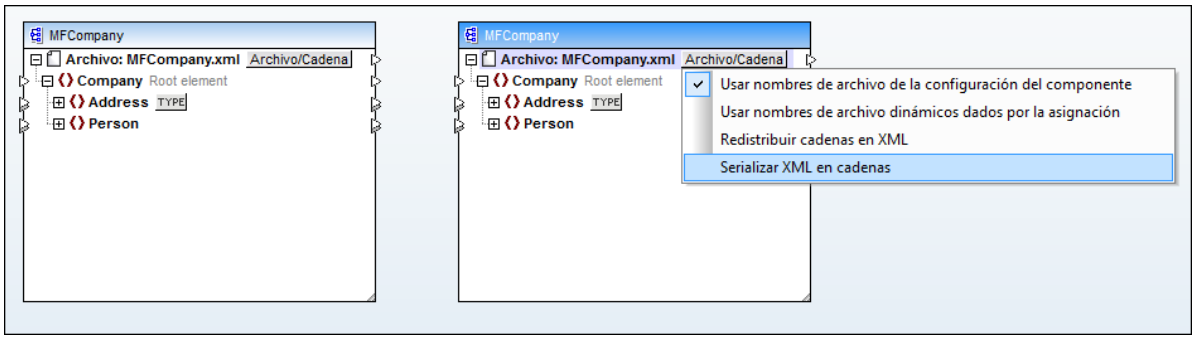
Nuestro objetivo es extraer cada elemento `<Person>` del archivo XML e insertarlo literalmente (incluidas las etiquetas XML) en un nuevo registro de BD de la tabla `PEOPLE` de una base de datos SQLite. La tabla `PEOPLE` solamente contiene dos columnas: `ID` y `PERSON`. Esta sería su definición completa:

```
CREATE TABLE PEOPLE (ID INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL, PERSON TEXT);
```

Una vez ejecutada la asignación, el resultado esperado es que la tabla `PEOPLE` tenga tantas filas como elementos `<Person>` contenía el archivo XML.

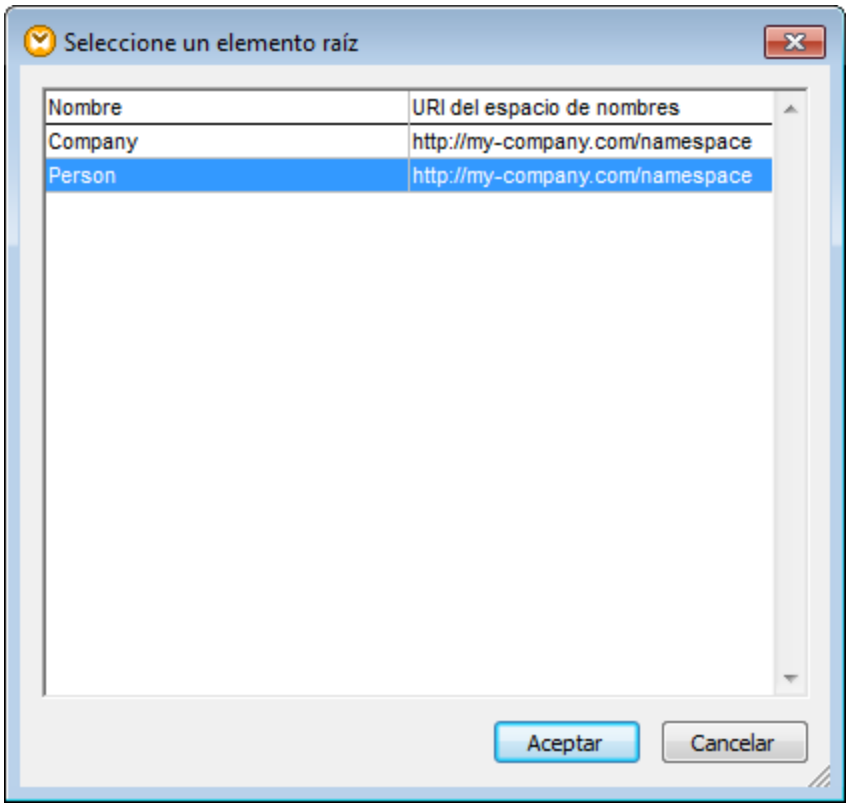
Para conseguir este objetivo debe seguir estos pasos:

1. Añada el componente XML de origen (con el comando de menú **Insertar | Archivo o esquema XML**). El archivo de muestra está en esta ubicación:
   <Documentos>\Altova\MapForce2023\MapForceExamples\Tutorial\MFCompany.xml.
2. Cree una copia del componente XML (mediante copia y pega).
3. En la segunda copia del componente XML haga clic en el botón Archivo/Cadena y seleccione **Serializar XML en cadenas** en el menú contextual.



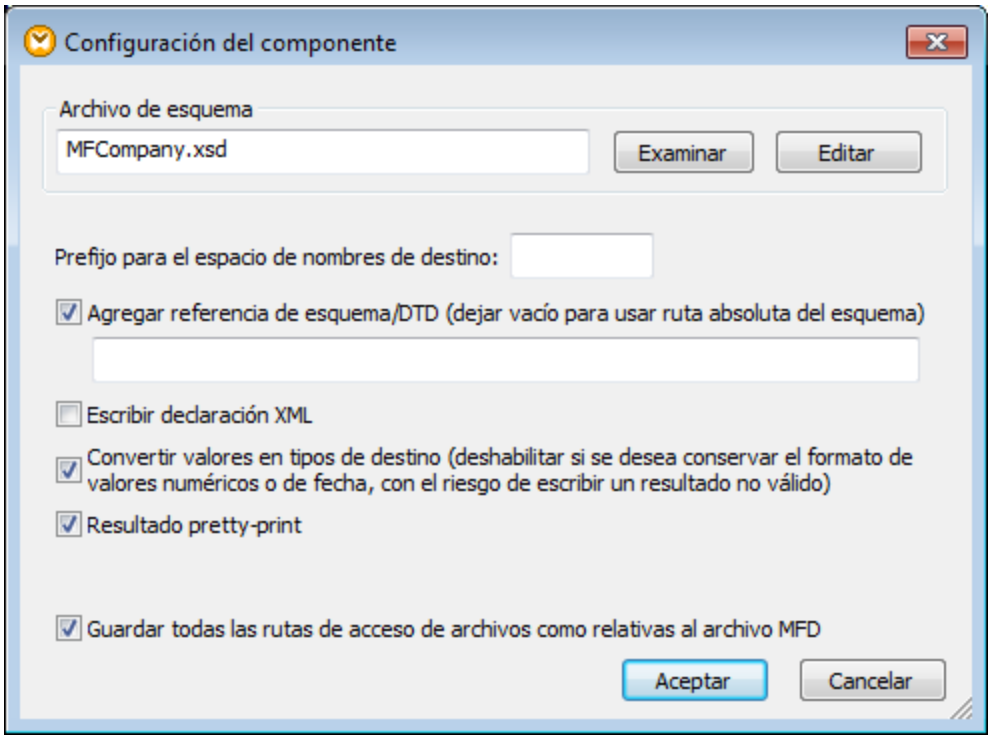
4. Haga clic con el botón derecho en la copia del componente XML y seleccione **Cambiar de elemento raíz** en el menú contextual. Ahora cambie el elemento raíz por `<Person>`.

 © 2016-2022 Altova GmbH

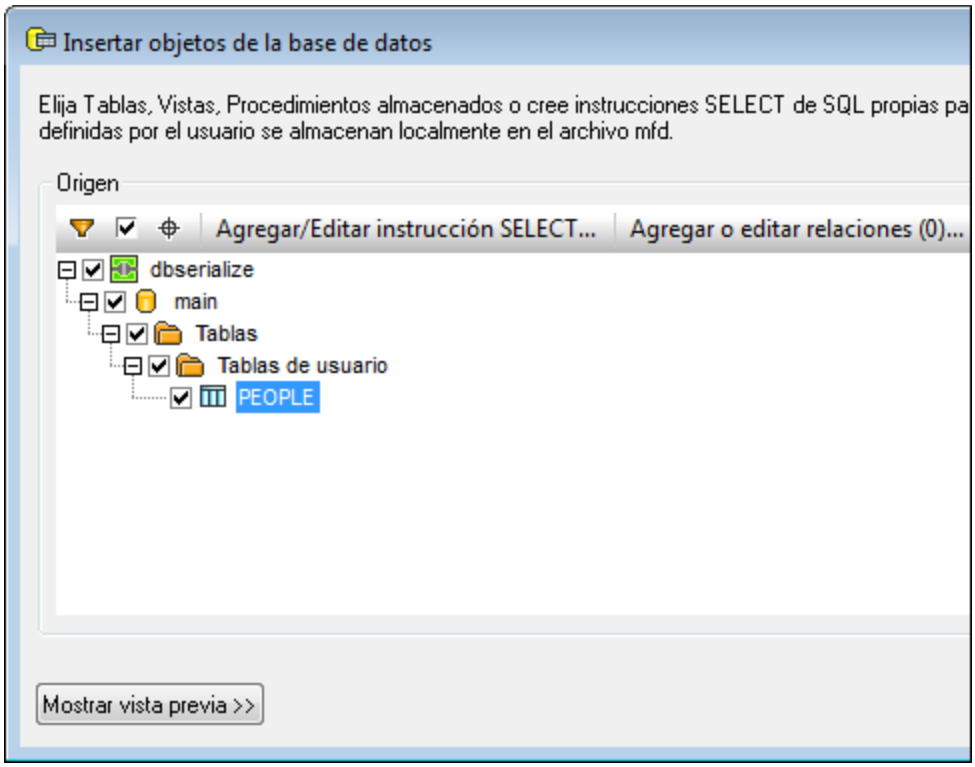


Por lo general, el elemento raíz puede cambiarse por cualquier elemento que tenga una declaración global (no local) en el esquema XML. Los elementos que no estén definidos globalmente en el esquema no aparecerán en el cuadro de diálogo "Seleccione un elemento raíz".

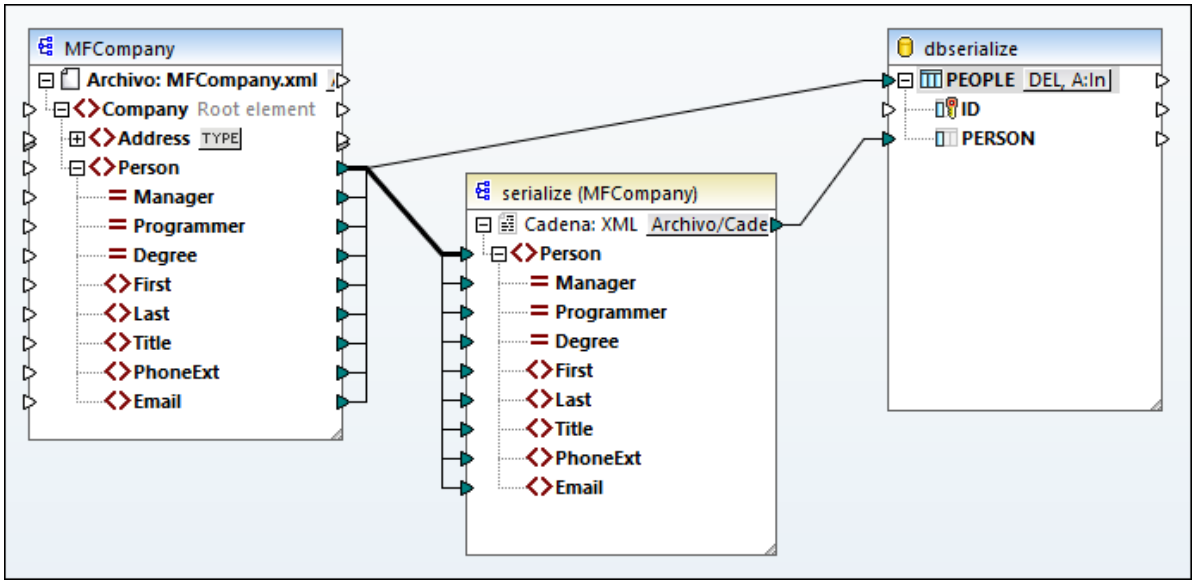
5. Haga doble clic en el componente y deshabilite la casilla *Escribir declaración XML*. Esto sirve para impedir que la declaración XML se escriba por cada elemento `<Person>`.



6. Añada el componente de base de datos SQLite de destino al área de asignación (la base de datos está en esta ubicación: **<Documentos>\Altova\MapForce2023\MapForceExamples\Tutorial\\dbserialize.db**). Para añadir el componente de base de datos seleccione el comando de menú **Insertar | Base de datos** (*véase también* *Conectarse a una base de datos* 161). Cuando solicite un objeto de base de datos, seleccione la tabla PEOPLE.

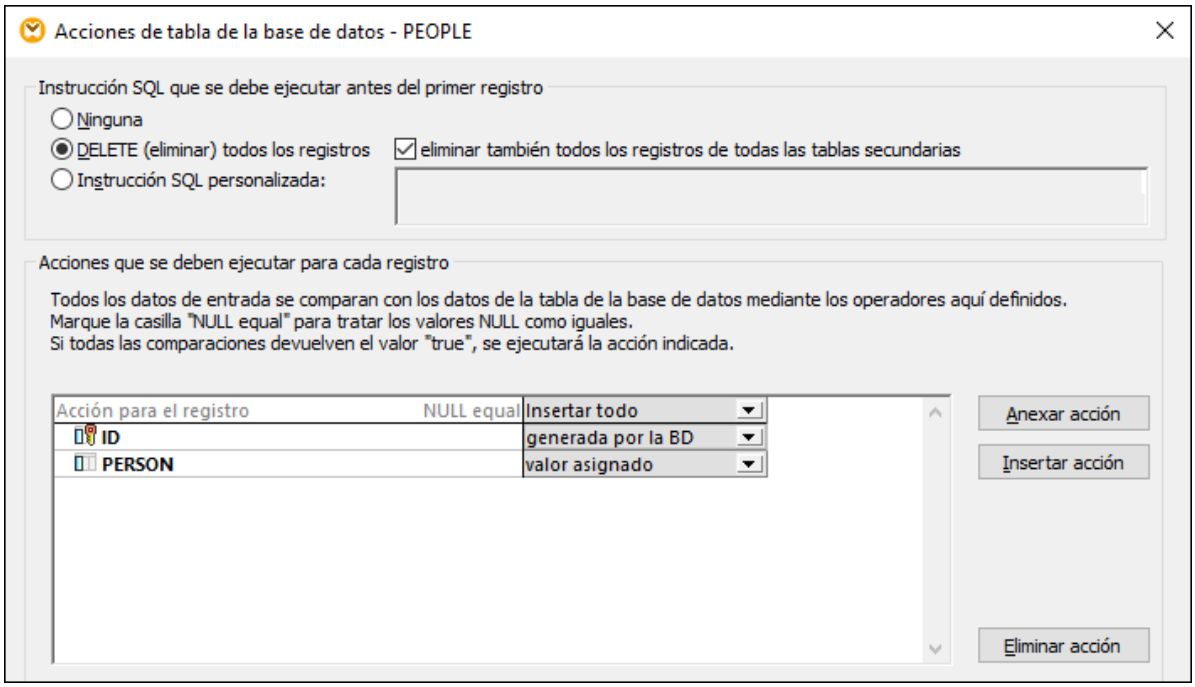


7. Conecte los componentes tal y como puede ver en la imagen siguiente. En el lado izquierdo de la asignación, el elemento `<Person>` se asigna al componente de serialización. En el lado derecho el valor de cadena serializado se inserta en la columna `PERSON` de la tabla de BD `PEOPLE`. Por último, el conector que une `<Person>` con la tabla `PEOPLE` indica que se debe crear un registro nuevo por cada elemento `<Person>` que se detecte.



8. Haga clic en el botón **A:In** del componente de base de datos y haga lo siguiente:
   a. Seleccione la opción **Eliminar todos los registros de la tabla**. Cuando se ejecute la asignación esto eliminará cualquier registro de la BD que exista antes de insertar registros nuevos.

b. Seleccione la opción **Generada por la BD**, junto a la columna *ID*. Esto garantiza que la BD genere el ID del registro. Observe que la opción **Generada por la BD** aparece solamente si la columna admite esta opción. Las columnas que no son un campo identitario ni autoincrementado está disponible la opción **max+1**; esta opción comprueba cuál es el valor máximo que ya existe en esa columna e inserta el siguiente número entero disponible incrementado por 1.



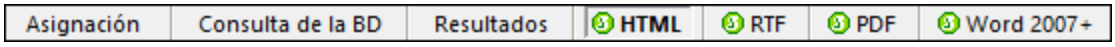
Observe que está seleccionada la acción **max()+1** para la columna ID. Esto indica que MapForce debe analizar cuál es el valor de ID máximo que ya existe en la BD y que debe insertar el entero siguiente, incrementado en 1.

Ya ha terminado de crear un diseño de asignación que serializa datos a una cadena. Si hace clic en el panel *Resultados*, la vista previa de la consulta SQL indica que en la BD se insertarán registros distintos para cada elemento `<Person>` del archivo XML.

 © 2016-2022 Altova GmbH

7.6 Paneles de resultados de StyleVision

En asignaciones en las que el componente de destino es XML el resultado de la asignación se puede previsualizar y guardar como documentos HTML, RTF, PDF y Word 2007+, siempre que Altova StyleVision esté instalado en su equipo. Si está usando la versión Enterprise de StyleVision, estas vistas previas contendrán también gráficos. Si hay disponible una vista previa de una asignación en alguno de esos formatos aparecerán nuevas pestañas junto a la de **Resultados** (*imagen siguiente*).



Nota:

- si está instalado StyleVision es posible previsualizar los resultados en los formatos **HTML**, **RTF** y **texto**.
- Con StyleVision Enterprise se pueden previsualizar los resultados en los formatos **HTML**, **RTF**, **PDF**, **Word 2007+** y **texto**.
- Para previsualizar los resultados como PDF es necesario tener instalados Java, Acrobat Reader y FOP () en su versión 0.93 o 1.0. FOP se instala automáticamente al instalar StyleVision a no ser que durante la instalación especifique que no desea instalarlo.
- En la edición de 64 bits de MapForce las previsualizaciones en formato Word 2007+ y RTF se abren como aplicaciones no integradas.
- Si su asignación contiene componentes que actúan a la vez como origen y como destino, la vista previa de StyleVision solo estará disponible para aquellos componentes para los que se haya activado el botón **Vista previa** . Para más información sobre este tipo de asignaciones de datos, consulte Asignaciones en cadena 840.

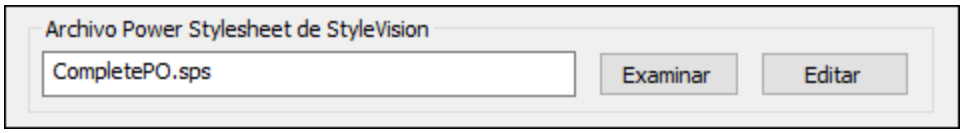
Para generar una vista previa de los datos de la asignación en los paneles de resultados de StyleVision necesita:

- Tener instalado Altova StyleVision en su equipo, bien como instalación independiente o como parte de Altova MissionKit.
- El componente de destino debe tener asociado un archivo SPS (StyleVision Power Stylesheet). El archivo de la hoja de estilos se puede crear o editar con StyleVision. Esta hoja de estilos no se puede cambiar directamente en MapForce, pero puede abrirla en StyleVision a través de MapForce. Una vez está lista la hoja de estilos, puede asignársela a un componente de MapForce, como mostramos a continuación.

Configurar los paneles de resultados de StyleVision

Siga las instrucciones a continuación para configurar los paneles de resultados de StyleVision.
Asignar una hoja de estilos StyleVision Power Stylesheet a un componente de destino
Para asignar una hoja de estilos SPS a un componente de destino siga estos pasos:

1. En StyleVision, cree el archivo de hoja de estilos. Asegúrese de que usa como origen el mismo esquema XML que el del componente de MapForce.
2. En MapForce, haga clic con el botón derecho en el componente XMLde destino y seleccione **Propiedades**.
3. En el cuadro de diálogo **Configuración del componente**, junto a **Archivo Power Stylesheet de StyleVision**, busque la hoja de estilos que creó antes (*imagen siguiente*).



Nota: la ruta al archivo Power Stylesheet de StyleVision puede ser absoluta o relativa. Para más detalles consulte Usar rutas de acceso relativas y absolutas 82.

Guardar los resultados generados por StyleVision
Puede guardar los resultados generados por StyleVision en un archivo de forma muy parecida a como guarda los resultados de cualquier otra asignación. Haga clic en el botón de la barra de herramientas (**Guardar el resultado generado**) o vaya al menú **Resultados** y haga clic en **Guardar archivo de salida**.

Automatizar la generación de distintos formatos con productos de Altova
Si quiere que su asignación genere automáticamente archivos HTML, PDF, RTF, Word 2007+ o texto (ya sea en el mismo equipo o en uno distinto, e incluso en otra plataforma) puede usar para ello MapForce Server o StyleVision Server. Estos son dos productos con licencias distintas que amplían las funciones de StyleVision y MapForce respectivamente. Para ello, cada aplicación tiene un papel específico:

- MapForce permite diseñar la asignación (`.mfd`) que define las entradas y salidas de la transformación de datos (por ejemplo BD en XML)
- MapForce Server ejecuta la asignación (`.mfx`) desde la línea de comandos o desde una API (en el mismo sistema operativo o en uno distinto)
- StyleVision permite diseñar la hoja de estilos (`.sps`) necesaria para transformar los resultados de la asignación en HTML, PDF, RTF o Word 2007+
- StyleVision Server ejecuta la hoja de estilos `.sps` que transforma los resultados de la asignación en el formato de destino que prefiera. Esto puede hacerse desde la línea de comandos o desde una API (en el mismo sistema operativo o en uno distinto).
- Tanto StyleVision Server como MapForce Server se pueden ejecutar a través de FlowForce Server (que requiere una licencia aparte), con el que las asignaciones de datos de MapForce y las transformaciones de StyleVision se pueden ejecutar como trabajos programados, con desencadenadores bajo demanda, es decir, de forma totalmente automatizada.

Ejemplos

En el ejemplo siguiente (**`MapForceExamples\CompletePO.mfd`**) puede ver el resultado de la asignación en el panel de resultados **HTML** de StyleVision. Esta asignación genera un pedido en formato XML. Haga clic con el botón derecho en el componente de destino, seleccione **Propiedades** y observe que la asignación tiene asignado un archivo `.sps`.

Si hace clic en el panel **HTML** verá este resultado:

 © 2016-2022 Altova GmbH

Fax +1 (321) 555 5155 - 9

office@nanonull.com

www.nanonull.com

Purchase Order Number: PO - _ _ _ _ _ _

TO: **Mrs./Mr. Ted Little**

Long Way
Los-Angeles
CA 34424
Our Customer Identifier: ID-3

Order Date: _____

Shipping Date: _____

Item	Quantity	Name	Unit Price ($)	Total ($)
3	5	Pants	34	170
1	17	T-Shirt	25	425
				595

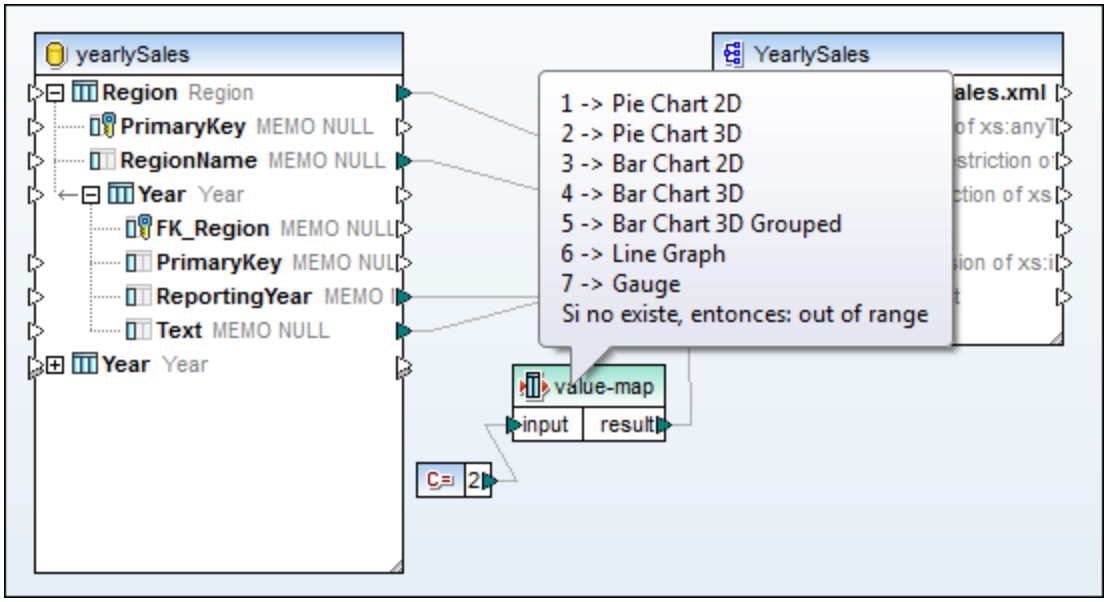
Other Comments: _____

_____ _____

Authorized Signature Date

Mapping | XSLT2 | DB Query | Output | HTML | RTF | PDF | Word 2007+

Otro ejemplo es **`YearlySales.mfd`**. Encontrará esta asignación en el siguiente directorio: **`<Documentos>\Altova\MapForce2023\MapForceExamples\Tutorial\YearlySales.mfd`**. La hoja de estilos asignada a este ejemplo se diseñó en StyleVision de forma que sea posible controlar el tipo de gráfico cambiando el valor del elemento `ChartType`. Esto permite cambiar el tipo de gráfico directamente desde la asignación. Más concretamente, puede cambiar el valor predeterminado de la constante a cualquier valor del 1 al 7. Si coloca el cursor del ratón sobre el componente **`value-map`** verá los valores que puede elegir (*imagen siguiente*).



El valor predeterminado de la constante es 2, que genera un gráfico 3D en forma de tarta en el resultado. Para mostrar otros tipos de gráfico, cambie este valor a cualquiera de los valores permitidos y haga clic en la pestaña **Resultados** para ver los resultados.

© 2016-2022 Altova GmbH

7.7 Generar la documentación de la asignación

Puede generar documentación detallada sobre cualquier asignación en formato HTML, Microsoft Word (.doc) o RTF. Si tiene instalado StyleVision también puede generar esa documentación en formato PDF.

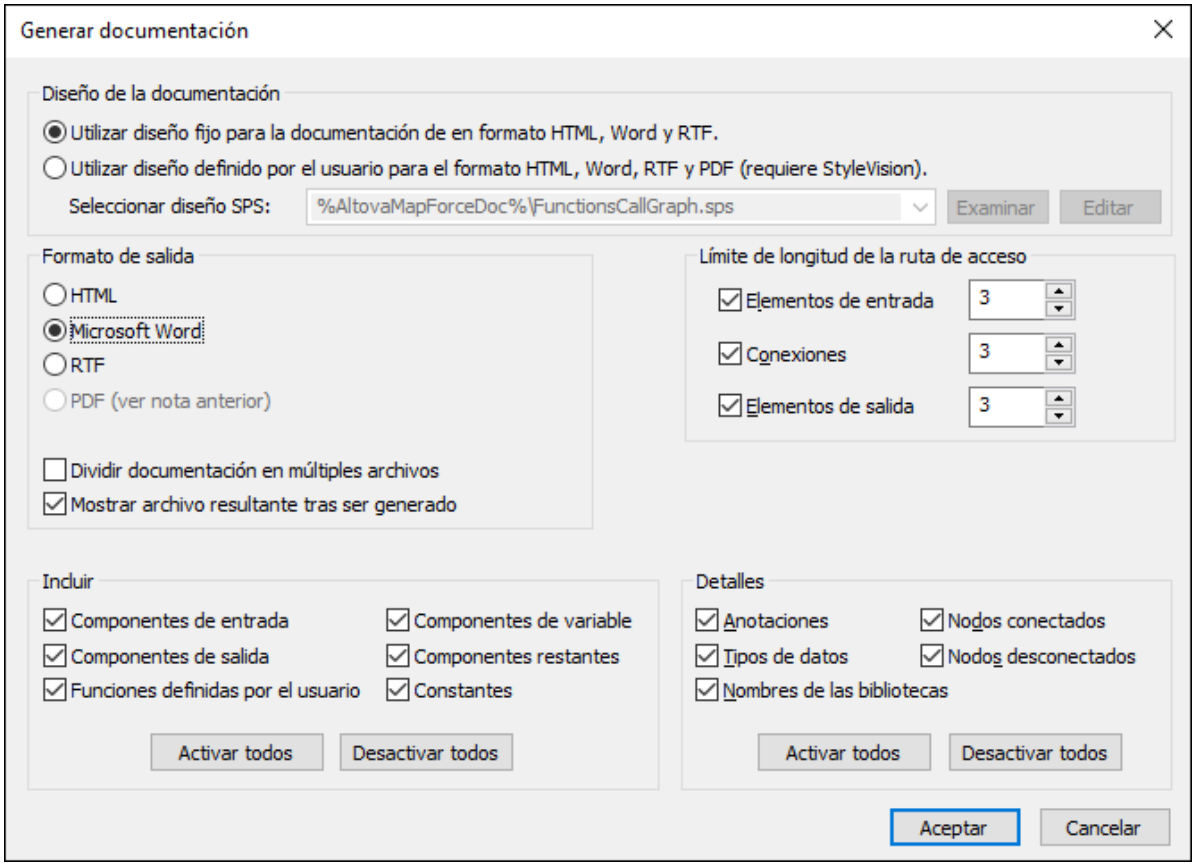
Requisitos

- debe tener instalado Microsoft Word 2000 o una versión más reciente si quiere generar documentación en formato Microsoft Word.
- debe tener instalado StyleVision si quiere generar documentación en formato PDF o personalizar el diseño de la documentación que genere.

Por defecto, la documentación se genera con un diseño fijo del que puede configurar varias opciones, como qué componentes incluir, cuántos niveles de las rutas se muestran, etc. Si tiene instalado StyleVision también puede utilizar los archivos SPS (StyleVision Power Stylesheet) o incluso crear un diseño nuevo en StyleVision.

Para generar la documentación de la asignación:

1. En el menú **Archivo** haga clic en **Generar documentación** para abrir el cuadro de diálogo del mismo nombre.



2. Configure las opciones que necesite y haga clic en **Aceptar**.

A continuación describimos las opciones que puede configurar en este cuadro de diálogo.

Diseño de la documentación

- *Utilizar diseño fijo....:* seleccione esta opción para usar la hoja de estilos integrada para dar formato a la documentación.
- *Utilizar diseño definido por el usuario...:* seleccione esta opción para usar una hoja de estilos SPS creada en StyleVision. MapForce viene con varios archivos SPS que puede usar para dar formato a la documentación y están disponibles en la carpeta `...\Documentos\Altova\MapForce2023\Documentation\MapForce\`. For details, see Predefined StyleVision Power Stylesheets 916.
- Botón **Examinar**: haga clic en este botón para buscar el archivo SPS predefinido que desea usar.
- Botón **Editar**: haga clic en este botón para editar el archivo SPS seleccionado en StyleVision.

Formato de salida

- Elija el formato de salida de la documentación: *HTML*, *Microsoft Word*, *RTF* o *PDF*. Los documentos de Microsoft Word se crean con la extensión de archivo `.doc` si utiliza el diseño fijo y con la extensión `.docx` si usa un archivo SPS de StyleVision. Para poder usar el formato de salida PDF debe tener instalado StyleVision y usar un archivo SPS de StyleVision para generar la documentación.
- Seleccione **Dividir documentación en múltiples archivos** si quiere generar varios archivos de documentación, uno por cada componente individual, como el componente de entrada o el de salida. Si usa un diseño fijo los enlaces entre documentos se crean automáticamente.
- Si marca la casilla *Mostrar archivo resultante tras ser generado*, MapForce abre los archivos generados en el explorador o la aplicación predeterminados para ello.

Límite de longitud de la ruta de acceso

Use estas opciones para definir el límite máximo de la longitud de la ruta de acceso que quiere que aparezca en los elementos de entrada o salida o en las conexiones. Por ejemplo, con la longitud predeterminada **3**, la ruta de un elemento sería **.../ShortPO/LineItems/LineItem**.

Incluir

Marque aquí los componentes que quiere incluir en la documentación.

Detalles

Use estas opciones para personalizar el nivel de detalle de la documentación generada. La opción Nombres de las bibliotecas inserta el prefijo principal de las funciones.

7.7.1 Hojas de estilos predefinidas de StyleVision

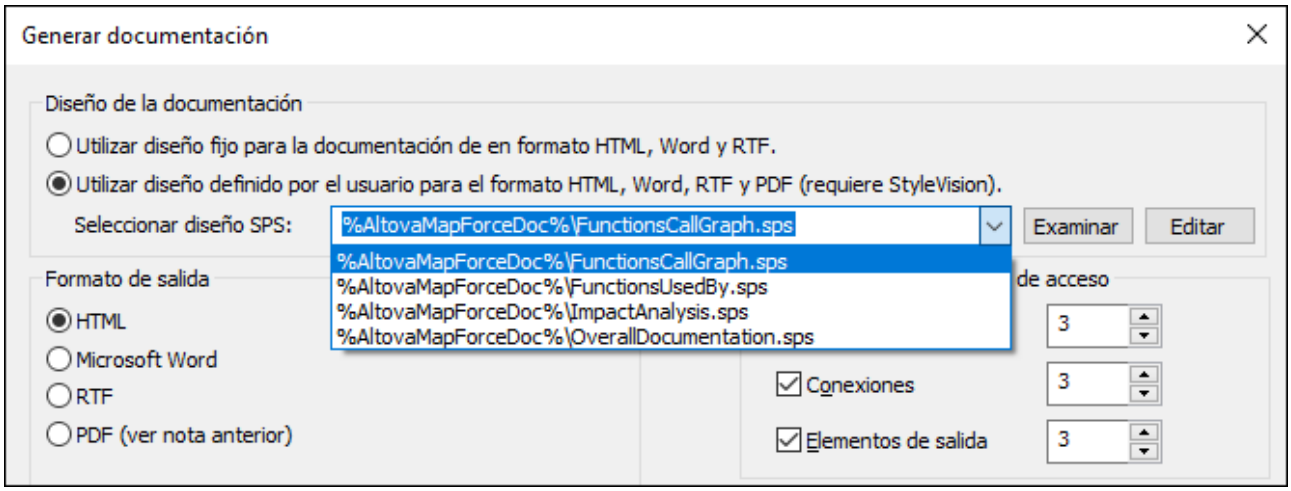
Si tiene instalado StyleVision en su equipo puede generar documentación a partir de asignaciones. Para ello seleccione uno de los archivos SPS (StyleVision Power Stylesheet) como plantilla en lugar del diseño fijo integrado. Puede elegir entre estos archivos SPS (StyleVision Power Stylesheet):

- **FunctionCallGraph.sps**: muestra el gráfico de llamadas de la asignación principal y las funciones definidas por el usuario que haya.
- **FunctionsUsedBy.sps**: muestra qué funciones usa, directa o indirectamente, la asignación.

 © 2016-2022 Altova GmbH

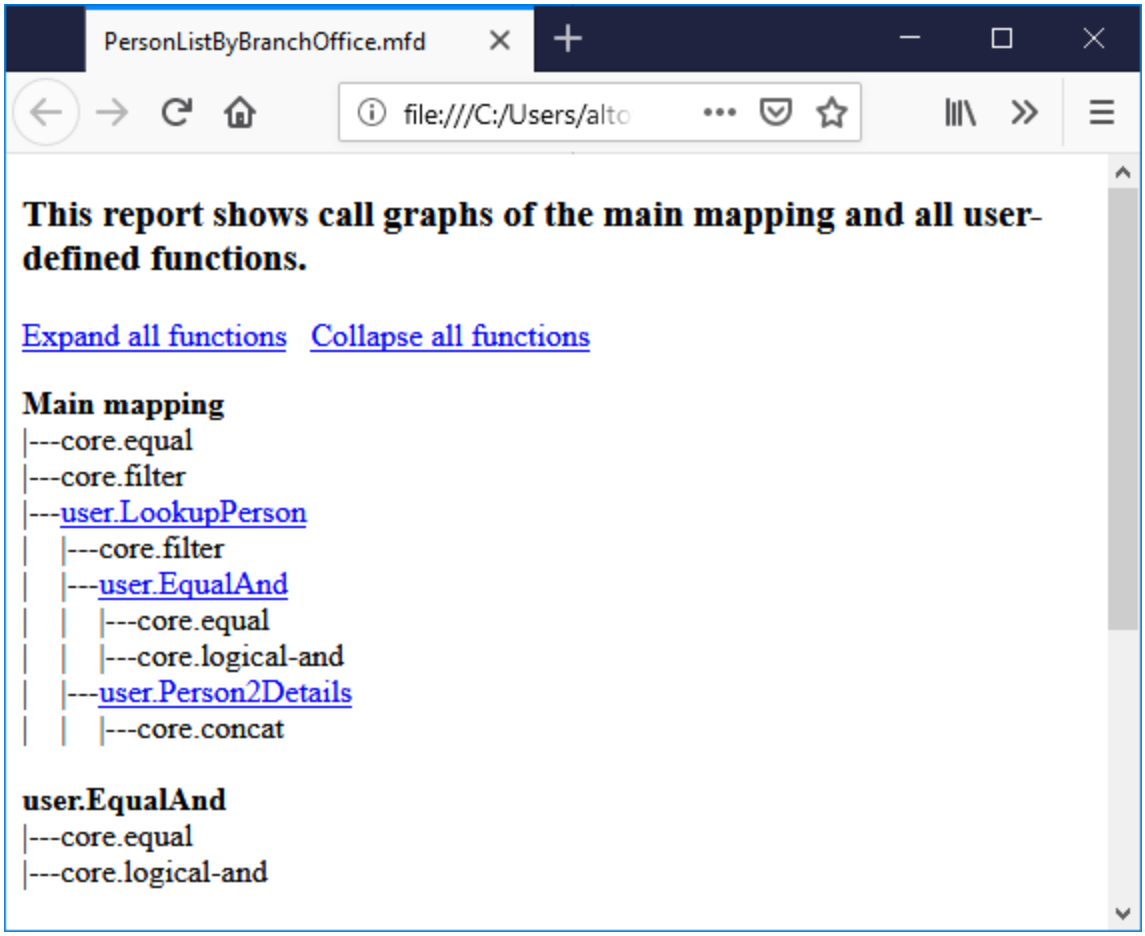
- **ImpactAnalysis.sps**: enumera todos los nodos de origen y de destino, así como la ruta de varias de las funciones al nodo de destino.
- **OverallDocumentation.sps**: muestra todos los nodos, las conexiones, las funciones y los nodos de destino. Esta es la plantilla que más detalles genera y es idéntica al "diseño fijo" integrado.

Puede seleccionar la hoja de estilos que necesite cada vez que vaya a generar documentación, como se muestra más abajo. Los archivos se encuentran en la carpeta **...\MapForce2023\Documentation\MapForce**.



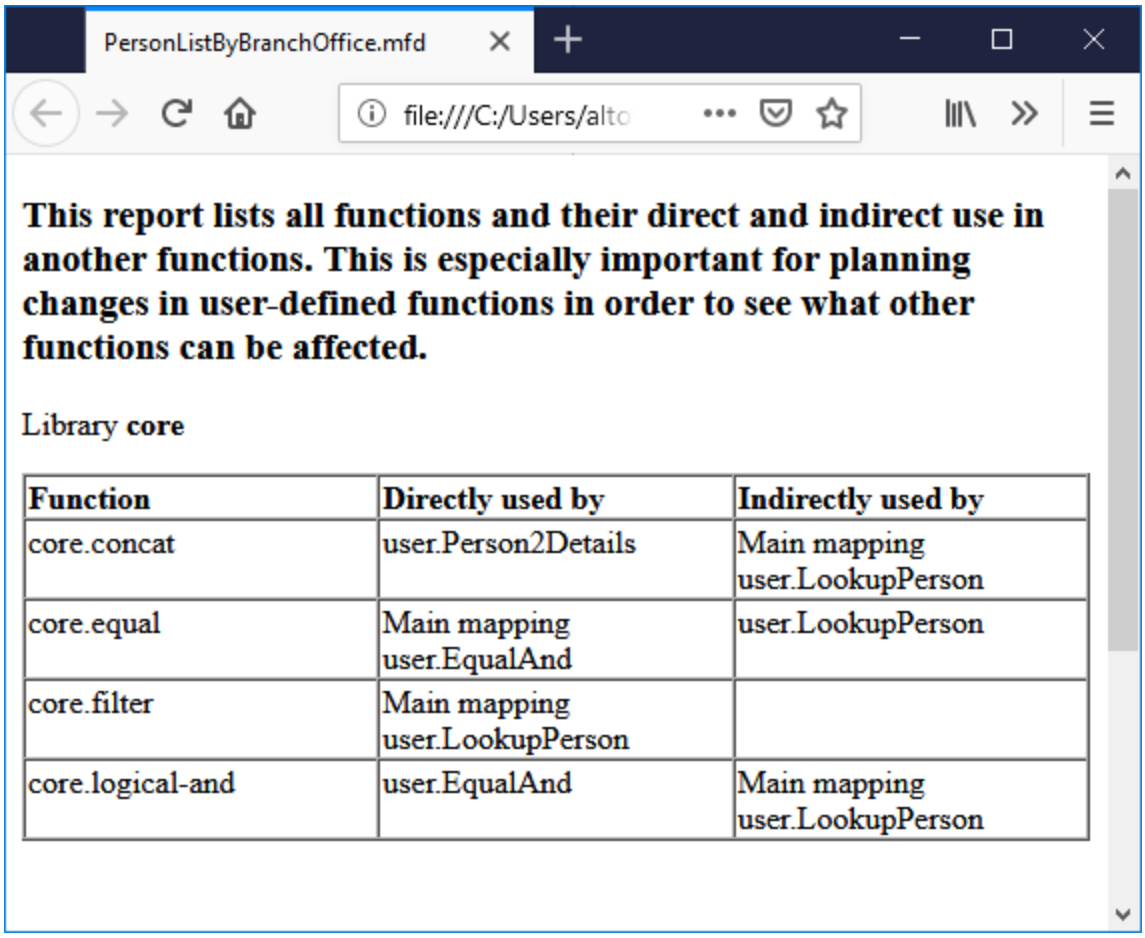
Los ejemplos siguientes ilustran los resultados producidos por cada una de estas hojas de estilos. Los ejemplos se generaron a partir de una de las asignaciones de ejemplo que vienen con MapForce: **PersonListByBranchOffice.mfd**. Aunque estos ejemplos muestran resultados HTML, el diseño de otros formatos es parecido. Para más información sobre cómo crear y personalizar archivos SPS consulte Diseño personalizado 921.

Hoja de estilo "FunctionCallGraph.sps"



© 2016-2022 Altova GmbH

Hoja de estilo "FunctionsUsedBy.sps"

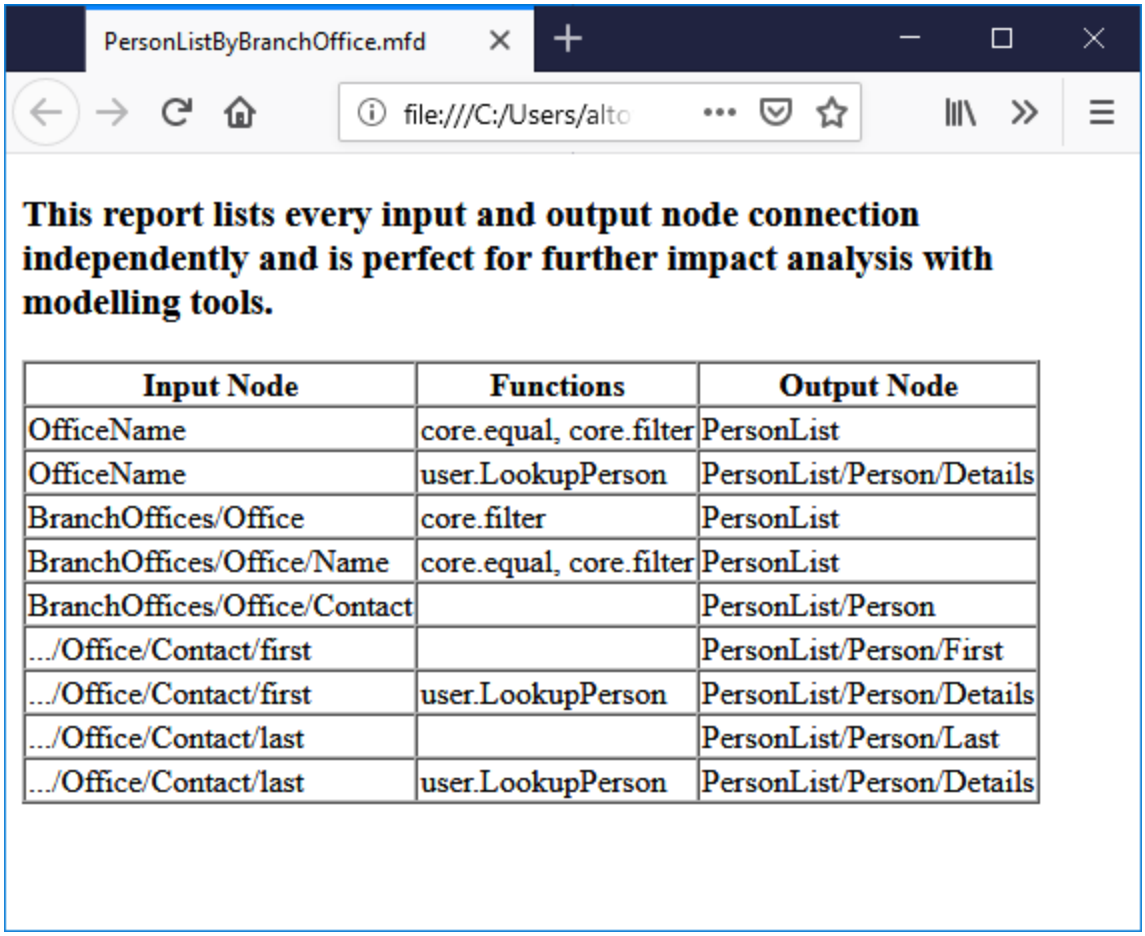


This report lists all functions and their direct and indirect use in another functions. This is especially important for planning changes in user-defined functions in order to see what other functions can be affected.

Library **core**

Function	Directly used by	Indirectly used by
core.concat	user.Person2Details	Main mapping user.LookupPerson
core.equal	Main mapping user.EqualAnd	user.LookupPerson
core.filter	Main mapping user.LookupPerson	
core.logical-and	user.EqualAnd	Main mapping user.LookupPerson

Hoja de estilo "ImpactAnalysis.sps"

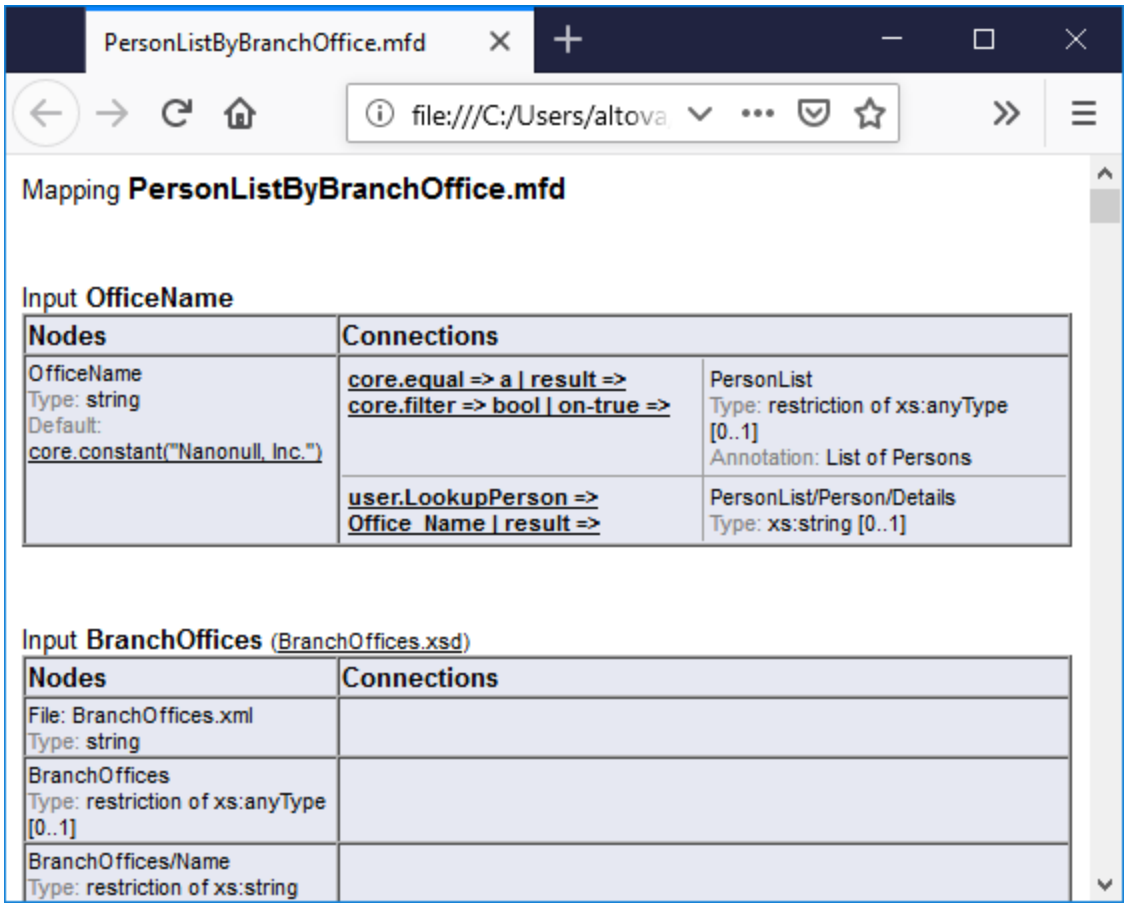


This report lists every input and output node connection independently and is perfect for further impact analysis with modelling tools.

Input Node	Functions	Output Node
OfficeName	core.equal, core.filter	PersonList
OfficeName	user.LookupPerson	PersonList/Person/Details
BranchOffices/Office	core.filter	PersonList
BranchOffices/Office/Name	core.equal, core.filter	PersonList
BranchOffices/Office/Contact		PersonList/Person
.../Office/Contact/first		PersonList/Person/First
.../Office/Contact/first	user.LookupPerson	PersonList/Person/Details
.../Office/Contact/last		PersonList/Person/Last
.../Office/Contact/last	user.LookupPerson	PersonList/Person/Details

© 2016-2022 Altova GmbH

Hoja de estilo "OverallDocumentation.sps"



7.7.2 Diseño personalizado

Además de usar el diseño fijo integrado, también puede crear hojas de estilos personalizadas a partir de la documentación generada a partir de la asignación con StyleVision (https://www.altova.com/es/stylevision). También puede cambiar cualquiera de las hojas de estilos predefinidas 916 ,por ejemplo ajustando el tipo de fuente u otros estilos.

Los archivos SPS (StyleVision Power Stylesheet) son diseños personalizados. La ventaja de usarlos para generar la documentación de la asignación es que puede configurar a su gusto el diseño de la documentación.

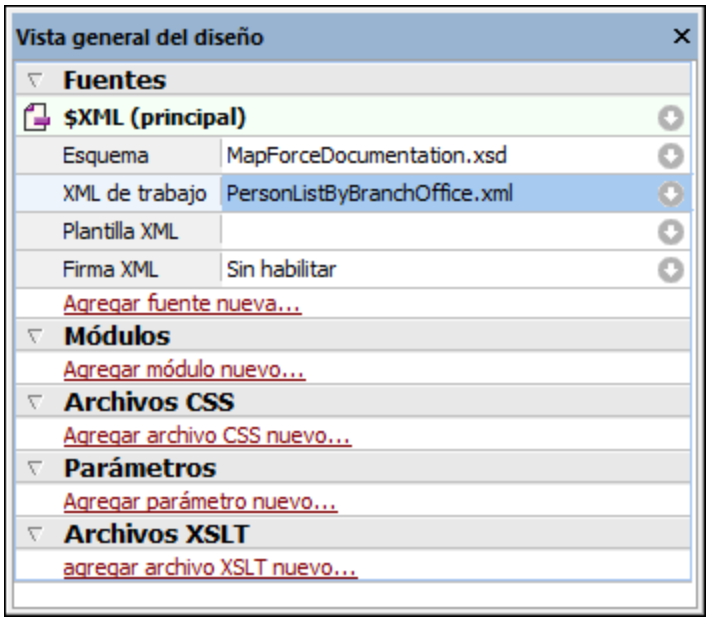
Para crear un archivo SPS personalizado necesita:

1. El esquema XML que determina la estructura de la documentación generada de MapForce. Este esquema se llama **MapForceDocumentation.xsd** y está incluido en los archivos que se instalan con MapForce. Lo encontrará en la carpeta **... \Documents\Altova\MapForce2023\Documentation\MapForce**. Tenga en cuenta que

MapForceDocumentation.xsd incluye el archivo **Documentation.xsd** ubicado en la carpeta de nivel superior a esta.

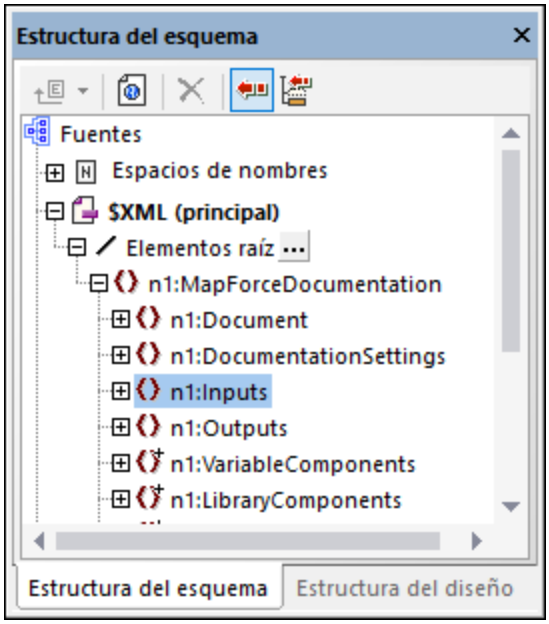
2. Datos de muestra para probar y previsualizar el diseño personalizado. Para ello puede usar este archivo XML: **... \Documentos\Altova\MapForce2023\Documentation\MapForce\SampleData\PersonListByBranchOffice.xml**.

Los archivos que mencionamos más arriba son referenciados desde la ventana Vista general del diseño de StyleVision, por ejemplo:



Para crear un diseño en StyleVision arrastre nodos desde la ventana Estructura del esquema hasta el área del diseño y asígneles estilos y propiedades.

© 2016-2022 Altova GmbH



También puede añadir componentes complementarios, como enlaces o imágenes, al diseño SPS. Para acceder a la vista previa del diseño en un formato en concreto, haga clic en la pestaña correspondiente: **HTML**, **RTF**, **PDF** o **Word 2007+**. Para más detalles consulte la documentación de StyleVision (https://www.altova.com/es/documentation).

8 Depuración de asignaciones

MapForce viene con un depurador de asignaciones integrado para el lenguaje de transformación integrado. El depurador de asignaciones le permitirá:

- Ver y analizar los valores que produce la asignación por cada uno de los conectores.
- Resaltar en la asignación el contexto (conjunto de nodos) responsable de producir un valor determinado.
- Ejecutar paso a paso una asignación de datos para ver cómo se procesa o calcula cada valor en tiempo real y generar una vista previa del resultado de la asignación a medida que se produce.
- Establecer puntos de interrupción en los que se debe detener la ejecución de la asignación y ver los valores que se están procesando en ese momento.
- Ver el historial de valores procesados por un conector desde que se empezó a ejecutar la asignación hasta la posición actual de la ejecución.

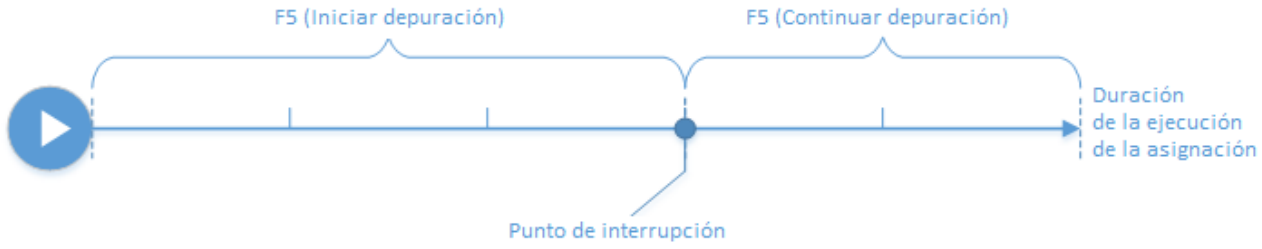
El depurador de asignaciones solamente está disponible si el lenguaje de transformación de la asignación es BUILT-IN. Si comienza a depurar una asignación configurada para otro tipo de lenguaje, MapForce solicitará que cambie el lenguaje por BUILT-IN. También puede convertir una asignación al lenguaje BUILT-IN seleccionando el comando de menú **Resultados | Motor de ejecución integrado**. En ambos casos la conversión a BUILT-IN solo será posible si la asignación carece de componentes incompatibles con el lenguaje BUILT-IN (como las funciones XSLT por ejemplo).

El depurador de MapForce se diferencia de los depuradores tradicionales en que no recorre el código de programa línea por línea (ya que en MapForce no es necesario escribir código). Por el contrario, el depurador de MapForce expone los resultados del código generado a partir de las asignaciones diseñadas. Concretamente, el depurador registra los valores que se pasan entre los componentes de la asignación a través de los conectores de entrada y salida. Los valores registrados se pueden analizar después en la asignación directamente o en ventanas especiales.

A continuación explicamos algunos casos de uso para trabajar con el depurador de asignaciones.

Depuración con puntos de interrupción

Cuando necesite detener la ejecución de la depuración en una posición determinada de la asignación, podrá establecer puntos de interrupción (tal y como se hace en los entornos de desarrollo tradicionales). La diferencia es que en MapForce los puntos de interrupción no se añaden a una línea de código sino a un conector de entrada o salida de un componente de la asignación. También puede añadir condiciones a los puntos de interrupción (lo cual puede ser de gran ayuda si solamente queremos que la ejecución se detenga solamente si se cumple cierta condición).

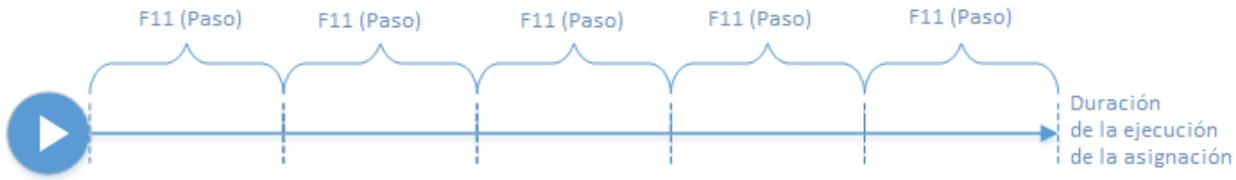


 © 2016-2022 Altova GmbH

Puede definir puntos de interrupción en los conectores pertinentes y ejecutar la asignación hasta que se encuentre el primer punto de interrupción y después continuar hasta el siguiente y así sucesivamente. De este modo podrá analizar el contexto de la asignación y los valores asociados a los conectores elegidos. También podrá acelerar o ralentizar la ejecución con ayuda de los comandos **Paso a paso por instrucciones**, **Paso a paso para salir**, **Paso a paso por procedimientos** y **Paso mínimo** que ofrece el depurador. Estos comandos sirven para pasar por alto partes de la asignación o ejecutar partes de la asignación más detalladamente.

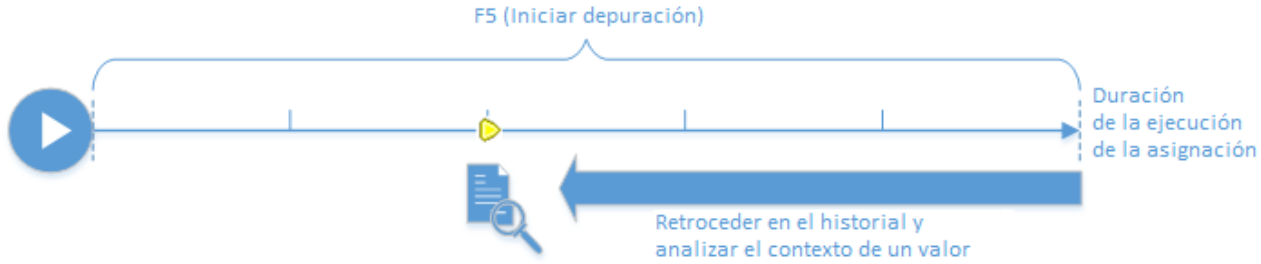
Depuración por pasos

Puede depurar una asignación por pasos y analizar el contexto de la asignación y los valores asociados a cada paso. Al igual que en el tipo de depuración anterior podrá acelerar o ralentizar la ejecución con los comandos **Paso a paso por instrucciones**, **Paso a paso para salir**, **Paso a paso por procedimientos** y **Paso mínimo** que ofrece el depurador.



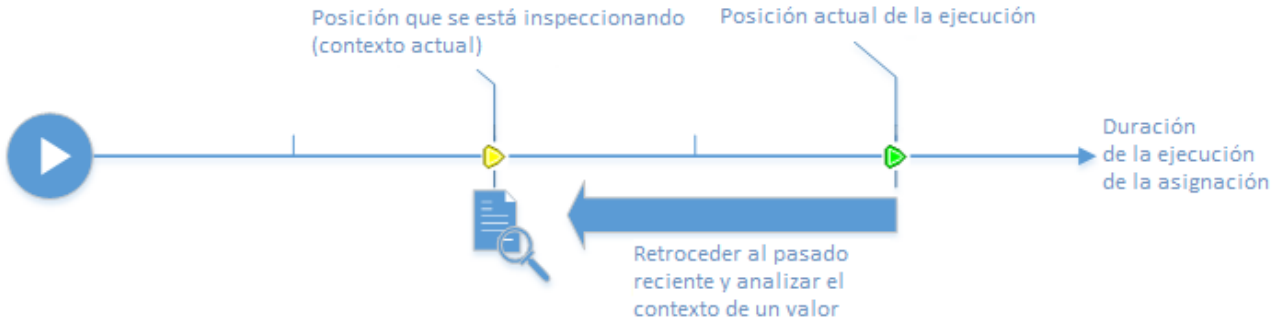
Analizar el registro de valores

Puede configurar MapForce para que recuerde el registro de todos los valores (historial de seguimiento) que fueron procesados por todos lo conectores mientras depura una asignación de datos. El historial de seguimiento puede deshabilitarse ya que puede no ser una opción adecuada para las asignaciones que hagan un uso intensivo de lo datos. Si se deshabilita el historial de seguimiento, podrá analizar el registro de todos los valores procesados por cada conector hasta la posición actual de la ejecución. También puede configurar MapForce para que recree el contexto de asignación asociado con un valor determinado (lo cual puede ayudarle a comprender el motivo por el que se produjo dicho valor).



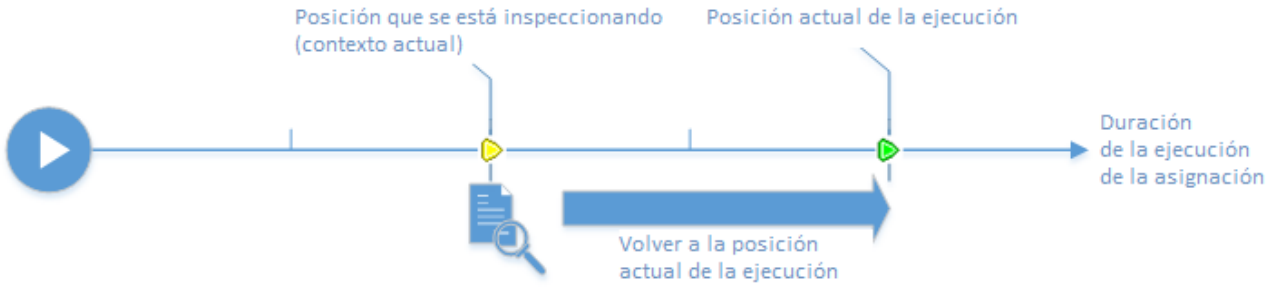
Establecer el contexto en un valor relacionado con la posición actual de la ejecución

Cuando el depurador se encuentra en una posición determinada de la ejecución, se puede analizar el contexto de un valor pasado relacionado con la posición actual de la ejecución (como si se retrocediera un poco en la ejecución):



El sentido del contexto es explicar por qué se calcula un valor. En otras palabras, describe cómo se generó un valor determinado de la asignación. El contexto suele ser la posición actual de la ejecución, aunque también puede ser un contexto del pasado reciente que MapForce permite definir. Cuando el contexto se establece en un valor determinado, MapForce resalta en la asignación directamente los nodos que son relevantes, ofrece consejos junto a los conectores y expone información adicional en las ventanas del depurador (ventanas Valores, Contexto y Puntos de interrupción).

Cuando termine de inspeccionar un contexto de la asignación que no sea igual a la posición actual de la ejecución, podrá restablecer el contexto en la posición actual de la ejecución:



Restricciones

- Cuando MapForce ejecuta una asignación, a veces optimiza el código de forma interna (por ejemplo, almacenando datos en caché o calculando resultados intermedios en puntos aleatorios). Esto puede provocar que sea imposible acceder a ciertos conectores (y, por tanto, a ciertos puntos de interrupción) durante la depuración. Cuando esto ocurre, MapForce muestra una notificación. Nota: las optimizaciones de código de MapForce (y por consiguiente el comportamiento que expone el depurador) pueden diferir en las diferentes versiones del software aunque el resultado de la asignación sea el mismo.
- El depurador puede depurar la generación resultante para cada componente de destino uno a uno. Si la asignación incluye varios componentes de destino, deberá seleccionar qué componente de destino se ejecuta en el depurador.
- Actualmente la depuración no es compatible con las acciones de tabla de BD (como "Insertar todo", "Actualizar si", etc.) de los componentes de base de datos.
- No se pueden establecer puntos de interrupción en estas entidades: constantes, la función `core | position`, elementos descendientes de las conexiones de copia total y parámetros de funciones definidas por el usuario insertadas.

 © 2016-2022 Altova GmbH

8.1 Preparar la depuración

Preparar la depuración puede ser fundamental en asignaciones masivas de datos que puedan necesitar mucha memoria de sistema a la hora de ejecutarse. Nos referimos a las asignaciones de datos que procesan archivos de entrada o salida de gran tamaño o que recorren reiteradamente colecciones de datos de gran tamaño.

Para completar la depuración más rápido y reducir el uso de memoria, recomendamos seguir estos pasos antes de iniciar la depuración:

- Si se trata de una asignación compleja, elimine o desconecte las partes de la asignación que no necesita depurar.
- Si la asignación utiliza archivos de entrada de gran tamaño, sustitúyalos por archivos de menor tamaño.
- Asegúrese de deshabilitar la opción **Guardar historial de seguimiento completo** (*véase Configurar el depurador* 1051).

Además, para garantizar que el depurador analiza los resultados correctos deberá repasar algunos aspectos:

- Si la asignación tiene varios componentes de destino, seleccione el componente de destino que desea depurar haciendo clic en el botón **Vista previa** ( ).
- Si la asignación es una asignación en cadena (*véase Asignaciones en cadena* 840), desactive el botón **Paso a través** ( ) del componente intermedio. MapForce no permite depurar componentes de paso a través.

Por último, tiene la opción de agregar puntos de interrupción en conectores importantes cuyos valores desee analizar. El depurador se detendrá en estos conectores a medida que avance por la asignación (*véase Agregar y quitar puntos de interrupción* 931).

8.2 Información sobre el modo depurador

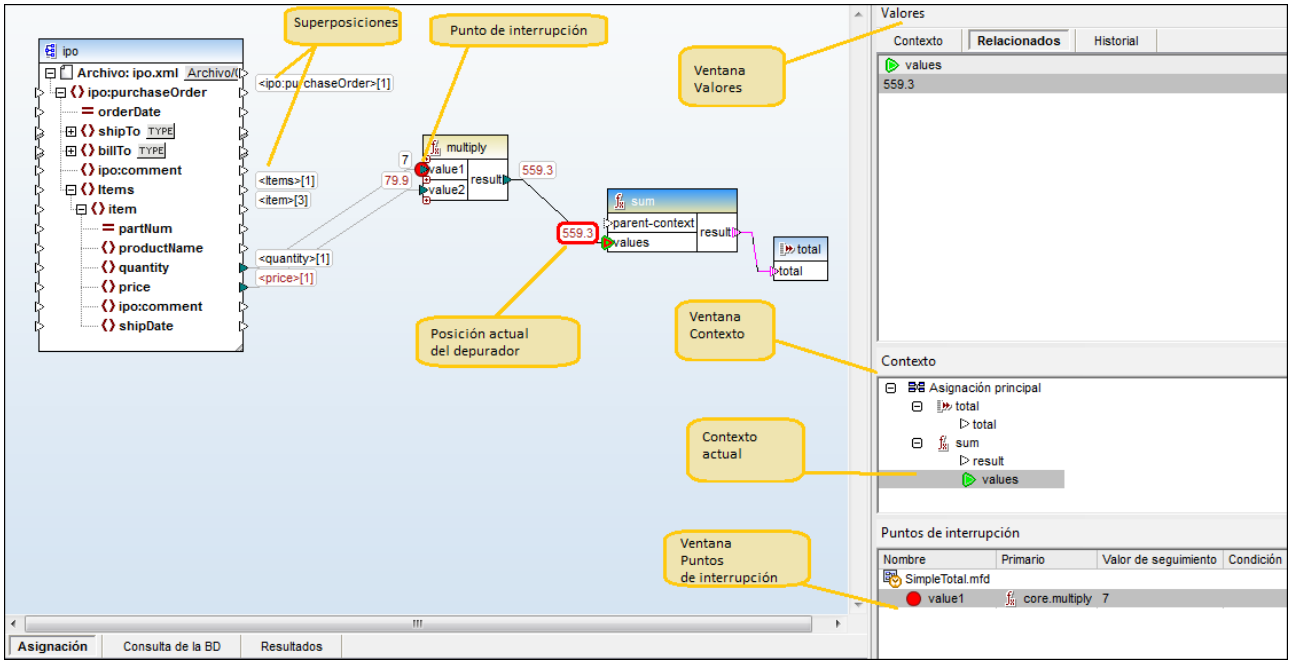
Cuando se inicia la depuración (pulsando **F5**, **F11** o **Ctrl + F11**), MapForce ejecuta la asignación en modo depurador.

> Mientras MapForce esté en modo depurador, la asignación es de solo lectura. Aunque los componentes se pueden mover por el área de asignación, la mayoría de los comandos estarán deshabilitados. Entre ellos los comandos de validación e implementación, de generación de código y documentación y los comandos para agregar componentes nuevos al área de asignación o para volver a cargar los componentes actuales.

El modo depurador permite analizar el contexto responsable de producir un valor determinado. La información que produce el depurador aparece en la asignación directamente, pero también en las ventanas Valores, Contexto y Puntos de interrupción. Estas tres ventanas se abren por defecto nada más iniciar la depuración y desaparecen cuando la depuración se detiene.

MapForce estará en modo depurador (y la asignación será de solo lectura) hasta que se detenga la depuración, pulsando **Mayús + F5** (o haciendo clic en el botón **Detener depuración** de la barra de herramientas).

En la siguiente imagen puede ver una asignación que se está depurando por pasos (pulsando **F11** para avanzar por los pasos de la asignación). La imagen corresponde a la asignación de muestra **SimpleTotal.mfd**, disponible en el directorio **<Documentos>\Altova\MapForce2023\MapForceExamples**.



Entorno de desarrollo MapForce en modo depurador

A continuación describimos las características de la interfaz del depurador de MapForce.

 © 2016-2022 Altova GmbH

Panel de asignación

Durante la depuración el panel de asignación incluye información adicional:

- Superposiciones de datos (ver más abajo) que muestran el valor actual y valores relacionados junto a cada conector.
- El contexto actual (la estructura que aparece en la ventana Contexto) se resalta de la siguiente manera:
  - Los conectores que están dentro de contexto son de rayas magenta ( ).
  - Los conectores que están en contexto ambiguo son de puntos magenta ( ).
  - Las conexiones que están dentro de contexto son de rayas magenta.
  - Las conexiones que están en contexto ambiguo son de rayas magenta más claro.
- La ubicación actual de la ejecución aparece marcada con un icono de conector verde ( ).

Superposiciones de datos

Los valores que procesa cada conector aparecen representados en forma de superposiciones de datos (rectángulos pequeños) junto al conector correspondiente. Una superposición de datos que está seleccionada tiene el borde sólido y rojo. Los valores que se modificaran desde el último paso aparecen en rojo oscuro. Para los nodos con contenido simple la superposición de datos combina dos valores: el nombre del nodo y el valor. Si el nombre de nodo se recorrió varias veces antes de la posición actual de la ejecución, el índice de la iteración actual se indica con un número que aparece entre corchetes.

Las superposiciones de datos funcionan de la siguiente manera:

- Si se pasa el puntero por encima de una superposición de datos, esta pasa a primer plano temporalmente. Si se hace clic en la superposición, esta pasa a primer plano permanentemente y además se selecciona el conector correspondiente.
- Pueden arrastrarse y colocarse en otra posición.
- Se mueven cuando se mueve un componente. Por tanto, si aparecen apiladas porque los componentes están demasiado cerca unos de otros, arrastre los componentes hasta otra posición y las superposiciones se moverán junto con su componente.
- Si se hace clic en una superposición de datos, su valor aparece en la ventana Valores.
- Si se hace clic en un conector también se selecciona su superposición de datos.

Puntos de interrupción

Los puntos de interrupción son acontecimientos señalados en los que la asignación debe detenerse durante la ejecución en modo depurador. Este término es utilizado en otros entornos de desarrollo integrado. Pero si en los entornos de desarrollo los puntos de interrupción se añaden en líneas de código, en MapForce se pueden añadir en conectores de entrada o salida (los iconos en forma de triángulo situados en los extremos de las conexiones). En el panel de asignación los puntos de interrupción se representan en forma de círculos rojos. Además, los puntos de interrupción definidos aparecen en la ventana Puntos de interrupción. Consulte el apartado Agregar y quitar puntos de interrupción 931 para obtener más información.

Posición actual del depurador

El triángulo verde ( ) indica la posición actual del depurador. Esta posición puede ser un conector de entrada o salida de cualquier componente.

El valor que se esté procesando en ese momento también aparece en la pestaña *Contexto* de la ventana Valores.

El conjunto de conexiones y conectores de rayas magenta indican el contexto actual de la asignación. La misma información aparece representada de forma jerárquica en la ventana Contexto (véase el apartado Ventana Contexto 936).

Cuando defina a mano el contexto de un valor, la posición actual del depurador será una posición del pasado en relación a la posición más actual de la ejecución. Para ayudarle a distinguir entre la posición más actual de la ejecución y la posición del pasado, el conector de la *posición actual* aparece con estos colores en la interfaz del depurador.

	El triángulo verde es *el presente*. Indica la posición actual de la ejecución (*véase Ver el valor actual de un conector* 941).
	El triángulo amarillo es *el pasado*. Indica que se trata de un conector del pasado, relativo a la posición actual de la ejecución. Esto puede ocurrir después de definirse un contexto a mano (*véase Establecer un contexto en un valor* 945).

Ventana Valores

La ventana Valores ofrece información sobre los valores que procesa la asignación. Permite ver qué procesa la asignación en la posición actual de la ejecución o en un contexto determinado que se puede establecer a mano. Consulte el apartado Ventana Valores 934 para obtener más información.

Ventana Contexto

La ventana Contexto ofrece una vista jerárquica del conjunto de nodos y funciones que son relevantes para la posición actual del depurador. Consulte el apartado Ventana Contexto 936 para obtener más información.

Ventana Puntos de interrupción

La ventana Puntos de interrupción muestra la lista de puntos de interrupción que se crearon desde que se inició MapForce. Si definió puntos de interrupción en varias asignaciones diferentes, todos ellos aparecerán en esta ventana. Consulte el apartado Ventana Puntos de interrupción 938 para obtener más información.

© 2016-2022 Altova GmbH

8.3 Agregar y quitar puntos de interrupción

Los puntos de interrupción son acontecimientos señalados en los que la asignación se debe interrumpir durante la ejecución en modo depurador. Los puntos de interrupción creados por el usuario se almacenan de forma global para todas la asignaciones y pueden consultarse en la ventana Puntos de interrupción. Los puntos de interrupción definidos son válidos hasta que el usuario los elimina o hasta que se cierra MapForce

Nota: no se pueden agregar puntos de interrupción en estas entidades: constantes, funciones `core | position`, elementos descendientes de conexiones de copia total y parámetros de funciones definidas por el usuario inline.

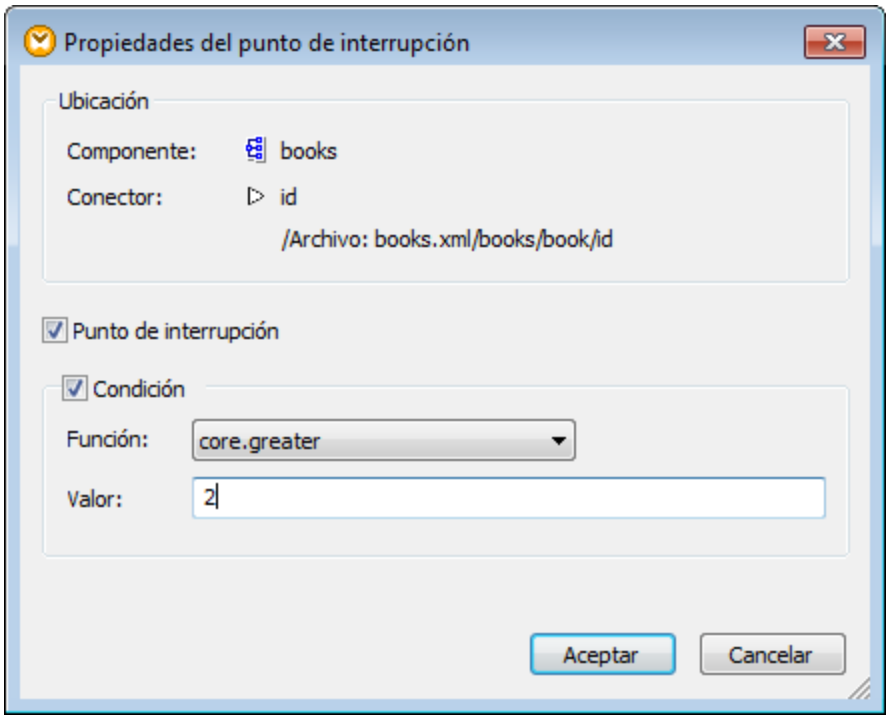
Los puntos de interrupción pueden ser simples o condicionales. Los simples detienen la ejecución de la asignación sin condiciones. Los condicionales detienen la ejecución de la asignación cuando se cumple una condición asignada previamente al punto de interrupción. Las condiciones pueden ser funciones de la biblioteca integrada de MapForce a las que se aportan valores personalizados. En otras palabras, si la condición devuelve el valor `true`, el punto de interrupción detendrá la ejecución de la asignación.

Hay dos maneras de crear puntos de interrupción simples:

- Haciendo clic con el botón derecho en un conector de entrada o salida (los triángulos situados a la izquierda o derecha de un componente) y seleccionando **Punto de interrupción del depurador** en el menú contextual.
- Haciendo clic en un conector de entrada o salida y pulsando **F9**.

Para crear un punto de interrupción condicional:

1. Haga clic con el botón derecho en un conector y seleccione **Propiedades del punto de interrupción** en el menú contextual.



2. En el cuadro de diálogo que aparece marque las casillas *Punto de interrupción* y *Condición*.
3. Seleccione la función pertinente en el cuadro combinado *Función* e introduzca el valor de la función (si procede). Por ejemplo, en la imagen anterior se definió un punto de interrupción que detendrá la ejecución de la asignación si el valor que pasa por el punto es superior a 2.

> Si el tipo de datos del conector donde se añade el punto de interrupción condicional no coincide con los tipos esperados por la función, MapForce intentará convertir el tipo de datos automáticamente. Si la conversión automática no es posible, la ejecución de la asignación fallará. Para evitarlo, asegúrese de usar tipos de datos compatibles. Por ejemplo, la función `core.starts-with` espera un valor de cadena, de modo que el conector del punto de interrupción debería tener el mismo tipo.

Quitar puntos de interrupción

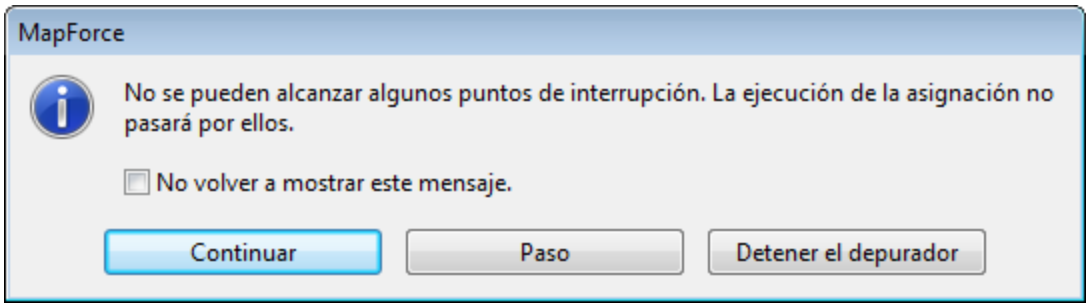
Para quitar un punto de interrupción haga clic con el botón derecho en el conector que lo contiene y seleccione **Punto de interrupción del depurador**. Otra opción es hacer clic en el conector de entrada o salida donde está definido el punto de interrupción y pulsar **F9**.

También puede eliminar puntos de interrupción desde la ventana Puntos de interrupción 938.

Puntos de interrupción inalcanzables

En algunos casos MapForce emite este mensaje:

 © 2016-2022 Altova GmbH



Este mensaje indica que el depurador no puede alcanzar algunos puntos de interrupción. Esto puede deberse a dos motivos distintos:

- Se definió un punto de interrupción en un conector que no participa en la asignación.
- MapForce no puede alcanzar el punto de interrupción debido a optimizaciones de la ejecución (consulte la información sobre restricciones 926).

Haga clic en Continuar para proceder al siguiente punto de interrupción (o para llegar al final de la ejecución de depuración). Haga clic en **Paso** para iniciar la depuración por pasos.

Este tipo de mensajes se puede deshabilitar marcando la casilla *No volver a mostrar este mensaje* o siguiendo estas instrucciones:

1. Haga clic en el comando de menú **Herramientas | Opciones**.
2. Haga clic en *Mensajes*.
3. Desactive la casilla *Informar sobre puntos de interrupción que no se puedan alcanzar*.

8.4 Ventana Valores

La ventana Valores ofrece información sobre los valores que procesa la asignación cuando MapForce está en modo depurador. La información que aparece en esta ventana dependerá de la posición actual del depurador y de los elementos de la interfaz en los que haga clic el usuario. Estas son las pestañas de la ventana Valores:

Pestaña *Contexto*

La pestaña *Contexto* muestra el valor que se está procesando actualmente (el mismo valor cuyo contexto aparece en la ventana Contexto). Se trata del valor que está en la posición actual de la ejecución del depurador o del valor de un conector que se procesó en el pasado. Para ayudarle a distinguir entre estos dos tipos de valores, MapForce usa iconos de colores distintos:

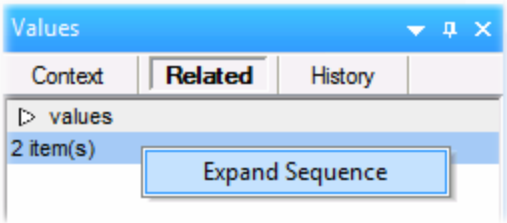
	El triángulo verde es *el presente*. Indica la posición actual de la ejecución (*véase* *Ver el valor actual de un conector* 941).
	El triángulo amarillo es *el pasado*. Indica que se trata de un conector del pasado, relativo a la posición actual de la ejecución. Esto puede ocurrir después de definirse un contexto a mano (*véase* *Establecer un contexto en un valor* 945).

Pestaña *Relacionados*

La pestaña *Relacionados* muestra los valores que están relacionados con el valor que se está procesando (o que representan su pasado reciente). En general no necesitará hacer clic en esta pestaña porque MapForce cambia a esta pestaña automáticamente cuando se hace clic en la superposición de datos de un conector que está relacionado con la posición actual de la ejecución del depurador (*véase* *Retroceder en el pasado reciente* 942).

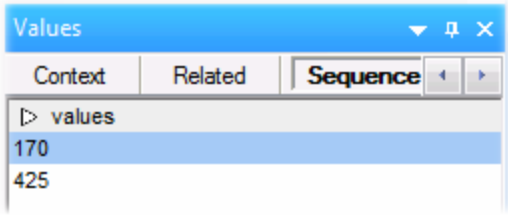
Pestaña *Secuencia*

Cuando está presente, esta pestaña sirve para acceder a los valores de un conector que procesa una secuencia. Esta pestaña solamente está presente si un conector procesó una secuencia de elementos (por ejemplo, una función de agregado como `sum` o `count`). Cuando se hace clic en la superposición de datos de un conector que procesó una secuencia de elementos, la ventana Valores incluirá la entrada "`n elementos`", siendo `n` el número de elementos que procesa el conector. Para acceder a cada valor, haga doble clic en esta entrada (o clic con el botón derecho y seleccione **Expandir secuencia** en el menú contextual).



Los valores aparecen después en la pestaña *Secuencia*.

© 2016-2022 Altova GmbH



Pestaña *Historial*

La pestaña *Historial* muestra valores procesados por un nodo determinado desde que se inició la depuración y hasta que se llegó a la posición actual de la ejecución (*véase* *Ver historial de valores procesados por un conector* 943).

8.5 Ventana Contexto

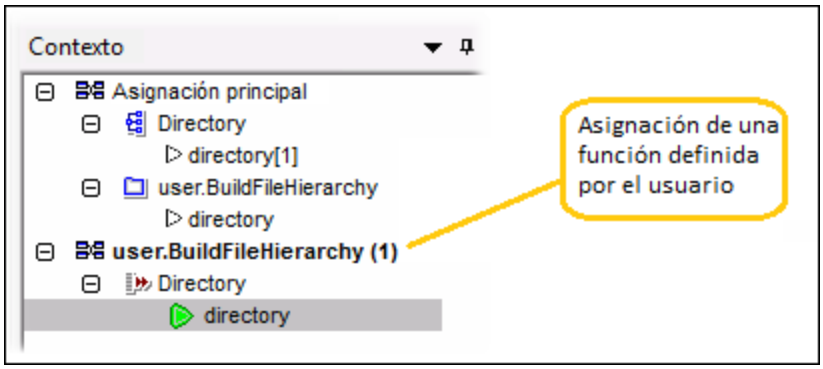
Cuando MapForce está en modo depurador, la ventana Contexto muestra una estructura de conectores que corresponden a la posición actual del depurador. En otras palabras, ofrece el contexto de asignación responsable de producir el valor actual de la asignación.

MapForce genera el contexto actual de la siguiente manera:

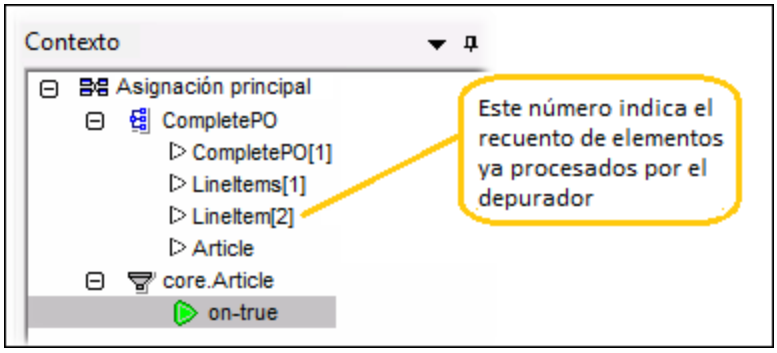
1. Empezar por el nodo raíz de la estructura de destino.
2. Descender hasta el nodo de destino actual.
3. Desde el nodo de destino actual, avanzar hacia la izquierda por los componentes que conduzcan hasta la posición actual. Estos componentes pueden ser filtros o componentes de ordenación, funciones definidas por el usuario o integradas, variables, etc.

La ventana Contexto ofrece información pero también es un asistente para la navegación. Para seleccionar un nodo determinado de la asignación desde el contexto actual directamente, haga clic con el botón derecho en el nodo en la ventana Contexto y elija **Seleccionar en la asignación**. Con esto puede evitar tener que desplazarse todo el rato por la asignación, si se trata de un diseño muy grande.

La ventana Contexto puede incluir estos iconos y anotaciones:

Icono	Descripción
	Representa la asignación a la que pertenece el contexto. Puede ser la asignación principal o la asignación de una función definida por el usuario. 
	Representa un conector. Los nodos de destino procesados hasta el momento aparecen entre corchetes.

© 2016-2022 Altova GmbH

Icono	Descripción
	
	Representa el conector actual (la posición más reciente de la ejecución). Se trata del origen del valor actual que aparece en la ventana Valores. En casos muy raros, es posible que un valor calculado se utilice para varios conectores. Cuando esto ocurra, podrían aparecer varios iconos verdes.
	Representa el conector actual cuando el depurador está en una posición del pasado relativo al puesto más reciente de la ejecución. Esto puede ocurrir después de que se establezca el contexto en un valor (*véase* *Establecer el contexto en un valor* 945).

Además de estos iconos y anotaciones, la ventana Contexto incluye los iconos estándar correspondientes a los tipos de componentes presentes en la asignación de datos.

Ventana Contexto y funciones definidas por el usuario

Si el contexto actual incluye funciones definidas por el usuario, éstas aparecerán también en la ventana Contexto. Recuerde que si el contexto actual es para calcular un valor de entrada de una función definida por el usuario, entonces el contexto se determinará de la siguiente manera:

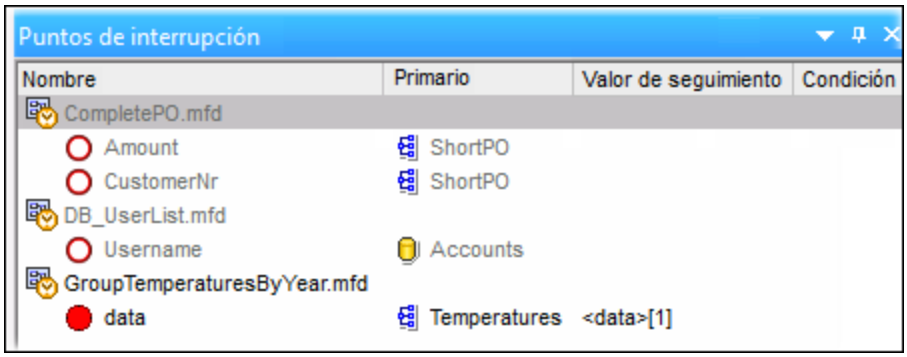
1. Desde el destino del conector de salida de la función definida por el usuario hasta el conector de entrada de la función definida por el usuario.
2. Desde esa posición hacia la izquierda.

Nota: en el contexto se puede dar la misma función definida por el usuario varias veces. Esto ocurre porque hay varias llamadas a función encadenadas o porque la función definida por el usuario está definida como recursiva.

8.6 Ventana Puntos de interrupción

La ventana Puntos de interrupción permite consultar y gestionar los puntos de interrupción de forma global. Esta ventana aparece en cuanto MapForce entra en el modo depurador. Para poder ver la ventana Puntos de interrupción en todo momento, seleccione el comando de menú **Vista | Ventanas del depurador | Puntos de interrupción**.

Esta ventana muestra todos los puntos de interrupción creados desde que se inició MapForce y agrupados según el archivo de asignación al que pertenecen. Mientras esté en ejecución, MapForce recuerda todos los puntos de interrupción asociados con cada asignación y los muestra en la ventana Puntos de interrupción (aunque cierre una asignación, sus puntos de interrupción siguen apareciendo en la ventana). La asignación que se esté depurando en ese momento aparece en el color de texto estándar, mientras que el resto de asignaciones (las que están cerradas o inactivas) aparecen atenuadas.



Haga doble clic en una asignación de la ventana Puntos de interrupción (o en sus puntos de interrupción) para abrirla rápidamente.

Nota importante: cuando cierre o reinicie MapForce, se eliminarán todos los puntos de interrupción.

La información de la ventana Puntos de interrupción está organizada en cuatro columnas:

Columna	Descripción
Nombre	El nombre del nodo al que pertenece el punto de interrupción.
Primario	El nombre del componente de asignación al que pertenece el punto de interrupción.
Valor de seguimiento	El valor que pasa por el conector donde está situado el punto de interrupción. El valor de seguimiento aparece durante la ejecución de depuración.
Condición	Si el punto de interrupción es condicional, muestra la condición que se debe cumplir.

Estos son los iconos que se utilizan para describir los puntos de interrupción en la ventana:

 © 2016-2022 Altova GmbH

Icono	Descripción.
	Punto de interrupción **activo**: denota un punto de interrupción de la asignación que se está depurando en ese momento.
	Punto de interrupción **inactivo**: denota un punto de interrupción de una asignación que está abierta, pero que no se está depurando en ese momento.
	Punto de interrupción **inaccesible**: denota un punto de interrupción que el depurador no puede alcanzar.
	Punto de interrupción **condicional**: denota un punto de interrupción que tiene adjunta una condición.

Para ver o cambiar las propiedades de un punto de interrupción:

- Haga clic con el botón derecho en el punto de interrupción y elija **Propiedades del punto de interrupción** en el menú contextual.

Para quitar un punto de interrupción:

- Haga clic con el botón derecho en el punto de interrupción y elija **Eliminar punto de interrupción** en el menú contextual.
- Haga clic en el punto de interrupción y pulse **Supr**.

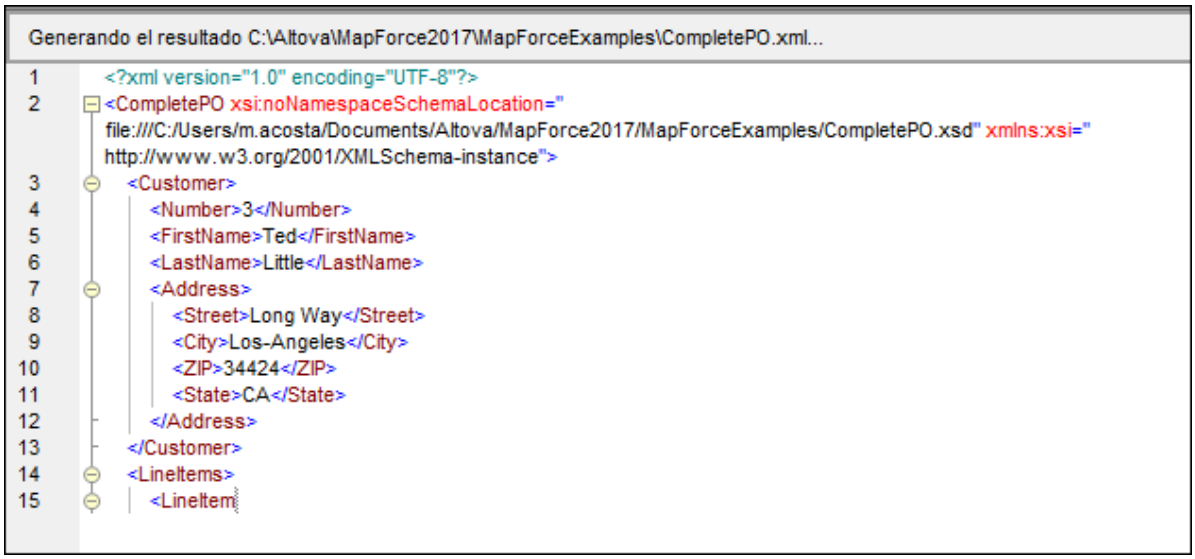
El comando del menú contextual **Eliminar todos los puntos de interrupción** quita todos los puntos de interrupción de la ventana, independientemente de la asignación de datos a la que pertenezcan.

Consulte también el apartado Agregar y quitar puntos de interrupción 931.

8.7 Vista previa parcial de resultados generados

Cuando se depura una asignación por pasos o por medio de puntos de interrupción, se puede ver el resultado generado hasta la posición actual del depurador. La vista previa de resultados parcialmente generados es compatible con componentes de destino XML, EDI y de texto plano.

Cuando pulse **F5** (sin haber definido ningún punto de interrupción), MapForce ejecuta por defecto toda la asignación en modo depurador, después cambia a el panel *Resultados* y muestra el resultado final generado. Sin embargo, si define puntos de interrupción o depura la asignación por pasos (con **F11** o **Ctrl + F11**), la ejecución del depurador se interrumpe mientras se genera el resultado de la asignación. Aunque llegados a este punto el resultado de la asignación solamente es parcial, podrá hacer clic en el panel *Resultados* y obtener una vista previa.



Restricciones

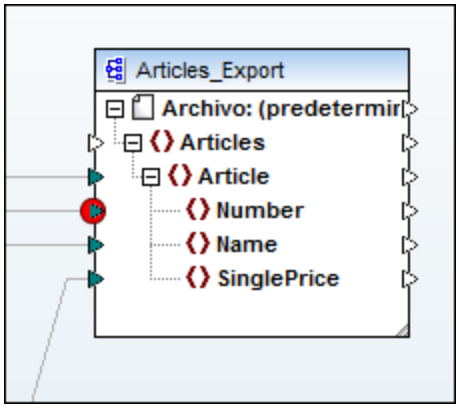
- El nodo de destino que se está calculando en ese momento no siempre aparece en el panel *Resultados*. Por ejemplo, los atributos XML se recopilan de forma interna y se escriben de una sola vez.
- Si el resultado produce varios archivos de salida, solamente se puede obtener una vista previa del archivo que se escribe en ese momento dado. Es decir, está deshabilitada la opción de cambiar a otro archivo de salida.

© 2016-2022 Altova GmbH

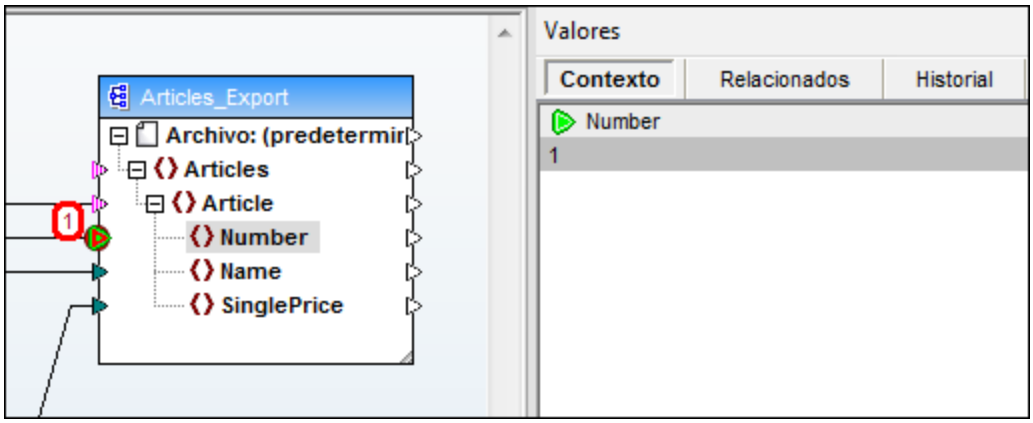
8.8 Ver el valor actual de un conector

Cuando la posición actual de la ejecución del depurador ( ) está en determinado conector (porque se depure por pasos o porque haya definido un punto de interrupción en el conector), el valor que está procesando en ese momento el conector aparece en la pestaña *Contexto* de la ventana Valores. Este valor es el que está a punto de escribirse en el resultado, es decir, *el presente.* También es el valor cuyo contexto aparece en la ventana Contexto 936.

Para comprender mejor este funcionamiento del depurador abra la asignación **PreserveFormatting.mfd** del directorio **<Documentos>\Altova\MapForce2023\MapForceExamples**. Haga clic en el conector de entrada del nodo `Number` del componente de destino y pulse **F9** para agregar un punto de interrupción en este conector.



Ahora pulse **F5** para iniciar la depuración y observar qué resultados se obtienen.



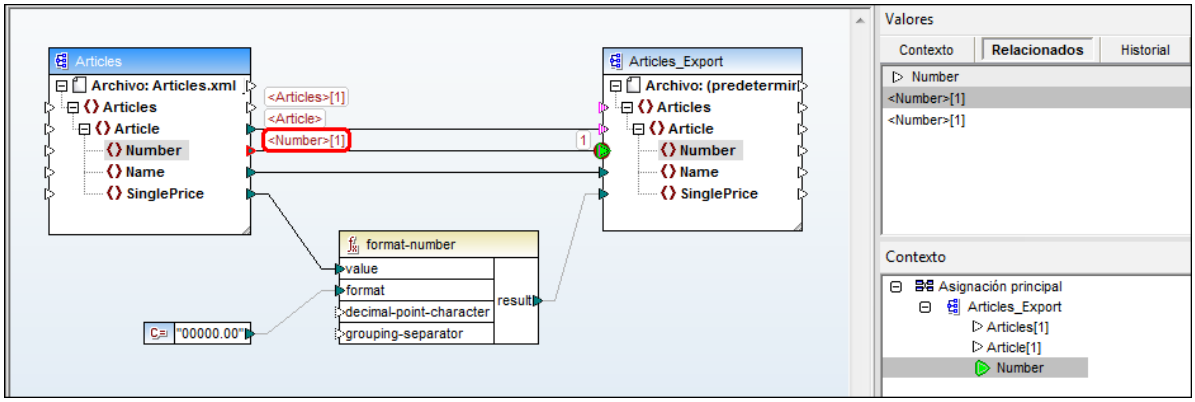
Tal y como se aprecia en la imagen anterior, la posición actual del depurador (y el punto de interrupción) está en el nodo `Number` del componente de destino. La ventana Valores indica que este nodo procesa el valor "1" (este valor también se resalta con un borde grueso rojo en la asignación).

8.9 Retroceder en el pasado reciente

Cuando se hace clic en una superposición de datos (los pequeños recuadros de la asignación) situada junto a un conector, en la ventana Valores aparece el nombre y, de forma opcional, el valor asociados al conector seleccionado. Ahora lo que importa no es la posición actual del depurador, sino la superposición de datos seleccionada. Esta vista puede entenderse como una manera de retroceder ligeramente en el historial de depuración. Se trata del *pasado reciente* porque la asignación muestra superposiciones de datos para los conectores recientes relacionados con la posición actual del depurador. Cuando se hace clic en una superposición de datos "relacionada", se abre automáticamente la pestaña *Relacionados* de la ventana Valores.

Para comprender mejor este funcionamiento del depurador abra la asignación **PreserveFormatting.mfd** del directorio **<Documentos>\Altova\MapForce2023\MapForceExamples**.

Haga clic en el conector situado junto al nodo `Number` del componente de destino y pulse **F9** para agregar un punto de interrupción. Acto seguido pulse **F5** para iniciar la depuración. Después haga clic en la superposición de datos (el pequeño recuadro) situada junto al nodo `Number` del componente de origen.



El depurador suele recorrer un conector varias veces mientras dure la asignación. Por este motivo, el índice actual de la iteración aparece entre corchetes (en la ventana Valores, en la superposición de datos y en la ventana Contexto): `<Number>[1]`. Además, como el conector lleva un valor, su valor también se representa tras el signo igual: **<Number>[1]=1**. En la ventana Valores aparece el mismo valor en una fila nueva (*imagen anterior*).

Si necesita conocer más información sobre un valor determinado, recuerde que puede recrear el contexto que lo produjo (*véase* *Establecer el contexto en un valor* 945).

 © 2016-2022 Altova GmbH

8.10 Ver historial de valores procesados por un conector

Si habilitó la opción *Guardar historial de seguimiento completo* (*véase* *Configurar el depurador* 1051), podrá ver un historial con todos los valores que procesó cada conector (hasta la posición actual de la ejecución).

El historial aparece cuando se hace clic en un conector y después en la pestaña *Historial* de la ventana Valores. Recuerde que esta operación solo es relevante si se trata de conectores que procesaron valores desde el comienzo de la ejecución hasta la posición actual de la depuración.

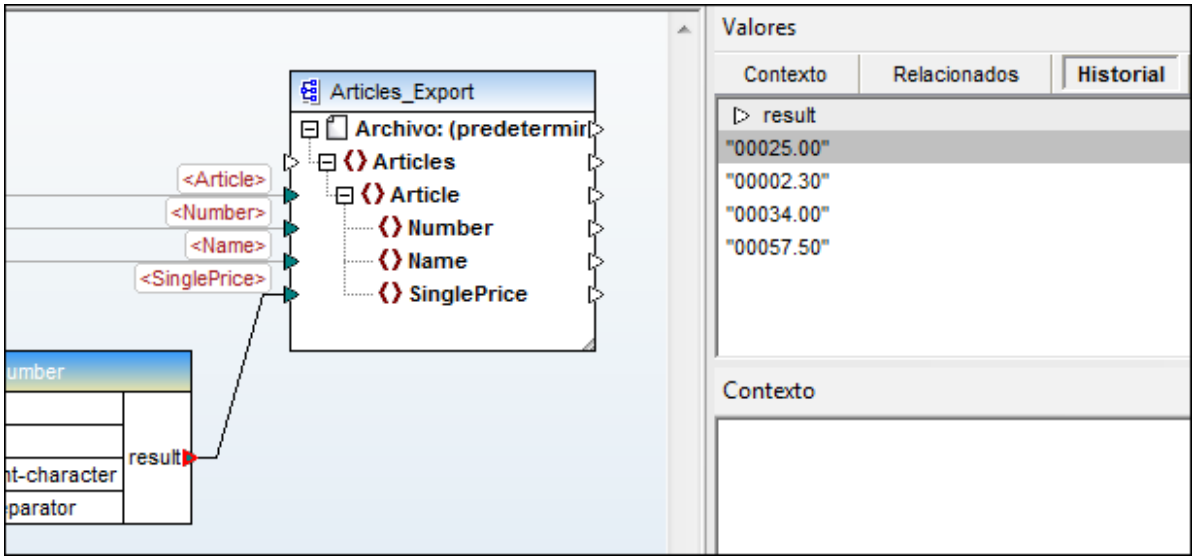
Para comprender mejor este funcionamiento del depurador vamos a depurar de principio a fin y sin puntos de interrupción una asignación y después veremos el historial de valores que fueron procesados por un conector determinado.

Para empezar abra la asignación **PreserveFormatting.mfd** del directorio **<Documentos>\Altova\MapForce2023\MapForceExamples**. Si ya tiene abierta esta asignación, asegúrese de:

- Eliminar todos los puntos de interrupción previamente definidos (*véase* *Agregar y quitar puntos de interrupción* 931).
- Detener la depuración si está en proceso (pulsando **Mayús + F5**).

Una vez detenida la depuración y eliminados todos los puntos de interrupción previamente definidos:

1. Pulse **F5** para iniciar una nueva sesión de depuración. Cuando pulse **F5**, MapForce ejecutará la asignación en modo depurador y cambiará automáticamente al panel *Resultados*.
2. Ahora haga clic en el panel *Asignación* para volver a la ventana de asignación principal.
3. Haga clic en el nodo `result` de la función `format-number` (marcado en rojo en la imagen siguiente).
4. Para terminar haga clic en la pestaña *Historial* de la ventana Valores y observe qué valores aparecen.

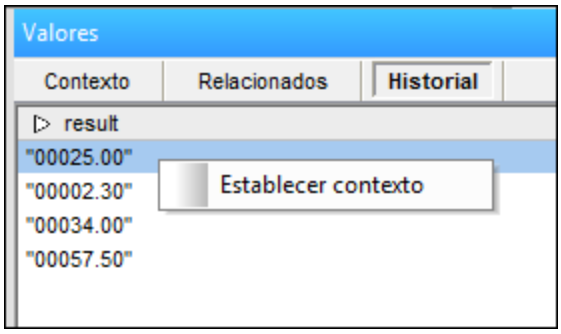


Como se aprecia en la imagen anterior, este nodo concreto (`result`) procesó un total de cuatro valores. Si necesita conocer más información sobre alguno de ellos, recuerde que puede recrear el contexto que lo produjo (*véase* *Establecer el contexto en un valor* 945).

© 2016-2022 Altova GmbH

8.11 Establecer el contexto en un valor

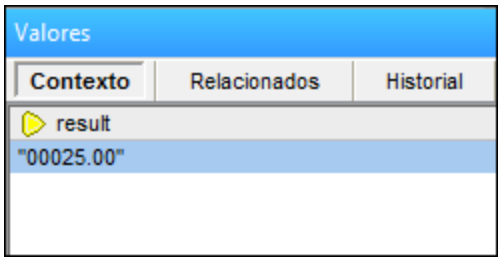
Establecer el contexto en un valor es una acción comparable con retroceder un paso hacia el pasado con el objetivo de obtener más información sobre el contexto de asignación que produjo dicho valor. Puede establecer el contexto en cualquier valor de los que aparecen en la ventana Valores (en cualquiera de sus pestañas). Si habilitó la opción *Guardar historial de seguimiento completo* (*véase* *Configurar el depurador* 1051), la pestaña *Historial* mostrará todos los valores que fueron procesados por el conector seleccionado. Por tanto, en este caso podrá establecer el contexto en cualquier valor del pasado de dicho conector.



Hay dos maneras de establecer el contexto en un valor desde la ventana Valores:

- Haciendo clic con el botón derecho en un valor y seleccionando **Establecer contexto** en el menú contextual.
- Haciendo doble clic en el valor.

Cuando se establece el contexto en un valor, MapForce resalta el área de asignación para recrear la situación que produjo dicho valor y rellena la ventana Valores y la ventana Contexto en función del contexto seleccionado. En el apartado Información sobre el modo depurador 928 se describen los iconos y señales visuales que se utilizan en el área de asignación. Para más información sobre el término *contexto* propiamente dicho consulte el apartado Ventana Contexto 936.



El conector de un contexto que se estableció a mano se señala en amarillo ( ). Esto indica que ya no estamos en la posición más reciente de la ejecución. Para volver a la posición más reciente de la ejecución (si procede), haga clic en el botón **Restaurar con actual** que aparece en la pestaña *Contexto* de la ventana Valores.

9 Automatizar tareas de MapForce con otros productos de Altova

Las asignaciones diseñadas en MapForce se pueden ejecutar en un entorno servidor (incluidos servidores Linux y macOS) y en estos motores de transformación de Altova que ofrecen un rendimiento nivel servidor (y cuyas licencias deben comprarse por separado):

- *RaptorXML Server:* la ejecución de asignaciones en este motor es ideal si el lenguaje de transformación de la asignación es XSLT 1.0, XSLT 2.0, XSLT 3.0 o XQuery (*véase Automatización con RaptorXML Server* 947).
- *MapForce Server (o MapForce Server Advanced Edition):* este motor de transformación es adecuado para las asignaciones cuyo lenguaje de transformación sea el motor integrado BUILT-IN*. El lenguaje BUILT-IN es compatible con la mayoría de las características de MapForce, mientras que MapForce Server (y en particular MapForce Server Advanced Edition) ofrece un rendimiento óptimo a la hora de ejecutar asignaciones de datos(*véase Automatización con MapForce Server* 948).

** el lenguaje de transformación BUILT-IN está disponible en las ediciones Professional y Enterprise de MapForce.*

Además, MapForce permite automatizar la generación de código XSLT, XQuery, C#, C++ y Java desde la interfaz de la línea de comandos. Esto incluye la compilación de archivos de ejecución (.mfx) para la ejecución con MapForce Server. Consulte la sección Interfaz de la línea de comandos de MapForce 963 para obtener más información.

 © 2016-2022 Altova GmbH

9.1 Automatización con RaptorXML Server

RaptorXML Server (en adelante RaptorXML) es el procesador XML y XBRL de alta velocidad y de tercera generación de Altova. Está optimizado para los estándares más recientes y para entornos de computación en paralelo. RaptorXML aprovecha la actual omnipresencia de equipos multinúcleo para ofrecer rapidísimas funciones de procesamiento de datos XML y XBRL.

Altova ofrece dos ediciones distintas de RaptorXML, que se puede descargar e instalar desde la página de descargas de Altova (https://www.altova.com/es/download-trial-server.html)::

- RaptorXML Server es un motor de procesamiento XML ultrarápido compatible con XML, XML Schema, XSLT, XPath, XQuery y muchos otros formatos.
- RaptorXML+XBRL Server ofrece todas las características de RaptorXML Server además de funciones de procesamiento y validación compatibles con toda la gama de estándares XBRL.

Si genera código en XSLT o en XQuery, MapForce crea un archivo de procesamiento por lotes llamado `.bat` que se guarda en la carpeta de salida que usted elija. Cuando se ejecuta, este archivo de procesamiento por lotes llama a RaptorXML Server y ejecuta la transformación XSLT (o XQuery) en el servidor.

Si tiene pensado ejecutar o automatizar asignaciones de datos de MapForce para otro formato de salida consulte el apartado Automatización con MapForce Server 948.

Nota: también puede obtener una vista previa del código XSLT 107 y del código XQuery con el motor integrado.

9.2 Automatización con MapForce Server

MapForce Server (https://www.altova.com/es/mapforce/mapforce-server.html) es un producto servidor para empresas compatible con sistemas Windows, Linux y macOS. MapForce Server se encarga de ejecutar asignaciones en un entorno servidor (incluso en plataformas no Windows) con un alto rendimiento. Puede ejecutar en el servidor cualquier asignación de MapForce que use el motor integrado como lenguaje de ejecución de destino (véase Seleccionar un lenguaje de transformación 24). MapForce Server puede funcionar en modo independiente (invocado desde la línea de comandos o desde la API) o como módulo de FlowForce Server.

Si decide utilizar MapForce Server en modo independiente, deberá compilar la asignación de datos de MapForce y copiarla en el equipo donde se ejecuta MapForce Server. La asignación se ejecutará con el comando `run` de la línea de comandos de MapForce Server. También puede ejecutar la asignación invocando el método `run` de la API de MapForce Server. Para más información consulte el apartado *Compilar asignaciones en archivos de ejecución de MapForce Server* 955.

Si decide ejecutar MapForce Server bajo la dirección de FlowForce Server, podrá implementar la asignación de datos en un equipo de destino a través de una conexión HTTP (o SSL/HTTPS) desde MapForce directamente. En el servidor podrá ejecutar la asignación en forma de trabajo programado o mediante una llamada a servicio web definida desde la interfaz de administración de FlowForce Server. Para más información consulte el apartado *Implementar asignaciones en FlowForce Server* 958.

Altova ofrece dos ediciones distintas de MapForce Server:

- MapForce Server
- MapForce Server Advanced Edition

La edición MapForce Server Advanced Edition ofrece las mismas características que MapForce Server pero está optimizada para determinado tipo de asignaciones, como aquellas que combinan o filtran gran cantidad de datos. El objetivo es aumentar la velocidad de ejecución.

Limitaciones:

- Incompatible con firmas XML digitales.
- Las conexiones de base de datos ADO, ADO.NET y ODBC solo son compatibles con Windows (*para información sobre otros sistemas operativos consulte Preparar asignaciones para ejecutarlas en servidores* 949).

Para más información sobre MapForce Server consulte la documentación de este producto (https://www.altova.com/es/documentation).

 © 2016-2022 Altova GmbH

9.3 Preparar asignaciones para ejecutarlas en servidores

Una asignación de datos diseñada y probada con MapForce puede hacer referencia a recursos (p. ej. bases de datos) situados fuera del equipo actual y en otro sistema operativo. Además, en MapForce, las rutas de acceso de las asignaciones de datos siguen por defecto las normas de Windows. Y, por último, el equipo donde se ejecuta MapForce Server puede ser compatible o no con las conexiones de base de datos con las que era compatible el equipo donde se diseñó la asignación de datos. Por todos estos motivos, será necesario preparar las asignaciones para ejecutaras en el entorno del servidor, sobre todo si el equipo de destino no es el mismo que el equipo de origen.

Nota: el término "equipo de origen" se usa para designar al equipo donde está instalado MapForce, mientras que "equipo de destino" se refiere al equipo donde está instalado MapForce Server o FlowForce Server. En los casos más sencillos, se trata del mismo equipo, pero en casos más complejos sería más posible ejecutar MapForce en un equipo Windows y ejecutar MapForce Server o FlowForce Server en un equipo Linux o macOS.

Por todo ello, lo más recomendable es comprobar que la asignación es válida (desde MapForce) antes de implementarla en FlowForce Server o compilarla en un archivo de ejecución de MapForce Server (*véase* *Validar asignaciones* 105).

Si MapForce Server se ejecuta de forma independiente (y no junto con FlowForce Server), estas son las licencias que necesitará:

- En el equipo de origen, necesitará la edición MapForce Enterprise o Professional para diseñar la asignación y compilarla en un archivo de ejecución de MapForce Server (.mfx), como se explica en el apartado Compilar asignaciones en archivos de ejecución de MapForce Server 955.
- En el equipo de destino, necesitará la edición MapForce Server o MapForce Server Advanced Edition para ejecutar la asignación de datos.

Si MapForce Server se gestiona desde FlowForce Server, entonces necesitará:

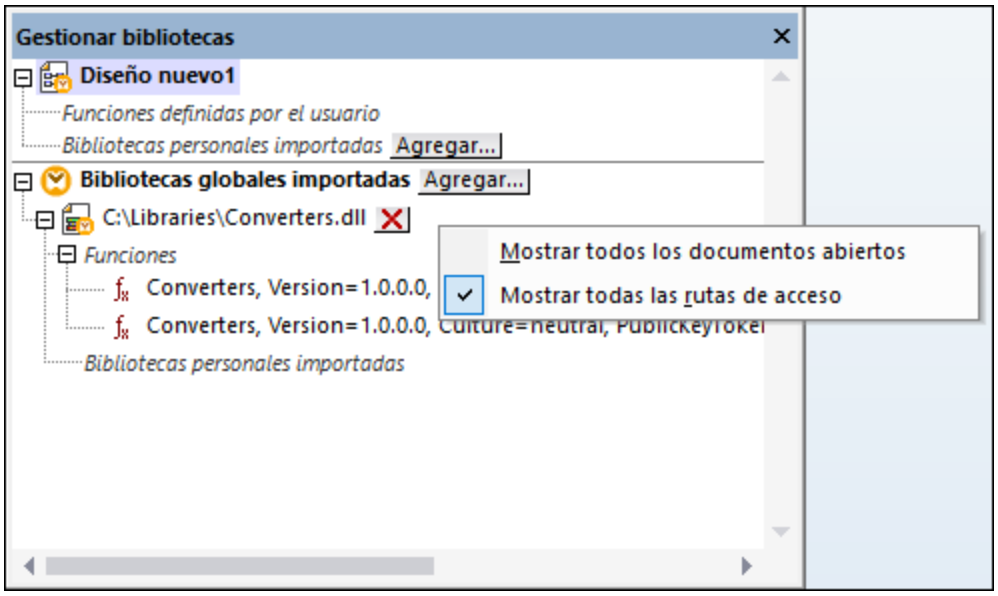
- En el equipo de origen, necesitará la edición MapForce Enterprise o Professional para diseñar la asignación e implementarla en el equipo de destino, como se explica en el apartado Implementar asignaciones en FlowForce Server 987.
- Tanto MapForce Server como FlowForce Server deben tener asignada una licencia en el equipo de destino. MapForce Server se encarga de ejecutar la asignación de datos, mientras que FlowForce Server sirve la asignación en forma de trabajo, lo cual permite acceder a numerosas características (como la ejecución bajo demanda o programada, la ejecución como servicio web, control de errores, procesamiento condicional, notificaciones por correo electrónico, etc.).
- FlowForce Server debe estar en funcionamiento en la dirección de red y puerto configurados. Es decir, el servicio "FlowForce Web Server" debe estar iniciado y configurado para aceptar conexiones de clientes HTTP (o HTTPS si corresponde) y no debe estar bloqueado por el servidor de seguridad. El servicio "FlowForce Server" también debe estar iniciado y en ejecución en la dirección y puerto designados.
- Debe tener una cuenta de usuario de FlowForce Server con permiso para acceder a uno de los contenedores (al contenedor **/public** pueden acceder todos los usuarios que inicien sesión).

Consideraciones generales

- Si tiene pensado ejecutar la asignación en un equipo de destino con una versión independiente de

MapForce Server, deberá copiar todos los archivos de entrada a los que haga referencia la asignación al equipo de destino. Si por el contrario MapForce Server se gestiona desde FlowForce Server, entonces no es necesario copiar estos archivos a mano porque los archivos de instancia y de esquema se incluyen en el paquete que se implementa en el equipo de destino, como se explica en el apartado Implementar asignaciones en FlowForce Server 987.

- Si la asignación incluye componentes de base de datos que necesitan controladores de base de datos específicos, dichos controladores deberán estar instalados en el equipo de destino también. Por ejemplo, si la asignación lee datos de una base de datos de Microsoft Access, en el equipo de destino también deberá tener instalado Microsoft Access o Microsoft Access Runtime (https://www.microsoft.com/es-es/download/details.aspx?id=50040).
- Cuando se implementa una asignación en plataformas que no son Windows, las conexiones de base de datos ADO, ADO.NET y ODBC se cambian automáticamente por JDBC. Las conexiones nativas SQLite y PostgreSQL se mantienen como tal y no requieren otra configuración. (*véase también "Conexiones de BD", más abajo*).
- Si la asignación contiene llamadas a funciones personalizadas (por ejemplo, a archivos `.dll` o `.class`), dichas dependencias no se implementan junto con la asignación porque no se conocen antes del tiempo de ejecución. En dichos casos, cópielas a mano en el equipo de destino. La ruta de acceso a los archivos `.dll` o `.class` en el servidor debe ser la misma en el cuadro de diálogo "Opciones" de MapForce; por ejemplo:



- Algunas asignaciones leen varios archivos de entrada con ayuda de una ruta de acceso comodín (véase Procesar varios archivos de entrada o salida de forma dinámica 892). En este caso el nombre de los archivos de entrada no se conocen antes del tiempo de ejecución y, por tanto, no se implementan. Para poder ejecutar la asignación correctamente los archivos de entrada deben existir en el equipo de destino.
- Si la ruta de salida de la asignación incluye directorios, estos deben existir en el equipo de destino. De lo contrario se generará un error al ejecutar la asignación. Para evitar eso, en MapForce los directorios que no existen se generan automáticamente si se habilita la opción Generar resultados en archivos temporales (*véase Cambiar opciones de MapForce* 1049).
- Si la asignación llama a un servicio web que requiere autenticación HTTPS con un certificado de cliente, el certificado debe transferirse al equipo de destino).
- Si la asignación se conecta a bases de datos basadas en archivos (como Microsoft Access y SQLite), deberá transferir a mano el archivo de base de datos al equipo de destino o guardarlo en un directorio compartido al que pueda acceder tanto el equipo de origen como el de destino (*ver más abajo*).

 © 2016-2022 Altova GmbH

Rutas portátiles

Si tiene pensado ejecutar la asignación en un servidor, asegúrese de que la asignación sigue la convención sobre rutas de acceso correspondiente y que utiliza una conexión de base de datos compatible.

1. Abra un archivo de diseño de asignación (.mfd) con MapForce en Windows.
2. En el menú **Archivo**, seleccione **Configurar asignación** y desmarque la casilla *Convertir las rutas de acceso en absolutas en el código generado* si estaba marcada.
3. En cada componente de la asignación abra el cuadro de diálogo "Propiedades" (haga doble clic en la barra de título del componente, por ejemplo) y cambie todas las rutas de los archivos de absolutas a relativas. Marque también la casilla *Guardar todas las rutas de acceso de archivos como relativas al archivo MFD*. Si le resulta más cómodo, puede copiar todos los archivos y esquemas de entrada en la misma carpeta en la que se encuentra la asignación para poder hacer referencia a ellos usando solamente su nombre.

Para más información sobre las rutas relativas y absolutas en el diseño de asignaciones (*véase* *Usar rutas de acceso relativas y absolutas* 82).

Es importante saber que tanto MapForce Server como FlowForce Server funcionan con un *directorio de trabajo* con el que se resuelven todas las rutas de acceso relativas (*véase* *Rutas de acceso en diferentes entornos de ejecución* 84). El directorio de trabajo se especifica en tiempo de ejecución:

- En FlowForce Server se especifica editando el parámetro `Directorio de trabajo` de un trabajo.
- En la API de MapForce Server API se especifica con la propiedad `WorkingDirectory` de la API de COM y .NET API o con el método `setWorkingDirectory` de la API de Java.
- En la línea de comandos de MapForce Server el directorio de trabajo es el directorio actual de la shell de comandos.

Conexiones de BD

Tenga en cuenta que los equipos Linux y macOS no son compatibles con conexiones ADO, ADO.NET y ODBC. Por tanto, si el equipo de destino es Linux o macOS, estas conexiones se convertirán en conexiones JDBC cuando la asignación se implemente en FlowForce o cuando se compile en un archivo de ejecución de MapForce Server. Cuando esto ocurra, antes de implementar o compilar la asignación podrá elegir entre dos opciones:

- crear la conexión JDBC con la base de datos (*véase* *Configurar una conexión JDBC* 183) o
- rellenar los datos de conexión de base de datos JDBC en la sección *Configuración JDBC* del componente de base de datos (*véase* *Configurar componentes de base de datos* 278).

Si la asignación utiliza una conexión nativa a una base de datos PostgreSQL o SQLite, la conexión nativa se conserva y la conversión en JDBC no tiene lugar (*véase* *Asignaciones de datos de BD en varios entornos de ejecución* 160). Si la asignación se conecta a una base de datos basada en archivos (como Microsoft Access y SQLite) deberá configurar algunas opciones más (*véase el punto siguiente*).

Para ejecutar asignaciones con conexiones JDBC es necesario tener instalado Java Runtime Environment o Java Development Kit en el equipo servidor. Este último puede ser Oracle JDK o uno de código abierto, como Oracle OpenJDK.

- La variable de entorno `JAVA_HOME` debe apuntar al directorio de instalación de JDK.

- En Windows tiene prioridad la ruta de acceso a un equipo virtual Java del registro frente a la variable `JAVA_HOME`.
- La plataforma JDK (32 o 64 bits) debe ser la misma que la de MapForce Server. De lo contrario es posible que reciba un error de tipo "No se puede acceder al equipo virtual Java".

Para configurar una conexión JDBC en Linux o macOS:

1. Descargue el controlador JDBC que ofrece el proveedor de la BD e instálelo en el sistema. Recuerde que debe seleccionar la versión de 32 bits si su sistema es de 32 bits y la versión de 54 bits si el sistema es de 64 bits.
2. Configure las variables de entorno con la ubicación donde está instalado el controlador JDBC. Por lo general será necesario configurar la variable `CLASSPATH`, pero quizá alguna más. Consulte la documentación del controlador JDBC para obtener más información.

Nota: en macOS, el sistema espera que las bibliotecas JDBC instaladas estén en el directorio `/Library/Java/Extensions`. Por tanto, se recomienda desempaquetar el controlador JDBC en esta ubicación. De lo contrario, deberá configurar el sistema para que busque la biblioteca JDBC en una ruta de acceso diferente.

Conexiones de cliente instantáneo de Oracle en macOS

Siga estas instrucciones si se va a conectar a bases de datos Oracle con el cliente **Oracle Database Instant Client** en macOs. Los requisitos son:

- Debe tener instalada la versión de Java 8.0 o posterior.Recuerde que si usa una versión de Java anterior a Java 8, también se podrá conectar con la biblioteca **JDBC Thin for All Platforms**, en cuyo caso puede ignorar las instrucciones que aparecen en este tema.
- Debe tener instalado Oracle Instant Client. El cliente Oracle Instant Client se puede descargar del sitio web oficial de Oracle. Tenga en cuenta que en la página de descargas hay varios paquetes y deberá seleccionar un paquete con compatibilidad con Oracle Call Interface (OCI), como el paquete Instant Client Basic, por ejemplo. Además, recuerde que debe seleccionar la versión de 32 bits si su sistema es de 32 bits o la versión de 64 bits si su sistema es de 64 bits

Tras descargar y desempaquetar el cliente Oracle Instant Client, edite el archivo de lista de propiedades (`.plist`) que viene con el instalador para que las variables de la tabla apunten a la ruta de acceso del controlador correspondiente. Por ejemplo:

Variable	Valor de muestra
`CLASSPATH`	`/opt/oracle/instantclient_11_2/ojdbc6.jar:/opt/oracle/instantclient_11_2/ojdbc5.jar`
`TNS_ADMIN`	`/opt/oracle/NETWORK_ADMIN`
`ORACLE_HOME`	`/opt/oracle/instantclient_11_2`
`DYLD_LIBRARY_PATH`	`/opt/oracle/instantclient_11_2`
`PATH`	`$PATH:/opt/oracle/instantclient_11_2`

© 2016-2022 Altova GmbH

Nota: los valores de las variables deben ajustarse a la ruta de acceso donde están instalados los archivos del cliente Oracle Instant Client.

Bases de datos basadas en archivos

Las bases de datos basadas en archivos (como Microsoft Access y SQLite) no se incluyen en el paquete que se implementa en FlowForce Server ni en el archivo de ejecución que se compila para MapForce Server. Por tanto, si el equipo de origen y de destino no son el mismo, deberá seguir estas instrucciones:

1. En MapForce haga clic con el botón derecho en la asignación y en el cuadro de diálogo "Configurar asignación" desactive la casilla *Convertir las rutas de acceso en absolutas en el código generado* (*véase* *Cambiar configuración de la asignación* 115).
2. Haga clic con el botón derecho en el componente de base de datos de la asignación y añada una conexión al archivo de base de datos usando una ruta de acceso relativa (*véase* *Configurar la ruta de acceso de BD basadas en archivos* 82). La manera más sencilla de evitar problemas relacionados con la ruta de acceso es guardar el diseño de asignación (archivo .mfd) en el mismo directorio que el archivo de base de datos y hacer referencia a este último desde la asignación usando el nombre de archivo (es decir, una ruta de acceso relativa).
3. Copie el archivo de base de datos en un directorio del equipo de destino (que llamaremos *directorio de trabajo*). Recuerde el nombre de este directorio porque lo necesitará para ejecutar la asignación en el servidor (ver más abajo).

Hay varias maneras de ejecutar este tipo de asignaciones en el servidor:

- Si tiene pensado ejecutar la asignación con MapForce Server bajo control de FlowForce Server, configure el trabajo de FlowForce Server para que apunte al directorio de trabajo creado previamente. El archivo de base de datos debe residir en este directorio de trabajo.
- Si tiene pensado ejecutar la asignación con MapForce Server de forma independiente y desde la línea de comandos, cambie del directorio actual al directorio de trabajo (p. ej. `cd ruta\de\directorio\de\trabajo`) antes de llamar al comando `run` de MapForce Server.
- Si tiene pensado ejecutar la asignación con la API de MapForce Server, especifique el directorio de trabajo mediante programación antes de ejecutar la asignación. Para esto sirve la propiedad `WorkingDirectory` del objeto MapForce Server en la API de COM y .NET. En la API de Java se utiliza el método `setWorkingDirectory`.

Si tanto el equipo de origen como el de destino es un equipo Windows que se ejecuta en una red local, también puede configurar la asignación para que lea el archivo de base de datos de un directorio compartido común:

1. Guarde el archivo de base de datos en un directorio compartido común al que pueda acceder tanto el equipo de origen como el de destino.
2. En la asignación haga clic con el botón derecho en el componente de base de datos y añada una conexión al archivo de base de datos usando una ruta de acceso absoluta (*véase* *Configurar la ruta de acceso de BD basadas en archivos* 82).

Recursos globales

Si la asignación incluye referencias a recursos globales en lugar de rutas de acceso directas o conexiones de base de datos, también podrá usar los recursos globales en el lado servidor. Cuando compile una asignación en un archivo de ejecución de MapForce Server (.mfx), las referencias a los recursos globales se mantienen intactas para que pueda usarlas en el lado servidor cuando ejecute la asignación. Cuando implemente una asignación en FlowForce Server puede elegir si quiere que esta use recursos en el servidor.

Para que las asignaciones (o las funciones de asignación, en el caso de FlowForce Server) se puedan ejecutar correctamente, los detalles del archivo, la carpeta o la conexión de BD que suministre como recursos globales deben ser compatibles con el entorno servidor. Por ejemplo, las rutas de acceso de los archivos y las carpetas deben usar la convención de Linux para rutas si la asignación se ejecuta en Linux. Asimismo, si define recursos globales como conexiones de BD, estas tienen que ser posibles en el equipo servidor.

Para más información consulte Recursos globales en MapForce Server 987 y Recursos globales en FlowForce Server 987..

 © 2016-2022 Altova GmbH

9.4 Compilar asignaciones en archivos de ejecución de MapForce Server

Si en la asignación de datos de MapForce eligió el motor de transformación integrado, la asignación se podrá ejecutar con MapForce y con MapForce Server (ver apartado anterior 948). Hay dos maneras de ejecutar asignaciones de datos con MapForce Server:

- Si MapForce Server se ejecuta en modo independiente (es decir, si no tiene FlowForce Server instalado), la asignación debe compilarse en un archivo de ejecución **.mfx** (ver más abajo). El archivo .mfx se puede ejecutar en la línea de comandos con el comando `run`. También puede ejecutarlo invocando el método `run` de la API de MapForce Server. Consulte la documentación de MapForce Server (https://manual.altova.com/es/MapForceServer/) para obtener más información.

- Si MapForce Server se ejecuta como módulo de FlowForce Server, la asignación se puede implementar en un equipo donde se ejecuten FlowForce Server y MapForce Server (*véase* *Implementar asignaciones en FlowForce Server* 958).

Requisitos

Consulte el apartado anterior (*Preparar asignaciones para ejecutarlas en servidores* 949).

Para compilar una asignación en un archivo de ejecución de MapForce Server (.mfx):

1. Abra una asignación de datos en MapForce (por ejemplo, **miAsignación.mfd**).
2. Haga clic en el comando de menú **Archivo | Compilar en archivo de ejecución de MapForce Server**.
3. Seleccione la carpeta donde desea guardar el archivo .mfx y cambie el nombre del archivo si lo necesita.
4. Haga clic en **Guardar**. El archivo de ejecución de MapForce Server **miAsignación.mfx** se genera en la carpeta seleccionada.

Para compilar una asignación en un archivo de ejecución de MapForce Server (.mfx) desde la línea de comandos:

- Ejecute MapForce desde la línea de comandos, especifique el archivo de asignación y la opción `/COMPILE` de la línea de comandos.

Por ejemplo, con el comando que aparece a continuación puede compilar la asignación **C:\Usuarios\altova\Documentos\Altova\MapForce2023\MapForceExamples\SimpleTotal.mfd** en un archivo de ejecución de MapForce Server que se creará en el directorio de salida **C:\Usuarios\altova\Desktop**.

```
"C:\Archivos de programa (x86)\Altova\MapForce2023\MapForce.exe" "C:
\Usuarios\altova\Documentos\Altova\MapForce2023\MapForceExamples\SimpleTotal.mfd" /COMPIL
E "C:\Usuarios\altova\Desktop"
```

Consulte la documentación de la interfaz de la línea de comandos de MapForce 963 para obtener más información.

Qué incluye el archivo .mfx

El archivo .mfx incluye esta información:

- la asignación, incluidas todas las funciones definidas por el usuario importadas de otras asignaciones
- Información[1] derivada de, según la asignación:
  - configuraciones EDI y FlexText
  - esquemas JSON
  - archivos .proto
  - taxonomías XBRL, excepto las relacionadas con la compatibilidad con tablas XBRL[2]
- Si la asignación incluye credenciales también se incluyen los campos de información sensible, cifrados, si marca la casilla *Incluir credencial cifrada en el archivo .mfx y en la implementación de la asignación*. De lo contrario esos campos se omiten. Para más detalles consulte Credenciales.

Notas:

1. Información derivada de configuraciones EDI o FlexText, esquemas JSON, taxonomías XBRL o archivos .proto se incrusta en el archivo .mfx en forma compilada y procesada. Esta información se usa al ejecutar la asignación para validar en el lado servidor. Los archivos de origen originales no se pueden obtener desde un archivo .mfx compilado. La información se compila en el estado en que esté en el momento de la compilación; si la modifica más adelante deberá volver a compilar también el archivo mfx.
2. Si la asignación hace referencia a taxonomías XBRL relacionadas con la compatibilidad con tablas XBRL, la información relativa a la taxonomía no se compila en el archivo .mfx. Sin embargo, sí que se incluye, junto con los archivos estáticos de entrada, si los hay, en el paquete que se implementa en FlowForce Server.

Los archivos de instancia de entrada (XML, CSV, texto) que use la asignación no se incluyen en el archivo .mfx compilado. Lo mismo ocurre con las bases de datos basadas en archivos, como son Access o SQLite.

Compilar asignaciones para una versión concreta de MapForce Server

Si su versión de MapForce Server es anterior a su versión de MapForce, puede que MapForce Server no pueda ejecutar correctamente los archivos creados con la versión más reciente de MapForce. Si este es su caso, puede compilar el archivo .mfx para una versión concreta de MapForce Server:

1. Haga clic en el comando de menú **Herramientas | Opciones** y después elija **Generación de código**.
2. En el grupo de opciones *Archivo de ejecución para servidor,* junto al campo *Generar para versión de MapForce Server*, seleccione la versión de MapForce Server que necesita en la lista desplegable.

 © 2016-2022 Altova GmbH

Cuando instale una versión más reciente de MapForce Server, recuerde que deberá cambiar esta opción según corresponda. También puede seleccionar la versión **más reciente** en la lista desplegable (opción predeterminada) y se compilará el archivo .mfx para la versión más reciente de MapForce Server (lo cual le permitirá beneficiarse de las mejoras y características más recientes del producto).

Para especificar una versión concreta de MapForce Server desde la línea de comandos debe ejecutar el comando `/COMPILE` con el conmutador `/MFXVERSION`.
Por ejemplo:

```
"C:\Archivos de programa (x86)
\Altova\MapForce2023\MapForce.exe" /COMPILE /MFXVERSION:2023
```

Consulte la documentación de la interfaz de la línea de comandos de MapForce 963 para obtener más información.

Otras opciones

La compilación de archivos de ejecución de MapForce Server también puede verse afectada por estas opciones de la aplicación:

Convertir todas las conexiones a BD ADO y ODBC en JDBC	Si esta opción está habilitada, las conexiones de base de datos ADO, ADO.NET y ODBC se transforman en JDBC (usando el controlador JDBC y la URL de la base de datos definida en el cuadro de diálogo "Configuración del componente". *Véase Configuración de componentes de base de datos* 278). La conexión JDBC se usará de forma implícita si el equipo de destino es un servidor Linux o macOS.
Ignorar las firmas digitales (incompatibles con MapForce Server)	Esta opción solamente es relevante en MapForce Enterprise Edition y está habilitada por defecto. Si utiliza firmas XML digitales, la asignación omitirá la información sobre firmas digitales porque MapForce Server no es compatible con esta característica

Para ver o cambiar estas opciones:

- Haga clic en el comando de menú **Herramientas | Opciones** y después elija **Generación de código**.

También puede acceder a estas opciones desde la interfaz de la línea de comandos 963.

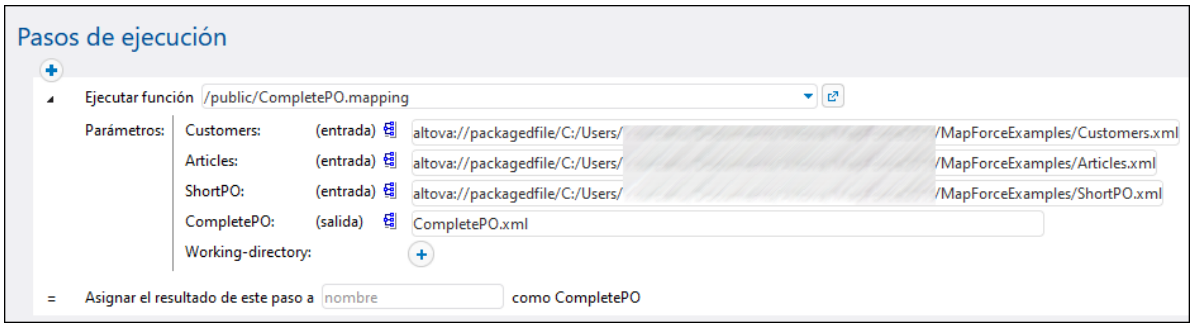
9.5 Implementar asignaciones en FlowForce Server

Al implementar una asignación en FlowForce Server MapForce se encarga de organizar los recursos usados por una asignación en concreto en un objeto que pasa mediante HTTP (o HTTP, según la configuración) al equipo en el que se está ejecutando FlowForce Server. Las asignaciones de MapForce se suelen implementar en FlowForce Server para automatizar su ejecución mediante trabajos de FlowForce. Una vez se haya implementado una asignación puede usarla para crear un trabajo completo de FlowForce y aprovechar todas las funcionalidades relacionadas (por ejemplo, definir condiciones desencadenantes para el trabajo, exponerlo como servicio web, etc.).

Nota: el término "equipo de origen" hace referencia al equipo en el que está instalado MapForce y "equipo de destino" se refiere al equipo en el que está instalado FlowForce Server. En un caso muy simple se trataría del mismo equipo. En uno más complejo, MapForce se ejecuta en un equipo con Windows y FlowForce Server en uno con Linux o macOS.

El paquete que se implementa en FlowForce incluye:

- La asignación. Una vez la haya implementado la asignación aparece en la interfaz administrativa de FlowForce como una función de asignación (.mapping) en la ruta que indique. Los componentes de origen se convierten en argumentos de entrada de la función y los componentes de destino en los argumentos de salida.



- Todos los archivos de instancia de entrada (XML, CSV, Text) que se usan en la asignación.

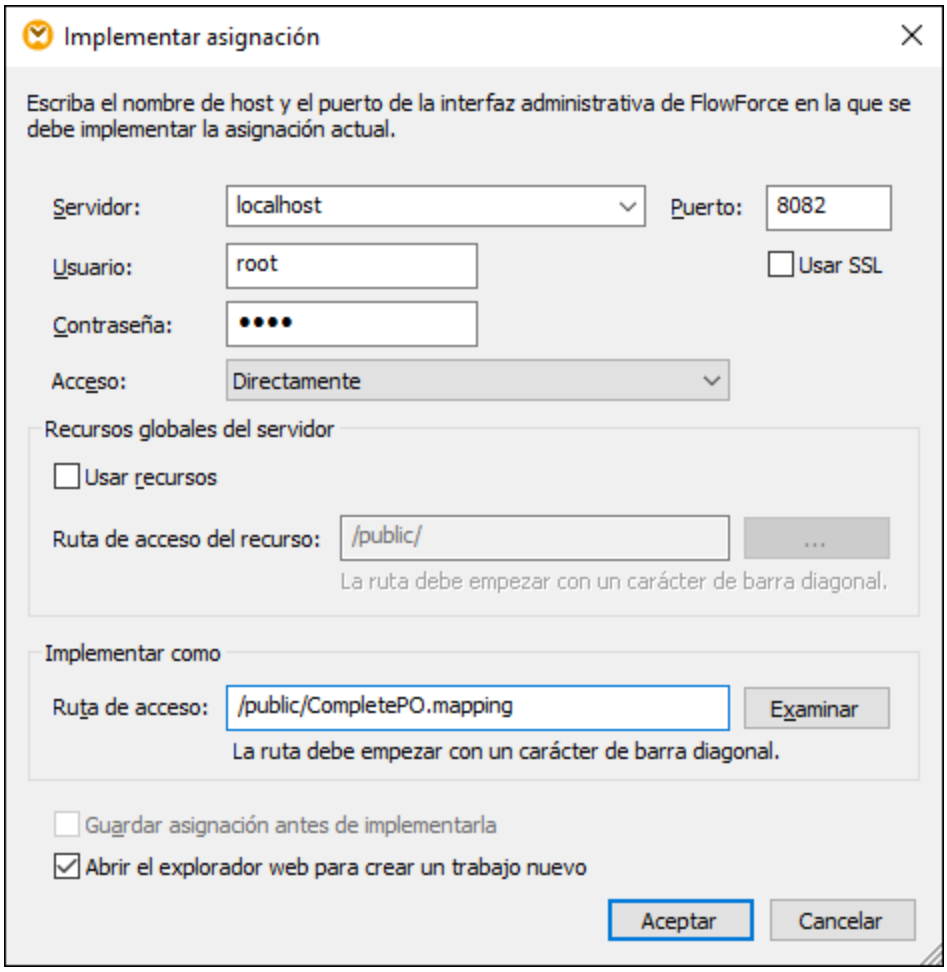
Requisitos

Consulte Preparar asignaciones para ejecutarlas en servidores 949.

Implementar la asignación en FlowForce Server

1. Ejecute MapForce y asegúrese de que el lenguaje de transformación es BUILT-IN (haga clic en el botón **Built-in** de la barra de herramientas o seleccione el comando de menú **Resultados | Motor de ejecución integrado**).
2. En el menú **Archivo** haga clic en **Implementar en FlowForce Server** para abrir el cuadro de diálogo "Implementar asignación".

 © 2016-2022 Altova GmbH



3. Introduzca los detalles de la implementación (como se describe más abajo) y haga clic en Aceptar. Si marcó la casilla **Abrir el explorador web para crear un trabajo nuevo** se abre la interfaz administrativa de FlowForce Server en el explorador y puede empezar a crear un trabajo de FlowForce de inmediato.

En esta tabla explicamos todas las opciones que puede configurar para las asignaciones en el cuadro de diálogo "Implementar asignación".

Opción	Descripción
Servidor, Puerto, Usar SSL	Introduzca el nombre de host del servidor (o la dirección IP) y el puerto de FlowForce Server. Puede usar **localhost** y **8082** si FlowForce Server se está ejecutando en el mismo equipo en el puerto predeterminado. Si no está seguro de ello, acceda a la interfaz administrativa de FlowForce Web y compruebe la dirección IP y el puerto que aparecen en la barra de direcciones del explorador web. Si tiene errores de conectividad compruebe si el equipo en el que se está ejecutando FlowForce Server está configurado para permitir conexiones entrantes en la dirección y el puerto indicados.

Opción	Descripción
	Para implementar la asignación con una conexión con cifrado SSL marque la casilla *Usar SSL*. Si esa opción está activa se asume que FlowForce Server ya se ha configurado para aceptar conexiones SSL. Para más información consulte la documentación de FlowForce Server (https://www.altova.com/es/documentation).
Usuario y contraseña	El nombre de usuario y la contraseña que debe introducir dependen del valor que se haya escogido en la lista desplegable de acceso (ver la opción siguiente). Si el valor es **<Predeterminado>** o **Directamente** puede introducir el nombre de usuario y la contraseña que usa para FlowForce Server. De lo contrario debe usar el nombre de usuario y la contraseña de su dominio y seleccionar el nombre de ese dominio en la lista desplegable de acceso.
Acceso	Si la integración de servicios de directorio está habilitada en FlowForce Server, seleccione el nombre de su dominio en esta lista desplegable e introduzca las credenciales correspondientes en los campos Nombre y Contraseña (véase la opción anterior).
Usar recursos, Ruta de acceso del recurso	Marque la casilla *Usar recursos* si quiere que la función de asignación use Recursos 969 una vez se haya implementado en el servidor. Si marca esta casilla también debe introducir la ruta de acceso en el servidor al recurso en cuestión en el campo de texto **Ruta de acceso del recurso**. Para seleccionar el recurso, haga clic en el botón con puntos suspensivos. Si en el servidor no hay recursos de entre los que escoger haga clic en **Implementar recursos globales** e implemente en el servidor el recurso global que necesite. Para más información consulte Implementar recursos en FlowForce Server 987 . Si no marca la casilla *Usar recursos* se resolverá cualquier recurso global en base a la configuración seleccionada actualmente. En el servidor, la función de asignación ya no buscará un recurso global, sino que usará el valor resuelto en su lugar.
Ruta de acceso	Haga clic en **Examinar** y seleccione la ruta de acceso donde debe guardarse la función de asignación en la jerarquía de contenedores de FlowForce Server. Por defecto, la ruta apunta al contenedor **/public** de FlowForce Server. Desde este cuadro de diálogo también puede crear contenedores nuevos o eliminar contenedores y asignaciones que ya existan, siempre que tenga los permisos y privilegios necesarios en FlowForce Server.
Guardar cambios en el diseño antes de implementarlo	Esta opción se habilita si implementa una asignación sin guardar. Marque esta casilla para guardar la asignación antes de implementarla.
Abrir el explorador web para crear un trabajo nuevo	Si marca esta casilla, una vez se haya implementado la asignación se abre la interfaz administrativa de FlowForce Web en el explorador para que pueda empezar a crear un trabajo de inmediato.

© 2016-2022 Altova GmbH

Resolución de problemas

En esta tabla explicamos los problemas que puede encontrarse al implementar una asignación y la solución correspondiente.

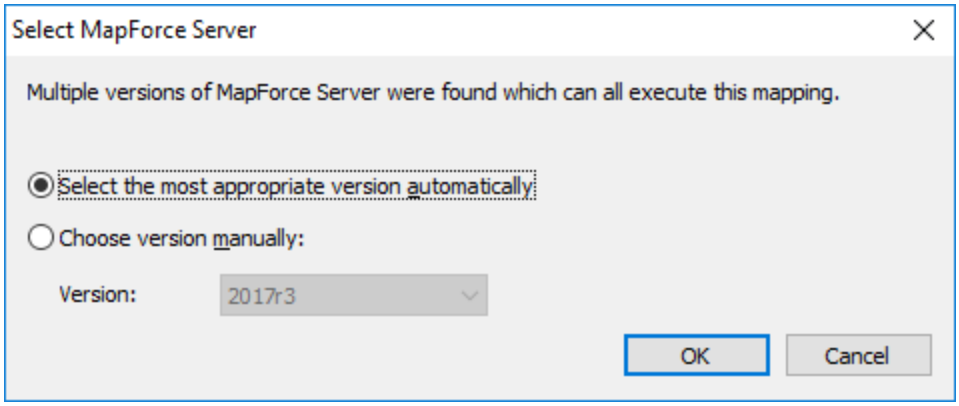
Problema	Solución

Al implementar la asignación se genera este error:
 	
`No se puede realizar la operación de entrada/salida en el archivo ...`
`Error de entrada/salida 28: No se puede establecer la conexión con el puerto de <server> 8082. Tiempo de espera agotado`
`Error del sistema 10060: error al intentar conectarse porque se superó el tiempo de espera o error en la conexión establecida porque el host no responde.`
	Asegúrese de que en el equipo de destino se está ejecutando el servidor web de FlowForce y que está configurado para escuchar conexiones en el puerto indicado (**8082** por defecto). También debe cerciorarse de que el firewall no está bloqueando las conexiones que entran a este puerto.
	
	El servicio *FlowForce Server* también debe estar en ejecución para que se pueda realizar la implementación.

Al implementar la asignación se genera este error:
 	
`No se puede realizar la operación de entrada/salida en el archivo...`
`Error de entrada/salida 413: La carga es demasiado grande`
	Este error puede aparecer si un archivo de entrada de la asignación implementada sobrepasa el tamaño límite máximo de solicitudes HTTP que FlowForce permite (unos 100 MB). Puede aumentar ese límite con la opción `x_request_body_size` (en bytes) en los archivos **flowforceweb.ini** y **flowforce.ini**. Para más detalles consulte la documentación de FlowForce Server.

Seleccionar la versión del servidor (solo Windows)

Si en el servidor en el que implementa la asignación hay varias versiones de MapForce Server ejecutándose bajo el control de FlowForce Server (solo en servidores Windows), entonces la aplicación le pedirá que indique también la versión de MapForce Server en la que quiere que se ejecute la asignación.



Nota: el cuadro de diálogo aparece si el directorio de instalación de FlowForce Server contiene archivos .tool para cada una de las versiones de MapForce Server que se está ejecutando bajo el control de FlowForce Server. Por defecto, al instalar MapForce Server como parte de la instalación de FlowForce Server, MapForce Server añade un archivo .tool a ese directorio automáticamente. La ruta donde se guardan los archivos .tool en FlowForce es: **C:\Program Files\Altova\FlowForceServer2023\tools**.

Si cuenta con otras versiones de MapForce Server que quiera ejecutar bajo el control de FlowForce Server es posible que deba copiar los archivos .tool correspondientes manualmente al directorio que acabamos de mencionar. Puede encontrar el archivo .tool de MapForce Server en: **C:\Program Files\Altova\MapForceServer2023\etc**.

© 2016-2022 Altova GmbH

9.6 Interfaz de la línea de comandos de MapForce

La sintaxis general para usar un comando de MapForce en la línea de comandos es:

```
MapForce.exe <filename> [/{target} [[<outputdir>] [/options]]]
```

Leyenda

Para indicar la sintaxis de la línea de comando se usa la siguiente notación:

Notación	Descripción
Texto sin corchetes ni llaves	Elementos que puede teclear tal y como aparecen
<Texto entre cuñas>	Marcador de posición para el que debe indicar un valor
[Texto entre corchetes]	Elementos opcionales
{Texto entre llaves}	Conjunto de elementos obligatorios; elija uno
Barra vertical (|)	Separador para elementos mutuamente excluyentes; elija uno
Puntos suspensivos (...)	Elementos que pueden repetirse

<filename>

El archivo del diseño de la asignación (.mfd) o del proyecto de la asignación (.mfp) a partir del cual se genera el código. Para generar código para todo el proyecto, indique el destino `/GENERATE` (*véase la siguiente tabla*) e introduzca la ruta del proyecto en el campo `<filename>`, por ejemplo **MapForceExamples.mfp**.

/{target}

Indica el lenguaje o entorno de destino para el que se genera el código. Son compatibles los siguientes destinos de generación de código.

Destino	Descripción
`/COMPILE[:compileoptions]`	Compila una asignación en un archivo de ejecución de MapForce Server (.mfx). También se pueden indicar las siguientes opciones: `JDBC` — Transforma todas las conexiones DB en JDBC usando el driver JDBC y la URL definidos en el cuadro de diálogo Configuración delde componentes de BD (*véase también Configuración de componentes de BD* 278) 278 . `NOXMLSIGNATURES` — Elimina la generación de firmas digitales en el archivo de de MapForce Server (observe que las firmas digitales no son compatibles con MapForce Server).

Destino	Descripción
/GENERATE	Genera código de proyecto para todas las asignaciones del archivo del proyecto usando la configuración actual de la carpeta (véase Gestionar las carpetas de proyecto 121). Si selecciona este destino, asegúrese de que indica un proyecto de MapForce (archivo .mfp) en el campo <filename>.
/XSLT	Genera código XSLT 1.0.
/XSLT2	Genera código XSLT 2.0.
/XSLT3	Genera código XSLT 3.0.
/XQuery	Genera código XQuery.
/JAVA	Genera código Java.
/CS	Genera código C#. Este comando también permite usar opciones específicas para generar el código, a saber: /CS[:{VS2010|VS2013|VS2015|VS2017|VS2019}] VS2010 Visual Studio 2010 VS2013 Visual Studio 2013 VS2015 Visual Studio 2015 VS2017 Visual Studio 2017 VS2018 Visual Studio 2018 Si no se especifica ninguna versión de Visual Studio, el código se generará usando la versión del programa que esté definida en las opciones del generador de código (*véase Opciones del generador de código* 1207).
/CPP	Genera código C++. Este comando también permite Este comando también permite usar opciones específicas para generar el código, a saber: /CPP[:{VS2010|VS2013|VS2015|VS2017|VS2019},{MSXML|XERCES3},{LIB|DLL},{MFC|NoMFC}] El primer grupo de opciones establece la versión de Visual Studio de destino. Valores válidos: VS2010 Visual Studio 2010 VS2013 Visual Studio 2013

© 2016-2022 Altova GmbH

Destino	Descripción
	`VS2015` Visual Studio 2015 `VS2017` Visual Studio 2017 `VS2019` Visual Studio 2019 El segundo grupo de opciones indica la biblioteca XML de destino para el código generado. Valores válidos: `MSXML` Genera código para MSXML 6.0 `XERCES3` Genera código para Xerces 3 El tercer grupo de opciones indica si deben generarse bibliotecas estáticas en vez de dinámicas. Valores válidos: `LIB` Genera bibliotecas LIB estáticas `DLL` Genera bibliotecas DLL El cuarto grupo de opciones indica si el código generado debe ser compatible o no con MFC. Valores válidos: `MFC` Habilita la compatibilidad con MFC `NoMFC` Deshabilita la compatibilidad con MFC Si no se especifican estas opciones, el código se generará usando la versión del programa que esté definida en las opciones del generador de código (*véase* *Opciones del generador de código* 1207).

<outputdir>

Parámetro opcional que indica el directorio de salida. Si no se indica una ruta de salida, se usará el directorio de trabajo actual. Observe que las rutas relativas lo son al directorio de trabajo actual.

Cuando el destino es `/GENERATE` y el parámetro `<outputdir>` no está definido, el lenguaje de generación de código, así como la ruta de salida de cada asignación, los determinan las opciones definidas para cada carpeta del proyecto (*véase* *Gestionar las carpetas de proyecto* 121).

Cuando el destino es `/GENERATE` y el parámetro `<outputdir>` está definido, el valor `<outputdir>` introducido en la línea de comandos tiene prioridad frente al directorio de salida definido a nivel del proyecto raíz, pero no la tiene frente a las opciones de generación de código definidas en cada una de las carpetas dentro del proyecto.

/options

Las opciones `/options` no se excluyen mutuamente. Se pueden indicar una o más de las siguientes opciones.

Opción	Descripción
`[/MFXVERSION[:<version>]`	Esta opción se puede aplicar si el destino es `/COMPILE`. Con ella se compila un archivo de ejecución de MapForce Server (.mfx) para una versión específica de MapForce Server. Como valor puede introducir cualquiera de las versiones de MapForce Server a partir de la versión **2013r2** hasta la más reciente. Consulte también Compilar asignaciones en archivos de ejecución de MapForce Server 956.
`/GLOBALRESOURCEFILE <filename>`	Esta opción se puede aplicar si la asignación de datos usa los recursos globales para resolver los archivos o las rutas a directorios de entrada o de salida, o bases de datos. Para más información, consulte Recursos globales de Altova 969. La opción `/GLOBALRESOURCEFILE` indica la ruta al archivo .xml de los recursos globales. Tenga en cuenta que si está indicado `/GLOBALRESOURCEFILE` también ha de estarlo `/GLOBALRESOURCECONFIG`.
`/GLOBALRESOURCECONFIG <config>`	Esta opción especifica el nombre de la configuración de los recursos globales (*véase la opción anterior*). Observe si está indicado `/GLOBALRESOURCEFILE` también ha de estarlo `/GLOBALRESOURCECONFIG`.
`/LIBRARY <libname> (...)`	Esta opción se usa junto con un lenguaje de generación de código de destino para especificar bibliotecas de funciones adicionales. Se puede usar varias veces para cargar varias bibliotecas. Estas bibliotecas se añaden de forma temporal (para esta ejecución) a las que se encuentran en **Herramientas | Opciones | Bibliotecas**.
`/LOG <logfilename>`	Genera un archivo de registro en la ruta indicada. `<logfilename>` puede ser una ruta completa, es decir, puede incluir tanto un directorio como un nombre de archivo. Sin embargo, si indica una ruta completa, el directorio debe existir para que se genere el archivo de registro. Si solo indica el nombre del archivo, entonces este se guardará en el directorio actual del símbolo del sistema de Windows.

Observaciones

- Las rutas relativas lo son al directorio de trabajo, que es el directorio de la aplicación que llama a MapForce en ese momento. Esto afecta a la ruta del nombre de archivo .mfd, del nombre de archivo .mfp, del directorio de salida, del nombre del archivo de registro y del nombre de archivo de los recursos globales. .
- No cierre la ruta con barra inversa y comillas en la línea de comandos (por ejemplo: "C:\Mi Directorio\"). El analizador sintáctico interpreta estos dos caracteres como comillas dobles literales. Use la doble barra inversa \\ si hay espacios en la línea de comando y necesita las comillas ("C:\Mi Directorio\\") o intente evitar el uso de espacios y, por ende, de comillas.

Ejemplos

1) Para iniciar MapForce y abrir la asignación `<filename>.mfd`, use:

 © 2016-2022 Altova GmbH

```
MapForce.exe <filename>.mfd
```

2) Para generar código XSLT 2.0 y crear también un archivo de registro con el nombre `<logfilename>`, use:

```
MapForce.exe <filename>.mfd /XSLT2 <outputdir> /LOG <logfilename>
```

3) Para generar código XSLT 2.0 teniendo en cuenta la configuración del recurso global `<grconfigname>` del archivo de recurso global `<grfilename>`, use:

```
Mapforce.exe <filename>.mfd /XSLT2 <outputdir> /GLOBALRESOURCEFILE
<grfilename> /GLOBALRESOURCECONFIG <grconfigname>
```

4) Para generar una aplicación en código C# para Visual Studio 2015 y generar un archivo de registro, use:

```
MapForce.exe <filename>.mfd /CS:VS2015 <outputdir> /LOG <logfilename>
```

5) Para generar una aplicación en C++ usando las opciones de generación de código definidas en **Herramientas | Opciones** y generar un archivo de registro, use:

```
MapForce.exe <filename>.mfd /CPP <outputdir> /LOG <logfilename>
```

6) Para generar una aplicación en C++ para Visual Studio 2015, MSXML, con bibliotecas estáticas, compatible con MFC y sin archivo de registro, use:

```
MapForce.exe <filename>.mfd /CPP:VS2015,MSXML,LIB,MFC
```

7) Para generar una aplicación en C++ para Visual Studio, Xerces, con bibliotecas dinámicas, sin compatibilidad con MFC y con archivo de registro, use:

```
MapForce.exe <filename>.mfd /CPP:VS2015,XERCES,DLL,NoMFC <outputdir> /LOG <logfilename>
```

8) Para generar una aplicación en Java y también un archivo de registro, use:

```
MapForce.exe <filename>.mfd /JAVA <outputdir> /LOG <logfilename>
```

9) Para generar código para todas las asignaciones del proyecto usando el lenguaje y el directorio de salida definidos en las opciones de la carpeta (de cada carpeta dentro del proyecto), use:

```
MapForce.exe <filename>.mfp /GENERATE /LOG <logfilename>
```

10) Para generar código Java para todas las asignaciones del archivo del proyecto, use:

```
MapForce.exe <filename>.mfp /JAVA /LOG <logfilename>
```

Con esta opción se ignora el lenguaje de generación de código definido en las opciones de la carpeta y se usa Java para todas las asignaciones.

11) Para indicar los archivos de entrada y de salida en la línea de comandos para una asignación en Java compilada con anterioridad, use:

```
java -jar <mappingfile>.jar /InputFileName <inputfilename> /OutputFileName
<outputfilename>
```

Los parámetros `/InputFileName` y `/OutputFileName` son nombres de componentes especiales de entrada de la asignación de datos de MapForce que permiten usar parámetros en la ejecución de la línea de comandos (*véase* *Pasar parámetros a la asignación* 477).

12) Para compilar una asignación en un archivo de ejecución de MapForce Server para la versión de MapForce 2023 y eliminar las firmas XML:

```
MapForce.exe <filename>.mfd /COMPILE:NOXMLSIGNATURES <outputdir> /MFXVERSION:2023 /LOG
<logfilename>
```

 © 2016-2022 Altova GmbH

10 Recursos globales de Altova

Los recursos globales de Altova son alias para recursos de archivo, carpeta y base de datos. Cada alias puede tener varias configuraciones y cada configuración se corresponde con un solo recurso. Esto quiere decir que puede alternar entre configuraciones al usar recursos globales. Por ejemplo, podría crear un recurso de base de datos con dos configuraciones: `desarrollo` y `producción`. Puede cambiar de una configuración a otra en función de sus necesidades. Por ejemplo, en MapForce podría recuperar la información tanto desde la base de datos de `desarrollo` como desde la de `producción`; para ello basta con elegir en una lista desplegable qué configuración quiere usar antes de previsualizar la asignación.

Los recursos globales se pueden usar en distintas aplicaciones de Altova (*véase la lista más abajo*).

Recursos globales en otros productos de Altova

Una vez los haya guardado como recursos globales, puede usar esos archivos, carpetas o conexiones de base de datos en varias de las aplicaciones de Altova. Por ejemplo, si suele tener que abrir el mismo archivo (o carpeta o conexión de base de datos) en distintas aplicaciones de escritorio de Altova puede que le resulte más cómodo definirlo como un recurso global. Así, si quiere cambiar la ruta de acceso solo tiene que hacerlo en un sitio. Por el momento, puede definir y usar recursos globales en estos productos de Altova:

- Altova Authentic
- DatabaseSpy
- MobileTogether Designer
- MapForce
- StyleVision
- XMLSpy
- FlowForce Server
- MapForce Server
- RaptorXML Server/RaptorXML+XBRL Server

Apartados de esta sección

En esta sección explicamos cómo crear y configurar distintos tipos de recursos globales. Estos son los apartados de esta sección:

- Configurar recursos globales, parte 1 970
- Configurar recursos globales, parte 2 972
- Archivos XML y recursos globales 976
- Carpetas como recursos globales 978
- Bases de datos como recursos globales 980
- Resultados de transformación como recursos globales 982
- Recursos globales en entornos de ejecución 986

10.1 Crear recursos globales

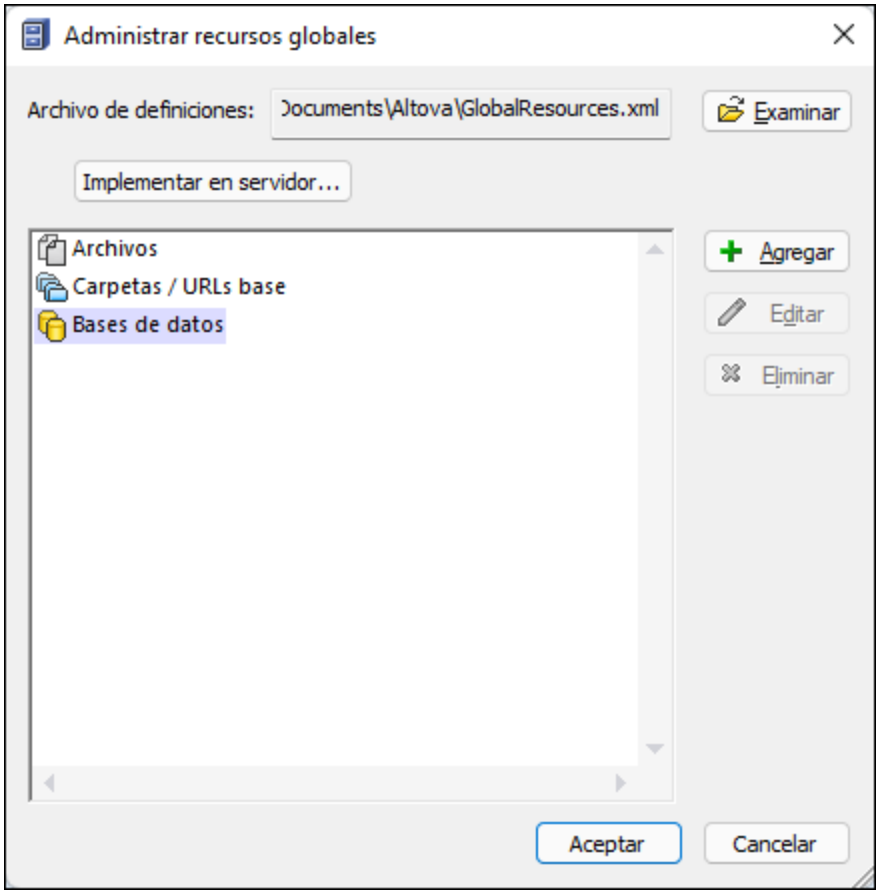
La configuración de los recursos globales tiene dos pasos: (I) crear el recurso global en el cuadro de diálogo **Administrar recursos globales** (*imagen siguiente*) y (ii) definir las propiedades de este recurso global en el cuadro de diálogo **Recursos globales**. La segunda parte se explica en el apartado siguiente 972.

Los recursos globales de Altova se definen en el cuadro de diálogo **Administrar recursos globales**, al que se puede acceder de dos formas:

- Haga clic en el comando de menú **Herramientas | Recursos globales**.
- Haga clic en el icono **Administrar recursos globales** de la barra de herramientas Recursos globales (*imagen siguiente*).

El archivo de definiciones de recursos globales

La información sobre los recursos globales se almacena en un archivo XML llamado archivo de definiciones de recursos globales. Este archivo se crea y guarda cuando se define el primer recurso global en el cuadro de diálogo **Administrar recursos globales** (*imagen siguiente*).



 © 2016-2022 Altova GmbH

Cuando se abre el cuadro de diálogo **Administrar recursos globales** por primera vez, el nombre y la ubicación predeterminados del archivo de definiciones de recursos globales aparece en el cuadro de texto *Archivo de definiciones* (*imagen anterior*):

```
C:\Usuarios\<usuario>\Documentos\Altova\GlobalResources.xml.
```

Este archivo se define como archivo predeterminado de definiciones de recursos globales para todas las aplicaciones de Altova. Así puede guardar en este archivo recursos globales desde cualquier aplicación de Altova y el recurso global estará a disposición de todas las aplicaciones de Altova. Para definir y guardar un recurso global en el archivo de definiciones, añada el recurso global en el cuadro de diálogo **Administrar recursos globales** y haga clic en **Aceptar**.

Para seleccionar un archivo de definiciones de recursos globales ya existente como archivo de definiciones activo, búsquelo con el botón **Examinar** del cuadro de texto *Definiciones* (*imagen anterior*).

En el cuadro de diálogo **Administrar recursos globales** también permite editar y eliminar recursos globales.

Notas:

- Puede asignar cualquier nombre al archivo de definiciones de recursos globales y guardarlo en cualquier ubicación a la que tengan acceso sus aplicaciones de Altova. Después en las aplicaciones de Altova sólo tiene que definir este archivo como archivo de definiciones de recursos globales (en el cuadro de texto *Definiciones*). Ahora el recurso global se puede usar en distintos productos de Altova con una sola definición para todos ellos.

- Si lo prefiere, también puede crear varios archivos de definiciones de recursos globales. Sin embargo, en cada aplicación de Altova sólo puede estar activo un archivo de definiciones y, por tanto, sólo las definiciones de ese archivo estarán a disposición de la aplicación. También puede restringir la disponibilidad de los recursos o ampliarla a varios productos.

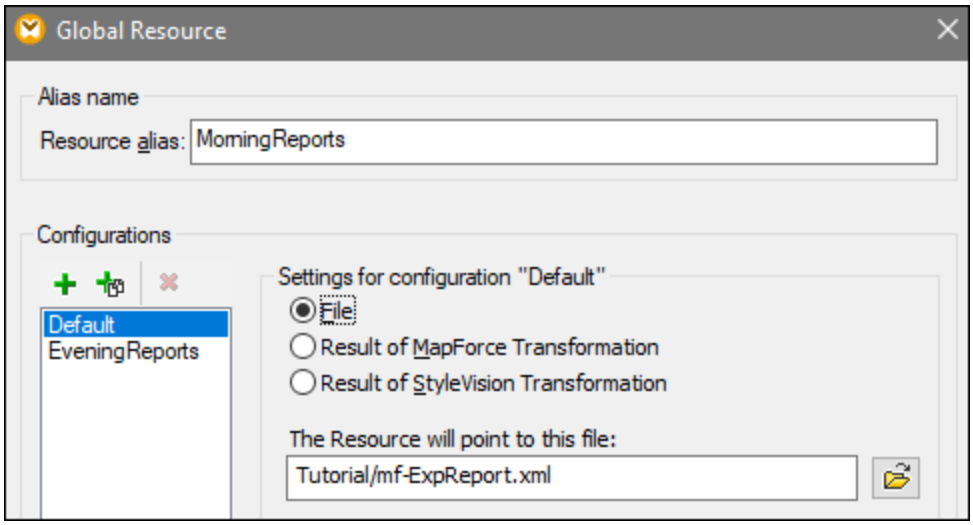
10.2 Global Resource Setup Part 2

The second part of the global resource setup consists in defining properties of a global resource in the **Global Resource** dialog box. The properties depend on the type of a global resource (*see subsections below*). You can access the **Global Resource** dialog box by clicking the **Add** button in the Manage Global Resources dialog box 970.

To find out more about setting up different types of global resources, see the following examples: XML Files as Global Resources 976, Folders as Global Resources 978, Databases as Global Resources 980.

Files

The file-specific properties are shown in the **Global Resource** dialog box below. The setup has three major parts: (i) the name of the file, (ii) the location of this file, and (iii) the list of configurations defined for this file alias.

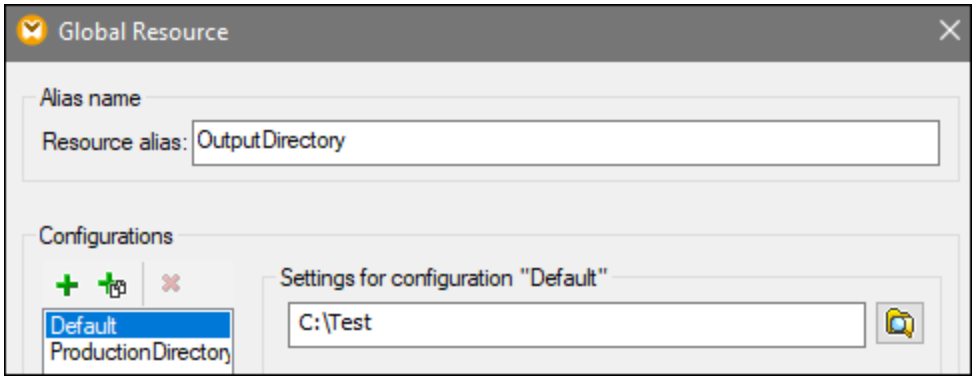


The settings *Result of MapForce Transformation* and *Result of StyleVision Transformation* are discussed in Transformation Results as Global Resources 982.

Folders

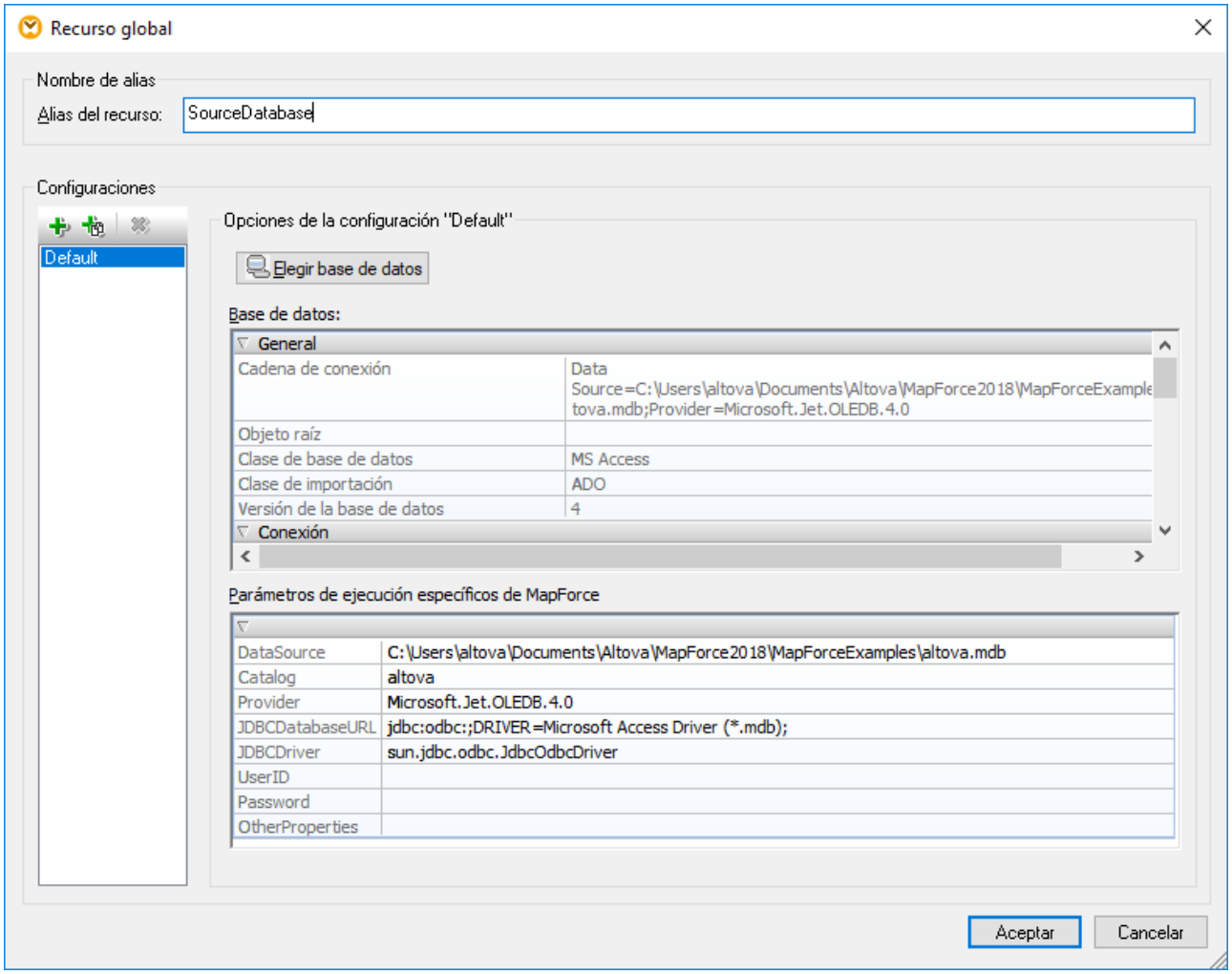
The folder-specific properties are shown in the **Global Resource** dialog box below. The setup has three major parts: (i) the name of the folder, (ii) the location of this folder, and (iii) the list of configurations defined for this folder alias.

© 2016-2022 Altova GmbH



Databases

When you add a database connection as a global resource, a connection wizard guides you through the steps required to set up the connection. For more information, see Start Database Connection Wizard 163. Once you complete the wizard, the database connection parameters are displayed in the **Global Resource** dialog box (*see screenshot below*).



In the **Global Resource** dialog box, it is possible to edit some of the database connection parameters. The parameters are grouped into two categories: database parameters and MapForce-specific execution parameters (*see below*).

Database

These parameters are shared among Altova applications. In MapForce, they are used at design time, that is, when the mapping is loaded, or when you click the **Output** pane in MapForce to preview the mapping.

MapForce-specific execution parameters

These parameters are applicable when you generate program code or compile a mapping to MapForce Server execution file (`.mfx`). They are used at mapping runtime as follows:

- In generated C++, C#, or Java program code.
- If you compile the mapping to a MapForce Server execution file, and automatic JDBC conversion takes place. For more information about automatic JDBC conversion, see Database mappings in various execution environments 160.

If a mapping uses a Global Resource to connect to a database, the database connection details in the **Global Resource** dialog box take precedence over those defined in the mapping. The Component Settings dialog box informs you that the connectivity parameters are defined as a Global Resource. To change the database

component to connect to the database directly (without using Global Resources), click **Change**, and follow the wizard steps to reconnect to the database.

Global Resource dialog icons

	Add Configuration: Pops up the Add Configuration dialog in which you enter the name of the configuration to be added.
	Add Configuration as Copy: Pops up the Add Configuration dialog in which you can enter the name of the configuration to be created as a copy of the selected configuration.
	Delete: Deletes the selected configuration.
	Open: Browse for the file to be created as the global resource.
	Open: Browse for the folder to be created as the global resource.

Global resource setup: General procedures

The broad procedure of creating and configuring global resources is described below:

1. Click the toolbar button (**Manage Global Resource**). Alternatively, go to the **Tools** menu and click **Global Resources**.
2. Click **Add** and select the resource type you wish to create (file, folder, database). The **Global Resource** dialog box will appear.
3. Enter a descriptive name in the **Resource alias** text box (e.g., `InputFile`).
4. Setting up the default configuration depends on the type of the global resource: (i) For a file or folder, browse for the file or folder to which this resource should point by default; (ii) for a database connection, click **Choose Database** and follow the Database Connection Wizard to connect to the database (see Connecting to a Database 161). This database connection will be used by default when you run the mapping.
5. If you need an additional configuration (e.g., an additional output folder), click the button in the **Global Resource** dialog box, enter the name of this configuration, and specify the path to this configuration.
6. Repeat the previous step for each additional configuration required.

Note: Database connections are supported as global resources only in MapForce Professional and Enterprise editions.

10.3 Ejemplo: ejecutar asignación con archivos de entrada variables

En este apartado explicamos cómo usar archivos XML como recursos globales. Puede haber situaciones en las que quiera cambiar un archivo XML de entrada varias veces al día. Por ejemplo, imaginemos que todas las mañanas ejecuta una asignación en concreto y genera un informe usando como entrada un archivo XML, y todas las tardes ejecuta la misma asignación para generar otro reporte, pero este usa un archivo XML distinto como entrada. Aquí es donde pueden ser útiles los recursos globales: en lugar de editar la asignación varias veces al día (o tener varias copias de la misma), podría configurar la asignación para que que lea un archivo definido como recurso global (lo que se conoce como *alias de archivo*). En este ejemplo, nuestro alias de archivo tiene dos configuraciones:

1. `Default`, que daría un archivo XML "de mañana" como entrada para la asignación.
2. `EveningReports`, que daría un archivo XML "de tarde" como entrada para la asignación.

Para crear y configurar el alias de archivo siga estos pasos que explicamos a continuación.

Paso 1: crear el recurso global

Primero debe crear un alias de archivo. Para ello siga estos pasos:

1. Haga clic en el botón de la barra de herramientas (**Administrar recursos globales**). Si lo prefiere, vaya al menú **Herramientas** y haga clic en **Recursos globales**.
2. Haga clic en **Agregar | Archivo** e introduzca un nombre en el cuadro de texto **Alias del recurso**. En este ejemplo podríamos usar `MorningReports`.
3. Haga clic en **Examinar** y seleccione **`Tutorial\mf-ExpReport.xml`**.
4. Haga clic en la sección **Agregar configuración** y llame a la segunda configuración `EveningReports`.
5. Haga clic en **Examinar** y seleccione **`Tutorial\mf-ExpReport2.xml`**.

Paso 2: usar el recurso global en la asignación

Ahora puede usar el recurso global que acaba de crear en la asignación. Para que esta lea el recurso global, siga estos pasos:

1. Abra la asignación **`Tutorial\Tut-ExpReport.mfd`**.
2. Haga doble clic el encabezado del componente de origen para abrir el cuadro de diálogo **Configuración del componente**.
3. Junto a **Archivo XML de entrada**, haga clic en **Examinar** y seleccione `MorningReports`. Haga clic en **Abrir**.
4. Se abrirá el cuadro de diálogo **Configuración del componente**. La ruta del archivo XML de entrada ahora se llama **`altova://file_resource/MorningReports`**, lo que indica que la ruta está usando un recurso global.

Paso 3: ejecutar la asignación con la configuración deseada

Ahora puede alternar entre los dos archivos XML de entrada antes de ejecutar la asignación:

- Para usar **`mf-ExpReport.xml`** como directorio de entrada, seleccione el elemento de menú **Herramientas | Configuración activa | Default**.

© 2016-2022 Altova GmbH

- Para usar **`mf-ExpReport2.xml`** como directorio de salida seleccione el elemento de menú **Herramientas| Configuración activa | ProductionDirectory**.

También puede seleccionar la configuración que desee en la lista desplegable de los **recursos globales** (*imagen siguiente*).

Para generar una vista previa del resultado de la asignación con cualquiera de las configuraciones haga clic en el panel **Resultados**.

10.4 Ejemplo: generar resultados en carpetas variables

En este apartado explicamos cómo usar las carpetas como recursos globales. Imaginemos que a veces necesita generar la salida de la asignación en varios directorios distintos. Con los recursos globales esto puede hacerse creando un alias de carpeta con dos configuraciones:

1. Configuración `Default`: genera la salida en **`C:\Test`**.
2. Configuración `Production`: genera la salida en **`C:\Production`**.

En los siguientes pasos explicamos cómo generar los resultados en distintas carpetas.

Paso 1: crear el recurso global

Primero debe crear un alias de carpeta. Para ello siga estos pasos:

1. Haga clic en el botón de la barra de herramientas (**Administrar recursos globales**). Si lo prefiere, vaya al menú **Herramientas** y haga clic en **Recursos globales**.
2. Haga clic en **Agregar | Carpeta** e introduzca un nombre en el cuadro de texto **Alias del recurso**. En este ejemplo podríamos usar `OutputDirectory`.
3. Haga clic en **Examinar** y seleccione la siguiente carpeta: C:\Test. Asegúrese primero de que esta carpeta ya existe en su sistema operativo.
4. Haga clic en e introduzca un nombre para la nueva configuración En este ejemplo, `ProductionDirectory`.
5. Haga clic en **Examinar** y seleccione la carpeta **`C:\Production`**. Asegúrese primero de que esta carpeta ya existe en su sistema operativo.

Paso 2: usar el recurso global en la asignación

Ahora ha creado el recurso global. Sin embargo, la asignación todavía no lo está usando. Para modificar la asignación para que use el alias de carpeta definido con anterioridad (Recurso global), siga estos pasos:

1. Abra la asignación **`Tutorial\Tut-ExpReport.mfd`**.
2. Haga doble clic el encabezado del componente de destino para abrir el cuadro de diálogo **Configuración del componente**.
3. Haga clic en **Cambiar a recursos globales** y luego en **Guardar**.
4. Guarde el archivo XML de salida como **`Output.xml`**. La ruta del archivo XML de salida ahora es **`altova://folder_resource/OutputDirectory/Output.xml`**, lo que indica que la ruta está usando un recurso global.

Paso 3: ejecutar la asignación con la configuración deseada

Ahora puede cambiar fácilmente el la carpeta de salida de la asignación antes de ejecutarla:

- Para usar **`C:\Test`** como directorio de salida, seleccione el elemento de menú **Herramientas| Configuración activa | Default**.
- Pra usar **`C:\Production`** como directorio de salida seleccione el elemento de menú **Herramientas| Configuración activa | ProductionDirectory**.

La salida de la asignación se escribe por defecto como un archivo temporal, a no ser que configure MapForce explícitamente para que escriba la salida en archivos permanentes. Para configurar MapForce para que genere archivos permanentes en lugar de temporales, siga estos pasos:

 © 2016-2022 Altova GmbH

1. En el menú **Herramientas** haga clic en el comando **Opciones**.
2. En la sección **General**, seleccione la opción **Escribir directamente en archivos de salida finales**.

10.5 Bases de datos como recursos globales

En este apartado explicamos cómo usar bases de datos como recursos globales. Imaginemos que necesita asignar datas desde bases de datos que tienen la misma estructura pero distintos datos. Al usar un recurso global de BD puede alternar entre esas bases de datos sin editar la asignación. Para conseguirlo crearemos un recurso global de BD (alias de BD) con dos configuraciones.

1. La configuración `Default` apuntará a la `BD estándar de desarrollo`. **`MapForceExamples\altova.mdb`**.
2. La configuración `ReleaseDatabase` apuntará a **`Tutorial\altova.mdb`**.

Para crear y configurar el alias de BD siga estos pasos que explicamos a continuación.

Paso 1: crear el recurso global

El primer paso es crear un alias de BD. Siga estos pasos:

1. Haga clic en el botón de la barra de herramientas (**Administrar recursos globales**). Si lo prefiere, vaya al menú **Herramientas** y haga clic en **Recursos globales**.
2. Haga clic en **Agregar | Base de datos** e introduzca un nombre descriptivo en el cuadro de texto **Alias del recurso**. En este ejemplo vamos a llamar a la configuración estándar `DevelopmentDatabase`.
3. Hga clic en **Elegir base de datos**, seleccione **Microsoft Access (ADO)** y navegue hasta **`MapForceExamples\altova.mdb`**.
4. Haga clic en y asigne un nombre a la segunda configuración.
5. Haga clic en **Elegir base de datos**, seleccione **Microsoft Access (ADO)** y navegue hasta **`Tutorial\altova.mdb`**.

Paso 2: usar el recurso global en la asignación

Ahora que ha creado el alias de BD debe modificar la asignación para poder usarlo:

1. Abra la asignación **`Tutorial\PersonDB.mfd`**.
2. Haga doble clic en el componente de base de datos para abrir el cuadro de diálogo **Configuración del componente**. Haga clic en **Modificar**.
3. Seleccione **Recursos globales** en el cuadro de diálogo **Seleccione una base de datos** y seleccione el alias `DevelopmentDatabase`. Por último, haga clic en **Conectarse**.
4. Cuando el cuadro de diálogo le pida que seleccione objetos de BD, no cambie la selección predeterminada, simplemente haga clic en **Aceptar**.

Los parámetros de conectividad se pueden cambiar haciendo clic en el botón de la barra de herramientas .

Paso 3: ejecutar la asignación con la configuración deseada

Ahora puede cambiar fácilmente a la BD deseada antes de ejecutar la asignación de datos:

- Para usar la configuración `DevelopmentDatabase` seleccione el elemento de menú **Herramientas| Configuración activa | Default**.
- Para usar la configuración `ReleaseDatabase` seleccione el elemento de menú **Herramientas| Configuración activa | ReleaseDatabase**.

© 2016-2022 Altova GmbH

También puede seleccionar la configuración que desee en la lista desplegable de los **recursos globales** (*imagen siguiente*).

Al alternar entre bases configuraciones, el cuadro de diálogo **Cambio en la configuración** le informa de que se ha modificado el recurso global. Haga clic en **Volver a cargar**.

Nota: las dos bases de datos de este ejemplo contienen datos parecidos, por lo que no hay diferencias en los resultados generados una vez se ha ejecutado la asignación.

10.6 Resultados de transformaciones de MapForce y StyleVision como recursos globales

Puede crear recursos globales que, en lugar de apuntar a un archivo estático, lean determinado archivo producido por una asignación de MapForce o por una transformación de StyleVision. En este apartado explicamos cómo crear recursos globales a partir del resultado de una transformación y a usar esos recursos globales en las diferentes aplicaciones de Altova.

Para que el resultado (salida) de una asignación esté disponible como recurso global tiene dos opciones: cambiar el lenguaje de transformación a BUILT-IN o hacer que la asignación contenga solamente componentes compatibles con el lenguaje BUILT-IN.

Nota:

- Los flujos que se mencionan más arriba son importantes si tiene varias aplicaciones de Altova instaladas en el mismo equipo.
- El resultado de las transformaciones de MapForce y StyleVision **no** se pueden consumir como recursos globales en los productos servidor de Altova ni en MapForce Basic Edition.

En el ejemplo siguiente explicamos cómo usar el resultado de una transformación de MapForce como recurso global.

Ejemplo: Resultados de transformaciones de MapForce

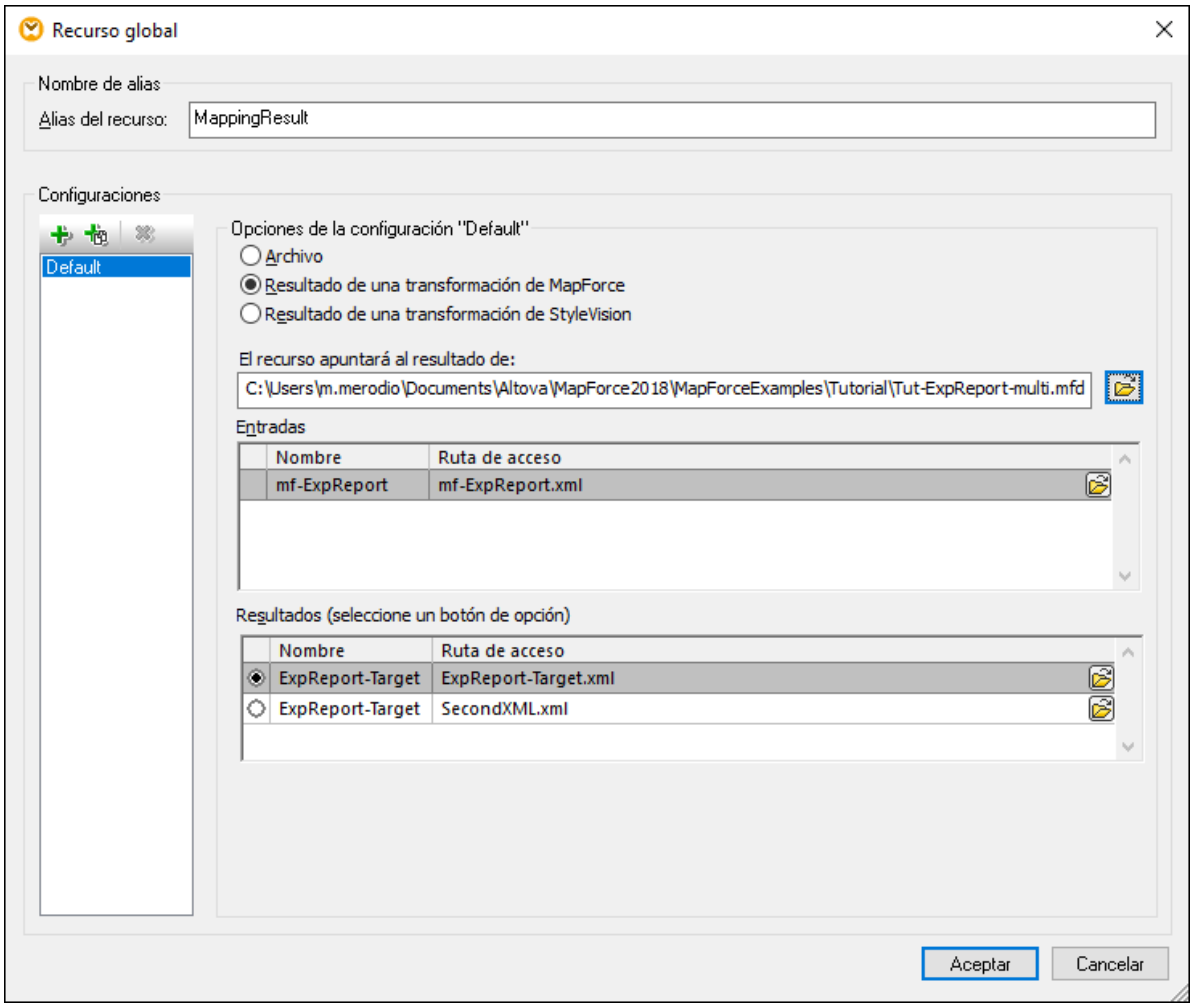
Con este ejemplo aprenderá a crear un flujo de trabajo entre Altova MapForce y Altova XMLSpy. Más concretamente, en este ejemplo explicamos cómo crear un recurso global a partir de una asignación de datos de MapForce, desencadenar la ejecución de esa asignación en XMLSpy y ver en XMLSpy los resultados generados por MapForce.

Paso 1: crear el recurso global

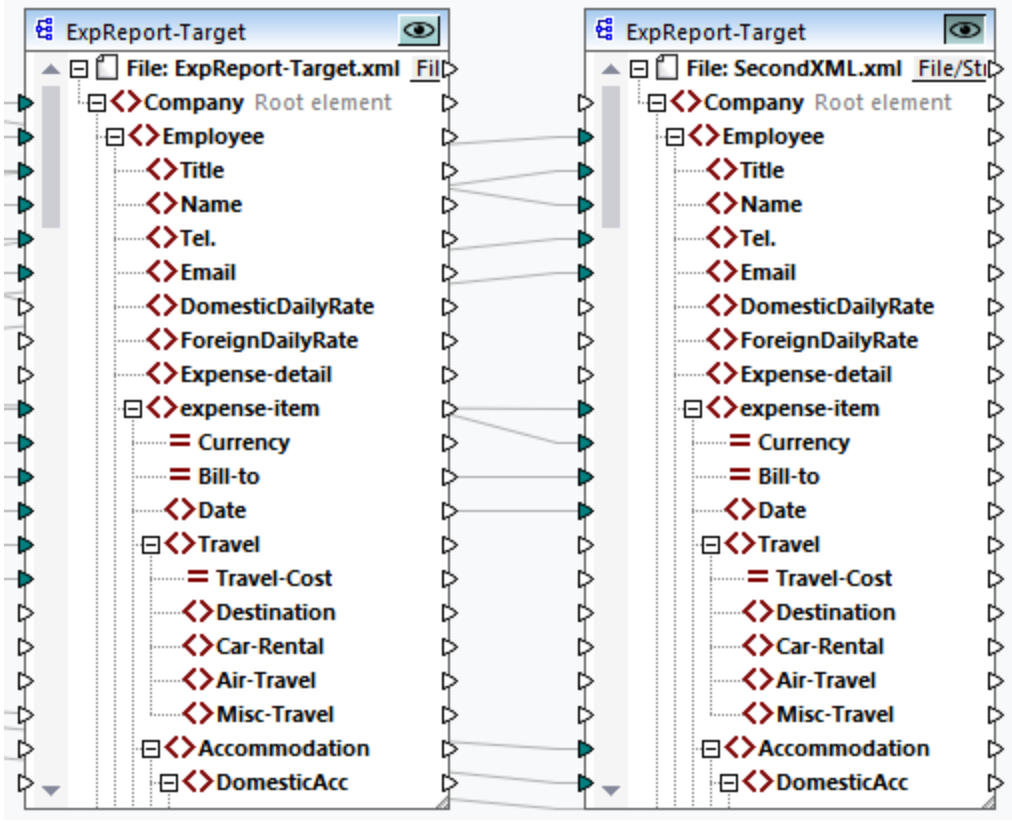
Para este paso puede usar MapForce o XMLSpy.

1. Haga clic en el botón de la barra de herramientas (**Administrar recursos globales**). Si lo prefiere, vaya al menú **Herramientas** y haga clic en **Recursos globales**.
2. Haga clic en **Agregar | Archivo** e introduzca un nombre descriptivo en el cuadro de texto **Alias del recurso**. En este ejemplo podríamos usar `MappingResult`.
3. Seleccione la opción **Resultado de una transformación de MapForce**.
4. Haga clic en **Examinar** y seleccione la asignación **`Tutorial\Tut-ExpReport-multi.mfd`**. Como puede ver a continuación, esta asignación tiene una entrada y dos salidas.

© 2016-2022 Altova GmbH



En la imagen siguiente se ven las dos salidas en el cuadro de diálogo **Recursos globales**. Ahora vamos a generar cada uno de los archivos de salida por separado en la carpeta **`C:\temp`** (*véase el paso 2 más abajo*).



Paso 2: Generar archivos de salida

Ahora queremos generar los dos archivos de salida (*imagen anterior*) en la carpeta **`C:\temp`** y cambiar sus nombres. Para conseguirlo vamos a crear una configuración para cada salida. Siga estos pasos:

1. En la sección *Resultados* del cuadro de diálogo **Recursos globales** haga clic en **Examinar** e introduzca **`C:\temp\file1.xml`**. Esta es la configuración predeterminada que producirá el primer archivo de salida.
2. Haga clic en Agregar configuración e introduzca un nombre para la nueva configuración (en este ejemplo, `Output2`). En la sección *Resultados* haga clic en el botón de opción que hay junto al segundo archivo (**`SecondXML.xml`**).
3. En la sección *Resultados* haga clic en **Examinar** e introduzca **`C:\temp\file2.xml`** como nombre de archivo de destino. Esta es la segunda configuración, que producirá el segundo archivo de salida.
4. Haga clic en **Aceptar**.

Paso 3: usar los recursos globales

A continuación explicamos cómo usar el recurso global que creamos en el paso anterior.

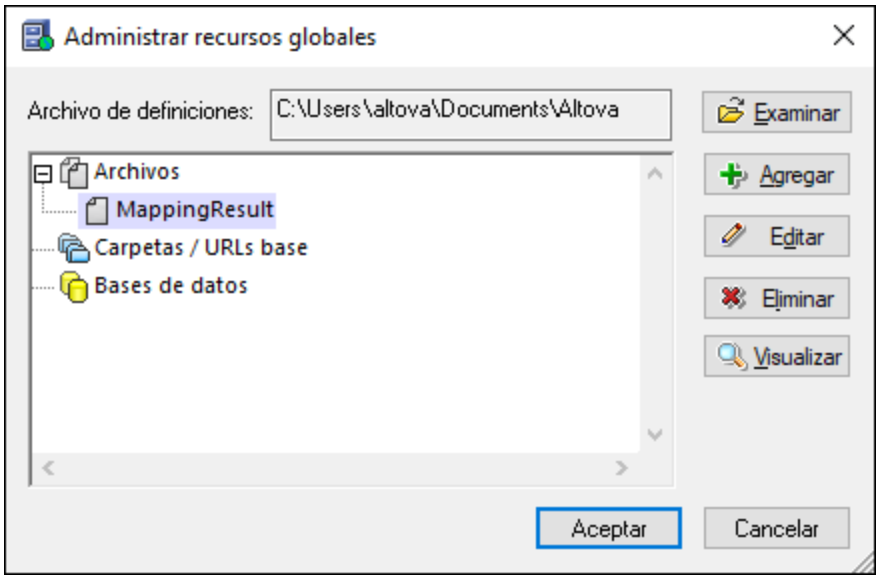
Configuración predeterminada

Para usar la configuración predeterminada en XMLSpy siga las instrucciones siguientes.

1. Ejecute XMLSpy.

 © 2016-2022 Altova GmbH

2. Vaya al menú **Herramientas** y haga clic en **Recursos globales**.
3. En la sección *Archivos* vaya al recurso global `MappingResult` y después haga clic en **Visualizar** (*imagen siguiente*). Ahora se ejecuta la asignación, que genera el archivo de salida predeterminado (file1.xml) y lo carga en el panel principal de XMLSpy. El archivo también se guarda como **`C:\temp\file1.xml`**.



Segunda configuración
Para desencadenar la ejecución de la asignación con la configuración alternativa, haga lo siguiente:

1. En el menú **Herramientas**, haga clic en **Configuración activa | Output2**.
2. Haga clic en **Volver a cargar** cuando aparezca el cuadro de diálogo de la imagen anterior.

Esto hará que el archivo de salida alternativo se cargue en el panel principal de XMLSpy. El archivo también se guarda como **`C:\temp\file2.xml`**.

10.7 Recursos globales en entornos de ejecución

En esta subsección explicamos cómo trabajar con los recursos globales en distintos entornos de ejecución. Estos son los apartados de esta subsección:

- Recursos globales en en el código generado 986
- Recursos globales en MapForce Server 987
- Recursos globales en FlowForce Server 987

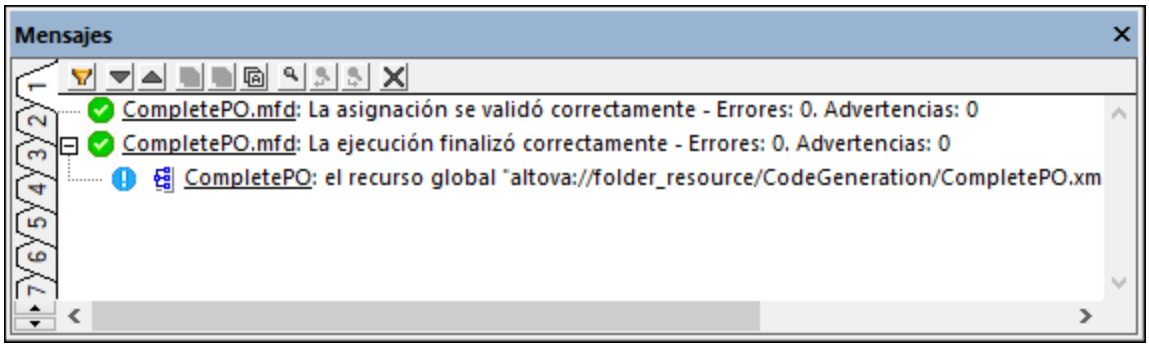
10.7.1 Recursos globales en en el código generado

En este apartado explicamos cómo usar recursos globales en el código generado. Para más información siga leyendo.

Recursos globales en XSLT, XSLT 2, XQuery
Recursos globales en XSLT, XSLT2 y XQuery Si usa los recursos globales al generar código XSLT o XSLT2, esto no afecta en absoluto a la hoja de estilos. Con o sin recursos globales, puede indicar de forma flexible los archivos de entrada y salida al ejecutar la hoja de estilos XSLT en su procesador XSLT. Lo mismo ocurre con el código XQuery generado.

Sin embargo existe una excepción: el archivo **`DoTransform.bat`** generado para ejecutarlo en RaptorXML. Los recursos globales que usa la asignación se resuelven en rutas en **`DoTransform.bat`**. Se aplica la configuración que esté seleccionada en la lista desplegable de recursos globales. Para obtener más información sobre cómo usar los recursos globales con RaptorXML, consulte la documentación de RaptorXML.

Recursos globales en C++, C#, Java
Al generar código de programa en C++, C# o Java se resuelven los recursos globales que usa la asignación. Por ejemplo, un alias de archivo o de carpeta definidos como recursos globales se convertirían en un archivo o una ruta de acceso a la carpeta. Si se selecciona una configuración en concreto de la lista desplegable de recursos globales, el código se genera para esa configuración. La ventana **Mensajes** muestra información sobre cómo se resolvió exactamente un recurso global (*imagen siguiente*).



Para generar código para una configuración de recursos globales en concreto, seleccione esa configuración en la lista desplegable de recursos globales antes de generar el código. También puede generar código desde la línea de comandos con los parámetros **`GLOBALRESOURCEFILE`** y **`GLOBALRESOURCECONFIG`**. Consulte Interfaz de la línea de comandos de MapForce 963 para obtener más información.

 © 2016-2022 Altova GmbH

No se puede hacer cambiar a o hacer referencia a recursos globales desde código generado. Lo que sí puede hacer es modificar el código para cambiar la ruta de los archivos de entrada y salida.

Nota: en C# o Java puede cambiar la ruta y el tipo de datos de la entrada o la salida1103.

10.7.2 Recursos globales en MapForce Server

Al compilar una asignación en un archivo de ejecución de MapForce Server (**.mfx**), cualquier referencia a los recursos globales que haya usado la asignación se mantiene sin resolver. Esto significa que debe suministrar estas referencias en el lado servidor para poder ejecutar correctamente la asignación. En concreto, en MapForce Server son necesarios los siguientes requisitos para ejecutar un archivo **.mfx** de datos que usa recursos globales:

1. *El archivo de definiciones de recursos globales*. En el equipo en el que se instala MapForce el archivo se llama **GlobalResources.xml** y puede encontrarlo en la carpeta **Documentos\Altova**. Puede copiar este archivo en el equipo en el que se ejecuta MapForce y también puede crear varios archivos si lo necesita. Consulte también el apartado Configurar recursos globales, parte 1970.
2. *El nombre de la configuración de los recursos globales*. Cada recurso global tiene una configuración predeterminada. Por supuesto, también puede crear otras configuraciones. Para más información consulte Configurar recursos globales, parte 2972.

En MapForce la ruta de acceso al archivo de definiciones de recursos globales y el nombre de la configuración de recursos globales se define o cambia desde la interfaz gráfica del usuario. En MapForce Server esta información se indica en tiempo de ejecución (*imagen siguiente*).

- Si ejecuta la asignación con la línea de comandos, use las opciones **--globalresourceconfig** y **--globalresourcefile** después de ejecutar el comando `run`, por ejemplo:

```
C:\Archivos de programa (x86)\Altova\MapForceServer2023\bin\MapForceServer.exe run
SomeMapping.mfx --globalresourcefile="C:
\Usuarios\usuario\Documentos\Altova\GlobalResources.xml" --
globalresourceconfig="Default"
```

- Si ejecuta la asignación a través de la API de MapForce Server debe llamar dos veces al método **SetOptions** antes de llamar al método **Run**. La primera llamada es necesaria para indicar la ruta al archivo de recursos globales como opción y la segunda es necesaria para indicar el nombre de la configuración de recursos globales.

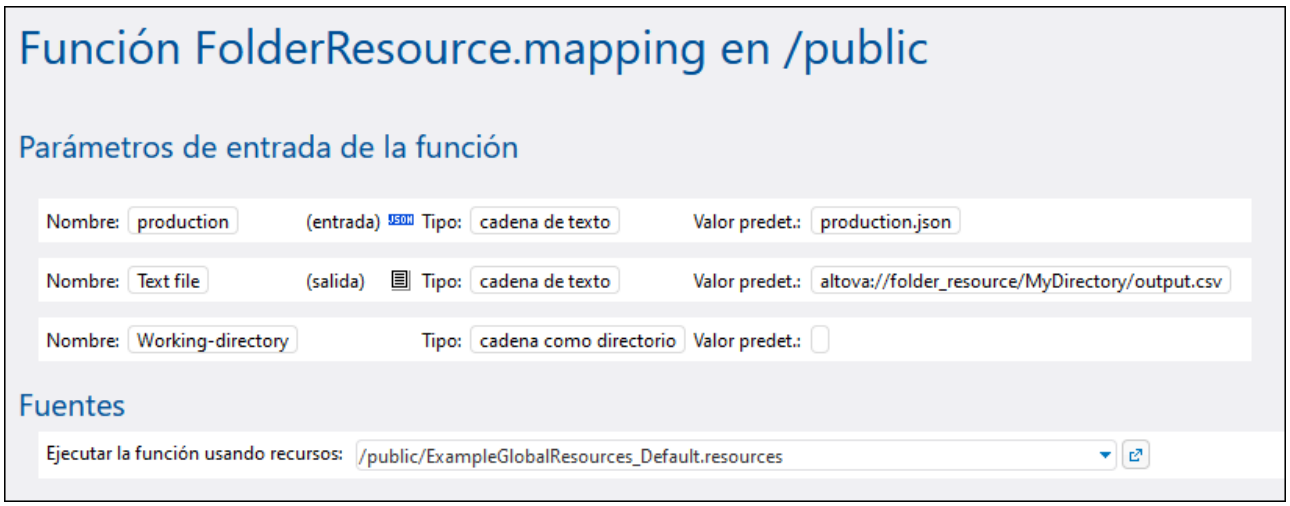
Para más información consulte la documentación de MapForce Server.

10.7.3 Recursos globales en FlowForce Server

En FlowForce Server los recursos globales no se almacenan en un archivo XML como en el caso de las aplicaciones de escritorio. En FlowForce Server los recursos son objetos reutilizables que pueden contener rutas de acceso a archivos o carpetas o a detalles de conexión de base de datos. Los recursos se pueden copiar, exportar e importar y usan el mismo mecanismo de acceso de usuario que otros objetos de FlowForce Server. En otras palabras, cualquier usuario de FlowForce puede usar cualquier recurso en sus funciones de

asignación, siempre que tengan los permisos correspondientes.

Una vez haya creado una asignación con recursos globales en MapForce, puede implementarla en FlowForce Server. En el momento de la implementación puede elegir si la asignación debe usar recursos globales. Si quiere que así sea, marque la casilla **Usar recursos** en el cuadro de diálogo de implementación Si no marca esta casilla, los recursos globales que use la asignación se resolverán en función de la configuración seleccionada. Si marca esta casilla, la función de asignación también necesitará recursos en FlowForce Server. La imagen siguiente muestra una función de asignación implementada en FlowForce que necesita recursos para ejecutarse. Observe que el primer parámetro toma la ruta de acceso predeterminada de un recurso.



Como se ve en la imagen, en FlowForce Server es la función de la asignación la que consume recursos globales, no el trabajo. Esa función de asignación lee en el recurso la ruta del primer archivo de entrada. Esto significa que todos los trabajos que usan esta función tienen la misma ruta a no ser que la sobrescriba desde la página de configuración del trabajo.

También puede implementar recursos globales en FlowForce Server como objetos independientes. Es decir, no necesita implementar una asignación primero para poder implementar un recurso global. Para más información consulte el apartado *Implementar recursos en FlowForce Server*.

Para más información sobre el consumo de recursos en FlowForce Server consulte la documentación de FlowForce Server.

Implementar recursos en FlowForce Server

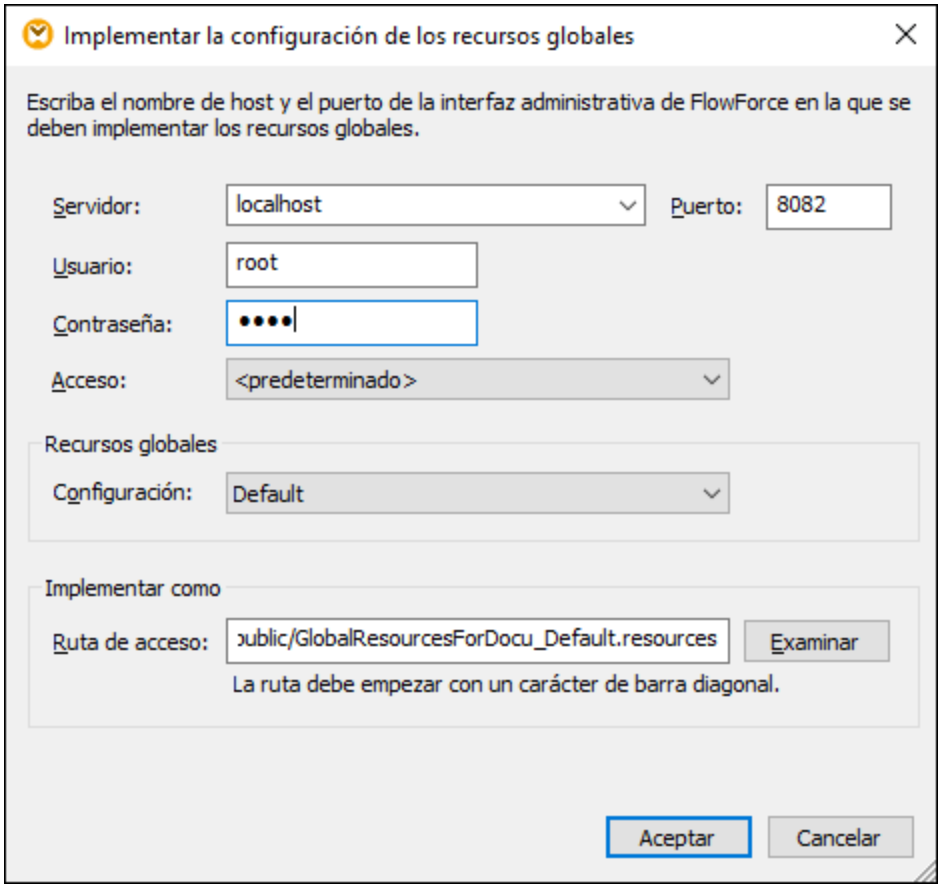
En FlowForce puede implementar recursos globales creados con MapForce. Para ello debe elegir la configuración con la que quiere implementar en el servidor el recurso en cuestión. Si necesita implementar todas las configuraciones de un mismo recurso en el servidor, entonces puede implementar ese recurso varias veces y elegir cada vez una configuración distinta al implementarlo. También puede cambiar el nombre de cada recurso global en el servidor y elegir el contenedor de destino en el servidor.

Puede implementar los recursos globales en FlowForce Server al mismo tiempo que implementa la asignación o en otro momento. Para implementar recursos globales en FlowForce Server:

1. Ejecute MapForce.

 © 2016-2022 Altova GmbH

2. Haga clic en el botón **Administrar recursos globales** de la barra de herramientas. Si lo prefiere, vaya al menú **Herramientas** y haga clic en **Recursos globales**.
3. Haga clic en **Implementar en el servidor**. Se abre el cuadro de diálogo **Implementar la configuración de los recursos globales** (*imagen siguiente*).



4. Introduzca los detalles de la conexión en FlowForce Server (servidor, puerto, usuario, contraseña y método de inicio de sesión). Estos parámetros son los mismos que los necesarios para implementar asignaciones en FlowForce Server 958.
5. Seleccione una configuración de la lista **Configuraciones**. Esta lista incluye todas las configuraciones del archivo de recursos globales actual 970. Solo se puede usar una configuración de recurso global por implementación. Si necesita implementar todas las configuraciones en el servidor, entonces tiene que implementar el mismo recurso varias veces y elegir cada vez una configuración distinta.
6. Seleccione una ruta de destino, que será donde se guarde el recurso en el servidor. Haga clic en **Examinar** para seleccionar un contenedor de destino de FlowForce Server o crear uno nuevo si lo prefiere.
7. Haga clic en **Aceptar**.

Cuando se implementa un recurso global en FlowForce Server, el resultado correspondiente aparece en la ventana **Mensajes**.

Nota: los recursos globales que se ejecutan en otras aplicaciones de Altova *no* son compatibles con entornos de servidor. Para más información, consulte Recursos globales de Altova 982.

11 Catálogos en MapForce

MapForce es compatible con un subconjunto del mecanismo de catalogación XML OASIS. El mecanismo de catalogación permite a MapForce recuperar de carpetas locales del usuario los esquemas (y hojas de estilos y otros archivos) usados con frecuencia. Esto incrementa la velocidad global de procesamiento, permite al usuario trabajar sin conexión (es decir, sin estar conectado a una red) y mejora la portabilidad de los documentos (porque los identificadores URI se tienen que cambiar sólo en los archivos de catálogo).

A continuación describimos cómo funciona el mecanismo de catalogación en MapForce.

- Cómo funcionan los catálogos 991
- Estructura de los catálogos en MapForce 993
- Personalizar catálogos 995
- Memoria insuficiente 997

Para más información consulte la especificación XML Catalogs.

© 2016-2022 Altova GmbH

11.1 Funcionamiento de los catálogos

Los catálogos se pueden usar para redirigir los esquemas DTD y XML. El concepto tras los mecanismos es el mismo en los dos casos, pero los detalles son distintos; los explicamos a continuación.

DTDs

Los catálogos se suelen usar para redirigir una llamada a un DTD o un URI local. Para ello se asignan, en el archivo de catálogo, identificadores públicos o del sistema al URI local necesario. De este modo, cuando se lee la declaración **DOCTYPE** en un archivo XML, el identificador de sistema o público localiza el recurso local necesario con ayuda de la asignación del archivo de catálogo.

Para los esquemas más utilizados el identificador **PUBLIC** suele estar predefinido y, por tanto, sólo hace falta que el URI del archivo de catálogo asigne el identificador **PUBLIC** a la copia local correcta. Cuando se analiza el documento XML, se lee el identificador **PUBLIC** del documento. Si se encuentra este identificador en un archivo de catálogo, se buscará la URL correspondiente del archivo de catálogo y se leerá el esquema desde esta ubicación. Por ejemplo, imaginemos que abrimos este archivo SVG en MapForce:

```
<?xml version="1.0" standalone="no"?>
<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 1.1//EN"
"http://www.w3.org/Graphics/SVG/1.1/DTD/svg11.dtd">

<svg width="20" height="20" xml:space="preserve">
  <g style="fill:red; stroke:#000000">
     <rect x="0" y="0" width="15" height="15"/>
     <rect x="5" y="5" width="15" height="15"/>
  </g>
</svg>
```

Se busca en el catálogo el identificador **PUBLIC** de este archivo SVG. Imaginemos que el archivo de catálogo contiene esta entrada:

```
<catalog>
  ...
    <public publicId="-//W3C//DTD SVG 1.1//EN" uri="schemas/svg/svg11.dtd"/>
  ...
</catalog>
```

En este caso hay una coincidencia del identificador **PUBLIC**. La consulta del SVG DTD se redirige al URL **schemas/svg/svg11.dtd** (que es relativo al archivo de catálogo). Este es un archivo local y se usa como DTD para el archivo SCG. Si en el catálogo no hay una asignación para el identificador **Public**, entonces se usa la URL del documento XML (en el archivo SVG del ejemplo anterior la URL de Internet es: **http://www.w3.org/Graphics/SVG/1.1/DTD/svg11.dtd**).

Esquemas XML

En MapForce también puede usar catálogos para redireccionar a un **esquema XML**. En el archivo de instancia XML, la referencia al esquema sucederá en el atributo **xsi:schemaLocation** del elemento de documento de nivel superior del documento XML. Por ejemplo:

```
xsi:schemaLocation="http://www.xmlspy.com/schemas/orgchart OrgChart.xsd"
```

El valor del atributo **xsi:schemaLocation** tiene dos partes: un espacio de nombres (verde) y un URI (resaltado). La parte del espacio de nombres se usa en el catálogo para la asignación con el recurso alternativo. Por ejemplo, esta entrada de catálogo redirige la referencia del esquema anterior a un esquema en una ubicación alternativa.

```
<uri name="http://www.xmlspy.com/schemas/orgchart" uri="C:\MySchemas\OrgChart.xsd"/>
```

Por lo general, la parte URI del valor del atributo **xsi:schemaLocation** es una ruta a la ubicación real del esquema. Sin embargo, si se hace referencia al esquema a través de un catálogo, no es necesario que la parte URI apunte a un esquema XML real, aunque el esquema debe existir para que el atributo **xsi:schemaLocation** siga siendo válido desde el punto de vista léxico. Por ejemplo, el valor `foo` sería suficiente para que la parte URI del valor del atributo sea válida.

© 2016-2022 Altova GmbH

11.2 Estructura de los catálogos de MapForce

Al iniciarse, MapForce carga un archivo llamado **RootCatalog.xml** (*cuya estructura aparece a continuación*), que contiene una lista de los archivos de catálogo que se buscarán. El usuario puede modificar esta lista y añadir tantos archivos de catálogo como desee, escribiendo cada archivo en un elemento **nextCatalog**. La aplicación busca cada uno de estos archivos de catálogo y sus URI se resuelven de acuerdo con sus asignaciones.

Extracto de RootCatalog.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<catalog xmlns="urn:oasis:names:tc:entity:xmlns:xml:catalog"
   xmlns:spy="http://www.altova.com/catalog_ext"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:schemaLocation="urn:oasis:names:tc:entity:xmlns:xml:catalog Catalog.xsd">
  <nextCatalog catalog="%PersonalFolder%/Altova/%AppAndVersionName%/CustomCatalog.xml"/>
  <!-- Include all catalogs under common schemas folder on the first directory level -->
  <nextCatalog spy:recurseFrom="%CommonSchemasFolder%" catalog="catalog.xml"
spy:depth="1"/>
  <nextCatalog spy:recurseFrom="%ApplicationWritableDataFolder%/pkgs/.cache"
catalog="remapping.xml" spy:depth="0"/>
  <nextCatalog catalog="CoreCatalog.xml"/>
</catalog>
```

El extracto anterior hace referencia a un catálogo personalizado (**CustomCatalog.xml**) y a un conjunto de catálogos que localizan los esquemas usados a menudo (como W3C XML Schemas y el esquema SVG).

- **CustomCatalog.xml** se encuentra en su carpeta personal (a la que puede acceder con la variable **%PersonalFolder%**). Es un archivo base en el que el usuario puede crear asignaciones propias. Puede añadir asignaciones a **CustomCatalog.xml** para cualquier esquema que necesite que no aparezca en los archivos de catálogo de la carpeta Common Schemas. Debe usar los elementos compatibles del mecanismo de catálogo OASIS (*véase más abajo*).
- La carpeta Common Schemas Folder (a la que se accede con la variable **%CommonSchemasFolder%**) contiene un conjunto de esquemas de uso habitual. Dentro de cada una de estas carpetas hay un archivo **catalog.xml** que asigna identificadores públicos y/o del sistema a URIs que apuntan a copias locales de los esquemas correspondientes.
- **CoreCatalog.xml** está en la carpeta de la aplicación XMLSpy y se usa para localizar esquemas y hojas de estilos que usan los procesos específicos de XMLSpy, como los archivos SPS de StyleVision, que se usan para generar documentos XML para la Vista Authentic de Altova.

Variables de ubicación

Las variables que se usan en **RootCatalog.xml** (*véase el extracto más arriba*) tienen estos valores:

%PersonalFolder%	La carpeta personal del usuario, por ejemplo C:\Users\<name>\Documents.
%CommonSchemasFolder%	C:\ProgramData\Altova\Common2023\Schemas
%ApplicationWritableDataFolder%	C:\ProgramData\Altova

Ubicación de los archivos de catálogo y los esquemas
Estas son las ubicaciones de los distintos archivos de los catálogos.

- **`RootCatalog.xml`** y **`CoreCatalog.xml`** se instalan en la carpeta de aplicación de XMLSpy.
- **`CustomCatalog.xml`** está ubicado en su carpeta **`MisDocumentos\Altova\MapForce`**.
- Cada archivo **`catalog.xml`** está en una carpeta de esquema y estas carpetas están dentro de la carpeta común de esquemas.

© 2016-2022 Altova GmbH

11.3 Personalizar catálogos

Cuando cree entradas en el archivo **CustomCatalog.xml** (o en cualquier otro archivo de catálogo que sea leído por MapForce), utilice únicamente los elementos que aparecen a continuación de la especificación de catálogos OASIS. En la lista que aparece más adelante explicamos los valores de los atributos de cada elemento. Si desea consultar una descripción más detallada, visite la página de la especificación XML Catalogs. Tenga en cuenta que cada uno de los elementos del subconjunto pueden llevar el atributo **xml:base**, que se usa para especificar el URI base del elemento.

- `<public publicId="PublicID of Resource" uri="URL of local file"/>`
- `<system systemId="SystemID of Resource" uri="URL of local file"/>`
- `<uri name="filename" uri="URL of file identified by filename"/>`
- `<rewriteURI uriStartString="StartString of URI to rewrite" rewritePrefix="String to replace StartString"/>`
- `<rewriteSystem systemIdStartString="StartString of SystemID" rewritePrefix="Replacement string to locate resource locally"/>`

Tenga en cuenta que:

- Cuando no exista un identificador público, como es el caso de casi todas las hojas de estilos, el identificador de sistema se puede asignar directamente a una URL con el elemento **system**.
- Un URI se puede asignar a otro URI con el elemento `uri`.
- Los elementos `rewriteURI` y `rewriteSystem` sirven para volver a escribir la parte inicial de un URI o identificador de sistema respectivamente. Gracias a ello se puede sustituir el principio de la ruta de acceso de un archivo y, por consiguiente, se puede apuntar a otro directorio. Para más información sobre estos elementos, consulte la especificación XML Catalogs.

A partir de su versión de 2014 MapForce cumple escrupulosamente con la especificación de catálogos XML XML Catalogs specification (OASIS Standard V1.1, 7 October 2005) Esta especificación separa estrictamente las consultas por identificador externo (las realizadas con un ID público o de sistema) de las búsquedas por URI (los URI que no son ID públicos ni de sistema). Por tanto, los URIs de espacios de nombres deben considerarse simplemente como URIs (no como IDs públicos o del sistema) y deben usarse como búsquedas por URI y no como consultas por identificador externo. En las versiones de MapForce previas a la de 2014 los URIs de espacios de nombres de esquemas se traducían con asignaciones **<public>**. Sin embargo, a partir de la versión 2014 es necesario utilizar asignaciones **<uri>**.

Antes de la versión v2014: `<public publicID="http://www.MyMapping.com/ref" uri="file:///C:/MyDocs/Catalog/test.xsd"/>`

A partir de la versión 2014: `<uri name="http://www.MyMapping.com/ref" uri="file:///C:/MyDocs/Catalog/test.xsd"/>`

Cómo encuentra MapForce un esquema de referencia

Para hacer referencia a un esquema desde un documento XML se usa el atributo **xsi:scemaLocation** (*más abajo*). El valor del atributo **xsi:schemaLocation** tiene dos partes: un espacio de nombres (verde) y un URI (resaltado).

```
xsi:schemaLocation="http://www.xmlspy.com/schemas/orgchart OrgChart.xsd"
```

A continuación damos los pasos que sigue MapForce de forma secuencial para encontrar un esquema de referencia. El esquema se carga en el primer paso que se ejecuta correctamente.

1. Consulte en el catálogo la parte del URI del valor **`xsi:schemaLocation`**. Si se encuentra una asignación, y esto incluye las asignaciones **`rewriteURI`**, use el URI resultante para cargar el esquema.
2. Consulte en el catálogo la parte del espacio de nombres del valor **`xsi:schemaLocation`**. Si se encuentra una asignación, y esto incluye las asignaciones **`rewriteURI`**, use el URI resultante para cargar el esquema.
3. Use la parte URI del valor **`xsi:schemaLocation`** para cargar el esquema.

Especificaciones de XML Schema

La información de la especificación XML Schema está integrada en MapForce y esta información interna se usa para validar los documentos de esquema XML (`.xsd`). Por tanto, en los documentos de esquema XML no se deberían hacer referencia a ningún esquema que defina la especificación XML Schema.

El archivo **`catalog.xml`** de la carpeta `%AltovaCommonSchemasFolder%\Schemas\schema` incluye referencias a las DTD que implementan especificaciones antiguas de XML Schema. Rogamos no valide sus documentos de esquema XML con estos esquemas. Los archivos referenciados se incluyen con el único objetivo de aportar información a MapForce para sus ayudantes de entrada, en caso de que el usuario quiera crear documentos basados en estas recomendaciones.

 © 2016-2022 Altova GmbH

11.4 Variables de entorno

En el elemento `nextCatalog` puede utilizar algunas variables de entorno Shell para indicar la ruta de acceso a las ubicaciones del sistema (*ver el fragmento anterior del archivo RootCatalog.xml*). Estas son las variables de entorno Shell compatibles:

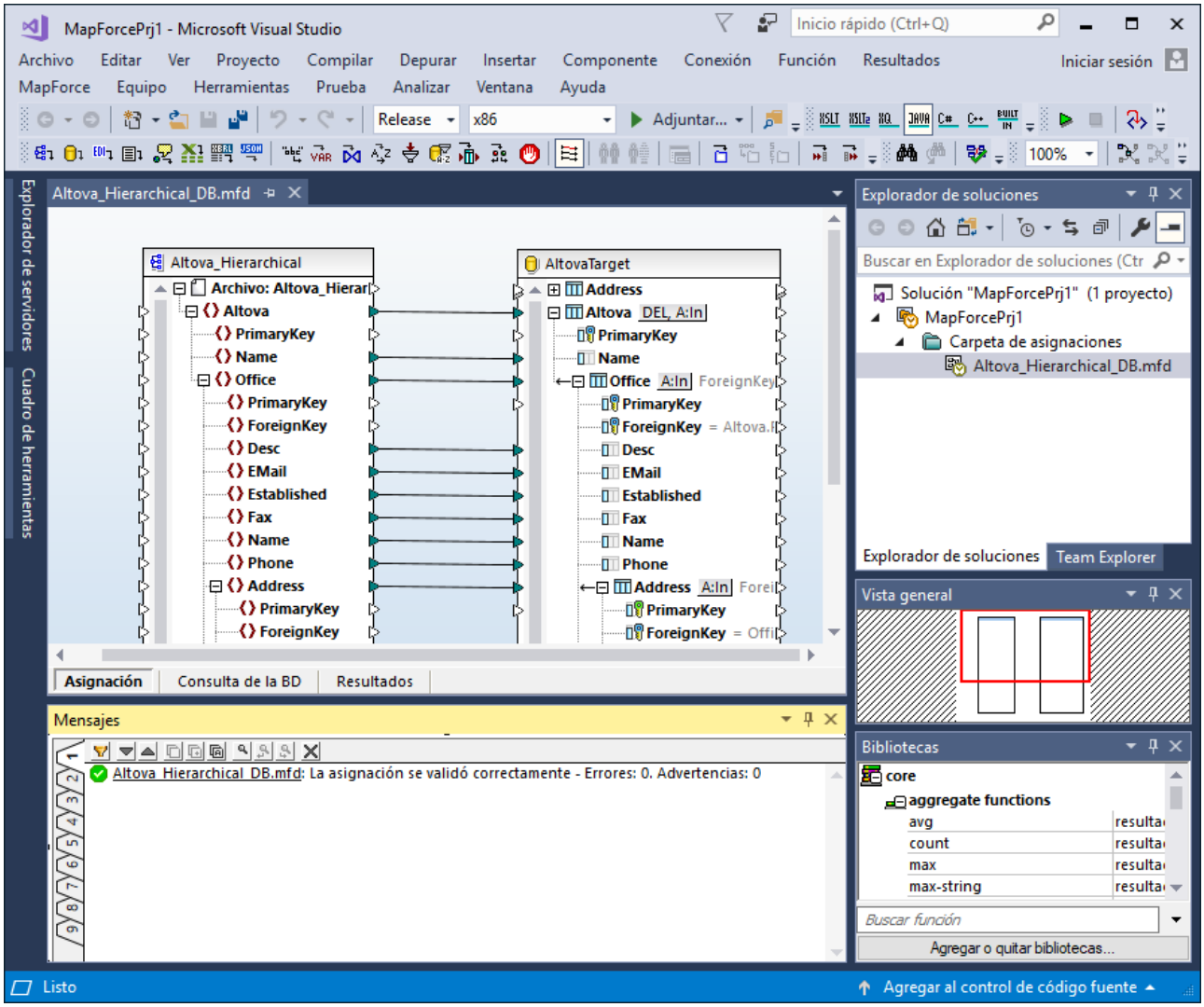
`%PersonalFolder%`	Ruta de acceso completa de la carpeta personal del usuario actual, por ejemplo `C:\Usuarios\<nombre>\Documentos`
`%CommonSchemasFolder%`	`C:\ProgramData\Altova\Common2023\Schemas`
`%ApplicationWritableDataFolder%`	`C:\ProgramData\Altova`
`%AltovaCommonFolder%`	C:\Archivos de programa\Altova\Common2023
`%DesktopFolder%`	Ruta de acceso completa de la carpeta Escritorio del usuario actual.
`%ProgramMenuFolder%`	Ruta de acceso completa de la carpeta del menú Programas del usuario actual.
`%StartMenuFolder%`	Ruta de acceso completa de la carpeta del menú Inicio del usuario actual.
`%StartUpFolder%`	Ruta de acceso completa de la carpeta Inicio del usuario actual.
`%TemplateFolder%`	Ruta de acceso completa de la carpeta de plantillas del usuario actual.
`%AdminToolsFolder%`	Ruta de acceso completa del directorio del sistema de archivos que almacena las herramientas administrativas del usuario actual.
`%AppDataFolder%`	Ruta de acceso completa de la carpeta Datos de programa del usuario actual.
`%CommonAppDataFolder%`	Ruta de acceso completa al directorio del sistema de archivos que sirve como repositorio de datos para aplicaciones locales (no roaming).
`%FavoritesFolder%`	Ruta de acceso completa de la carpeta Favoritos del usuario actual.
`%PersonalFolder%`	Ruta de acceso completa de la carpeta personal del usuario actual.
`%SendToFolder%`	Ruta de acceso completa de la carpeta SendTo del usuario actual.
`%FontsFolder%`	Ruta de acceso completa de la carpeta Fuentes del sistema.
`%ProgramFilesFolder%`	Ruta de acceso completa de la carpeta Archivos de programa del usuario actual.
`%CommonFilesFolde`	Ruta de acceso completa de la carpeta Common files del usuario actual.

r%	
%WindowsFolder%	Ruta de acceso completa de la carpeta Windows del usuario actual.
%SystemFolder%	Ruta de acceso completa de la carpeta System del usuario actual.
%LocalAppDataFolder%	Ruta de acceso completa al directorio del sistema de archivos que sirve como repositorio de datos para aplicaciones locales (no roaming).
%MyPicturesFolder%	Ruta de acceso completa a la carpeta Mis imágenes.

© 2016-2022 Altova GmbH

12 Complemento de MapForce para Visual Studio

Puede integrar MapForce 2023 en Microsoft Visual Studio, versiones 2012/2013/2015/2017/2019/2022. Esto permite combinar las funciones de asignación de datos de MapForce con el entorno de desarrollo de Visual Studio. Cuando tenga el complemento de MapForce habilitado, podrá crear proyectos y archivos de asignación de datos desde Visual Studio directamente (*imagen siguiente*).



Instalación

Siga estas instrucciones para instalar el complemento de MapForce para Visual Studio:

1. Instale Microsoft Visual Studio 2012/2013/2015/2017/2019/2022. Tenga en cuenta que a partir de su versión 2022, Visual Studio solo está disponible como aplicación de 64 bits.
2. Instale MapForce (ediciones Enterprise o Professional Edition). Si tiene instalada la versión 2022 o una más avanzada, debe instalar la versión de 64 bits de MapForce.

3. Descargue y ejecute el paquete de integración de MapForce. Este paquete está disponible en la página de descargas de MapForce (ediciones Enterprise y Professional) de www.altova.com/es.

Tras instalar el paquete de integración podrá empezar a usar MapForce en el entorno de Visual Studio.

Nota importante

Debe usar el paquete de integración correspondiente a su versión de MapForce (la versión actual es 2023). El paquete de integración no es específico de ninguna versión, por lo que se puede usar tanto para la edición Enterprise como Professional.

Información sobre menús y funciones

Si habilita el complemento de MapForce para Visual Studio podrá acceder a distintos menús y funciones de MapForce (*imagen siguiente*). Puede personalizar los menús y barras de herramientas de MapForce en el menú **Herramientas | Personalizar** de Visual Studio.

Nota: en Visual Studio 2019 y versiones posteriores se puede acceder a las funciones de MapForce desde el menú **Extensiones** de Visual Studio. En las versiones anteriores de Visual Studio puede acceder a las funciones de MapForce desde los menús principales de Visual Studio.

Temas

Puede seleccionar uno de los temas de MapForce en el menú **MapForce** de Visual Studio. Los temas son Clásico, Ligero y Oscuro.

Crear y abrir archivos/proyectos

Si habilita el complemento de MapForce para Visual Studio, este le permitirá crear, abrir y trabajar con asignaciones y proyectos de asignación directamente en Visual Studio. Para crear un archivo de asignación nuevo en Visual Studio, use el comando de menú **Archivo | Nuevo**. Para crear un proyecto nuevo, use el comando de menú **Archivo | Proyecto nuevo**. Para abrir archivos o proyectos de asignación puede usar estos menús de Visual Studio: **Archivo | Abrir | Archivo** o **Archivo | Abrir | Proyecto/Solución**. Después solo tiene que buscar los tipos de archivos relacionados con MapForce.

Recursos globales

Los recursos globales 969 de MapForce están disponibles en el menú de Visual Studio **MapForce | Administrar recursos globales**. En Visual Studio 2019 y versiones posteriores, el menú correspondiente es **Extensiones | MapForce | Administrar recursos globales**.

Depuración de datos

Una vez haya abierto el archivo de asignación, los comandos de depuración de asignaciones están en el menú **MapForce | Depuración** y en la barra de herramientas **Depuración**. También puede iniciar la depuración 1040 con combinaciones de teclas de acceso rápido. En Visual Studio 2019 y versiones posteriores, el menú correspondiente es **Extensiones | MapForce | Depuración**.

Opciones de MapForce

Puede encontrar las opciones de MapForce en el menú de Visual Studio **Herramientas | Opciones de MapForce**

 © 2016-2022 Altova GmbH

- Configurar el panel Asignación

  Si abre una asignación en el panel principal de Visual Studio se habilita el menú **Vista | MapForce**. Este menú incluye las mismas opciones que el menú **Vista** de la versión independiente de MapForce.

- Ventana Bibliotecas

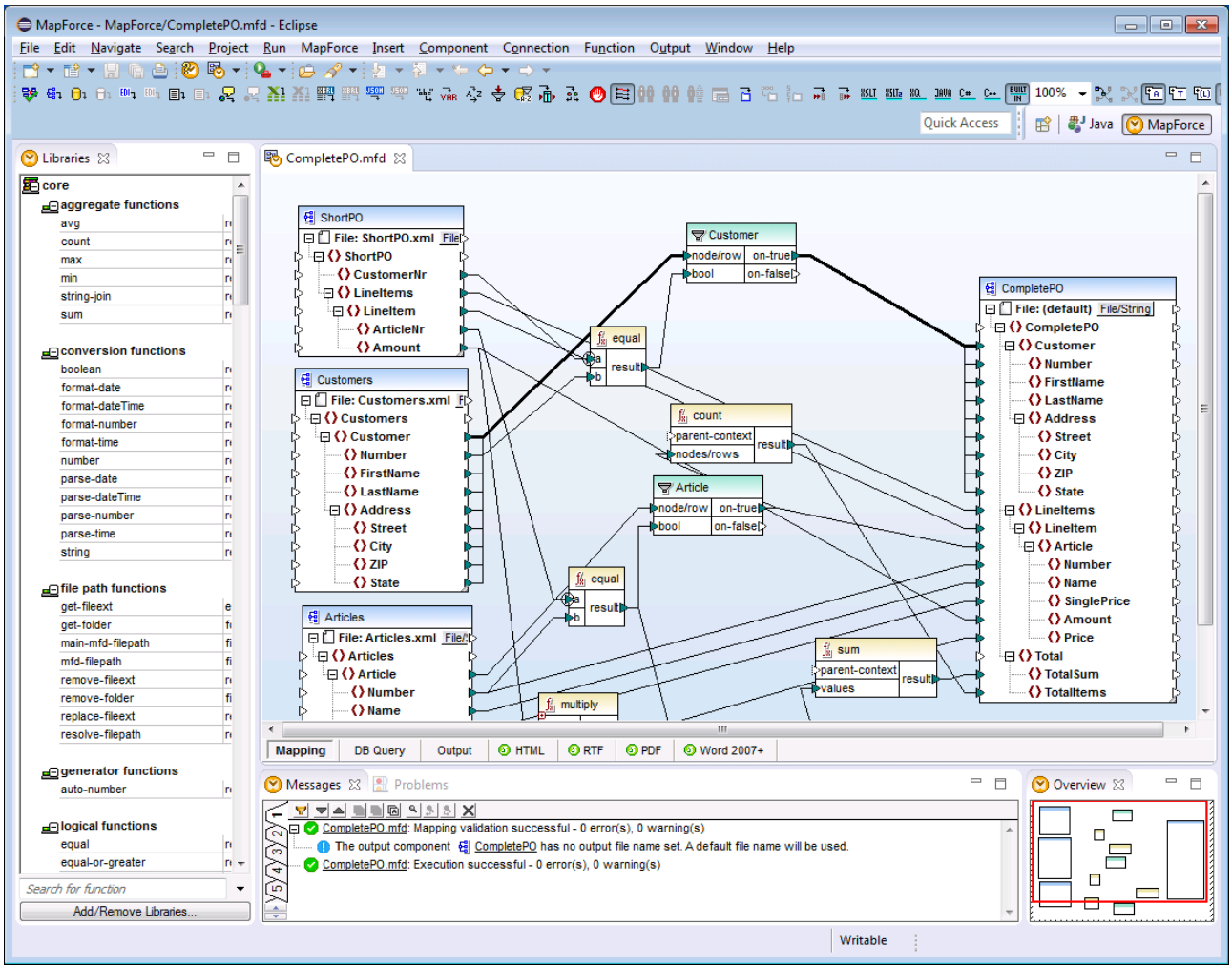
  En la ventana **Bibliotecas** de MapForce no se puede ver Visual Studio, pero lo puede habilitar desde el menú **Vista | MapForce | Ventana Bibliotecas** (este menú se habilita en Visual Studio una vez se abre un archivo de asignación). Una vez se habilita la ventana **Bibliotecas**, puede acoplarla en distintas partes de la interfaz.

- Ayuda y soporte técnico

  Los menús de MapForce **Ayuda**, **Centro de soporte técnico**, **Buscar actualizaciones** y **Acerca de** se encuentran en el menú de Visual Studio **Ayuda | Ayuda de MapForce**

13 Complemento de MapForce para Eclipse

Eclipse es un marco de trabajo de código abierto que incluye varios tipos de aplicaciones en forma de complementos. MapForce Enterprise y Professional se pueden integrar en Eclipse 2022-09, 2022-06, 2022-03, 2021-12 y podrá acceder a todas las funciones de MapForce desde Eclipse directamente.



Complemento de MapForce Enterprise Edition para Eclipse

Los apartados de esta sección ofrecen instrucciones para instalar y usar el complemento de MapForce en Eclipse.

- Instalación 1003
- Perspectiva MapForce 1005
- Acceso a menús y funciones frecuentes 1008
- Trabajar con asignaciones y proyectos 1012
- Ampliar el complemento de MapForce para Eclipse 1022

 © 2016-2022 Altova GmbH

13.1 Instalación

Requisitos

- Eclipse 2022-09, 2022-06, 2022-03, 2021-12 (http://www.eclipse.org).
- Un Java JRE/JDK para la plataforma de 64 bits
- MapForce Enterprise o Professional Edition de 64 bits

Nota: todos los requisitos anteriores deben corresponder a la plataforma de 64 bits. El complemento para Eclipse de versiones anteriores de 32 bits ya no es compatible, aunque es posible que aún funcione.

Una vez haya comprobado que cumple con los requisitos de la lista anterior, puede instalar el paquete de integración de MapForce de 64 bits para integrar la aplicación en Eclipse. La integración se puede llevar a cabo al instalar el paquete de integración o de forma manual desde Eclipe una vez haya instalado el paquete de integración. El paquete de integración de MapForce se puede descargar en https://www.altova.com/es/components/download.

Nota: debe cerrar Eclipse para poder instalar o desinstalar el paquete de integración de MapForce.

Instalar el complemento de MapForce para Eclipse

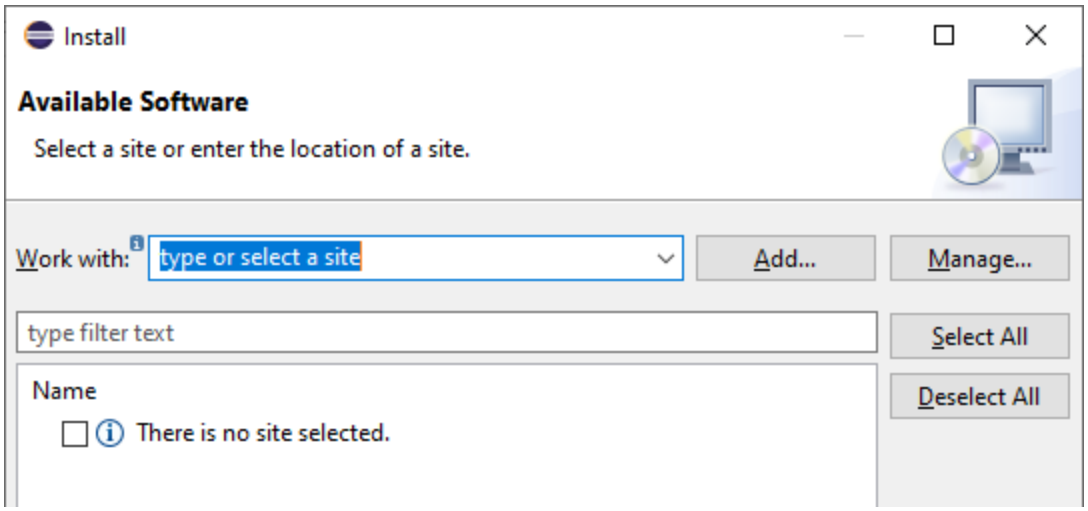
Puede integrar el complemento de MapForce en Eclipse ante la instalación del paquete de integración de MapForce. Para ello:

1. Ejecute el paquete de integración de MapForce para iniciar el asistente de instalación.
2. Cuando el instalador se lo pida seleccione *Instalar el complemento para Eclipse* y haga clic en **Aceptar**.
3. Cuando le pida que escoja cómo quiere integrar el complemento de MapForce en Eclipse seleccione *Permitir integración de MapForce en Eclipse* y navegue hasta el directorio en el que está el ejecutable de Eclipse (`eclipse.exe`).
4. Haga clic en **Siguiente** y complete la instalación.

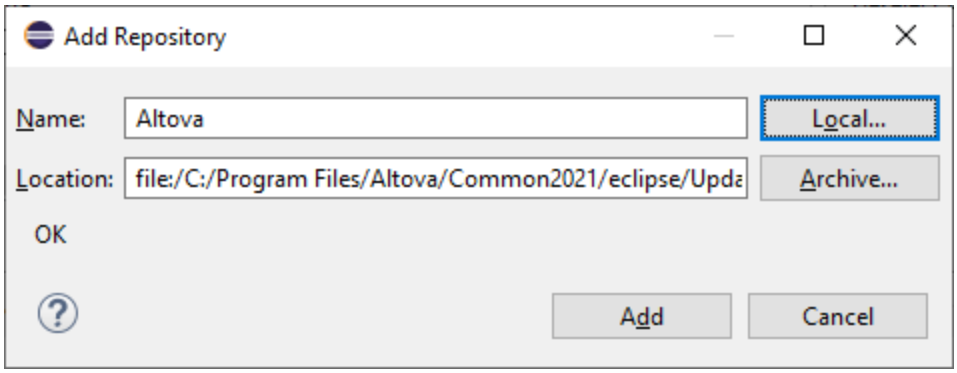
Si elige la integración automática los menús y la perspectiva de MapForce se habilitarán en Eclipse la siguiente vez que lo inicie.

Integrar el complemento de MapForce para Eclipse manualmente

1. En Eclipse seleccione el comando de menú **Ayuda | Instalar software nuevo**.
2. En el cuadro de diálogo Instalar haga clic en **Agregar**.



3. En el cuadro de diálogo "Agregar repositorio" haga clic en **Local**. Navegue hasta la carpeta `C:\Program Files\Altova\Common2023\eclipse\UpdateSite` y selecciónsela. Elija un nombre para el sitio (por ejemplo, "Altova").



4. Repita los pasos 2-3 del punto anterior pero esta vez elija la carpeta `C:\Program Files\Altova\MapForce2023\eclipse\UpdateSite` y un nombre como "Altova MapForce".
5. En el cuadro de diálogo seleccione *Solo sitios locales*. A continuación seleccione la carpeta "Altova category" y haga clic en **Siguiente**.
6. Revise los elementos que va a instalar y haga clic en **Siguiente** para continuar.
7. Marque la casilla correspondiente para aceptar el acuerdo de licencia.
8. Haga clic en Finalizar para terminar la instalación.

Nota: si hay algún problema con el complemento (iconos que faltan, por ejemplo), pruebe a iniciar Eclipse desde la línea de comandos con el elemento flag `-clean`.

 © 2016-2022 Altova GmbH

13.2 Perspectiva MapForce

En Eclipse una perspectiva es una vista de la interfaz del usuario configurada con función de operador. Cuando el complemento de MapForce está integrado en Eclipse, se crea automáticamente una perspectiva MapForce predeterminada. Esta perspectiva es una interfaz gráfica del usuario que incluye los elementos de la interfaz de MapForce: sus vistas de edición, menús, ayudantes de entrada y otras barras laterales.

Al abrir un tipo de archivo asociado a MapForce (por ejemplo, `.mfd`), el archivo se puede editar en la perspectiva MapForce. También puede abrir archivos de otro tipo en otra perspectiva de Eclipse. Además, puede cambiar de perspectiva en todo momento, lo cual permite editar o procesar el archivo en otro entorno. Por tanto, las perspectivas tienen dos ventajas:

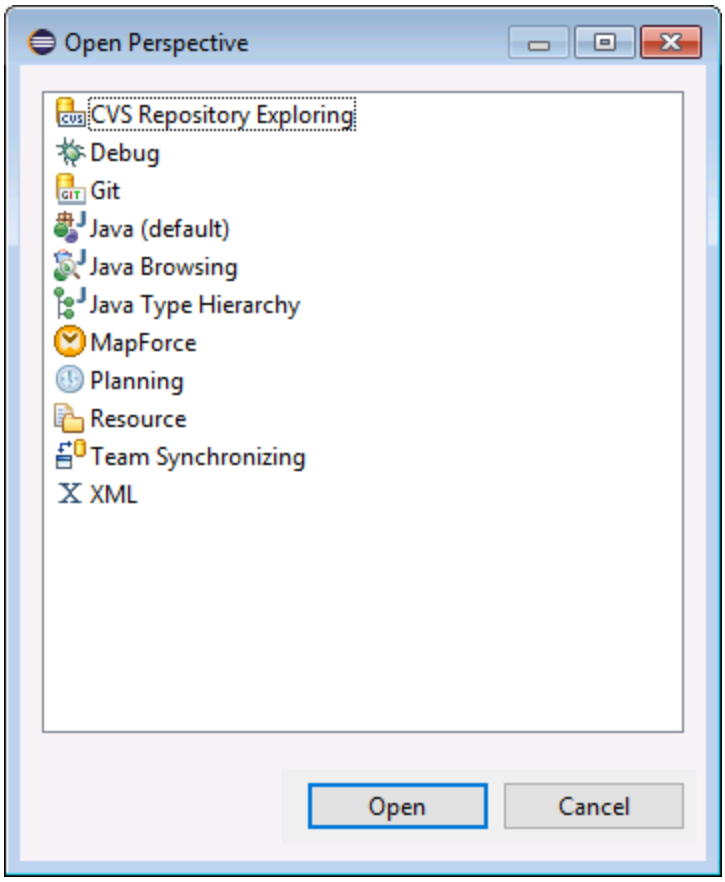
1. Permiten cambiar el entorno de trabajo para el archivo activo
2. Y permiten cambiar de un archivo a otro sin necesidad de abrir un entorno de desarrollo nuevo (el entorno asociado está disponible en la perspectiva)

Para trabajar con la perspectiva MapForce es necesario:

- Cambiar a la perspectiva MapForce.
- Configurar las preferencias de la perspectiva MapForce.
- Personalizar la perspectiva MapForce.

Cambiar a la perspectiva MapForce

En Eclipse seleccione el comando **Window | Perspective | Open Perspective | Other**. En el diálogo que aparece (*imagen siguiente*) seleccione **MapForce** y haga clic en **OK**.



La ventana vacía o el documento activo tendrán ahora la perspectiva MapForce. Así es como se cambia de perspectiva desde el menú. Para cambiar más rápido de una perspectiva a otra se puede incluir la perspectiva deseada en el submenú **Open Perspective**, sobre la opción **Other**. Esto se hace en el cuadro de diálogo de personalización.

También puede cambiar de perspectiva al abrir un archivo o activar un archivo. La perspectiva asociada al tipo de archivo se abre automáticamente al abrir el archivo por primera vez. Antes de que cambie la perspectiva, aparece un cuadro de diálogo donde debe indicar si desea abrir automáticamente la perspectiva asociada a este tipo de archivo..

Marque la casilla *Do Not Ask Again* (No volver a preguntar) si desea asociar la perspectiva a ese tipo de archivo sin tener que confirmarlo cada vez que abra un archivo de este tipo. Después haga clic en **OK**.

Configurar las preferencias de la perspectiva MapForce

Para abrir el cuadro de diálogo de preferencias, seleccione el comando **Window | Preferences**. En la lista de perspectivas del panel de la izquierda, seleccione MapForce y después las opciones que desea usar. Para terminar haga clic en **OK**.

Las preferencias de una perspectiva incluyen:

- una opción para cambiar automáticamente de perspectiva al abrir un archivo de un tipo determinado (*ver párrafo anterior*)
- opciones para incluir o excluir algunas barras de herramientas de MapForce

 © 2016-2022 Altova GmbH

- Acceso a las opciones de MapForce.

Personalizar la perspectiva MapForce

En Eclipse puede configurar qué accesos directos y comandos aparecen en cada perspectiva. Para abrir el cuadro de diálogo para personalizar la perspectiva (*imagen siguiente*), active la perspectiva que desea personalizar (en este caso la perspectiva MapForce) y seleccione el comando **Window | Perspective | Customize Perspective**.

- En las pestañas Tool Bar Visibility y Menu Visibility puede indicar qué barras de herramientas y menús quiere que se vean.
- En la pestaña Action Set Availability puede añadir conjuntos de acciones a sus menús principales y a la barra de herramientas. Si quiere habilitar un grupo de acciones marque la casilla correspondiente.
- En la pestaña Shortcuts del cuadro de diálogo "Customize Perspective" puede definir accesos directos para los submenús. Seleccione el submenú en el cuadro combinado Submenus, después seleccione una categoría de accesos directos y por último seleccione los accesos directos que desea incluir en la perspectiva.

Haga clic en **Apply & Close** para terminar de personalizar la perspectiva y que los cambios surtan efecto.

13.3 Acceso a menús y funciones frecuentes

En Eclipse puede acceder a la mayoría de funciones de MapForce desde los mismos menús que ofrece la versión independiente, salvo por algunos menús y comandos, que son distintos en el complemento para Eclipse (ver más abajo). Sin embargo, recuerde que puede personalizar la perspectiva MapForce desde Eclipse (véase Perspectiva MapForce 1005).

Nota: en Eclipse algunos grupos de menú o comandos de MapForce están deshabilitados (o no están disponibles) si el contexto no es relevante. Por ejemplo, el menú **Insertar** solamente está disponible si el archivo activo es un diseño de asignación (.mfd).

Para más información sobre los menús estándar de MapForce consulte la sección Referencia de menús 1024.

Comandos generales de MapForce

En la versión independiente de MapForce los comandos relacionados con los archivos de diseño de asignaciones (como **Validar**, **Implementar en FlowForce Server**, **Generar código**, etc.) están en el menú **Archivo**. En Eclipse, sin embargo, estos comandos están en el menú **MapForce** y en la barra de herramientas MapForce. Recuerde que los comandos para abrir y guardar archivos (incluidos los archivos de proyecto de MapForce) están en el menú **File** (Archivo) de Eclipse.

Barra de herramientas MapForce de Eclipse

El botón de la barra de herramientas abre el archivo de ayuda de MapForce.

El botón de la barra de herramientas incluye comandos relacionados con archivos de MapForce Si expande este botón, los comandos de la lista dependerán del tipo de archivo que esté activo en el editor de Eclipse. Por ejemplo, los comandos relacionados con los archivos de diseño de asignaciones (.mfd) solo están disponibles cuando dicho tipo de archivo está activo en el editor de Eclipse.

Recursos globales

Hay dos maneras de acceder a los recursos globales y gestionarlos:

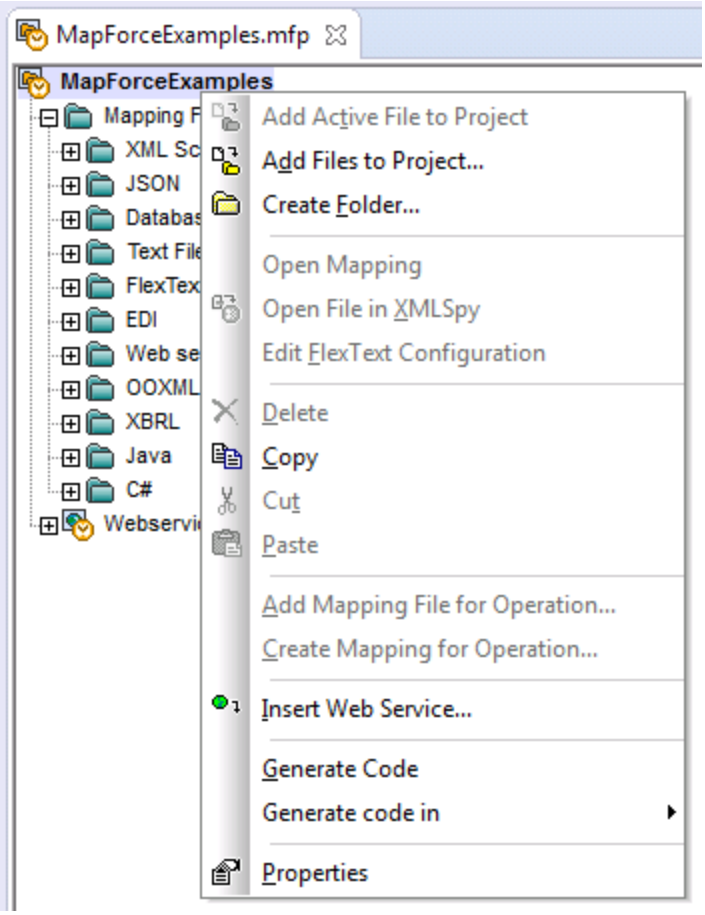
- Haga clic en el botón de la barra de herramientas y elija **Recursos globales**.
- En el menú MapForce haga clic en **Recursos globales**.

Proyectos de MapForce

En la versión independiente de MapForce el menú **Proyecto** incluye varios comandos para trabajar con archivos de proyecto de asignación (.mfp). En Eclipse estos comandos están disponibles en otras posiciones:

- Los comandos para abrir o guardar proyectos están en el menú **File** (Archivo) de Eclipse.
- El resto de comandos relacionados con los proyectos están disponibles en menús contextuales. Para ver estos comandos primero debe crear o abrir un proyecto de MapForce (.mfp) en Eclipse y después hacer clic con el botón derecho en el proyecto.

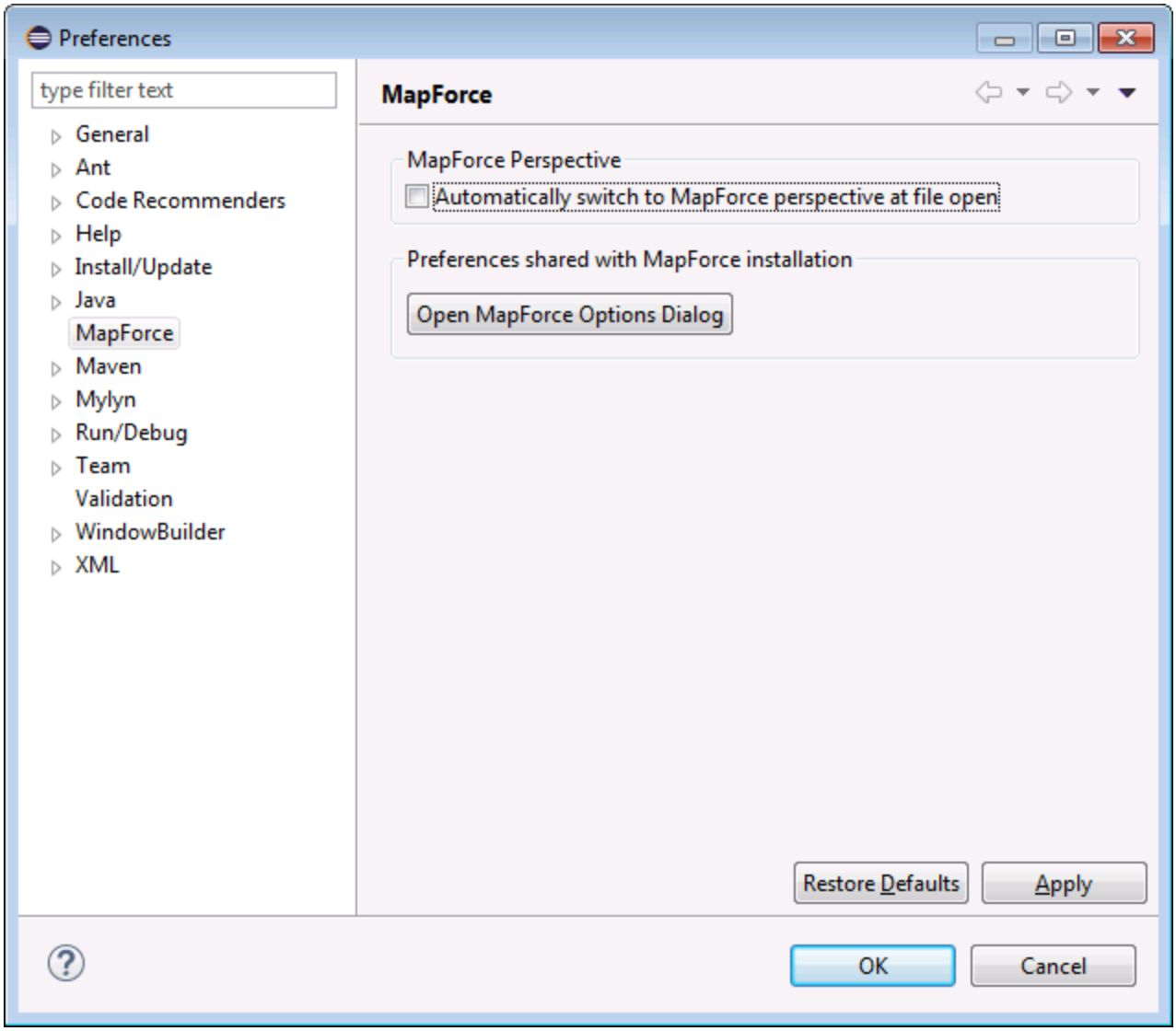
 © 2016-2022 Altova GmbH



Tenga en cuenta que, además de proyectos estándar de MapForce (.mfp), en Eclipse también puede crear proyectos de tipo MapForce/Eclipse. Este tipo de proyectos son de carácter mixto y pueden configurarse para la generación automática de código de MapForce. Para más información consulte el apartado Trabajar con asignaciones y proyectos 1012.

Opciones de MapForce

Las opciones de MapForce están disponibles en el menú **Window | Preferences** de Eclipse. En el cuadro de diálogo de preferencias ("Preferences") seleccione **MapForce** y después haga clic en **Abrir cuadro de diálogo Opciones de MapForce**.



Cuadro de diálogo de preferencias de Eclipse

Ventana Bibliotecas

En Eclipse la ventana Bibliotecas de MapForce está disponible en forma de vista y está situada por defecto a la izquierda de la ventana principal del editor. Todas las vistas de MapForce están visibles en la interfaz de Eclipse cuando está activa la perspectiva MapForce[1005].

Versión del complemento de MapForce

Para comprobar qué versión tiene instalada del complemento de MapForce para Eclipse, en Eclipse seleccione la opción de menú **Help | About Eclipse**. Después seleccione el icono de MapForce.

 © 2016-2022 Altova GmbH

Ayuda y soporte técnico

Los menús Ayuda, Soporte técnico, Buscar actualizaciones y Acerca de están en el menú **Help | Ayuda de MapForce** de Eclipse.

13.4 Trabajar con asignaciones y proyectos

Cuando está instalado el complemento de MapForce para Eclipse, podrá crear asignaciones y proyectos de asignación de datos desde un proyecto de Eclipse igual que en la versión independiente de MapForce. Para diseñar, probar, compilar e implementar asignaciones y para generar código de asignación de datos puede crear un proyecto de Eclipse nuevo o usar uno que ya exista (p. ej. un proyecto de Java al que desee añadir asignaciones de MapForce).

Además, podrá trabajar con todas sus asignaciones de datos dentro de un tipo de proyecto especial que está disponible cuando está instalado el complemento de MapForce: el proyecto MapForce/Eclipse. A no ser que prefiera personalizarlo, el proyecto MapForce/Eclipse se asigna por defecto a un generador Java Builder y a un generador de código MapForce. Además tiene dos naturalezas Eclipse: la naturaleza MapForce y la naturaleza JDT (Java Development Tools). Por tanto, se puede decir que, cuando se guardan o modifican sus recursos (p. ej. un archivo de asignación de datos), el proyecto MapForce/Eclipse se comporta de la siguiente manera:

- Si está habilitada la opción **Project > Build automatically**, el código de asignación de datos se genera automáticamente. Cuando en el proyecto MapForce/Eclipse existen archivos de proyecto de MapForce, el lenguaje de generación de código y las carpetas de destino de los resultados vienen determinados por la configuración de cada archivo de proyecto. De lo contrario, Eclipse solicita al usuario que elija una ubicación.
- Los errores y mensajes aparecen en las vistas Mensajes y Problemas.

Esta sección incluye varios apartados:

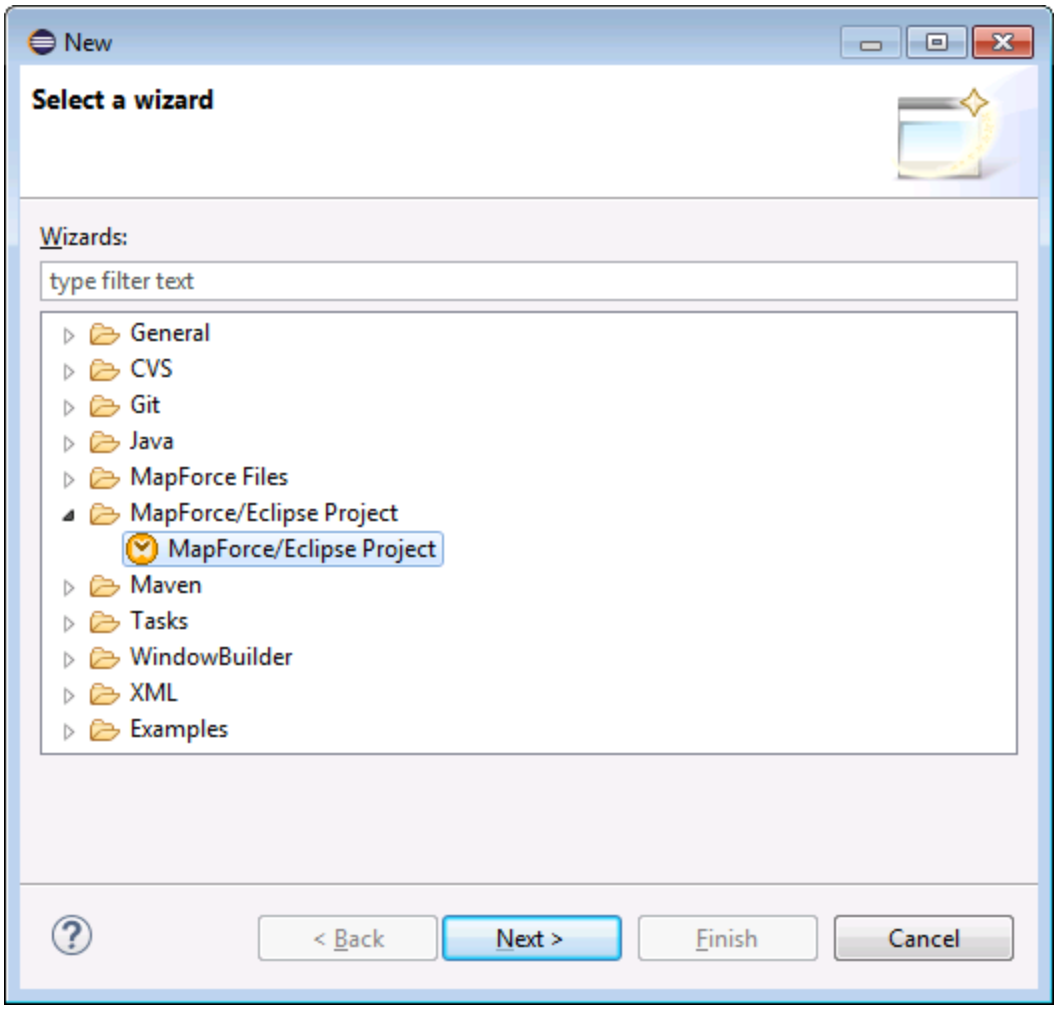
- Crear un proyecto MapForce/Eclipse 1012
- Crear asignaciones nuevas 1014
- Importar asignaciones a un proyecto de Eclipse 1016
- Configurar la generación automática de código MapForce 1019

13.4.1 Crear un proyecto MapForce/Eclipse

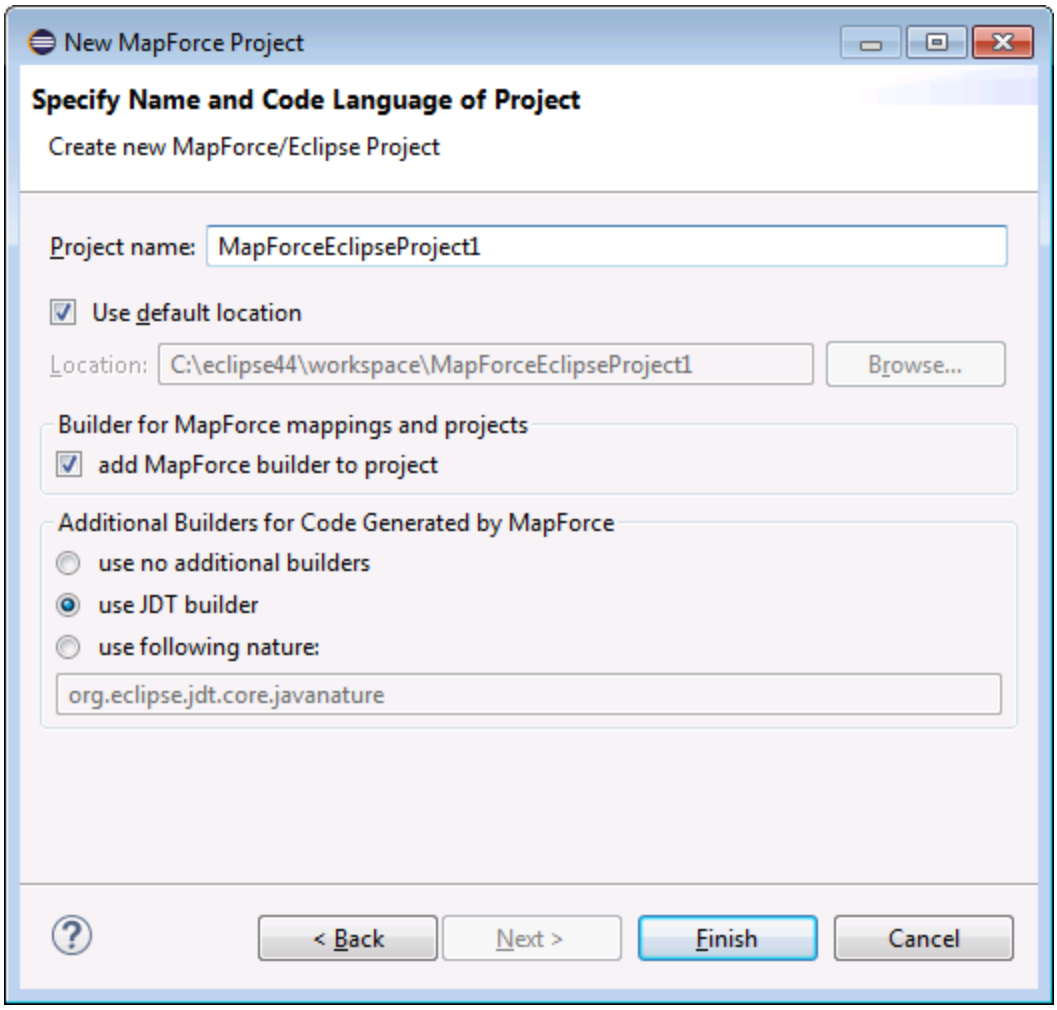
Para crear un proyecto MapForce/Eclipse:

1. En el menú **File** haga clic en **New | Other**.
2. Seleccione la categoría **MapForce/Eclipse Project**.

© 2016-2022 Altova GmbH



3. Haga clic en **Next** para continuar.



4. Introduzca el nombre del proyecto y elija la ubicación donde se debe guardar. Deje las opciones *add MapForce builder to project* y *use JDT builder* como están.
5. Haga clic en **Finish** para finalizar.

13.4.2 Crear asignaciones nuevas

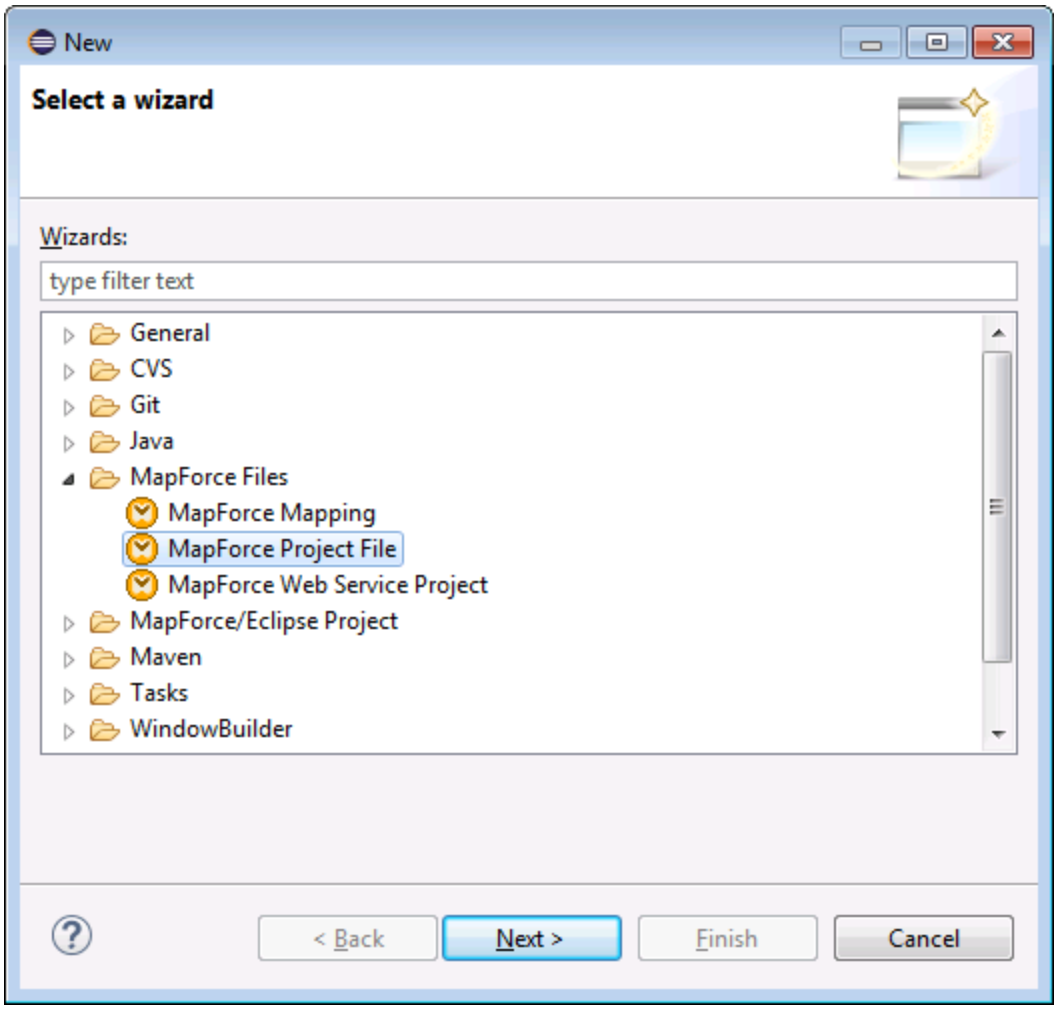
Dentro de un proyecto de Eclipse puede crear estos tipos de archivos de MapForce:

- Asignaciones de datos de MapForce
- Archivos de proyecto de MapForce
- Proyectos de servicio web de MapForce (solo en la edición MapForce Enterprise)

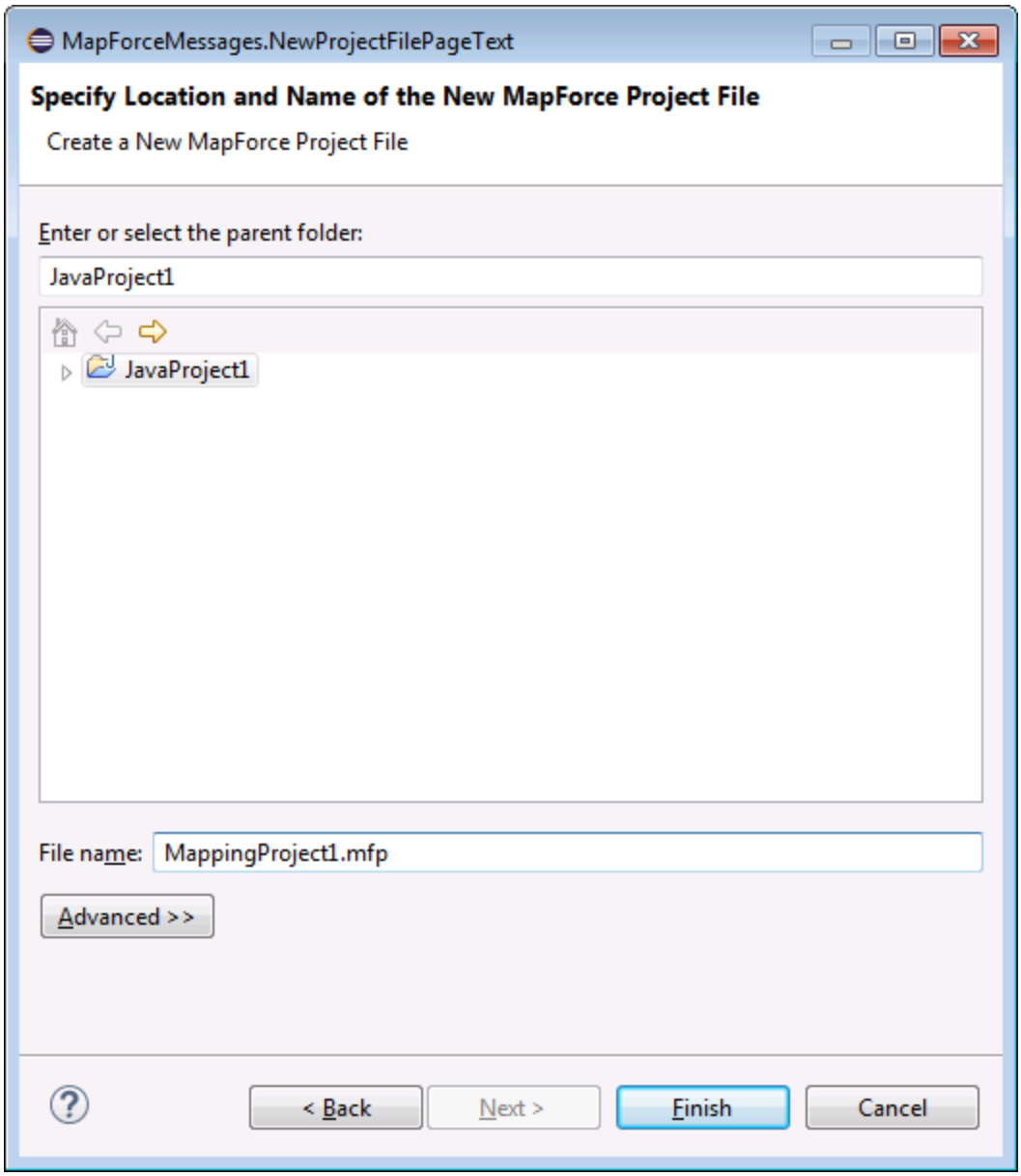
Para crear este tipo de archivos dentro de un proyecto de Eclipse:

1. Cree un proyecto de Eclipse nuevo o abra un proyecto actual.
2. En el menú **File** haga clic en **New | Other**.

 © 2016-2022 Altova GmbH



3. Seleccione el tipo de archivo que desea crear en el cuadro de diálogo y haga clic en **Next** para continuar.



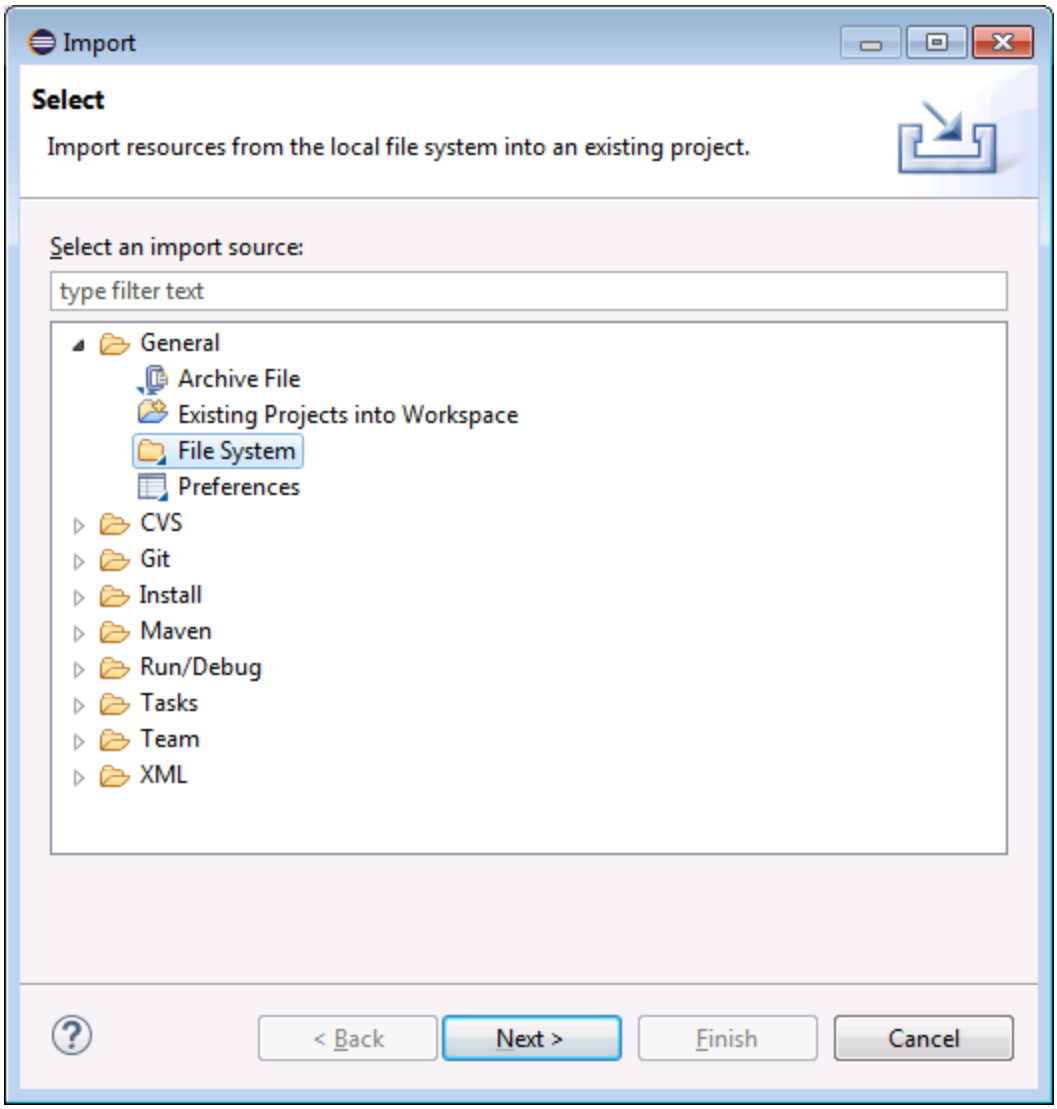
4. Seleccione una carpeta raíz del proyecto actual y haga clic en **Finish** para finalizar.

13.4.3 Importar asignaciones a un proyecto de Eclipse

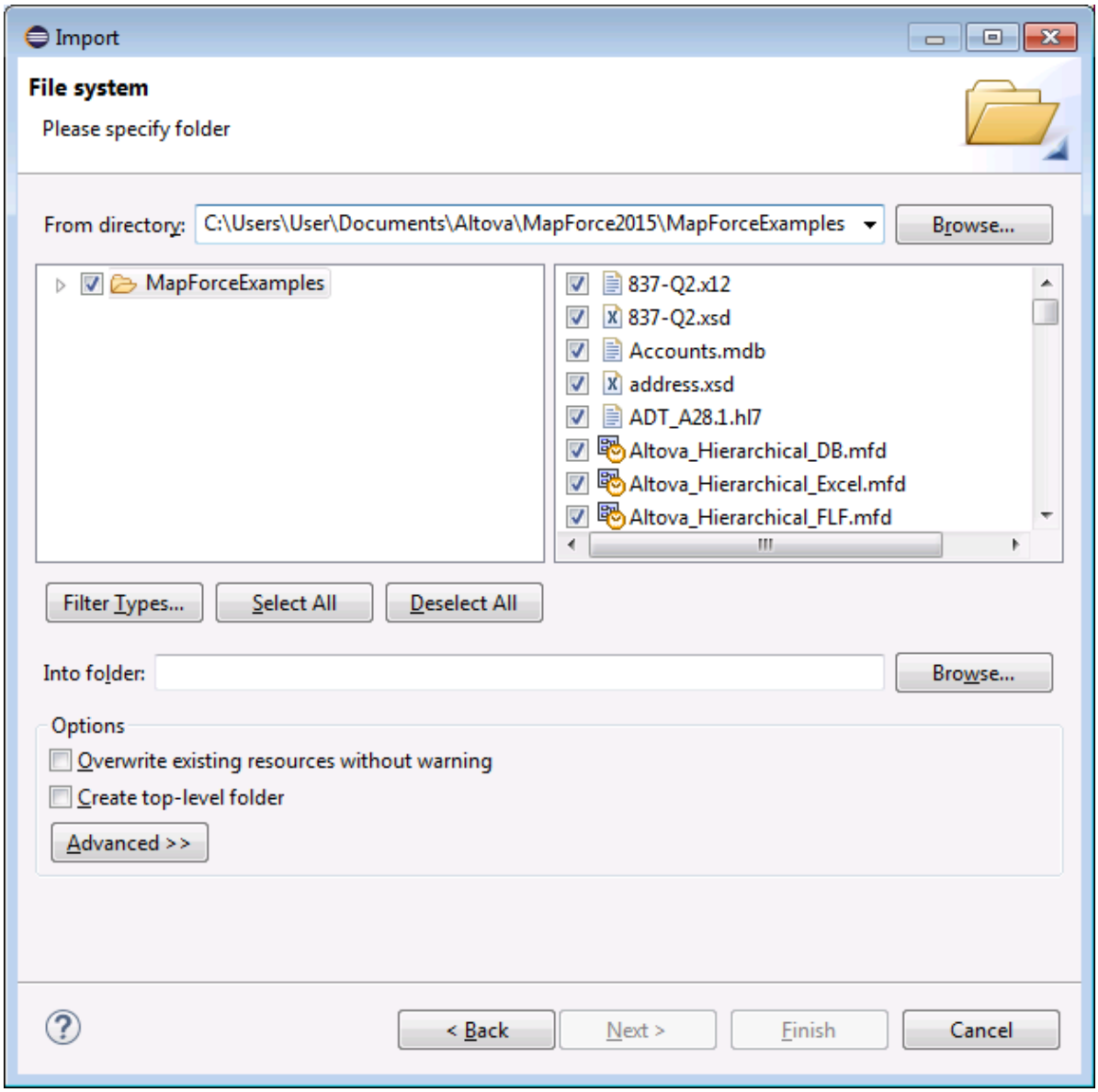
Para importar asignaciones de datos de MapForce y sus archivos dependientes en un proyecto de Eclipse:

1. Abra el proyecto en el que desea importar archivos.
2. En el menú **File** haga clic en **Import**.

 © 2016-2022 Altova GmbH

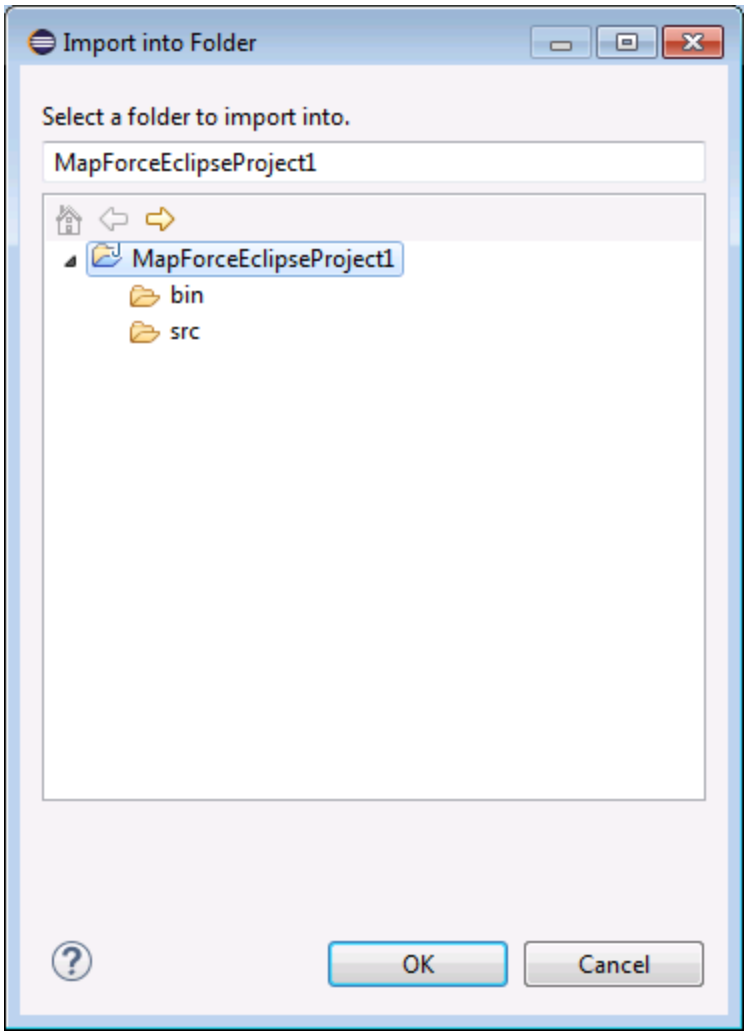


3. Seleccione **File System** y después haga clic en **Next** para continuar.



4. En el campo **From directory** navegue hasta la ubicación de los archivos que desea importar y seleccione los archivos necesarios.
5. En el campo **Into folder** haga clic en el botón **Browse** y seleccione el proyecto donde desea agregar los archivos (p. ej. *MapForceEclipseProyecto1*).

 © 2016-2022 Altova GmbH



6. Haga clic en **OK** para aceptar y después en **Finish** para finalizar.

13.4.4 Configurar la generación automática de código MapForce

La generación automática de código de MapForce está habilitada por defecto en todos los proyectos MapForce/Eclipse (*véase* *Crear un proyecto MapForce/Eclipse* 1012). Si desea habilitar la generación automática de código de MapForce en un proyecto actual que no sea de tipo *MapForce/Eclipse*, deberá hacerlo de forma manual, añadiendo el proyecto al generador de código *Generación de código de MapForce MapForce* y a la naturaleza *MapForce*.

Para agregar el generador *Generación de código de MapForce* a un proyecto:

- Añada al archivo **.project** de Eclipse las líneas de código que aparecen resaltadas en amarillo a continuación:

```
<buildSpec>
    <buildCommand>
       <name>org.eclipse.jdt.core.javabuilder</name>
       <arguments>
       </arguments>
    </buildCommand>
    <buildCommand>
       <name>com.altova.mapforceeclipseplugin.MapForceBuilder</name>
       <arguments>
       </arguments>
    </buildCommand>
</buildSpec>
```

Para agregar la naturaleza *MapForce* a un proyecto:

- Añada al archivo **.project** de Eclipse las líneas de código que aparecen resaltadas en amarillo a continuación:

```
<natures>
   <nature>org.eclipse.jdt.core.javanature</nature>
   <nature>com.altova.mapforceeclipseplugin.MapForceNature</nature>
</natures>
```

Consejo: puede abrir el archivo **.project** rápidamente desde la vista de navegación *Navigator* de Eclipse (para habilitar esta vista haga clic en el comando de menú **Window | Show View | Navigator**).

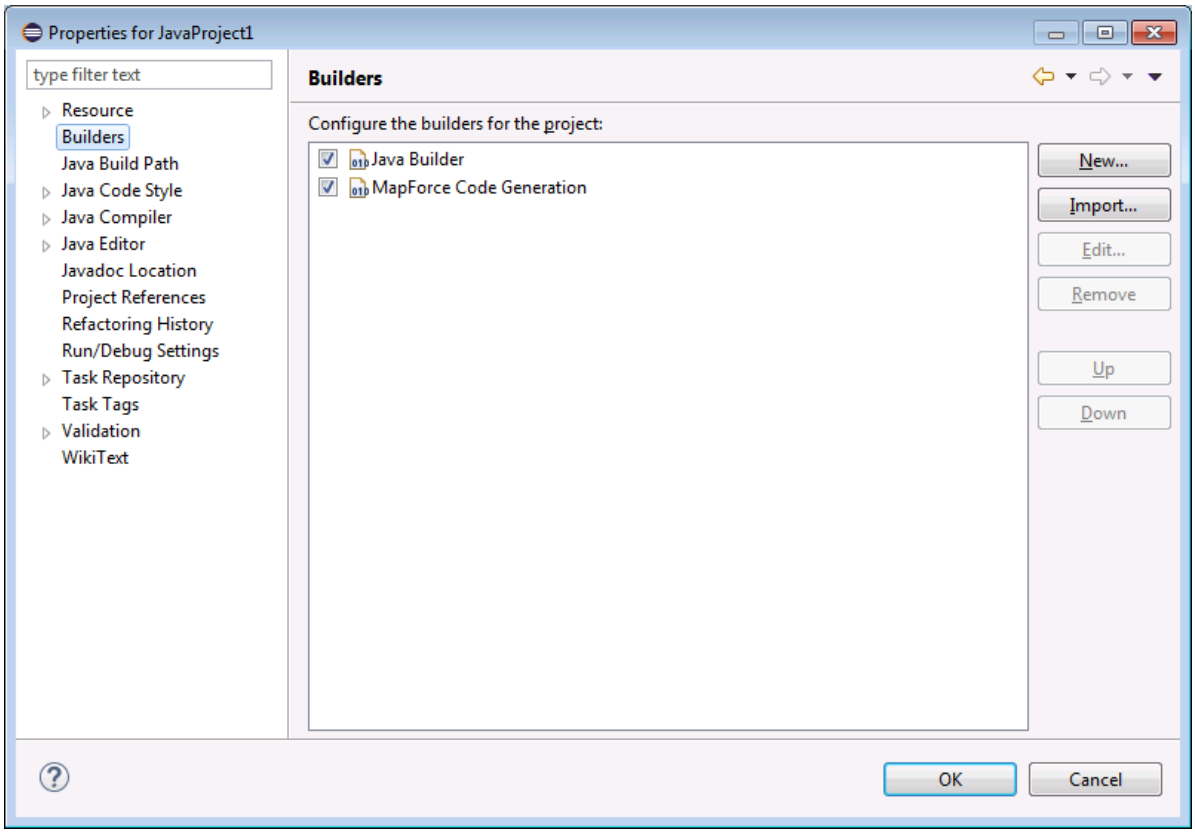
Para activar / desactivar la generación automática de código de MapForce:

- En el menú **Project** haga clic en el comando **Build automatically**.

Para deshabilitar el generador de código de MapForce:

1. En el menú **Project** haga clic en **Properties**.
2. Después haga clic en **Builders**.

© 2016-2022 Altova GmbH



3. Desactive la casilla *Generación de código de MapForce*.

13.5 Ampliar el complemento de MapForce para Eclipse

El complemento de MapForce para Eclipse ofrece un punto de extensión Eclipse cuyo identificador es **com.altova.mapforceeclipseplugin.MapForceAPI**. Puede usar este punto de extensión para adaptar o ampliar las funciones del complemento de MapForce. El punto de extensión ofrece acceso a la interfaz de COM del control MapForce[1419] y de la API de MapForce[1225].

El paquete de instalación de MapForce para Eclipse contiene un ejemplo de complemento que utiliza este punto de extensión y busca eventos de archivo abierto en todas las asignaciones nuevas de MapForce. Además establece el nivel de zoom de la vista en un 70%.

La documentación de JavaDoc para el punto de extensión está en el directorio de instalación del complemento de MapForce (que suele ser **C:\Archivos de programa\Altova\MapForce2023\eclipse\docs**).

Antes de instalar y ejecutar el complemento de MapForce de ejemplo, compruebe que cumple con estos requisitos:

- Utiliza Java (32 bits), Eclipse (32 bits), MapForce (32 bits) y el paquete de integración de MapForce (32 bits).
- Tiene instalado el complemento JDT (Java Development Tools).
- Tiene instalado el entorno de desarrollo de complementos PDE de Eclipse.

Para importar el proyecto de complemento MapForce de ejemplo al espacio de trabajo:

1. Inicie Eclipse.
2. En el menú **File** haga clic en **Import**.
3. Seleccione la opción **General | Existing projects into Workspace** y haga clic en **Next** para continuar.
4. Haga clic en el botón **Browse...** situado junto al campo de selección del directorio raíz (*Select root directory*) y elija el directorio del proyecto de ejemplo (p. ej. **C:\Archivos de programa\Altova\MapForce2023\eclipse\workspace\MapForceExtension**).
5. Ahora marque la casilla *Copy projects into workspace* (copiar proyectos en el espacio de trabajo) y haga clic en **Finish** para finalizar. En el espacio de trabajo se crea un proyecto nuevo llamado "MapForceExtension".

Para ejecutar el complemento de extensión de ejemplo:

1. Cambie a la perspectiva Java.
2. En el menú de ejecución **Run** haga clic en el comando **Run Configurations**.
3. Haga clic con el botón derecho en **Eclipse Application** y seleccione **New**. (Si no puede ver la opción **Eclipse Application** en la lista, significa que el entorno de desarrollo de complementos PDE de Eclipse no está instalado. Para instalar PDE haga clic en el comando **Install New Software** del menú de ayuda **Help** e instale **Eclipse Plugin Development Tools** desde el sitio de descargas "*The Eclipse Project Updates*".)
4. Introduzca un nombre nuevo para la nueva configuración (p. ej. *SampleMapForcePlugin*) y haga clic en **Apply** para aplicar la configuración.
5. Compruebe que el complemento del espacio de trabajo **MapForceClient** está seleccionado en la pestaña de complementos *Plug-ins*.
6. Haga clic en el botón **Run** para comenzar la ejecución. Esto abre un área de trabajo nuevo en Eclipse.

 © 2016-2022 Altova GmbH

7. Abra cualquier asignación de MapForce en el nuevo área de trabajo y observe que se abre con un nivel de zoom del 70%.

14 Comandos de menú

Esta sección describe los comandos de menú de MapForce, que son:

- Archivo 1025
- Edición 1028
- Insertar 1029
- Proyecto 1032
- Componente 1034
- Conexión 1036
- Función 1037
- Resultados 1038
- Depuración 1040
- Vista 1042
- Herramientas 1044
- Ventanas 1061
- Ayuda 1062

© 2016-2022 Altova GmbH

14.1 Archivo

En este apartado puede consultar todos los comandos del menú **Archivo**.

- Nuevo

  Crea un documento de asignación nuevo. En las ediciones Professional y Enterprise también puede crear un proyecto de asignación de datos (`.mfp`). Para más detalles consulte Proyectos 118.

- Abrir

  Abre archivos de asignación (`.mfd`). En las ediciones Professional y Enterprise también abre proyectos de asignación de datos (`.mfp`). más detalles consulte Proyectos 118.

- Guardar/Guardar como/Guardar todos

  La opción **Guardar** guarda la asignación activa usando el nombre de archivo activo en ese momento. La opción **Guardar como** guarda la asignación activa con otro nombre o permite al usuario indicar un nombre nuevo si es la primera vez que se guarda el archivo. La opción **Guardar todos** guarda todos los archivos que están abiertos.

- Volver a cargar

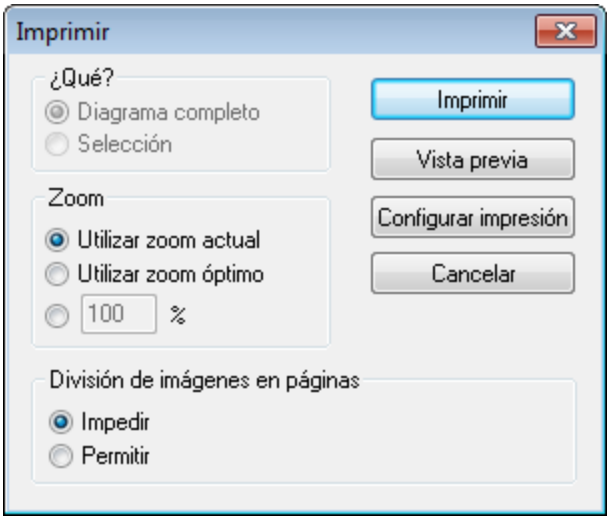
  Vuelve a cargar el archivo de asignación activo. Aparece un aviso preguntando si desea descartar los cambios realizados o guardarlos.

- Cerrar/Cerrar todos

  La opción **Cerrar** cierra el archivo de asignación activo. La opción **Cerrar todos** cierra todos los archivos de asignación activos. Aparece un aviso preguntando si desea descartar los cambios realizados o guardarlos.

- Imprimir/Vista previa de impresión/Configurar impresión

  La opción **Imprimir** abre el cuadro de diálogo Imprimir, desde donde puede imprimir la asignación (*imagen siguiente*). La opción **Utilizar zoom actual** conserva el factor de zoom actual La opción **Utilizar zoom óptimo** ajusta el tamaño de la asignación al tamaño de la página. También puede especificar el factor de zoom en porcentaje. Las barras de desplazamiento de los componentes de la asignación no se imprimen. También puede permitir o impedir la división de las imágenes en varias páginas.



El comando **Vista previa de impresión** abre el cuadro de diálogo **Imprimir** con las mismas opciones que se describen más arriba. **Configurar impresión** abre el cuadro de diálogo Configurar **impresión**, donde puede seleccionar la impresora que desea usar y definir la configuración del papel.

- Validar asignación

  Comprueba si todas las asignaciones (conectores) son válidas y emite advertencias y errores de validación (véase Validar asignaciones [105]).

- Configurar asignación

  Abre el cuadro de diálogo donde puede definir las opciones de configuración del documento activo (véase Cambiar configuración de la asignación [115]).

- Abrir el gestor de credenciales (*Enterprise Edition*)

  Abre el **Gestor de credenciales**, donde puede gestionar las credenciales que necesite para las asignaciones que funcionan con autenticación HTTP básica o autorización OAuth 2.0.

- Generar código en el lenguaje seleccionado/Generar código en

  Genera código en el lenguaje seleccionado para la asignación activa. Para más información sobre los lenguajes de transformación disponibles consulte Lenguajes de transformación [24].

  Generar código en | XSLT (XSLT2, XSLT3)
  Genera los archivos XSLT necesarios para la transformación de datos a partir de los archivos de origen. Al hacer clic en este comando se abre el cuadro de diálogo **Buscar carpeta**, donde puede seleccionar la ubicación de los archivos XSLT. El nombre de los archivos XSLT que se generan se define en el cuadro de diálogo Configurar asignación [115].

  Generar código en | XQuery | Java | C# | C++ (ediciones Professional and Enterprise)
  Este comando genera código fuente en uno de los lenguajes de transformación seleccionados. Al hacer clic en este comando se abre el cuadro de diálogo **Buscar carpeta**, donde puede seleccionar la ubicación de los archivos generados. El nombre de los archivos de aplicación que se generan (así como

 © 2016-2022 Altova GmbH

el del archivo de proyecto C#: `*.csproj`, del archivo de solución `*.sln` o del archivo de proyecto C++ `*.vcproj`) se define el cuadro de diálogo "Configurar asignación" (véase Cambiar configuración de la asignación 115).

El nombre de archivo creado por el código ejecutado es el que aparece en el campo **Archivo XML de salida** del cuadro de diálogo Configuración del componente 1035 si el destino es un archivo o esquema XML.

- Compilar en archivo de ejecución de MapForce Server (*ediciones Professional y Enterprise*)

  Genera un archivo que se puede ejecutar directamente en MapForce Server para ejecutar la transformación de la asignación de datos. Para más detalles consulte Compilar una asignación de MapForce 955.

- Implementar en FlowForce Server (*Professional and Enterprise editions*)

  Implementa la asignación activa en FlowForce Server. Para más detalles véase Implementar una asignación de MapForce 958.

- Generar documentación (*ediciones Professional y Enterprise*)

  Genera documentación para los proyectos de asignación en varios formatos de salida. Para más información consulte Generar y personalizar la documentación de la asignación 915).

- Archivos recientes

  Una lista con archivos abiertos recientemente.

- Salir

  Cierra la aplicación. Antes aparece un aviso preguntando si desea guardar los archivos que estén sin guardar.

14.2 Edición

En este apartado puede consultar todos los comandos del menú **Edición**. La mayoría de los comandos del menú **Edición** se habilitan cuando se abre el panel **Resultados** o cuando se consulta la vista previa de código XSLT en el panel **XSLT**.

- Deshacer

  MapForce ofrece un número ilimitado de operaciones deshacer, que puede usar para retroceder en todos los cambios realizados. También puede usar el icono .

- Rehacer

  Este comando permite rehacer cambios anulados anteriormente. Puede avanzar y retroceder por el historial de cambios usando estos dos comandos. También puede usar el icono .

- Buscar

  Permite buscar una cadena de texto en los paneles **XSLT**, **XSLT2**, **XSLT3**, **XQuery** (ediciones *Professional y Enterprise*) y **Resultados**. También puede usar el icono .

- Buscar siguiente

  Busca la siguiente instancia del término de búsqueda. También puede usar el icono .

- Buscar anterior

  Busca la instancia anterior del término de búsqueda. También puede usar el icono .

- Cortar / Copiar / Pegar / Eliminar

  Estos comandos de edición estándar de Windows permiten cortar, copiar, etc. componentes y funciones de la ventana de asignación.

- Seleccionar todo

  Selecciona todos los componentes del panel **Asignación** o todo el texto de los paneles **XSLT**, **XSLT2**, **XSLT3**, **XQuery** (ediciones *Professional y Enterprise*) y **Resultados**.

© 2016-2022 Altova GmbH

14.3 Insertar

En este apartado puede consultar todos los comandos del menú **Insertar**.

- Archivo o esquema XML

  Inserta un archivo de esquema XML o de instancia XML a la asignación. Si selecciona un archivo XML sin referencia a un esquema, deberá generar automáticamente el esquema XML correspondiente 124. Si selecciona un archivo de esquema XML, aparece un aviso preguntando si desea incluir un archivo XML de instancia que aporte los datos para la vista previa. También puede agregar un archivo XML/XSD con el comando de la barra de herramientas .

- Base de datos (*ediciones Professional y Enterprise*)

  Inserta una base de datos como componente de origen o destino. Consulte Bases de datos 159. También puede agregar un componente de BD con el comando de la barra de herramientas . En MapForce Enterprise Edition también puede agregar BDs NoSQL como componentes.

- EDI (*Enterprise Edition*)

  Inserta un documento EDI como componente de origen o destino. También puede agregar un componente EDI con el comando de la barra de herramientas .

- Archivo de texto (*Professional and Enterprise editions*)

  Inserta un archivo plano (es decir, un archivo de texto CSV o de longitud fija) en la asignación de datos. Ambos tipos de archivo pueden usarse como componente de origen o de destino. Para más detalles consulte Archivos CSV y archivos de texto 445. También puede agregar un archivo de texto con el comando de la barra de herramientas . MapForce Enterprise Edition también permite procesar archivos de texto con FlexText.

- Función de servicio web (*Enterprise Edition*)

  Inserta en la asignación una llamada a un servicio web. También puede agregar un servicio web con el comando de la barra de herramientas .

- Archivo Excel 2007+ (*Enterprise Edition*)

  Inserta un archivo Microsoft Excel 2007+ (Office Open XML) (`.xlsx`) como componente de origen o destino. Si no tiene Excel 2007+ instalado, esto no le impide crear asignaciones de datos entre archivos Excel 2007+. El único impedimento es que no podrá consultar la vista previa de resultados en el panel **Resultados**. Sí podrá guardar los resultados con el comando **Guardar el archivo de salida** del menú **Resultados**. También puede agregar un archivo de Excel con el comando de la barra de herramientas .

- Documento XBRL (*Enterprise Edition*)

  Inserta un documento de instancia XBRL o una taxonomía XBRL en la asignación. También puede agregar

un componente XBRL con el comando de la barra de herramientas .

- Archivo o esquema JSON (*Enterprise Edition*)

  Inserta un archivo o un esquema JSON en la asignación. También puede agregar un componente JSON con el comando de la barra de herramientas .

- Archivo de Protocol Buffers (*Enterprise Edition*)

  Añade a la asignación un archivo binario cifrado en formato Protocol Buffers. También puede agregar un componente Protocol Buffers con el comando de la barra de herramientas .

- Insertar componente de entrada

  Si en el área de asignación está visible una asignación de datos, este comando inserta un componente de entrada. Si en el área de asignación está visible una función definida por el usuario, este comando agrega un componente de entrada en la función. Para más información consulte Pasar parámetros a la asignación 477 y Parámetros en funciones definidas por el usuario 581. También puede agregar un componente de entrada simple con el comando de la barra de herramientas .

- Insertar componente de salida

  Si en el área de asignación está visible una asignación de datos, este comando inserta un componente de salida. Si en el área de asignación está visible una función definida por el usuario, este comando agrega un componente de salida en la función. Para más información véase Obtener valores de cadena de una asignación 488 y Parámetros en funciones definidas por el usuario 581. También puede agregar un componente de salida simple con el comando de la barra de herramientas .

- Constante

  Inserta una constante que aporta datos fijos a un conector de entrada. Los datos se introducen en un cuadro de diálogo que aparece cuando se empieza a crear la constante. Puede elegir entre estos tipos de datos: `cadena`, `número`, `demás opciones`. También puede agregar una constante con el comando de la barra de herramientas .

- Variable

  Inserta una variable 492 intermedia que equivale a una función definida por el usuario no inline. Las variables son componentes estructurales, sin archivos de instancia, que sirven para simplificar el proceso de asignación. También puede agregar una variable con el comando de la barra de herramientas .

- Combinar (*ediciones Professional y Enterprise*)

  El componente de combinación permite combinar datos en los modos SQL y non-SQL. También puede agregar un componente de combinación con el comando de la barra de herramientas . Para más detalles consulte Combinar datos 517.

- Componente de ordenación: nodos/filas

© 2016-2022 Altova GmbH

Inserta un componente que permite ordenar nodos (véase Ordenar datos [505]). También puede agregar una variable con el comando de la barra de herramientas .

- Filtro: nodos/filas

  Inserta un componente de filtro que filtra los datos de cualquier componente compatible con MapForce, incluidas BDs. Consulte la sección Filtros y condiciones [543] para obtener más información. También puede agregar un filtro con el comando de la barra de herramientas .

- WHERE/ORDER de SQL/NoSQL (*ediciones Professional y Enterprise*)

  Inserta un componente que filtra datos de la base de datos de forma condicional. Para más información véase Componente WHERE/ORDER de SQL [352]. También puede acceder al componente WHERE/ORDER de SQL/NoSQL con el comando de la barra de herramientas .

- Asignación de valores

  Inserta un componente que transforma un valor de entrada en un valor de salida usando una tabla de búsqueda. Es un componente muy práctico si necesita asignar un conjunto de valores a otro conjunto de valores (p. ej. números de meses en nombres de meses). Para más información consulte Usar asignaciones de valores [533]. También puede agregar este componente con el comando de la barra de herramientas .

- Condición If-Else

  Inserta un componente de tipo `If-Else` para filtrar valores simples de forma condicional. Consulte la sección Filtros y condiciones [543] para obtener más información. También puede agregar un componente de tipo `If-Else` con el comando de la barra de herramientas .

- Excepción (*Professional and Enterprise editions*)

  Este componente permite interrumpir el proceso de asignación cuando se cumple determinada condición. También puede agregar un componente de excepción con el comando de la barra de herramientas . En Enterprise Edition este componente también permite definir mensajes de error (fault) en proyectos de asignación WSDL. Consulte Agregar excepciones [529] para obtener más información sobre excepciones.

14.4 Proyecto

MapForce integra la interfaz de múltiples documentos y permite al usuario agrupar las asignaciones de datos en proyectos de asignación (*véase* *Trabajar con proyectos de asignación* 118). En este apartado puede consultar todos los comandos del menú **Proyecto**.

- Volver a cargar el proyecto

  Vuelve a cargar el proyecto activo y cambia a la ventana **Proyecto**.

- Cerrar el proyecto

  Cierra el proyecto activo.

- Guardar el proyecto

  Guarda el proyecto activo.

- Agregar archivos al proyecto

  Permite agregar asignaciones al proyecto actual desde el cuadro de diálogo **Abrir**.

- Agregar archivo activo al proyecto

  Agrega el archivo activo al proyecto que está abierto.

- Crear carpeta

  Agrega una carpeta nueva a la estructura del proyecto activo. Este comando solamente está disponible cuando esta operación es posible (*véase* *Gestionar las carpetas de proyecto* 121).

- Abrir asignación (*Enterprise Edition*)

  Abre la asignación seleccionada/resaltada en la ventana **Proyecto**.

- Crear asignación para la operación (*Enterprise Edition*)

  Crea un archivo de asignación para la operación seleccionada del proyecto WSDL.

- Agregar archivo de asignación para la operación (*Enterprise Edition*)

  Allows you to add a previously saved mapping file to the currently active WSDL operation.

- Insertar servicio web (*Enterprise Edition*)

  Permite insertar un servicio web basado en un archivo WSDL ya existente..

- Abrir archivo en XMLSpy

  Abre el archivo WSDL seleccionado en Altova XMLSpy.

- Generar código para todo el proyecto

  Genera código de proyecto para todo el proyecto activo en la ventana **Proyecto**. El código se genera en

 © 2016-2022 Altova GmbH

el lenguaje seleccionado para todos los archivos de asignación **.mfd** en cada una de las carpetas.

- Generar código en

  Genera código de proyecto en el lenguaje seleccionado en el menú emergente.

- Propiedades

  Abre un cuadro de diálogo donde puede definir la configuración global del proyecto (*véase* *Configurar la generación de código* 120).

- Proyectos recientes

  Una lista con proyectos abiertos recientemente.

14.5 Componente

En este apartado puede consultar todos los comandos del menú **Componente**.

- Cambiar de elemento raíz

  Permite cambiar el elemento raíz del documento de instancia XML.

- Editar la definición del esquema en XMLSpy

  Seleccione esta opción (después de hacer clic en el documento/esquema XML) para abrir el archivo de esquema XML en la vista Esquema de Altova XMLSpy.

- Editar la configuración de FlexText (*Enterprise Edition*)

  Abre el módulo FlexText donde puede editar un archivo de FlexText ya existente.

- Agregar, quitar o editar objetos de la base de datos (*ediciones Professional y Enterprise*)

  Permite agregar, eliminar o cambiar los objetos de la base de datos dentro del componente de base de datos.. Para más detalles consulte Bases de datos 159.

- Crear asignación a EDI X12 997 (*Enterprise Edition*)

  La confirmación funcional X12 997 informa del estado del intercambio EDI. Todos los errores encontrados durante el procesamiento del documento aparecen en esta confirmación. MapForce puede generar documentos X12 997 automáticamente en el panel *Asignación*.

- Crear asignación a EDI X12 999 (*Enterprise Edition*)

  El conjunto de transacciones de la confirmación de implementación X12 999 notifica el incumplimiento de las normas de implementación HIPAA y errores de aplicación. A cada transacción EDI enviada a una organización debe contestarse enviando una transacción 999.

- Actualizar (*ediciones Professional y Enterprise*)

  Vuelve a cargar la estructura del componente de BD activo desde la base de datos.

- Agregar delante/detrás un duplicado de entrada

  Inserta una copia (un clon) del elemento seleccionado antes del elemento seleccionado. Para más información consulte Duplicar elementos de entrada 81.

- Quitar el duplicado

  Quita el elemento duplicado seleccionado

- Comentarios/procesar instrucciones

  Esta opción permite insertar comentarios e instrucciones de procesamiento 135 en componentes XML.

- Escribir contenido como secciones CDATA

 © 2016-2022 Altova GmbH

Las secciones CDATA 136 trata como unidades completas bloques de texto que normalmente se interpretarían como marcado.

- Acciones de tabla de la base de datos (*ediciones Professional y Enterprise*)

  Permite definir qué acciones de tabla se deben llevar a cabo con los datos asignados en cada tabla de la BD de destino. Consulte la sección Configurar acciones de tabla de la base de datos 327 para más información.

- Consultar la base de datos (*ediciones Professional y Enterprise*)

  Crea una instrucción SELECT basada en la tabla o en el campo activos en el componente de base de datos. Este comando se habilita al hacer clic una vez en la tabla o el campo y la instrucción SELECT se introduce automáticamente en la ventana **Select**.

- Alinear la estructura a la izquierda

  Coloca todos los elementos de un componente en el lado izquierdo de la ventana.

- Alinear la estructura a la derecha

  Coloca todos los elementos de un componente en el lado derecho de la ventana.

- Propiedades

  Abre un cuadro de diálogo con las opciones de configuración actuales del componente. Consulte Cambiar configuración de los componentes 79.

14.6 Conexión

En este apartado puede consultar todos los comandos del menú **Conexión**.

- Conectar automáticamente los secundarios equivalentes

  Activa o desactiva la opción **Conexión automática de secundarios** así como el icono de la barra de herramientas. Para más información sobre conexiones y sus tipos consulte Conexiones 87.

- Configurar la conexión de secundarios equivalentes

  Abre el cuadro de diálogo "Configurar la conexión de secundarios equivalentes" donde puede definir las opciones de conexión. Consulte Conectar elementos secundarios equivalentes 94.

- Conectar los secundarios equivalentes

  Este comando permite crear varios conectores para los elementos que tienen el mismo nombre en el esquema de origen y en el de destino. Aparece un cuadro de diálogo donde puede definir la conexión. Las opciones definidas en este cuadro de diálogo se conservan y se aplican cada vez que se conecten dos elementos siempre y cuando esté activo el icono **Conexión automática de secundarios** de la barra de herramientas. Consulte también Conectar elementos secundarios equivalentes 94.

- Basada en el destino (estándar)

  Cambia el tipo de conector a una asignación estándar. Para más información consulte Conexiones basadas en el destino 91.

- Copia total (copia los elementos secundarios)

  Crea conectores para todos los elementos secundarios equivalentes, donde cada uno de los conectores secundarios aparece como una rama del conector primario. Consulte también Conexiones de copia total 95.

- Basada en origen (contenido mixto)

  Cambia el tipo de conector a una asignación basada en origen o de contenido mixto (texto y nodos secundarios) en el mismo orden que el del archivo XML de origen. Para más información consulte Conexiones basadas en el origen 91.

- Propiedades

  Abre un cuadro de diálogo donde puede definir la configuración específica del conector actual. Consulte también Configuración de las conexiones 97.

© 2016-2022 Altova GmbH

14.7 Función

En este apartado puede consultar todos los comandos del menú **Función**.

- Crear una función definida por el usuario

  Crea una función nueva definida por el usuario 575 (UDF por sus siglas en inglés). También puede crear una función definida por el usuario con el comando de la barra de herramientas .

- Crear función definida por el usuario a partir de la selección

  Crea una función nueva definida por el usuario basada en los elementos que están seleccionados en el área de asignación. Para más detalles consulte Crear funciones definidas por el usuario 578. También puede crear una función definida por el usuario a partir de la selección con el comando de la barra de herramientas .

- Configuración de la función

  Abre el cuadro de diálogo de configuración de la función definida por el usuario que está activa para poder cambiar su configuración. Para más detalles consulte Editar funciones definidas por el usuario 580.

- Quitar función

  Elimina la función definida por el usuario que está activa (siempre y cuando el contexto lo permita).

- Insertar componente de entrada

  Si en el área de asignación está visible una asignación de datos, este comando inserta un componente de entrada (*véase Entradas simples* 478). Si en el área de asignación está visible una función definida por el usuario, este comando agrega un componente de entrada en la función (*véase Parámetros en funciones definidas por el usuario* 581). También puede insertar un componente de entrada simple con el comando de la barra de herramientas .

- Insertar componente de salida

  Si en el área de asignación está visible una asignación de datos, este comando inserta un componente de salida (*véase Salidas simples* 489). Si en el área de asignación está visible una función definida por el usuario, este comando agrega un componente de salida a la función función nueva definida por el usuario 575. También puede insertar un componente de salida simple con el comando de la barra de herramientas .

14.8 Resultados

En este apartado puede consultar todos los comandos del menú **Resultados**.

- XSLT 1.0, XSLT 2.0, XSLT 3.0, XQuery, Java, C#, C++, Motor de ejecución integrado

  Establece el lenguaje de transformación en el que se debe ejecutar la asignación de datos. Los lenguajes de transformación disponibles dependen de la edición de MapForce. Para más detalles consulte Seleccionar un lenguaje de transformación [24]. También puede seleccionar los lenguajes de transformación en la barra de herramientas.

- Validar archivo de salida

  Valida el archivo XML de salida con el esquema al que se hace referencia. Consulte Validación [105].

- Guardar el archivo de salida

  Guarda en un archivo los datos visibles en el panel **Resultados**.

- Guardar todos los archivos de salida

  Guarda todos los archivos de salida generados de las asignaciones dinámicas [892]. Consulte también el Tutorial 4 [64].

- Volver a generar archivo de salida

  Vuelve a generar los datos visibles en el panel **Resultados**

- Ejecutar script SQL/NoSQL (*Professional and Enterprise editions*)

  Si hay un script SQL/NoSQL en el panel **Resultados**, el script ejecuta la asignación en la base de datos de destino, teniendo en cuenta las acciones de tabla definidas.Para saber más sobre bases de datos consulte Bases de datos [159].

- Insertar o quitar marcador

  Inserta o quita un marcador en la posición del cursor en el panel **Resultados**.

- Marcador anterior/siguiente

  Navega hasta el marcador anterior/siguiente en el panel **Resultados**.

- Quitar todos los marcadores

  Quita todos los marcadores que están definidos en el panel **Resultados**.

- Texto XML pretty-print

  Ajusta el formato del documento XML en el panel **Output** para presentarlo de forma estructurada. A cada nodo secundario se le aplica una sangría (1 tabulación) con respecto a su primario. Para controlar el tamaño de la sangría use el cuadro de diálogo Características de la vista Texto [108] del grupo de opciones *Tabulaciones*.

- Configurar la vista Texto

 © 2016-2022 Altova GmbH

Abre el cuadro de diálogo Configurar la vista Texto, donde puede personalizar la configuración de esta vista en los paneles **XQuery** (ediciones *Professional y Enterprise*), **Resultados** y **XSLT**. También muestra las teclas de acceso rápido que se pueden usar en la ventana. Para más información consulte el apartado Características de la vista Texto 108.

14.9 Depuración

En este apartado puede consultar todos los comandos del menú **Depuración**.

- Iniciar depuración

  Inicia o continúa la depuración hasta llegar a un punto de interrupción o hasta que finaliza la asignación. También puede iniciar la depuración con el comando de la barra de herramientas . Para más información sobre el modo de depuración consulte Depuración de asignaciones 924.

- Detener depuración

  Detiene la depuración. Esto comando cierra el modo depurador y rehabilita el modo estándar de MapForce. También puede detener la depuración con el comando de la barra de herramientas . Para más información sobre el modo de depuración consulte Depuración de asignaciones 924.

- Paso a paso por instrucciones

  Ejecuta la asignación hasta que se termina un solo paso en cualquier posición de la asignación. En el depurador el término ***paso*** se usa para designar a un grupo lógico de cálculos dependientes que normalmente producen un solo elemento de una secuencia.

  Dependiendo del contexto de la asignación, este comando puede leerse como "ir a la izquierda / ir al secundario de destino / ir al primario de origen". También puede acceder a este comando con el botón de la barra de herramientas . Para más información sobre el modo de depuración consulte Depuración de asignaciones 924.

- Paso a paso por procedimientos

  Continúa con la ejecución hasta que finaliza el paso actual (o se termina otra vez para otro elemento de la secuencia) o hasta que finaliza un paso no relacionado. Este comando se salta los cálculos que son entradas del paso actual. También puede acceder a este comando con el botón de la barra de herramientas . Para más información sobre el modo de depuración consulte Depuración de asignaciones 924.

- Paso a paso para salir

  Continúa con la ejecución hasta que el resultado del paso actual se consume o hasta que se ejecuta un paso que no es una entrada ni un secundario del consumo. Este comando sale del cálculo actual. Dependiendo del contexto de la asignación, este comando puede leerse como "ir a la derecha / ir al primario de destino / ir al secundario de origen". También puede acceder a este comando con el botón de la barra de herramientas . Para más información sobre el modo de depuración consulte Depuración de asignaciones 924.

- Paso mínimo

  Continúa con la ejecución hasta que se produce o consume un valor. Este comando subdivide un paso y se detiene dos veces por cada conexión: cuando su origen produce un valor y cuando su destino lo consume. MapForce no siempre calcula los valores en el orden que sugiere la asignación, así que los eventos de producción y consumo no siempre van de la mano. También puede acceder a este comando con el botón de la barra de herramientas . Para más información sobre el modo de depuración consulte Depuración de asignaciones 924.

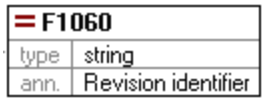
© 2016-2022 Altova GmbH

14.10 Vista

En este apartado puede consultar todos los comandos del menú **Vista**.

- Mostrar anotaciones

  Muestra las anotaciones del esquema XML en la ventana del componente. También puede activar esta opción con el botón de la barra de herramientas . Si también se activa el icono **Mostrar tipos**, el tipo y la anotación aparecen en una tabla (*imagen siguiente*). También puede usar las anotaciones para etiquetar conexiones. Para más detalle consulte Configuración de la conexión 99.



- Mostrar tipos

  Muestra los tipos de datos del esquema para cada elemento o atributo. También puede activar esta opción con el botón de la barra de herramientas . Si también se activa el icono **Mostrar anotaciones**, el tipo y la anotación aparecen en una tabla (*véase el punto anterior*).

- Mostrar biblioteca en el título de la función

  Muestra el nombre de la biblioteca en paréntesis en el título de la función. También puede activar esta opción con el botón de la barra de herramientas .

- Mostrar información rápida

  Muestra información rápida con un texto explicativo al pasar el puntero del ratón por encima de la función. Si también se activa el icono **Mostrar información rápida**, también puede ver la información sobre los tipos de datos en el componente.

- Opciones de visualización XBRL (*Enterprise Edition*)

  MapForce permite configurar estas opciones XBRL:

  - El lenguaje de etiquetado de los elementos XBRL y sus anotaciones
  - Los roles de etiquetado preferidos para nombres de elementos XBRL
  - El tipo específico de roles de etiquetado de las anotaciones de los elementos XBRL
  - Paquetes de taxonomías XBRL personales

- Mostrar los conectores del componente seleccionado/Mostrar los conectores desde su origen a su destino

  Permite elegir qué conexiones resaltar. Para saber más acerca de cómo funciona esta opción consulte Conexiones 89.

- Zoom

  Abre el cuadro de diálogo de **zoom**, donde puede indicar el factor de zoom en formato numérico o con un control deslizante.

 © 2016-2022 Altova GmbH

- Atrás/Adelante

  Retrocede o avanza en las asignaciones que están abiertas en el panel Asignación.

- Barra de estado

  Muestra/oculta la **barra de estado** que está visible bajo la ventana **Mensajes**.

- Bibliotecas/Administrar bibliotecas

  Muestra/oculta la ventana **Biblioteca**, que contiene todas las funciones. La opción **Administrar bibliotecas** activa o desactiva la ventana **Administrar bibliotecas**.

- Mensajes

  Muestra/oculta la ventana Mensajes 34. La ventana **Mensajes** se activa automáticamente cuando se genera código para mostrar el resultado de la validación.

- Vista general

  Muestra/oculta la ventana Vista general 33. Arrastre el rectángulo rojo por la ventana de vista general para navegar por el panel **Asignación**.

- Ventana Proyecto (*ediciones Professional y Enterprise*)

  Muestra/oculta la ventana **Proyecto**. Para saber más sobre proyectos consulte Proyectos 118.

- Depurar ventanas (*ediciones Professional y Enterprise*)

  El modo depurador permite analizar el contexto en que se produce un valor en concreto. Puede encontrar esta información directamente en la asignación y en los las ventanas **Valores**, **Contexto** y **Puntos de interrupción**. Para más información consulte Información sobre el modo depurador 928.

14.11 Herramientas

En este apartado puede consultar todos los comandos del menú **Herramientas**.

- Recursos globales

  Abre el cuadro de diálogo **Administrar recursos globales** donde puede agregar, editar o eliminar recursos globales en el archivo XML de recursos globales (*véase Recursos globales de Altova* 969).

- Configuración activa

  Permite seleccionar la configuración de recurso global que está activa en una lista que enumera todas las configuraciones definidas previamente como recursos globales 969.
  Allows you to select the currently active global resource configuration from a list of configurations. To create and configure different types of global resources, see recursos globales 969.

- Crear asignación invertida

  Crea una asignación *invertida* a partir de la asignación activa que será la base de una asignación nueva. Tenga en cuenta que el resultado no es una asignación completa porque en la asignación invertida solamente se conservan las conexiones directas entre los componentes de la asignación original. Es muy posible que la asignación resultante no sea válida o que no se pueda ejecutar (al abrir el panel **Resultados**) sin antes realizar ciertos cambios.

  Cuando se invierte una asignación de datos, el componente de origen pasa a ser el componente de destino y viceversa. Si se asignó un archivo de instancia XML de entrada o salida a un componente, estos archivos también se permutarán.

  Estos son los datos que se conservan:

  - conexiones directas entre los componentes
  - conexiones directas entre los componentes de una asignación en cadena
  - el tipo de conexión 91: estándar, de contenido mixto y de copia total
  - la configuración de componentes de paso a través
  - los componentes de base de datos (*ediciones Professional and Enterprise*)

  Estos son los datos que no se conservan:

  - conexiones a través de funciones, filtros, etc. así como las funciones, los filtros, etc.
  - funciones definidas por el usuario
  - componentes de servicio web (*Enterprise Edition*)

- Gestor de taxonomías XBRL (*Enterprise Edition*)

  El gestor de taxonomías XBRL permite instalar y administrar taxonomías XBRL.

- Gestor de esquemas XML

  XML Schema Manager is an Altova tool that provides a centralized way to install and manage XML schemas (DTDs for XML and XML Schemas) for use across all Altova's XBRL-enabled applications. For more information, see Schema Manager 143.

- Personalizar

© 2016-2022 Altova GmbH

Abre un cuadro de diálogo donde puede personalizar la interfaz gráfica del usuario de MapForce, incluidas las barras de herramientas que aparecen en la interfaz. También permite personalizar los menús[1045] y las *teclas de acceso rápido*[1047].

- Restaurar barras de herramientas y ventanas

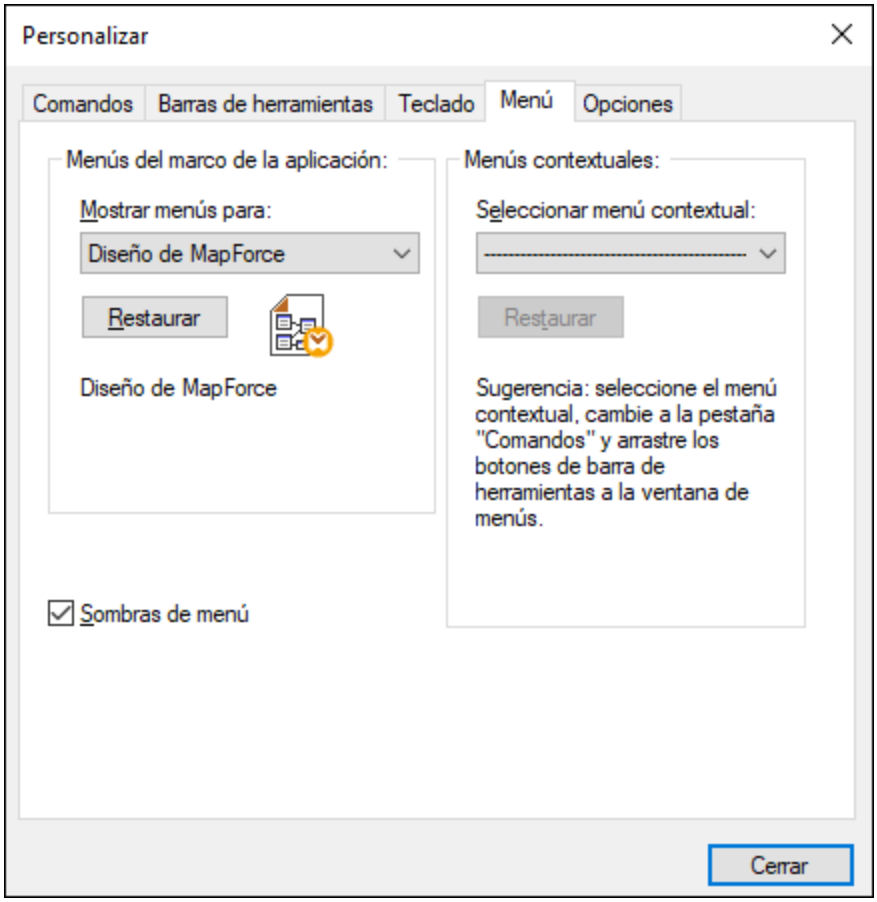
  Restaura las barras de herramientas, los ayudantes de entrada, las ventanas acopladas, etc. a su estado predeterminado. Es necesario reiniciar MapForce para que los cambios surtan efecto.

- Opciones

  Abre un cuadro de diálogo donde puede cambiar la configuración predeterminada de MapForce (*véase Cambiar opciones de MapForce*[1049]).

14.11.1 Personalizar menús

Puede personalizar los menús estándar de MapForce, así como los menús contextuales (por ejemplo, para añadir, cambiar o eliminar comandos). También puede restaurar cualquier menú personalizado a su estado predeterminado (**Restaurar**). Para personalizar menús vaya al menú **Herramientas**, haga clic en **Opciones** y después en la pestaña **Menú** (*imagen siguiente*).



Personalizar menús
barra de menú **Predeterminada** es la que aparece si no hay ningún documento abierto en la ventana principal. La barra de menú **Diseño de MapForce** es la que aparece cuando se abren una o más asignaciones. Cada una de estas barras de herramientas se puede personalizar por separado y los cambios que haga en una de ellas no afectarán a las demás.

Para personalizar una barra de herramientas, selecciónela en la lista desplegable **Mostrar menús para**. Ahora cambie a la pestaña *Comandos* y arrastre los comandos que quiera desde la lista de comandos hasta la barra de herramientas o el menú.

Eliminar comandos de menús y restablecer barras de menú
Para eliminar un menú entero o uno de sus componentes:

1. Seleccione una de estas opciones de la lista desplegable **Mostrar menús para**:

   - **Predeterminado** (muestra los menús disponibles si no hay ningún documento abierto)
   - **Diseño de MapForce** (muestra los menús disponibles si hay una asignación abierta)

2. Con el cuadro de diálogo "Personalizar" abierto, seleccione (i) el menú que quiere borrar de la barra de herramientas de la aplicación o (ii) el comando que quiere eliminar de uno de estos menús.
3. Ahora puede (i) arrastrar el menú fuera desde la barra de herramientas o el comando fuera del menú, o (ii) hacer clic con el botón derecho en el menú o el comando de menú y seleccionar **Eliminar**.

 © 2016-2022 Altova GmbH

Puede restaurar cualquier barra de menú a su estado original de instalación; para ello selecciónela en la lista desplegable **Mostrar menús para** y haga clic en el botón **Restaurar**.

Personalizar los menús contextuales de la aplicación
Los menús contextuales son aquellos que aparecen si hace clic con el botón derecho en ciertos objetos de la interfaz de la aplicación. Para personalizar esos menús contextuales:

1. Seleccione el menú contextual de la lista desplegable **Seleccionar menú contextual**. Se abre el menú contextual.
2. Haga clic en la pestaña **Comandos**.
3. Arrastre un comando desde la lista Comandos hasta el menú contextual.
4. Para eliminar un comando de un menú contextual, haga clic con el botón derecho en ese comando del menú contextual y seleccione **Eliminar**. Otra opción es arrastrar el comando fuera del menú contextual.

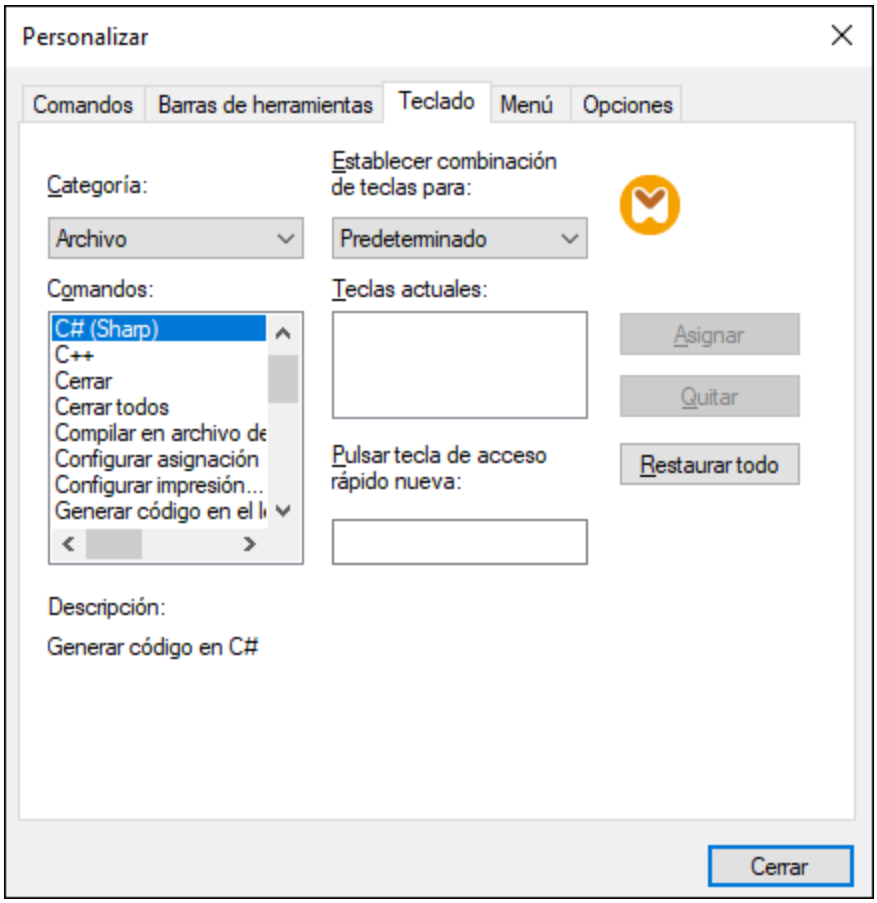
Puede restaurar cualquier menú contextual a su estado original de instalación; para ello selecciónela en la lista desplegable **Seleccionar menú contextual** y haga clic en el botón **Restaurar**.

Sombras de menú
Marque la casilla *Sombras de menú* para que todos los menús tengan sombra.

14.11.2 Personalizar teclas de acceso rápido

En MapForce las teclas de acceso rápido se definen y modifican así: haga clic en el menú **Herramientas** y después en **Personalizar**. Después haga clic en la pestaña **Teclado**. Para asignar una tecla de acceso rápido a un comando:

1. Seleccione **Herramientas | Personalizar** y haga clic en la pestaña *Teclado* del cuadro de diálogo **Personalizar** (*imagen siguiente*).
2. Haga clic en el cuadro combinado *Categoría* y seleccione el menú de MapForce donde está el comando.
3. En el cuadro de lista *Comandos* seleccione el comando al que desea asignar una tecla de acceso rápido.
4. Haga clic en el cuadro de texto *Pulsar tecla de acceso rápido nueva* y haga clic en el botón **Asignar** para asignar la tecla de acceso rápido al comando.



La tecla de acceso rápido aparece ahora en el cuadro de lista *Teclas actuales*. Para borrar el texto del cuadro de texto *Pulsar tecla de acceso rápido,* pulse cualquier tecla de control (**Ctrl**, **Alt** o **Mayús**).

Nota: el cuadro combinado *Establecer combinación de teclas para:* no tiene por ahora ninguna función.

Teclas de acceso rápido

Estas son las teclas de acceso rápido asignadas por defecto:

Teclas de acceso rápido	
F1	Menú **Ayuda**
F2	Marcador siguiente (panel *Resultados*)
F3	Buscar Siguiente
F10	Activar barra de menú
+ del teclado numérico	Expandir el nodo actual
- del teclado numérico	Contraer el nodo actual
* del teclado numérico	Expandir todo a partir del nodo actual
CTRL + TAB	Cambia de una asignación a otra
CTRL + F6	Recorre todas las ventanas abiertas
CTRL + F4	Cierra la asignación activa

 © 2016-2022 Altova GmbH

Alt + F4	Cierra MapForce
Alt + F, F, 1	Abre el último archivo
Alt + F, T, 1	Abre el último proyecto
CTRL + N	Archivo nuevo
CTRL + O	Abrir archivo
CTRL + S	Guardar archivo
CTRL + P	Imprimir archivo
CTRL + A	Seleccionar todo
CTRL + X	Cortar
CTRL + C	Copiar
CTRL + V	Pegar
CTRL + Z	Deshacer
CTRL + Y	Rehacer
Supr	Eliminar componente (con previo aviso)
Mayús + Supr	Eliminar componente (sin previo aviso)
CTRL + F	Buscar
F3	Buscar siguiente
Mayús + F3	Buscar anterior
Teclas de dirección	
(arriba / abajo)	Seleccionar el siguiente / anterior elemento del componente
Esc	Abandonar los cambios / cerrar el cuadro de diálogo
Entrar	Confirma una selección
Teclas de acceso rápido del panel Resultados	
CTRL + F2	Insertar o quitar marcador
F2	Siguiente marcador
Mayús + F2	Marcador anterior
CTRL + Mayús + F2	Quitar todos los marcadores
Teclas de acceso rápido de zoom	
CTRL + rueda del ratón hacia adelante	Acercarse
CTRL + rueda del ratón hacia detrás	Alejarse
CTRL + 0 (cero)	Restablecer el nivel de zoom

14.11.3 Opciones

Puede cambiar las opciones generales y preferencias de MapForce con el comando **Herramientas | Opciones**. A continuación puede ver las opciones disponibles.

- Generales

En la sección *Generales* puede definir estas opciones:

- *Mostrar logotipo* | *Al iniciar el programa* Muestra u oculta el logotipo (pantalla de bienvenida) cuando se inicia MapForce.

- *Mostrar fondo degradado* Habilita y deshabilita el fondo degradado del panel **Asignación**. También puede limitar la presentación de anotaciones a N líneas. Si el texto de la anotación tiene varias líneas y habilita esta opción, solamente se presentarán las primeras X líneas del componente. Esta opción también afecta a las instrucciones SELECT que estén visibles en los componentes.

- *Codificación predeterminada para componentes nuevos.*

  Nombre de la codificación: La codificación predeterminada para archivos XML nuevos se puede definir seleccionando la opción que prefiera de la lista desplegable. Si selecciona una codificación de dos o cuatro bytes como codificación predeterminada (es decir, UTF-16, UCS-2 o UCS-4) también puede elegir entre el orden de bytes little-endian o big-endian. Esta configuración se puede cambiar individualmente para cada uno de los componentes (véase Cambiar la configuración de los componentes 79).

  Orden de bytes: Los documentos con codificación de caracteres de dos o cuatro bytes se pueden guardar con criterio de ordenación de bytes little-endian o big-endian. También puede indicar si se debe incluir una marca de orden de bytes.

- *Configurar la vista previa:* La opción *Usar tiempo de espera de la ejecución* indica el tiempo de espera de ejecución para la vista previa de resultados en el panel **Resultados**.

- *Al activar la pestaña Resultados:* Puede generar resultados en archivos temporales o escribir el resultado directamente en un archivo de resultados (*imagen siguiente*).

  Generar resultados en archivos temporales: Esta es la opción predeterminada. Si el archivo de salida contiene carpetas que no existen todavía, Mapforce se encargará de crearlas. En las ediciones Professional y Enterprise: Si quiere implementar y ejecutar la asignación de datos en un servidor, todos los directorios de la ruta deben existir también en el servidor; de lo contrario se generará un error de ejecución. Consulte también Preparar asignaciones para ejecutarlas en servidores 949.

  Escribir directamente en archivos de salida finales Si la ruta de salida contiene carpetas que no existen todavía se generará un error de asignación. Esta opción sobrescribe los archivos de salida sin solicitar la confirmación previa del usuario.

- *Presentar texto en tramos de N millones de caracteres* Indica el tamaño máximo del texto que aparece en el panel **Resultados** cuando se genera la vista previa de resultados de asignaciones que generan archivos XML. Si el resultado excede este valor, el texto restante se hará visible al hacer clic sobre un botón **Cargar más**. Para más información, consulte Vista previa de resultados 105.

Edición

Estas son las opciones disponibles en esta página:

 © 2016-2022 Altova GmbH

- *Alinear componentes al arrastrarlos con el ratón:* Indica si los componentes y funciones deben alinearse con los demás componentes cuando se arrastran con el ratón (véase Alinear componentes 80).

- *Eliminación inteligente de componentes:* MapForce permite conservar conexiones aunque se hayan eliminado algunos componentes de transformación 75. Mantener conexiones puede ser especialmente útil si tiene varias conexiones secundarias, ya que no tiene que restaurar cada una de ellas manualmente después de eliminar el componente de transformación en cuestión. Para más detalles consulte Conservar conexiones tras eliminación de componentes 102.

Mensajes

En la ventana *Mensajes* puede volver a habilitar notificaciones de mensajes deshabilitados previamente al activar la casilla No volver a mostrar este mensaje.

Generación (*ediciones Enterprise y Professional*)

En *Generación de código* puede configurar opciones relacionadas con la generación de código de programa y de archivos de ejecución de MapForce Server. Para más información consulte los apartados Opciones del generador de código 1207 y Compilar asignaciones de datos en archivos de ejecución de MapForce Server 955 respectivamente.

Java

Puede que necesite añadir esta ruta personal de acceso a un equipo virtual java si está usando un equipo virtual java que no tiene instalador ni crea entradas de registro (por ejemplo, OpenJDK, de Oracle). También puede querer usar esta ruta para suprimir, por la razón que fuere, cualquier otra ruta que MapForce haya detectado automáticamente. Para más detalles consulte Java 1052.

XBRL (*Enterprise Edition*)

En MapForce puede configurar estas opciones XBRL generales para toda la aplicación:

- El idioma de etiqueta de los elementos XBRL y de sus anotaciones
- Las funciones de etiqueta preferidas para el nombre de los elementos XBRL
- El tipo concreto de funciones de etiqueta para las anotaciones de los elementos XBRL
- Paquetes de taxonomías XBRL personales

Depurador (*ediciones Enterprise y Professional*)

En la sección *Depurador* puede definir estas opciones:

- *Longitud máxima de almacenamiento de valores:* Define la longitud de cadena de los valores que aparecen en la ventana **Valores** (15 caracteres como mínimo). Recuerde que si define una longitud de almacenamiento alta, puede que se agote la memoria del sistema.

- *Guardar historial de seguimiento completo* Determina si MapForce debe guardar el historial de todos los valores procesados por todos los conectores de todos los componentes de la asignación mientras dure la depuración. Si activa esta opción, todos los valores procesados por MapForce desde el principio de la ejecución de depuración se almacenarán en memoria y se podrán analizar en la ventana **Valores** hasta que se detenga la depuración. No se recomienda activar esta opción si tiene pensado depurar asignaciones que hagan un uso intensivo de datos porque podría ralentizar la ejecución de depuración y agotar la memoria del sistema. Si desactiva

esta opción, MapForce guardará solamente el historial de seguimiento más reciente de los nodos relacionados con la posición actual de la ejecución.

- Base de datos (*ediciones Enterprise y Professional*)

  En la sección *Base de datos* puede definir la consulta de la base de datos. Para más detalles consulte Configurar la consulta de la base de datos 1053.

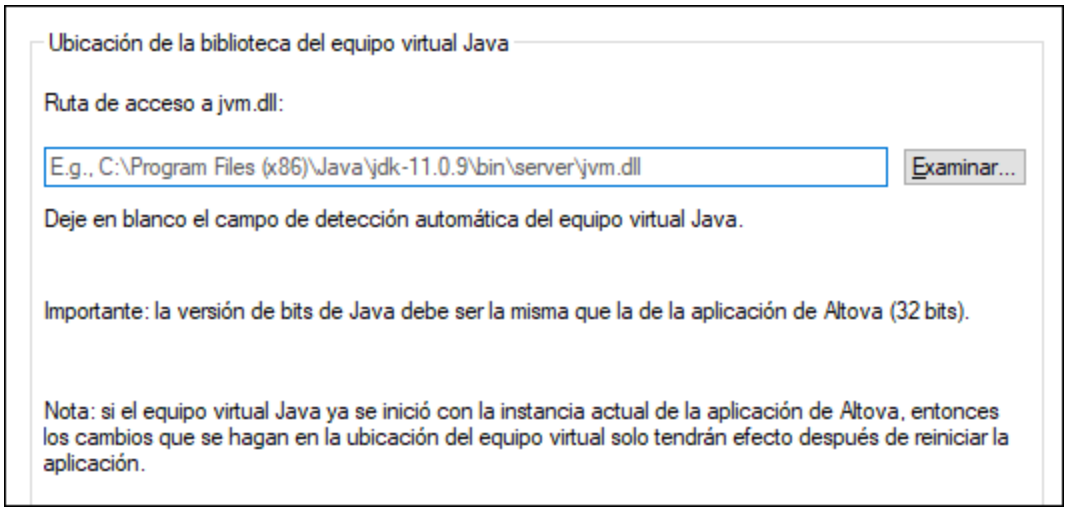
- Proxy de red

  La sección *Proxy de red* permite personalizar la configuración del proxy de red. Esta opción afecta a cómo se conecta la aplicación a Internet. Por defecto, la aplicación usa la configuración del proxy del sistema, por lo que no necesita configurarlo para que funcione. Para más detalles consulte Configuración del proxy de red 1058.

14.11.3.1 Java

En la pestaña *Java* puede introducir la ruta de acceso a un equipo virtual java en su sistema de archivos. Tenga en cuenta que no siempre es necesario agregar una ruta de acceso personal a un equipo virtual. Por defecto, MapForce intenta detectar esta ruta automáticamente leyendo (en este orden) el registro de Windows y la variable de entorno JAVA_HOME. Si se detecta automáticamente cualquier otra ruta de equipo virtual java, tendrá prioridad la ruta personal que se indica en este cuadro de diálogo.

Puede que necesite añadir esta ruta personal de acceso a un equipo virtual java si está usando un equipo virtual java que no tiene instalador ni crea entradas de registro (por ejemplo, OpenJDK, de Oracle). También puede querer usar esta ruta para suprimir, por la razón que fuere, cualquier otra ruta que MapForce haya detectado automáticamente.



Observe lo siguiente:

- la ruta de acceso al equipo virtual java es común a todas las aplicaciones de escritorio de Altova (no a las de servidor). En consecuencia, si cambia esta ruta en una de ellas, el cambio afectará

automáticamente al resto de aplicaciones de Altova.
- la ruta debe apuntar al archivo jvm.dll desde los directorios **\bin\server** o **\bin\client**, relativos al directorio en el que está instalado el JDK.
- la plataforma de MapForce (versión de 31 o de 64 bits) debe ser la misma que la del JDK.
- después de cambiar la ruta de acceso al escritorio virtual java debe reiniciar MapForce para que surta efecto la nueva configuración.

Cambiar la ruta de acceso al escritorio virtual java afecta a:

- la conectividad JDBC
- las funciones de extensión Java para XSLT/XPath

Esta opción no afecta a la generación de código Java.

14.11.3.2 Configurar la consulta de la base de datos

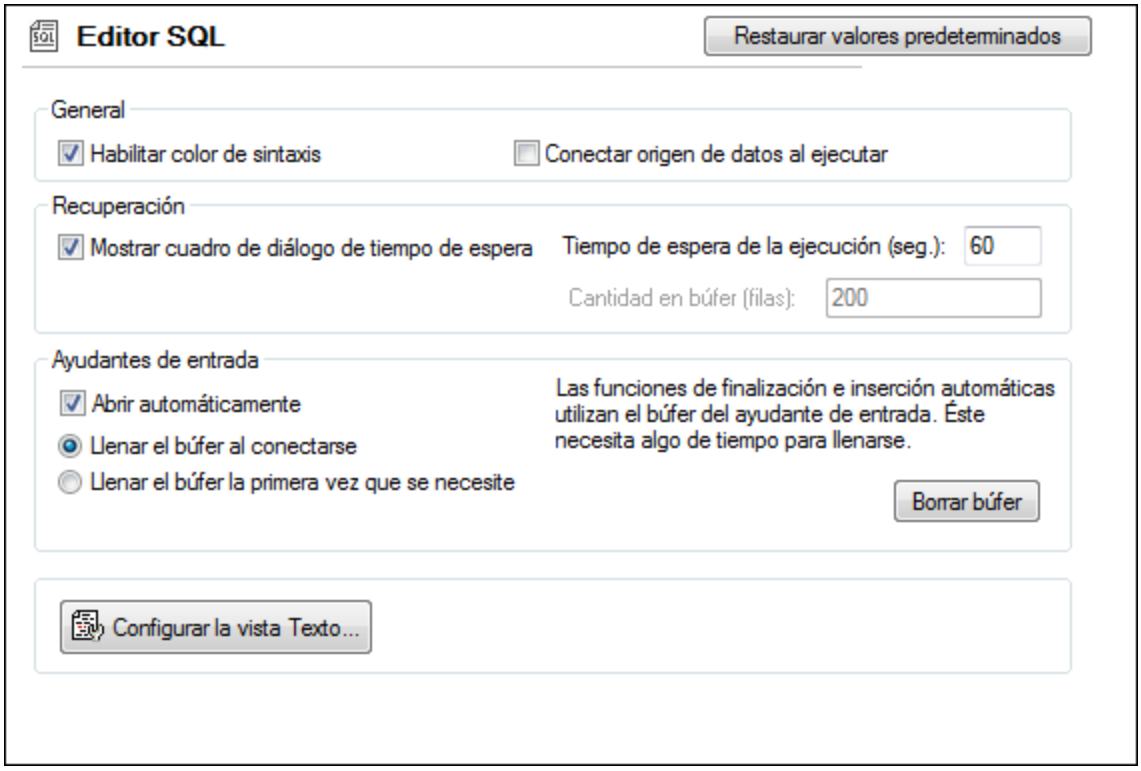
En esta sección explicamos cómo configurar varias de las opciones del editor SQL. Puede acceder a estas opciones de varias formas:

- Al abrir el panel **Consulta de BD** y hacer clic en (**Opciones**).
- Al seleccionar **Herramientas | Opciones | Base de datos** y después la sección relevante.

En la sección *Base de datos* puede definir las opciones del editor SQL, de codificación, la vista de resultados, generación de parámetros SQL y las fuentes. Para más detalles siga leyendo.

Editor SQL

La sección *Editor SQL* permite cambiar las opciones generales del editor SQL (*imagen siguiente*). A continuación describimos las opciones disponibles.



- *Generales* Para ver los elementos de la sintaxis SQL en distintos colores habilite el color de sintaxis. Marque la casilla *Conectar origen de datos al ejecutar* para conectarse al origen de datos relevante automáticamente siempre que se ejecute una instrucción SQL.

- *Recuperación* Marque la casilla *Mostrar cuadro de diálogo de tiempo de espera* para cambiar la configuración de tiempo de espera cuando se supere el plazo de ejecución permitido. Especifique en segundos el tiempo de espera máximo permitido para la ejecución SQL (*tiempo de espera de la ejecución*). También puede definir el número de filas que se ponen en un búfer.

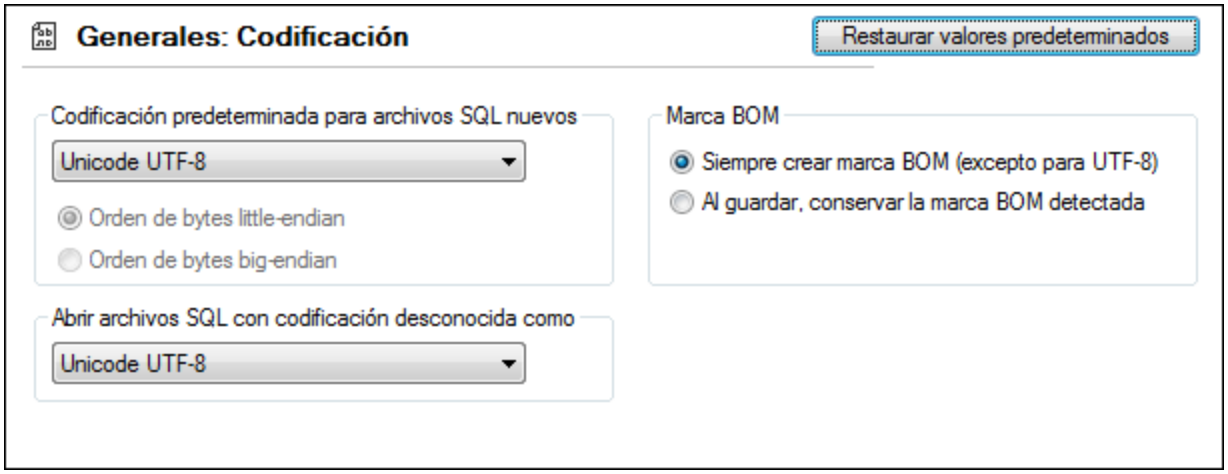
- *Ayudantes de entrada* Para habilitar la finalización automática cuando escriba instrucciones SQL, marque la casilla *Abrir automáticamente* (véase también Finalización automática 277). Puede elegir cuándo rellenar el búfer del ayudante de entrada: al conectarse a un origen de datos o cuando el búfer se use por primera vez. Recuerde que llenar el búfer puede tardar unos minutos. Para restaurar el búfer haga clic en el botón **Borrar búfer**.

- *Configurar la vista Texto:* Permite definir las distintas opciones de la vista Texto, como los márgenes, las pestañas, las ayudas visuales, las opciones de resaltado automático y las teclas de navegación rápida. Para más información consulte Características en la vista Texto 108.

Codificación

En la sección *Codificación* puede especificar las opciones de codificación para los archivos SQL creados o abiertos con el editor SQL (*imagen siguiente*).

 © 2016-2022 Altova GmbH



- *Codificación predeterminada para archivos SQL nuevos:* Aquí puede definir la codificación predeterminada de los archivos nuevos, de modo que cada documento nuevo incluya la misma codificación. Si selecciona una codificación de dos o cuatro bytes como codificación predeterminada (es decir, UTF-16, UCS-2 o UCS-4) también puede elegir entre el orden de bytes little-endian o big-endian para los archivos SQL. La codificación de los archivos no se ve afectada por esta opción.

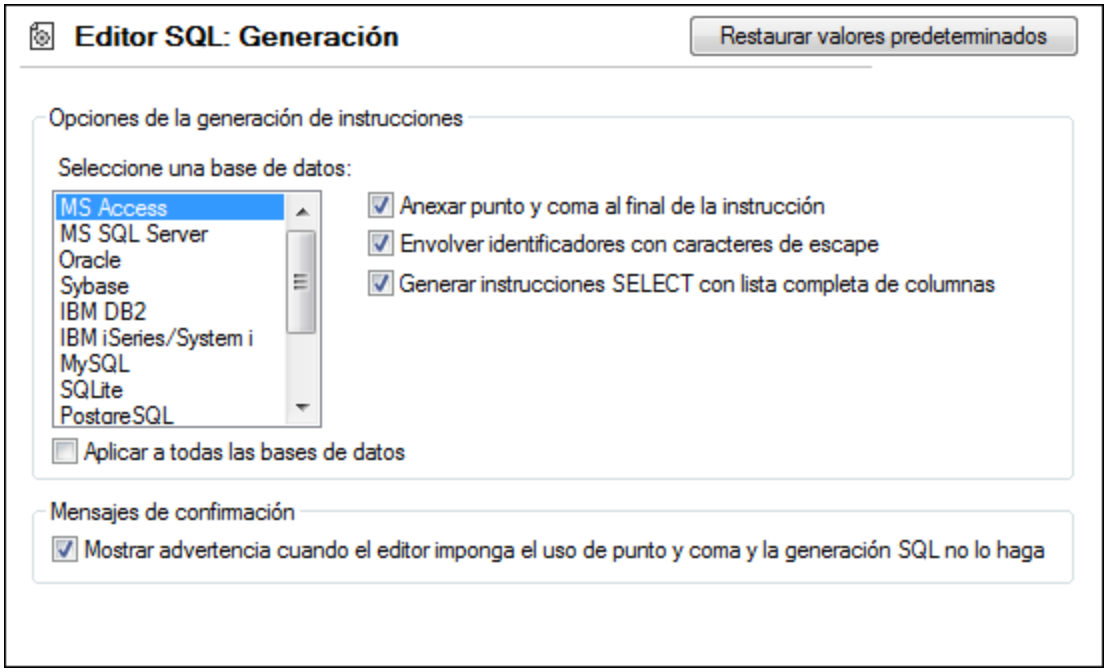
- *Abrir archivos SQL con codificación desconocida como:* Puede seleccionar la codificación con la que quiere abrir un archivo SQL con codificación desconocida.

- *BOM:* Elija si quiere crear un carácter de marca orden de bytes (BOM) para cifrar distinto a UTF-8 o si quiere conservar el BOM detectado.

Nota: los archivos SQL para los que no se ha indicado una codificación se guardan cifrados en UTF-8.

Generación de código SQL

En la sección *Generación* puede indicar la sintaxis de generación de las instrucciones SQL para varios tipos de BD (*imagen siguiente*).

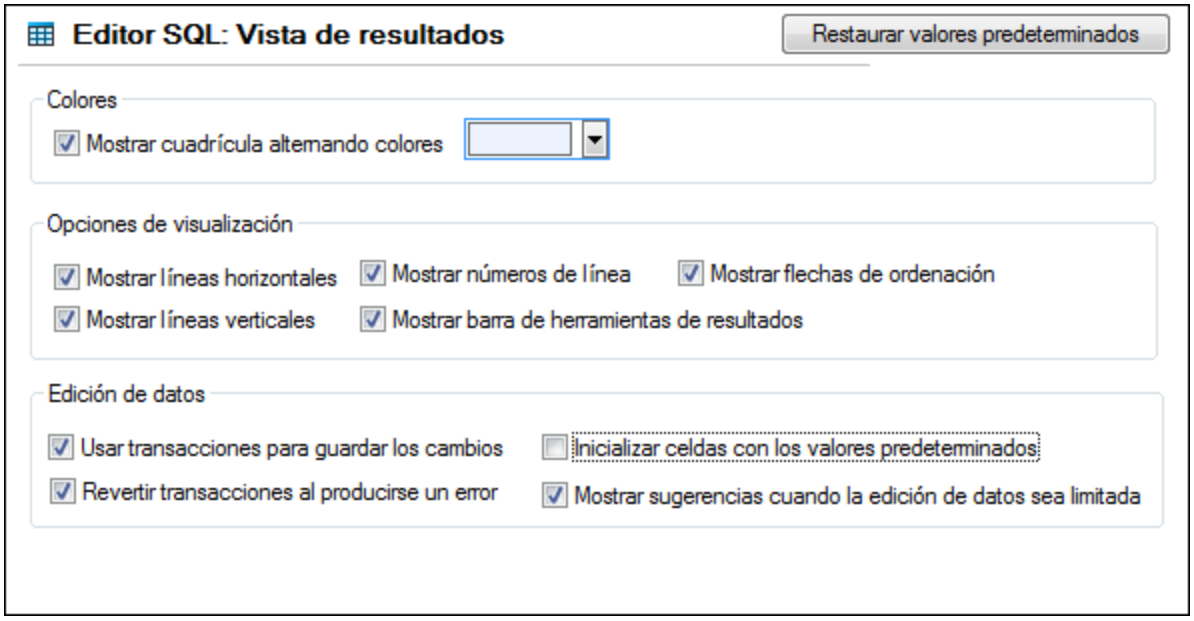


Para definir las preferencias de sintaxis para una BD en concreto, selecciónela en la lista y después habilite o deshabilite las opciones de la derecha. También puede aplicar las mismas opciones a todas las bases de datos (con la casilla *Aplicar a todas las bases de datos*). Recuerde que si usa la misma configuración para todas las bases de datos es posible que no pueda editar datos en Oracle, IBM DB2 e iSeries con una conexión JDBC.

Vista de resultados

Puede configurar el aspecto de la pestaña **Resultados** del panel **Consulta de la BD** en la sección *Vista de resultados* (*imagen siguiente*).

 © 2016-2022 Altova GmbH



Marque la casilla Ver cuadrícula *Mostrar cuadrícula alternando colores* para que las filas de la **vista de resultados** aparezcan como una cuadrícula simple o como cuadrícula filas blancas y de otro color. También puede configurar los colores. El grupo *Opciones de visualización* sirve para configurar el aspecto de las líneas horizontales y verticales de la cuadrícula y para habilitar los números de las líneas y la **barra de herramientas de resultados**.

El grupo de opciones *Edición de datos* sirve para configurar las transacciones, para definir si las celdas se deben rellenar con valores predeterminados y para especificar si la aplicación debe mostrar una sugerencia cuando la edición de datos sea limitada.

Fuentes del editor de texto

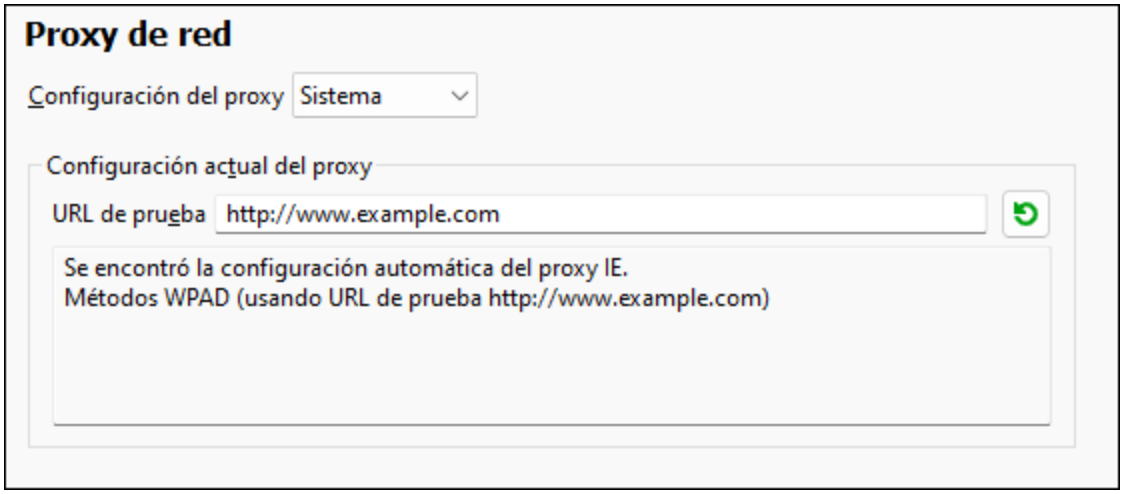
En la *Fuentes del editor de texto* puede configurar el color y las fuentes de las instrucciones SQL (*imagen siguiente*) Puede elegir el tipo de fuente, el estilo y el tamaño del texto que aparece en el Editor SQL. Recuerde que se usará la misma fuente y el mismo tamaño para todos los tipos de texto. Solamente el estilo se puede cambiar para cada tipo de texto. Si accede a **Fuentes del editor de texto** desde el panel **Consulta de BD** puede hacer clic en el botón **Restaurar valores predeterminados** para volver a la configuración original.

14.11.3.3 Configuración del proxy de red

El cuadro de diálogo **Proxy de red** (*imagen siguiente*) permite personalizar la configuración del proxy de red. El sistema viene con una configuración predeterminada para el proxy, por lo que este funcionará sin necesidad de configurarlo, pero si quiere usar un proxy de red alternativo puede usar estas opciones para cambiar la configuración como quiera.

Nota: la configuración del proxy de red es común a todas las aplicaciones de Altova MissionKit. En consecuencia, si cambia esta configuración en cualquiera de esas aplicaciones, el cambio afectará automáticamente a todas las demás.

 © 2016-2022 Altova GmbH



Usar la configuración del proxy del sistema
Usa los parámetros de Internet Explorer (IE), que se pueden configurar desde las opciones del proxy de red. También consulta los parámetros configurados con **`netsh.exe winhttp`**.

Configuración automática del proxy
Existen las siguientes opciones:

- *Configuración de detección automática:* consulta un script WPAD (**`http://wpad.LOCALDOMAIN/wpad.dat`**) vía DHCP o DNS y lo usa para configurar el proxy.
- *URL del script*: indica una HTTP URL a un script (`.pac`) de configuración automática del proxy cuyos parámetros se aplican para configurar el proxy.
- *Volver a cargar*: reinicia y vuelve a cargar la configuración automática actual del proxy. Esta acción requiere Windows 8 o superior y puede llegar a tardar 30 segundos en tener efecto.

Configuración manual del proxy
Puede indicar manualmente el nombre completo de host y el puerto para los proxys de los respectivos protocolos. Es posible que haya un esquema compatible incluido en el nombre de host (por ejemplo: **`http://hostname`**). Si el proxy es compatible no es necesario que el esquema sea el mismo que el protocolo correspondiente.

Existen las siguientes opciones:

- *Usar este servidor de proxy para todos los protocolos:* usa el nombre de host y el puerto del Proxy HTTP para todos los protocolos.
- *Ningún proxy para:* muestra una lista de elementos separados por punto y coma (`;`) que pueden ser nombres de host, nombres de dominios o direcciones IP para hosts para los que no hay que usar proxy. Las direcciones IP no se pueden truncar y las direcciones IPv6 deben colocarse entre corchetes (por ejemplo: `[2606:2800:220:1:248:1893:25c8:1946]`). Los nombres de dominio deben empezar por punto (por ejemplo: `.example.com`).
- *No use el servidor proxy para direcciones locales*: si se marca esta opción, se añade el elemento `<local>` a la lista *Ningún proxy para*. Si se selecciona esta opción no se usará proxy para: (i) `127.0.0.1`, (ii) [::1], (iii) todos los nombres de host que no contengan punto (`.`).

Nota: si ha configurado un servidor de proxy y quiere implementar una asignación en Altova FlowForce Server, debe seleccionar la opción *No usar el servidor proxy para direcciones locales*.

Configuración actual del proxy
Proporciona un registro detallado de la detección del proxy. Se puede actualizar con el botón **Actualizar** a la derecha de *URL de prueba* (por ejemplo, al cambiar la URL de prueba o cuando se ha cambiado la configuración del proxy).

- *URL de prueba:* una URL de prueba se puede usar para ver qué proxy hay que usar para esa URL en concreto. No se trata de una URL de entrada/salida. Este campo no debe estar vacío si se ha optado por la configuración automática del proxy (seleccionando *Usar la configuración del proxy del sistema* o *Configuración automática del proxy*).

 © 2016-2022 Altova GmbH

14.12 Ventanas

En este apartado puede consultar todos los comandos del menú **Ventanas**.

- En cascada

  Este comando reorganiza todas las ventanas abiertas **en forma de cascada** (es decir, escalonadas).

- En mosaico horizontal

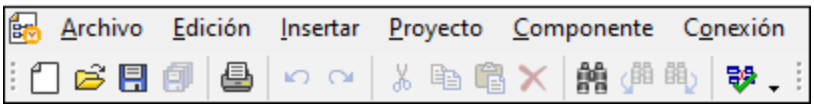
  Este comando reorganiza todas las ventanas abiertas **en forma de mosaico horizontal**, mostrando todas las ventanas a la vez.

- En mosaico vertical

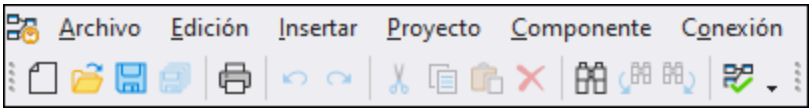
  Este comando reorganiza todas las ventanas abiertas **en forma de mosaico vertical**, mostrando todas las ventanas a la vez.

- Tema clásico/ligero/oscuro

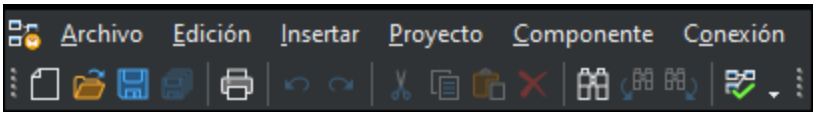
  MapForce permite elegir entre estos temas: *clásico*, *ligero* y *oscuro*. A continuación puede ver ejemplos de los tres temas. La opción predeterminada es el tema clásico.



Tema clásico



Tema ligero



Tema oscuro

- 1 <NombreAsignación>

  Hace referencia al primer diseño de asignación abierto. Si hay más asignaciones abiertas en ese momento también aparecerán en el menú contextual.

- Windows

  Esta lista enumera todas las ventanas que están abiertas en cada momento y permite cambiar de una ventana a otra. También puede usar las teclas **Ctrl+Tabulador** o **Ctrl+F6** para pasar de una ventana a otra.

14.13 Ayuda

En este apartado puede consultar todos los comandos del menú **Ayuda**.

Contenido

Abre la ayuda en pantalla de MapForce por la tabla de contenido, que aparece en el panel izquierdo de la ventana de ayuda. Esta tabla de contenido ofrece un resumen de la documentación. Haga clic en una entrada de la tabla para abrir la sección correspondiente de la documentación.

Índice

Abre la ayuda en pantalla de MapForce por el índice de palabras clave, que aparece en el panel izquierdo de la ventana de ayuda. Este índice enumera todas las palabras clave de la ayuda y permite navegar a un tema con solo hacer doble clic en la palabra clave correspondiente. Si una palabra clave está asociada a varios temas, la ventana de ayuda muestra todos estos temas en pantalla.

Buscar

Abre la ayuda en pantalla de MapForce por la función de búsqueda, que aparece en el panel izquierdo de la ventana de ayuda. Para buscar un término introduzca el término de búsqueda en el campo de consulta y pulse **Entrar**. El sistema de ayuda realiza una búsqueda en toda la documentación y devuelve una lista de resultados. Haga doble clic en una entrada para abrir la sección correspondiente de la documentación.

Activación del software

Tras descargar el producto de software de Altova puede registrarlo o activarlo con una clave de evaluación gratuita o con una clave de licencia permanente.

- ***Licencia gratuita:*** cuando inicie el software por primera vez aparecerá el cuadro de diálogo "Activación del software". Este cuadro de diálogo incluye un botón para solicitar una licencia de evaluación gratuita. Introduzca su nombre, el nombre de su compañía y su dirección de correo electrónico y haga clic en **Enviar solicitud**. Altova le enviará un archivo de licencia al correo electrónico proporcionado. Guarde el archivo en su equipo. Al hacer clic en **Enviar solicitud** aparece un campo en la parte inferior del cuadro de diálogo, que es donde debe introducir la ruta de acceso al archivo de licencia. Puede escribir la ruta de acceso o navegar hasta el archivo de licencia. Por último, haga clic en **Aceptar**. (En el cuadro de diálogo de activación del software también puede hacer clic en **Cargar licencia nueva** para acceder al cuadro de diálogo en el que se introduce la ruta de acceso.) El software permanecerá desbloqueado durante 30 días.
- ***Clave de licencia permanente:*** el cuadro de diálogo "Activación del software" también incluye un botón para comprar una clave de licencia permanente. Este botón conduce a la tienda en línea de Altova, donde podrá adquirir una clave de licencia permanente para el producto. Recibirá por correo electrónico un archivo que contiene los datos de la licencia. Existen tres tipos de licencias permanentes: de tipo instalado, de usuario concurrente y de usuario designado. Las *licencias de tipo instalado* son cada una para un único equipo. En el caso de las *licencias de usuario concurrentes*, si adquiere una de estas licencias para **N** usuarios podrá instalar la licencia en un máximo de **10N** equipos pero solo **N** usuarios podrán usar el software al mismo tiempo. Las *licencias de usuario designado* autorizan a un usuario específico a usar el software en un máximo de 5 equipos distintos. Para activar su software haga clic en **Cargar una licencia nueva** e introduzca la ruta de acceso al archivo de licencia en el cuadro de diálogo "Activación del software" que aparece. Por último, haga clic en **Aceptar**.

Nota: en el caso de licencias para varios usuarios, se le pedirá a cada usuario que introduzca su

 © 2016-2022 Altova GmbH

nombre.

Claves por correo electrónico y las distintas formas de activar las licencias de los productos de Altova
El correo electrónico que recibirá de Altova contiene, en un adjunto, el archivo de la licencia, que tiene la extensión **`.altova_licenses`**.

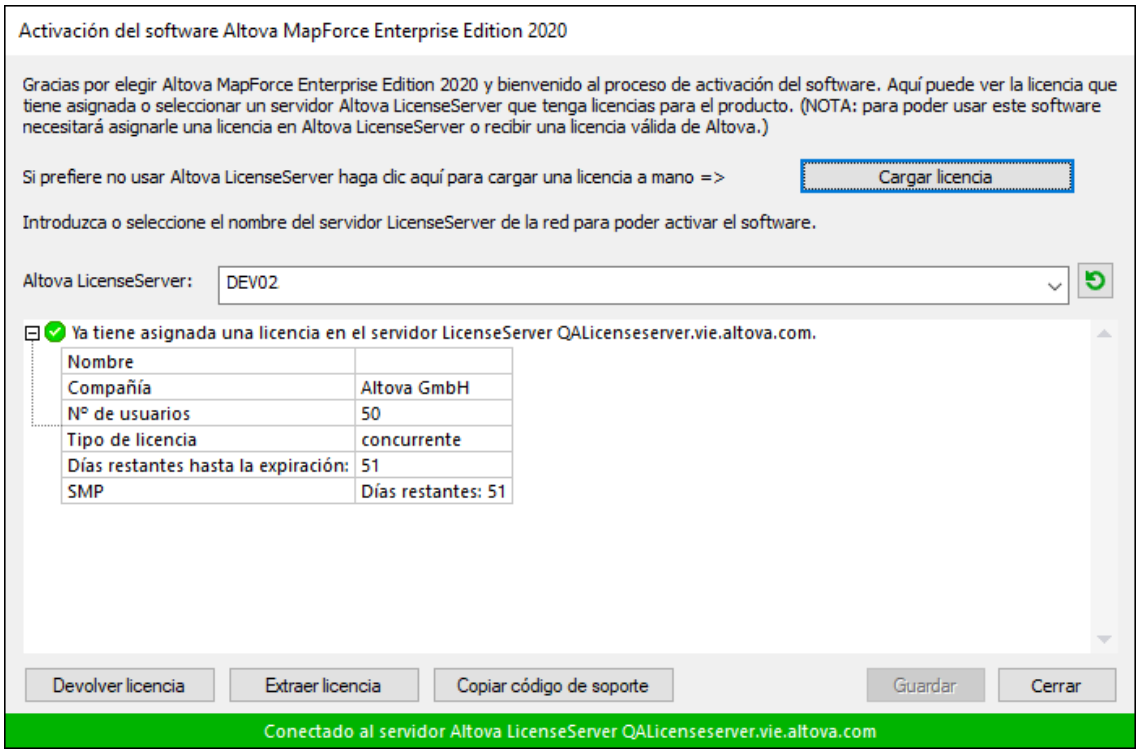
Para activar su producto de Altova, puede optar por una de las siguientes opciones:

- Guardar el archivo de licencia (**`.altova_licenses`**) en su equipo, hacer doble clic en el archivo de licencia, introducir los detalles necesarios en el cuadro de diálogo que aparece y finalmente hacer clic en **Aplicar claves**.
- Guardar el archivo de licencia (**`.altova_licenses`**) en su equipo. En su producto de Altova seleccione el comando de menú **Ayuda | Activación del software** y después **Cargar una licencia nueva**. Navegue hasta el archivo de licencia o introduzca la ruta de acceso al archivo de licencia en el cuadro de diálogo "Activación del software" y haga clic en **Aceptar**.
- Guardar el archivo de licencia (**`.altova_licenses`**) en su equipo y cargarlo desde esa ubicación a su Altova LicenseServer. Puede: (i) adquirir la licencia de su producto Altova con el cuadro de diálogo de activación de software del producto (*véase más abajo*) o (ii) asignar la licencia al producto de Altova LicenseServer. Para obtener más información sobre la gestión de licencias con el LicenseServer, lea el resto de esta sección.

El cuadro de diálogo "Activación del software" (*imagen siguiente*) se abre con el comando **Ayuda | Activación del software**.

Hay dos maneras de activar el software:

- registrando la licencia en el cuadro de diálogo "Activación del software". Para ello haga clic en **Cargar una licencia nueva** y navegue hasta el archivo de la licencia. Haga clic en **Aceptar** para confirmar la ruta de acceso al archivo de licencia y para confirmar los datos que haya introducido (su nombre, en el caso de licencias para más de un usuario); a continuación, haga clic en **Guardar** para finalizar el proceso.
- asignando una licencia a través de un servidor Altova LicenseServer de la red: para adquirir una licencia a través de un servidor Altova LicenseServer de la red haga clic en el botón **Usar Altova LicenseServer**, situado al final del cuadro de diálogo. Seleccione el equipo en el que está instalado el LicenseServer que quiere usar. Tenga en cuenta que la autodetección de los License Servers funciona con emisiones enviadas por LAN. Este tipo de emisiones se limitan a una subred, por lo que Altova License Server debe estar en la misma subred que el equipo del cliente para que funcione la autodetección. Si esta no funciona, introduzca el nombre del servidor. Para ello es necesario que el servidor LicenseServer tenga una licencia para su producto en el repositorio de licencias. Si así es, el cuadro de diálogo "Activación del software" emite un mensaje a tal efecto (*imagen siguiente*). Haga clic en el botón **Guardar** para adquirir la licencia.



Una vez se ha adquirido una licencia para un equipo específico (es decir, "instalada") del servidor LicenseServer, no se puede devolver al mismo hasta 7 días después. Transcurridos estos 7 días podrá devolver la licencia de ese equipo (con el botón **Devolver licencia**) para que pueda ser adquirida por otro cliente. No obstante, el administrador de LicenseServer puede anular asignaciones de licencias desde la interfaz web del servidor LicenseServer en cualquier momento. Observe que únicamente se pueden devolver las licencias instaladas en equipos específicos, no las licencias concurrentes.

Extracción de licencias

Puede extraer una licencia del repertorio durante un período máximo de 30 días de modo que la licencia se almacene en el equipo donde se ejecuta el producto. Esto le permitirá trabajar sin conexión a Internet, lo cual puede ser útil si desea trabajar en un entorno que no dispone de acceso a su servidor Altova LicenseServer (p. ej. cuando el producto servidor de Altova está instalado en un equipo portátil y el usuario se encuentra de viaje). Mientras la licencia esté extraída, LicenseServer indicará que la licencia está en uso y no podrá ser utilizada por ningún otro equipo. La licencia vuelve automáticamente a su estado insertado cuando finaliza el período de extracción de la licencia. La licencia extraída también se puede insertar en el servidor en cualquier momento con el botón **Insertar** del cuadro de diálogo "Activación del software".

Siga estas instrucciones para extraer una licencia:

1. En el cuadro de diálogo "Activación del software" haga clic en el botón **Extraer licencia** (*imagen anterior*).
2. Aparece el cuadro de diálogo "Extracción de licencias". Seleccione el periodo de extracción y haga clic en **Extraer**.
3. La licencia se extrae y ocurren dos cosas: (i) el cuadro de diálogo "Activación del software" muestra información sobre la extracción de la licencia, incluida la fecha y la

 © 2016-2022 Altova GmbH

hora en la que expira el plazo de extracción y (ii) en lugar del botón **Extraer licencia**, aparece el botón **Insertar licencia**. Para insertar la licencia basta con hacer clic en este botón. Como la licencia vuelve automáticamente a su estado de inserción cuando finaliza el plazo de extracción, compruebe que el plazo seleccionado coincide con el período de tiempo que tiene pensado trabajar sin conexión a Internet.

Para devolver una licencia esta debe ser de la misma versión principal que el producto de Altova con el que se extrajo. Por tanto, es recomendable devolver la licencia antes de actualizar el producto de Altova correspondiente a la siguiente versión principal.

Nota: para poder extraer licencias esta característica debe estar habilitada en el servidor LicenseServer. Si esta característica no está habilitada, recibirá un mensaje de error a tal efecto cuando trate de extraer una licencia. Cuando esto ocurra, póngase en contacto con el administrador de su servidor LicenseServer.

Copiar código de soporte
Haga clic en **Copiar código de soporte** para copiar los detalles de la licencia en el portapapeles. Esta es la información que deberá introducir al ponerse en contacto con el equipo de soporte técnico a través del formulario de soporte técnico.

Altova LicenseServer es una práctica herramienta para administrar en tiempo real todas las licencias de Altova de la red y ofrece información detallada sobre cada licencia, asignaciones a clientes y uso de las licencias. La ventaja de usar este producto está en sus características administrativas. Altova LicenseServer puede descargarse gratis del sitio web de Altova. Para más información consulte la documentación de Altova LicenseServer.

Formulario de pedido

Hay dos maneras de comprar licencias para los productos de Altova: con el botón **Comprar una licencia permanente** del cuadro de diálogo "Activación del software" (*ver apartado anterior*) o con el comando **Ayuda | Formulario de pedido**, que le lleva directamente a la tienda en línea de Altova, donde puede adquirir claves de licencias para los productos de Altova.

Registro del software

Este comando abre la página de registro de productos de Altova en una pestaña del explorador. Si registra el software, recibirá información sobre actualizaciones y versiones nuevas del producto.

Buscar actualizaciones

Comprueba si existe una versión más reciente del producto en el servidor de Altova y emite un mensaje a tal efecto.

Centro de soporte técnico

Es un enlace al centro de soporte técnico del sitio web de Altova. El centro de soporte técnico incluye preguntas frecuentes, foros de debate y un formulario para ponerse en contacto con el equipo de soporte técnico de Altova.

Preguntas más frecuentes

Es un enlace a la página de preguntas frecuentes del sitio web de Altova. Esta página se actualiza constantemente con las preguntas que recibimos de nuestros clientes.

- Descargar herramientas gratis y componentes

  Es un enlace al centro de descargas de componentes del sitio web de Altova. Aquí puede descargar software adicional para usarlo con los productos de Altova, como procesadores XSLT y XSL-FO y paquetes de integración. Estos componentes suelen ser totalmente gratis.

- MapForce en Internet

  Es un enlace al sitio web de Altova, donde encontrará más información sobre MapForce, otros productos de Altova y tecnologías relacionadas.

- Cursos de MapForce

  Es un enlace a la página de cursos del sitio web de Altova. Aquí puede seguir todos los cursos sobre productos y tecnologías relacionados con la línea de software de Altova.

- Acerca de MapForce

  Abre la pantalla de presentación de la aplicación, que incluye el número de versión del producto e información sobre copyright. Si usa la versión de 64 bits de la aplicación, esto se ve en el nombre de la aplicación, que lleva el sufijo (x64). La versión de 32 bits no lleva ningún sufijo.

© 2016-2022 Altova GmbH

15 Generador de código

El generador de código es una característica integrada de MapForce que sirve para generar código Java, C++ o C# a partir de archivos de asignación de datos diseñados con MapForce. Puede generar código a partir de asignaciones de datos con un solo origen y destino, pero también de asignaciones con varios orígenes de datos y varios destinos. Como resultado obtendrá una aplicación entera y totalmente funcional que lleva a cabo la operación de asignación de datos. Una vez generado el código, la asignación se podrá ejecutar con solo iniciar la aplicación recién generada. También podrá importar el código generado en otras aplicaciones o ampliarlo con más funciones.

El objetivo principal del código generado es ejecutar una asignación de datos de MapForce. Si su asignación usa esquemas XML o DTD, puede generar bibliotecas contenedoras de esquemas para los esquemas que usa la asignación, lo que permite leer o escribir datos en o desde instancias XML. Las bibliotecas contenedoras de esquemas permiten trabajar con datos XML de forma programática usando tipos generados a partir del esquema, sin prestar demasiada atención a la API XML subyacente. Para más detalles consulte Generar código fuente a partir de esquemas XML o DTD 1110.

El código generado se expresa en los lenguajes de programación C++, Java o C#.

Lenguaje de destino	**C++**	**C#**	**Java**
Entornos de desarrollo	Microsoft Visual Studio 2013, 2015, 2017, 2019, 2022	Microsoft Visual Studio 2013, 2015, 2017, 2019, 2022 Marcos de destino: • .NET Framework • .NET Core 3.1 • .NET 5.0 • .NET 6.0	Java SE JDK 8 u 11 (incluido OpenJDK) Eclipse 4.4 o superior Apache Ant (compile el archivo.xml)
Implementaciones XML DOM	MSXML 6.0 Apache Xerces 3	System.Xml	JAXP
API de BD	ADO	ADO.NET	JDBC

Nota: solo se puede considerar seguro el código generado de MapForce si también lo son el XML DOM subyacente y las bibliotecas API de bases de datos. Aunque no se puede comprobar ni garantizar que el código generado sea seguro, es posible que se puedan ejecutar varias instancias del código de asignación al mismo tiempo sin problemas en la mayoría de los casos.

C++

Puede configurar si el resultado generado en C++ debe usar MSXML 6.0 o Apache Xerces 3. MapForce genera archivos completos de proyecto (.vcproj) y solución (.sln) para todas las versiones de Visual Studio compatibles (véase la tabla de más arriba). El código generado puede ser compatible con MFC.

Requisitos:

1. Para compilar el código C++ generado debe tener Windows SDK instalado en su equipo.
2. Para usar Xerces 3 para C++ debe instalarlo y compilarlo según las instrucciones de http://xerces.apache.org/xerces-c/. Asegúrese de que añade la variable de entorno XERCES3 que apunta al directorio en el que instaló Xerces, por ejemplo `C:\xerces-c-3.2.2`. Además, la variable de entorno PATH debe incluir la ruta de la ubicación donde se encuentran los archivos binarios Xerces, por ejemplo: `%XERCES3%\bin`.
3. Al compilar código C++ para Visual Studio mientras usa una biblioteca Xerces precompilada para Visual C++ debe cambiar la configuración del compilador en todos los proyectos de la solución:

   a) Seleccione todos los proyectos en el explorador de soluciones.
   b) En el menú **Proyecto** haga clic en **Propiedades**.
   c) Haga clic en **Propiedades de la configuración | C/C++ | Lenguaje**.
   d) En la lista de configuraciones seleccione *Todas las configuraciones*.
   e) Cambie **Treat wchar_t as Built-in Type** por **No (/Zc:wchar_t-)**.

C#

El código C# generado lo puede usar cualquier lenguaje de programación apto para .NET, como VB.NET, Managed C++ o J#. Los archivos de proyecto se pueden generar para todas las versiones compatibles de Visual Studio (véase la tabla de más arriba).

Java

El código Java generado se escribe conforme a la API Java para procesamiento XML (JAXP) e incluye un archivo de compilación Ant y archivos de proyecto para las versiones compatibles de Java y Eclipse (véase la tabla de más arriba).

Resultados generados

La carpeta de destino designada incluye todas las bibliotecas y los archivos necesarios para ejecutar la asignación, es decir:

- Un número variable de bibliotecas de Altova necesarias para la asignación (por ejemplo, bibliotecas de funciones de Altova, bibliotecas de BD)
- Una aplicación de asignación completa. Al compilarla y ejecutarla la aplicación lleva a cabo la transformación de la aplicación.

Plantillas de generación de código

El código generado se puede personalizar con un archivo de plantilla 1209 llamado SPL (Spy Programming Language). Esto puede ser útil, por ejemplo, si quiere personalizar código conforme a las convenciones de su empresa, o bien reemplazar bibliotecas concretas en el código generado.

© 2016-2022 Altova GmbH

15.1 Novedades

Versión 2022 R2

- Compatibilidad con Visual Studio 2022.
- Compatibilidad con .NET 6.0.

Versión 2021 R2

- Al generar código en C# puede seleccionar .NET Core 3.1, .NET 5.0 o .NET 6.0 como marco de destino desde las opciones del generador de código 1207 (además de la compatibilidad con proyectos .NET Framework que ya existe).

Versión 2020 R2

- El código generado para las bibliotecas contenedoras de esquemas XML ahora ofrecen un mayor control sobre los espacios de nombres y los prefijos de los elementos. Por ejemplo, hemos introducido métodos nuevos para declarar o sobrescribir los espacios de nombres de un elemento o para anexar un elemento con un espacio de nombres prefijado. Consulte Ejemplo: orden de compra 1147 .

Versión 2020

- Compatibilidad con Visual Studio 2019.
- Fin de compatibilidad con Visual Studio 2008.

Versión 2018

- Compatibilidad con Microsoft Visual Studio 2013, 2015, 2017.
- Fin de compatibilidad con Visual Studio 2005 y Xerces 2.x.

Versión 2014

- Eliminación de la opción de modo de compatibilidad para la generación de código.

Versión 2011
Corrección de fallos y mejoras.

Versión 2010 R3

- Compatibilidad con Microsoft Visual Studio 2010.
- Compatibilidad con MSXML 6.0 en el código C++ generado.
- Compatibilidad con componentes de destino de 64 bits en proyectos C++ y C#.

Versión 2010

- Las facetas "enumeration" de XML Schema ya están disponibles como constantes simbólicas en las clases generadas (usando plantillas de 2007r3).

Versión 2009 sp1

- Compatibilidad con Apache Xerces 3.x (las versiones Xerces 2.6.x y posteriores siguen siendo compatibles).

Versión 2009

- Se ha rediseñado la implementación de la asignación generada para que sea compatible con secuencias y agrupaciones. No se han realizado cambios en la API.
- Sin novedades.

Versión 2008 R2

- Generación de archivos de proyectos de Visual Studio 2008 para C# y C++.
- El código de asignación C# y Java generado con MapForce puede usar lectores y sistemas de escritura, secuencias, cadenas o documentos DOM como componentes de origen y destino.

Versión 2008

- Las plantillas SPL 2007 R3 se han mejorado:
    - Ahora puede eliminar elementos uno por uno
    - Ahora puede acceder a metadatos del esquema (p. ej. nombres de elementos, facetas, enumeraciones, apariciones, etc.)
    - Los tipos complejos derivados por extensión ahora se generan como clases derivadas

Versión 2007 R3

El generador de código se ha rediseñado para la versión 2007 Release 3 con el objetivo de simplificar el uso del código generado, reducir el volumen del código y mejorar el rendimiento.

- Gestión de documentos y nodos XML con propiedad explícita para evitar pérdida de memoria y habilitar subprocesos múltiples
- Sintaxis nueva para evitar colisiones de nombres
- Tipos de datos nuevos para un uso más sencillo y un mayor rendimiento (uso de tipos nativos cuando sea posible, control de valores null...)
- Los atributos ya no se generan como colecciones
- El contenido de elementos simples ahora también se considera un atributo especial
- Un modelo de objetos interno nuevo (importante para las plantillas SPL personalizadas)
- Modo de compatibilidad para generar código en el estilo de versiones anteriores
- Clases contenedoras ahora sólo se generan a petición para código más pequeño

© 2016-2022 Altova GmbH

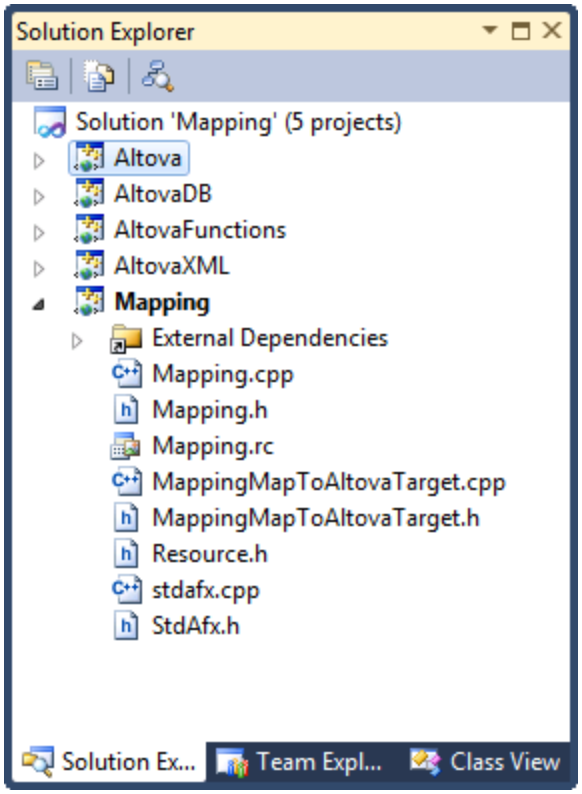
15.2 Generar código C++

Puede generar código C++ para Visual Studio 2013, 2015, 2017, 2019, 2022. El código generado incluye archivos **.sln** y .**vcproj** para Visual Studio. A la hora de generar código debe tener en cuenta que:

- algunas características de MapForce no son compatibles con el código de programa generado. Para más información consulte Funciones compatibles en el código generado 1482.
- puede generar código a partir de un diseño de asignación (.mfd) o de un proyecto de asignación de datos (.mfp). Si genera código a partir de una asignación de datos, la aplicación resultante ejecuta la transformación de asignación de datos correspondiente. Por el contrario, si genera código a partir de un proyecto de MapForce (.mfp) que incluya varias asignaciones de datos, la aplicación resultante ejecuta en bloque todas las asignaciones que incluye el proyecto.
- las opciones generales de generación de código se pueden cambiar haciendo clic en el comando de menú **Herramientas | Opciones** y después en la pestaña *Generación*. Consulte Opciones del generador de código 1207.
- puede cambiar el nombre de la aplicación que se genera y otras opciones de configuración desde el comando de menú **Archivo | Configurar asignación**. El nombre predeterminado de la aplicación es **Mapping**. Consulte también Configuración de la asignación 115.
- para más información sobre las rutas ene l código generado consulte el apartado Rutas de acceso según el entorno de ejecución 84.
- si la asignación de datos contiene componentes de base de datos, puede ver determinadas opciones de la base de datos haciendo clic en el componente y seleccionando el comando de menú **Componente | Propiedades**. Consulte Configurar componentes de BD 278.
- la compatibilidad para las conexiones de BD varía según la plataforma y existen tipos de conexiones que no son compatibles con todas ellas. Si quiere conectar una asignación a una BD, elija una conexión de BD que sea compatible con el entorno de destino para el que genera el código. Para más detalles consulte Asignaciones de BD en distintos entornos de ejecución 160.

Las soluciones C++ generadas en MapForce suelen incluir:

- varias bibliotecas firmadas por Altova que la asignación de datos necesita (todas empiezan con el prefijo **Altova**) y
- el proyecto de asignación principal (en nuestro ejemplo: **Mapping**), que incluye la aplicación de asignación de datos y los archivos independientes.



Solución C++ de muestra generada con MapForce

Esta sección incluye varios apartados:

- Generar código a partir de una asignación de datos 1072
- Generar código a partir de un proyecto de asignación de datos 1073
- Compilar el proyecto 1074
- Ejecutar la aplicación 1074

15.2.1 Generar código a partir de una asignación de datos

Para generar código C++ a partir de un archivo de diseño de asignación de datos (.mfd):

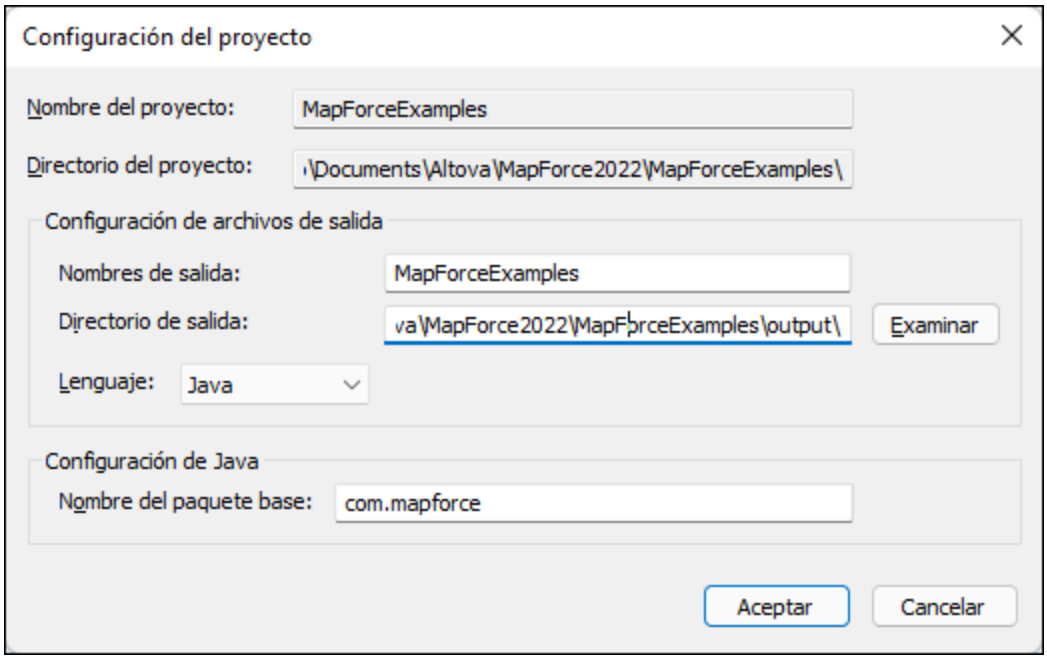
1. Revise y seleccione las opciones de generación de código 1207 adecuadas.
2. En el menú **Archivo** haga clic en el comando **Generar código en | C++**.
3. Seleccione el directorio de destino para los archivos que se deben generar y haga clic en Aceptar para confirmar. El resultado de la generación de código (mensaje de error o éxito) aparece en la ventana Mensajes.

El nombre predeterminado de la aplicación generada es **Mapping**. Si lo prefiere, puede modificar el nombre predeterminado así como otras opciones de configuración de la aplicación (*véase Cambiar configuración de la asignación* 115).

15.2.2 Generar código a partir de un proyecto de asignación

Para generar código a partir de un proyecto de asignación de datos (.mfp):

1. Si todavía no lo ha hecho, abra el proyecto de asignación de datos en MapForce.
2. Haga clic con el botón derecho en la ventana Proyecto y después elija el comando **Propiedades**.



3. Revise y modifique la configuración del proyecto según proceda (sobre todo debe comprobar que el lenguaje de destino y el directorio de salida están configurados correctamente). Después haga clic en **Aceptar**.
4. En el menú **Proyecto** haga clic en el comando **Generar código para todo el proyecto**.

El progreso y el resultado del proceso de generación de código (con mensajes de error o éxito) aparece reflejado en la ventana Mensajes.

La aplicación generada recibe el mismo nombre que el nombre del proyecto, pero si el nombre del proyecto contiene espacios, en el nombre de la aplicación los espacios se convierten en caracteres de subrayado. El código se genera por defecto en el directorio donde está ubicado el proyecto de MapForce, en el subdirectorio **output**.

Para cambiar el directorio de salida y el nombre del proyecto haga clic en el proyecto en la ventana Proyecto y después seleccione el comando de menú **Proyecto | Propiedades**. Si el proyecto de MapForce tiene carpetas, podrá cambiar las opciones de generación de código para cada una de las carpetas por separado (clic con el botón derecho en una carpeta y después Propiedades en el menú contextual). Todas las carpetas del proyecto heredan por defecto las opciones de generación de código del proyecto de MapForce.

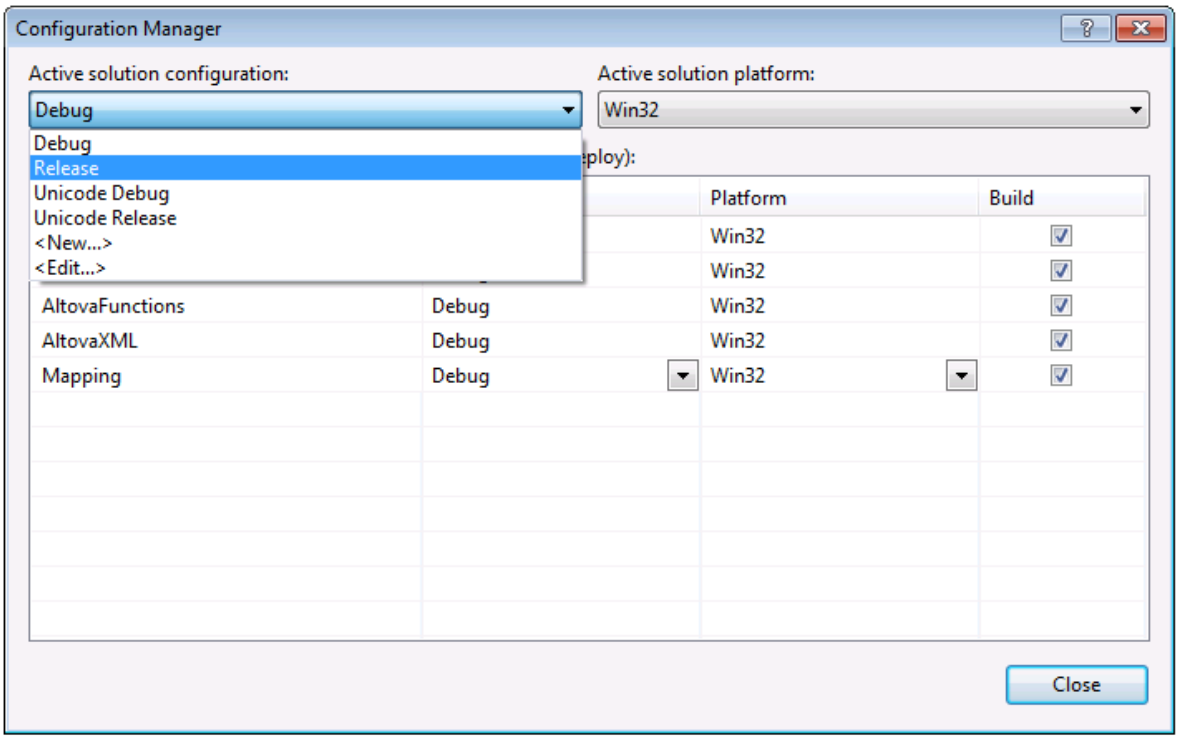
15.2.3 Compilar el proyecto

Una vez generado el código C++, el siguiente paso consiste en compilarlo en Visual Studio:

1. Abra el archivo de solución generado (.sln) en Visual Studio.

El nombre predeterminado del archivo de solución es **Mapping.sln** y su ubicación es el subdirectorio **Mapping** relativo al directorio donde guardó el código generado. Si cambió el nombre de la aplicación desde la configuración de la asignación de datos, entonces el nombre del archivo .sln también se modificó de la forma correspondiente. Por ejemplo, si cambió el nombre de la aplicación por **MiAplicación**, entonces el archivo de solución se llamará **MiAplicación.sln** y estará ubicado en el subdirectorio **MiAplicación**.

2. En el menú **Compilar** haga clic en **Administrador de configuración**.



3. Seleccione la configuración de compilación correspondiente (**Depuración**, **Versión**, **Depuración Unicode**, **Versión Unicode**). Recuerde que solamente las opciones Unicode admiten el conjunto de caracteres Unicode completo en XML y en otros archivos. Las opciones no Unicode trabajan con la página de código local de su instalación de Windows.
4. En el menú **Compilar** haga clic en **Compilar solución**.

15.2.4 Ejecutar la aplicación

Cuando termine de compilar el proyecto de Visual Studio, se genera una aplicación de la línea de comandos llamada **Mapping.exe**. (Recuerde que si cambió el nombre de la aplicación en la configuración de la asignación de datos, el nombre del ejecutable también se modificó consecuentemente.)

 © 2016-2022 Altova GmbH

Podrá encontrar la aplicación de asignación de datos en uno de estos subdirectorios relativos al archivo .sln (dependiendo de la opción de compilación elegida):

- Debug
- Release
- Unicode Debug
- Unicode Release

Para ejecutar la aplicación abra un símbolo del sistema, cambie el directorio actual por la ruta de acceso del ejecutable y ejecútelo. Por ejemplo:

```
C:\codegen\DB_CompletePOcpp\Mapping\Debug>Mapping.exe
Mapping Application
Finished

C:\codegen\DB_CompletePOcpp\Mapping\Debug>_
```

15.3 Generar código C#

Puede código en .NET C# para cualquier asignación o proyecto de asignación en el que todos los componentes de asignación sean compatibles con C#. Para ver un resumen de los componentes compatibles con cada uno de los lenguajes, C# incluido, consulte Funciones compatibles en el código generado 1482. Tenga en cuenta que MapForce suele avisar con notificaciones o mensajes de advertencia si intenta añadir componentes que no son compatibles con un lenguaje en concreto.

Puede generar código en C# no solo para la plataforma .NET Framework, sino también para .NET Core 3.1, .NET 5.0 y .NET 6.0. Si escoge esta última, el resultado es una aplicación .NET (Core) de escritorio multiplataforma que se puede ejecutar en Windows, macOS y Linux.

A la hora de generar código tenga en cuenta:

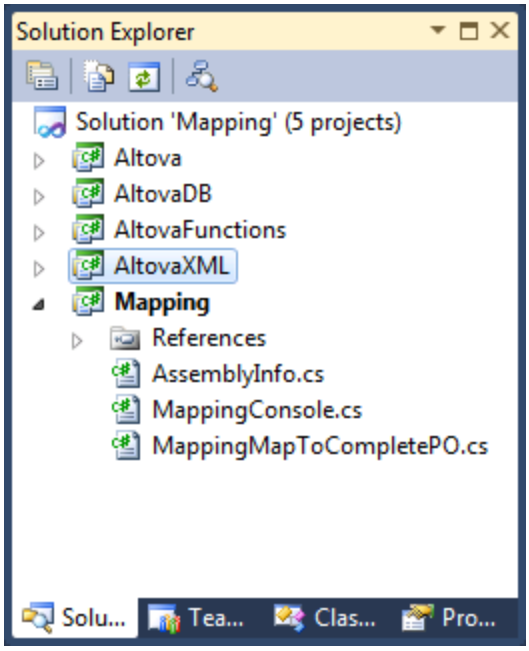
- algunas características de MapForce no son compatibles con el código de programa generado. Para más información consulte Funciones compatibles en el código generado 1482.
- puede generar código a partir de un diseño de asignación (.mfd) o de un proyecto de asignación de datos (.mfp). Si genera código a partir de una asignación de datos, la aplicación resultante ejecuta la transformación de asignación de datos correspondiente. Por el contrario, si genera código a partir de un proyecto de MapForce (.mfp) que incluya varias asignaciones de datos, la aplicación resultante ejecuta en bloque todas las asignaciones que incluye el proyecto.
- las opciones generales de generación de código se pueden cambiar haciendo clic en el comando de menú **Herramientas | Opciones** y después en la pestaña *Generación*. Consulte Opciones del generador de código 1207.
- puede cambiar el nombre de la aplicación que se genera y otras opciones de configuración desde el comando de menú **Archivo | Configurar asignación**. El nombre predeterminado de la aplicación es **Mapping**. Consulte también Configuración de la asignación 115.
- para más información sobre las rutas ene l código generado consulte el apartado Rutas de acceso según el entorno de ejecución 84.
- si la asignación de datos contiene componentes de base de datos, puede ver determinadas opciones de la base de datos haciendo clic en el componente y seleccionando el comando de menú **Componente | Propiedades**. Consulte Configurar componentes de BD 278.
- la compatibilidad para las conexiones de BD varía según la plataforma y existen tipos de conexiones que no son compatibles con todas ellas. Si quiere conectar una asignación a una BD, elija una conexión de BD que sea compatible con el entorno de destino para el que genera el código. Para más detalles consulte Asignaciones de BD en distintos entornos de ejecución 160.

Si su asignación usa componentes de BD asegúrese de crear la conexión de BD con controladores de BD compatibles con la plataforma en la que quiere ejecutar el programa generado. Por ejemplo, las conexiones de BD ADO y algunos controladores ODBC no con compatibles con Linux.

Una solución generada con MapForce suele incluir:

- Archivos de solución (.sln) y de proyecto (.csproj) que se puedan abrir con Visual Studio.
- Varias bibliotecas de Altova necesarias para la aplicación (todas llevan el prefijo **Altova**).
- El proyecto principal de asignación (en este ejemplo, **Mapping**), que incluye la aplicación de la asignación y los archivos dependientes.

 © 2016-2022 Altova GmbH

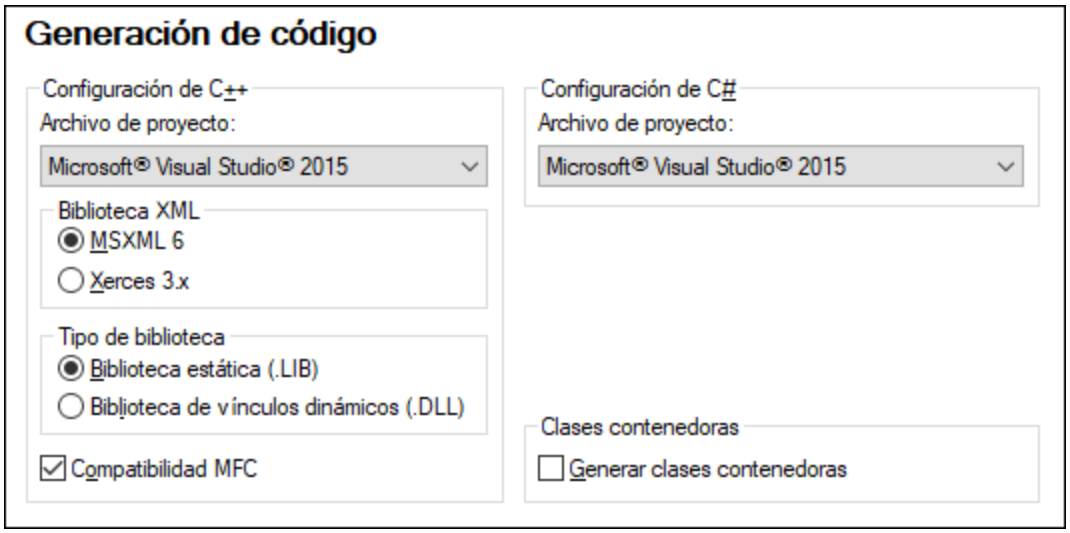


Solución C# de muestra generada con MapForce

15.3.1 Seleccionar la plataforma de destino

Antes de generar código en C# puede definir la plataforma de destino a nivel programático:

1. En el menú **Herramientas** haga clic en **Opciones**.
2. Haga clic en la pestaña *Generación de código*.
3. En el grupo Configuración de C# seleccione un valor de la lista desplegable según corresponda.



Seleccione la opción **Microsoft .NET Core 3.1**, **Microsoft .NET 5.0** o **Microsoft .NET 6.0** para generar una solución de Visual Studio que apunte a esas plataformas. Si quiere que la plataforma **.NET Framework** apunte a una versión concreta de Visual Studio seleccione cualquiera de las opciones **Microsoft Visual Studio 2010-2019** (en este caso, la solución generada apuntará a la versión de .NET Framework que corresponda a esa versión de Visual Studio).

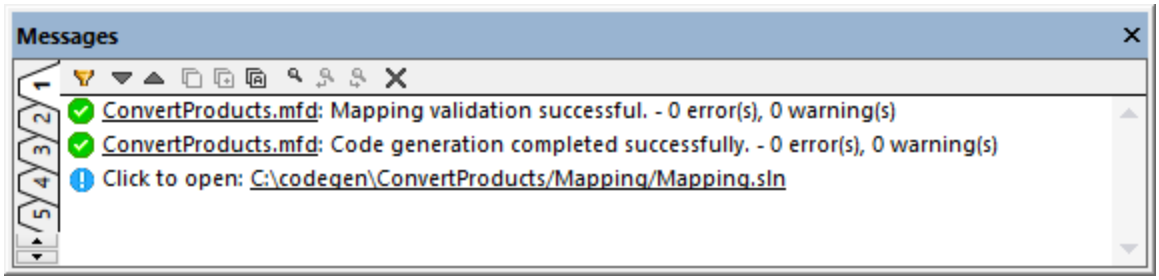
Si marca la casilla **Generar clases contenedoras** se crean más proyectos en el código generado, uno por cada esquema XML o componente DTD de la asignación. Los proyectos contenedores que se generen permiten manipular de forma programática las instancias XML correspondientes fuera de la asignación. Las bibliotecas contenedoras son opcionales e independientes del programa de asignación, aunque puede usarlas para este si lo necesita. Para más información sobre las clases contenedoras consulte Generar código fuente a partir de esquemas XML o DTD 1110.

15.3.2 Generar código a partir de una asignación de datos

Para generar código C# a partir de un archivo de diseño de asignación (.mfd):

1. Si no lo ha hecho ya, cree una asignación nueva o abra en MapForce una que ya exista.
2. Seleccione C# como lenguaje de transformación de la asignación. Para ello haga clic en el botón de la barra de herramientas o seleccione el comando de menú **Resultados | C# (Sharp)**.
3. Configure o cambie las opciones de generación de código en C# como se explica en Seleccionar la plataforma de destino 1077.
4. Seleccione el comando de menú **Archivo | Generar código en | C# (Sharp)**. Ahora la aplicación le pedirá que seleccione un directorio de destino.
5. Navegue hasta el directorio que prefiera (por ejemplo "C:\codegen\ConvertProducts") y haga clic en **Aceptar**.

El resultado de la generación de código aparece en la ventana resultados, por ejemplo:

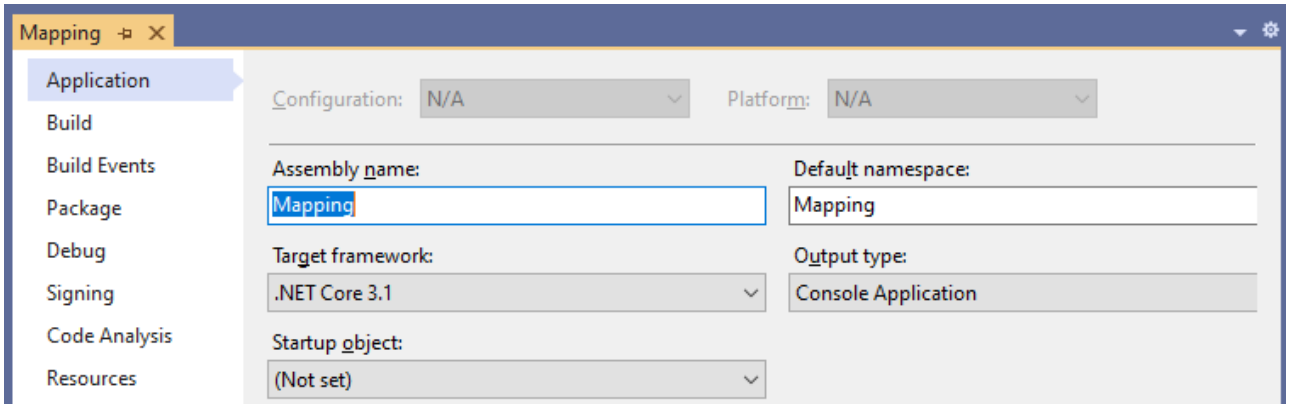


También puede hacer clic en la ventana Mensajes para abrir la solución generada en Visual Studio.

Tenga en cuenta que si elige generar código para las plataformas .NET 5.0, .NET 6.0 o .NET Core necesita tener los SDKs correspondientes y una versión compatible de Visual Studio. Para saber qué paquete debe descargar para su sistema operativo y plataforma consulte la página de Microsoft (https://dotnet.microsoft.com/download).

Para comprobar de nuevo el marco de destino del proyecto desde Visual Studio haga clic con el botón derecho en el proyecto "Mapping" en el explorador de soluciones, seleccione Properties y observe el valor que aparece en la lista desplegable **Target Framework**, por ejemplo:

 © 2016-2022 Altova GmbH



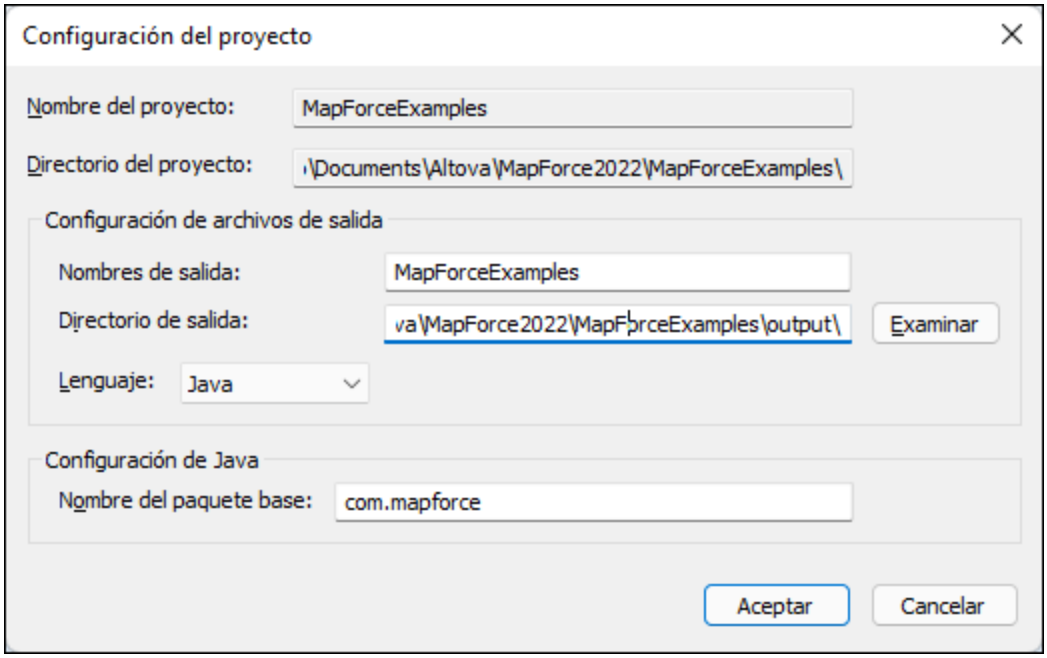
El nombre predeterminado de la aplicación generada es **Mapping**. Puede cambiar este nombre y otras opciones antes de generar el código desde el cuadro de diálogo "Configurar asignación" 115.

15.3.3 Generar código a partir de un proyecto de asignación

Puede generar código C# a partir de proyectos de asignación (.mfp) que consistan en varios archivos de diseño de asignación (.mfd). Para ello todos los archivos del proyecto deben ser compatibles con C#, es decir, todos sus componentes deben serlo también, como se explica en Funciones compatibles en el código generado 1482.

Para generar código a partir de un proyecto de asignación de datos (.mfp):

1. Si todavía no lo ha hecho, abra el proyecto de asignación de datos en MapForce.
2. Haga clic con el botón derecho en la ventana Proyecto y después elija el comando **Propiedades**.



3. Revise y modifique la configuración del proyecto según proceda (sobre todo debe comprobar que el

lenguaje de destino y el directorio de salida están configurados correctamente). Después haga clic en **Aceptar**.

4. En el menú **Proyecto** haga clic en el comando **Generar código para todo el proyecto**.

El progreso y el resultado del proceso de generación de código (con mensajes de error o éxito) aparece reflejado en la ventana Mensajes.

La aplicación generada recibe el mismo nombre que el nombre del proyecto, pero si el nombre del proyecto contiene espacios, en el nombre de la aplicación los espacios se convierten en caracteres de subrayado. El código se genera por defecto en el directorio donde está ubicado el proyecto de MapForce, en el subdirectorio **output**.

Para cambiar el directorio de salida y el nombre del proyecto haga clic en el proyecto en la ventana Proyecto y después seleccione el comando de menú **Proyecto | Propiedades**. Si el proyecto de MapForce tiene carpetas, podrá cambiar las opciones de generación de código para cada una de las carpetas por separado (clic con el botón derecho en una carpeta y después Propiedades en el menú contextual). Todas las carpetas del proyecto heredan por defecto las opciones de generación de código del proyecto de MapForce.

15.3.4 Compilar el proyecto

Una vez generado el código C#, el siguiente paso consiste en compilarlo en Visual Studio:

1. Abra el archivo de solución generado (.sln) en Visual Studio.

El nombre predeterminado del archivo de solución es **Mapping.sln** y su ubicación es el subdirectorio **Mapping** relativo al directorio donde guardó el código generado. Si cambió el nombre de la aplicación desde la configuración de la asignación de datos, entonces el nombre del archivo .sln también se modificó de la forma correspondiente. Por ejemplo, si cambió el nombre de la aplicación por **MiAplicación**, entonces el archivo de solución se llamará **MiAplicación.sln** y estará ubicado en el subdirectorio **MiAplicación**.

2. En el menú **Compilar** haga clic en **Administrador de configuración**.
3. Seleccione la configuración de compilación correspondiente (**Depuración**, **Versión**).
4. En el menú **Compilar** haga clic en **Compilar solución**.

Ejecutar la aplicación

Cuando termine de compilar el proyecto de Visual Studio, se genera una aplicación de la línea de comandos llamada **Mapping.exe**. (Recuerde que si cambió el nombre de la aplicación en la configuración de la asignación de datos 115, el nombre del ejecutable también se modificó consecuentemente.)

Podrá encontrar la aplicación de asignación de datos en uno de estos subdirectorios relativos al archivo .sln (dependiendo de la opción de compilación elegida):

- bin\Debug
- bin\Release

Para ejecutar la aplicación abra un símbolo del sistema, cambie el directorio actual por la ruta de acceso del ejecutable y ejecútelo. Por ejemplo:

 © 2016-2022 Altova GmbH

```
C:\codegen\DB_CompletePOcs\Mapping\bin\Release>Mapping.exe
Mapping Application
Connecting to CustomersAndArticles database...

Finished

C:\codegen\DB_CompletePOcs\Mapping\bin\Release>
```

15.3.5 Compilar proyectos .NET 5 y .NET Core

Para compilar y ejecutar una solución generada para .NET Core 3.1, .NET 5.0 o NET 6.0 debe tener instalado el SDK correspondiente. Para saber qué paquete debe descargar para su sistema operativo y plataforma consulte el sitio web de Microsoft (https://dotnet.microsoft.com/download). Una vez haya descargado e instalado el paquete correspondiente puede visualizar fácilmente el SDK instalado tecleando este comando en la línea de comandos:

```
dotnet --list-sdks
```

Si tiene el SDK de .NET Core 3.1, .NET 5.0 o NET 6.0 puede compilar el código generado, bien desde la línea de comandos o en Visual Studio. Si está usando Visual Studio, tenga en cuenta que en el momento en que se redacta esta documentación (diciembre de 2020) los SDK de .NET Core 3.1, .NET 5.0 o NET 6.0 son compatibles con Visual Studio 2019 (v16.7).

Compilar desde la línea de comandos

Para compilar el código generado desde la línea de comandos:

1. Abra una ventana de la línea de comandos y cambie el directorio por el que contiene el código generado. Por ejemplo, si seleccionó **C:\codegen\ConvertProducts** como directorio de destino y la configuración predeterminada de la asignación ejecute:

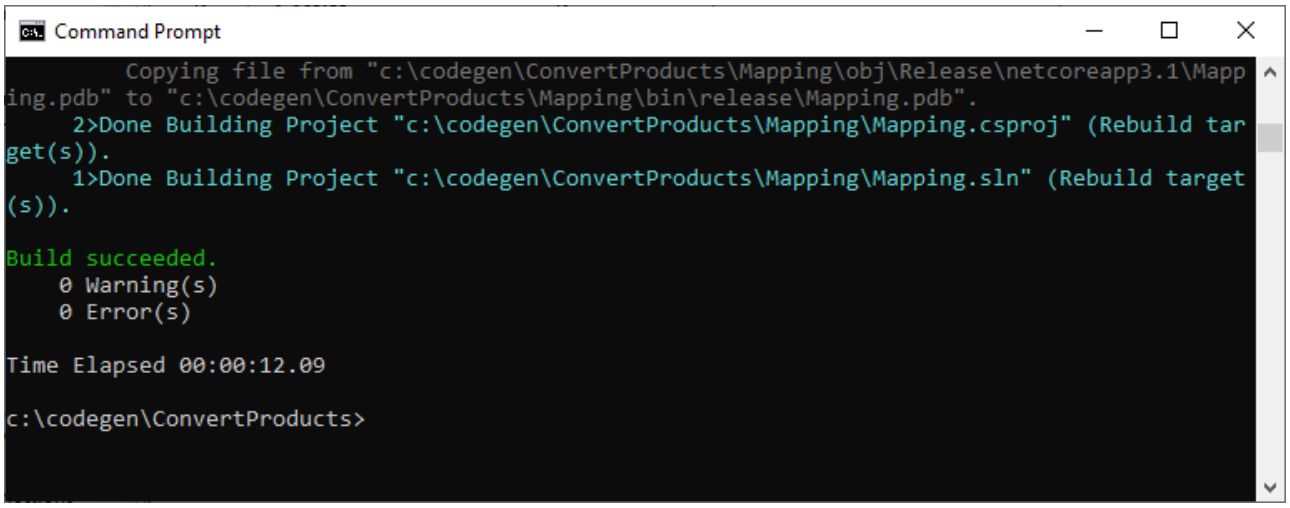
   ```
   cd C:\codegen\ConvertProducts
   ```

2. Ejecute este comando:

   ```
   dotnet build Mapping\Mapping.sln --no-incremental --configuration Release --
   verbosity normal --output Mapping\bin\release
   ```

Nota: la ruta de acceso a la solución de la asignación y su nombre pueden ser distintos si cambió la propiedad Nombre de la aplicación en el cuadro de diálogo "Cambiar configuración de la asignación" 115.

El resultado si el código se compila correctamente debe parecerse a este:



Compilar con Visual Studio

Para compilar el código generado en Visual Studio:

1. Haga doble clic en **Mapping.sln** (o el nombre de solución personalizado que corresponda) para abrirlo en Visual Studio.
2. En el menú **Build** haga clic en **Configuration Manager**.
3. Seleccione la configuración de compilación que necesite (Debug, Release) y la plataforma de la solución (x64, x86, Any CPU).
4. En el menú **Build** seleccione **Build Solution (Ctrl+Shift+B)**.

Ejecutar el programa generado

Una vez haya compilado el código generado aparece un **Mapping.exe** (o un programa con un nombre personalizado, según la configuración) en el directorio de resultados del proyecto. La ruta de ese directorio depende de la configuración de compilación que escogiera antes (Debug, Release).

Si compila el código generado en Linux, el ejecutable que se genera se llamará simplemente **Mapping**, sin extensión.

En Windows puede ejecutar el programa haciendo doble clic en el ejecutable en el explorador de Windows o llamando al ejecutable desde una ventana de la línea de comandos. En otras plataformas deberá llamar al ejecutable desde la línea de comandos con un comando como este:

```
./Mapping
```

Si la asignación hace referencia a archivos de entrada o de salida en disco, el programa solo se ejecutará correctamente si se pueden resolver las rutas. Por ejemplo, si configuró la aplicación para que un componente busque un archivo de entrada en el mismo directorio con una ruta relativa, entonces ese archivo debe estar presente en el directorio de ejecución del programa. Para más información sobre cómo gestionar rutas de acceso al diseñar una asignación consulte Usar rutas de acceso absolutas y relativas 82.

 © 2016-2022 Altova GmbH

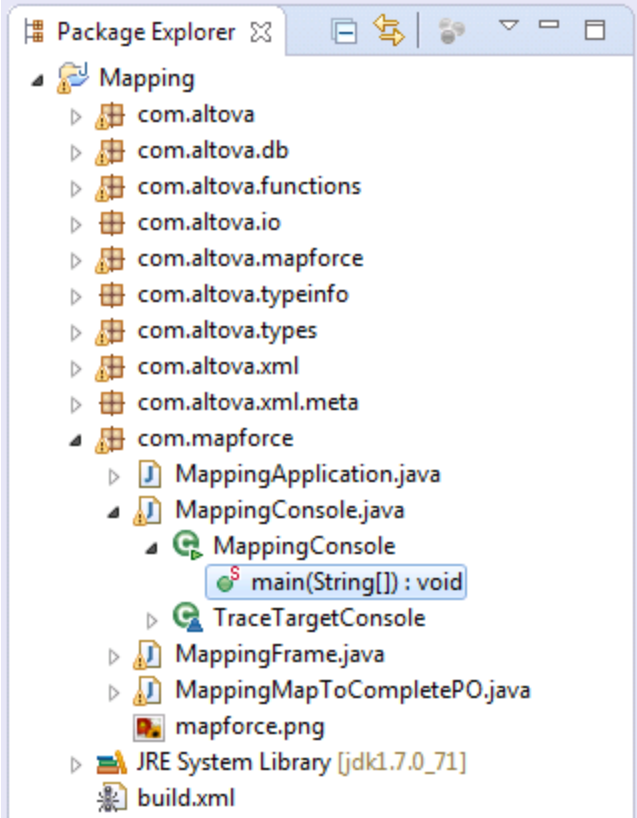
15.4 Generar código Java

Puede generar código de programa para Java Java SE JDK 8 u 11 (incluido OpenJDK). A la hora de generar código debe tener en cuenta que:

- algunas características de MapForce no son compatibles con el código de programa generado. Para más información consulte Funciones compatibles en el código generado 1482.
- puede generar código a partir de un diseño de asignación (.mfd) o de un proyecto de asignación de datos (.mfp). Si genera código a partir de una asignación de datos, la aplicación resultante ejecuta la transformación de asignación de datos correspondiente. Por el contrario, si genera código a partir de un proyecto de MapForce (.mfp) que incluya varias asignaciones de datos, la aplicación resultante ejecuta en bloque todas las asignaciones que incluye el proyecto.
- las opciones generales de generación de código se pueden cambiar haciendo clic en el comando de menú **Herramientas | Opciones** y después en la pestaña *Generación*. Consulte Opciones del generador de código 1207.
- puede cambiar el nombre de la aplicación que se genera y otras opciones de configuración desde el comando de menú **Archivo | Configurar asignación**. El nombre predeterminado de la aplicación es **Mapping**. Consulte también Configuración de la asignación 115.
- para más información sobre las rutas ene l código generado consulte el apartado Rutas de acceso según el entorno de ejecución 84.
- si la asignación de datos contiene componentes de base de datos, puede ver determinadas opciones de la base de datos haciendo clic en el componente y seleccionando el comando de menú **Componente | Propiedades**. Consulte Configurar componentes de BD 278.
- la compatibilidad para las conexiones de BD varía según la plataforma y existen tipos de conexiones que no son compatibles con todas ellas. Si quiere conectar una asignación a una BD, elija una conexión de BD que sea compatible con el entorno de destino para el que genera el código. Para más detalles consulte Asignaciones de BD en distintos entornos de ejecución 160.

Los proyectos Java generados en MapForce suelen incluir:

- Varios paquetes Java firmados por Altova que la asignación de datos necesita (todos empiezan con el prefijo **com.altova**).
- El paquete **com.mapforce**, que incluye la aplicación de asignación de datos y sus archivos independientes como puede ver más abajo el nombre de este paquete se puede cambiar). Los dos archivos más importantes de este paquete son:
  - la aplicación de asignación de datos Java como aplicación de diálogo (**MappingApplication.java**) y
  - la aplicación de asignación de datos Java como aplicación de consola (**MappingConsole.java**).
- Un archivo **build.xml** que se puede ejecutar con Apache Ant para compilar el proyecto y generar archivos JAR.



Aplicación Java de muestra generada con MapForce (en entorno Eclipse)

Esta sección incluye varios apartados:

- Generar código a partir de una asignación de datos 1084
- Generar código a partir de un proyecto de asignación de datos 1085
- Control de referencias JDBC 1087
- Compilar el proyecto con Ant 1086
- Ejemplo: ejecutar y compilar código Java con Eclipse y Ant 1089

15.4.1 Generar código a partir de una asignación de datos

Para generar código Java a partir de un archivo de diseño de asignación (.mfd):

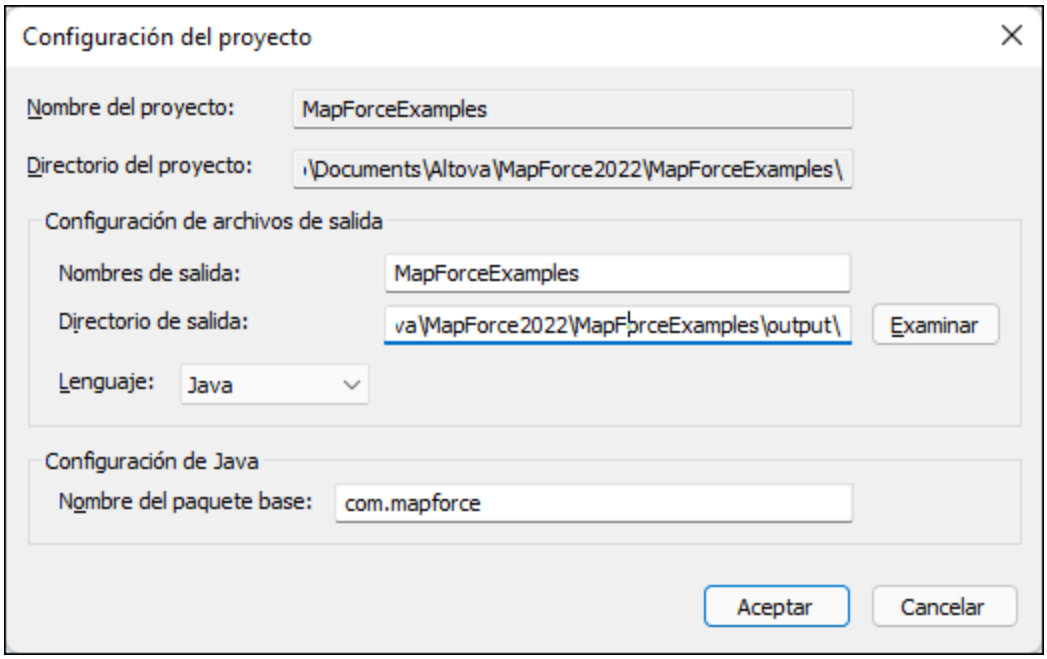
1. Revise y configure las opciones de generación de código (véase Opciones del generador de código 1207).
2. En el menú Archivo haga clic en Generar código en | Java.
3. Seleccione un directorio de destino para los archivos generados y haga clic en Aceptar para confirmar. El resultado de la generación de código (mensaje de éxito o error) aparece en la ventana Mensajes.

El nombre predeterminado de la aplicación generada es **Asignación** y el nombre predeterminado del paquete base es **com.mapforce**. Puede cambiar estas opciones predeterminadas en el cuadro de diálogo "Configurar asignación" (*véase Cambiar configuración de la asignación* 115).

15.4.2 Generar código a partir de un proyecto de asignación

Para generar código a partir de un proyecto de asignación de datos (.mfp):

1. Si todavía no lo ha hecho, abra el proyecto de asignación de datos en MapForce.
2. Haga clic con el botón derecho en la ventana Proyecto y después elija el comando **Propiedades**.



3. Revise y modifique la configuración del proyecto según proceda (sobre todo debe comprobar que el lenguaje de destino y el directorio de salida están configurados correctamente). Después haga clic en **Aceptar**.
4. En el menú **Proyecto** haga clic en el comando **Generar código para todo el proyecto**.

El progreso y el resultado del proceso de generación de código (con mensajes de error o éxito) aparece reflejado en la ventana Mensajes.

La aplicación generada recibe el mismo nombre que el nombre del proyecto, pero si el nombre del proyecto contiene espacios, en el nombre de la aplicación los espacios se convierten en caracteres de subrayado. El código se genera por defecto en el directorio donde está ubicado el proyecto de MapForce, en el subdirectorio **output**.

Para cambiar el directorio de salida y el nombre del proyecto haga clic en el proyecto en la ventana Proyecto y después seleccione el comando de menú **Proyecto | Propiedades**. Si el proyecto de MapForce tiene carpetas, podrá cambiar las opciones de generación de código para cada una de las carpetas por separado (clic con el botón derecho en una carpeta y después Propiedades en el menú contextual). Todas las carpetas del proyecto heredan por defecto las opciones de generación de código del proyecto de MapForce.

15.4.3 Compilar el proyecto con Ant

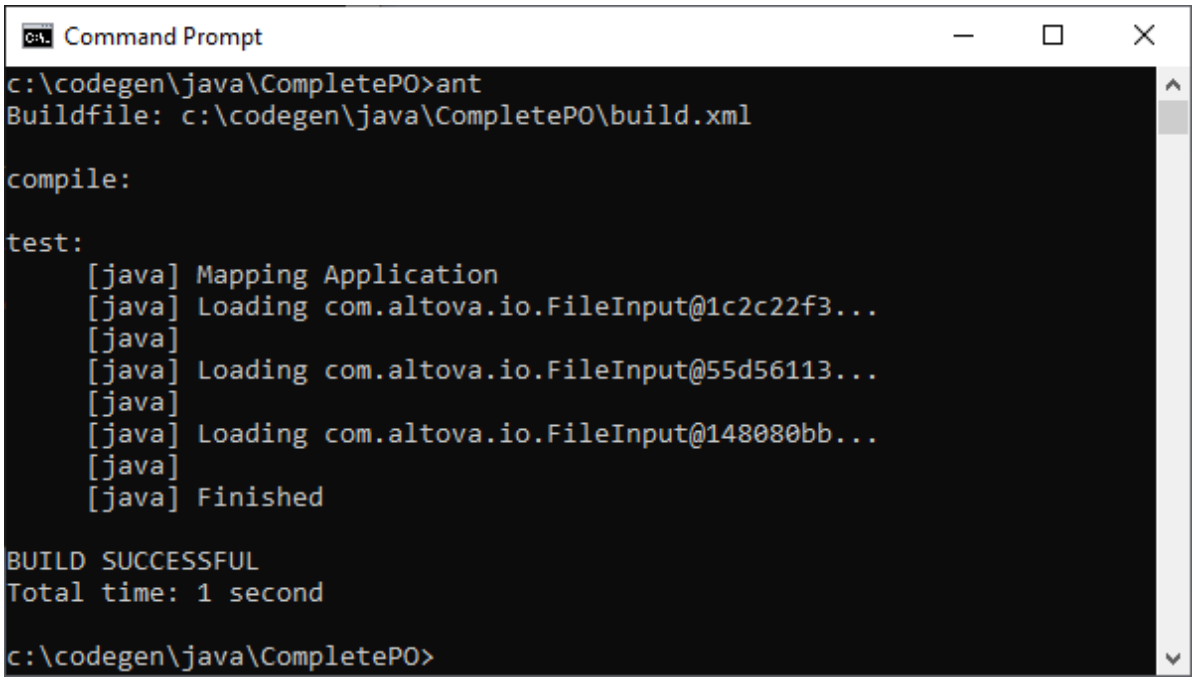
Apache Ant es una popular biblioteca Java (y herramienta de la línea de comandos) que automatiza la compilación de proyectos Java (*véase http://ant.apache.org/manual/*). Ant trabaja con los archivos de compilación, que definen el origen y el destino de la compilación de código y opciones de compilación concretas. Los proyectos generados con MapForce incluyen un archivo llamado **build.xml** que Ant puede reconocer, de modo que puede usar Ant para compilar proyectos generados con MapForce fácilmente.

Ant puede estar disponible en su sistema o estar integrado con Eclipse (o con otros entornos IDE Java). El sitio web http://ant.apache.org/manual/ incluye instrucciones para instalar Ant en el sistema. Para aprender a usar Ant en Eclipse puede consultar la documentación de Eclipse.

Para saber si en su sistema existe una versión independiente de Ant (y no la versión integrada en Eclipse) abra un símbolo del sistema y teclee `ant` en la línea de comandos. Si Ant ya está disponible, el mensaje será parecido a este: `Buildfile: buildxml does not exist!` Este mensaje indica que Ant está instalado y está intentando compilar un archivo **build.xml** pero dicho archivo no existe en el directorio actual. Si ejecuta Ant desde un directorio que contenga un archivo **build.xml**, Ant ejecutará el archivo **build.xml** con las opciones de compilación definidas en él.

Para compilar un proyecto Java generado con MapForce con Ant:

1. Abra un símbolo del sistema y navegue hasta el directorio donde se generó el proyecto Java (recuerde que el directorio debe contener el archivo **build.xml**).
2. En el símbolo del sistema introduzca `ant`. Este comando compilará y ejecutará el código Java de acuerdo con las opciones de compilación definidas en el archivo **build.xml**.



Para generar un archivo JAR con Ant:

 © 2016-2022 Altova GmbH

- En el símbolo del sistema introduzca `ant jar`.

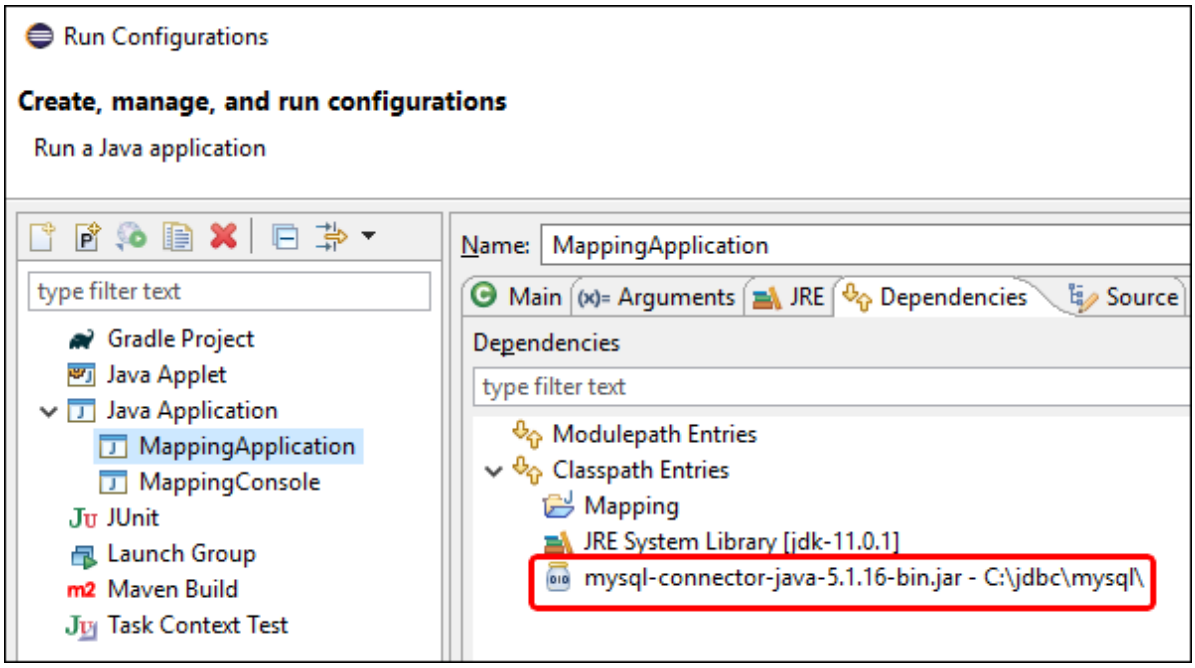
Si necesita ayuda con la sintaxis y las opciones de los comandos de Ant, introduzca `ant -help` en la línea de comandos.

15.4.4 Control de referencias JDBC

Si la asignación se conecta a una base de datos por JDBC, compruebe que los controladores JDBC que utiliza la asignación están instalados en el sistema (*véase Crear una conexión JDBC*). Para ver la configuración JDBC actual de cualquier componente de base de datos en MapForce haga clic en el componente y después seleccione el comando de menú **Componente | Propiedades**.

Si hace una prueba de ejecución de la aplicación Java que ha generado
Si ejecuta en Eclipse la aplicación Java que ha generado para probarla, puede que tenga que añadir el controlador JDBC como entrada "classpath" en la configuración actual de ejecución. De lo contrario la ejecución de prueba en Eclipse podría generar un mensaje de error como: `java.lang.ClassNotFoundException: com.mysql.jdbc.Driver`. Si tomamos como ejemplo la BD de MySQL, puede añadir el controlador JDBC como dependencia en eclipse siguiendo estos pasos:

1. Genere el código Java desde MapForce e importe el proyecto en Eclipse usando el comando de menú **Archivo | Importar | Proyectos existentes en el área de trabajo**.
2. En el menú Ejecutar haga clic en **Ejecutar configuraciones**. En la parte izquierda del cuadro de diálogo, bajo "Aplicación Java", hay dos configuraciones llamadas **MappingApplication** y **MappingConsole**.
3. Haga clic en la configuración que necesite (por ejemplo, **MappingApplication**).
4. En la pestaña *Dependencias* haga clic en **Entradas classpath** y después en **Añadir JARs externas**.
5. Navegue hasta el archivo .jar del controlador JDBC, por ejemplo:



6. Haga clic en **Ejecutar** para ejecutar el programa con el controlador JDBC de BD añadido como dependencia.

Si aparece el mensaje de error mencionado más arriba al ejecutar el archivo de Ant **build.xml**, añada el controlador JDBC a la classpath de la tarea "test". El siguiente fragmento de código es un ejemplo de una tarea "test" de Ant que incluye una referencia al archivo .jar del controlador JDBC.

```
<target name="test" depends="compile">
   <java classpath="C:\codegen\java\mysql_mapping"
classname="com.mapforce.MappingConsole" fork="true" failonerror="true">
      <classpath>
         <pathelement path="${classpath}"/>
         <pathelement location="C:\jdbc\mysql\mysql-connector-java-5.1.16-bin.jar"/>
      </classpath>
      <arg line="${cmdline}"/>
   </java>
</target>
```

Nota: el ejemplo anterior asume que el archivo .jar existe y está en la ubicación **C:\jdbc\mysql\mysql-connector-java-5.1.16-bin.jar**. Asegúrese de cambiar la ruta del archivo .jar como corresponda a su entorno.

Incluir controladores JDBC en el manifiesto de la aplicación

Si construye archivos JAR a partir de la aplicación Java generada, añada una referencia al controlador de la BD en la sección "manifiesto" del archivo **build.xml**. Esto garantiza que la referencia al controlador de BD está disponible en el archivo del manifiesto (**MANIFEST.MF**) una vez haya creado el proyecto.

Para añadir la referencia de BD al archivo del manifiesto:

1. Localice el elemento `manifest` en el archivo **build.xml**.
2. Añada un elemento nuevo llamado `attribute` donde el atributo "name" sea "Class-Path" y el atributo de "value" sea el nombre del archivo .jar. Por ejemplo, para MySQL 5.1.16 el elemento nuevo tendría este aspecto:

   ```
   <attribute name="Class-Path" value="mysql-connector-java-5.1.16-bin.jar"/>
   ```

   El elemento `manifest` del archivo **build.xml** ahora tiene este aspecto:

   ```
   <manifest file="C:\codegen\java\mysql_mapping/META-INF/MANIFEST.MF"
   mode="replace">
      <attribute name="Created-By" value="MapForce 2023"/>
      <attribute name="Main-Class" value="com.mapforce.MappingConsole"/>
      <attribute name="Class-Path" value="mysql-connector-java-5.1.16-bin.jar"/>
   </manifest>
   ```

3. Copie el archivo JAR del controlador JDBC a la carpeta que contenga el archivo JAR de la aplicación generada.

© 2016-2022 Altova GmbH

15.4.5 Ejemplo: generar y ejecutar código Java

Este ejemplo explica paso a paso cómo se puede generar una aplicación Java con MapForce y cómo se puede compilar con el entorno Eclipse y Apache Ant. Cuando termine de seguir los pasos del ejemplo, obtendrá una aplicación Java completa compilada que ejecuta una de las asignaciones de muestra que vienen con MapForce.

Si ya sabe compilar aplicaciones Java con Eclipse y Ant, debe saber que para este ejemplo no hay ningún requisito especial aparte de estos:

- Debe tener instalado en su sistema un Java Development Kit (JDK), Eclipse y Ant. Eclipse suele incluir una versión de Ant, pero también puede instalar Ant por separado (*véase también Compilar el proyecto con Ant* 1086).
- Para ejecutar Eclipse con OpenJDK, la variable de entorno PATH suele necesitar que incluya la ruta al directorio **bin** del JDK (por ejemplo, **C:\Java\jdk-11.0.1\bin**).
- Si ejecuta una versión independiente de Apache Anta, como en este ejemplo, la variable de entorno PATH también debería incluir la ruta al directorio **bin** de Ant (por ejemplo, **C:\apache-ant-1.10.5\bin**) para que pueda ejecutar Ant sin tener que escribir toda la ruta de acceso al ejecutable en la línea de comandos.

Para este ejemplo se usó esta configuración:

- MapForce de 64 bits
- OpenJDK 11.0.1 de 64 bits
- Eclipse IDE para desarrolladores de Java, versión 2019-03 (4.11.0), de 64 bits
- Apache Ant 1.10.6 (instalado como versión independiente) y Apache Ant 1.10.5 (versión incluida en Eclipse)

Paso 1: Generar código Java

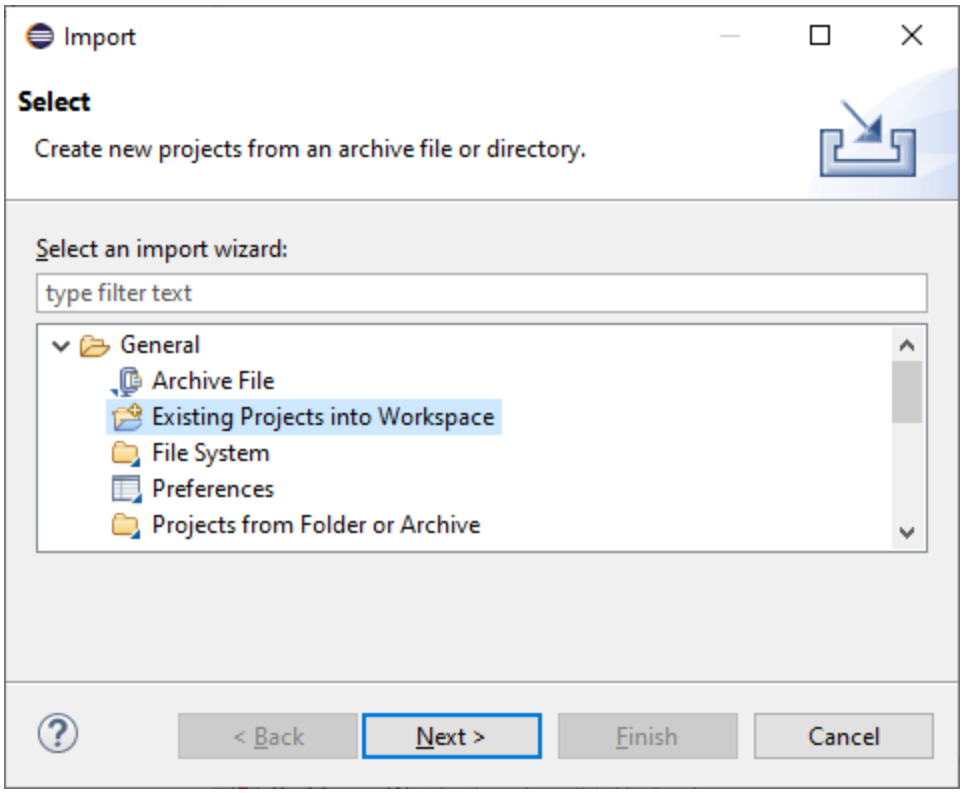
Para generar el código Java en MapForce:

1. En el menú **Archivo** haga clic en el comando **Abrir** y navegue hasta la asignación de datos **CompletePO.mfd** del directorio <Documentos>\Altova\MapForce2023\MapForceExamples\.
2. En el menú **Resultados** haga clic en el comando Java. Esto activa el lenguaje de transformación Java.
3. En el menú **Archivo** haga clic en el comando **Generar código en | Java**. Cuando la aplicación lo solicite, navegue hasta el directorio donde se debe guardar el proyecto Java (p. ej en el espacio de trabajo predeterminado de Eclipse). Para nuestro ejemplo usaremos el directorio de trabajo **C:\codegen\java\CompletePO**. Necesitaremos hacer referencia a este directorio más adelante.

Paso 2: importar el proyecto en Eclipse

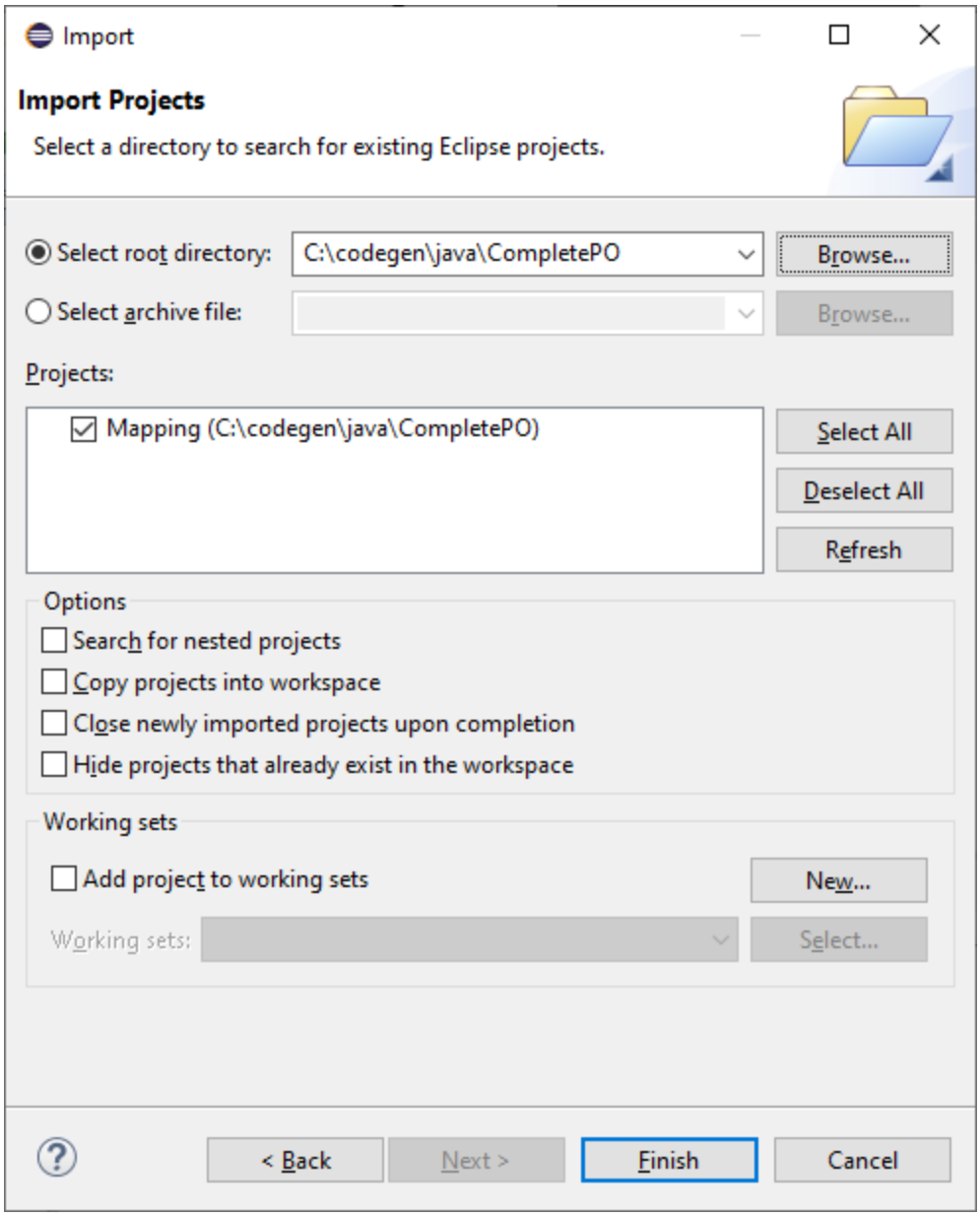
Para importar el proyecto en Eclipse:

1. Si todavía no lo ha hecho, ejecute Eclipse y cambie a la perspectiva Java predeterminada (con el comando de menú **Window | Open Perspective**).
2. En el menú **File** haga clic en el comando **Import** y después seleccione **Existing Projects into Workspace (proyectos actuales al espacio de trabajo)**.

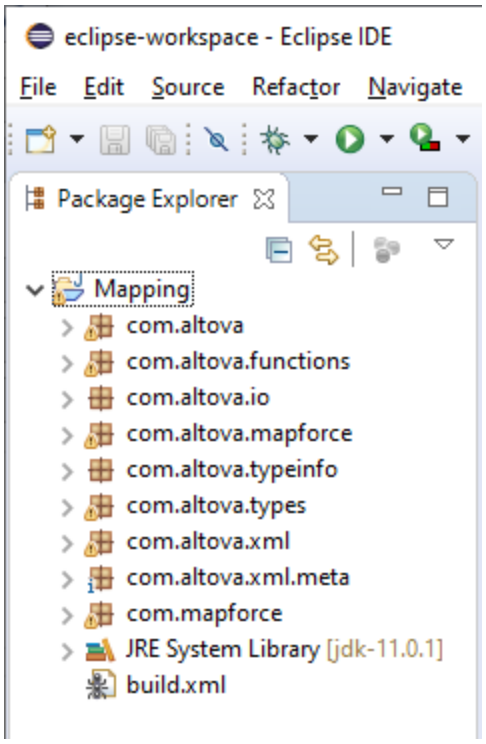


3. Haga clic en **Next para continuar**.

© 2016-2022 Altova GmbH



4. Navegue hasta la carpeta donde guardó el código generado y después haga clic en **Finish** para finalizar. El proyecto Java creado con MapForce aparece ya en la vista del explorador de paquetes *Package Explorer*. Si no puede ver la vista *Package Explorer,* puede activarla con el comando de menú **Window | Show View | Package Explorer**.

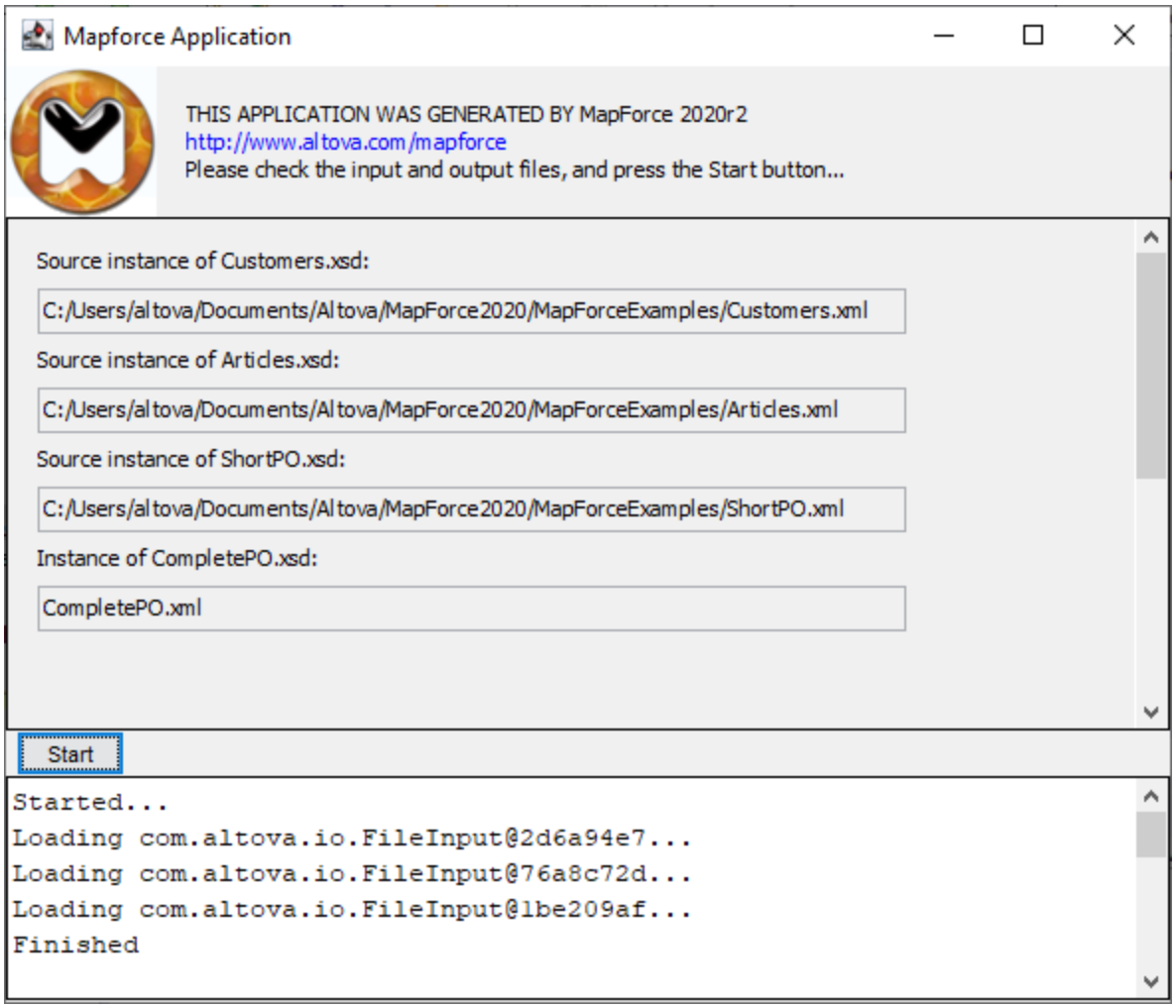


Paso 3: Ejecutar el proyecto Java como aplicación de interfaz gráfica

Para ejecutar el proyecto Java como aplicación de interfaz gráfica:

1. En la vista *Package Explorer* de Eclipse seleccione el archivo **MappingApplication.java** situado en el paquete **com.mapforce**.
2. En el menú de ejecución **Run** haga clic en el comando **Run As | Java application para ejecutar el archivo como aplicación Java**.
3. En la ventana de la aplicación MapForce que aparece haga clic en **Start** para iniciar la ejecución de la asignación de datos.

 © 2016-2022 Altova GmbH

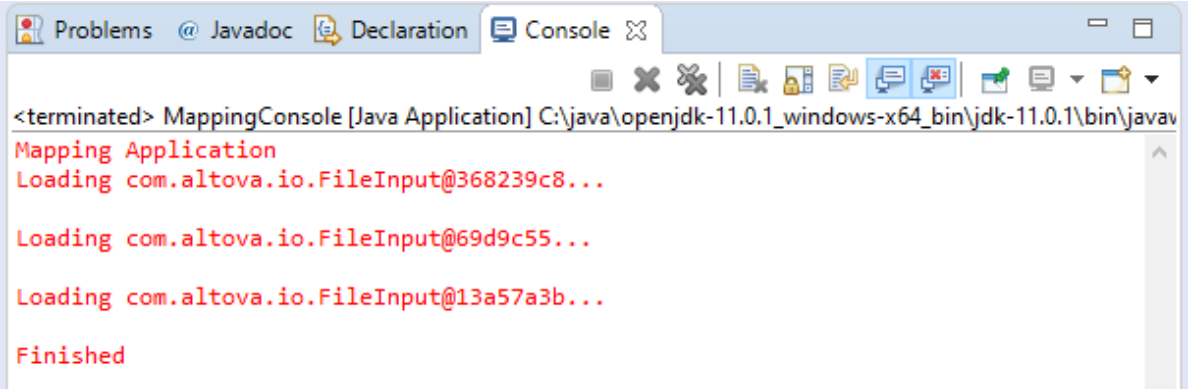


Si Eclipse detecta errores de configuración del sistema o de tiempo de ejecución, recibirá un mensaje de Eclipse (*véase* *Control de referencias JDBC* 1087). Si no se encuentran errores, la aplicación Java ejecuta la transformación de asignación de datos y genera el archivo **CompletePO.xml** en la ruta de acceso de salida (en nuestro ejemplo: **C:\workspace\CompletePO\CompletePO.xml**).

Paso 4: ejecutar el proyecto como aplicación de consola

Para ejecutar el proyecto como aplicación de consola:

1. En la vista del explorador de paquetes *Package Explorer* de Eclipse seleccione el archivo **MappingConsole.java** situado en el paquete **com.mapforce**.
2. En el menú de ejecución **Run** haga clic en el comando **Run As | Java application para ejecutar el archivo como aplicación Java**.



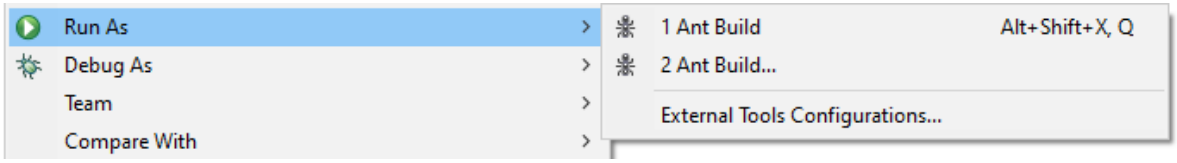
Si Eclipse detecta errores de configuración del sistema o de tiempo de ejecución, recibirá un mensaje de Eclipse (*véase* *Control de referencias JDBC* 1087). Si no se encuentran errores, la aplicación Java ejecuta la transformación de asignación de datos y genera el archivo **CompletePO.xml** en la ruta de acceso de salida (en nuestro ejemplo: **C:\workspace\CompletePO\CompletePO.xml**).

Paso 5: generar el archivo JAR

Si Apache Ant viene incluido con Eclipe puede generar el archivo JAR directamente desde el entorno de desarrollo integrado Eclipse. De lo contrario, para generar el archivo JAR en la linea de comandos fuera de Eclipse, debe instalar Apache Ant por separado.

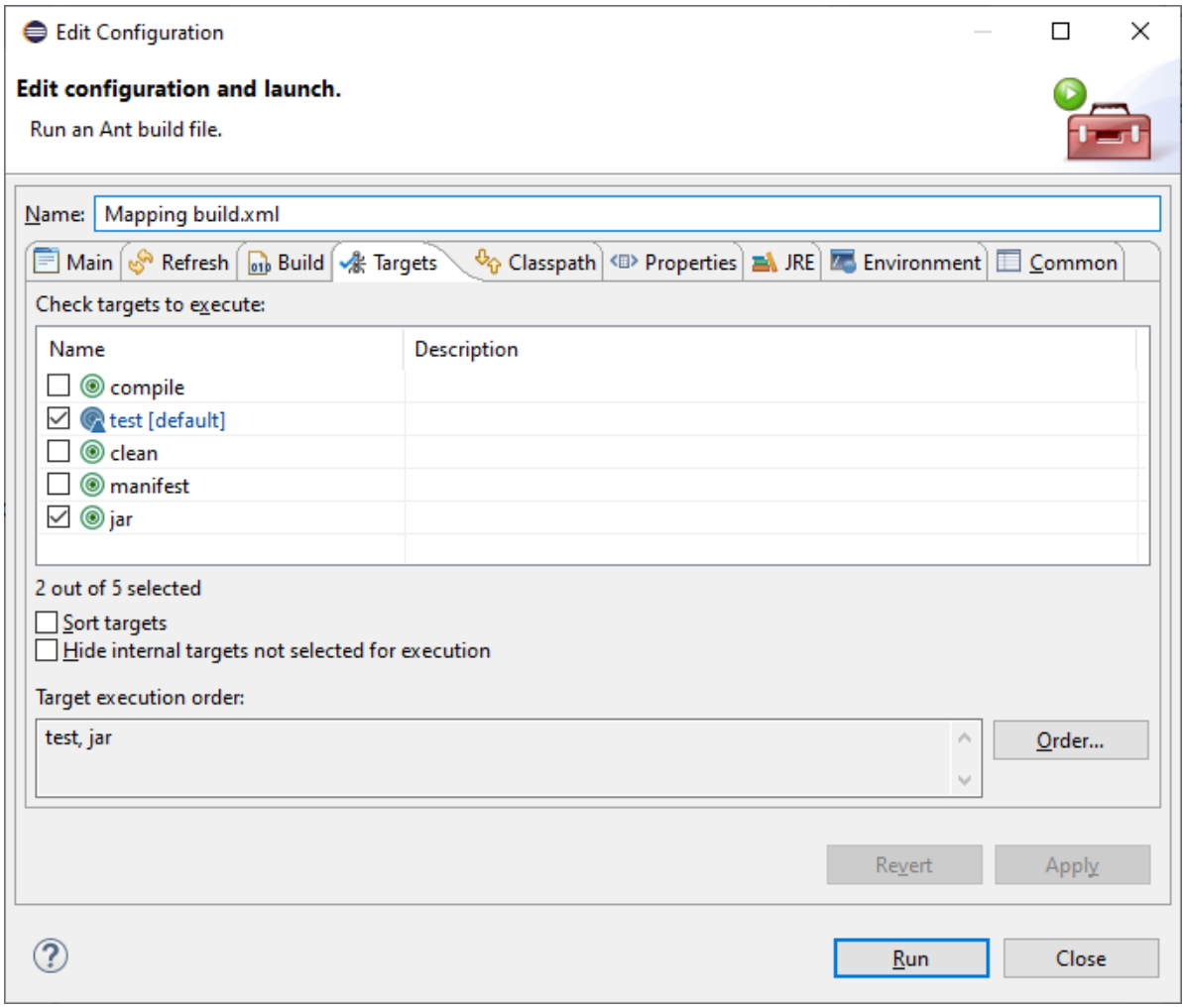
Para generar el archivo .jar con Ant en Eclipse:

1. En la vista Package Explorer de Eclipse, haga clic con el botón derecho en el archivo **build.xml** y apunte al menú **Run As**. En este menú se muestran dos opciones de ejecutar el archivo de generación de Ant.Si elige la primera, Eclipse inicia la generación con las opciones predeterminadas. Si elige la segunda opción, puede cambiar las opciones antes de iniciar la generación. Elija la segunda opción.



2. Haga clic en los componentes de destino que quiere incluir en la generación de Ant. En este ejemplo hemos seleccionado **test** y **jar**.

 © 2016-2022 Altova GmbH



3. Haga clic en **Run** (*ejecutar*). Eclipse ejecuta el archivo de generación de Ant y muestra el resultado en la vista Console.

Para generar el archivo JAR en la línea de comando:

1. Asegúrese de que el directorio en el que está instalado Apache Ant (por ejemplo, **C:\apache-ant-1.10.5\bin**) se añade a la variable de entorno PATH.
2. Abra la ventana de la línea de comandos y cambie el directorio actual por aquel en el que quiere generar el código Java (en este ejemplo, **C:\codegen\java\CompletePO**). Este es también el directorio en el que se genera el archivo **build.xml**.
3. En la línea de comandos, escriba:

```
ant clean jar
```

Ant ejecuta los destinos "clean" y "jar" del archivo build.xml (y destinos dependientes) y muestra el resultado en la línea de comandos:

Command Prompt

```
c:\codegen\java\CompletePO>ant clean jar
Buildfile: c:\codegen\java\CompletePO\build.xml

clean:

compile:
    [javac] Compiling 94 source files

manifest:

jar:
      [jar] Building jar: c:\codegen\java\CompletePO\Mapping.jar

BUILD SUCCESSFUL
Total time: 3 seconds

c:\codegen\java\CompletePO>
```

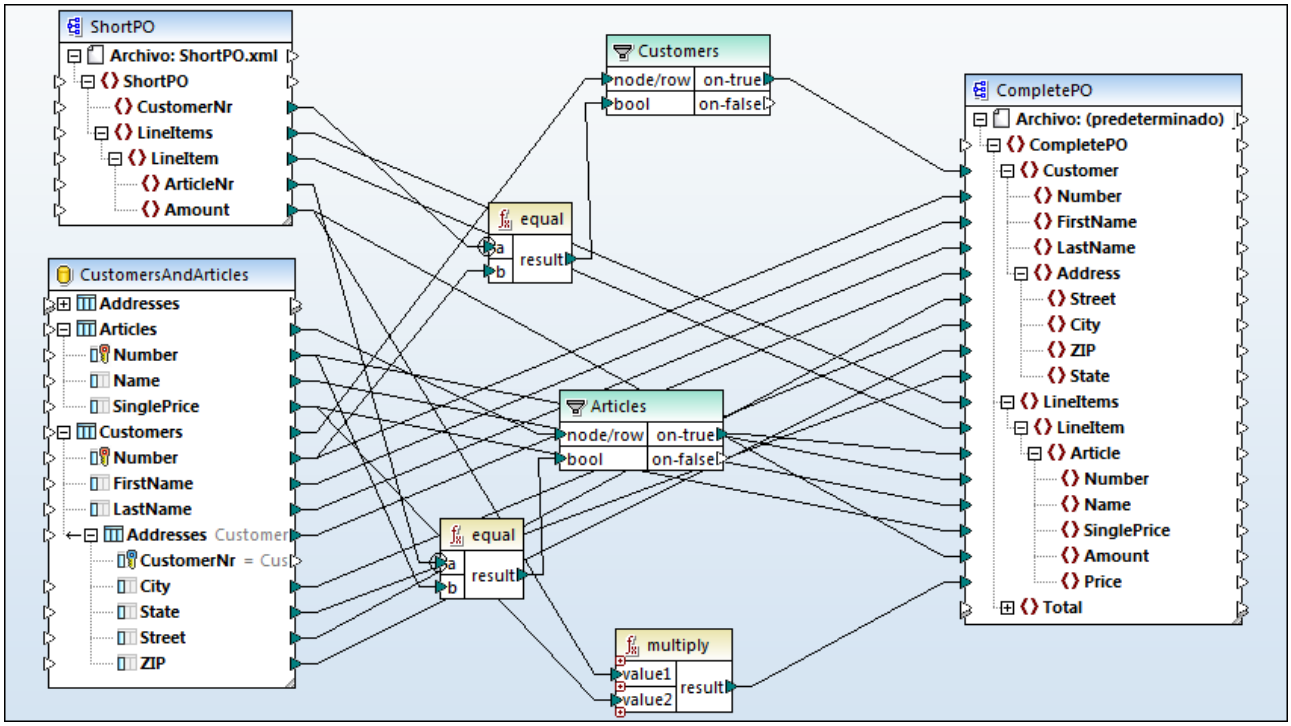

© 2016-2022 Altova GmbH

15.5 Integrar código generado con MapForce

El código generado con MapForce se puede integrar o se puede adaptar a otras aplicaciones, aunque el resultado de la generación de código sea una aplicación completa y totalmente funcional. Algunas situaciones en las que puede ser necesario modificar el código generado:

- Si necesita definir archivos de origen y destino personalizados para la aplicación de asignación de datos.
- Si necesita añadir códigos de control de errores personalizados.
- Si en el código C# o Java desea cambiar el tipo de datos de la entrada de la asignación mediante programación (p. ej. cambiar tipos string por tipos stream).

En esta sección explicamos cómo conseguir estos objetivos y para ello se usa la asignación de muestra **DB_CompletePO.mfd** disponible en el directorio <Documentos>\Altova\MapForce2023\MapForceExamples\.



Asignación de muestra DB_CompletePO.mfd de MapForce

Como puede ver en la imagen, la asignación de datos está compuesta por dos orígenes y un destino:

- **ShortPO.xml** es un archivo XML de origen.
- **CustomersAndArticles.mdb** es una base de datos de origen.
- **CompletePO.xml** es el archivo XML de destino.

En el código generado estos orígenes y destinos se convertirán en dos parámetros de entrada y uno de salida que se aportan al método `run` para ejecutar la asignación de datos (ver apartados de la sección). Por ahora basta con tener en cuenta estos aspectos básicos de la generación de código en MapForce:

- El número de orígenes y destinos del diseño de asignación corresponde al número de parámetros de asignación del método **run** en el código generado.
- Si modifica el número de orígenes y destinos de la asignación, entonces deberá volver a generar el código.
- Si realiza cambios en el código generado y vuelve a generar el código en la misma ubicación, todos los cambios se sobrescribirán.

Si una asignación de datos incluye componentes de base de datos, el método **run** generado incluye el objeto de conexión de base de datos en la ubicación correspondiente. Por ejemplo, si la asignación de datos utiliza tres orígenes (contenido de texto, contenido XML y una base de datos) para asignar datos a un solo archivo de salida, MapForce generará este método **run**:

Java

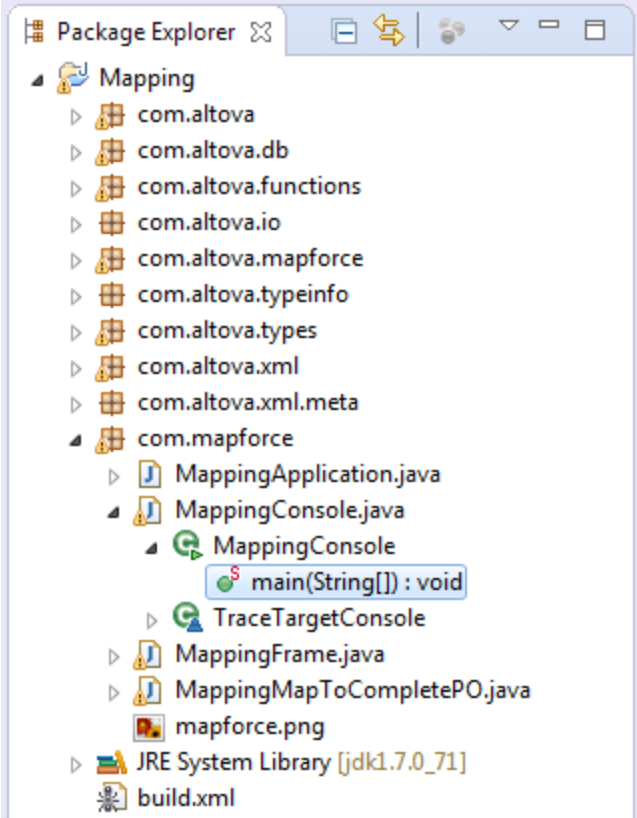
```
void run(Input in1, Input in2, java.sql.Connection dbConn, Output out1);
```

El orden de los argumentos es importante. Y, como verá en los ejemplos de esta sección, cuando integre el código generado, puede modificar los parámetros dbConn o usar los parámetros predeterminados generados por MapForce.

15.5.1 Ejemplo: Java

El entorno IDE Java utilizado para este ejemplo es Eclipse. Para empezar debe generar código Java a partir de la asignación de ejemplo **DB_CompletePO** (situada en el directorio <Documentos>\Altova\MapForce2023\MapForceExamples\) y después importe el proyecto en Eclipse.

 © 2016-2022 Altova GmbH



Aplicación Java de muestra generada con MapForce (en entorno Eclipse)

Para editar la aplicación de consola Java generada debe encontrar el método `main(String[] args)` de la aplicación generada (*imagen anterior*). Si no modificó el nombre predeterminado del paquete base antes de generar el código, este método estará en la clase `MappingConsole` del paquete `com.mapforce`. Si, por el contrario, modificó el nombre predeterminado del paquete base, entonces estará en la clase `MappingConsole` del paquete.

Para editar la aplicación de interfaz gráfica Java generada debe encontrar la posición desde la que se invoca al método `run` en la aplicación generada. Si no modificó el nombre predeterminado del paquete base antes de generar el código, el método **`run`** se invoca desde la clase `MappingFrame.java` del paquete `com.mapforce`.

La muestra de código que aparece a continuación es un extracto del método `main` de la aplicación de consola Java generada. Los orígenes y destinos de la asignación de datos aparecen resaltados en amarillo y están definidos como parámetros del método **`run`**. Como esta asignación utiliza una conexión de base de datos, el parámetro correspondiente a la conexión tiene una estructura especial. Concretamente está compuesta por la cadena de conexión (`jdbc:odbc:;DRIVER=Microsoft Access Driver (*.mdb);DBQ=CustomersAndArticles.mdb`) y por dos argumentos vacíos reservados para el **nombre de usuario** y para la **contraseña** (en texto no cifrado) si se trata de una base de datos que requiere esta información.

Recuerde que las rutas de acceso del código que aparece a continuación son relativas y no absolutas.

```
com.altova.io.Input ShortPO2Source =
com.altova.io.StreamInput.createInput("ShortPO.xml");
com.altova.io.Output CompletePO2Target = new com.altova.io.FileOutput("CompletePO.xml");

MappingMapToCompletePOObject.run(
                  com.altova.db.Dbs.newConnection(
                     "jdbc:odbc:;DRIVER=Microsoft Access Driver
(*.mdb);DBQ=CustomersAndArticles.mdb",
                     "",
                     ""),
                  ShortPO2Source,
                  CompletePO2Target);
```

Para definir archivos de origen o destino de la asignación personales:

- Busque los parámetros que se pasan al método `run` y edítelos según corresponda. En el código de ejemplo anterior com.altova.db.Dbs.newConnection y ShortPO2Source constituyen la entrada de la asignación y CompletePO2Target es la salida.

Para agregar un código de control de errores adicional:

- Si se trata de una aplicación de consola Java, edite el código que aparece debajo del código `catch` (`Exception e`).
- Si se trata de una aplicación de interfaz gráfica Java, edite el código que aparece debajo del código `catch` (`Exception ex`).

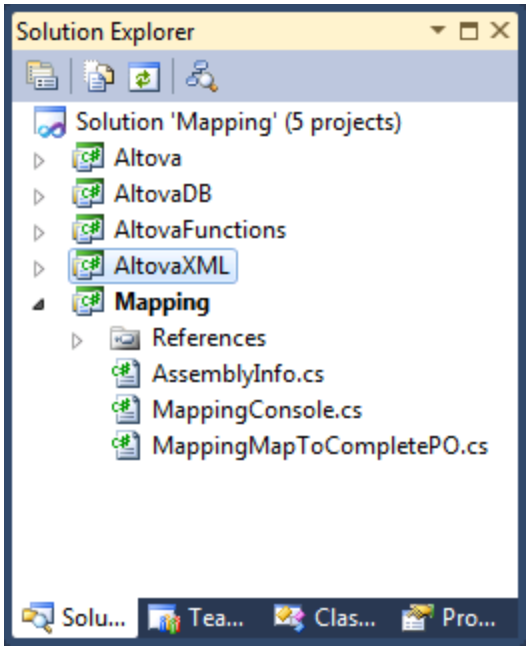
Si necesita instrucciones para modificar el tipo de datos de los parámetros que se dan como entrada/salida de la asignación consulte el apartado Cambiar el tipo de datos de la entrada/salida de la asignación (C#, Java) 1103.

15.5.2 Ejemplo: C#

El entorno IDE utilizado para este ejemplo es Visual Studio 2010. Para empezar debe generar código C# a partir de la asignación de ejemplo **DB_CompletePO** (situada en el directorio <Documentos>\Altova\MapForce2023\MapForceExamples\) y después abra la solución en Visual Studio.

El nombre predeterminado del archivo de solución es **Mapping.sln** y su ubicación es el subdirectorio **Mapping** relativo al directorio donde guardó el código generado. Si cambió el nombre de la aplicación desde la configuración de la asignación de datos, entonces el nombre del archivo .sln también se modificó de la forma correspondiente. Por ejemplo, si cambió el nombre de la aplicación por **MiAplicación**, entonces el archivo de solución se llamará **MiAplicación.sln** y estará ubicado en el subdirectorio **MiAplicación**.

 © 2016-2022 Altova GmbH



Solución C# de muestra generada con MapForce

Abra el archivo **MappingConsole.cs** y busque el método `main(String[] args)`. La muestra de código que aparece a continuación es un extracto de este método. Los orígenes y destinos de la asignación de datos aparecen resaltados en amarillo y están definidos como parámetros del método **Run**. Como esta asignación lee datos de una base de datos, también hay un parámetro de entrada que es una cadena de conexión a una base de datos. Si lo necesita, puede modificar la cadena de conexión de la base de datos.

Recuerde que las rutas de acceso del código que aparece a continuación son relativas y no absolutas.

```
Altova.IO.Input ShortPO2Source = Altova.IO.StreamInput.createInput("ShortPO.xml");
Altova.IO.Output CompletePO2Target = new Altova.IO.FileOutput("CompletePO.xml");

MappingMapToCompletePOObject.Run(
                "Provider=Microsoft.Jet.OLEDB.4.0; Data Source=CustomersAndArticles.mdb;
",
                ShortPO2Source,
                CompletePO2Target);
```

Para definir archivos de origen o destino de la asignación personales:

- Busque los parámetros que se pasan al método **Run** y edítelos según corresponda. En el código de ejemplo anterior la entrada de la asignación es una cadena de conexión a la base de datos **CustomersAndArticles.mdb** y también `ShortPO2Source`. La salida de la asignación es `CompletePO2Target`.

Para agregar un código de control de errores adicional:

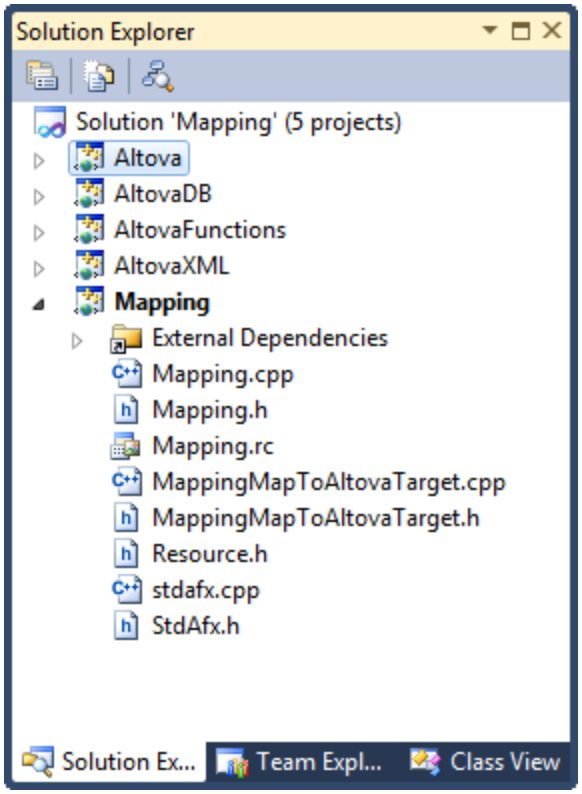
- Edite el código que aparece debajo del código `catch` (`Exception e`).

Si necesita instrucciones para modificar el tipo de datos de los parámetros que se dan como entrada/salida de la asignación consulte el apartado Cambiar el tipo de datos de la entrada/salida de la asignación (C#, Java) 1103.

15.5.3 Ejemplo: C++

El entorno IDE utilizado para este ejemplo es Visual Studio 2010. Para empezar debe generar código C++ a partir de la asignación de ejemplo **DB_CompletePO** (situada en el directorio <Documentos>\Altova\MapForce2023\MapForceExamples\) y después abra la solución en Visual Studio.

El nombre predeterminado del archivo de solución es **Mapping.sln** y su ubicación es el subdirectorio **Mapping** relativo al directorio donde guardó el código generado. Si cambió el nombre de la aplicación desde la configuración de la asignación de datos, entonces el nombre del archivo .sln también se modificó de la forma correspondiente. Por ejemplo, si cambió el nombre de la aplicación por **MiAplicación**, entonces el archivo de solución se llamará **MiAplicación.sln** y estará ubicado en el subdirectorio **MiAplicación**.



Solución C++ de muestra generada con MapForce

Abra el archivo **MappingConsole.cpp** y busque el método `_tmain`. La muestra de código que aparece a continuación es un extracto de este método. Los orígenes y destinos de la asignación están definidos como parámetros del método **Run**. Como esta asignación lee datos de una base de datos, también hay un parámetro

de entrada que es una cadena de conexión a una base de datos. Si lo necesita, puede modificar la cadena de conexión de la base de datos.

Recuerde que las rutas de acceso del código que aparece a continuación son relativas y no absolutas.

```
MappingMapToCompletePO MappingMapToCompletePOObject;
         MappingMapToCompletePOObject.Run(
            _T("Provider=Microsoft.Jet.OLEDB.4.0; Data Source=CustomersAndArticles.mdb;
"),
            _T("ShortPO.xml"),
            _T("CompletePO.xml"));
```

Para definir archivos de origen o destino de la asignación personales:

- Busque los parámetros que se pasan al método **Run** y edítelos según corresponda. En el código de ejemplo anterior la entrada de la asignación es una cadena de conexión a la base de datos **CustomersAndArticles.mdb** y también _T("ShortPO.xml"). La salida de la asignación es _T("CompletePO.xml") .

Para agregar un código de control de errores adicional:

- Edite el código que aparece debajo del código `catch` (`CAltovaException& e`).

15.5.4 Cambiar tipo de datos de entrada/salida de la asignación (C# y Java)

Después de generar código Java o C# con MapForce tiene la opción de cambiar el tipo de datos de entrada o salida de la asignación; para ello basta con editar el código generado. Esto quiere decir que puede usar como parámetros de asignación objetos de tipos distintos a los que se generan por defecto. Por ejemplo, en lugar de que la asignación lea los datos de entrada de un archivo guardado en disco puede usar una cadena de texto o un objeto se secuencia como entrada. Esta opción solo está disponible para el código generado en C# o Java.

Puede consultar los tipos de objetos que puede usar como entrada o salida en la primera columna de la tabla siguiente. El resto de columnas indica qué formatos de datos son compatibles con ese tipo en concreto. Para leer una definición más precisa de cada uno de los tipo consulte el punto "Definiciones de tipos" más abajo.

	XML	JSON*	Microsoft Excel*	EDI (incl. X12, HL7)*	FlexText*	CSV/Texto
Archivos	Sí	Sí	Sí	Sí	Sí	Sí
Secuencias	Sí	Sí	Sí	Sí	Sí	Sí
Cadenas	Sí	Sí	–	Sí	Sí	Sí
Escritor/lector	Sí	Sí	–	Sí	Sí	Sí
Documentos DOM	Sí	–	–	–	–	–

** Formatos compatibles con la edición MapForce Enterprise Edition solamente.*

Para cambiar el tipo de datos de entrada o de salida de una asignación:

1. Genere código C# o Java desde una asignación.
2. En el código generado encuentre la llamada al método `run` (en Java) o `Run` (en C#):

   a. Si está usando C#, abra el archivo **MappingConsole.cs**.
   b. Si está usando Java, abra el archivo **MappingConsole.java** (consola) o el archivo **MappingFrame.java** (IGU).

Nota: el nombre del archivo puede ser distinto si cambió el nombre de la aplicación en la configuración de la asignación 115. Por ejemplo, si cambió el nombre a "MyApp", el nombre del archivo generado será, respectivamente, **MyAppConsole.js**, **MyAppConsole.java** y **MyAppFrame.java**.

3. Cree una instancia del tipo que requiera (consulte el punto "Definiciones de tipos").
4. Indique los objetos declarados como parámetros al método `run` (en Java) o `Run` (en C#), como se puede ver en el ejemplo siguiente.

El método `run` es el método más importante de las clases de asignación que se generan. Tiene un parámetro por cada componente de origen o destino *estático* de la asignación y un parámetro final para el componente de salida. Si una asignación contiene componentes que procesan varios archivos de forma dinámica 892, los parámetros correspondientes no aparecen en el código generado porque en este caso los nombres de los archivos se procesan de forma dinámica dentro de la asignación.

Definiciones de tipos

En C#, los tipos que puede usar como parámetros del método `Run` son clases que se definen en el espacio de nombres `Altova.IO`. Las clases base son `Altova.IO.Input` y `Altova.IO.Output` respectivamente.

Tipos C#

Archivos	`Altova.IO.FileInput(string nombreArchivo)` `Altova.IO.FileOutput(string nombreArchivo)`
Secuencias	`Altova.IO.StreamInput(System.IO.Stream secuencia)` `Altova.IO.StreamOutput(System.IO.Stream secuencia)`

 © 2016-2022 Altova GmbH

Cadenas	`Altova.IO.StringInput(string contenido)` `Altova.IO.StringOutput(System.Text.StringBuilder generadorCadenas)`
Escritor/lector	`Altova.IO.ReaderInput(System.IO.TextReader lector)` `Altova.IO.WriterOutput(System.IO.TextWriter escritor)`
Documentos DOM	`Altova.IO.DocumentInput(System.Xml.XmlDocument documento)` `Altova.IO.DocumentOutput(System.Xml.XmlDocument documento)`

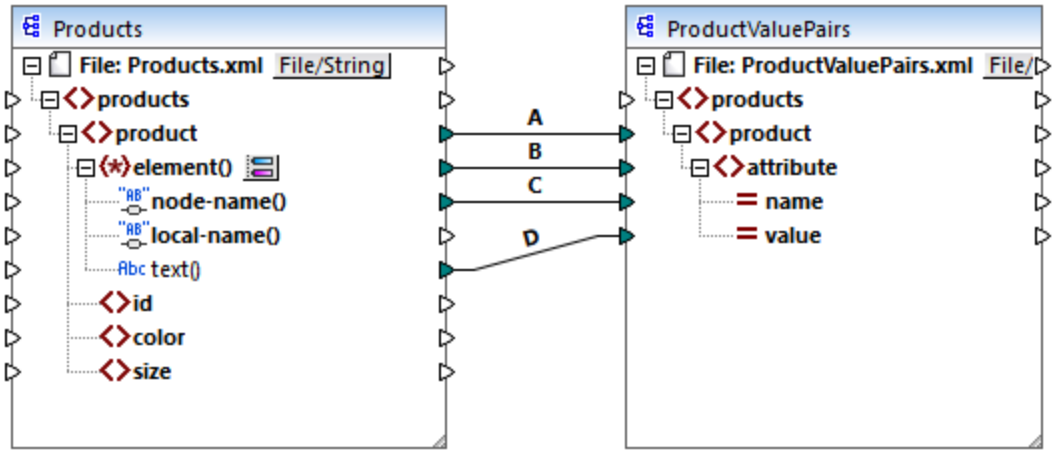
En Java, los tipos que puede usar como parámetros del método **`run`** son clases que se definen en el paquete `com.altova.io`. Las clases base son `com.altova.io.Input` y `com.altova.io.Output` respectivamente.

Tipos Java

Archivos	`com.altova.io.FileInput(String nombreArchivo)` `com.altova.io.FileOutput(String nombreArchivo)`
Secuencias	`com.altova.io.StreamInput(java.io.InputStream secuencia)` `com.altova.io.FileOutput(String nombreArchivo)`
Cadenas	`Altova.IO.StringInput(string contenido)` `com.altova.io.StringOutput()`
Escritor/lector	`com.altova.io.ReaderInput(java.io.Reader lector)` `com.altova.io.WriterOutput(java.io.Writer escritor)`
Documentos DOM	`com.altova.io.DocumentOutput(org.w3c.dom.Document documento)` `com.altova.io.DocumentOutput(org.w3c.dom.Document documento)`

Ejemplo

En este ejemplo usaremos la asignación **ConvertProducts.mfd** para ilustrar cómo cambiar el tipo de datos de entrada y salida de forma programática. Después de instalar MapForce y ejecutarlo al menos una vez puede encontrar esta asignación en el directorio: C:\Usuarios\<usuario>\Documentos\Altova\MapForce**2023\MapForceExamples\Tutorials**.

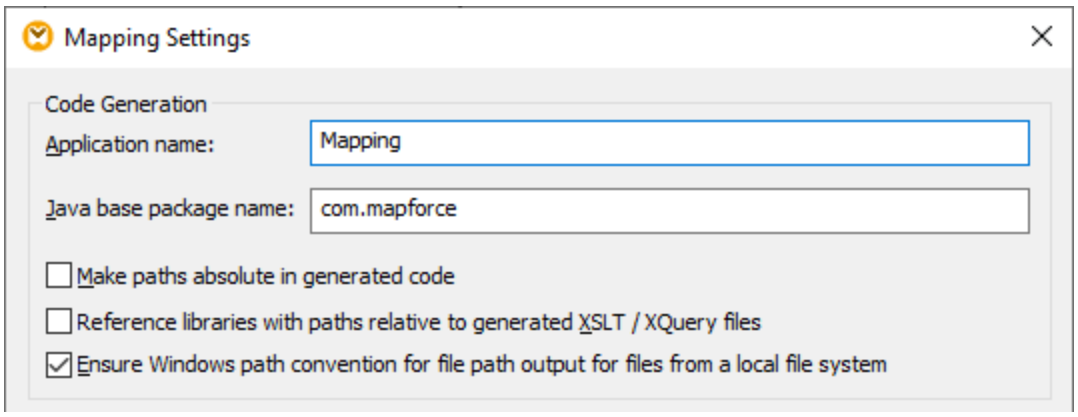


ConvertProducts.mfd

Como se ve más arriba, la asignación convierte los datos de origen de un document XML en otro documento XML. En este caso los objetivos son dos:

1. Generar código de programa Java y C# desde esta asignación.
2. Cambiar el tipo de datos del componente de entrada a cadena de texto.
3. Cambiar el tipo de datos del componente de destino a un escritor de cadenas.

Para generar el código de programa, abra la asignación **ConvertProducts.mfd** y ejecute el comando **Archivo | Generar código en | C#** (o **Java**) En este ejemplo asumimos que la asignación mantiene la configuración predeterminada (**ConvertProducts.mfd**).



En este ejemplo el código generado se escribe en los directorios de destino siguientes (puede cambiar las rutas si quiere):

- **C:\codegen\cs\ConvertProducts**, para C#
- **C:\codegen\java\ConvertProducts**, para Java

Una vez haya generado el código, abra **MappingConsole.cs** (en C#) o **MappingConsole.java** (en Java) y busque estas líneas:

C#

```
Altova.IO.Input Products2Source = Altova.IO.StreamInput.createInput("Products.xml");
Altova.IO.Output ProductValuePairs2Target = new
Altova.IO.FileOutput("ProductValuePairs.xml");
```

Java

```
com.altova.io.Input Products2Source =
com.altova.io.StreamInput.createInput("Products.xml");
com.altova.io.Output ProductValuePairs2Target = new
com.altova.io.FileOutput("ProductValuePairs.xml");
```

Convierta las líneas anteriores en comentario y cambie el código como sigue:

C#

```
//Altova.IO.Input Products2Source = Altova.IO.StreamInput.createInput("Products.xml");
```

 © 2016-2022 Altova GmbH

```
//Altova.IO.Output ProductValuePairs2Target = new
Altova.IO.FileOutput("ProductValuePairs.xml");

Altova.IO.Input Products2Source = new Altova.IO.StringInput("<?xml version=\"1.0\"
encoding=\"UTF-8\"?>\r\n" +
                                        "<products
xmlns:xsi=\"http://www.w3.org/2001/XMLSchema-instance\"
xsi:noNamespaceSchemaLocation=\"products.xsd\">\r\n" +
                                        "   <product>\r\n" +
                                        "      <id>100</id>\r\n" +
                                        "      <color>blue</color>\r\n" +
                                        "      <size>XXL</size>\r\n" +
                                        "   </product>\r\n" +
                                        "</products>\r\n");

System.IO.StringWriter writer = new System.IO.StringWriter(new
System.Text.StringBuilder());
Altova.IO.Output ProductValuePairs2Target = new Altova.IO.WriterOutput(writer);

try
{
    MappingMapToProductValuePairsObject.Run(Products2Source, ProductValuePairs2Target);

    // Print out the writer object
    Console.Write(writer.ToString());
}
finally
{
    Products2Source.Close();
    ProductValuePairs2Target.Close();
}
```

Java

```
//com.altova.io.Input Products2Source =
com.altova.io.StreamInput.createInput("Products.xml");
//com.altova.io.Output ProductValuePairs2Target = new
com.altova.io.FileOutput("ProductValuePairs.xml");

com.altova.io.Input Products2Source = new com.altova.io.StringInput("<?xml
version=\"1.0\" encoding=\"UTF-8\"?>\r\n" +
               "<products xmlns:xsi=\"http://www.w3.org/2001/XMLSchema-instance\"
xsi:noNamespaceSchemaLocation=\"products.xsd\">\r\n" +
               "   <product>\r\n" +
               "      <id>100</id>\r\n" +
               "      <color>blue</color>\r\n" +
               "      <size>XXL</size>\r\n" +
               "   </product>\r\n" +
               "</products>\r\n");

java.io.StringWriter writer = new java.io.StringWriter();
com.altova.io.Output ProductValuePairs2Target = new com.altova.io.WriterOutput(writer);
```

```
try {
   MappingMapToProductValuePairsObject.run(Products2Source, ProductValuePairs2Target);

   // Print out the writer object
   System.out.print(writer.toString());

   } finally {
      (Products2Source).close();
      ProductValuePairs2Target.close();
}
```

En los fragmentos de código C# y Java anteriores ocurre lo siguiente:

- Las dos líneas originales que indican la entrada y la salida del método `run` ahora son comentarios. Esto quiere decir que la asignación ya no lee datos de **Products.xml**. De hecho, ni siquiera necesitamos copiar este archivo en el directorio de trabajo del programa.
- El tipo `Products2Source` se declara como un `StringInput` que indica el contenido del archivo XML que se quiere procesar.
- El tipo `ProductValuePairs2Target` se declara como un tipo `WriterOutput` que toma un escritor de cadenas como argumento.
- Cuando la asignación ha terminado de ejecutarse, el contenido del escritor de cadenas se envía a la consola.

Normas de uso para secuencias y objetos escritor/lector

Al usar secuencias binarias u objetos lector/escritor como entrada o salida de la asignación debe tener en cuenta que:

- Se supone que los objetos de secuencia binaria y los objetos lector/escritor se abren y están listos antes de llamar al método `run`.
- Por defecto el método `run` cierra la secuencia cuando se termina. Para evitar este comportamiento basta con insertar (o convertir de comentario en código) esta línea antes de llamar al método `run`:

C#

```
MappingMapToSomething.CloseObjectsAfterRun = false;
```

Java

```
MappingMapToSomething.setCloseObjectsAfterRun(false);
```

Nota: asegúrese de que cambia el nombre `MappingMapToSomething` por el del objeto de asignación según el código generado.

Normas de uso para cadenas de texto

En Java el constructor de `StringOutput` no toma ningún argumento. Para acceder al contenido de cadena que produce la asignación puede usar el método `getString()`, por ejemplo:

Java

 © 2016-2022 Altova GmbH

```
com.altova.io.Input Products2Source =
com.altova.io.StreamInput.createInput("Products.xml");
com.altova.io.StringOutput ProductValuePairs2Target = new com.altova.io.StringOutput();

try {
   // Run the mapping
   MappingMapToProductValuePairsObject.run(Products2Source, ProductValuePairs2Target);
   // Get the string object
   String str = ProductValuePairs2Target.getString().toString();
}
```

En C#, el constructor de `StringOutput` toma un parámetro de tipo `StringBuilder` que debe declarar de antemano. Si el argumento `StringBuilder` ya contiene datos, el resultado de la asignación se le anexa.

C#

```
Altova.IO.Input Products2Source = Altova.IO.StreamInput.createInput("Products.xml");
System.Text.StringBuilder sb = new System.Text.StringBuilder();
Altova.IO.Output ProductValuePairs2Target = new Altova.IO.StringOutput(sb);

try
{
   // Run the mapping
   MappingMapToProductValuePairsObject.Run(Products2Source, ProductValuePairs2Target);
   // Get the string output
   String str = sb.ToString();
}
```

Para ejecutar estos fragmentos de código puede usar el mismo proyecto generado que en el ejemplo anterior. Sin embargo, debe asegurarse de que copia el archivo de **Products.xml** de **C:\Users\<username>\Documents\Altova\MapForce2023\MapForceExamples\Tutorials** en el directorio de trabajo de su programa, ya que el código de la asignación lee datos de ese archivo.

Normas de uso para documentos DOM

Cuando use documentos DOM como entrada o salida de una asignación debe tener en cuenta que:

- La instancia del documento dada como parámetro al constructor `DocumentOutput` debe estar vacía.
- Después de llamar a **`run`** el documento DOM que genera el constructor de `DocumentOutput` ya contiene el resultado de la asignación y puede manipular el documento como quiera.

15.6 Generar código fuente a partir de esquemas XML o DTD

Cuando genera código a partir de una asignación de datos, MapForce genera una aplicación completa que ejecuta todos los pasos de la asignación de datos automáticamente. Si lo desea, también puede generar bibliotecas para todos los esquemas XML utilizados en la asignación. Estos esquemas permitirán al código crear o leer con facilidad las instancias XML utilizadas o creadas por el código de la asignación.

Para generar bibliotecas para todos los esquemas XML utilizados en la asignación de datos marque la casilla *Generar clases contenedoras* en el cuadro de diálogo "Opciones" (*véase* *Opciones del generador de código* 1207). La próxima vez que genere código MapForce no sólo creará la aplicación de asignación de datos sino que además generará clases contenedoras para todos los esquemas que utiliza la asignación:

C++ o C#	**Java**	**Objetivo**
Altova	`com.altova`	Biblioteca base que contiene compatibilidad con Common Language Runtime, idéntica para todos los esquemas.
AltovaXML	`com.altova.xml`	Biblioteca base que contiene compatibilidad con Common Language Runtime para XML, idéntica para todos los esquemas.
MiEsquema	*`com.MiEsquema`*	Biblioteca que contiene declaraciones generadas a partir del esquema de entrada, llamado archivo de esquema o DTD. Esta biblioteca es un contenedor DOM (W3C Document Object Model) que permite leer, modificar y crear documentos XML de forma sencilla y segura. Todos los datos están dentro del DOM y hay métodos para extraer y actualizar datos y crear datos nuevos en el DOM. El código C++ generado es compatible con Microsoft MSXML o Apache Xerces 3. Por lo general, la sintaxis para usar el código generado es parecida en ambas implementaciones salvo por algunas diferencias (por ejemplo, Xerces admite más funciones de sobrecarga). El código C# generado utiliza la biblioteca System.XML estándar de .NET como implementación DOM subyacente. El código Java generado utiliza JAXP como interfaz DOM subyacente.

Cuando cree prototipos de aplicaciones a partir de esquemas XML que cambien con frecuencia, a veces será necesario generar código una y otra vez en el mismo directorio para que los cambios en el esquema se reflejen inmediatamente en el código. Recuerde que la aplicación de prueba que se genera y las bibliotecas de Altova se sobrescribirán cada vez que genere código en el mismo directorio de destino. Por

 © 2016-2022 Altova GmbH

tanto, recuerde que no debe añadir código a la aplicación de prueba que se genera, sino que debe integrar las bibliotecas de Altova en el proyecto (ver *Integrar bibliotecas contenedoras de esquemas* 1120).

Además de las bibliotecas base mencionadas en la tabla, también se generan algunas bibliotecas de ayuda. Estas bibliotecas son utilizadas por las bibliotecas base de Altova y no deben ser utilizadas en integraciones personales porque pueden sufrir cambios en versiones futuras de la aplicación.

Espacios de nombres y generación de nombres

MapForce genera clases que se corresponden con todos los elementos o tipos complejos declarados que redefinen todos los tipos complejos del esquema XML, conservando la derivación de clases tal y como definan las extensiones de tipos complejos del esquema. Cuando se trate de esquemas complejos que importen componentes de esquemas desde varios espacios de nombres distintos, MapForce conserva esta información generando los espacios de nombres C#/C++ o los paquetes Java correspondientes.

Por lo general, el generador de código intenta conservar los nombres del esquema original para los espacios de nombres, clases y miembros generados. En el lenguaje de destino los caracteres no válidos que se encuentren en los identificadores se reemplazan con "_". A los nombres que estén en conflicto con otros nombres o palabras reservadas se les anexa un número para que sean nombres únicos. Si cambia la configuración predeterminada de la plantilla SPL 1209, puede que la generación de nombres se vea afectada.

Los espacios de nombres del esquema XML se convierten en paquetes en Java o en espacios de nombres en C#/C++, usando siempre el prefijo de espacios de nombres del esquema como espacio de nombres del código. La biblioteca completa se adjunta en un paquete o espacio de nombres derivado del nombre del archivo de esquema. Así podrá usar varias bibliotecas generadas en un programa sin que haya conflictos entre nombres.

Tipos de datos

XML Schema tiene un modelo de tipos de datos más complejo que Java, C# o C++. El generador de código convierte los tipos XML Schema integrados en tipos primitivos propios de cada lenguaje de programación o en clases derivadas con la biblioteca de Altova. Los tipos complejos y derivados definidos en el esquema se convierten en clases en la biblioteca que se genera. Las facetas de enumeración de tipos simples se convierten en constantes simbólicas.

La asignación entre tipos simples se puede configurar en la Referencia de SPL 1209.

Si los archivos de instancia XML utilizan tipos de esquema relacionados con horas y duración, estos se convierten en clases nativas de Altova en el código generado. Para más información consulte estos apartados:

- Referencia de las clases generadas (C++) 1155
- Referencia de las clases generadas (C#) 1172
- Referencia de las clases generadas (Java) 1188

Para más información sobre la conversión de tipos y otros detalles propios de cada lenguaje consulte estos apartados:

- Bibliotecas contenedoras de esquemas (C++) 1113
- Bibliotecas contenedoras de esquemas (C#) 1115
- Bibliotecas contenedoras de esquemas (Java) 1118

Gestión de memoria

Una estructura DOM está compuesta por nodos que siempre son propiedad de un documento DOM concreto, aunque el nodo no forme parte del contenido del documento en ese momento. Todas las clases generadas son referencias a los nodos DOM que representan, no valores. Esto significa que si se asigna una instancia de una clase generada, el valor no se copia, sino que se crea una referencia adicional a los mismos datos.

Compatibilidad con XML Schema

Estas construcciones de XML Schema se traducen en el código:

a) Espacios de nombres XML

b) Tipos simples:

- Tipos de XML Schema integrados
- Tipos simples derivados por extensión
- Tipos simples derivados por restricción
- Facetas
- Enumeraciones
- Patrones

c) Tipos complejos

- Nodo anyType integrado
- Tipos complejos definidos por el usuario
- Derivados por extensión: se asignan a clases derivadas
- Derivados por restricción
- Contenido complejo
- Contenido simple
- Contenido mixto

Estas características de XML Schema avanzadas no son compatibles con las clases contenedoras generadas:

- Comodines: `xs:any` y `xs:anyAttribute`
- Modelos de contenido (sequence, choice, all). El compositor de nivel superior está disponible en el SPL 1209, pero las clases generadas no lo exigen.
- Valores predeterminados y fijos para atributos. Están disponibles en el SPL 1209, pero las clases generadas no los exigen ni establecen.
- Los atributos `xsi:type`, tipos abstractos. Cuando necesite escribir el atributo `xsi:type`, utilice el método `SetXsiType()` de las clases generadas.
- Tipos de unión: no todas las combinaciones son compatibles.
- Los grupos de sustitución son compatibles en parte solamente (se resuelven como "choice").
- Atributos `nillable="true"` y `xsi:nil`
- Restricciones de unicidad
- Restricciones de identidad (`key` y `keyref`)

© 2016-2022 Altova GmbH

15.6.1 Bibliotecas contenedoras de esquemas (C++)

Tipos de caracteres

El código C++ generado se puede compilar con o sin compatibilidad con Unicode. Dependiendo de la configuración elegida, los tipos `string_type` y `tstring` se definirán como `std::string` o `std::wstring` y estarán formados por caracteres estrechos o anchos. Para usar caracteres Unicode en su archivo XML que no se puedan representar con el conjunto de caracteres de 8 bits actual deberá habilitar la compatibilidad con Unicode. Deberá prestar especial atención a las macros `_T()`. Esta macro garantizan el correcto almacenamiento de las constantes de cadena (tanto si se compila para programas Unicode como para programas no Unicode).

Tipos de datos

Esta es la correspondencia predeterminada de tipos XML Schema con tipos de datos C++:

XML Schema	C++	Observaciones
`xs:string`	`string_type`	`string_type` se define como `std::string` o `std:wstring`
`xs:boolean`	`bool`	
`xs:decimal`	`double`	C++ no tiene un tipo decimal por lo que se debe usar `double`.
`xs:float, xs:double`	`double`	
`xs:integer`	`__int64`	`xs:integer` tiene un rango ilimitado que se corresponde con `__int64` para mayor eficacia.
`xs:nonNegativeInteger`	`unsigned __int64`	ver anterior
`xs:int`	`int`	
`xs:unsignedInt`	`unsigned int`	
`xs:dateTime, date, time, gYearMonth, gYear, gMonthDay, gDay, gMonth`	`altova::DateTime` 1155	
`xs:duration`	`altova::Duration` 1159	
`xs:hexBinary and xs:base64Binary`	`std::vector<unsigned char>`	La codificación y decodificación de datos binarios se realiza automáticamente.
`xs:anySimpleType`	`string_type`	

Los tipos XML Schema que faltan en la lista son tipos derivados y corresponde al mismo tipo C++ que su correspondiente tipo base.

Clases generadas

Por cada tipo del esquema se genera una clase que contiene un miembro por cada atributo y elemento del tipo. Los miembros se llaman igual que los atributos o elementos del esquema original (en caso de conflicto se anexa un número). Para los tipos simples se generan operadores de asignación y conversión. Para los tipos simples con facetas enumeration, se pueden usar los métodos `GetEnumerationValue()` y `SetEnumerationValue(int)` junto con las constantes generadas para cada valor de enumeración. Además, el método `StaticInfo()` permite acceder a la información del esquema como uno de estos tipos:

`altova::meta::SimpleType` 1165
`altova::meta::ComplexType` 1163

Las clases generadas a partir de tipos complejos incluyen el método `SetXsiType()`, que permite configurar el atributo `xsi:type` del tipo. Este método puede ser muy útil a la hora de crear elementos de instancia XML de un tipo derivado.

Además de las clases para los tipos declarados en el esquema XML, se genera también una clase de documento ("CDoc"). Esta clase contiene todos los elementos raíz posibles como miembros, así como varios métodos más. Para más información sobre esta clase consulte `[SuEsquema]::[CDoc]` 1166.

Nota: el nombre real de la clase depende del nombre del esquema .xsd.

Por cada atributo o elemento miembro de un tipo de esquema se genera una clase nueva. Para más información sobre dichas clases consulte:

`[SuEsquema]::AtributoMiembro` 1170
`[SuEsquema]::ElementoMiembro` 1170

Nota: el nombre real de las clases depende del nombre del atributo o elemento del esquema.

Consulte también el apartado Ejemplo: usar las bibliotecas contenedoras del esquema 1123.

Control de errores

Los errores se notifican por medio de excepciones. Estas son las clases de excepción que están definidas en el espacio de nombres `altova`:

Clase	Clase base	Descripción
`Error`	`std::logic_error`	Error de lógica de programa interna (independiente de los datos de entrada).
`Exception`	`std::runtime_error`	Clase base para errores de tiempo de ejecución.
`InvalidArgumentsException`	`Exception`	Se llamó a un método con valores de argumento no válidos.

© 2016-2022 Altova GmbH

`ConversionException`	`Exception`	Excepción generada cuando falla una conversión de tipo.
`StringParseException`	`ConversionException`	Un valor del espacio léxico no se puede convertir en un espacio de valor.
`ValueNotRepresentableExceptio n`	`ConversionException`	Un valor del espacio de valor no se puede convertir en espacio léxico.
`OutOfRangeException`	`ConversionException`	Un valor de origen no se puede representar en el dominio de destino.
`InvalidOperationException`	`Exception`	Se intentó una operación que no es válida en el contexto dado.
`DataSourceUnavailableExceptio n`	`Exception`	Se encontró un problema al intentar cargar una instancia XML.
`DataTargetUnavailableExceptio n`	`Exception`	Se encontró un problema al intentar guardar una instancia XML.

Todas las clases de excepción contienen un texto de mensaje y un puntero que apunta a una posible excepción interna.

Método	Finalidad
`string_type message()`	Devuelve una descripción textual de la excepción.
`std::exception inner()`	Devuelve la excepción que causó esta excepción (si procede) o devuelve NULL.

Acceso a la información del esquema

La biblioteca generada permite acceder a información estática del esquema por medio de estas clases. Todos los métodos se declaran como `const`.

`altova::meta::Attribute` 1162
`altova::meta::ComplexType` 1163
`altova::meta::Element` 1164
`altova::meta::SimpleType` 1165

Los métodos que devuelven una de las clases de metadatos devuelven un objeto NULL si no existe la propiedad correspondiente.

15.6.2 Bibliotecas contenedoras de esquemas (C#)

Esta es la correspondencia predeterminada de tipos XML Schema con tipos de datos C#:

XML Schema	C#	Observaciones
xs:string	string	
xs:boolean	bool	
xs:decimal	decimal	xs:decimal tiene un rango y una precisión ilimitados y se corresponde decimal para una mayor eficacia.
xs:float, xs:double	double	
xs:long	long	
xs:unsignedLong	ulong	
xs:int	int	
xs:unsignedInt	uint	
xs:dateTime, date, time, gYearMonth, gYear, gMonthDay, gDay, gMonth	Altova.Types.DateTime 1172	
xs:duration	Altova.Types.Duration 1177	
xs:hexBinary and xs:base64Binary	byte[]	La codificación y decodificación de datos binarios se realiza automáticamente.
xs:anySimpleType	string	

Los tipos XML Schema que faltan en la lista son tipos derivados y corresponde al mismo tipo C# que su correspondiente tipo base.

Clases generadas

Por cada tipo del esquema se genera una clase que contiene un miembro por cada atributo y elemento del tipo. Los miembros se llaman igual que los atributos o elementos del esquema original (en caso de conflicto se anexa un número). Para los tipos simples se generan operadores de asignación y conversión. Para los tipos simples con facetas enumeration, se pueden usar los métodos `GetEnumerationValue()` y `SetEnumerationValue(int)` junto con las constantes generadas para cada valor de enumeración. Además, el método `StaticInfo()` permite acceder a la información del esquema como uno de estos tipos:

Altova.Xml.Meta.SimpleType 1181
Altova.Xml.Meta.ComplexType 1180

Las clases generadas a partir de tipos complejos incluyen el método `SetXsiType()`, que permite configurar el atributo `xsi:type` del tipo. Este método puede ser muy útil a la hora de crear elementos de instancia XML de un tipo derivado.

 © 2016-2022 Altova GmbH

Además de las clases para los tipos declarados en el esquema XML, se genera también una clase de documento ("Doc"). Esta clase contiene todos los elementos raíz posibles como miembros, así como varios métodos más. Para más información sobre esta clase consulte [SuEsquema].[Doc] 1182 .

Nota: el nombre real de la clase depende del nombre del esquema .xsd.

Por cada atributo o elemento miembro de un tipo de esquema se genera una clase nueva. Para más información sobre dichas clases consulte:

[SuTipoEsquema].AtributoMiembro 1185
[SuTipoEsquema].ElementoMiembro 1186

Nota: el nombre real de las clases depende del nombre del atributo o elemento del esquema.

Control de errores

Los errores se notifican por medio de excepciones. Estas son las clases de excepción que están definidas en el espacio de nombres `Altova`:

Clase	Clase base	Descripción
`ConversionException`	`Exception`	Excepción generada cuando falla una conversión de tipo.
`StringParseException`	`ConversionException`	Un valor del espacio léxico no se puede convertir en espacio de valor.
`DataSourceUnavailableException`	`System.Exception`	Se encontró un problema al intentar cargar una instancia XML.
`DataTargetUnavailableException`	`System.Exception`	Se encontró un problema al intentar guardar una instancia XML.

Además se suelen usar estas excepciones .NET:

Clase	Descripción
`System.Exception`	Clase base para errores de tiempo de ejecución.
`System.ArgumentException`	Se llamó a un método con valores de argumento no válidos o falló una conversión de tipo.
`System.FormatException`	Un valor del espacio léxico no se puede convertir en espacio de valor.
`System.InvalidCastException`	No se puede convertir un valor en otro tipo.
`System.OverflowException`	No se puede representar un valor de origen en el dominio de destino.

Acceso a la información del esquema

La biblioteca generada permite acceder a información estática del esquema por medio de estas clases.

Altova.Xml.Meta.Attribute 1179
Altova.Xml.Meta.ComplexType 1180
Altova.Xml.Meta.Element 1181
Altova.Xml.Meta.SimpleType 1181

Las propiedades que devuelven una de las clases de metadatos devuelven un objeto NULL si no existe la propiedad correspondiente.

15.6.3 Bibliotecas contenedoras de esquemas (Java)

Esta es la correspondencia predeterminada de tipos XML Schema con tipos de datos Java:

XML Schema	Java	Observaciones
xs:string	String	
xs:boolean	boolean	
xs:decimal	java.math.BigDecimal	
xs:float, xs:double	double	
xs:integer	java.math.BigInteger	
xs:long	long	
xs:unsignedLong	java.math.BigInteger	Java no tiene tipos sin firmar.
xs:int	int	
xs:unsignedInt	long	Java no tiene tipos sin firmar.
xs:dateTime, date, time, gYearMonth, gYear, gMonthDay, gDay, gMonth	com.altova.types.DateTime 1189	
xs:duration	com.altova.types.Duration 1194	
xs:hexBinary and xs:base64Binary	byte[]	La codificación y decodificación de datos binarios se realiza automáticamente.
xs:anySimpleType	string	

Los tipos XML Schema que faltan en la lista son tipos derivados y corresponde al mismo tipo Java que su correspondiente tipo base.

 © 2016-2022 Altova GmbH

Clases generadas

Por cada tipo del esquema se genera una clase que contiene un miembro por cada atributo y elemento del tipo. Los miembros se llaman igual que los atributos o elementos del esquema original (en caso de conflicto se anexa un número). Para los tipos simples se generan operadores de asignación y conversión. Para los tipos simples con facetas enumeration, se pueden usar los métodos `GetEnumerationValue()` y `SetEnumerationValue(int)` junto con las constantes generadas para cada valor de enumeración. Además, el método `StaticInfo()` permite acceder a la información del esquema como uno de estos tipos:

`com.altova.xml.meta.SimpleType` 1199
`com.altova.xml.meta.ComplexType` 1198

Las clases generadas a partir de tipos complejos incluyen el método `SetXsiType()`, que permite configurar el atributo `xsi:type` del tipo. Este método puede ser muy útil a la hora de crear elementos de instancia XML de un tipo derivado.

Además de las clases para los tipos declarados en el esquema XML, se genera también una clase de documento ("Doc"). Esta clase contiene todos los elementos raíz posibles como miembros, así como varios métodos más. Para más información sobre esta clase consulte `com.[SuEsquema].[Doc]` 1200.

Nota: el nombre real de la clase depende del nombre del esquema .xsd.

Por cada atributo o elemento miembro de un tipo de esquema se genera una clase nueva. Para más información sobre dichas clases consulte:

`com.[SuEsquema].[SuTipoEsquema].AtributoMiembro` 1203
`com.[SuEsquema].[SuTipoEsquema].ElementoMiembro` 1204

Nota: el nombre real de las clases depende del nombre del atributo o elemento del esquema.

Control de errores

Los errores se notifican por medio de excepciones. Estas son las clases de excepción que están definidas en el espacio de nombres `com.altova`:

Clase	Clase base	Descripción
`SourceInstanceUnvailableException`	`Exception`	Se encontró un problema al intentar cargar una instancia XML.
`TargetInstanceUnavailableException`	`Exception`	Se encontró un problema al intentar guardar una instancia XML.

Además se suelen usar estas excepciones Java:

Clase	Descripción
`java.lang.Error`	Error de lógica de programa interna (independiente de los datos de entrada).
`java.lang.Exception`	Clase base para errores de tiempo de ejecución.
`java.lang.IllegalArgumentsException`	Se llamó a un método con valores de argumento no válidos o falló una conversión de tipo.
`java.lang.ArithmeticException`	Excepción generada cuando falla una conversión de tipo numérico.

Acceso a la información del esquema

La biblioteca generada permite acceder a información estática del esquema por medio de estas clases.

`com.altova.xml.meta.Attribute` 1197
`com.altova.xml.meta.ComplexType` 1198
`com.altova.xml.meta.Element` 1199
`com.altova.xml.meta.SimpleType` 1199

Las propiedades que devuelven una de las clases de metadatos devuelven un objeto NULL si no existe la propiedad correspondiente.

15.6.4 Integrar las bibliotecas contenedoras de esquemas

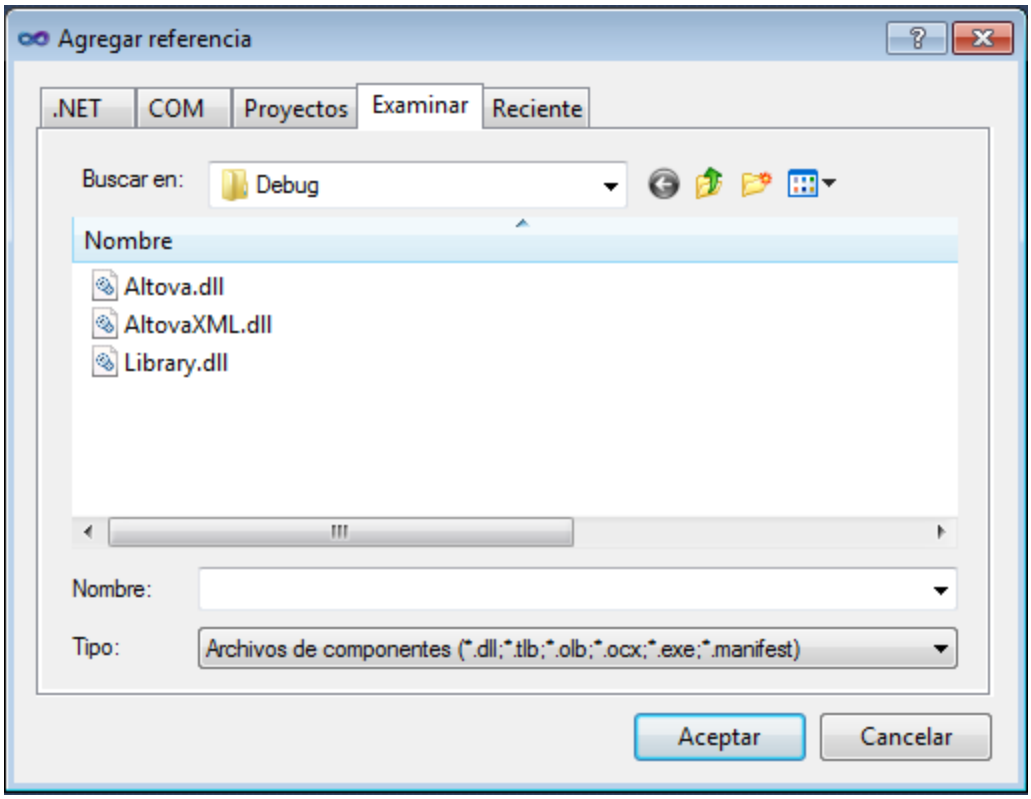
Para usar las bibliotecas de Altova en su proyecto basta con hacerles referencia o incluirlas en el proyecto (ver más abajo).

C#

Para integrar las bibliotecas de Altova en un proyecto C# actual:

1. Después de generar código a partir de un esquema con MapForce (p. ej. a partir de **YourSchema.xsd**), compile la solución **YourSchema.sln** que acaba de generar en Visual Studio. Esta solución está en una carpeta de proyecto que se llama igual que el esquema.
2. Haga clic con el botón derecho en el proyecto en Visual Studio y seleccione **Agregar referencia**.
3. En la pestaña *Examinar* busque las bibliotecas **Altova.dll**, **AltovaXML.dll** y **YourSchema.dll**, ubicadas en el directorio de salida de los proyectos generados (p. ej. **bin\Debug**).

 © 2016-2022 Altova GmbH



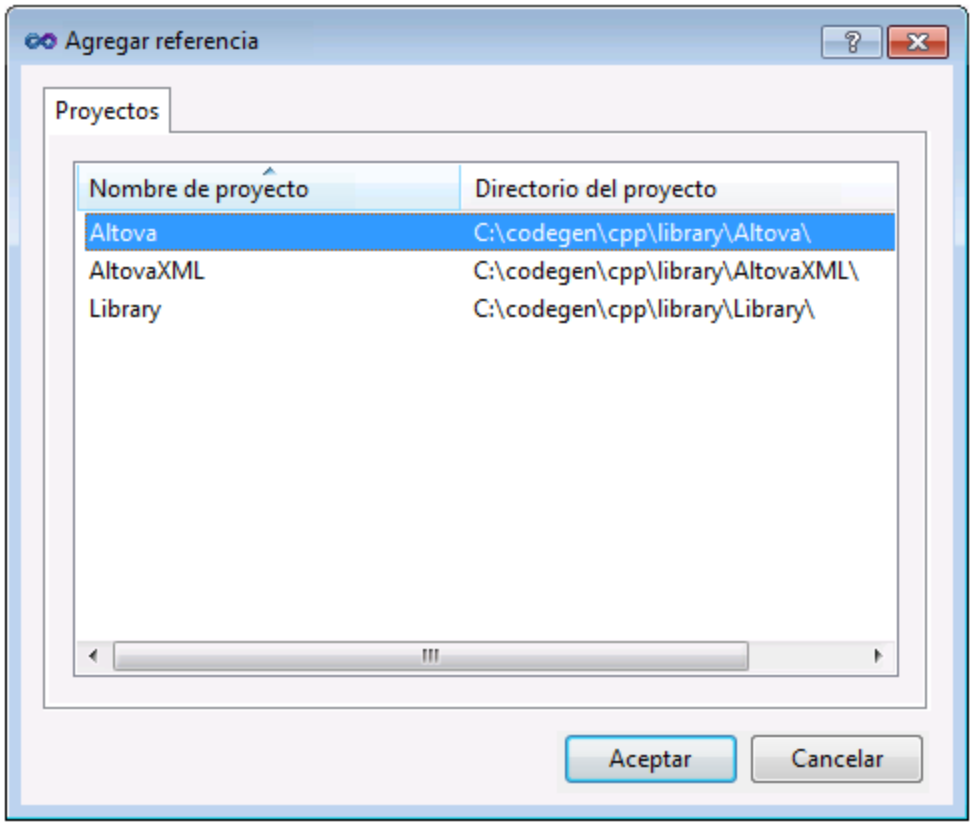
C++

La forma más sencilla de integrar bibliotecas en un proyecto C++ es agregar a la solución los archivos de proyecto generados. Por ejemplo, imagine que generó código a partir de un esquema llamado **Library.xsd** y que seleccionó el directorio de destino **c:\codegen\cpp\library**. Las bibliotecas generadas en este caso están en estas rutas de acceso:

- c:\codegen\cpp\library\Altova.vcxproj
- c:\codegen\cpp\library\AltovaXML\AltovaXML.vcxproj
- c:\codegen\cpp\library\Library.vcxproj

Primero abra la solución **c:\codegen\cpp\library\Library.sln** y compílela en Visual Studio.

Después abra su solución de Visual Studio (en Visual Studio 2010, por ejemplo). Haga clic con el botón derecho en la solución y seleccione **Agregar | Proyecto existente**. Ahora añada uno a uno los archivos de proyecto enumerados más arriba. Tenga paciencia mientras Visual Studio analiza los archivos. Después haga clic con el botón derecho en el proyecto y seleccione **Propiedades**. En el cuadro de diálogo "Páginas de propiedades" seleccione **Propiedades comunes | Framework y referencias** y después haga clic en **Agregar referencia nueva**. Añada estos proyectos: *Altova*, *AltovaXML* y *Library*.



Si necesita ayuda a la hora de usar las funciones de una biblioteca personal consulte la documentación de MSDN:

- Si tiene pensado generar bibliotecas estáticas, consulte https://msdn.microsoft.com/es-es/library/ms235627(v=vs.100).aspx
- Si tiene pensado generar bibliotecas dinámicas, consulte https://msdn.microsoft.com/es-es/library/ms235636(v=vs.100).aspx

La opción de generar bibliotecas estáticas o dinámicas se encuentra entre las opciones de generación de código (*véase Opciones del generador de código* 1207).

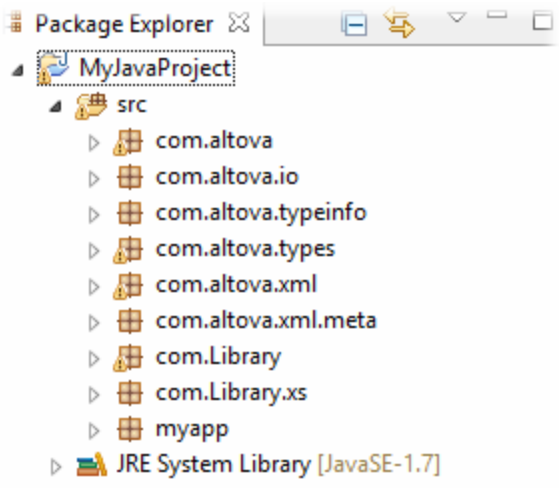
Java

Puede integrar sus paquetes de Altova en su proyecto Java copiando el directorio **com** del código generado en el directorio que almacena los paquetes fuente de su proyecto Java (p. ej. **C:\Workspace\MyJavaProject\src**). Por ejemplo, imagine que generó código en el directorio **c:\codegen\java\library**. Las clases de Altova generadas en este caso están en **c:\codegen\java\library\com**.

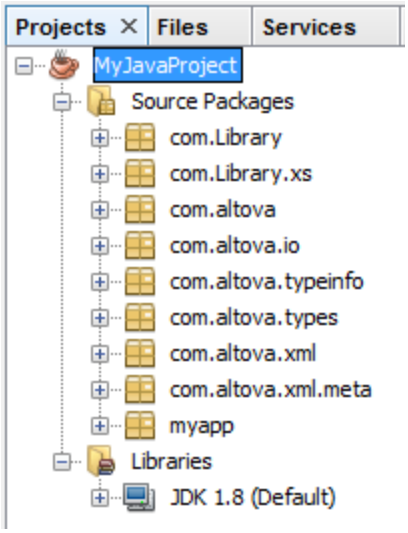
Tras copiar las bibliotecas actualice el proyecto (en Eclipse seleccione el proyecto en el explorador de paquetes y pulse **F5. En NetBeans IDE 8.0 seleccione el comando de menú Source | Scan for External Changes).**

 © 2016-2022 Altova GmbH

Una vez terminada la operación de copia, los paquetes de Altova aparecerán en el explorador de paquetes (en Eclipse) o bajo *Source Packages* en el panel *Proyects* (en NetBeans IDE).



Paquetes de Altova en Eclipse 4.4



Paquetes de Altova en NetBeans IDE 8.0.2

15.6.5 Ejemplo: biblioteca

Este ejemplo explica cómo utilizar las bibliotecas contenedoras de esquemas que se generan para escribir o leer mediante programación documentos XML conformes con el esquema. Antes de usar el código de muestra, recomendamos dedicar un tiempo a comprender la estructura del esquema de ejemplo[1124]. Este esquema es necesario para generar las bibliotecas de código que se utilizan en este ejemplo y si conoce y comprende su estructura podrá trabajar más fácilmente con el código que se genera cuando necesite adaptarlo a sus requisitos personales.

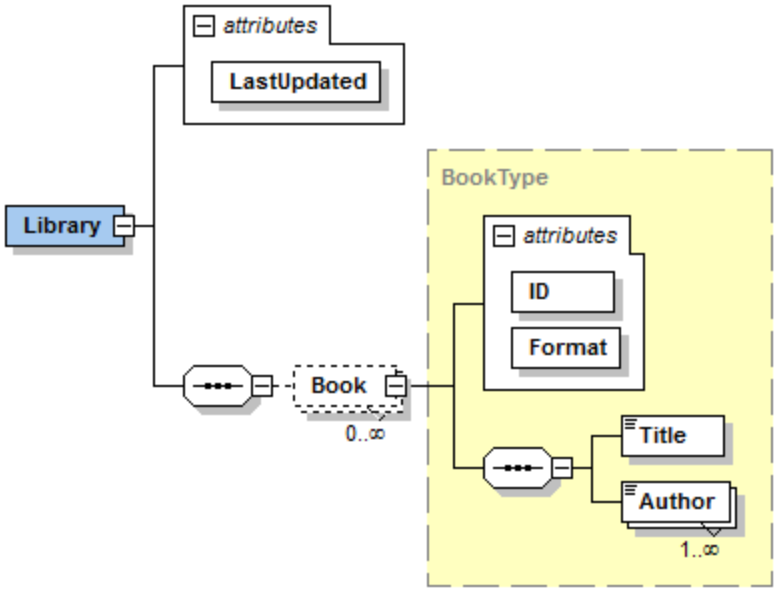
15.6.5.1 Ejemplo de esquema

El esquema que se utiliza en este ejemplo describe una biblioteca de libros. Su definición completa aparece más abajo. Si quiere obtener los mismos resultados que se obtienen en esta sección, guarde este fragmento de código como **Doc.xsd**. Este esquema se necesita para generar las bibliotecas de código utilizadas más adelante en esta sección.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns="http://www.nanonull.com/LibrarySample"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.nanonull.com/LibrarySample" elementFormDefault="qualified"
attributeFormDefault="unqualified">
   <xs:element name="Library">
      <xs:complexType>
         <xs:sequence>
            <xs:element name="Book" type="BookType" minOccurs="0" maxOccurs="unbounded"/>
         </xs:sequence>
         <xs:attribute name="LastUpdated" type="xs:dateTime"/>
      </xs:complexType>
   </xs:element>
   <xs:complexType name="BookType">
      <xs:sequence>
         <xs:element name="Title" type="xs:string"/>
         <xs:element name="Author" type="xs:string" maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:attribute name="ID" type="xs:integer" use="required"/>
      <xs:attribute name="Format" type="BookFormatType" use="required"/>
   </xs:complexType>
   <xs:complexType name="DictionaryType">
      <xs:complexContent>
         <xs:extension base="BookType">
            <xs:sequence>
               <xs:element name="FromLang" type="xs:string"/>
               <xs:element name="ToLang" type="xs:string"/>
            </xs:sequence>
         </xs:extension>
      </xs:complexContent>
   </xs:complexType>
   <xs:simpleType name="BookFormatType">
      <xs:restriction base="xs:string">
         <xs:enumeration value="Hardcover"/>
         <xs:enumeration value="Paperback"/>
         <xs:enumeration value="Audiobook"/>
         <xs:enumeration value="E-book"/>
      </xs:restriction>
   </xs:simpleType>
</xs:schema>
```

Library es un elemento raíz de un `complexType` cuya representación gráfica aparece a continuación (tomada de la vista Esquema de XMLSpy):

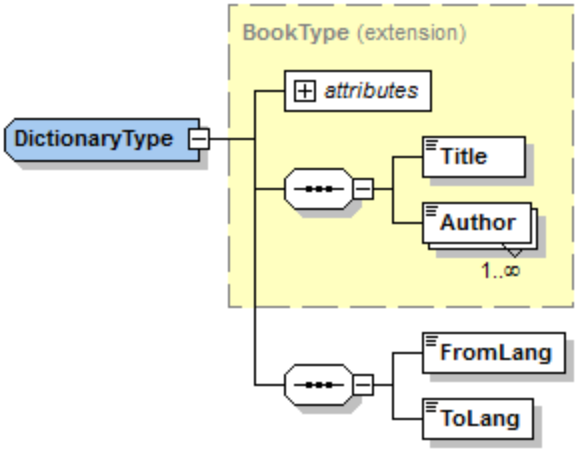
 © 2016-2022 Altova GmbH



Como puede ver, la biblioteca tiene un atributo **LastUpdated** (definido comos `xs:dateTime`) y almacena una secuencia de libros. Cada libro es un `xs:complexType` y tiene dos atributos: **ID** (definido como `xs:integer`) y **Format**. El formato de los libros puede ser tapa dura (`hardcover`), bolsillo (`paperback`), audiolibro (`audiobook`) o libro electrónico (`e-book`). En el esquema el atributo **Format** está definido como `xs:simpleType` que usa una enumeración de los mencionados valores.

Además cada libro tiene un elemento **Title** (definido como `xs:string`) y como mínimo un elemento **Author** (definido como `xs:string`).

La biblioteca también puede incluir libros que son diccionarios. Estos tienen el tipo `DictionaryType`, que se deriva por extensión del tipo `BookType`. Es decir, un diccionario hereda todos los atributos y elementos de un libro y además toma dos elementos más: **FromLang** y **ToLang** (*imagen siguiente*).



Los elementos **FromLang** y **ToLang** almacenan el idioma de partida y de destino del diccionario.

Veamos a continuación una instancia XML válida de acuerdo con el esquema anterior (siempre y cuando esté en el mismo directorio que el esquema):

```
<?xml version="1.0" encoding="utf-8"?>
<Library xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://www.nanonull.com/LibrarySample"
xsi:schemaLocation="http://www.nanonull.com/LibrarySample Library.xsd" LastUpdated="2016-
02-03T17:10:08.4977404">
   <Book ID="1" Format="E-book">
      <Title>The XMLSpy Handbook</Title>
      <Author>Altova</Author>
   </Book>
   <Book ID="2" Format="Paperback" xmlns:n1="http://www.nanonull.com/LibrarySample"
xsi:type="n1:DictionaryType">
      <Title>English-German Dictionary</Title>
      <Author>John Doe</Author>
      <FromLang>English</FromLang>
      <ToLang>German</ToLang>
   </Book>
</Library>
```

En el apartado siguiente explicamos cómo leer o escribir en dicho archivo mediante programación. Para empezar es necesario generar el código contenedor del esquema (siguiendo las instrucciones del apartado Generar código a partir de esquemas XML o documentos DTD 1110).

15.6.5.2 Leer y escribir documentos XML (C++)

Tras generarse el código a partir del esquema de ejemplo 1124 se crea una aplicación de prueba C++ junto con varias bibliotecas Altova secundarias.

Información sobre las bibliotecas C++ generadas

La clase central del código generado es la clase `CDoc`, que representa el documento XML. Dicha clase se genera por cada esquema y su nombre depende del nombre del archivo de esquema (en nuestro ejemplo es **Doc.xsd**). Como puede verse en el diagrama, esta clase aporta métodos para cargar documentos desde archivos, secuencias binarias o cadenas de texto (o para guardar documentos en archivos, secuencias y cadenas). Para ver una descripción de los miembros que expone esta clase consulte la referencia de la clase (`[SuEsquema]::[CDoc]` 1166).

 © 2016-2022 Altova GmbH

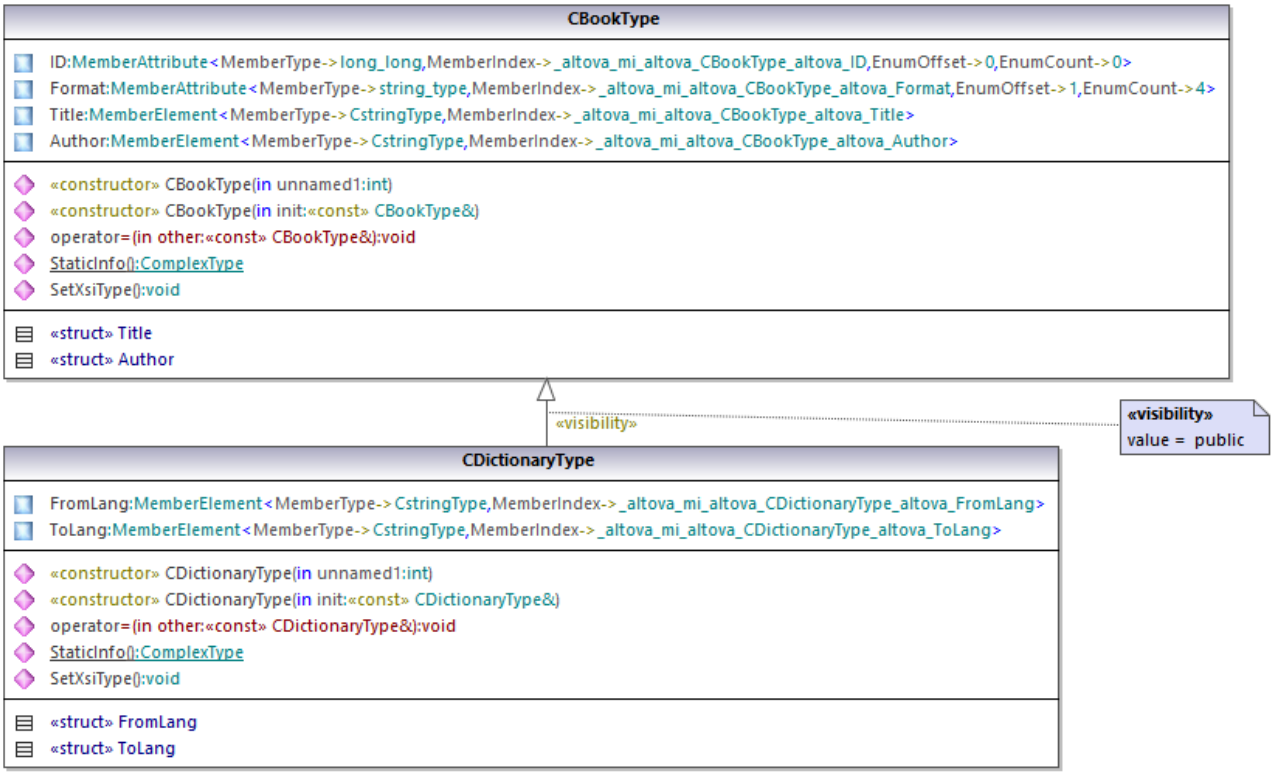
```
CDoc
Library:MemberElement<MemberType->CLibraryType,MemberIndex->_altova_mi_altova_CDoc_altova_Library>
«constructor» CDoc(in unnamed1:int)
«constructor» CDoc(in init:«const» CDoc&)
operator=(in other:«const» CDoc&):void
StaticInfo():ComplexType
SetXsiType():void
LoadFromFile(in fileName:«const» string_type&):CDoc
LoadFromString(in xml:«const» string_type&):CDoc
LoadFromBinary(in data:«const» vector<_Ty->unsigned_char>&):CDoc
SaveToFile(in fileName:«const» string_type&, in prettyPrint:bool):void
SaveToFile(in fileName:«const» string_type&, in prettyPrint:bool, in omitXmlDecl:bool):void
SaveToFile(in fileName:«const» string_type&, in prettyPrint:bool, in encoding:«const» string_type&):void
SaveToFile(in fileName:«const» string_type&, in prettyPrint:bool, in omitXmlDecl:bool, in encoding:«const» string_type&):void
SaveToFile(in fileName:«const» string_type&, in prettyPrint:bool, in encoding:«const» string_type&, in bBigEndian:bool, in bBOM:bool):void
SaveToFile(in fileName:«const» string_type&, in prettyPrint:bool, in omitXmlDecl:bool, in encoding:«const» string_type&, in bBigEndian:bool, in bBOM:bool):void
SaveToString(in prettyPrint:bool):string_type
SaveToString(in prettyPrint:bool, in omitXmlDecl:bool):string_type
SaveToBinary(in prettyPrint:bool):vector<_Ty->unsigned_char>
SaveToBinary(in prettyPrint:bool, in encoding:«const» string_type&):vector<_Ty->unsigned_char>
SaveToBinary(in prettyPrint:bool, in encoding:«const» string_type&, in bBigEndian:bool, in bBOM:bool):vector<_Ty->unsigned_char>
CreateDocument():CDoc
DestroyDocument():void
SetDTDLocation(in dtdLocation:«const» string_type&):void
SetSchemaLocation(in schemaLocation:«const» string_type&):void
DeclareAllNamespacesFromSchema(in node:TypeBase&):void
...
```

El campo `Library` de la clase `CDoc` representa la raíz real del documento. **Library** es un elemento del archivo XML, de modo que en el código C++ tiene una clase plantilla como tipo (`MemberElement`). La clase plantilla expone métodos y propiedades para interactuar con el elemento **Library**. En general, cada atributo y cada elemento de un tipo del esquema recibe un tipo en el código generado con las clases plantilla `MemberAttribute` y `MemberElement` respectivamente. Para más información consulte la referencia de las clases `[SuEsquema]::AtributoMiembro` 1170 y `[SuEsquema]::ElementoMiembro` 1170.

La clase `CLibraryType` se genera a partir del tipo complejo **LibraryType** del esquema. Observe que la clase `CLibraryType` contiene los campos `Book` y `LastUpdated`. Según la lógica mencionada previamente, estos campos corresponden al elemento **Book** y al atributo **LastUpdated** del esquema y permiten manipular elementos y atributos del documento XML de instancia mediante programación (operaciones anexar, eliminar, etc.).

```
CLibraryType
LastUpdated:MemberAttribute<MemberType->DateTime,MemberIndex->_altova_mi_altova_CLibraryType_altova_LastUpdated,EnumOffset->0,EnumCount->0>
Book:MemberElement<MemberType->CBookType,MemberIndex->_altova_mi_altova_CLibraryType_altova_Book>
«constructor» CLibraryType(in unnamed1:int)
«constructor» CLibraryType(in init:«const» CLibraryType&)
operator=(in other:«const» CLibraryType&):void
StaticInfo():ComplexType
«struct» Book
```

Como **DictionaryType** es un tipo complejo derivado de **BookType** esta relación también se refleja en las clases generadas. Como puede verse en el diagrama, la clase `CDictionaryType` hereda la clase `CBookType`.



Si su esquema XML define tipos simples como enumeraciones, los valores enumerados están disponibles como valores `enum` en el código generado. En el esquema utilizado en este ejemplo, el formato de los libros puede ser tapa dura, bolsillo, libro electrónico y audiolibro. Por tanto, en el código generado estos valores estarán disponibles a través de un `enum` que es miembro de la clase `CBookFormatType`.

Escribir un documento XML

1. En Visual Studio abra la solución **LibraryTest.sln** que se generó a partir del esquema **Library** mencionado anteriormente.

> Cuando cree prototipos de aplicaciones a partir de esquemas XML que cambien con frecuencia, a veces será necesario generar código una y otra vez en el mismo directorio para que los cambios en el esquema se reflejen inmediatamente en el código. Recuerde que la aplicación de prueba que se genera y las bibliotecas de Altova se sobrescribirán cada vez que genere código en el mismo directorio de destino. Por tanto, recuerde que no debe añadir código a la aplicación de prueba que se genera, sino que debe integrar las bibliotecas de Altova en el proyecto (ver *Integrar bibliotecas contenedoras de esquemas* 1120).

2. En el explorador de soluciones abra el archivo **LibraryTest.cpp** y edite el método `Example()` como se indica a continuación:

```
#include <ctime> // obligatorio para obtener hora actual
using namespace Library; // obligatorio para trabajar con bibliotecas de Altova
```

 © 2016-2022 Altova GmbH

```
void Example()
{
   // Crear un documento XML nuevo vacío
   CDoc libDoc = CDoc::CreateDocument();

   // Crear el elemento raíz <Library> y agregarlo al documento
   CLibraryType lib = libDoc.Library.append();

   // Obtener la hora actual y establecer el atributo "LastUpdated" usando clases de
Altova
   time_t t = time(NULL);
   struct tm * now = localtime( & t );
   altova::DateTime dt = altova::DateTime(now->tm_year + 1900, now->tm_mon + 1, now-
>tm_mday, now->tm_hour, now->tm_min, now->tm_sec);
   lib.LastUpdated = dt;

   // Crear un nuevo <Book> y agregarlo a la biblioteca
   CBookType book = lib.Book.append();

   // Establecer el atributo "ID" del libro
   book.ID = 1;

   // Establecer el atributo "Format" del <Book> usando una constante de enumeración
   book.Format.SetEnumerationValue( CBookFormatType::k_Paperback );

   // Agregar los elementos <Title> y <Author> y establecer valores
   book.Title.append() = _T("The XML Spy Handbook");
   book.Author.append() = _T("Altova");

   // Anexar un diccionario (libro de tipo derivado) y rellenar sus atributos y elementos
   CDictionaryType dictionary = CDictionaryType(lib.Book.append().GetNode());
   dictionary.ID = 2;
   dictionary.Format.SetEnumerationValue( CBookFormatType::k_E_book);
   dictionary.Title.append() = _T("English-German Dictionary");
   dictionary.Author.append() = _T("John Doe");
   dictionary.FromLang.append() = _T("English");
   dictionary.ToLang.append() = _T("German");

   // Como diccionario es un tipo derivado, establecer el atributo xsi:type del elemento
book
   dictionary.SetXsiType();

   // De forma opcional establecer la ubicación del esquema
   libDoc.SetSchemaLocation(_T("Library.xsd"));

   // Guardar el documento XML en un archivo con la codificación predeterminada(UTF-8)
   // "true" causes the file to be pretty-printed.
   libDoc.SaveToFile(_T("GeneratedLibrary.xml"), true);

   // Destruir el documento
   libDoc.DestroyDocument();
}
```

3. Pulse **F5** para iniciar la depuración. Si el código se ejecuta correctamente, el archivo **GeneratedLibrary.xml** se crea en el directorio de salida de la solución.

Leer un documento XML

1. Abra la solución **LibraryTest.sln** en Visual Studio.
2. Guarde el código que aparece a continuación como **Library1.xml** en un directorio que el código de programa pueda leer (p. ej. en el mismo directorio que **LibraryTest.sln**).

```
<?xml version="1.0" encoding="utf-8"?>
<Library xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://www.nanonull.com/LibrarySample"
xsi:schemaLocation="http://www.nanonull.com/LibrarySample Library.xsd" LastUpdated="2016-
02-03T17:10:08.4977404">
   <Book ID="1" Format="E-book">
      <Title>The XMLSpy Handbook</Title>
      <Author>Altova</Author>
   </Book>
   <Book ID="2" Format="Paperback" xmlns:n1="http://www.nanonull.com/LibrarySample"
xsi:type="n1:DictionaryType">
      <Title>English-German Dictionary</Title>
      <Author>John Doe</Author>
      <FromLang>English</FromLang>
      <ToLang>German</ToLang>
   </Book>
</Library>
```

3. En el explorador de soluciones abra el archivo **LibraryTest.cpp** y edite el método `Example()` como se indica a continuación.

```
using namespace Doc;
void Example()
{
   // Cargar documento XML
   CDoc libDoc = CDoc::LoadFromFile(_T("Library1.xml"));

   // Obtener el primer y único elemento raíz <Library>
   CLibraryType lib = libDoc.Library.first();

   // Compruebe si existe un elemento:
   if (!lib.Book.exists())
   {
      tcout << "This library is empty." << std::endl;
      return;
   }

   // iteración: por cada <Book>...
   for (Iterator<CBookType> itBook = lib.Book.all(); itBook; ++itBook)
   {
      // valores de salida del atributo ISBN y del primer y único elemento title
      tcout << "ID: " << itBook->ID << std::endl;
      tcout << "Title: " << tstring(itBook->Title.first()) << std::endl;
```

 © 2016-2022 Altova GmbH

```
      // leer y compara un valor de enumeración
      if (itBook->Format.GetEnumerationValue() == CBookFormatType::k_Paperback)
         tcout << "This is a paperback book." << std::endl;

      // por cada <Author>...
      for (CBookType::Author::iterator itAuthor = itBook->Author.all(); itAuthor; +
+itAuthor)
         tcout << "Author: " << tstring(itAuthor) << std::endl;

      // alternativa: usar recuento e índice
      for (unsigned int j = 0; j < itBook->Author.count(); ++j)
         tcout << "Author: " << tstring(itBook->Author[j]) << std::endl;
   }

   // Destruir el documento
   libDoc.DestroyDocument();
}
```

4. Pulse **F5** para iniciar la depuración.

15.6.5.3 Leer y escribir documentos XML (C#)

Tras generarse el código a partir del esquema de ejemplo 1124 se crea una aplicación de prueba C# junto con varias bibliotecas Altova secundarias.

Información sobre las bibliotecas C# generadas

La clase central del código generado es la clase `Doc2`, que representa el documento XML. Dicha clase se genera por cada esquema y su nombre depende del nombre del archivo de esquema (en nuestro ejemplo es **Doc.xsd**). Observe que esta clase se llama `Doc2` para evitar conflictos entre los nombres de espacio de nombres. Como puede verse en el diagrama, esta clase aporta métodos para cargar documentos desde archivos, secuencias binarias o cadenas de texto (o para guardar documentos en archivos, secuencias y cadenas). Para ver una descripción de esta clase consulte la referencia de la clase (`[SuEsquema].[Doc]` 1182).

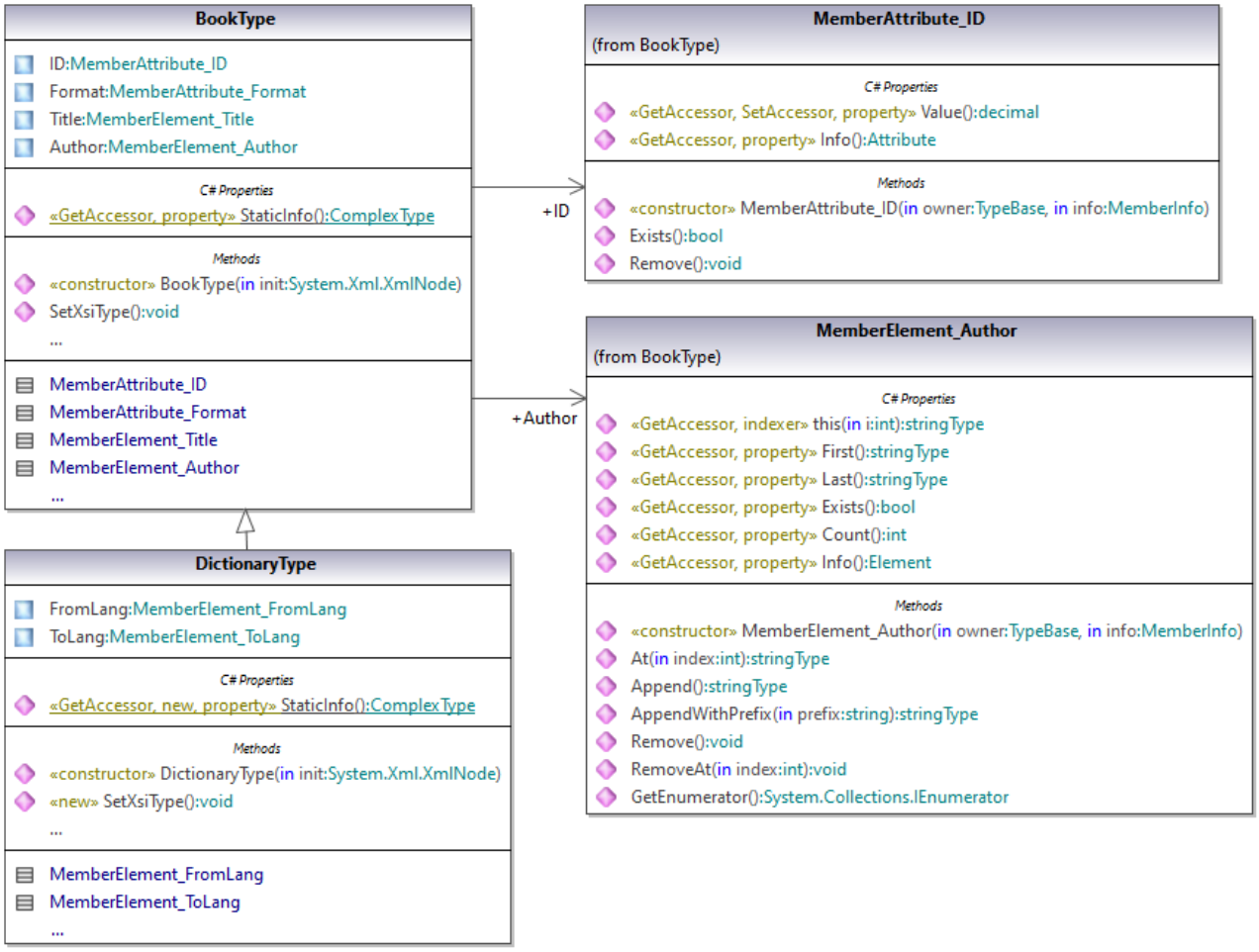
```
Doc2
Library:MemberElement_Library
C# Properties
«GetAccessor, property» StaticInfo():ComplexType
Methods
LoadFromFile(in filename:string):Doc2
LoadFromString(in xmlstring:string):Doc2
LoadFromBinary(in binary:byte[*]):Doc2
SaveToFile(in filename:string, in prettyPrint:bool):void
SaveToFile(in filename:string, in prettyPrint:bool, in omitXmlDecl:bool):void
SaveToFileWithLineEnd(in filename:string, in prettyPrint:bool, in omitXmlDecl:bool, in lineend:string):void
SaveToFile(in filename:string, in prettyPrint:bool, in omitXmlDecl:bool, in encoding:string):void
SaveToFile(in filename:string, in prettyPrint:bool, in encoding:string, in lineend:string):void
SaveToFile(in filename:string, in prettyPrint:bool, in omitXmlDecl:bool, in encoding:string, in lineend:string):void
SaveToFile(in filename:string, in prettyPrint:bool, in omitXmlDecl:bool, in encoding:string, in bBigEndian:bool, in bBOM:bool, in lineend:string):void
SaveToString(in prettyPrint:bool):string
SaveToString(in prettyPrint:bool, in omitXmlDecl:bool):string
SaveToBinary(in prettyPrint:bool):byte[*]
SaveToBinary(in prettyPrint:bool, in encoding:string):byte[*]
SaveToBinary(in prettyPrint:bool, in encoding:string, in bBigEndian:bool, in bBOM:bool):byte[*]
CreateDocument():Doc2
CreateDocument(in encoding:string):Doc2
SetDTDLocation(in dtdLocation:string):void
SetSchemaLocation(in schemaLocation:string):void
DeclareAllNamespacesFromSchema(in node:TypeBase):void
«constructor» Doc2(in init:System.Xml.XmlNode)
SetXsiType():void
...
```

El miembro `Library` de la clase `Doc2` representa la raíz real del documento.

De acuerdo con las reglas de generación de código mencionadas en el apartado Bibliotecas contenedoras de esquemas (C#)[1115], se generan clases miembro por cada atributo y por cada elemento de un tipo. En el código generado el nombre de dichas clases miembros va precedido por el prefijo `MemberAttribute_` y `MemberElement_` respectivamente. Son ejemplos de dichas clases: `MemberAttribute_ID` y `MemberElement_Author`, que se generan a partir del elemento **Author** y del atributo **ID** de un libro respectivamente (en el siguiente diagrama están anidadas bajo `BookType`). Dichas clases permiten manipular los correspondientes elementos y atributos del documento XML de instancia mediante programación (operaciones anexar, eliminar, establecer valor, etc.). Para más información consulte la referencia de las clases `[SuTipoEsquema].MemberAttribute`[1185] y `[SuTipoEsquema].MemberElement`[1186].

Como en el esquema **DictionaryType** es un tipo complejo derivado de **BookType**, esta relación también se refleja en las clases generadas. Como puede verse en el diagrama siguiente, la clase `DictionaryType` hereda la clase `BookType`.

© 2016-2022 Altova GmbH



Si su esquema XML define tipos simples como enumeraciones, los valores enumerados están disponibles como valores `Enum` en el código generado. En el esquema utilizado en este ejemplo, el formato de los libros puede ser tapa dura, bolsillo, libro electrónico y audiolibro. Por tanto, en el código generado estos valores estarán disponibles a través de un `Enum` que es miembro de la clase `BookFormatType`.

Escribir un documento XML

1. En Visual Studio abra la solución **LibraryTest.sln** que se generó a partir del esquema Library mencionado anteriormente.

> Cuando cree prototipos de aplicaciones a partir de esquemas XML que cambien con frecuencia, a veces será necesario generar código una y otra vez en el mismo directorio para que los cambios en el esquema se reflejen inmediatamente en el código. Recuerde que la aplicación de prueba que se genera y las bibliotecas de Altova se sobrescribirán cada vez que genere código en el mismo directorio de destino. Por tanto, recuerde que no debe añadir código a la aplicación de prueba que se genera, sino que debe integrar las bibliotecas de Altova en el proyecto (ver *Integrar bibliotecas contenedoras de esquemas* 1120).

2. En el explorador de soluciones abra el archivo **LibraryTest.cs** y edite el método `Example()` como se indica a continuación:

```
        protected static void Example()
        {
            // Crear una biblioteca XML nueva
            Library2 doc = Library2.CreateDocument();
            // Anexar el elemento raíz
            LibraryType root = doc.Library3.Append();

            // Crear la fecha de generación de biblioteca con ayuda de la clase DateTime
de Altova
            Altova.Types.DateTime dt = new Altova.Types.DateTime(System.DateTime.Now);
            // Anexar la fecha a la raíz
            root.LastUpdated.Value = dt;

            // Agregar un libro nuevo
            BookType book = root.Book.Append();
            // Establecer el valor del atributo ID
            book.ID.Value = 1;
            // Establecer el formato del libro (enumeración)
            book.Format.EnumerationValue = BookFormatType.EnumValues.eHardcover;
            // Establecer los elementos Title y Author
            book.Title.Append().Value = "The XMLSpy Handbook";
            book.Author.Append().Value = "Altova";

            // Anexar un diccionario (libro de tipo derivado) y rellenar sus atributos y
elementos
            DictionaryType dictionary = new DictionaryType(root.Book.Append().Node);
            dictionary.ID.Value = 2;
            dictionary.Title.Append().Value = "English-German Dictionary";
            dictionary.Format.EnumerationValue = BookFormatType.EnumValues.eE_book;
            dictionary.Author.Append().Value = "John Doe";
            dictionary.FromLang.Append().Value = "English";
            dictionary.ToLang.Append().Value = "German";
            // Como es un tipo derivado, establecer el atributo xsi:type del elemento
book
            dictionary.SetXsiType();

            // De forma opcional, establecer la ubicación del esquema (ajustar la ruta de
acceso
            // si el esquema no está en la misma carpeta que el archivo de instancia que
se generó)
            doc.SetSchemaLocation("Library.xsd");

            // Guardar el documento XML con la opción pretty print habilitada
            doc.SaveToFile("GeneratedLibrary.xml", true);
        }
```

3. Pulse **F5** para iniciar la depuración. Si el código se ejecuta correctamente, el archivo **GeneratedLibrary.xml** se crea en el directorio de salida de la solución (que suele ser **bin/Debug**).

 © 2016-2022 Altova GmbH

Leer un documento XML

1. Abra la solución **LibraryTest.sln** en Visual Studio.
2. Guarde el código que aparece a continuación en un archivo llamado **Library.xml** en el directorio de salida del proyecto (predeterminado: **bin/Debug**). Este es el archivo que leerá el código de programa.

```
<?xml version="1.0" encoding="utf-8"?>
<Library xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://www.nanonull.com/LibrarySample"
xsi:schemaLocation="http://www.nanonull.com/LibrarySample Library.xsd" LastUpdated="2016-
02-03T17:10:08.4977404">
   <Book ID="1" Format="E-book">
      <Title>The XMLSpy Handbook</Title>
      <Author>Altova</Author>
   </Book>
   <Book ID="2" Format="Paperback" xmlns:n1="http://www.nanonull.com/LibrarySample"
xsi:type="n1:DictionaryType">
      <Title>English-German Dictionary</Title>
      <Author>John Doe</Author>
      <FromLang>English</FromLang>
      <ToLang>German</ToLang>
   </Book>
</Library>
```

3. En el explorador de soluciones abra el archivo **LibraryTest.cs** y edite el método `Example()` como se indica a continuación:

```
protected static void Example()
{
    // Cargar el archivo XML en una instancia nueva de Library
    Doc2 doc = Doc2.LoadFromFile("Library.xml");
    // Obtener el elemento raíz
    LibraryType root = doc.Library.First;

    // Leer la fecha de generación de Library
    Altova.Types.DateTime dt = root.LastUpdated.Value;
    string dt_as_string = dt.ToString(DateTimeFormat.W3_dateTime);
    Console.WriteLine("The library generation date is: " + dt_as_string);

    // Iteración: por cada <Book>...
    foreach (BookType book in root.Book)
    {
      // Valores de salida del atributo ID y del (primer y único) elemento title
      Console.WriteLine("ID:    " + book.ID.Value);
      Console.WriteLine("Title: " + book.Title.First.Value);

      // Leer y comparar un valor de enumeración
      if (book.Format.EnumerationValue == BookFormatType.EnumValues.ePaperback)
         Console.WriteLine("This is a paperback book.");
```

```
      // Iteración: por cada <Author>
      foreach (xs.stringType author in book.Author)
         Console.WriteLine("Author: " + author.Value);

      // Determinar si se trata de un libro de tipo derivado
      if (book.Node.Attributes.GetNamedItem("xsi:type") != null)
      {
         // Buscar el valor del atributo xsi:type
         string xsiTypeValue = book.Node.Attributes.GetNamedItem("xsi:type").Value;
         // Obtener el URI de espacio de nombres
        // y el prefijo de consulta de este espacio de nombres
         string namespaceUri = book.Node.NamespaceURI;
         string prefix = book.Node.GetPrefixOfNamespace(namespaceUri);

         // si este libro tiene DictionaryType
         if (namespaceUri == "http://www.nanonull.com/LibrarySample" &&
xsiTypeValue.Equals(prefix + ":DictionaryType"))
         {
            // generar campos adicionales
            DictionaryType dictionary = new DictionaryType(book.Node);
            Console.WriteLine("Language from: " + dictionary.FromLang.First.Value);
            Console.WriteLine("Language to: " + dictionary.ToLang.First.Value);
         }
         else
         {
            throw new Exception("Unexpected book type");
         }
      }
    }

    Console.ReadLine();
}
```

4. Pulse **F5** para iniciar la depuración. Si el código se ejecuta correctamente, el código de programa leerá el archivo **GeneratedLibrary.xml** y su contenido aparecerá en el resultado de la consola.

Leer y escribir elementos y atributos

El acceso a los atributos y elementos se consigue con la propiedad `Value` de la clase de elemento o atributo miembro que se genera. Por ejemplo:

```
// Valores de salida del atributo ID y del (primer y único) elemento title
Console.WriteLine("ID:    " + book.ID.Value);
Console.WriteLine("Title: " + book.Title.First.Value);
```

Para obtener el valor del elemento **Title** en este ejemplo concreto también se utilizó el método `First()`. Esto se debe a que este es el primer (y único) elemento **Title** de un libro. Cuando necesite seleccionar un elemento concreto de una lista por medio del índice, utilice el método `At()`.

 © 2016-2022 Altova GmbH

La clase que se genera por cada elemento miembro de un tipo implementa la interfaz estándar `System.Collections.IEnumerable`. Esto permite recorrer varios elementos del mismo tipo. En este ejemplo concreto puede recorrer todos los libros de un objeto Library como se puede ver a continuación:

```
// Iteración: por cada <Book>...
foreach (BookType book in root.Book)
{
    // your code here...
}
```

Para agregar un elemento nuevo utilice el método `Append()`. Por ejemplo, a continuación se anexa el elemento raíz al documento:

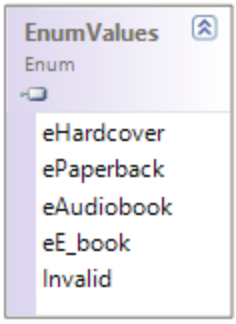
```
// Anexar el elemento raíz a la biblioteca
LibraryType root = doc.Library.Append();
```

Puede establecer el valor de un atributo (como ID) de la siguiente manera:

```
// Establecer el valor del atributo ID
book.ID.Value = 1;
```

Leer y escribir valores de enumeración

Si el esquema XML define tipos simples como enumeraciones, los valores enumerados estarán disponibles como valores `Enum` en el código generado. En el esquema utilizado en este ejemplo el formato de un libro puede ser tapa dura, bolsillo, libro electrónico y audiolibro. Por tanto, en el código generado estos valores estarán disponibles a través de una `Enum`:



Para asignar valores de enumeración a un objeto utilice código como este:

```
// Establecer el formato del libro (enumeración)
book.Format.EnumerationValue = BookFormatType.EnumValues.eHardcover
```

Puede leer dichos valores de enumeración desde documento XML de instancia de la siguiente manera:

```
// Leer y comparar un valor de enumeración
if (book.Format.EnumerationValue == BookFormatType.EnumValues.ePaperback)
Console.WriteLine("This is a paperback book.");
```

Cuando la condición IF no sea suficiente, cree un modificador para determinar cada valor de enumeración y procesarlos según corresponda en cada caso.

Trabajar con tipos xs:dateTime y xs:duration

Si el esquema desde el que se genera código utiliza tipos de hora y duración como `xs:dateTime` o `xs:duration`, estos tipos se convierten en clases nativas de Altova en el código generado. Por tanto, para escribir un valor de fecha o duración en el documento XML debe seguir estas instrucciones:

1. Construya un objeto `Altova.Types.DateTime`[1172] o `Altova.Types.Duration`[1177] (bien desde `System.DateTime` o utilizando partes como horas y minutos como en `Altova.Types.DateTime`[1172] y `Altova.Types.Duration`[1177]).
2. Establezca el objeto como valor del elemento o atributo que necesita. Por ejemplo:

```
// Crear la fecha de generación de la biblioteca usando la clase DateTime de Altova
Altova.Types.DateTime dt = new Altova.Types.DateTime(System.DateTime.Now);
// Anexar la fecha a la raíz
root.LastUpdated.Value = dt;
```

Para leer una fecha o duración de un documento XML:

1. Declare el valor de elemento (o atributo) como objeto `Altova.Types.DateTime`[1172] o `Altova.Types.Duration`[1177].
2. Aplique formato al elemento o atributo que necesita. Por ejemplo:

```
// Leer la fecha de generación de la biblioteca
Altova.Types.DateTime dt = root.LastUpdated.Value;
string dt_as_string = dt.ToString(DateTimeFormat.W3_dateTime);
Console.WriteLine("The library generation date is:
```

Para más información consulte la referencia de las clases `Altova.Types.DateTime`[1172] y `Altova.Types.Duration`[1177].

Trabajar con tipos derivados

Si su esquema XML define tipos derivados puede conservar la derivación de tipos en los documentos XML que cree o cargue mediante programación. Tomando el esquema utilizado en este ejemplo, el fragmento de código que aparece a continuación explica cómo se crea un libro nuevo de tipo derivado `DictionaryType`:

```
// Anexar un diccionario (libro de tipo derivado) y rellenar sus atributos y elementos
DictionaryType dictionary = new DictionaryType(root.Book.Append().Node);
dictionary.ID.Value = 2;
dictionary.Title.Append().Value = "English-German Dictionary";
dictionary.Author.Append().Value = "John Doe";
dictionary.FromLanguage.Append().Value = "English";
dictionary.ToLanguage.Append().Value = "German";
```

 © 2016-2022 Altova GmbH

```
// Como es un tipo derivado, asegúrese de establecer el atributo xsi:type del elemento
book
dictionary.SetXsiType();
```

Recuerde que es importante establecer el atributo `xsi:type` del libro recién creado. Esto garantiza que el esquema interprete correctamente el tipo de libro a la hora de validar el documento XML.

El fragmento de código que aparece a continuación identifica un libro de tipo derivado `DictionaryType` en la instancia XML que se carga. Primero se busca el valor del atributo `xsi:type` del nodo book. Si el URI de espacio de nombres de este nodo es `http://www.nanonull.com/LibrarySample` y el prefijo de consulta URI y el tipo coinciden con el valor del atributo `xsi:type`, entonces sabemos que se trata de un diccionario.

```
      // Determinar si este libro es de tipo derivado
      if (book.Node.Attributes.GetNamedItem("xsi:type") != null)
      {
          // Buscar el valor del atributo xsi:type
          string xsiTypeValue = book.Node.Attributes.GetNamedItem("xsi:type").Value;
          // Obtener el URI de espacio de nombres y el prefijo de consulta de este
espacio de nombres
          string namespaceUri = book.Node.NamespaceURI;
          string prefix = book.Node.GetPrefixOfNamespace(namespaceUri);

          // Si este libro tiene DictionaryType
          if (namespaceUri == "http://www.nanonull.com/LibrarySample" &&
xsiTypeValue.Equals(prefix + ":DictionaryType"))
          {
             // generar campos adicionales
             DictionaryType dictionary = new DictionaryType(book.Node);
             Console.WriteLine("Language from: " + dictionary.FromLang.First.Value);
             Console.WriteLine("Language to: " + dictionary.ToLang.First.Value);
          }
          else
          {
              throw new Exception("Unexpected book type");
          }
      }
```

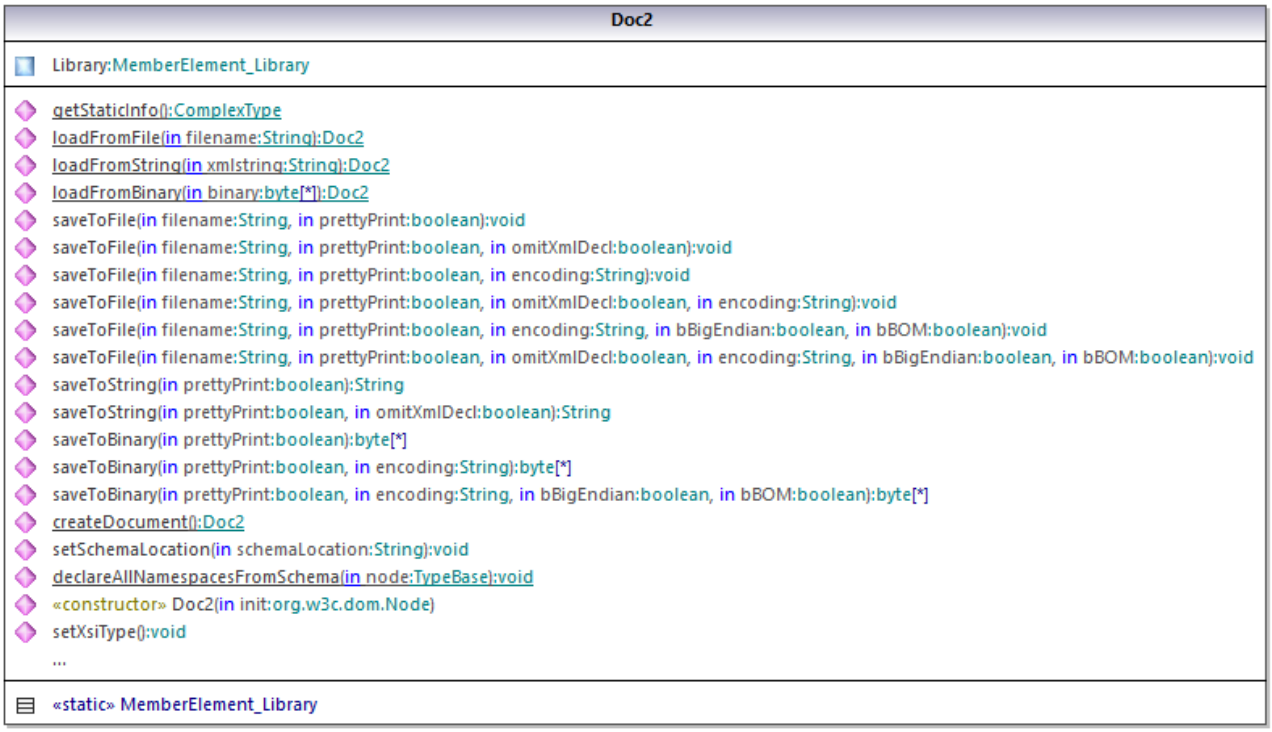
15.6.5.4 Leer y escribir documentos XML (Java)

Tras generarse el código a partir del esquema de ejemplo 1124 se crea un proyecto de prueba Java junto con varias bibliotecas Altova secundarias.

Información sobre las bibliotecas Java generadas

La clase central del código generado es la clase `Doc2`, que representa el documento XML. Dicha clase se genera por cada esquema y su nombre depende del nombre del archivo de esquema (en nuestro ejemplo es **Doc.xsd**). Observe que esta clase se llama `Doc2` para evitar conflictos entre los nombres de espacio de nombres. Como puede verse en el diagrama, esta clase aporta métodos para cargar documentos desde archivos, secuencias binarias o cadenas de texto (o para guardar documentos en archivos, secuencias y

cadenas). Para ver una descripción de esta clase consulte la referencia de la clase com.[YourSchema].[Doc] 1200.

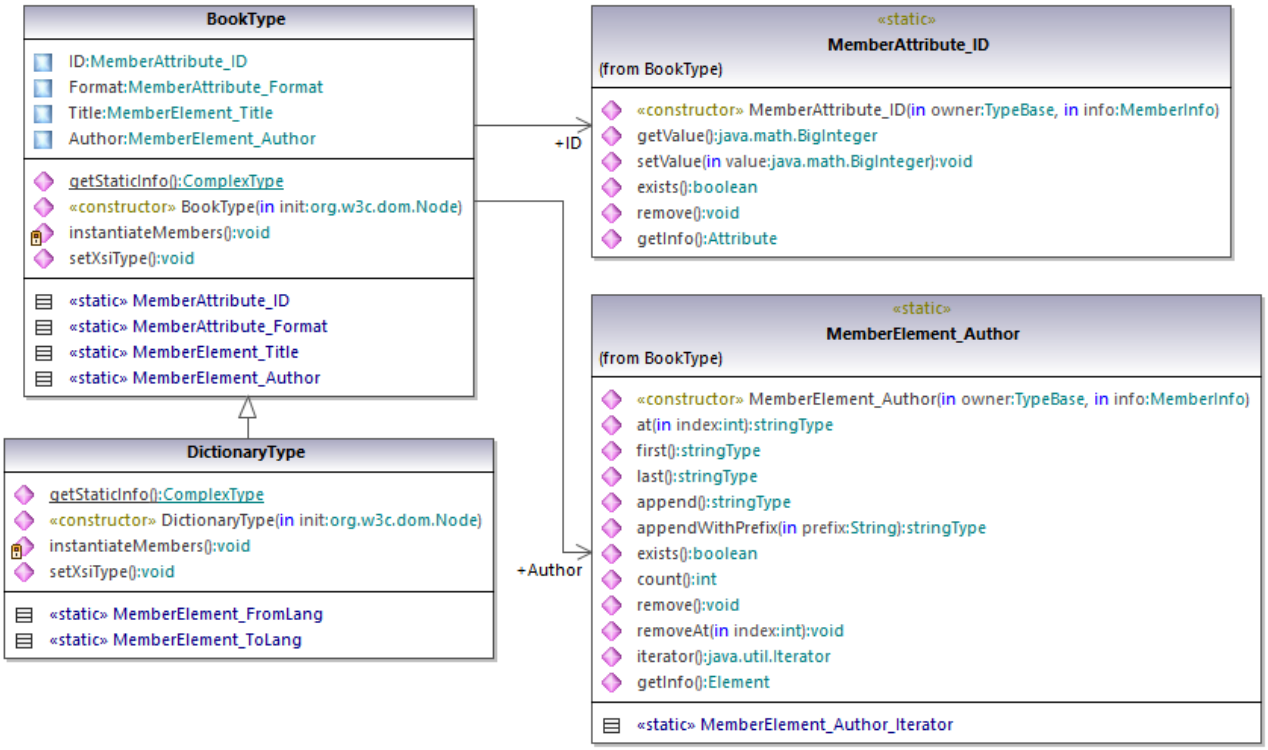


El miembro `Library` de la clase `Doc2` representa la raíz real del documento.

De acuerdo con las reglas de generación de código mencionadas en el apartado Bibliotecas contenedoras de esquemas (Java) 1118, se generan clases miembro por cada atributo y por cada elemento de un tipo. En el código generado el nombre de dichas clases miembros va precedido por el prefijo `MemberAttribute_` y `MemberElement_` respectivamente. En el diagrama anterior puede ver ejemplos de dichas clases: `MemberAttribute_ID` y `MemberElement_Author`, que se generan a partir del elemento **Author** y del atributo **ID** de un libro respectivamente. Dichas clases permiten manipular los correspondientes elementos y atributos del documento XML de instancia mediante programación (operaciones anexar, eliminar, establecer valor, etc.). Para más información consulte la referencia de las clases com.[SuEsquema].[SuTipoEsquema].MemberAttribute 1203 y com.[SuEsquema].[SuTipoEsquema].MemberElement 1204.

Como en el esquema **DictionaryType** es un tipo complejo derivado de **BookType**, esta relación también se refleja en las clases generadas. Como puede verse en el diagrama, la clase `DictionaryType` hereda la clase `BookType`.

 © 2016-2022 Altova GmbH



Si su esquema XML define tipos simples como enumeraciones, los valores enumerados están disponibles como valores `Enum` en el código generado. En el esquema utilizado en este ejemplo, el formato de los libros puede ser tapa dura, bolsillo, libro electrónico y audiolibro. Por tanto, en el código generado estos valores estarán disponibles a través de un `Enum` que es miembro de la clase `BookFormatType`.

Escribir un documento XML

1. En el menú **File** de Eclipse haga clic en el comando **Import** y seleccione **Existing Projects into Workspace** (Proyectos en el espacio de trabajo). Después haga clic en **Next** para continuar.
2. Junto a **Select root directory** (Seleccionar directorio raíz) haga clic en el botón **Browse**. Después seleccione el directorio donde generó el código Java y haga clic en **Finish** para finalizar.
3. En el explorador de paquetes de Eclipse expanda el paquete **com.LibraryTest** y abra el archivo **LibraryTest.java**.

> Cuando cree prototipos de aplicaciones a partir de esquemas XML que cambien con frecuencia, a veces será necesario generar código una y otra vez en el mismo directorio para que los cambios en el esquema se reflejen inmediatamente en el código. Recuerde que la aplicación de prueba que se genera y las bibliotecas de Altova se sobrescribirán cada vez que genere código en el mismo directorio de destino. Por tanto, recuerde que no debe añadir código a la aplicación de prueba que se genera, sino que debe integrar las bibliotecas de Altova en el proyecto (ver *Integrar bibliotecas contenedoras de esquemas* 1120).

4. Edite el método `Example()` como se indica a continuación:

```
protected static void example() throws Exception {
      // crear un documento XML nuevo vacío
      Doc2 libDoc = Doc2.createDocument();

      // crear el elemento raíz <Library> y agregarlo al documento
      LibraryType lib = libDoc.Library.append();

      // establecer el atributo "LastUpdated"
      com.altova.types.DateTime dt = new com.altova.types.DateTime(DateTime.now());
      lib.LastUpdated.setValue(dt);

      // crear un nuevo <Book> y rellenar sus elementos y atributos
      BookType book = lib.Book.append();
      book.ID.setValue(java.math.BigInteger.valueOf(1));
      book.Format.setEnumerationValue( BookFormatType.EPAPERBACK );
      book.Title.append().setValue("The XML Spy Handbook");
      book.Author.append().setValue("Altova");

      // crear un diccionario (libro de tipo derivado) y rellenar sus elementos y
atributos
      DictionaryType dict = new DictionaryType(lib.Book.append().getNode());
      dict.ID.setValue(java.math.BigInteger.valueOf(2));
      dict.Title.append().setValue("English-German Dictionary");
      dict.Format.setEnumerationValue(BookFormatType.EE_BOOK);
      dict.Author.append().setValue("John Doe");
      dict.FromLang.append().setValue("English");
      dict.ToLang.append().setValue("German");
      dict.setXsiType();

      // establecer la ubicación del esquema (opcional)
      libDoc.setSchemaLocation("Library.xsd");

      // guardar el documento XML en un archivo con la codificación predeterminada (UTF-
8). el valor "true" aplica formato pretty-print al archivo.
      libDoc.saveToFile("Library1.xml", true);
   }
```

5. Compile y ejecute el proyecto Java. Si el código se ejecuta correctamente, se crea el archivo **Library1.xml** en el directorio del proyecto.

Leer un documento XML

1. En el menú **File** de Eclipse haga clic en el comando **Import** y seleccione **Existing Projects into Workspace** (Proyectos en el espacio de trabajo). Después haga clic en **Next** para continuar.
2. Junto a **Select root directory** (Seleccionar directorio raíz) haga clic en el botón **Browse**. Después seleccione el directorio donde generó el código Java y haga clic en **Finish** para finalizar.
3. Guarde el código que aparece a continuación en un archivo llamado **Library1.xml** en un directorio local (deberá referirse a la ruta de acceso del archivo **Library1.xml** desde el código).

```
<?xml version="1.0" encoding="utf-8"?>
<Library xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
```

 © 2016-2022 Altova GmbH

```
xmlns="http://www.nanonull.com/LibrarySample"
xsi:schemaLocation="http://www.nanonull.com/LibrarySample Library.xsd" LastUpdated="2016-
02-03T17:10:08.4977404">
   <Book ID="1" Format="E-book">
      <Title>The XMLSpy Handbook</Title>
      <Author>Altova</Author>
   </Book>
   <Book ID="2" Format="Paperback" xmlns:n1="http://www.nanonull.com/LibrarySample"
xsi:type="n1:DictionaryType">
      <Title>English-German Dictionary</Title>
      <Author>John Doe</Author>
      <FromLang>English</FromLang>
      <ToLang>German</ToLang>
   </Book>
</Library>
```

4. En el explorador de paquetes de Eclipse expanda el paquete **com.LibraryTest** y abra el archivo **LibraryTest.java**.
5. Edite el método `Example()` como se indica a continuación:

```
protected static void example() throws Exception {
     // cargar documento XML desde una ruta de acceso (ajustar la ruta si procede)
     Doc2 libDoc = Doc2.loadFromFile("Library1.xml");

     // obtener el primer (y único) elemento raíz <Library>
     LibraryType lib = libDoc.Library.first();

     // comprobar si existe un elemento:
     if (!lib.Book.exists()) {
        System.out.println("This library is empty.");
        return;
     }

     // leer el tipo DateTime del esquema
     com.altova.types.DateTime dt = lib.LastUpdated.getValue();
     System.out.println("The library was last updated on: " + dt.toDateString());

     // iteración: por cada <Book>...
     for (java.util.Iterator itBook = lib.Book.iterator(); itBook.hasNext();) {
        BookType book = (BookType) itBook.next();
        // valores de salida del atributo ID y del (primer y único) elemento title
        System.out.println("ID: " + book.ID.getValue());
        System.out.println("Title: " + book.Title.first().getValue());

        // leer y comparar un valor de enumeración
        if (book.Format.getEnumerationValue() == BookFormatType.EPAPERBACK)
           System.out.println("This is a paperback book.");

        // por cada <Author>...
        for (java.util.Iterator itAuthor = book.Author.iterator(); itAuthor
              .hasNext();)
           System.out.println("Author: " + ((com.Doc.xs.stringType)
```

```
itAuthor.next()).getValue());

        // buscar el tipo derivado de este libro
         // usando el valor del atributo xsi:type y usando DOM
        org.w3c.dom.Node bookNode = book.getNode();
        if (bookNode.getAttributes().getNamedItem("xsi:type") != null) {
           // obtener el valor del atributo xsi:type
           String xsiTypeValue =
bookNode.getAttributes().getNamedItem("xsi:type").getNodeValue();

           // obtener el URI de espacio de nombres y prefijo de consulta de este espacio
de nombres
           String namespaceUri = bookNode.getNamespaceURI();
           String lookupPrefix = bookNode.lookupPrefix(namespaceUri);

           // si xsi:type coincide con el URI de espacio de nombres y con el tipo del
nodo book
           if (namespaceUri == "http://www.nanonull.com/LibrarySample"
                 && (  xsiTypeValue.equals(lookupPrefix + ":DictionaryType" )))    {
              // ...entonces se trata de un libro de tipo derivado (diccionario)

              DictionaryType dictionary = new DictionaryType(   book.getNode());
              // generar el valor de los elementos "FromLang" y "ToLang" elements
              System.out.println("From language: " +
dictionary.FromLang.first().getValue());
              System.out.println("To language: " + dictionary.ToLang.first().getValue());
           }
           else
           {
              // emitir un error
              throw new java.lang.Error("This book has an unknown type.");
           }
        }
     }
   }
```

6. Compile y ejecute el proyecto Java. Si el código se ejecuta correctamente, el código de programa leerá el archivo **Library1.xml** y su contenido aparecerá en la vista de consola *Console*.

Leer y escribir elementos y atributos

El acceso a los atributos y elementos se consigue con el método `getValue()` de la clase de elemento o atributo miembro que se genera. Por ejemplo:

```
// valores de salida del atributo ID y del (primer y único) elemento title
System.out.println("ID: " + book.ID.getValue());
System.out.println("Title: " + book.Title.first().getValue());
```

Para obtener el valor del elemento **Title** en este ejemplo concreto también se utilizó el método `first()`. Esto se debe a que este es el primer (y único) elemento **Title** de un libro. Cuando necesite seleccionar un elemento concreto de una lista por medio del índice, utilice el método `at()`.

 © 2016-2022 Altova GmbH

Para recorrer varios elementos utilice la iteración basada en índice o `java.util.Iterator`. Por ejemplo, puede recorrer los libros de la biblioteca de la siguiente manera:

```
// iteración basada en índice
for (int j = 0; j < lib.Book.count(); ++j ) {
   // your code here
}

// iteración alternativa usando java.util.Iterator
for (java.util.Iterator itBook = lib.Book.iterator(); itBook.hasNext();) {
      // introduzca aquí su código
   }
```

Para agregar un elemento nuevo utilice el método `append()`. Por ejemplo, este código anexa un elemento raíz **Library** vacío al documento:

```
// crear el elemento raíz <Library> y agregarlo al documento
LibraryType lib = libDoc.Library
```

Tras anexar el elemento, puede establecer el valor de cualquiera de sus elementos o atributos con ayuda del método `setValue()`.

```
// establecer el valor del elemento Title
book.Title.append().setValue("The XML Spy Handbook");
// establecer el valor del atributo ID
book.ID.setValue(java.math.BigInteger.valueOf(1));
```

Leer y escribir valores de enumeración

Si el esquema XML define tipos simples como enumeraciones, los valores enumerados estarán disponibles como valores `Enum` en el código generado. En el esquema utilizado en este ejemplo el formato de un libro puede ser tapa dura, bolsillo, libro electrónico y audiolibro. Por tanto, en el código generado estos valores estarán disponibles a través de una `Enum` (véase el diagrama de la clase `BookFormatType` que aparece más arriba). Para asignar valores de enumeración a un objeto utilice código parecido a este:

```
// establecer un valor de enumeración
book.Format.setEnumerationValue( BookFormatType.EPAPERBACK
```

Dichos valores de enumeración se pueden leer desde documentos XML de instancia de la siguiente manera:

```
// leer un valor de enumeración
if (book.Format.getEnumerationValue() == BookFormatType.EPAPERBACK)
         System.out.println("This is a paperback book."
```

Cuando la condición IF no sea suficiente, cree un modificador para determinar cada valor de enumeración y procesarlo según corresponda en cada caso.

Trabajar con tipos xs:dateTime y xs:duration

Si el esquema desde el que se genera código utiliza tipos de hora y duración como `xs:dateTime` o `xs:duration`, estos tipos se convierten en clases nativas de Altova en el código generado. Por tanto, para escribir un valor de fecha o duración en el documento XML debe seguir estas instrucciones:

1. Construya un objeto `com.altova.types.DateTime` 1189 o `com.altova.types.Duration` 1194.
2. Establezca el objeto como valor del elemento o atributo que necesita. Por ejemplo:

```
// establecer el valor de un atributo de tipo DateTime
com.altova.types.DateTime dt = new com.altova.types.DateTime(DateTime.now());
lib.LastUpdated.setValue(dt);
```

Para leer una fecha o duración de un documento XML:

1. Declare el valor de elemento (o atributo) como objeto `com.altova.types.DateTime` 1189 o `com.altova.types.Duration` 1194.
2. Aplique formato al elemento o atributo que necesita. Por ejemplo:

```
// leer un tipo DateTime
com.altova.types.DateTime dt = lib.LastUpdated.getValue();
   System.out.println("The library was last updated on: " + dt.toDateString());
```

Para más información consulte la referencia de las clases `com.altova.types.DateTime` 1189 y `com.altova.types.Duration` 1194.

Trabajar con tipos derivados

Si su esquema XML define tipos derivados puede conservar la derivación de tipos en los documentos XML que cree o cargue mediante programación. Tomando el esquema utilizado en este ejemplo, el fragmento de código que aparece a continuación explica cómo se crea un libro nuevo de tipo derivado `DictionaryType`:

```
// crear un diccionario (libro de tipo derivado) y rellenar sus elementos y atributos
DictionaryType dict = new DictionaryType(lib.Book.append().getNode());
dict.ID.setValue(java.math.BigInteger.valueOf(2));
dict.Title.append().setValue("English-German Dictionary");
dict.Format.setEnumerationValue(BookFormatType.EE_BOOK);
dict.Author.append().setValue("John Doe");
dict.FromLang.append().setValue("English");
dict.ToLang.append().setValue("German");
dict.setXsiType();
```

Recuerde que es importante establecer el atributo `xsi:type` del libro recién creado. Esto garantiza que el esquema interprete correctamente el tipo de libro a la hora de validar el documento XML.

El fragmento de código que aparece a continuación identifica un libro de tipo derivado `DictionaryType` en la instancia XML que se carga. Primero se busca el valor del atributo `xsi:type` del nodo book. Si el URI de

 © 2016-2022 Altova GmbH

espacio de nombres de este nodo es `http://www.nanonull.com/LibrarySample` y el prefijo de consulta URI y el tipo coinciden con el valor del atributo `xsi:type`, entonces sabemos que se trata de un diccionario.

```
          // buscar el tipo derivado de este libro
         // usando el valor del atributo xsi:type y usando DOM
         org.w3c.dom.Node bookNode = book.getNode();
         if (bookNode.getAttributes().getNamedItem("xsi:type") != null) {
            // obtener el valor del atributo xsi:type
            String xsiTypeValue =
bookNode.getAttributes().getNamedItem("xsi:type").getNodeValue();

            // obtener el URI de espacio de nombres y el prefijo de consulta del nodo
book
            String namespaceUri = bookNode.getNamespaceURI();
            String lookupPrefix = bookNode.lookupPrefix(namespaceUri);

            // si xsi:type coincide con el URI de espacio de nombres y el tipo del nodo
book
            if (namespaceUri == "http://www.nanonull.com/LibrarySample"
                  && (  xsiTypeValue.equals(lookupPrefix + ":DictionaryType" )))    {
               // ...entonces se trata de un libro de tipo derivado (diccionario)

               DictionaryType dictionary = new DictionaryType(   book.getNode());
               // generar el valor de los elementos "FromLang" y "ToLang" elements
               System.out.println("From language: " +
dictionary.FromLang.first().getValue());
               System.out.println("To language: " +
dictionary.ToLang.first().getValue());
            }
            else
            {
               // emitir un error
               throw new java.lang.Error("This book has an unknown type.");
            }
         }
```

15.6.6 Ejemplo: orden de compra

Este ejemplo explica cómo trabajar con código de programa generado a partir de un esquema XML "principal" que importa otros esquemas. Cada uno de los esquemas importados tiene un espacio de nombres de destino distinto. El objetivo es crear de forma programática un documento XML en el que todos los elementos estén prefijados de acuerdo a su espacio de nombres. Más concretamente, el documento XML creado a partir de su código C++, C# o Java debería tener un aspecto parecido a este:

```
<?xml version="1.0" encoding="utf-8"?>
<p:Purchase xsi:schemaLocation="http://NamespaceTest.com/Purchase Main.xsd"
            xmlns:p="http://NamespaceTest.com/Purchase"
            xmlns:o="http://NamespaceTest.com/OrderTypes"
            xmlns:c="http://NamespaceTest.com/CustomerTypes"
            xmlns:cmn="http://NamespaceTest.com/CommonTypes"
            xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
```

```
    <p:OrderDetail>
        <o:Item>
            <o:ProductName>Lawnmower</o:ProductName>
            <o:Quantity>1</o:Quantity>
            <o:UnitPrice>148.42</o:UnitPrice>
        </o:Item>
    </p:OrderDetail>
    <p:PaymentMethod>VISA</p:PaymentMethod>
    <p:CustomerDetails>
        <c:Name>Alice Smith</c:Name>
        <c:DeliveryAddress>
            <cmn:Line1>123 Maple Street</cmn:Line1>
            <cmn:Line2>Mill Valley</cmn:Line2>
        </c:DeliveryAddress>
        <c:BillingAddress>
            <cmn:Line1>8 Oak Avenue</cmn:Line1>
            <cmn:Line2>Old Town</cmn:Line2>
        </c:BillingAddress>
    </p:CustomerDetails>
</p:Purchase>
```

El esquema principal que usamos en este ejemplo se llama **Main.xsd**. Como se ve en el extracto de código siguiente, el esquema importa otros tres esquemas: **CommonTypes.xsd**, **CustomerTypes.xsd** y **OrderTypes.xsd**. Para obtener los mismos resultados que en este ejemplo, guarde los extractos de código siguientes en archivos y guárdelos con los nombres de archivo que acabamos de mencionar. Tenga en cuenta que el esquema asigna cada uno de los prefijos **ord**, **pur**, **cmn** y **cust** a algún espacio de nombres (Order types, Purchase types, Common types y Customer types, respectivamente). Esto significa que, en el código generado, las clases que corresponden a Orders, Purchases, Customers, etc. están disponibles bajo el espacio de nombres correspondiente.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
      targetNamespace="http://NamespaceTest.com/Purchase"
      xmlns:ord="http://NamespaceTest.com/OrderTypes"
      xmlns:pur="http://NamespaceTest.com/Purchase"
      xmlns:cmn="http://NamespaceTest.com/CommonTypes"
      xmlns:cust="http://NamespaceTest.com/CustomerTypes"
      elementFormDefault="qualified">
    <xs:import schemaLocation="CommonTypes.xsd"
namespace="http://NamespaceTest.com/CommonTypes" />
    <xs:import schemaLocation="CustomerTypes.xsd"
namespace="http://NamespaceTest.com/CustomerTypes" />
    <xs:import schemaLocation="OrderTypes.xsd"
namespace="http://NamespaceTest.com/OrderTypes" />
    <xs:element name="Purchase">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="OrderDetail" type="ord:OrderType" />
                <xs:element name="PaymentMethod" type="cmn:PaymentMethodType" />
                <xs:element ref="pur:CustomerDetails" />
            </xs:sequence>
```

 © 2016-2022 Altova GmbH

```
        </xs:complexType>
    </xs:element>
    <xs:element name="CustomerDetails" type="cust:CustomerType" />
</xs:schema>
```

Main.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
      targetNamespace="http://NamespaceTest.com/CommonTypes"
      elementFormDefault="qualified">
   <xs:complexType name="AddressType">
      <xs:sequence>
         <xs:element name="Line1" type="xs:string"/>
         <xs:element name="Line2" type="xs:string"/>
      </xs:sequence>
   </xs:complexType>
   <xs:simpleType name="PriceType">
      <xs:restriction base="xs:decimal">
         <xs:fractionDigits value="2"/>
      </xs:restriction>
   </xs:simpleType>
   <xs:simpleType name="PaymentMethodType">
      <xs:restriction base="xs:string">
         <xs:enumeration value="VISA"/>
         <xs:enumeration value="MasterCard"/>
         <xs:enumeration value="Cash"/>
         <xs:enumeration value="AMEX"/>
      </xs:restriction>
   </xs:simpleType>
</xs:schema>
```

CommonTypes.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
      targetNamespace="http://NamespaceTest.com/CustomerTypes"
      xmlns:cmn="http://NamespaceTest.com/CommonTypes"
      elementFormDefault="qualified">
    <xs:import schemaLocation="CommonTypes.xsd"
namespace="http://NamespaceTest.com/CommonTypes" />
    <xs:complexType name="CustomerType">
        <xs:sequence>
            <xs:element name="Name" type="xs:string" />
            <xs:element name="DeliveryAddress" type="cmn:AddressType" />
            <xs:element name="BillingAddress" type="cmn:AddressType" />
        </xs:sequence>
    </xs:complexType>
</xs:schema>
```

CustomerTypes.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
      targetNamespace="http://NamespaceTest.com/OrderTypes"
      xmlns:cmn="http://NamespaceTest.com/CommonTypes"
      elementFormDefault="qualified">
    <xs:import schemaLocation="CommonTypes.xsd"
namespace="http://NamespaceTest.com/CommonTypes" />
    <xs:complexType name="OrderType">
        <xs:sequence>
            <xs:element maxOccurs="unbounded" name="Item">
                <xs:complexType>
                    <xs:sequence>
                        <xs:element name="ProductName" type="xs:string" />
                        <xs:element name="Quantity" type="xs:int" />
                        <xs:element name="UnitPrice" type="cmn:PriceType" />
                    </xs:sequence>
                </xs:complexType>
            </xs:element>
        </xs:sequence>
    </xs:complexType>
</xs:schema>
```

OrderTypes.xsd

Para completar el ejemplo, siga estos pasos:

1. Guarde en disco todos los esquemas de los extractos de código anteriores sin cambiarles el nombre.
2. Genere el código contenedor de esquema a partir del esquema de más arriba **Main.xsd** siguiendo los pasos descritos en Generar código fuente a partir de esquemas XML o DTD[1110]. Si ha seguido todos los pasos hasta ahora, debería haber generado un programa compilable en el lenguaje de su elección (C++, C# o Java).
3. Añada código a su programa C++, C# o Java desde el extracto que corresponda en cada caso:

   - Espacios de nombres y prefijos XML (C++)[1150]
   - Espacios de nombres y prefijos XML (C#)[1152]
   - Espacios de nombres y prefijos XML (Java)[1153]

15.6.6.1 Espacios de nombres y prefijos XML (C++)

Una vez haya generado código a partir del esquema de ejemplo[1147] se crea una aplicación C++ de prueba, junto con varias bibliotecas de Altova de apoyo. Recuerde que el esquema de ejemplo (Main.xsd) tiene varias declaraciones de espacios de nombres. En consecuencia, el código generado incluye espacios de nombres que corresponden a alias de espacios de nombres (prefijos) del esquema, que son: **`Main::ord`**, **`Main::pur`**, **`Main::cmn`** y **`Main::cust`**.

Por lo general, para controlar espacios de nombres y prefijos XML con ayuda de las bibliotecas contenedoras de esquemas, debe contar con estos métodos:

- DeclareAllNamespacesFromSchema()[1166]. Llame a este método si quiere declarar los mismos espacios de nombres en su instancia XML que en el esquema. En cambio, si necesita distintos espacios de nombres, como en este ejemplo, entonces use **`DeclareNamespace()`**. El método

DeclareAllNamespacesFromSchema() no se usa en este ejemplo porque lo que queremos es crear elementos XML con prefijos que sean ligeramente distintos a los que se declaran en el esquema.

- DeclareNamespace() [1168]. Llame a este método para crear o sobrescribir el atributo del prefijo del espacio de nombres de un elemento. Para ello primero debe crear ese elemento, bien con el método **append()** o con **appendWithPrefix()**, como explicamos a continuación.
- appendWithPrefix(). [1170] Use este método para anexar un elemento de instancia con un prefijo concreto. Para crear la instancia XML de este ejemplo bastó con llamar a este método solamente para el elemento raíz. Todos los demás elementos se anexaron usando simplemente append() [1170] y sus prefijos se añadieron automáticamente basándose en los espacios de nombres a los que pertenece el nombre del elemento.

El extracto de código siguiente muestra cómo crear un documento XML con varias declaraciones de espacios de nombres y nombres de elementos prefijados. Más concretamente, genera una instancia de orden de compra, como se puede ver en el Ejemplo: orden de compra [1147]. Es importante tener en cuenta que por motivos ilustrativos en el ejemplo algunos prefijos se han sobrescrito en la instancia XML (es decir, no son exactamente los mismos que los que se declaran en el esquema).

```
void Example()
{
   // Create the XML document and append the root element
   Main::pur::CMain doc = Main::pur::CMain::CreateDocument();
   Main::pur::CPurchaseType purchase = doc.Purchase.appendWithPrefix(_T("p"));

   // Set schema location
   doc.SetSchemaLocation(_T("Main.xsd"));

   // Declare namespaces on root element
   purchase.DeclareNamespace(_T("o"), _T("http://NamespaceTest.com/OrderTypes"));
   purchase.DeclareNamespace(_T("c"), _T("http://NamespaceTest.com/CustomerTypes"));
   purchase.DeclareNamespace(_T("cmn"), _T("http://NamespaceTest.com/CommonTypes"));

   // Append the OrderDetail element
   Main::ord::COrderType order = purchase.OrderDetail.append();
   Main::ord::CItemType item = order.Item.append();
   item.ProductName.append() = _T("Lawnmower");
   item.Quantity.append() = 1;
   item.UnitPrice.append() = 148.42;

   // Append the PaymentMethod element
   Main::cmn::CPaymentMethodTypeType paymentMethod = purchase.PaymentMethod.append();
   paymentMethod.SetEnumerationValue(Main::cmn::CPaymentMethodTypeType::k_VISA);

   // Append the CustomerDetails element
   Main::cust::CCustomerType customer = purchase.CustomerDetails.append();
   customer.Name.append() = _T("Alice Smith");
   Main::cmn::CAddressType deliveryAddress = customer.DeliveryAddress.append();
   deliveryAddress.Line1.append() = _T("123 Maple Street");
   deliveryAddress.Line2.append() = _T("Mill Valley");
   Main::cmn::CAddressType billingAddress = customer.BillingAddress.append();
   billingAddress.Line1.append() = _T("8 Oak Avenue");
   billingAddress.Line2.append() = _T("Old Town");

   // Save to file and release object from memory
```

```
   doc.SaveToFile(_T("Main1.xml"), true);
   doc.DestroyDocument();
}
```

15.6.6.2 Espacios de nombres y prefijos XML (C#)

Una vez haya generado código a partir del esquema de ejemplo 1147 se crea una aplicación C# de prueba, junto con varias bibliotecas de Altova de apoyo. Recuerde que el esquema de ejemplo (Main.xsd) tiene varias declaraciones de espacios de nombres. En consecuencia, el código generado incluye espacios de nombres que corresponden a alias de espacios de nombres (prefijos) del esquema, que son: **`Main::ord`**, **`Main::pur`**, **`Main::cmn`** y **`Main::cust`**.

Por lo general, para controlar espacios de nombres y prefijos XML con ayuda de las bibliotecas contenedoras de esquemas, debe contar con estos métodos:

- DeclareAllNamespacesFromSchema() 1166: llame a este método si quiere declarar los mismos espacios de nombres en su instancia XML que en el esquema. En cambio, si necesita distintos espacios de nombres, como en este ejemplo, entonces use **`DeclareNamespace()`**. El método **`DeclareAllNamespacesFromSchema()`** no se usa en este ejemplo porque lo que queremos es crear elementos XML con prefijos que sean ligeramente distintos a los que se declaran en el esquema.
- DeclareNamespace() 1184: llame a este método para crear o sobrescribir el atributo del prefijo del espacio de nombres de un elemento. Para ello primero debe crear ese elemento, bien con el método **`Append()`** o con **`AppendWithPrefix()`**, como explicamos a continuación.
- AppendWithPrefix() 1186: use este método para anexar un elemento de instancia con un prefijo concreto. Para crear la instancia XML de este ejemplo bastó con llamar a este método solamente para el elemento raíz. Todos los demás elementos se anexaron usando simplemente Append() 1186 y sus prefijos se añadieron automáticamente basándose en los espacios de nombres a los que pertenece el nombre del elemento.

El extracto de código siguiente muestra cómo crear un documento XML con varias declaraciones de espacios de nombres y nombres de elementos prefijados. Más concretamente, genera una instancia de orden de compra, como se puede ver en el Ejemplo: orden de compra 1147. Es importante tener en cuenta que por motivos ilustrativos en el ejemplo algunos prefijos se han sobrescrito en la instancia XML (es decir, no son exactamente los mismos que los que se declaran en el esquema).

```
protected static void Example()
{
   // Create the XML document and append the root element
   pur.Main2 doc = pur.Main2.CreateDocument();
   pur.PurchaseType purchase = doc.Purchase.AppendWithPrefix("p");

   // Set schema location
   doc.SetSchemaLocation(@"Main.xsd");

   // Declare namespaces on root element
   purchase.DeclareNamespace("o", "http://NamespaceTest.com/OrderTypes");
   purchase.DeclareNamespace("c", "http://NamespaceTest.com/CustomerTypes");
   purchase.DeclareNamespace("cmn", "http://NamespaceTest.com/CommonTypes");

   // Append the OrderDetail element
```

 © 2016-2022 Altova GmbH

```
    ord.OrderType order = purchase.OrderDetail.Append();
    ord.ItemType item = order.Item.Append();
    item.ProductName.Append().Value = "Lawnmower";
    item.Quantity.Append().Value = 1;
    item.UnitPrice.Append().Value = 148.42M;

    // Append the PaymentMethod element
    cmn.PaymentMethodTypeType paymentMethod = purchase.PaymentMethod.Append();
    paymentMethod.EnumerationValue = cmn.PaymentMethodTypeType.EnumValues.eVISA;

    // Append the CustomerDetails element
    cust.CustomerType customer = purchase.CustomerDetails.Append();
    customer.Name.Append().Value = "Alice Smith";
    cmn.AddressType deliveryAddress = customer.DeliveryAddress.Append();
    deliveryAddress.Line1.Append().Value = "123 Maple Street";
    deliveryAddress.Line2.Append().Value = "Mill Valley";
    cmn.AddressType billingAddress = customer.BillingAddress.Append();
    billingAddress.Line1.Append().Value = "8 Oak Avenue";
    billingAddress.Line2.Append().Value = "Old Town";

    // Save to file
    doc.SaveToFile("PurchaseOrder.xml", true);
}
```

15.6.6.3 Espacios de nombres y prefijos XML (Java)

Una vez haya generado código a partir del esquema de ejemplo 1147 se crea una aplicación Java de prueba, junto con varias bibliotecas de Altova de apoyo. Recuerde que el esquema de ejemplo (Main.xsd) tiene varias declaraciones de espacios de nombres. En consecuencia, el código generado incluye espacios de nombres que corresponden a alias de espacios de nombres (prefijos) del esquema, que son: **com.Main.ord**, **com.Main.pur**, **com.Main.cmn** y **com.Main.cust**. .

Por lo general, para controlar espacios de nombres y prefijos XML con ayuda de las bibliotecas contenedoras de esquemas, debe contar con estos métodos:

- declareAllNamespacesFromSchema() 1200: llame a este método si quiere declarar los mismos espacios de nombres en su instancia XML que en el esquema. En cambio, si necesita distintos espacios de nombres, como en este ejemplo, entonces use **DeclareNamespace()**. El método **DeclareAllNamespacesFromSchema()** no se usa en este ejemplo porque lo que queremos es crear elementos XML con prefijos que sean ligeramente distintos a los que se declaran en el esquema.
- declareNamespace() 1202: llame a este método para crear o sobrescribir el atributo del prefijo del espacio de nombres de un elemento. Para ello primero debe crear ese elemento, bien con el método **append()** o con **appendWithPrefix()**, como explicamos a continuación.
- appendWithPrefix() 1204: Use este método para anexar un elemento de instancia con un prefijo concreto. Para crear la instancia XML de este ejemplo bastó con llamar a este método solamente para el elemento raíz. Todos los demás elementos se anexaron usando simplemente append() 1170 y sus prefijos se añadieron automáticamente basándose en los espacios de nombres a los que pertenece el nombre del elemento.

El extracto de código siguiente muestra cómo crear un documento XML con varias declaraciones de espacios de nombres y nombres de elementos prefijados. Más concretamente, genera una instancia de orden de

compra, como se puede ver en el Ejemplo: orden de compra[1147]. Es importante tener en cuenta que por motivos ilustrativos en el ejemplo algunos prefijos se han sobrescrito en la instancia XML (es decir, no son exactamente los mismos que los que se declaran en el esquema).

```
protected static void example() throws Exception {
   // Create the XML document and append the root element
   com.Main.pur.Main2 doc = com.Main.pur.Main2.createDocument();
   com.Main.pur.PurchaseType purchase = doc.Purchase.appendWithPrefix("p");

   // Set schema location
   doc.setSchemaLocation("Main.xsd");

   // Declare namespaces on root element
   purchase.declareNamespace("o", "http://NamespaceTest.com/OrderTypes");
   purchase.declareNamespace("c", "http://NamespaceTest.com/CustomerTypes");
   purchase.declareNamespace("cmn", "http://NamespaceTest.com/CommonTypes");

   // Append the OrderDetail element
   com.Main.ord.OrderType order = purchase.OrderDetail.append();
   com.Main.ord.ItemType item = order.Item.append();
   item.ProductName.append().setValue("Lawnmower");
   item.Quantity.append().setValue(1);
   java.math.BigDecimal price = new java.math.BigDecimal("148.42");
   item.UnitPrice.append().setValue(price);

   // Append the PaymentMethod element
   com.Main.cmn.PaymentMethodTypeType paymentMethod = purchase.PaymentMethod.append();
   paymentMethod.setEnumerationValue(com.Main.cmn.PaymentMethodTypeType.EVISA);

   // Append the CustomerDetails element
   com.Main.cust.CustomerType customer = purchase.CustomerDetails.append();
   customer.Name.append().setValue("Alice Smith");
   com.Main.cmn.AddressType deliveryAddress = customer.DeliveryAddress.append();
   deliveryAddress.Line1.append().setValue("123 Maple Street");
   deliveryAddress.Line2.append().setValue("Mill Valley");
   com.Main.cmn.AddressType billingAddress = customer.BillingAddress.append();
   billingAddress.Line1.append().setValue("8 Oak Avenue");
   billingAddress.Line2.append().setValue("Old Town");

   // Save to file
   doc.saveToFile("PurchaseOrder.xml", true);
}
```

 © 2016-2022 Altova GmbH

15.7 Referencia de las clases generadas (C++)

Esta sección describe las clases C++ que se generan con MapForce a partir de una DTD o esquema XML (*véase* *Generar código a partir de esquemas XML o DTD* 1110). Puede integrar estas clases en su código para leer, modificar y escribir documentos XML.

Nota: el código generado incluye otras clases compatibles que no se enumeran en este apartado y que pueden cambiar en el futuro.

15.7.1 altova::DateTime

Esta clase permite procesar atributos o elementos XML de tipo hora y fecha (p. ej. `xs:dateTime`).

Constructores

Nombre	Descripción
`DateTime()`	Inicializa una instancia nueva de la clase `DateTime` en la medianoche 12:00:00 del 1 de enero de 0001.
`DateTime(__int64 value, short timezone)`	Inicializa una instancia nueva de la clase `DateTime`. El parámetro `value` representa el número de ciclos (intervalos de 100 nanosegundos) que han pasado desde la medianoche 12:00:00 del 1 de enero de 0001.
`DateTime(int year, unsigned char month, unsigned char day, unsigned char hour, unsigned char minute, double second)`	Inicializa una instancia nueva de la clase `DateTime` en el año, mes, día, hora, minuto y segundo dados como argumento.
`DateTime(int year, unsigned char month, unsigned char day, unsigned char hour, unsigned char minute, double second, short timezone)`	Inicializa una instancia nueva de la clase `DateTime` en el año, mes, día, hora, minuto, segundo y zona horaria dados como argumento. La zona horaria se expresa en minutos y puede ser positiva o negativa. Por ejemplo, la zona horaria "UTC-01:00" se expresa como "-60".

Métodos

Nombre	Descripción
`unsigned char Day() const`	Devuelve el día del mes del objeto `DateTime` actual. Los valores devueltos están comprendidos entre 1 y 31.
`int DayOfYear() const`	Devuelve el día del año del objeto `DateTime` actual. Los valores devueltos están comprendidos entre 1 y 366.

Nombre	Descripción
`bool HasTimezone() const`	Devuelve el valor binario **true** si el objeto `DateTime` actual tiene definida una zona horaria. De lo contrario devuelve **false**.
`unsigned char Hour() const`	Devuelve la hora del objeto `DateTime` actual. Los valores devueltos están comprendidos entre 0 y 23.
`static bool IsLeapYear(int year)`	Devuelve el valor binario **true** si el año de la clase `DateTime` es un año bisiesto. De lo contrario devuelve **false**.
`unsigned char Minute() const`	Devuelve el minuto del objeto `DateTime` actual. Los valores devueltos están comprendidos entre 0 y 59.
`unsigned char Month() const`	Devuelve el mes del objeto `DateTime` actual. Los valores devueltos están comprendidos entre 1 y 12.
`__int64 NormalizedValue() const`	Devuelve el valor del objeto `DateTime` expresado como UTC.
`double Second() const`	Devuelve el segundo del objeto `DateTime` actual. Los valores devueltos están comprendidos entre 0 y 59.
`void SetTimezone(short tz)`	Establece la zona horaria del objeto `DateTime` actual en el valor de zona horaria dado como argumento. El argumento **tz** se expresa en minutos y puede ser positivo o negativo.
`short Timezone() const`	Devuelve la zona horaria en minutos del objeto `DateTime` actual. Antes de usar este método asegúrese de que el objeto realmente tiene una zona horaria (llamando al método `HasTimezone()`).
`__int64 Value() const`	Devuelve el valor del objeto `DateTime` expresado en el número de ciclos (intervalos de 100 nanosegundos) que han pasado desde la medianoche 12:00:00 del 1 de enero de 0001.
`int Weekday() const`	Devuelve el día de la semana del objeto `DateTime` actual como entero. Los valores están comprendidos entre 0 y 6, siendo 0 el lunes (ISO-8601).
`int Weeknumber() const`	Devuelve el número de la semana del año del objeto `DateTime` actual. Los valores devueltos son conformes a ISO-8601.
`int WeekOfMonth() const`	Devuelve el número de la semana del mes del objeto `DateTime` actual. Los valores devueltos son conformes a ISO-8601.
`int Year() const`	Devuelve el año del objeto `DateTime` actual.

Ejemplo

```
void Example()
{
```


© 2016-2022 Altova GmbH

```
       // inicializar una nueva instancia de DateTime en la medianoche 12:00:00 del 1 de
enero de 0001
       altova::DateTime dt1 = altova::DateTime();

       // inicializar una nueva instancia de DateTime usando el año, mes, día, hora,
minuto y segundo
       altova::DateTime dt2 = altova::DateTime(2015, 11, 10, 9, 8, 7);

       // inicializar una nueva instancia de DateTime usando el año, mes, día, hora,
minuto, segundo y la zona horaria UTC +01:00
        altova::DateTime dt = altova::DateTime(2015, 11, 22, 13, 53, 7, 60);

       // Obtener el valor de este objeto DateTime
       std::cout << "El número de ciclos del objeto DateTime es: " << dt.Value() <<
std::endl;

       // Obtener el año
       cout << "El año es: " << dt.Year() << endl;
       // Obtener el mes
       cout << "El mes es: " << (int)dt.Month() << endl;
       // Obtener el día del mes
       cout << "El día del mes es: " << (int) dt.Day() << endl;
       // Obtener el día del año
       cout << "El día del año es: " << dt.DayOfYear() << endl;
       // Obtener la hora
       cout << "La hora es: " << (int) dt.Hour() << endl;
       // Obtener el minuto
       cout << "El minuto es: " << (int) dt.Minute() << endl;
       // Obtener el segundo
       cout << "El segundo es: " << dt.Second() << endl;
       // Obtener el día de la semana
       cout << "El día de la semana es: " << dt.Weekday() << endl;
       // Obtener el número de la semana
       cout << "La semana del año es: " << dt.Weeknumber() << endl;
       // Obtener la semana del mes
       cout << "La semana del mes es: " << dt.WeekOfMonth() << endl;

       // Comprobar si una instancia de DateTime tiene una zona horaria
       if (dt.HasTimezone() == TRUE)
       {
              // generar el valor de la zona horaria
              cout <<  "La zona horaria es: " << dt.Timezone() << endl;
       }
       else
       {
              cout <<  "No se definió la zona horaria." << endl;
       }

       // Construir un objeto DateTime con una zona horaria UTC+01:00 (Viena)
       altova::DateTime vienna_dt = DateTime(2015, 11, 23, 14, 30, 59, +60);
       // Generar el resultado en formato legible
       cout << "La hora de Viena: "
               << (int) vienna_dt.Month()
               << "-" << (int) vienna_dt.Day()
               << " " << (int) vienna_dt.Hour()
```

```
               << ":" << (int) vienna_dt.Minute()
               << ":" << (int) vienna_dt.Second()
               << endl;

       // Convertir el valor en hora UTC
       DateTime utc_dt = DateTime(vienna_dt.NormalizedValue());
       // Generar el resultado en formato legible
       cout << "La hora UTC:    "
              << (int) utc_dt.Month()
              << "-" << (int) utc_dt.Day()
              << " " << (int) utc_dt.Hour()
              << ":" << (int) utc_dt.Minute()
              << ":" << (int) utc_dt.Second()
              << endl;

       // Comprobar si el año es bisiest
       int year = 2016;
       if( altova::DateTime::IsLeapYear(year) )
       { cout << year << " es año bisiesto" << endl; }
       else
       { cout << year << " no es año bisiesto" << endl; }
}
```


© 2016-2022 Altova GmbH

15.7.2 altova::Duration

Esta clase permite procesar atributos o elementos XML de tipo `xs:duration`.

Constructores

Nombre	Descripción
`Duration()`	Inicializa una instancia nueva de la clase `Duration` en un valor vacío.
`Duration(const DayTimeDuration& dt)`	Inicializa una instancia nueva de la clase `Duration` en una duración definida por el argumento `dt` (*véase altova::DayTimeDuration* 1161).
`Duration(const YearMonthDuration& ym)`	Inicializa una instancia nueva de la clase `Duration` en la duración definida por el argumento `ym` (véase *altova::YearMonthDuration* 1162).
`Duration(const YearMonthDuration& ym, const DayTimeDuration& dt)`	Inicializa una instancia nueva de la clase `Duration` en la duración definida por los argumentos `dt` y `ym` (*véase altova::YearMonthDuration* 1162 *y altova::DayTimeDuration* 1161).

Métodos

Nombre	Descripción
`int Days() const`	Devuelve el número de días de la instancia de `Duration` actual.
`DayTimeDuration DayTime() const`	Devuelve el día y la duración en horas de la instancia de `Duration` actual expresada como objeto `DayTimeDuration` (*véase altova::DayTimeDuration* 1161).
`int Hours() const`	Devuelve el número de horas de la instancia de `Duration` actual.
`bool IsNegative() const`	Devuelve el valor binario **true** si la instancia de `Duration` actual es negativa.
`bool IsPositive() const`	Devuelve el valor binario **true** si la instancia de `Duration` actual es positiva.
`int Minutes() const`	Devuelve el número de minutos de la instancia de `Duration` actual.
`int Months() const`	Devuelve el número de meses de la instancia de `Duration` actual.

Nombre	Descripción
`double Seconds() const`	Devuelve el número de segundos de la instancia de `Duration` actual.
`YearMonthDuration YearMonth() const`	Devuelve el año y la duración en meses de la instancia de `Duration` actual expresada como objeto `YearMonthDuration` (*véase* *altova::YearMonthDuration* 1162).
`int Years() const`	Devuelve el número de años de la instancia de `Duration` actual.

Ejemplo

En este ejemplo se crea un objeto `Duration` nuevo y se leen sus valores.

```
void ExampleDuration()
{
       // Crear un objeto Duration vacío
       altova::Duration empty_duration = altova::Duration();

       // Crear un objeto Duration usando un valor de duración actual
       altova::Duration duration1 = altova::Duration(empty_duration);

       // Crear una duración YearMonth de seis años y cinco meses
       altova::YearMonthDuration yrduration = altova::YearMonthDuration(6, 5);

       // Crear una duración DayTime de cuatro días, tres horas, dos minutos y un segundo
       altova::DayTimeDuration dtduration = altova::DayTimeDuration(4, 3, 2, 1);

       // Crear un objeto Duration combinando las dos duraciones creadas previamente
       altova::Duration duration = altova::Duration(yrduration, dtduration);

       // Obtener el número de años de esta instancia de Duration
       cout << "Años:  " << duration.Years() << endl;

       // Obtener el número de meses de esta instancia de Duration
       cout << "Meses: " << duration.Months() << endl;

       // Obtener el número de días de esta instancia de Duration
       cout << "Días:   " << duration.Days() << endl;

       // Obtener el número de horas de esta instancia de Duration
       cout << "Horas:  " << duration.Hours() << endl;

       // Obtener el número de minutos de esta instancia de Duration
       cout << "Minutos: " << duration.Minutes() << endl;

       // Obtener el número de segundos de esta instancia de Duration
       cout << "Segundos: " << duration.Seconds() << endl;
}
```

 © 2016-2022 Altova GmbH

15.7.3 altova::DayTimeDuration

Esta clase permite procesar tipos XML Schema de duración compuestos por un día y una hora.

Constructores

Nombre	Descripción
`DayTimeDuration()`	Inicializa una instancia nueva de la clase `DayTimeDuration` en un valor vacío.
`DayTimeDuration(int days, int hours, int minutes, double seconds)`	Inicializa una instancia nueva de la clase `DayTimeDuration` en el número de días, horas, minutos y segundos dados como argumento.
`explicit DayTimeDuration(__int64 value)`	Inicializa una instancia nueva de la clase `DayTimeDuration` en una duración compuesta por tantos ciclos (intervalos de 100 nanosegundos) como aporta el argumento **value**.

Métodos

Nombre	Descripción
`int Days() const`	Devuelve el número de días de la instancia de `DayTimeDuration` actual.
`int Hours() const`	Devuelve el número de horas de la instancia de `DayTimeDuration` actual.
`bool IsNegative() const`	Devuelve el valor binario **true** si la instancia de `DayTimeDuration` actual es negativa.
`bool IsPositive() const`	Devuelve el valor binario **true** si la instancia de `DayTimeDuration` actual es positiva.
`int Minutes() const`	Devuelve el número de minutos de la instancia de `DayTimeDuration` actual.
`double Seconds() const`	Devuelve el número de segundos de la instancia de `DayTimeDuration` actual.
`__int64 Value() const`	Devuelve el valor (en ciclos) de la instancia de `DayTimeDuration` actual.

15.7.4 altova::YearMonthDuration

Esta clase permite procesar tipos XML Schema de duración compuestos por un año y un mes.

Constructores

Nombre	Descripción
`YearMonthDuration()`	Inicializa una instancia nueva de la clase `YearMonthDuration` en un valor vacío.
`YearMonthDuration(int years, int months)`	Inicializa una instancia nueva de la clase `YearMonthDuration` en el número de años y meses dados por los argumentos **years** y **months**.
`explicit YearMonthDuration(int value)`	Inicializa una instancia nueva de la clase `YearMonthDuration` en una duración compuesta por tantos ciclos (intervalos de 100 nanosegundos) como da el argumento **value**.

Métodos

Nombre	Descripción
`bool IsNegative() const`	Devuelve el valor binario **true** si la instancia de `YearMonthDuration` actual es negativa.
`bool IsPositive() const`	Devuelve el valor binario **true** si la instancia de `YearMonthDuration` actual es positiva.
`int Months() const`	Devuelve el número de meses de la instancia de `YearMonthDuration` actual.
`int Value() const`	Devuelve el valor (en ciclos) de la instancia de `YearMonthDuration` actual.
`int Years()`	Devuelve el número de años de la instancia de `YearMonthDuration` actual.

15.7.5 altova::meta::Attribute

Esta clase permite acceder a la información del esquema sobre clases generadas a partir de atributos. Recuerde que esta clase no tiene como objetivo aportar información dinámica sobre las distintas instancias de un atributo de un documento XML. Por el contrario, permite obtener información mediante programación sobre cualquier atributo que esté definido en el esquema XML.

 © 2016-2022 Altova GmbH

Métodos

Nombre	Descripción
`SimpleType GetDataType()`	Devuelve el tipo del contenido del atributo.
`string_type GetLocalName()`	Devuelve el nombre local del atributo.
`string_type GetNamespaceURI()`	Devuelve el URI de espacio de nombres del atributo.
`bool IsRequired()`	Devuelve true si el atributo es obligatorio.

Operadores

Nombre	Descripción
`bool operator()`	Devuelve true si este no es el atributo NULL.
`bool operator!()`	Devuelve true si este es el atributo NULL.

15.7.6 altova::meta::ComplexType

Esta clase permite acceder a información del esquema sobre clases generadas a partir de tipos complejos. Recuerde que esta clase no tiene como objetivo aportar información dinámica sobre las distintas instancias de un tipo complejo de un documento XML. Por el contrario, permite obtener información mediante programación sobre cualquier tipo complejo que esté definido en el esquema XML.

Métodos

Nombre	Descripción
`Attribute FindAttribute(const char_type* localName, const char_type* namespaceURI)`	Busca el atributo que tiene el nombre local y el URI de espacio de nombres especificados.
`Element FindElement(const char_type* localName, const char_type* namespaceURI)`	Busca el elemento que tiene el nombre local y el URI de espacio de nombres especificados.
`std::vector<Attribute> GetAttributes()`	Devuelve una lista de todos los atributos.
`ComplexType GetBaseType()`	Devuelve el tipo base de este tipo.
`SimpleType GetContentType()`	Devuelve el tipo simple del contenido.

Nombre	Descripción
`std::vector<Element> GetElements()`	Devuelve una lista de todos los elementos.
`string_type GetLocalName()`	Devuelve el nombre local del tipo.
`string_type GetNamespaceURI()`	Devuelve el URI de espacio de nombres del tipo.

Operadores

Nombre	Descripción
`bool operator()`	Devuelve true si este no es el tipo complejo NULL.
`bool operator!()`	Devuelve true si este es el tipo complejo NULL.

15.7.7 altova::meta::Element

Esta clase permite acceder a la información del esquema sobre clases generadas a partir de elementos. Recuerde que esta clase no tiene como objetivo aportar información dinámica sobre las distintas instancias de un elemento de un documento XML. Por el contrario, permite obtener información mediante programación sobre cualquier elemento que esté definido en el esquema XML.

Métodos

Nombre	Descripción
`ComplexType GetDataType()`	Devuelve el tipo del elemento. Recuerde que este siempre es un tipo complejo incluso si se declaró como tipo simple en el esquema original. Use GetContentType() del objeto devuelto para obtener el tipo de contenido simple.
`string_type GetLocalName()`	Devuelve el nombre local del elemento.
`unsigned int GetMaxOccurs()`	Devuelve el valor maxOccurs que está definido en el esquema.
`unsigned int GetMinOccurs()`	Devuelve el valor minOccurs que está definido en el esquema.
`string_type GetNamespaceURI()`	Devuelve el URI de espacio de nombres del elemento.

© 2016-2022 Altova GmbH

Operadores

Nombre	Descripción
`bool operator()`	Devuelve false si este no es el elemento NULL.
`bool operator!()`	Devuelve true si este es el elemento NULL.

15.7.8 altova::meta::SimpleType

Esta clase permite acceder a la información del esquema sobre clases generadas a partir de tipos simples. Recuerde que esta clase no tiene como objetivo aportar información dinámica sobre las distintas instancias de un tipo simple de un documento XML. Por el contrario, permite obtener información mediante programación sobre cualquier tipo simple que esté definido en el esquema XML.

Métodos

Nombre	Descripción
`SimpleType GetBaseType()`	Devuelve el tipo base de este tipo.
`std::vector<string_type> GetEnumerations()`	Devuelve una lista de todas las facetas de enumeración.
`unsigned int GetFractionDigits()`	Devuelve el valor de esta faceta.
`unsigned int GetLength()`	Devuelve el valor de esta faceta.
`string_type GetLocalName()`	Devuelve el nombre local de este tipo.
`string_type GetMaxExclusive()`	Devuelve el valor de esta faceta.
`string_type GetMaxInclusive()`	Devuelve el valor de esta faceta.
`unsigned int GetMaxLength()`	Devuelve el valor de esta faceta.
`string_type GetMinExclusive()`	Devuelve el valor de esta faceta.
`string_type GetMinInclusive()`	Devuelve el valor de esta faceta.
`unsigned int GetMinLength()`	Devuelve el valor de esta faceta.
`string_type GetNamespaceURI()`	Devuelve el URI de espacio de nombres del tipo.

Nombre	Descripción
`std::vector<string_type> GetPatterns()`	Devuelve una lista de todas las facetas de patrón.
`unsigned int GetTotalDigits()`	Devuelve el valor de esta faceta.
`WhitespaceType GetWhitespace()`	Devuelve el valor de la faceta de espacio en blanco, que es uno de estos: • Whitespace_Unknown • Whitespace_Preserve • Whitespace_Replace • Whitespace_Collapse

Operadores

Nombre	Descripción
`bool operator()`	Devuelve true si este no es el SimpleType NULL.
`bool operator!()`	Devuelve true si este es el SimpleType NULL.

15.7.9 [SuEsquema]::[CDoc]

Cuando se genera código a partir de un esquema XML, el código generado ofrece una clase de documento que se llama igual que el esquema. Esta clase tiene como miembros todos los elementos raíz posibles, así como los métodos que aparecen a continuación. Observe que en los nombres de método, `CDoc` corresponde al nombre de la clase de documento generada.

Métodos

Nombre	Descripción
`static CDoc CreateDocument()`	Crea un documento XML nuevo vacío. Debe publicarse con DestroyDocument().
`static void DeclareAllNamespacesFromSchema(TypeBase& node)`	Declara todos los espacios de nombres del esquema XML en el elemento dado como argumento (que suele ser el elemento XML raíz). Es útil llamar a este método si su esquema tiene varias declaraciones de espacios de nombres, cada una de ellas está asignada a un prefijo y quiere declararlas todas en el elemento dado como argumento.
`void DestroyDocument()`	Destruye un documento. Todas las referencias al documento y a sus nodos se invalidarán. Esta llamada debe hacerse cuando se termina de trabajar con un documento.

 © 2016-2022 Altova GmbH

Nombre	Descripción
`static CDoc LoadFromBinary(const std:vector<unsigned char>& xml)`	Carga un documento XML desde una matriz de bytes.
`static CDoc LoadFromFile(const string_type& fileName)`	Carga un documento XML desde un archivo.
`static CDoc LoadFromString(const string_type& xml)`	Carga un documento XML desde una cadena de texto.
`std::vector<unsigned char> SaveToBinary(bool prettyPrint)`	Guarda un documento XML en una matriz de bytes. Si su valor es true, el argumento `prettyPrint` ajusta el formato del documento XML para facilitar su lectura.
`std::vector<unsigned char> SaveToBinary(bool prettyPrint, const string_type & encoding)`	Guarda un documento XML en una matriz de bytes (con formato pretty-print opcional) con la codificación especificada.
`std::vector<unsigned char> SaveToBinary(bool prettyPrint, const string_type & encoding, bool bBigEndian, bool bBOM)`	Guarda un documento XML en una matriz de bytes (con formato pretty-print opcional) con la codificación especificada. Para codificaciones Unicode se puede especificar el orden de bytes y la marca BOM Unicode.
`void SaveToFile(const string_type & fileName, bool prettyPrint)`	Guarda un documento XML en un archivo, con formato pretty-print opcional.
`void SaveToFile(const string_type & fileName, bool omitXmlDecl)`	Guarda un documento XML en un archivo. Si el argumento `omitXmlDecl` tiene el valor true, la declaración XML no se escribirá.
`void SaveToFile(const string_type & fileName, bool omitXmlDecl, const string_type & encoding)`	Guarda un documento XML en un archivo. Si el argumento `omitXmlDecl` tiene el valor true, la declaración XML no se escribirá.
`void SaveToFile(const string_type & fileName, bool prettyPrint, bool omitXmlDecl, const string_type & encoding, bool bBigEndian, bool bBOM)`	Guarda un documento XML en un archivo (con formato pretty-print opcional) con la codificación especificada. Para codificaciones Unicode se puede especificar el orden de bytes y la marca BOM Unicode.
`void SaveToFile(const string_type & fileName, bool prettyPrint, bool omitXmlDecl, const string_type & encoding, bool bBigEndian, bool bBOM, const string_type & lineend)`	Guarda un documento XML en un archivo(con formato pretty-print opcional) con la codificación y el final de línea especificados. Para codificaciones Unicode se puede especificar el orden de bytes y la marca BOM Unicode. Este método solo está disponible si generó código para la biblioteca XML Xerces3 (*véase* *Opciones del generador de código* 1207).
`void SaveToFile(const string_type& fileName, bool prettyPrint, bool omitXmlDecl, const string_type & encoding, const string_type & lineend)`	Guarda un documento XML en un archivo, (con formato pretty-print opcional) con la codificación y el final de línea especificados.

Nombre	Descripción
	Este método solo está disponible si generó código para la biblioteca XML Xerces3 (*véase* *Opciones del generador de código*[1207]).
`void SaveToFile(const string_type & fileName, bool prettyPrint, const string_type & encoding)`	Guarda un documento XML en un archivo, (con formato pretty-print opcional) con la codificación especificada.
`void SaveToFile(const string_type& fileName, bool prettyPrint, const string_type & encoding, bool bBigEndian, bool bBOM)`	Guarda un documento XML en un archivo, (con formato pretty-print opcional) con la codificación especificada. Para codificaciones Unicode se puede especificar el orden de bytes y la marca BOM Unicode.
`void SaveToFile(const string_type& fileName, bool prettyPrint, const string_type & encoding, bool bBigEndian, bool bBOM, const string_type & lineend)`	Guarda un documento XML en un archivo con la codificación y el final de línea especificados. Para codificaciones Unicode se puede especificar el orden de bytes y la marca BOM Unicode. Este método solo está disponible si generó código para la biblioteca XML Xerces3 (*véase* *Opciones del generador de código*[1207]).
`void SaveToFile(const string_type& fileName, bool prettyPrint, const string_type & encoding, const string_type & lineend)`	Guarda un documento XML en un archivo, (con formato pretty-print opcional) con la codificación y el final de línea especificados. Este método solo está disponible si generó código para la biblioteca XML Xerces3 (véase Opciones del generador de código[1207]).
`string_type SaveToString(bool prettyPrint)`	Guarda un documento XML en una cadena de texto, con formato pretty-print opcional.
`string_type SaveToString(bool prettyPrint, bool omitXmlDecl)`	Guarda un documento XML en una cadena de texto, con formato pretty-print opcional. Si el argumento `omitXmlDecl` tiene el valor true, la declaración XML no se escribirá.
`void SetDTDLocation(const string_type & dtdLocation)`	Agrega una declaración DOCTYPE con el ID de sistema especificado. Es obligatorio que ya exista un elemento raíz. Este método es incompatible con MSXML porque no se puede agregar una declaración DOCTYPE a un documento que está en memoria.
`void SetSchemaLocation(const string_type & schemaLocation)`	Agrega un atributo `xsi:schemaLocation` o `xsi:noNamespaceSchemaLocation` al elemento raíz. Es obligatorio que ya exista un elemento raíz.

15.7.10 [SuEsquema]::[ElementType]

Esta clase permite manipular elementos XML desde el esquema. Los métodos de esta clase se pueden llamar en elementos, pero no en el documento XML. Observe que no necesita instanciar la clase directamente para

 © 2016-2022 Altova GmbH

poder llamar a los métodos de esta clase. Cualquier elemento creado con los métodos **append()** o **appendWithPrefix()** es de tipo **append()** or **appendWithPrefix()**.

Métodos

Nombre	Descripción

`void DeclareNamespace(const string_type prefix, const string_type nsURI)`	Observe que para llamar al método no necesita instanciar la clase TypeBase directamente. Cualquier elemento que cree usando los métodos **append()** o **appendWithPrefix()** hereda del tipo TypeBase.
	
	Este método toma dos argumentos de tipo cadena: el prefijo y el URI de espacio de nombres que quiere usar. El prefijo dado como argumento se asigna al valor del URI de espacio de nombres dado como argumento. Si el prefijo dado como argumento está vacío, el método crea o sobrescribe la declaración de espacio de nombres predeterminada del elemento.
	
	Por ejemplo, imagine que el documento XML tiene un elemento XML llamado "purchase". Si llama a
	
	`purchase.DeclareNamespace(_T("ord"), _T("http://OrderTypes"));`
	
	entonces el documento XML se convierte en
	
	`<purchase xmlns:ord="http://OrderTypes" />`
	
	Otro ejemplo. Si llama a:
	
	`purchase.DeclareNamespace(_T(""), _T("http://OrderTypes"));`
	
	entonces el documento XML se convierte en
	
	`<purchase xmlns="http://OrderTypes" />`
	
	Nota: el espacio de nombres declarado se usa al anexar elementos secundarios o atributos posteriormente, conforme a estas normas:
	
	1. Si el espacio de nombres secundario es el predeterminado, use un prefijo vacío.
	2. Si el espacio de nombres secundario es igual que el del elemento matriz, use el prefijo matriz.
	3. De lo contrario, busque el prefijo más cercano desde el

Nombre	Descripción
	elemento matriz hasta el principio usando el algoritmo de búsqueda que se describe en la sección "B.2: Namespace Prefix Lookup" de https://www.w3.org/TR/2002/WD-DOM-Level-3-Core-20021022/namespaces-algorithms.html. 4. Si no se encuentra ningún prefijo para el espacio de nombres del elemento, entonces use un prefijo vacío.

15.7.11 [SuEsquema]::MemberAttribute

Cuando se genera código a partir de un esquema XML, se crea una clase como esta por cada atributo miembro de un tipo.

Métodos

Nombre	Descripción
`bool exists()`	Devuelve true si el atributo existe.
`int GetEnumerationValue()`	Solo se genera para tipos de enumeración. Devuelve una de las constantes generadas para los posibles valores o "no válido" si el valor no coincide con ninguno de los valores enumerados en el esquema.
`altova::meta::Attribute info()`	Devuelve un objeto para consultar información del esquema (*véase* `altova::meta::Attribute` 1162).
`void remove()`	Quita el atributo de su elemento primario.
`void SetEnumerationValue(int)`	Solo se genera para tipos de enumeración. Pase una de las constantes generadas para los posibles valores a este método para establecer el valor.

15.7.12 [SuEsquema]::MemberElement

Cuando se genera código a partir de un esquema XML, se crea una clase como esta por cada elemento miembro de un tipo. En la tabla que aparece a continuación `MemberType` corresponde al nombre del elemento miembro propiamente dicho.

© 2016-2022 Altova GmbH

Métodos

Nombre	Descripción
`Iterator<MemberType> all()`	Devuelve un objeto para recorrer instancias del elemento miembro.
`MemberType append()`	Crea un elemento nuevo y lo anexa a su primario.
`MemberType appendWithPrefix(string_type prefix)`	Crea un elemento nuevo que tiene el prefijo dado como argumento y lo anexa a su elemento superior. Para ver un ejemplo consulte Ejemplo: orden de compra 1147.
`unsigned int count()`	Devuelve el recuento de elementos.
`int GetEnumerationValue()`	Solo se genera para tipos de enumeración. Devuelve una de las constantes generadas para los posibles valores o "no válido" si el valor no coincide con ninguno de los valores enumerados en el esquema.
`bool exists()`	Devuelve true si existe al menos un elemento.
`MemberType first()`	Devuelve la primera instancia del elemento miembro.
`MemberType operator[](unsigned int index)`	Devuelve el elemento miembro especificado por el índice.
`altova::meta::Element info()`	Devuelve un objeto para consultar información del esquema (*véase* `altova::meta::Element` 1164).
`MemberType last()`	Devuelve la última instancia del elemento miembro.
`void remove()`	Elimina todas las instancias del elemento en su primario.
`void removeAt(unsigned int index)`	Elimina la instancia del elemento especificada por el índice.
`void SetEnumerationValue(int)`	Solo se genera para tipos de enumeración. Pase una de las constantes generadas para los posibles valores a este método para establecer el valor.

15.8 Referencia de las clases generadas (C#)

Esta sección describe las clases C# que se generan con MapForce a partir de una DTD o esquema XML (*véase* *Generar código a partir de esquemas XML o DTD* 1110). Puede integrar estas clases en su código para leer, modificar y escribir documentos XML.

Nota: el código generado incluye otras clases compatibles que no se enumeran en este apartado y que pueden cambiar en el futuro.

15.8.1 Altova.Types.DateTime

Esta clase permite procesar atributos o elementos XML que tienen tipos de fecha y hora (p. ej. `xs:dateTime`).

Constructores

	Nombre	Descripción
	`DateTime(DateTime obj)`	Inicializa una instancia nueva de la clase `DateTime` en el objeto `DateTime` dado como argumento.
	`DateTime(System.DateTime newvalue)`	Inicializa una instancia nueva de la clase `DateTime` en el objeto `System.DateTime` dado como argumento.
	`DateTime(int year, int month, int day, int hour, int minute, double second, int offsetTZ)`	Inicializa una instancia nueva de la clase `DateTime` en el año, mes, día, hora, minuto, segundo y diferencia de zona horaria dados como argumentos.
	`DateTime(int year, int month, int day, int hour, int minute, double second)`	Inicializa una instancia nueva de la clase `DateTime` en el año, mes, día, hora, minuto y segundo dados como argumentos.
	`DateTime(int year, int month, int day)`	Inicializa una instancia nueva de la clase `DateTime` en el año, mes y día dados como argumentos.

Propiedades

	Nombre	Descripción
	`bool HasTimezone`	Obtiene un valor booleano que indica si `DateTime` tiene una zona horaria.
	`static DateTime Now`	Obtiene un objeto `DateTime` que está definido con la fecha y hora actuales de este equipo.
	`short TimezoneOffset`	Obtiene o establece la diferencia de zona horaria, en minutos, del objeto `DateTime`.

© 2016-2022 Altova GmbH

	Nombre	Descripción
	`System.DateTime Value`	Obtiene o establece el valor del objeto `DateTime` como valor de un `System.DateTime`.

Métodos

	Nombre	Descripción
	`int CompareTo(object obj)`	La clase `DateTime` implementa la interfaz `IComparable`. Este método compara la instancia actual de `DateTime` con otro objeto y devuelve un entero que indica si la posición de la instancia actual es anterior, posterior o igual que la del otro objeto en el criterio de ordenación. Consulte también https://msdn.microsoft.com/es-es/library/system.icomparable.compareto(v=vs.110).aspx
	`override bool Equals(object obj)`	Devuelve **true** si el objeto especificado es igual que el objeto actual. De lo contrario devuelve **false**.
	`System.DateTime GetDateTime(bool correctTZ)`	Devuelve un objeto `System.DateTime` de la instancia `Altova.Types.DateTime` actual. El argumento booleano `correctTZ` especifica si la hora del objeto devuelto debe ajustarse a la zona horaria de la instancia `Altova.Types.DateTime` actual.
	`override int GetHashCode()`	Devuelve el código hash de la instancia actual.
	`int GetWeekOfMonth()`	Devuelve el número de la semana del mes como entero.

	`static DateTime Parse( string s )`	Crea un objeto `DateTime` a partir de la cadena dada como argumento. Por ejemplo, estos valores de cadena de muestra se convertirían correctamente en un objeto `DateTime`:
		
		`2015-01-01T23:23:23`
		`2015-01-01`
		`2015-11`
		`23:23:23`
		
		Si la cadena no se puede convertir en un objeto `DateTime`, entonces se genera una excepción.
		
		Tenga en cuenta que este método es estático y solo se le puede llamar en la clase `Altova.Types.DateTime` propiamente dicha y no en una instancia de la clase.

	`static DateTime Parse(string s, DateTimeFormat format)`	Crea un objeto `DateTime` a partir de una cadena usando el formato dado como argumento. Para ver una lista de formatos posibles consulte Altova.Types.DateTimeFormat 1175.

	Nombre	Descripción
		Si la cadena no se puede convertir en un objeto `DateTime`, entonces se genera una excepción. Tenga en cuenta que este método es estático y solo se le puede llamar en la clase `Altova.Types.DateTime` propiamente dicha y no en una instancia de la clase.
	`override string ToString()`	Convierte el objeto `DateTime` en una cadena.
	`string ToString(DateTimeFormat format)`	Convierte el objeto `DateTime` en una cadena usando el formato dado como argumento. Para ver una lista de formatos posibles consulte Altova.Types.DateTimeFormat 1175.

Operadores

Nombre	Descripción
`!=`	Determina si `DateTime` a no es igual que `DateTime` b.
`<`	Determina si `DateTime` a es menor que `DateTime` b.
`<=`	Determina si `DateTime` a es menor o igual que `DateTime` b.
`==`	Determina si `DateTime` a es igual que `DateTime` b.
`>`	Determina si `DateTime` a es mayor que `DateTime` b.
`>=`	Determina si `DateTime` a es mayor o igual que `DateTime` b.

Ejemplos

Antes de usar estos ejemplos en su programa debe importar los tipos de Altova:

```
using Altova.Types;
```

Este ejemplo muestra cómo se pueden crear objetos `DateTime`:

```
protected static void DateTimeExample1()
{
   // Crear un objeto DateTime a partir de la hora actual del sistema
   Altova.Types.DateTime dt = new Altova.Types.DateTime(System.DateTime.Now);
   Console.WriteLine("La hora actual es: " + dt.ToString());

   // Crear un objeto Altova DateTime a partir de las partes (sin zona horaria)
```

 © 2016-2022 Altova GmbH

```
   Altova.Types.DateTime dt1 = new Altova.Types.DateTime(2015, 10, 12, 10, 50, 33);
   Console.WriteLine("Mi hora es : " + dt1.ToString());

   // Crear un objeto Altova DateTime a partir de las partes (con zona horaria UTC+60
minutos)
   Altova.Types.DateTime dt2 = new Altova.Types.DateTime(2015, 10, 12, 10, 50, 33, 60);
   Console.WriteLine("Mi hora con zona horaria es : " + dt2.ToString());

   // Crear un objeto Altova DateTime analizando una cadena
   Altova.Types.DateTime dt3 = Altova.Types.DateTime.Parse("2015-01-01T23:23:23");
   Console.WriteLine("La hora se creó a partir de la cadena: " + dt3.ToString());

   // Crear un objeto Altova DateTime analizando una cadena con formato de fecha de
esquema
   Altova.Types.DateTime dt4 = Altova.Types.DateTime.Parse("2015-01-01",
DateTimeFormat.W3_date);
   Console.WriteLine("La hora se creó a partir de una cadena con formato de fecha de
esquema: " + dt4.ToString());
}
```

Este ejemplo muestra cómo se puede dar formato a objetos `DateTime`:

```
protected static void DateTimeExample2()
{
   // Crear un objeto DateTime a partir de la hora actual del sistema
   Altova.Types.DateTime dt = new Altova.Types.DateTime(System.DateTime.Now);

   // Generar la fechaHora sin formato
   Console.WriteLine("Unformatted time: " + dt.ToString());

   // Generar este DateTime con formato usando lo siguiente
   Console.WriteLine("S_DateTime:       " + dt.ToString(DateTimeFormat.S_DateTime));
   Console.WriteLine("S_Days:           " + dt.ToString(DateTimeFormat.S_Days));
   Console.WriteLine("S_Seconds:        " + dt.ToString(DateTimeFormat.S_Seconds));
   Console.WriteLine("W3_date:          " + dt.ToString(DateTimeFormat.W3_date));
   Console.WriteLine("W3_dateTime:      " + dt.ToString(DateTimeFormat.W3_dateTime));
   Console.WriteLine("W3_gDay:          " + dt.ToString(DateTimeFormat.W3_gDay));
   Console.WriteLine("W3_gMonth:        " + dt.ToString(DateTimeFormat.W3_gMonth));
   Console.WriteLine("W3_gMonthDay:     " + dt.ToString(DateTimeFormat.W3_gMonthDay));
   Console.WriteLine("W3_gYear:         " + dt.ToString(DateTimeFormat.W3_gYear));
   Console.WriteLine("W3_gYearMonth:    " + dt.ToString(DateTimeFormat.W3_gYearMonth));
   Console.WriteLine("W3_time:          " + dt.ToString(DateTimeFormat.W3_time));
}
```

15.8.2 Altova.Types.DateTimeFormat

El tipo de enumeración `DateTimeFormat` tiene estos valores de constante:

Valor	Descripción	Ejemplo
S_DateTime	Da al valor el formato de un dateTime estándar, con una precisión de una décimo millonésima de segundo, incluida la zona horaria.	`2015-11-12 12:19:03.9019132+01:00`
S_Days	Da al valor el formato del número de días que han pasado desde la época UNIX.	`735913.6318973451087962962963`
S_Seconds	Da al valor el formato del número de segundos que han pasado desde la épica UNIX, con una precisión de una décimo millonésima de segundo.	`63582937678.0769062`
W3_date	Da al valor el formato de fecha de esquema.	`2015-11-12`
W3_dateTime	Da al valor el formato de fechaHora de esquema.	`2015-11-12T15:12:14.5194251`
W3_gDay	Da al valor el formato de gDay de esquema.	`---12` (si la fecha es el día 12 del mes)
W3_gMonth	Da al valor el formato de gMonth de esquema.	`--11` (si el mes es noviembre)
W3_gMonthDay	Da al valor el formato de gMonthDay de esquema.	`--11-12` (si la fecha es el día 12 de noviembre)
W3_gYear	Da al valor el formato de gYear de esquema.	`2015` (si el año es 2015)
W3_gYearMonth	Da al valor el formato de gYearMonth de esquema.	`2015-11` (si el años es 2015 y el mes es noviembre)
W3_time	Da al valor el formato de hora de esquema, con una precisión de una décimo millonésima de segundo.	`15:19:07.5582719`

© 2016-2022 Altova GmbH

15.8.3 Altova.Types.Duration

Esta clase permite procesar atributos o elementos XML de tipo `xs:duration`.

Constructores

	Nombre	Descripción
	`Duration(Duration obj)`	Inicializa una instancia nueva de la clase `Duration` en el objeto `Duration` dado como argumento.
	`Duration(System.TimeSpan newvalue)`	Inicializa una instancia nueva de la clase `Duration` en el objeto `System.TimeSpan` dado como argumento.
	`Duration(long ticks)`	Inicializa una instancia nueva de la clase `Duration` en el número de ciclos dado como argumento.
	`Duration(int newyears, int newmonths, int days, int hours, int minutes, int seconds, double partseconds, bool bnegative)`	Inicializa una instancia nueva de la clase `Duration` en una duración compuesta por las partes dadas como argumentos.

Propiedades

	Nombre	Descripción
	`int Months`	Obtiene o establece el número de meses de la instancia actual de `Duration`.
	`System.TimeSpan Value`	Obtiene o establece el valor (como `System.TimeSpan`) de la instancia actual de `Duration`.
	`int Years`	Obtiene o establece el número de años de la instancia actual de `Duration`.

Métodos

	Nombre	Descripción
	`override bool Equals(object other)`	Devuelve **true** si el objeto especificado es igual al objeto actual. De lo contrario devuelve **false**.
	`override int GetHashCode()`	Devuelve el código hash de la instancia actual.

	Nombre	Descripción
	`bool IsNegative()`	Devuelve **true** si la instancia actual de `Duration` representa una duración negativa.
	`static Duration Parse( string s, ParseType pt )`	Devuelve un objeto `Altova.Types.Duration` analizado desde una cadena dada como argumento, usando el tipo de análisis dado como argumento. Son valores válidos de tipo de análisis: **DURATION** Analizar duración asumiendo que existen las partes año, mes, día y duración. **YEARMONTH** Analizar duración asumiendo que solo existen las partes año y mes. **DAYTIME** Analizar duración asumiendo que solo existen las partes día y hora. Recuerde que este método es estático solo se le puede llamar en la clase propiamente dicha y no desde una instancia de la clase.
	`override string ToString()`	Convierte la instancia `Duration` actual en una cadena. Por ejemplo, un intervalo de tiempo de 3 horas, 4 minutos y 5 segundos se convertiría en la cadena "PT3H4M5S".
	`string ToYearMonthString()`	Convierte la instancia `Duration` actual en una cadena usando el tipo de análisis YEARMONTH.

Operadores

Nombre	Descripción
`!=`	Determina si `Duration` a no es igual que `Duration` b.
`==`	Determina si `Duration` a es igual que `Duration` b.

Ejemplos

Antes de usar estos ejemplos en su programa debe importar los tipos de Altova:

```
using Altova.Types;
```

Este ejemplo muestra cómo se pueden crear objetos `Duration`:

```
protected static void DurationExample1()
{
   // Crear un intervalo de tiempo nuevo de 3 horas, 4 minutos y 5 segundos
   System.TimeSpan ts = new TimeSpan(3, 4, 5);
```

 © 2016-2022 Altova GmbH

```
   // Crear un Duration a partir del intervalo de tiempo
   Duration dr = new Duration(ts);
   // El resultado es: PT3H4M5S
   Console.WriteLine("Duration created from TimeSpan: " + dr.ToString());

   // Crear un Altova.Types.Duration negativo a partir de 6 años, 5 meses, 4 días, 3
horas, 2 minutos, 1 segundo y .33 de un segundo
   Duration dr1 = new Duration(6, 5, 4, 3, 2, 1, .33, true);
   // El resultado es: -P6Y5M4DT3H2M1.33S
   Console.WriteLine("Duration created from parts: " + dr1.ToString());

   // Crear un Duration a partir de una cadena usando el tipo de análisis DAYTIME
   Duration dr2 = Altova.Types.Duration.Parse("-P4DT3H2M1S", Duration.ParseType.DAYTIME);
   // El resultado es: -P4DT3H2M1S
   Console.WriteLine("Duration created from string: " + dr2.ToString());

   // Crear una duración a partir de ciclos
   Duration dr3 = new Duration(System.DateTime.UtcNow.Ticks);
   // Generar el resultado
   Console.WriteLine("Duration created from ticks: " + dr3.ToString());
}
```

Este ejemplo muestra cómo se pueden obtener valores de los objetos `Duration`:

```
protected static void DurationExample2()
{
   // Crear un Altova.Types.Duration negativo a partir de 6 años, 5 meses, 4 días, 3
horas, 2 minutos, 1 segundo y .33 de un segundo
   Duration dr = new Duration(6, 5, 4, 3, 2, 1, .33, true);
   // El resultado es: -P6Y5M4DT3H2M1.33S
   Console.WriteLine("The complete duration is: " + dr.ToString());

   // Obtener solo la parte año y mes como cadena
   string dr1 = dr.ToYearMonthString();
   Console.WriteLine("The YEARMONTH part is: " + dr1);

   // Obtener el número de años de la duración
   Console.WriteLine("Years: " + dr.Years);

   // Obtener el número de meses de la duración
   Console.WriteLine("Months: " + dr.Months);
}
```

15.8.4 Altova.Xml.Meta.Attribute

Esta clase permite acceder a la información del esquema sobre clases generadas a partir de atributos. Recuerde que esta clase no tiene como objetivo aportar información dinámica sobre las distintas instancias de un atributo de un documento XML. Por el contrario, permite obtener información mediante programación sobre cualquier atributo que esté definido en el esquema XML.

Propiedades

	Nombre	Descripción
	`SimpleType DataType`	Devuelve el tipo del contenido del atributo.
	`string LocalName`	Devuelve el nombre local del atributo.
	`string NamespaceURI`	Devuelve el URI de espacio de nombres del atributo.
	`XmlQualifiedName QualifiedName`	Devuelve el nombre completo del atributo.
	`bool Required()`	Devuelve true si el atributo es obligatorio.

15.8.5 Altova.Xml.Meta.ComplexType

Esta clase permite acceder a información del esquema sobre clases generadas a partir de tipos complejos. Recuerde que esta clase no tiene como objetivo aportar información dinámica sobre las distintas instancias de un tipo complejo de un documento XML. Por el contrario, permite obtener información mediante programación sobre cualquier tipo complejo que esté definido en el esquema XML.

Propiedades

	Nombre	Descripción
	`Attribute[] Attributes`	Devuelve una lista de todos los atributos.
	`ComplexType BaseType`	Devuelve el tipo base de este tipo o null si no existe el tipo base.
	`SimpleType ContentType`	Devuelve el tipo simple del contenido.
	`Element[] Elements`	Devuelve una lista de todos los elementos.
	`string LocalName`	Devuelve el nombre local de este tipo.
	`string NamespaceURI`	Devuelve el URI de espacio de nombres del tipo.
	`XmlQualifiedName QualifiedName`	Devuelve el nombre completo de este tipo.

Métodos

	Nombre	Descripción
	`ComplexType BaseType`	Devuelve el tipo base de este tipo.

© 2016-2022 Altova GmbH

	Nombre	Descripción
	`bool Equals(obj)`	Comprueba si dos objetos de información remiten al mismo tipo, basándose en la comparación de los nombres completos. Devuelve true si el tipo tiene el mismo nombre completo.
	`Attribute FindAttribute(string localName, string namespaceURI)`	Busca el atributo con el nombre local y el URI de espacio de nombres especificados.
	`Element FindElement(string localName, string namespaceURI)`	Busca el elemento con el nombre local y el URI de espacio de nombres especificados.

15.8.6 Altova.Xml.Meta.Element

Esta clase permite acceder a la información del esquema sobre clases generadas a partir de elementos. Recuerde que esta clase no tiene como objetivo aportar información dinámica sobre las distintas instancias de un elemento de un documento XML. Por el contrario, permite obtener información mediante programación sobre cualquier elemento que esté definido en el esquema XML.

Propiedades

	Nombre	Descripción
	`ComplexType DataType`	Devuelve el tipo del elemento. Recuerde que este siempre es un tipo complejo incluso si se declaró como simple en el esquema original. Use `ContentType` del objeto devuelto para obtener el tipo de contenido simple.
	`string` `LocalName`	Devuelve el nombre local del elemento.
	`int` `MaxOccurs`	Devuelve el valor `maxOccurs` definido en el esquema.
	`int` `MinOccurs`	Devuelve el valor `minOccurs` definido en el esquema.
	`string` `NamespaceURI`	Devuelve el URI de espacio de nombres del elemento.
	`XmlQualifiedName QualifiedName`	Devuelve el nombre completo del elemento.

15.8.7 Altova.Xml.Meta.SimpleType

Esta clase permite acceder a la información del esquema sobre clases generadas a partir de tipos simples. Recuerde que esta clase no tiene como objetivo aportar información dinámica sobre las distintas instancias de un tipo simple de un documento XML. Por el contrario, permite obtener información mediante programación

sobre cualquier tipo simple que esté definido en el esquema XML.

Propiedades

	Nombre	Descripción
	`SimpleType BaseType`	Devuelve el tipo base de este tipo.
	`string[] Enumerations`	Devuelve una lista de todas las facetas de enumeración.
	`int FractionDigits`	Devuelve el valor de esta faceta.
	`int Length`	Devuelve el valor de esta faceta.
	`string LocalName`	Devuelve el nombre local de este tipo.
	`string MaxExclusive`	Devuelve el valor de esta faceta.
	`string MaxInclusive`	Devuelve el valor de esta faceta.
	`int MaxLength`	Devuelve el valor de esta faceta.
	`string MinExclusive`	Devuelve el valor de esta faceta.
	`string MinInclusive`	Devuelve el valor de esta faceta.
	`int MinLength`	Devuelve el valor de esta faceta.
	`string NamespaceURI`	Devuelve el URI de espacio de nombres del tipo.
	`string[] Patterns`	Devuelve las facetas de patrón o null si no hay patrones especificados.
	`XmlQualifiedName QualifiedName`	Devuelve el nombre completo de este tipo.
	`int TotalDigits`	Devuelve el valor de esta faceta.
	`WhitespaceType Whitespace`	Devuelve la faceta de normalización de espacios en blanco.

15.8.8 [SuEsquema].[Doc]

Cuando se genera código a partir de un esquema XML, el código generado ofrece una clase de documento que se llama igual que el esquema. Esta clase tiene como miembros todos los elementos raíz posibles, así como los métodos que aparecen a continuación. Observe que en los nombres de método, `CDoc` corresponde al nombre de la clase de documento generada.

 © 2016-2022 Altova GmbH

Métodos

	Nombre	Descripción
	static Doc CreateDocument()	Crea un documento XML nuevo vacío.
	static Doc CreateDocument(**string** encoding)	Crea un documento XML nuevo vacío, con codificación de tipo "encoding"
	static void DeclareAllNamespacesFromSchema(**Altova.Xml.TypeBase** node)	Declara todos los espacios de nombres del esquema XML en el elemento dado como argumento (que suele ser el elemento XML raíz). Es útil llamar a este método si su esquema tiene varias declaraciones de espacios de nombres, cada una de ellas está asignada a un prefijo y quiere declararlas todas en el elemento dado como argumento.
	static Doc LoadFromBinary(**byte**[] binary)	Carga un documento XML desde una matriz de bytes.
	static Doc LoadFromFile(**string** filename)	Carga un documento XML desde un archivo.
	static Doc LoadFromString(**string** xmlstring)	Carga un documento XML desde una cadena de texto.
	byte[] SaveToBinary(**bool** prettyPrint)	Guarda un documento XML en una matriz de bytes, con formato pretty-print opcional.
	byte[] SaveToBinary(**bool** prettyPrint, **string** encoding)	Guarda un documento XML en una matriz de bytes (con formato pretty-print opcional) con la codificación especificada.
	byte[] SaveToBinary(**bool** prettyPrint, **string** encoding, **bool** bBigEndian, **bool** bBOM)	Guarda un documento XML en una matriz de bytes, (con formato pretty-print opcional) con la codificación, el orden de bytes y la marca BOM especificados.
	void SaveToFile(**string** fileName, **bool** prettyPrint)	Guarda un documento XML en un archivo, con formato pretty-print opcional.
	void SaveToFile(**string** fileName, **bool** prettyPrint, **bool** omitXmlDecl)	Guarda un documento XML en un archivo, con formato pretty-print opcional. Si el argumento omitXmlDecl tiene el valor true, la declaración XML no se escribirá.
	void SaveToFile(**string** fileName, **bool** prettyPrint, **bool** omitXmlDecl, **string** encoding)	Guarda un documento XML en un archivo, (con formato pretty-print opcional) con la codificación especificada. Si el argumento omitXmlDecl tiene el valor true, la declaración XML no se escribirá.
	void SaveToFile(**string** fileName, **bool** prettyPrint,	Guarda un documento XML en un archivo, (con formato pretty-print opcional) con la codificación y los caracteres de final de línea especificados.

	Nombre	Descripción
	`string encoding, string lineend)`	
	`void SaveToFile(string fileName, bool prettyPrint, bool omitXmlDecl, string encoding, string lineend)`	Guarda un documento XML en un archivo, (con formato pretty-print opcional) con la codificación y los caracteres de final de línea especificados. Si el argumento `omitXmlDecl` tiene el valor true, la declaración XML no se escribirá.
	`void SaveToFile(string fileName, bool prettyPrint, bool omitXmlDecl, string encoding, bool bBigEndian, bool bBOM, string lineend)`	Guarda un documento XML en un archivo, (con formato pretty-print opcional) con la codificación, el orden de bytes, la marca BOM y los caracteres de final de línea especificados. Si el argumento `omitXmlDecl` tiene el valor true, la declaración XML no se escribirá.
	`void SaveToFileWithLineEnd(string fileName, bool prettyPrint, bool omitXmlDecl, string lineend)`	Guarda un documento XML en un archivo, con formato pretty-print opcional y con caracteres de final de línea. Si el argumento `omitXmlDecl` tiene el valor true, la declaración XML no se escribirá.
	`string SaveToString(bool prettyPrint)`	Guarda un documento XML en un archivo, con formato pretty-print opcional.
	`string SaveToString(bool prettyPrint, bool omitXmlDecl)`	Guarda un documento XML en un archivo, con formato pretty-print opcional. Si el argumento `omitXmlDecl` tiene el valor true, la declaración XML no se escribirá.
	`void SetDTDLocation(string dtdLocation)`	Agrega una declaración DOCTYPE con el ID de sistema especificado. Es obligatorio que ya exista un elemento raíz.
	`void SetSchemaLocation(string schemaLocation)`	Agrega un atributo xsi:schemaLocation o xsi:noNamespaceSchemaLocation al elemento raíz. Es obligatorio que ya exista un elemento raíz.

15.8.9 [SuEsquema].[ElementType]

Esta clase permite manipular elementos XML desde el esquema. Los métodos de esta clase se pueden llamar en elementos, pero no en el documento XML. Observe que no necesita instanciar la clase directamente para poder llamar a los métodos de esta clase. Cualquier elemento creado con los métodos **`append()`** o **`appendWithPrefix()`** es de tipo **`append()`** or **`appendWithPrefix()`**.

Métodos

	Nombre	Descripción
	`void DeclareNamespace(string prefix, string nsURI)`	Este método toma dos argumentos de tipo cadena: el prefijo y el URI de espacio de nombres que quiere usar. El prefijo dado como argumento se asigna al valor del URI de espacio de nombres dado como argumento. Si el prefijo dado como

 © 2016-2022 Altova GmbH

	Nombre	Descripción

		argumento está vacío, el método crea o sobrescribe la declaración de espacio de nombres predeterminada del elemento.
		
		Por ejemplo, imagine que el documento XML tiene un elemento XML llamado "purchase". Si llama a
		
		`purchase.DeclareNamespace("ord", "http://OrderTypes");`
		
		entonces el documento XML se convierte en
		
		`<purchase xmlns:ord="http://OrderTypes" />`
		
		Otro ejemplo. Si llama a:
		
		`purchase.DeclareNamespace("", "http://OrderTypes");`
		
		entonces el documento XML se convierte en
		
		`<purchase xmlns="http://OrderTypes" />`
		
		Nota: el espacio de nombres declarado se usa al anexar elementos secundarios o atributos posteriormente, conforme a estas normas:
		
		1. Si el espacio de nombres secundario es el predeterminado, use un prefijo vacío.
		2. Si el espacio de nombres secundario es igual que el del elemento matriz, use el prefijo matriz.
		3. De lo contrario, busque el prefijo más cercano desde el elemento matriz hasta el principio usando el algoritmo de búsqueda que se describe en la sección "B.2: Namespace Prefix Lookup" de https://www.w3.org/TR/2002/WD-DOM-Level-3-Core-20021022/namespaces-algorithms.html.
		4. Si no se encuentra ningún prefijo para el espacio de nombres del elemento, entonces use un prefijo vacío.

15.8.10 [SuTipoEsquema].MemberAttribute

Cuando se genera código a partir de un esquema XML, se crea una clase como esta por cada atributo miembro de un tipo. En las tablas que aparecen a continuación `AttributeType` corresponde al tipo del atributo miembro propiamente dicho.

Métodos

	Nombre	Descripción
	`bool Exists()`	Devuelve true si el atributo existe.
	`void Remove()`	Quita el atributo de su elemento primario.

Propiedades

	Nombre	Descripción
	`int EnumerationValue`	Solo se genera para tipos de enumeración. Establece u obtiene el valor del atributo con ayuda de una de las constantes generadas para los posibles valores. Devuelve "no válido" si el valor no coincide con ninguno de los valores enumerados en el esquema.
	`Altova.Xml.Meta.Attribute Info`	Devuelve un objeto para consultar información del esquema (*véase* `Altova.Xml.Meta.Attribute` 1179).
	`AttributeType Value`	Establece u obtiene el valor del atributo.

15.8.11 [SuTipoEsquema].MemberElement

Cuando se genera código a partir de un esquema XML, se crea una clase como esta por cada elemento miembro de un tipo. La clase implementa la interfaz estándar `System.Collections.IEnumerable` para que pueda utilizarse con la instrucción `foreach`.

En la tabla que aparece a continuación `MemberType` corresponde al tipo del elemento miembro propiamente dicho.

Métodos

	Nombre	Descripción
	`MemberType Append()`	Crea un elemento nuevo y lo anexa a su primario.

© 2016-2022 Altova GmbH

	Nombre	Descripción
	`MemberType AppendWithPrefix(string prefix)`	Crea un elemento nuevo que tiene el prefijo dado como argumento y lo anexa a su elemento superior. Para ver un ejemplo consulte Ejemplo: orden de compra 1147.
	`MemberType At(int index)`	Devuelve el elemento miembro especificado por el índice.
	`System.Collections.IEnumerator GetEnumerator()`	Devuelve un objeto para recorrer instancias del elemento miembro.
	`void Remove()`	Elimina todas las instancias del elemento en su primario.
	`void RemoveAt(int index)`	Elimina la instancia del elemento especificada por el índice.

Propiedades

	Nombre	Descripción
	`int Count`	Devuelve el recuento de elementos.
	`int EnumerationValue`	Solo se genera para tipos de enumeración. Establece u obtiene el valor del elemento con ayuda de una de las constantes generadas para los posibles valores. Devuelve "no válido" si el valor no coincide con ninguno de los valores enumerados en el esquema.
	`bool Exists`	Devuelve true si existe al menos un elemento.
	`MemberType First`	Devuelve la primera instancia del elemento miembro.
	`Altova.Xml.Meta.Element Info`	Devuelve un objeto para consultar información del esquema (véase `Altova.Xml.Meta.Element` 1181).
	`MemberType Last`	Devuelve la última instancia del elemento miembro.
	`MemberType this[int index]`	Devuelve el elemento miembro especificado por el índice.
	`MemberType Value`	Establece u obtiene el contenido del elemento (solo se genera si el elemento puede tener contenido mixto o simple).

15.9 Referencia de las clases generadas (Java)

Esta sección describe las clases Java que se generan con MapForce a partir de una DTD o esquema XML (*véase* *Generar código a partir de esquemas XML o DTD* 1110). Puede integrar estas clases en su código para leer, modificar y escribir documentos XML.

Nota: el código generado incluye otras clases compatibles que no se enumeran en este apartado y que pueden cambiar en el futuro.

© 2016-2022 Altova GmbH

15.9.1 com.altova.types.DateTime

Esta clase permite procesar atributos o elementos XML que tienen tipos de fecha y hora (p. ej. `xs:dateTime`).

Constructores

	Nombre	Descripción
C	**public** DateTime()	Inicializa una instancia nueva de la clase `DateTime` en un valor vacío.
C	**public** DateTime(DateTime newvalue)	Inicializa una instancia nueva de la clase `DateTime` en el valor `DateTime` dado como argumento.
C	**public** DateTime(**int** newyear, **int** newmonth, **int** newday, **int** newhour, **int** newminute, **int** newsecond, **double** newpartsecond, **int** newoffsetTZ)	Inicializa una instancia nueva de la clase `DateTime` en el año, mes, día, hora, minuto, segundo, la parte fraccionaria del segundo y la zona horaria dados como argumentos. La parte fraccionaria del segundo newpartsecond debe estar comprendida entre 0 y 1. La diferencia de zona horaria newoffsetTZ puede ser negativa o positivo y se expresa en minutos.
C	**public** DateTime(**int** newyear, **int** newmonth, **int** newday, **int** newhour, **int** newminute, **int** newsecond, **double** newpartsecond)	Inicializa una instancia nueva de la clase `DateTime` en el año, mes, día, hora, minuto, segundo y la parte fraccionaria de un segundo dados como argumentos.
C	**public** DateTime(**int** newyear, **int** newmonth, **int** newday)	Inicializa una instancia nueva de la clase `DateTime` en el año, mes y día dados como argumentos.
C	**public** DateTime(Calendar newvalue)	Inicializa una instancia nueva de la clase `DateTime` en el valor `java.util.Calendar` dado como argumento.

Métodos

	Nombre	Descripción
S	**static** DateTime now()	Devuelve la hora actual como un objeto `DateTime`.
S	**static** DateTime parse(String s)	Devuelve un objeto `DateTime` analizado desde el valor de cadena dado como argumento. Por ejemplo, estos valores de cadena de muestra se convertirían correctamente en un objeto `DateTime`: 2015-11-24T12:54:47.969+01:00 2015-11-24T12:54:47 2015-11-24

	Nombre	Descripción
	int getDay()	Devuelve el día de la instancia DateTime actual.
	int getHour()	Devuelve la hora de la instancia DateTime actual.
	int getMillisecond()	Devuelve el milisegundo de la instancia DateTime actual, como valor entero.
	int getMinute()	Devuelve el minuto de la instancia DateTime actual.
	int getMonth()	Devuelve el mes de la instancia DateTime actual.
	double getPartSecond()	Devuelve la parte fraccionaria del segundo de la instancia DateTime actual, como valor **double**. El valor devuelto es mayor que cero y menor que uno. Por ejemplo: 0.313
	int getSecond()	Devuelve el segundo de la instancia DateTime actual.
	int getTimezoneOffset()	Devuelve la diferencia de zona horaria, en minutos, de la instancia DateTime actual. Por ejemplo, la zona horaria "UTC-01:00" se devolvería como: -60
	Calendar getValue()	Devuelve la instancia DateTime actual como un valor java.util.Calendar.
	int getWeekday()	Devuelve el día de la semana de la instancia DateTime actual. Los valores están comprendidos entre el 0 y el 6, siendo 0 el lunes (ISO-8601).
	int getYear()	Devuelve el año de la instancia DateTime actual.
	int hasTimezone()	Devuelve información sobre la zona horaria de la instancia DateTime actual. Los valores devueltos posibles son: CalendarBase.TZ_MISSING — No está definida la diferencia de zona horaria. CalendarBase.TZ_UTC — La zona horaria es UTC. CalendarBase.TZ_OFFSET — Está definida una diferencia de zona horaria.
	void setDay(**int** nDay)	Establece el día de la instancia DateTime actual en el valor dado como argumento.
	void setHasTimezone(**int** nHasTZ)	Establece la información de zona horaria de la instancia DateTime actual en el valor dado como argumento. Este método puede utilizarse para eliminar la información de zona horaria o establecer la zona horaria en UTC (hora

© 2016-2022 Altova GmbH

	Nombre	Descripción
		universal coordinada). Los valores válidos para el argumento nHasTZ son: `CalendarBase.TZ_MISSING` Establecer la diferencia de zona horaria como indefinida. `CalendarBase.TZ_UTC` Establecer la zona horaria en UTC. `CalendarBase.TZ_OFFSET` Si el objeto actual tiene una diferencia de zona horaria, dejarla sin cambiar.
	`void setHour( int nHour )`	Establece la hora de la instancia `DateTime` actual en el valor dado como argumento.
	`void setMinute( int nMinute )`	Establece el minuto de la instancia `DateTime` actual en el valor dado como argumento.
	`void setMonth( int nMonth )`	Establece el mes de la instancia `DateTime` actual en el valor dado como argumento.
	`void setPartSecond( double nPartSecond )`	Establece la parte fraccionaria del segundo de la instancia `DateTime` actual en el valor dado como argumento.
	`void setSecond( int nSecond )`	Establece el segundo de la instancia `DateTime` actual en el valor dado como argumento.
	`void setTimezoneOffset( int nOffsetTZ )`	Establece la diferencia de zona horaria de la instancia `DateTime` actual en el valor dado como argumento. El valor nOffsetTZ debe ser un entero (negativo o positivo) y debe expresarse en minutos.
	`void setYear( int nYear )`	Establece el año de la instancia `DateTime` actual en el valor dado como argumento.
	`String toString()`	Devuelve la representación de cadena de la instancia `DateTime` actual. Por ejemplo: `2015-11-24T15:50:56.968+01:00`

Ejemplos

Antes de usar este ejemplo en su programa debe importar los tipos de Altova:

```
import com.altova.types.*;
```

Este ejemplo muestra cómo se pueden crear objetos `DateTime`:

```
protected static void DateTimeExample1()
{
   // Inicializar una instancia nueva de la clase DateTime en la hora actual
   DateTime dt = new DateTime(DateTime.now());
   System.out.println("DateTime created from current date and time: " + dt.toString());

   // Inicializar una instancia nueva de la clase DateTime aportando las partes
   DateTime dt1 = new DateTime(2015, 11, 23, 14, 30, 24, .459);
   System.out.println("DateTime from parts (no timezone): " + dt1.toString());

   // Inicializar una instancia nueva de la clase DateTime aportando las partes
   DateTime dt2 = new DateTime(2015, 11, 24, 14, 30, 24, .459, -60);
   System.out.println("DateTime from parts (with negative timezone): " + dt2.toString());

   // Inicializar una instancia nueva de la clase DateTime analizando un valor de cadena
       DateTime dt3 = DateTime.parse("2015-11-24T12:54:47.969+01:00");
   System.out.println("DateTime parsed from string: " + dt3.toString());
}
```

Este ejemplo muestra cómo se pueden obtener valores de objetos `DateTime`:

```
protected static void DateTimeExample2()
   {
      // Inicializar una instancia nueva de la clase DateTime en la hora actual
      DateTime dt = new DateTime(DateTime.now());

      // Generar el año, mes y día de esta instancia DateTime con formato
      String str1 = String.format("Year: %d; Month: %d; Day: %d;", dt.getYear(),
dt.getMonth(), dt.getDay());
      System.out.println(str1);

      // Generar la hora, el minuto y el segundo de esta instancia DateTime con formato
      String str2 = String.format("Hour: %d; Minute: %d; Second: %d;", dt.getHour(),
dt.getMinute(), dt.getSecond());
      System.out.println(str2);

      // Devolver la zona horaria (en minutos) de esta instancia DateTime
      System.out.println("Timezone: " + dt.getTimezoneOffset());

      // Obtener DateTime como valor java.util.Calendar
      java.util.Calendar dt_java = dt.getValue();
      System.out.println("" + dt_java.toString());

      // Devolver el día de la semana de esta instancia DateTime
      System.out.println("Weekday: " + dt.getWeekday());

      // Comprobar si la instancia DateTime tiene definida una zona horaria
      switch(dt.hasTimezone())
      {
         case CalendarBase.TZ_MISSING:
            System.out.println("Sin zona horaria.");
            break;
         case CalendarBase.TZ_UTC:
```

 © 2016-2022 Altova GmbH

```
            System.out.println("La zona horaria es UTC.");
            break;
         case CalendarBase.TZ_OFFSET:
            System.out.println("Este objeto tiene una zona horaria.");
            break;
         default:
            System.out.println("Imposible determinar si se definió una zona horaria.");
            break;
      }
   }
```

Este ejemplo muestra cómo se puede cambiar la diferencia horaria de un objeto `DateTime`:

```
protected static void DateTimeExample3()
{
   // Crear un objeto DateTime nuevo con zona horaria -0100 UTC
   DateTime dt = new DateTime(2015, 11, 24, 14, 30, 24, .459, -60);
   // Generar el valor antes del cambio
   System.out.println("Before: " + dt.toString());
   // Cambiar la diferencia a +0100 UTC
   dt.setTimezoneOffset(60);
   // Generar el valor después del cambio
   System.out.println("After:  " + dt.toString());
}
```

15.9.2 com.altova.types.Duration

Esta clase permite procesar atributos o elementos XML de tipo `xs:duration`.

Constructores

	Nombre	Descripción
C	`Duration(Duration newvalue)`	Inicializa una instancia nueva de la clase `Duration` en el objeto `Duration` dado como argumento.
C	`Duration(int newyear, int newmonth, int newday, int newhour, int newminute, int newsecond, double newpartsecond, boolean newisnegative)`	Inicializa una instancia nueva de la clase `Duration` en una duración construida con las partes dadas como argumentos.

Métodos

	Nombre	Descripción
S	`static Duration getFromDayTime( int newday, int newhour, int newminute, int newsecond, double newpartsecond )`	Devuelve un objeto `Duration` creado a partir del número de días, horas, minutos, segundos y partes fraccionarias de segundo dados como argumentos.
S	`static Duration getFromYearMonth( int newyear, int newmonth )`	Devuelve un objeto `Duration` creado a partir del número de años y meses dados como argumentos.
S	`static Duration parse( String s )`	Devuelve un objeto `Duration` creado a partir de la cadena dada como argumento. Por ejemplo, la cadena - P1Y1M1DT1H1M1.333S puede utilizarse para crear una duración negativa de un año, un mes, un día, una hora, un minuto, un segundo y 0.333 partes fraccionarias de un segundo. Para crear una duración negativa basta con anexar el signo menos (-) a la cadena.
S	`static Duration parse( String s, ParseType pt )`	Devuelve un objeto `Duration` creado a partir de la cadena dada como argumento, usando un formato de análisis concreto. El formato de análisis puede ser cualquiera de estos: **ParseType.DAYTIME** Puede utilizarse cuando la cadena **s** está compuesta por cualquiera de estos: días, horas, minutos,

© 2016-2022 Altova GmbH

	Nombre	Descripción
		segundos, partes fraccionarias de segundo. Por ejemplo - P4DT4H4M4.774S. **ParseType.DURATION** Puede utilizarse cuando la cadena **s** está compuesta por cualquiera de estos: años, meses, días, horas, minutos, segundos, partes fraccionarias de segundo. Por ejemplo P1Y1M1DT1H1M1.333S. **ParseType.YEARMONTH** Puede utilizarse cuando la cadena **s** está compuesta por cualquiera de estos: años, meses. Por ejemplo: P3Y2M.
	`int getDay()`	Devuelve el número de días de la instancia `Duration` actual.
	`long getDayTimeValue()`	Devuelve el valor de día y hora (en milisegundos) de la instancia `Duration` actual. Años y meses se omiten.
	`int getHour()`	Devuelve el número de horas de la instancia `Duration` actual.
	`int getMillisecond()`	Devuelve el número de milisegundos de la instancia `Duration` actual.
	`int getMinute()`	Devuelve el número de minutos de la instancia `Duration` actual.
	`int getMonth()`	Devuelve el número de meses de la instancia `Duration` actual.
	`double getPartSecond()`	Devuelve el partes fraccionarias de segundo de la instancia `Duration` actual.
	`int getSecond()`	Devuelve el número de segundos de la instancia `Duration` actual.
	`int getYear()`	Devuelve el número de años de la instancia `Duration` actual.
	`int getYearMonthValue()`	Devuelve el valor de año y mes (en meses) de la instancia `Duration` actual. Días, horas, segundos y milisegundos se omiten.
	`boolean isNegative()`	Devuelve el valor booleano **true** si la instancia `Duration` actual es negativa.
	`void setDayTimeValue(long l)`	Establece la duración en el número de milisegundos dados como argumento, afectando solo a la parte día y hora de la duración.

	Nombre	Descripción
	`void setNegative( boolean isnegative )`	Convierte la instancia `Duration` actual en una duración negativa.
	`void setYearMonthValue(int l)`	Establece la duración en el número de meses dados como argumento. Esto solo afecta la parte años y meses de la duración.
	`String toString()`	Devuelve la representación de cadena de la instancia `Duration`. Por ejemplo: `-P4DT4H4M4.774S`
	`String toYearMonthString()`	Devuelve la representación de cadena de la parte YearMonth de la instancia `Duration` actual. Por ejemplo: `P1Y2M`

Ejemplos

Antes de usar este ejemplo en su programa debe importar los tipos de Altova:

```
import com.altova.types.*;
import com.altova.types.Duration.ParseType;
```

Este ejemplo muestra cómo se pueden crear objetos `Duration`:

```
protected static void ExampleDuration()
{
   // Crear una duración negativa de 1 año, 1 mes, 1 día, 1 hora, 1 minuto, 1 segundo y
0.333 partes fraccionarias de segundo
   Duration dr = new Duration(1, 1, 1, 1, 1, 1, .333, true);

   // Crear una duración a partir de un objeto Duration que ya existe
   Duration dr1 = new Duration(dr);

   // Crear una duración de 4 días, 4 horas, 4 minutos, 4 segundos, .774 partes
fraccionarias de segundo
   Duration dr2 = Duration.getFromDayTime(4, 4, 4, 4, .774);

   // Crear una duración de 3 años y 2 meses
   Duration dr3 = Duration.getFromYearMonth(3, 2);

   // Crear una duración a partir de una cadena
   Duration dr4 = Duration.parse("-P4DT4H4M4.774S");

   // Crear una duración a partir de una cadena, usando formatos de análisis concretos
   Duration dr5 = Duration.parse("-P1Y1M1DT1H1M1.333S", ParseType.DURATION);
   Duration dr6 = Duration.parse("P3Y2M", ParseType.YEARMONTH);
```

 © 2016-2022 Altova GmbH

```
   Duration dr7 = Duration.parse("-P4DT4H4M4.774S", ParseType.DAYTIME);
}
```

Este ejemplo muestra cómo se puede obtener y establecer el valor de objetos `Duration`:

```
protected static void DurationExample2()
{
   // Crear una duración de 1 año, 2 meses, 3 días, 4 horas, 5 minutos, 6 segundos y 333
milisegundos
   Duration dr = new Duration(1, 2, 3, 4, 5, 6, .333, false);
   // Generar el número de días de esta duración
   System.out.println(dr.getDay());

   // Crear una duración positiva de un año y 333 milisegundos
   Duration dr1 = new Duration(1, 0, 0, 0, 0, 0, .333, false);
   // Generar el valor de día y hora en milisegundos
   System.out.println(dr1.getDayTimeValue());

   // Crear una duración positiva de 1 año, 1 mes, 1 día, 1 hora, 1 minuto, 1 segundo y
333 milisegundos
   Duration dr2 = new Duration(1, 1, 1, 1, 1, 1, .333, false);
   // Generar el valor de año y mes en meses
   System.out.println(dr2.getYearMonthValue());

   // Crear una duración positiva de 1 año y 1 un mes
   Duration dr3 = new Duration(1, 1, 0, 0, 0, 0, 0, false);
   // Generar el valor
   System.out.println("The duration is now: " + dr3.toString());
   // Establecer la parte DayTime de la duración en 1000 milisegundos
   dr3.setDayTimeValue(1000);
   // Generar el valor
   System.out.println("The duration is now: " + dr3.toString());
   // Generar la parte de mes y año de la duración
   dr3.setYearMonthValue(1);
   // Generar el valor
   System.out.println("The duration is now: " + dr3.toString());
   // Generar la parte de mes y año de la duración
   System.out.println("The YearMonth part of the duration is: " +
dr3.toYearMonthString());
}
```

15.9.3 com.altova.xml.meta.Attribute

Esta clase permite acceder a la información del esquema sobre clases generadas a partir de atributos. Recuerde que esta clase no tiene como objetivo aportar información dinámica sobre las distintas instancias de un atributo de un documento XML. Por el contrario, permite obtener información mediante programación sobre cualquier atributo que esté definido en el esquema XML.

Métodos

	Nombre	Descripción
	`SimpleType getDataType()`	Devuelve el tipo del contenido del atributo.
	`String getLocalName()`	Devuelve el nombre local del atributo.
	`String getNamespaceURI()`	Devuelve el URI de espacio de nombres del atributo.
	`boolean` `isRequired()`	Devuelve true si el atributo es obligatorio.

15.9.4 com.altova.xml.meta.ComplexType

Esta clase permite acceder a información del esquema sobre clases generadas a partir de tipos complejos. Recuerde que esta clase no tiene como objetivo aportar información dinámica sobre las distintas instancias de un tipo complejo de un documento XML. Por el contrario, permite obtener información mediante programación sobre cualquier tipo complejo que esté definido en el esquema XML.

Métodos

	Nombre	Descripción
	`Attribute findAttribute(String localName, String namespaceURI)`	Busca el atributo con el nombre local y el URI de espacio de nombres especificados.
	`Element findElement(String localName, String namespaceURI)`	Busca el elemento con el nombre local y el URI de espacio de nombres especificados.
	`Attribute[] GetAttributes()`	Devuelve una lista de todos los atributos.
	`ComplexType getBaseType()`	Devuelve el tipo base de este tipo.
	`SimpleType getContentType()`	Devuelve el tipo simple del contenido.
	`Element[] GetElements()`	Devuelve una lista de todos los elementos.
	`String getLocalName()`	Devuelve el nombre local de este tipo.
	`String getNamespaceURI()`	Devuelve el URI de espacio de nombres del tipo.

© 2016-2022 Altova GmbH

15.9.5 com.altova.xml.meta.Element

Esta clase permite acceder a la información del esquema sobre clases generadas a partir de elementos. Recuerde que esta clase no tiene como objetivo aportar información dinámica sobre las distintas instancias de un elemento de un documento XML. Por el contrario, permite obtener información mediante programación sobre cualquier elemento que esté definido en el esquema XML.

Métodos

	Nombre	Descripción
	`ComplexType getDataType()`	Devuelve el tipo del elemento. Recuerde que este siempre es un tipo complejo incluso si se declaró como simple en el esquema original. Use `getContentType()` del objeto devuelto para obtener el tipo de contenido simple.
	`String getLocalName()`	Devuelve el nombre local del elemento.
	`int getMaxOccurs()`	Devuelve el valor maxOccurs definido en el esquema.
	`int getMinOccurs()`	Devuelve el valor minOccurs definido en el esquema.
	`String getNamespaceURI()`	Devuelve el URI de espacio de nombres del elemento.

15.9.6 com.altova.xml.meta.SimpleType

Esta clase permite acceder a la información del esquema sobre clases generadas a partir de tipos simples. Recuerde que esta clase no tiene como objetivo aportar información dinámica sobre las distintas instancias de un tipo simple de un documento XML. Por el contrario, permite obtener información mediante programación sobre cualquier tipo simple que esté definido en el esquema XML.

Métodos

	Nombre	Descripción
	`SimpleType getBaseType()`	Devuelve el tipo base de este tipo.
	`String[] getEnumerations()`	Devuelve una matriz de todas las facetas de enumeración.
	`int getFractionDigits()`	Devuelve el valor de esta faceta.
	`int getLength()`	Devuelve el valor de esta faceta.
	`String getLocalName()`	Devuelve el nombre local del tipo.
	`String getMaxExclusive()`	Devuelve el valor de esta faceta.
	`String getMaxInclusive()`	Devuelve el valor de esta faceta.

	Nombre	Descripción
	`int getMaxLength()`	Devuelve el valor de esta faceta.
	`String getMinExclusive()`	Devuelve el valor de esta faceta.
	`String getMinInclusive()`	Devuelve el valor de esta faceta.
	`int getMinLength()`	Devuelve el valor de esta faceta.
	`String getNamespaceURI()`	Devuelve el URI de espacio de nombres del tipo.
	`String[] getPatterns()`	Devuelve una matriz de todas las facetas de patrón.
	`int getTotalDigits()`	Devuelve el valor de esta faceta.
	`int getWhitespace()`	Devuelve el valor de la faceta de espacio en blanco, que es uno de estos: `com.altova.typeinfo.WhitespaceType.Whitespace_Unknown` `com.altova.typeinfo.WhitespaceType.Whitespace_Preserve` `com.altova.typeinfo.WhitespaceType.Whitespace_Replace` `com.altova.typeinfo.WhitespaceType.Whitespace_Collapse`

15.9.7 com.[SuEsquema].[Doc]

Cuando se genera código a partir de un esquema XML, el código generado ofrece una clase de documento que se llama igual que el esquema. Esta clase tiene como miembros todos los elementos raíz posibles, así como los miembros que aparecen a continuación. Observe que en los nombres de método, "Doc" corresponde al nombre de la clase de documento generada.

Métodos

	Nombre	Descripción
S	`static Doc createDocument()`	Crea un documento XML nuevo vacío.
S	`static void declareAllNamespacesFromSchema(com.altova.xml.TypeBase node)`	Declara todos los espacios de nombres del esquema XML en el elemento dado como argumento (que suele ser el elemento XML raíz). Es útil llamar a este método si su esquema tiene varias declaraciones de espacios de nombres, cada una de ellas está asignada a un prefijo y quiere declararlas todas en el elemento dado como argumento.
S	`static Doc loadFromBinary(byte[] xml)`	Carga un documento XML desde una matriz de bytes.

 © 2016-2022 Altova GmbH

	Nombre	Descripción
S	**static** Doc loadFromFile(String fileName)	Carga un documento XML desde un archivo.
S	**static** Doc loadFromString(String xml)	Carga un documento XML desde una cadena de texto.
	byte[] saveToBinary(**boolean** prettyPrint)	Guarda un documento XML en una matriz de bytes, con formato pretty-print opcional.
	byte[] saveToBinary(**boolean** prettyPrint, String encoding)	Guarda un documento XML en una matriz de bytes (con formato pretty-print opcional) con la codificación especificada.
	byte[] saveToBinary(**boolean** prettyPrint, String encoding, **boolean** bigEndian, **boolean** writeBOM)	Guarda un documento XML en una matriz de bytes (con formato pretty-print opcional) con la codificación especificada. Para codificaciones Unicode se puede especificar el orden de bytes y la marca BOM Unicode.
	void saveToFile(String fileName, **boolean** prettyPrint)	Guarda un documento XML en un archivo, con formato pretty-print opcional.
	void saveToFile(String fileName, **boolean** prettyPrint, **boolean** omitXmlDecl)	Guarda un documento XML en un archivo, con formato pretty-print opcional, con codificación UTF-8. Si el argumento omitXmlDecl tiene el valor true, la declaración XML no se escribirá.
	void saveToFile(String fileName, **boolean** prettyPrint, **boolean** omitXmlDecl, String encoding)	Guarda un documento XML en un archivo (con formato pretty-print opcional) con la codificación especificada. Si el argumento omitXmlDecl tiene el valor true, la declaración XML no se escribirá.
	void saveToFile(String fileName, **boolean** prettyPrint, **boolean** omitXmlDecl, String encoding, **boolean** bBigEndian, **boolean** bBOM)	Guarda un documento XML en un archivo (con formato pretty-print opcional) con la codificación especificada. Si el argumento omitXmlDecl tiene el valor true, la declaración XML no se escribirá. Para codificaciones Unicode se puede especificar el orden de bytes y la marca BOM Unicode.
	void saveToFile(String fileName, **boolean** prettyPrint, String encoding)	Guarda un documento XML en un archivo, (con formato pretty-print opcional) con la codificación especificada.
	void saveToFile(String fileName, **boolean** prettyPrint, String encoding, **boolean** bBigEndian, **boolean** bBOM)	Guarda un documento XML en un archivo (con formato pretty-print opcional) con la codificación especificada. Para codificaciones Unicode se puede especificar el orden de bytes y la marca BOM Unicode.
	String saveToString(**boolean** prettyPrint)	Guarda un documento XML en una cadena de texto, con formato pretty-print opcional.
	String saveToString(**boolean** prettyPrint, **boolean**	Guarda un documento XML en una cadena de texto, con formato pretty-print opcional. Si el argumento omitXmlDecl tiene el valor

	Nombre	Descripción
	`omitXmlDecl)`	true, la declaración XML no se escribirá.
	`void` `setSchemaLocation(String schemaLocation)`	Agrega un atributo `xsi:schemaLocation` o `xsi:noNamespaceSchemaLocation` al elemento raíz. Es obligatorio que ya exista un elemento raíz.

15.9.8 com.[SuEsquema].[ElementType]

Esta clase permite manipular elementos XML desde el esquema. Los métodos de esta clase se pueden llamar en elementos, pero no en el documento XML. Observe que no necesita instanciar la clase directamente para poder llamar a los métodos de esta clase. Cualquier elemento creado con los métodos **`append()`** o **`appendWithPrefix()`** es de tipo **`append()`** or **`appendWithPrefix()`**.

Métodos

	Nombre	Descripción

	`void declareNamespace(String prefix, String nsURI)`	Este método toma dos argumentos de tipo cadena: el prefijo y el URI de espacio de nombres que quiere usar. El prefijo dado como argumento se asigna al valor del URI de espacio de nombres dado como argumento. Si el prefijo dado como argumento está vacío, el método crea o sobrescribe la declaración de espacio de nombres predeterminada del elemento.
		
		Por ejemplo, imagine que el documento XML tiene un elemento XML llamado "purchase". Si llama a
		
		`purchase.declareNamespace("ord", "http://OrderTypes");`
		
		entonces el documento XML se convierte en
		
		`<purchase xmlns:ord="http://OrderTypes" />`
		
		Otro ejemplo. Si llama a:
		
		`purchase.declareNamespace("", "http://OrderTypes");`
		
		entonces el documento XML se convierte en
		
		`<purchase xmlns="http://OrderTypes" />`

 © 2016-2022 Altova GmbH

	Nombre	Descripción
		Nota: el espacio de nombres declarado se usa al anexar elementos secundarios o atributos posteriormente, conforme a estas normas: 1. Si el espacio de nombres secundario es el predeterminado, use un prefijo vacío. 2. Si el espacio de nombres secundario es igual que el del elemento matriz, use el prefijo matriz. 3. De lo contrario, busque el prefijo más cercano desde el elemento matriz hasta el principio usando el algoritmo de búsqueda que se describe en la sección "B.2: Namespace Prefix Lookup" de https://www.w3.org/TR/2002/WD-DOM-Level-3-Core-20021022/namespaces-algorithms.html. 4. Si no se encuentra ningún prefijo para el espacio de nombres del elemento, entonces use un prefijo vacío.

15.9.9 com.[SuEsquema].[SuTipoEsquema].MemberAttribute

Cuando se genera código a partir de un esquema XML, se crea una clase como esta por cada atributo miembro de un tipo. En la tabla que aparece a continuación `AttributeType` corresponde al tipo del atributo miembro propiamente dicho.

Métodos

	Nombre	Descripción
	`boolean exists()`	Devuelve true si el atributo existe.
	`int getEnumerationValue()`	Solo se genera para tipos de enumeración. Devuelve una de las constantes generadas para los posibles valores o "no válido" si el valor no coincide con ninguno de los valores enumerados en el esquema.
	`com.altova.xml.meta.Attribute getInfo()`	Devuelve un objeto para consultar información del esquema (véase `com.altova.xml.meta.Attribute` 1197).
	`AttributeType getValue()`	Obtiene el valor del atributo.
	`void remove()`	Quita el atributo de su elemento primario.
	`void setEnumerationValue(int)`	Solo se genera para tipos de enumeración. Pase una de las constantes generadas para los posibles

	Nombre	Descripción
		valores a este método para establecer el valor.
	`void setValue(AttributeType value)`	Establece el valor del atributo.

15.9.10 com.[SuEsquema].[SuTipoEsquema].MemberElement

Cuando se genera código a partir de un esquema XML, se crea una clase como esta por cada elemento miembro de un tipo. En la tabla que aparece a continuación `MemberType` corresponde al tipo del elemento miembro propiamente dicho.

Métodos

	Nombre	Descripción
	`MemberType append()`	Crea un elemento nuevo y lo anexa a su primario.
	`MemberType appendWithPrefix(String prefix)`	Crea un elemento nuevo que tiene el prefijo dado como argumento y lo anexa a su elemento superior. Para ver un ejemplo consulte Ejemplo: orden de compra 1147.
	`MemberType at(int index)`	Devuelve la instancia del elemento miembro en el índice especificado.
	`int count()`	Devuelve el recuento de elementos.
	`boolean exists()`	Devuelve true si existe al menos un elemento.
	`MemberType first()`	Devuelve la primera instancia del elemento miembro.
	`int getEnumerationValue()`	Solo se genera para tipos de enumeración. Devuelve una de las constantes generadas para los posibles valores o "no válido" si el valor no coincide con ninguno de los valores enumerados en el esquema.
	`com.altova.xml.meta.Element getInfo()`	Devuelve un objeto para consultar información del esquema (véase `com.altova.xml.meta.Element` 1199).
	`MemberType getValue()`	Obtiene el contexto del elemento (solo se genera si el elemento puede tener contenido simple o mixto).

© 2016-2022 Altova GmbH

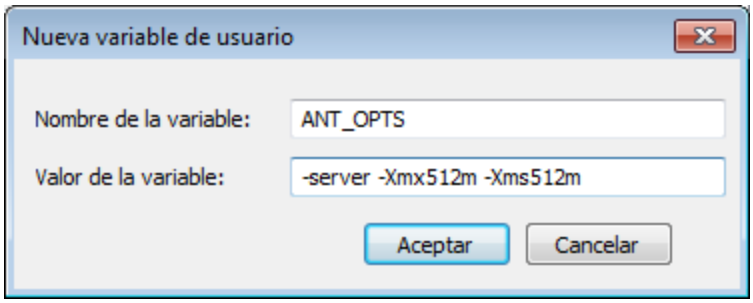
	Nombre	Descripción
	`java.util.Iterator iterator()`	Devuelve un objeto para recorrer instancias del elemento miembro.
	`MemberType last()`	Devuelve la última instancia del elemento miembro.
	`void remove()`	Elimina todas las instancias del elemento en su primario.
	`void removeAt(int index)`	Elimina la instancia del elemento especificada por el índice.
	`void setEnumerationValue(int index)`	Solo se genera para tipos de enumeración. Pase una de las constantes generadas para los posibles valores a este método para establecer el valor.
	`void setValue(MemberType value)`	Establece el contexto del elemento (solo se genera si el elemento puede tener contenido simple o mixto).

15.10 Consejos para la generación de código

Solución para las excepciones de memoria insuficiente durante la compilación de Java

Las asignaciones de datos complejas que incluyen esquemas de gran tamaño pueden producir una gran cantidad de código, lo cual puede dar lugar a una excepción java.lang.OutofMemory durante la compilación con Ant. Para resolver esto:

- Añada la variable de entorno **ANT_OPTS**, que asigna al compilador ciertas opciones de Ant, como la opción de memoria. Configure su valor como se muestra a continuación.



- Para asegurarse de que el compilador y el código generado se ejecutan en el mismo proceso que Ant, cambie el valor del atributo **fork**, de build.xml, por `false`.

Quizás sea necesario personalizar los valores, dependiendo de la cantidad de memoria disponible en su equipo y del tamaño del proyecto en el que esté trabajando. Para más información, consulte su documentación de Java VM.

> Es posible que, al ejecutar el comando `ant jar`, reciba un mensaje de error parecido a este: "`[...] el archivo contiene más de 65535 entidades`". Para evitar este error, recomendamos utilizar Ant 1.9 o superior y, además, añadir `zip64mode="as-needed"` al elemento `<jar>` en el archivo **build.xml**.

Reservar nombres de métodos

Cuando personalice la generación de código usando los archivos spl que vienen con la aplicación, quizás sea necesario reservar nombres para evitar conflictos con otros símbolos. Para hacerlo:

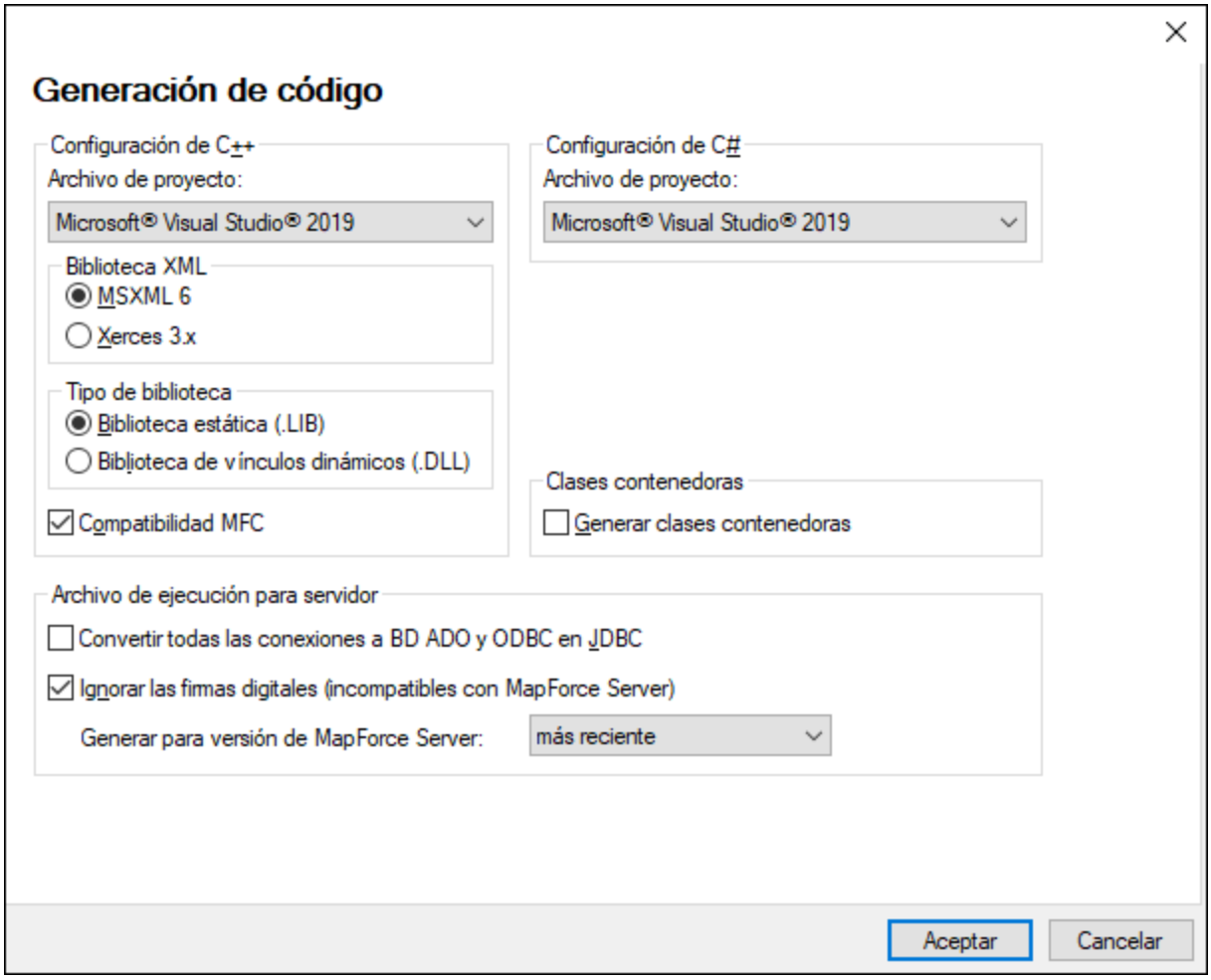
1. Navegue hasta el directorio de instalación del programa, por ejemplo `C:\Archivos de programa\Altova\MapForce2023\spl\java\`.
2. En el subdirectorio **spl** ubique el directorio que corresponda al lenguaje de programación, por ejemplo **..\spl\java**.
3. Abra el archivo **settings.spl** e inserte una línea nueva en la sección de reserva, por ejemplo **`reserve "miPalabraReservada"`**.
4. Vuelva a generar el código de programa.

© 2016-2022 Altova GmbH

15.11 Opciones del generador de código

Para ver o cambiar la configuración de MapForce para la generación de código:

- En el menú **Herramientas** haga clic en **Opciones** y después en **Generación de código**.



Estas son las opciones de configuración del generador de código:

Configuración de C++	Define la configuración del compilador para el entorno C++: • La edición de Visual Studio (2013, 2015, 2017, 2019, 2022) • La biblioteca XML (MSXML, Xerces 3.x) • Si deben generarse bibliotecas estáticas o dinámicas • Si debe generarse código compatible con MFC o no

Configuración de C#	Seleccione respectivamente la opción **Microsoft .NET Core 3.1**, **Microsoft .NET 5.0** o **Microsoft .NET 6.0** para generar una solución de Visual Studio para estas plataformas. Si necesita usar la plataforma **.NET Framework** para un proyecto de Visual Studio concreto, seleccione cualquiera de las opciones de **Microsoft Visual Studio 2010-2019**; en este caso la solución generada apuntará a la versión de .NET Framework que corresponde a la versión de Visual Studio que haya elegido.
Clases contenedoras	Permite generar clases contenedoras para esquemas XML (*véase Generar código a partir de esquemas XML o DTD* 1110). Estas clases contenedoras se pueden usar en otro código que incluya el código generado con MapForce.
Archivo de ejecución del servidor	Estas opciones son relevantes cuando se compilan asignaciones en archivos de ejecución de MapForce Server. No afectan a la generación de código C#, C++ ni Java. Para más información consulte el apartado Compilar asignaciones en archivos de ejecución de MapForce Server 955.

© 2016-2022 Altova GmbH

15.12 SPL: el lenguaje de programación Spy

Esta sección ofrece una introducción al lenguaje de programación Spy (en adelante, *SPL*), que es el lenguaje de plantillas del generador de código.

En esta sección asumimos que el usuario tiene cierta experiencia en programación y ciertos conocimientos sobre operadores, variables, funciones y clases, así como sobre conceptos básicos de programación orientada a objetos, que se usa en gran medida en SPL.

Las plantillas utilizadas por MapForce están en la carpeta **...\MapForce\spl**. Puede usar estos archivos para orientarse a la hora de crear sus propias plantillas.

¿Cómo funciona el generador de código?

Las entradas del generador de código son los archivos de plantilla (`.spl`) y el modelo de objetos que viene con XMLSpy. Los archivos de plantilla contienen instrucciones SPL (para crear archivos, leer información del modelo de objetos y realizar cálculos) intercaladas con fragmentos de código literal en el lenguaje de programación de destino.

El archivo de plantilla lo interpreta el generador de código y produce archivos de código fuente **.cpp**, **.java** y **.cs** , archivos de proyecto o cualquier otro tipo de proyecto, dependiendo de la plantilla. El código fuente se puede compilar después en un archivo ejecutable que accede a los datos XML descritos por el archivo de esquema.

Los archivos SPL tienen acceso a una gran variedad de información que se recopila de los esquemas de origen. Recuerde que un archivo SPL no está unido a ningún esquema concreto, sino que permite acceder a todos los esquemas. Asegúrese de que escribe sus archivos SPL de forma genérica y de que evita estructuras, etc. que pertenezcan a esquemas concretos.

Ejemplo: crear un archivo nuevo en SPL

Se trata de un archivo SPL muy básico. Crea un archivo llamado **test.cpp** y coloca la instrucción `include` dentro del archivo. El comando de cierre (`close`) termina la plantilla.

```
[create "test.cpp"]
#include "stdafx.h"
[close]
```

15.12.1 Estructura básica de SPL

Un archivo SPL contiene texto literal, que se debe reproducir, intercalado con instrucciones del generador de código.

Las instrucciones del generador de código van entre corchetes: "**[**" y "**]**".
Entre cada par de corchetes se pueden introducir varias instrucciones. Las instrucciones se separan con una línea nueva o con dos puntos "**:**".

Estos dos ejemplos son igual de válidos:

Ejemplo nº1

```
[$x = 42
$x = $x + 1]
```

Ejemplo nº2

```
[$x = 42: $x = $x + 1]
```

Agregar texto a los archivos

El texto que no aparece entre corchetes **[ ]** se escribe directamente en el archivo de salida actual. Si no hay un archivo de salida actual, el texto se pasa por alto (consulte el apartado Crear archivos de salida[1214]). Para reproducir corchetes literales, introduzca el carácter de escape **** de la siguiente manera: **\[** texto **\].** Para reproducir el carácter ****, use ****.

Comentarios

Los comentarios introducidos en un bloque de instrucciones empiezan con el carácter **'** y terminan en la línea siguiente o con un carácter de cierre de bloque: **]**.

15.12.2 Declaraciones

Estas instrucciones se evalúan mientras se analiza el archivo de plantilla SPL. A estas instrucciones **no** les afectan las instrucciones de control de flujo como condiciones, bucles o subrutinas y siempre se evalúan exactamente una vez.

Estas palabras clave, como todas las palabras clave de SPL, no distinguen entre mayúsculas y minúsculas.

Recuerde que todas estas declaraciones deben estar dentro de un bloque delimitado por corchetes.

map *clave nombreasignación* to valor *[, clave to valor]...*

Esta instrucción añade información a una asignación. Consulte el resto del apartado para ver usos concretos de esta instrucción.

map *schemanativetype tipoesquema* to *tipoindicado*

El tipo integrado de XML Schema indicado se asignará al tipo o clase nativos indicados, usando el formateador indicado. Esta configuración es propia de la versión 2007r3 o posterior. *Tipoindicado* es el nombre de un tipo o clase nativos, seguido de una coma y de la instancia de clase del formateador.

Ejemplo:
```
map schemanativetype "double" to "double,Altova::DoubleFormatter"
```

map type *tipoesquema* to *nombreclase*

El tipo integrado de XML Schema indicado se asignará a la clase indicada. Esta configuración es propia de la generación de código para la versión 2007 o anterior.

 © 2016-2022 Altova GmbH

Ejemplo:
```
map type "float" to "CSchemaFloat"
```

default *configuración* is *valor*

Esta instrucción permite determinar cómo se derivan los nombres de clase y miembro desde el esquema XML. Recuerde que los nombres de configuración sí distinguen entre mayúsculas y minúsculas.

Ejemplo:
```
default "InvalidCharReplacement" is "_"
```

Nombre de configuración	Explicación
`ValidFirstCharSet`	Caracteres permitidos para iniciar un identificador
`ValidCharSet`	Caracteres permitidos para otros caracteres en un identificador
`InvalidCharReplacement`	El carácter que reemplazará todos los caracteres de los nombres que no están en la configuración ValidCharSet
`AnonTypePrefix`	Prefijo para nombres de tipos anónimos*
`AnonTypeSuffix`	Sufijo para nombres de tipos anónimos*
`ClassNamePrefix`	Prefijo para los nombres de las clases generadas
`ClassNameSuffix`	Sufijo para los nombres de las clases generadas
`EnumerationPrefix`	Prefijo para constantes simbólicas declaradas para valores de enumeración
`EnumerationUpperCase`	"on" para poner los nombres de constante de enumeración en mayúsculas
`FallbackName`	Si un nombre tiene sólo caracteres que no están en ValidCharSet, use este

* Los nombres de tipos anónimos se construyen con AnonTypePrefix + nombre de elemento + AnonTypeSuffix

reserve *palabra*

Añade la palabra indicada a la lista de palabras reservadas. Esto garantiza que esa palabra se genere como nombre de clase o miembro.

Ejemplo:
```
reserve "while"
```

include *nombrearchivo*

Ejemplo:
```
include "Module.cpp"
```

incluye el archivo indicado como SPL de origen. Esto permite dividir su plantilla en varios archivos para editarlos y gestionarlos más fácilmente.

15.12.3 Variables

Los archivos SPL importantes exigen el uso de variables. Algunas variables vienen predefinidas 1212 por el generador de código y se pueden crear variables nuevas con solo asignarles valores.

El carácter **$** se usa cuando se **declara** o **usa** una variable, un nombre de variable siempre tiene el prefijo **$**. Los nombres de variable distinguen **entre mayúsculas y minúsculas**.

Tipos de variables:

- entero, que también se usa como binario, siendo 0 equivalente a false y cualquier otro valor equivale a true
- cadena
- objeto, que viene dado por MapForce
- iterador, ver instrucción foreach 1216

El tipo de variable se declara en la primera asignación de valor:

```
[$x = 0]
```

ahora x es un entero.

```
[$x = "cadenaDePrueba"]
```

ahora x es una cadena.

Cadenas

Las constantes de cadena siempre aparecen entre comillas dobles, como en el ejemplo anterior. **\n** y **\t** dentro de comillas dobles se considera una línea nueva y una tabulación, respectivamente. **\"** es una comilla doble literal y **** es una barra diagonal inversa. Las constantes de cadena también pueden ocupar más de una línea.

Para la concatenación de cadenas se usa el carácter **&**:

```
[$BasePath = $outputpath & "/" & $JavaPackageDir]
```

Objetos

Los objetos representan la información que contiene el esquema XML así como las estructuras de bases de datos, los archivos de texto y las asignaciones. Los objetos tienen propiedades. Con el operador **.** puede acceder a estas propiedades. En SPL no puede crear objetos nuevos (vienen predefinidos por el generador de código y se derivan de la asignación de entrada), pero se pueden asignar objetos a variables.

Ejemplo:

```
class [=$class.Name]
```

Este ejemplo reproduce la palabra "class" seguida de un espacio y el valor de la propiedad **Name** del objeto **$class**.

15.12.4 Variables predefinidas

Después de que el generador de código termine de analizar un archivo de esquema, los objetos de esta tabla existen en el motor de plantillas.

 © 2016-2022 Altova GmbH

Nombre	Tipo	Descripción
$schematype	entero	1 para DTD, 2 para XML Schema
$TheLibrary	Biblioteca 1221	La biblioteca que se deriva del esquema XML o de la DTD
$module	cadena	Nombre del esquema de origen sin la extensión
$outputpath	cadena	Ruta de acceso de salida especificada por el usuario o ruta de acceso de salida predeterminada

Para generación de código **C++** solamente:

Nombre	Tipo	Descripción
$domtype	entero	1 para MSXML, 2 para Xerces
$libtype	entero	1 para static LIB, 2 para DLL
$mfc	booleano	True si se marcó la casilla *Compatibilidad con MFC*
$VSVersion	entero	Especifica la versión de Visual Studio. Valores válidos: **0** Sin proyecto de Visual Studio **2010** Visual Studio 2010 **2013** Visual Studio 2013 **2015** Visual Studio 2015 **2017** Visual Studio 2017 **2019** Visual Studio 2019

Para generación de código **C#** solamente:

Nombre	Tipo	Descripción
$VSVersion	entero	Especifica la versión de Visual Studio. Valores válidos: **0** Sin proyecto de Visual Studio **2010** Visual Studio 2010 **2013** Visual Studio 2013 **2015** Visual Studio 2015 **2017** Visual Studio 2017 **2019** Visual Studio 2019

15.12.5 Crear archivos de salida

Estas instrucciones sirven para crear archivos de salida con ayuda de la función de generación de código.

Recuerde que todas estas instrucciones deben estar dentro de un bloque delimitado por corchetes.

create *nombrearchivo*

crea un archivo nuevo. El archivo tiene que cerrarse con la instrucción **close**. El resultado se escribe en el archivo indicado.

Ejemplo:
```
[create $outputpath & "/" & $JavaPackageDir & "/" & $application.Name & ".java"]
package [=$JavaPackageName];

public class [=$application.Name]Application {
...
}
[close]
```

close

cierra el archivo de salida actual

=$*variable*

escribe el valor de la variable indicada en el archivo de salida actual.

Ejemplo:
```
[$x = 20+3]
El resultado de su cálculo es [=$x] - ¡que tenga un buen día!
```

- El archivo de salida incluirá:

```
El resultado de su cálculo es 23 - ¡que tenga un buen día!
```

write *cadena*

escribe la cadena en el archivo de salida actual.

Ejemplo:
```
[write "C" & $name]
```

Esto también se puede escribir de esta manera:
```
C[=$name]
```

filecopy *origen* to *destino*

copia el archivo de origen al archivo de destino, sin interpretarlo.

 © 2016-2022 Altova GmbH

Ejemplo:
```
filecopy "java/mapforce/mapforce.png" to $outputpath & "/" & $JavaPackageDir &
"/mapforce.png"
```

15.12.6 Operadores

Los operadores de SPL funcionan como en casi todos los lenguajes de programación.

A continuación ofrecemos la lista de operadores de SPL ordenados por orden de prioridad (de mayor a menor):

.	Acceso a la propiedad del objeto
( )	Agrupación de expresiones
true	Constante binaria "true"
false	Constante binaria "false"
&	Concatenación de cadenas de texto
-	Signo para un número negativo
not	Negación lógica
*	Multiplicación
/	División
%	Resto
+	Suma
-	Resta
<=	Menor o igual que
<	Menor que
>=	Mayor o igual que
>	Mayor que
=	Igual
<>	No igual
and	Conjunción lógica (con evaluación en cortocircuito)
or	Disyunción lógica (con evaluación en cortocircuito)
=	Asignación

15.12.7 Condiciones

En SPL puede usar instrucciones `if` estándar. Esta es la sintaxis para estas instrucciones:

```
if condición
       instrucciones
else
```

```
        instrucciones
endif
```

o sin la parte **else**:

```
if condición
        instrucciones
endif
```

Nota: observe que la condición no aparece entre paréntesis.
Al igual que en otros lenguajes de programación, las condiciones se construyen con operadores[1215] lógicos y de comparación.

Ejemplo:

```
[if $namespace.ContainsPublicClasses and $namespace.Prefix <> ""]
        lo que usted quiera ['inserta lo que usted quiera en el archivo de resultados]
[endif]
```

Instrucción *switch*

SPL también incluye una instrucción de control para múltiples selecciones.

Sintaxis:

```
switch $variable
        case X:
                instrucciones
        case Y:
        case Z:
                instrucciones
        default:
                instrucciones
endswitch
```

Las etiquetas "case" son constantes o variables.

Al igual que C, SPL no admite el paso implícito de una etiqueta case a otra y, por tanto, no es necesario usar la instrucción `break`.

15.12.8 Colecciones y foreach

Colecciones e iteradores

Una colección contiene varios objetos, como una matriz normal y corriente. Los iteradores sirven para almacenar e incrementar índices de matriz al acceder a objetos.

Sintaxis:

```
foreach iterador in colección
        instrucciones
next
```

Ejemplo nº1:

```
[foreach $class in $classes
       if not $class.IsInternal
              ]      class [=$class.Name];
[      endif
next]
```

Ejemplo 2:

```
[foreach $i in 1 To 3
      Write "// Step " & $i & "\n"
      ‘ Do some work
next]
```

En la primera línea **$classes** es el **objeto global** 1212 de todos los tipos generados. Es una colección de diferentes objetos de clase.

La instrucción **foreach** recorre todos los elementos de **$classes** y en cada uno de ellos ejecuta el código que sigue a la instrucción hasta llegar a la instrucción **next**.

En cada iteración se asigna **$class** al siguiente objeto de clase. Simplemente se trabaja con el objeto de clase en lugar de usar `classes[i]->Name()` como en C++.

Los iteradores de colección tienen estas propiedades adicionales:

`Index`	El índice actual, empezando por 0
`IsFirst`	true si el objeto actual es el primer objeto de la colección (el índice es 0)
`IsLast`	true si el objeto actual es el último objeto de la colección
`Current`	El objecto actual (este es implícito si no se especifica y se puede omitir)

Ejemplo:

```
[foreach $enum in $facet.Enumeration
       if not $enum.IsFirst
              ], [
       endif
       ]"[=$enum.Value]"[
next]
```

Rutinas para la manipulación de colecciones:

collection **SortByName**(bAscending)
devuelve una colección cuyos elementos están ordenados por nombre (con distinción de mayúsculas y minúsculas) en orden ascendente o descendente.

collection **SortByNameNoCase**(bAscending)
devuelve una colección cuyos elementos están ordenados por nombre (sin distinción de mayúsculas y minúsculas) en orden ascendente o descendente.

Ejemplo:

```
$SortedNestedClassifier = $Class.nestedClassifier.SortByNameNoCase( true )
```

collection **SortByKind**(bAscending)
devuelve una colección cuyos elementos están ordenados por tipo (p. ej. clase, interfaz...) en orden ascendente o descendente.

collection **SortByKindAndName**(bAscendingKind, bAscendingName)
devuelve una colección cuyos elementos están ordenados por tipo (p. ej. clase, interfaz...) en orden ascendente o descendente y si el tipo es idéntico al nombre (con distinción de mayúsculas y minúsculas en orden ascendente o descendente).

collection **SortByKindAndNameNoCase**(bAscending)
devuelve una colección cuyos elementos están ordenados por tipo (p. ej. clase, interfaz...) en orden ascendente o descendente y si el tipo es idéntico al nombre (sin distinción de mayúsculas y minúsculas en orden ascendente o descendente).

15.12.9 Subrutinas

El generador de código admite subrutinas en forma de procedimientos o funciones.

Características:

- Se pasan valores por valor y por referencia
- Parámetros locales/globales (locales dentro de las subrutinas)
- Variables locales
- Invocación
- Invocación recurrente (las subrutinas se pueden llamar a sí mismas)

15.12.9.1 Declaración de subrutinas

Subrutinas

Ejemplo de sintaxis:

```
Sub SimpleSub()
... lines of code
EndSub
```

- **Sub** es la palabra clave que denota el procedimiento.
- **SimpleSub** es el nombre asignado a la subrutina.
- Los **paréntesis** pueden contener una lista de parámetros.
- El bloque de código de una subrutina empieza inmediatamente después del paréntesis de cierre.
- **EndSub** denota el final del bloque de código.

Nota: no se permiten **declaraciones** de subrutinas recursivas o en cascada (es decir, una subrutina no puede contener otra subrutina).

Parámetros

Los parámetros también se pueden pasar con procedimientos usando esta sintaxis:

 © 2016-2022 Altova GmbH

- Todos los parámetros deben ser variables.
- Las variables deben tener el prefijo **$**.
- Las variables locales se definen en una subrutina.
- Las variables globales se declaran de forma explícita, fuera de las subrutinas.
- Si hay varios parámetros, se separan con comas dentro de los paréntesis.
- Los parámetros pueden pasar valores.

Parámetros: pasar valores

Los parámetros se pueden pasar de dos maneras: por valor y por referencia, usando las palabras clave **ByVal** y **ByRef** respectivamente.

Sintaxis:

```
' definir sub CompleteSub()
[Sub CompleteSub( $param, ByVal $paramByValue, ByRef $paramByRef )
] ...
```

- **ByVal** especifica que el parámetro se pasa por valor. Recuerde que la mayoría del os objetos solamente se pueden pasar por referencia.
- **ByRef** especifica que el parámetro se pasa por referencia. Se trata del método predeterminado si no se especifica ByVal ni ByRef.

Valores devueltos de funciones

Para devolver un valor desde una subrutina use la instrucción **return**. Este tipo de función se puede llamar desde dentro de una expresión.

Ejemplo:

```
' definir una función
[Sub MakeQualifiedName( ByVal $namespacePrefix, ByVal $localName )
if $namespacePrefix = ""
  return $localName
else
  return $namespacePrefix & ":" & $localName
endif
EndSub
]
```

15.12.9.2 Invocación de subrutinas

Use **call** para invocar una subrutina, seguido del nombre y los parámetros del procedimiento (si procede).

```
Call SimpleSub()
```

o

```
Call CompleteSub( "FirstParameter", $ParamByValue, $ParamByRef )
```

Invocación de funciones

Para invocar una función (cualquier subrutina que contenga una instrucción **return**) use su nombre dentro de una expresión. No use la instrucción **call** para llamar a funciones.

Ejemplo:

```
$QName = MakeQualifiedName($namespace, "entry")
```

15.12.9.3 Ejemplo

Este ejemplo muestra la declaración e invocación de subrutinas.

```
[create $outputpath & $module & "output.txt"

' definir sub SimpleSub()
Sub SimpleSub()
]SimpleSub() called
[endsub

' ejecutar sub SimpleSub()
Call SimpleSub()

$ParamByValue        = "Original Value"
]ParamByValue        = [=$ParamByValue]
[$ParamByRef  = "Original Value"
]ParamByRef   = [=$ParamByRef]

' definir sub CompleteSub()
[Sub CompleteSub( $param, ByVal $paramByValue, ByRef $paramByRef )
]CompleteSub called.
       param = [=$param]
       paramByValue = [=$paramByValue]
       paramByRef = [=$paramByRef]
[$ParamByRef = "Local Variable"
$paramByValue = "new value"
$paramByRef = "new value"
]      Set values inside Sub
[$ParamByRef = "Local Variable"
$paramByValue = "new value"
$paramByRef = "new value"
]CompleteSub finished.
[endsub

' ejecutar sub CompleteSub()
Call CompleteSub( "FirstParameter", $ParamByValue, $ParamByRef )
]
ParamByValue=[=$ParamByValue]
ParamByRef=[=$ParamByRef]
[
Close
]
```

15.12.10 Tipos integrados

Esta sección describe las propiedades de los tipos integrados usados en las variables predefinidas 1212 que describen el esquema analizado.

15.12.10.1 Library

Este objeto representa toda la biblioteca generada a partir del esquema XML o de la DTD.

Propiedad	Tipo	Descripción
SchemaNamespaces	Colección Namespace 1221	Los espacios de nombres de esta biblioteca
SchemaFilename	string	Nombre del XSD o DTD del que se deriva esta biblioteca
SchemaType	integer	1 para la DTD, 2 para el esquema XML
Guid	string	Un ID único global
CodeName	string	El nombre de la biblioteca generada (derivado del nombre del archivo de esquema)

15.12.10.2 Namespace

Se genera un objeto namespace por cada espacio de nombres del esquema XML. Los componentes del esquema que no están en ningún espacio de nombres se incluyen en un objeto namespace especial con un NamespaceURI vacío. Recuerde que para las DTD también se derivan espacios de nombres de los atributos cuyo nombre comienza por "xmlns".

Propiedad	Tipo	Descripción
CodeName	string	Nombre para el código generado (derivado del prefijo)
LocalName	string	Prefijo del espacio de nombres
NamespaceURI	string	URI del espacio de nombres
Types	Colección Type 1221	Todos los tipos incluidos en este espacio de nombres
Library	Library 1221	La biblioteca que contiene este espacio de nombres

15.12.10.3 Type

Este objeto representa un tipo simple o complejo. Se usa para generar una clase en el lenguaje de destino. Hay un tipo adicional por cada biblioteca que representa al documento, que tiene como miembros todos los elementos raíz posibles.
Los tipos anónimos tienen la propiedad LocalName vacía.

Propiedad	Tipo	Descripción
CodeName	string	Nombre para el código generado (derivado del nombre local o de la declaración de objeto primario)
LocalName	string	Nombre original en el esquema
Namespace	Namespace 1221	Espacio de nombres que contiene este tipo

Attributes	Colección Member 1222	Atributos que contiene este tipo*
Elements	Colección Member 1222	Elementos secundarios que contiene este tipo
IsSimpleType	boolean	True para tipos simples y false para tipos complejos
IsDerived	boolean	True si el tipo se deriva de otro tipo, lo cual también se representa con un objeto Type
IsDerivedByExtension	boolean	True si el tipo se deriva por extensión
IsDerivedByRestriction	boolean	True si el tipo se deriva por restricción
IsDerivedByUnion	boolean	True si el tipo se deriva mediante union
IsDerivedByList	boolean	True si el tipo se deriva mediante list
BaseType	Type	El tipo base de este tipo (si IsDerived es true)
IsDocumentRootType	boolean	True si este tipo representa al documento propiamente dicho
Library	Library 1221	La biblioteca que contiene este tipo
IsFinal	boolean	True si se declaró como final en el esquema
IsMixed	boolean	True si este tipo puede tener contenido mixto
IsAbstract	boolean	True si este tipo se declaró como abstracto
IsGlobal	boolean	True si este tipo se declaró globalmente en el esquema
IsAnonymous	boolean	True si este tipo se declaró localmente en un elemento

Para tipos simples solamente:

Propiedad	Tipo	Descripción
IsNativeBound	boolean	True si existe un NativeBinding para este tipo
NativeBinding	NativeBinding 1223	El NativeBinding para este tipo
Facets	Facets 1224	Las facetas de este tipo
Whitespace	string	Acceso rápido a la faceta Whitespace

* Los tipos complejos que tienen contenido de texto (es decir, tipos que tienen contenido mixto y un complexType con simpleContent) tienen un miembro atributo adicional sin nombre que representa el contenido de texto.

15.12.10.4 Member

Este objeto representa a un atributo o elemento del esquema XML. Se usa para crear miembros de clases de tipos.

Propiedad	Tipo	Descripción
CodeName	string	El nombre para el código generado (derivado del nombre local o de la declaración del objeto

© 2016-2022 Altova GmbH

		primario)
LocalName	string	El nombre original en el esquema. Aparece vacía para el miembro especial que representa el contenido de texto de tipos complejos.
NamespaceURI	string	El URI de espacio de nombres de este elemento/atributo dentro de documentos/secuencias de instancia XML.
DeclaringType	Type 1221	El tipo que originalmente declara el miembro (igual a containingType para miembros no heredados)
ContainingType	Type 1221	El tipo del cual este es miembro
DataType	Type 1221	El tipo de datos del contenido de este miembro
Library	Library 1221	La biblioteca que contiene el DataType de este miembro
IsAttribute	boolean	True para atributos, falso para elementos
IsOptional	boolean	True si minOccurs = 0 o si es un atributo opcional
IsRequired	boolean	True si minOccurs > 0 o si es un atributo obligatorio
IsFixed	boolean	True para atributos fijos, el valor está en la propiedad Default
IsDefault	boolean	True para los atributos con valor predeterminado, el valor está en la propiedad Default
IsNillable	boolean	True para elementos nillable
IsUseQualified	boolean	True si la propiedad NamespaceURI no está vacía
MinOccurs	integer	minOccurs como en el esquema. 1 para atributos obligatorios
MaxOccurs	integer	maxOccurs como en el esquema. 0 para atributos prohibidos y -1 para unbounded
Default	string	Valor predeterminado

15.12.10.5 NativeBinding

Este objeto representa el enlace de un tipo simple con un tipo nativo en el lenguaje de programación de destino, tal y como indica la asignación "schemanativetype".

Propiedad	**Tipo**	**Descripción**
ValueType	string	Tipo nativo
ValueHandler	string	Instancia de clase de formateador

15.12.10.6 Facets

Este objeto representa todas las facetas de un tipo simple.
Las facetas heredadas se combinan con las facetas declaradas de forma explícita. Si hay una faceta Length, entonces MinLength y MaxLength tienen el mismo valor.

Propiedad	Tipo	Descripción
DeclaringType	Type	El tipo en el que están declaradas las facetas
Whitespace	string	"preserve", "collapse" o "replace"
MinLength	integer	Valor de faceta
MaxLength	integer	Valor de faceta
MinInclusive	integer	Valor de faceta
MinExclusive	integer	Valor de faceta
MaxInclusive	integer	Valor de faceta
MaxExclusive	integer	Valor de faceta
TotalDigits	integer	Valor de faceta
FractionDigits	integer	Valor de faceta
List	Colección Facet	Todas las facetas en forma de lista

Facet

Este objeto representa una sola faceta cuyo valor calculado está vigente en un tipo concreto.

Propiedad	Tipo	Descripción
LocalName	string	Nombre de la faceta
NamespaceURI	string	Espacio de nombres de la faceta
FacetType	string	Puede ser "normalization", "lexicalspace", "valuespace-length", "valuespace-enum" o "valuespace-range"
DeclaringType	Type 1221	El tipo en el que se declara esta faceta
FacetCheckerName	string	El nombre del Facet Checker (de la asignación schemafacet)
FacetValue	string o integer	El valor real de esta faceta

© 2016-2022 Altova GmbH

16 La API de MapForce

La API basada en COM de MapForce permite a los clientes acceder a las funciones de MapForce desde un código o una aplicación personalizados y automatizar un gran número de tareas.

La API COM de MapForce sigue las especificaciones generales para servidores de automatización definidas por Microsoft. MapForce se registra automáticamente como objeto de servidor COM durante su instalación. Una vez que el objeto de servidor COM está registrado, puede invocarlo desde dentro de aplicaciones y lenguajes de programación que admitan llamadas COM. Esto permite acceder a la API de MapForce no solo desde entornos de desarrollo mediante .NET, C++ y Visual Basic, sino también desde lenguajes como JScript y VBScript.

Nota: si usa la API de MapForce para crear una aplicación que quiere distribuir a otros clientes, MapForce debe estar instalado en cada uno de esos equipos cliente. Además deberá implementar su código personalizado de integración (o instalar su aplicación) en cada uno de esos equipos cliente.

16.1 Acceder a la API

Para acceder a la API COM de MapForce debe crear primero una nueva instancia del objeto `Application` en su aplicación (o script). Una vez haya creado este objeto puede interactuar con MapForce invocando sus métodos y propiedades cuando sea necesario (por ejemplo, al crear un documento nuevo, al abrir uno ya existente, al generar código de asignación, etc.).

Requisitos previos

Para que el objeto COM de MapForce esté disponible en su proyecto de Visual Studio, añada una referencia al archivo de la biblioteca de tipos de MapForce (.tlb). Las siguientes instrucciones se han redactado para Visual Studio 2013, pero no difieren demasiado de las de otras versiones de Visual Studio:

1. En el menú **Proyecto**, haga clic en **Agregar referencia**.
2. Haga clic en Examinar y seleccione el archivo **MapForce.tlb** dentro de la carpeta de instalación de MapForce.

Hay un cliente de muestra de la API de MapForce en C# en la siguiente ruta: **(Mis) Documentos\Altova\MapForce2023\MapForceExamples\API\C#**.

En Java se puede acceder a la API de MapForce con bibliotecas puente Java-COM. Estas bibliotecas se encuentran en la carpeta de instalación de MapForce: **C:\Program Files (x86) \Altova\MapForce2022\JavaAPI** (esta ruta es válida para una versión de 32 bits MapForce se esté ejecutando en una versión de Windows de 64 bits; de lo contrario hay que adaptar la ruta como corresponda).

- `AltovaAutomation.dll`: un envoltorio JNI para los servidores de automatización de Altova
- `AltovaAutomation.jar`: clases en Java para acceder a los servidores de automatización de Altova
- `MapForceAPI.jar`: clases en Java que envuelven la interfaz de automatización de MapForce
- `MapForceAPI_JavaDoc.zip`: un archivo `Javadoc` que contiene documentación de ayuda para la API Java

Para que se pueda acceder al servidor de automatización de MapForce directamente desde el código Java, las bibliotecas antes mencionadas deben estar en la variable de Java `classpath`.

Puede ver un ejemplo de un cliente de la API de MapForce en Java en la siguiente ruta: **(Mis) Documentos\Altova\MapForce2023\MapForceExamples\API\Java**.

En lenguajes de programación como JScript o VBScripts, se puede acceder al objeto COM de MapForce con el Microsoft Windows Script Host (*véase https://msdn.microsoft.com/en-us/library/9bbdkx3k.aspx*). Estos scripts se pueden escribir con un editor de texto y no necesitan ser compilados, ya que se ejecutan con el Windows Script Host incluido en Windows. (Para comprobar si el Windows Script Host se está ejecutando, teclee `wscript.exe /?` en la línea de comandos). Puede ver un ejemplo de un cliente de la API de MapForce en JScript en la siguiente ruta: **(My) Documents\Altova\MapForce2023\MapForceExamples\API\JScript**.

Nota: en la versión de 32 bits de MapForce, el nombre registrado o identificador programático (ProgId) del objeto COM es `MapForce.Application`. Para la versión de 64 bits de MapForce, el nombre es `MapForce_x64.Application`. Sin embargo, debe tener en cuenta que el programa que realiza las llamadas accede a las entradas de registro CLASSES de su propio subárbol o grupo (de 32 o 64 bits). Por tanto, si ejecuta scripts usando la línea de comandos estándar y el explorador de Windows en una instancia Windows de 64 bits, el programa accederá a las entradas de registro de 64 bits, que apuntan a la versión de 64 bits de MapForce. Por eso, si tiene instaladas tanto la versión de 32 bits como la

 © 2016-2022 Altova GmbH

versión de 64 bits de MapForce necesitará dar un rodeo para llamar a la versión de 32 bits. Por ejemplo, si el programa que realiza las llamadas es Windows Scripting Host, haga lo siguiente:

```
wscript.exe "C:\Users\...
\Documents\Altova\MapForce2023\MapForceExamples\API\JScript\start.js"
```

Instrucciones

Recomendamos que el código de su cliente observe las siguientes recomendaciones:

- No guarde en la memoria referencias a objetos más tiempo del que necesite. Si un usuario interactúa entre dos llamadas de su cliente, no se puede garantizar que esas referencias sigan siendo válidas.
- Tenga en cuenta que si ocurre un crash en el código de su cliente puede que sigan existiendo instancias de MapForce en el sistema. Para consultar más detalles acerca de cómo evitar mensajes de error, consulte Gestión de errores 1230.
- Libere referencias de forma explícita si está usando lenguajes como C++.

Crear el objeto Application

La sintaxis para crear el objeto de inicio `Application` depende del lenguaje de programación, como mostramos a continuación:

Visual Basic

```
Dim objMapForce As MapForceLib.Application = New MapForceLib.Application
```

VBA

```
' Create a new instance of MapForce.
Dim objMapForce As Application
Set objMapForce = CreateObject("MapForce.Application")
```

VBScript

```
' Access a running instance, or create a new instance of MapForce.
Set objMapForce = GetObject("MapForce.Application");
```

C#

```
// Create a new instance of MapForce via its automation interface.
MapForceLib.Application objMapForce = new MapForceLib.Application();
```

Java

```
// Start MapForce as COM server.
com.altova.automation.MapForce.Application objMapForce = new Application();
// COM servers start up invisible so we make it visible
objMapForce.setVisible(true);
```

JScript

```
// Access a running instance, or create a new instance of MapForce.
try
{
       objMapForce = WScript.GetObject ("", "MapForce.Application");
       // unhide application if it is a new instance
       objMapForce.Visible = true;
}
catch(err) { WScript.Echo ("Can't access or create MapForce.Application"); }
```


© 2016-2022 Altova GmbH

16.2 El modelo de objetos

El punto de partida para cualquier aplicación que use la API de MapForce es el objeto `Application`[1254]. Para llamar al resto de interfaces se usan otros objetos, aunque `Application` continúa siendo el punto de partida.

El modelo de objeto de la API de MapForce se puede representar como sigue (cada nivel de indentación indica una relación primario-secundario con el nivel directamente superior) :

```
Application[1254]
   Options[1370]
   Project[1380]
      ProjectItem[1396]
   Documents[1330]
      Document[1310]
         MapForceView[1350]
         Mapping[1357]
            Component[1288]
               Datapoint[1307]
            Components[1302]
            Connection[1305]
         Mappings[1367]
   ErrorMarkers[1339]
      ErrorMarker[1334]
AppOutputLines[1283]
   AppOutputLine[1272]
      AppOutputLines
         ...
      AppOutputLineSymbol[1285]
```

Para más información sobre cómo crear una instancia del objeto `Application`, consulte el apartado Acceder a la API[1226]. Para ver la referencia de los objetos expuestos por la API, consulte el apartado Referencia de los objetos[1254].

16.3 Gestión de errores

La API de MapForce devuelve errores de dos formas distintas. Todos los métodos de la API devuelven un elemento `HRESULT`. Este valor de retorno informa al emisor de la llamada de cualquier fallo durante la ejecución del método. Si la llamada se ha realizado con éxito, el valor de retorno es igual a `S_OK`. C/C++. Por lo general, los programadores suelen usar `HRESULT` para detectar errores.

Visual Basic, los lenguajes de programación y otros entornos de desarrollo de alto nivel no dan acceso al programador al `HRESULT` de retorno de una llamada COM, sino que usan el segundo mecanismo de detección de errores compatible con la API de MapForce, la interfaz `IErrorInfo`. Si ocurre un error, la API crea un objeto nuevo que implementa la interfaz `IErrorInfo`. El entorno de desarrollo usa esta interfaz para llenar su propio mecanismo de gestión de errores con la información obtenida.

A continuación describimos en detalle cómo gestionar los errores detectados por la API de MapForce en distintos entornos de desarrollo.

Visual Basic

Una forma común de gestionar errores en Visual Basic es definir un gestor de errores. Este se puede definir con la instrucción `Cuando haya erIErrorInfo.rores`. Normalmente el gestor muestra un mensaje de error y hace limpieza de código para evitar referencias duplicadas o cualquier tipo de recursos.Visual Basic rellena su propio objeto `Err` con la información que obtiene de la interfaz `IErrorInfo`.

```
Sub Validate()
  'place variable declarations here

  'set error handler
  On Error GoTo ErrorHandler

  'if generation fails, program execution continues at ErrorHandler:
  objMapForce.ActiveDocument.GenerateXSLT()

  'additional code comes here

  'exit
  Exit Sub

  ErrorHandler:
  MsgBox("Error: " & (Err.Number - vbObjectError) & Chr(13) &
     "Description: " & Err.Description)
End Sub
```

JavaScript

La implementación de Microsoft del JavaScript (JScript) ofrece un mecanismo try-catch para gestionar los errores derivados de las llamadas COM. Este mecanismo es muy parecido al enfoque VisualBasic en el sentido de que también se declara un objeto de error que contiene la información necesaria.

 © 2016-2022 Altova GmbH

```
function Generate()
{
   // please insert variable declarations here

   try
   {
      objMapForce.ActiveDocument.GenerateXSLT();
   }
   catch(Error)
   {
      sError = Error.description;
      nErrorCode = Error.number & 0xffff;
      return false;
   }

   return true;
}
```

C/C++

C/C++ ofrece un acceos fácil al `HRESULT` de la llamada COM y a la `IErrorInterface`.

```
HRESULT hr;

// Call GenerateXSLT() from the MapForce API
if(FAILED(hr = ipDocument->GenerateXSLT()))
{
   IErrorInfo *ipErrorInfo = Null;

   if(SUCCEEDED(::GetErrorInfo(0, &ipErrorInfo)))
   {
      BSTR   bstrDescr;
      ipErrorInfo->GetDescription(&bstrDescr);

      // handle Error information
      wprintf(L"Error message:\t%s\n",bstrDescr);
      ::SysFreeString(bstrDescr);

      // release Error info
      ipErrorInfo->Release();
   }
}
```

16.4 Ejemplo de proyecto en C#

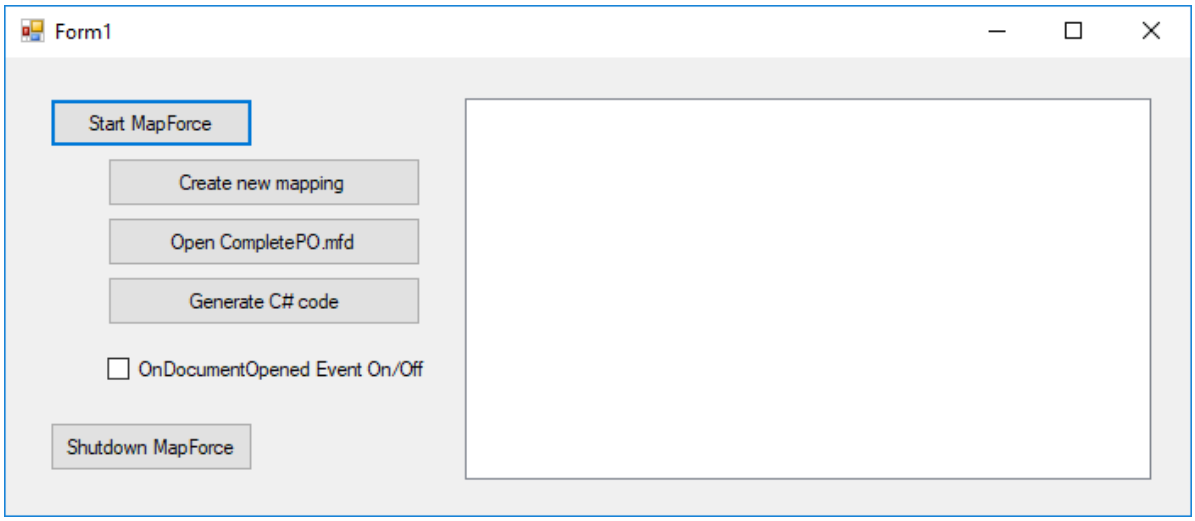
Una vez haya instalado MapForce podrá encontrar un ejemplo de proyecto cliente de la API de MapForce en C# en el directorio **C:/Users/<username>/Documents/Altova/MapForce2023/MapForceExamples/API**.

Puede compilar y ejecutar el proyecto con Visual Studio 2008 o superior. Para compilar y ejecutar el ejemplo, abra el archivo de la solución .sln en Visual Studio y ejecute **Depurar | Iniciar depuración** o **Depurar | Iniciar sin depurar**.

Nota: si tiene un sistema operativo de 64 bits y está usando una versión de MapForce de 32 bits, añada la plataforma **x86** al Administrador de configuración de la solución y compile el ejemplo usando esta configuración. Se puede crear una nueva plataforma x86 (para la solución activa de Visual Studio) en el cuadro de diálogo Nueva plataforma de solución (**Compilar | Administrador de configuración | Plataforma de soluciones activas | <Nueva...>**)

Cuando ejecute el ejemplo se mostrará un cuadro de diálogo de Windows que contiene botones que invocan operaciones básicas de MapForce:

- Start MapForce (iniciar MapForce)
- Create a new mapping design (Crear un nuevo diseño de asignación)
- Open the CompletePO.mfd (abrir el archivo CompletePO.mfd de la carpeta **...\MapForceExamples**). Recuerde que puede que tenga que ajustar la ruta para que apunte a la carpeta **\MapForceExamples** en su equipo.
- Generate C# code in a temp directory (Generar código C# en un directorio temporal)
- Shut down MapForce (cerrar MapForce)



Fragmento de código

El fragmento contiene comentarios para facilitar su comprensión. El código consiste básicamente en una serie de controladores para los botones de la interfaz de usuario que se muestra más arriba.

```
using System;
using System.Collections.Generic;
```

 © 2016-2022 Altova GmbH

```
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;

namespace WindowsFormsApplication2
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

        // An instance of MapForce accessed via its automation interface.
        MapForceLib.Application MapForce;

        // Location of examples installed with MapForce
        String strExamplesFolder;

        private void Form1_Load(object sender, EventArgs e)
        {
        }

        // handler for the "Start MapForce" button
        private void StartMapForce_Click(object sender, EventArgs e)
        {
            if (MapForce == null)
            {
                Cursor.Current = Cursors.WaitCursor;

                // if we have no MapForce instance, we create one a nd make it visible.
                MapForce = new MapForceLib.Application();
                MapForce.Visible = true;

                // locate examples installed with MapForce.
                int majorVersionYear = MapForce.MajorVersion + 1998;
                strExamplesFolder = Environment.GetEnvironmentVariable("USERPROFILE") +
"\\My Documents\\Altova\\MapForce" + Convert.ToString(majorVersionYear) + "\
\MapForceExamples\\";

                Cursor.Current = Cursors.Default;
            }
            else
            {
                // if we have already an MapForce instance running we toggle its
visibility flag.
                MapForce.Visible = !MapForce.Visible;
            }
        }

        // handler for the "Open CompletePO.mfd" button
        private void openCompletePO_Click(object sender, EventArgs e)
        {
            if (MapForce == null)
                StartMapForce_Click(null, null);
```

```
            // Open one of the sample files installed with the product.
            MapForce.OpenDocument(strExamplesFolder + "CompletePO.mfd");
        }

        // handler for the "Create new mapping" button
        private void newMapping_Click(object sender, EventArgs e)
        {
            if (MapForce == null)
                StartMapForce_Click(null, null);

            // Create a new mapping
            MapForce.NewMapping();
        }

        // handler for the "Shutdown MapForce" button
        // shut-down application instance by explicitly releasing the COM object.
        private void shutdownMapForce_Click(object sender, EventArgs e)
        {
            if (MapForce != null)
            {
                // allow shut-down of MapForce by releasing UI
                MapForce.Visible = false;

                // explicitly release COM object
                try
                {
                    while
(System.Runtime.InteropServices.Marshal.ReleaseComObject(MapForce) > 0) ;
                }
                finally
                {
                    // avoid later access to this object.
                    MapForce = null;
                }
            }
        }

        // handler for button "Generate C# Code"
        private void generateCppCode_Click(object sender, EventArgs e)
        {
            if (MapForce == null)
                listBoxMessages.Items.Add("start MapForce first.");
            // COM errors get returned to C# as exceptions. We use a try/catch block to
handle them.
            try
            {
                MapForceLib.Document doc = MapForce.ActiveDocument;

                listBoxMessages.Items.Add("Active document " + doc.Name);
                doc.GenerateCHashCode();

            }
            catch (Exception ex)
            {
                // The COM call was not successful.
                // Probably no application instance has been started or no document is
open.
                MessageBox.Show("COM error: " + ex.Message);
            }
```

```
        }

        delegate void addListBoxItem_delegate(string sText);
        // called from the UI thread
        private void addListBoxItem(string sText)
        {
            listBoxMessages.Items.Add(sText);
        }
        // wrapper method to allow to call UI controls methods from a worker thread
        void syncWithUIthread(Control ctrl, addListBoxItem_delegate methodToInvoke,
String sText)
        {
            // Control.Invoke: Executes on the UI thread, but calling thread waits for
completion before continuing.
            // Control.BeginInvoke: Executes on the UI thread, and calling thread doesn't
wait for completion.
            if (ctrl.InvokeRequired)
                ctrl.BeginInvoke(methodToInvoke, new Object[] { sText });
        }

        // event handler for OnDocumentOpened event
        private void handleOnDocumentOpened(MapForceLib.Document i_ipDocument)
        {
            String sText = "";

            if (i_ipDocument.Name.Length > 0)
                sText = "Document " + i_ipDocument.Name + " was opened!";
            else
                sText = "A new mapping was created.";

            // we need to synchronize the calling thread with the UI thread because
            // the COM events are triggered from a working thread
            addListBoxItem_delegate methodToInvoke = new
addListBoxItem_delegate(addListBoxItem);
            // call syncWithUIthread with the following arguments:
            // 1 - listBoxMessages - list box control to display messages from COM events
            // 2 - methodToInvoke  - a C# delegate which points to the method which will
be called from the UI thread
            // 3 - sText           - the text to be displayed in the list box
            syncWithUIthread(listBoxMessages, methodToInvoke, sText);
        }

        private void checkBoxEventOnOff_CheckedChanged(object sender, EventArgs e)
        {
            if (MapForce != null)
            {
                if (checkBoxEventOnOff.Checked)
                    MapForce.OnDocumentOpened += new
MapForceLib._IApplicationEvents_OnDocumentOpenedEventHandler(handleOnDocumentOpened);
                else
                    MapForce.OnDocumentOpened -= new
MapForceLib._IApplicationEvents_OnDocumentOpenedEventHandler(handleOnDocumentOpened);
            }
        }
    }
}
```

16.5 Ejemplo de proyecto en Java

Una vez haya instalado MapForce podrá encontrar un ejemplo de proyecto cliente de la API de MapForce en Java en el directorio **C:/Users/<username>/Documents/Altova/MapForce2023/MapForceExamples/API**.

Puede probar el ejemplo en Java directamente desde la línea de comandos usando el archivo `BuildAndRun.bat` o puede compilar y ejecutar el proyecto de ejemplo desde Eclipse. Ambas opciones vienen explicadas a continuación.

Lista de archivos

La carpeta de ejemplos en Java contiene todos los archivos necesarios para ejecutar el proyecto de ejemplo. Estos archivos son los siguientes:

`AltovaAutomation.dll`	puente Java-COM bridge: parte DLL
`AltovaAutomation.jar`	puente Java-COM: parte de biblioteca Java
`MapForceAPI.jar`	clases Java de la API de MapForce
`RunMapForce.java`	código fuente del ejemplo en Java
`BuildAndRun.bat`	fichero batch para compilar y ejecutar el código del ejemplo desde la línea de comandos. Se necesita una carpeta en la que la Java Virtual Machine esté como parámetro.
`.classpath`	archivo de ayuda del proyecto de Eclipse
`.project`	archivo del proyecto de Eclipse
`MapForceAPI_JavaDoc.zip`	archivo `Javadoc` que contiene la documentación de ayuda para la Java API

¿Qué hace el ejemplo?

El ejemplo inicia MapForce y realiza algunas operaciones como abrir y cerrar documentos. Al finalizar, MapForce sigue abierto y debe cerrarlo de forma manual.

Ejecutar el ejemplo desde la línea de comandos

Para ejecutar el ejemplo desde la línea de comandos, abra la ventana del símbolo del sistema, vaya a la carpeta de Java de los ejemplos de la API (*encontrará más arriba la ubicación*) y teclee:

```
buildAndRun.bat "<Path-to-the-Java-bin-folder>"
```

La carpeta binaria de Java debe ser la del Java Development Kit (JDK) 7 o superior.

Pulse la tecla de retorno. La fuente en Java en `RunMapForce.java` se compilará y después se ejecutará.

Cargar el ejemplo en Eclipse

Abra Eclipse y use el comando Archivo **|** Importar... **|** General **|** Proyectos existentes en el espacio de trabajo para añadir el archivo del proyecto de Eclipse (`.project`) ubicado en la carpeta de Java de la carpeta de ejemplos de la API (*encontrará más arriba su ubicación*). El proyecto RunMapForce aparecerá en su explorador de paquetes o explorador.

Seleccione el proyecto y después el comando **Ejecutar como | Aplicación en Java** para ejecutar el ejemplo.

 © 2016-2022 Altova GmbH

Nota: puede seleccionar un nombre o un método de clase de la Java API y pulsar F1 para obtener ayuda para esa clase o ese método.

Fragmento de código fuente en Java

A continuación se muestra el código fuente en Java del archivo de ejemplo `RunMapForce.java` con comentarios.

```
// access general JAVA-COM bridge classes
import java.util.Iterator;

import com.altova.automation.libs.*;

// access XMLSpy Java-COM bridge
import com.altova.automation.MapForce.*;
import com.altova.automation.MapForce.Enums.ENUMProgrammingLanguage;

/**
 * A simple example that starts XMLSpy COM server and performs a few operations on it.
 * Feel free to extend.
 */
public class RunMapForce
{
   public static void main(String[] args)
   {
      // an instance of the application.
      Application mapforce = null;

      // instead of COM error handling use Java exception mechanism.
      try
      {
         // Start MapForce as COM server.
         mapforce = new Application();
         // COM servers start up invisible so we make it visible
         mapforce.setVisible(true);

         // The following lines attach to the application events using a default
implementation
         // for the events and override one of its methods.
         // If you want to override all document events it is better to derive your
listener class
         // from DocumentEvents and implement all methods of this interface.
         mapforce.addListener(new ApplicationEventsDefaultHandler()
         {
            @Override
            public void onDocumentOpened(Document i_ipDoc) throws AutomationException
            {
               String name = i_ipDoc.getName();

               if (name.length() > 0)
                  System.out.println("Document " + name + " was opened.");
               else
```

```
                    System.out.println("A new mapping was created.");
                }
            });

            // Locate samples installed with the product.
            int majorVersionYear = mapforce.getMajorVersion() + 1998;
            String strExamplesFolder = System.getenv("USERPROFILE") + "\\Documents\\Altova\
\MapForce" + Integer.toString(majorVersionYear) + "\\MapForceExamples\\";
            // create a new MapForce mapping and generate c++ code
            Document newDoc = mapforce.newMapping();
            ErrorMarkers err1 = newDoc.generateCodeEx(ENUMProgrammingLanguage.eCpp);
            display(err1);
            // open CompletePO.mfd and generate c++ code
            Document doc = mapforce.openDocument(strExamplesFolder + "CompletePO.mfd");
            ErrorMarkers err2 = doc.generateCodeEx(ENUMProgrammingLanguage.eCpp);
            display(err2);

            doc.close();
            doc = null;

            System.out.println("Watch MapForce!");
        }
        catch (AutomationException e)
        {
            // e.printStackTrace();
        }
        finally
        {
            // Make sure that MapForce can shut down properly.
            if (mapforce != null)
                mapforce.dispose();

            // Since the COM server was made visible and still is visible, it will keep
running
            // and needs to be closed manually.
            System.out.println("Now close MapForce!");
        }
    }

    public static void display(ErrorMarkers err) throws AutomationException
    {
        Iterator<ErrorMarker> itr = err.iterator();

        if (err.getCount() == 0)
            System.out.print("Code generation completed successfully.\n");

        while (itr.hasNext())
        {
            String sError = "";
            Object element = itr.next();
            if (element instanceof ErrorMarker)
                sError = ((ErrorMarker)element).getText();
            System.out.print("Error text: " + sError + "\n");
        }
```


© 2016-2022 Altova GmbH

```
    }
}
```

16.6 Ejemplos en JScript

Una vez haya instalado MapForce podrá encontrar un ejemplo de proyecto cliente de la API de MapForce para JScript en el directorio **C:/Users/<username>/Documents/Altova/MapForce2023/MapForceExamples/API**.

Los archivos de ejemplo se pueden ejecutar de dos maneras:

- *Desde la línea de comandos:*
  Abra la ventana del símbolo del sistema y teclee el nombre de uno de los scripts de ejemplo (por ejemplo, `Start.js`). El Windows Scripting Host que viene con Windows ejecutará el script.

- *Desde el explorador de Windows:*
  En el explorador de Windows, navegue hasta el archivo JScript y haga doble clic en él. El Windows Scripting Host que viene con Windows ejecutará el script. Una vez se haya ejecutado el script, la consola se cierra automáticamente.

Están incluidos los siguientes archivos:

`Start.js`	Inicia MapForce registrado como servidor de automatización o se conecta a una instancia en ejecución (*véase* *Iniciar la aplicación* 1240).
`DocumentAccess.js`	Muestra cómo abrir, iterar y cerrar documentos (*véase* *Acceso a documentos simples* 1241).
`GenerateCode.js`	Muestra cómo invocar la generación de código usando JScript (*véase* *Generar código* 1242).
`Readme.txtOfrece`	Ofrece ayuda básica para ejecutar los scripts.

Esta documentación también incluye algunos extractos de código JScript:

- Ejemplo: generación de código 1244
- Ejemplo: ejecución de asignaciones 1246
- Ejemplo: ayuda con el proyecto 1249

16.6.1 Iniciar la aplicación

El siguiente fragmento de código JScript inicia y cierra la aplicación. Si ya se está ejecutando una instancia de la aplicación, se llama a esa aplicación.

Nota: en la versión de 32 bits de MapForce el nombre registrado o identificador programático (ProgId) del objeto COM es `MapForce.Application`. Para la versión de 64 bits de MapForce el nombre es `MapForce_x64.Application`.

```
// Initialize application's COM object. This will start a new instance of the application
and
// return its main COM object. Depending on COM settings, a the main COM object of an
already
// running application might be returned.
```

```
try {   objMapForce = WScript.GetObject("", "MapForce.Application");   }
catch(err) {}

if( typeof( objMapForce ) == "undefined" )
{
   try   {   objMapForce = WScript.GetObject("", "MapForce_x64.Application")   }
   catch(err)
   {
      WScript.Echo( "Can't access or create MapForce.Application" );
      WScript.Quit();
   }
}

// if newly started, the application will start without its UI visible. Set it to
visible.
objMapForce.Visible = true;

WScript.Echo(objMapForce.Edition + " has successfully started. ");

objMapForce.Visible = false; // will shutdown application if it has no more COM
connections
//objMapForce.Visible = true;   // will keep application running with UI visible
```

El fragmento de código anterior está disponible como archivo simple (*véase* *JScript Examples* 1240). Para ejecutar el script, inícielo desde la línea de comandos o desde el explorador de Windows.

16.6.2 Acceso a documentos simples

El siguiente fragmento de código muestra cómo abrir documentos, convertir un documento en el documento activo, iterar los documentos abiertos y cerrar documentos.

```
// Initialize application's COM object. This will start a new instance of the application
and
// return its main COM object. Depending on COM settings, a the main COM object of an
already
// running application might be returned.
try {   objMapForce = WScript.GetObject("", "MapForce.Application");   }
catch(err) {}

if( typeof( objMapForce ) == "undefined" )
{
   try   {   objMapForce = WScript.GetObject("", "MapForce_x64.Application")   }
   catch(err)
   {
      WScript.Echo( "Can't access or create MapForce.Application" );
      WScript.Quit();
   }
}

// if newly started, the application will start without its UI visible. Set it to
visible.
```

```
objMapForce.Visible = true;

// ***************************** code snippet for "Simple Document Access"
************************

// Locate examples via USERPROFILE shell variable. The path needs to be adapted to major
release versions.
objWshShell = WScript.CreateObject("WScript.Shell");
majorVersionYear = objMapForce.MajorVersion + 1998
strExampleFolder = objWshShell.ExpandEnvironmentStrings("%USERPROFILE%") + "\\Documents\
\Altova\\MapForce" + majorVersionYear + "\\MapForceExamples\\";

objMapForce.Documents.OpenDocument(strExampleFolder + "CompletePO.mfd");
objMapForce.Documents.OpenDocument(strExampleFolder + "Altova_Hierarchical_DB.mfd");

// ***************************** code snippet for "Simple Document Access"
************************

// ***************************** code snippet for "Iteration"
************************************

// go through all open documents using a JScript Enumerator
for (var iterDocs = new Enumerator(objMapForce.Documents); !iterDocs.atEnd();
iterDocs.moveNext())
{
   objName = iterDocs.item().Name;
   WScript.Echo("Document name: " + objName);
}

// go through all open documents using index-based access to the document collection
for (i = objMapForce.Documents.Count; i > 0; i--)
   objMapForce.Documents.Item(i).Close();


// ***************************** code snippet for "Iteration"
************************************

//objMapForce.Visible = false;      // will shutdown application if it has no more COM
connections
objMapForce.Visible = true;   // will keep application running with UI visible
```

El fragmento de código anterior está disponible como archivo de ejemplo (*véase* *Ejemplos en JScript* 1240). Para ejecutar el script, inícielo desde la línea de comandos o desde el explorador de Windows.

16.6.3 Generar código

El siguiente fragmento de código en JScript muestra cómo abrir documentos, convertir un documento en el documento activo, iterar los documentos abiertos y cerrar documentos.

```
// Initialize application's COM object. This will start a new instance of the application
and
```

 © 2016-2022 Altova GmbH

```
// return its main COM object. Depending on COM settings, a the main COM object of an
already
// running application might be returned.
try {   objMapForce = WScript.GetObject("", "MapForce.Application");   }
catch(err) {}

if( typeof( objMapForce ) == "undefined" )
{
   try   {   objMapForce = WScript.GetObject("", "MapForce_x64.Application")   }
   catch(err)
   {
      WScript.Echo( "Can't access or create MapForce.Application" );
      WScript.Quit();
   }
}

// if newly started, the application will start without its UI visible. Set it to
visible.
objMapForce.Visible = true;

// **************************** code snippet for "Simple Document Access"
*************************

// Locate examples via USERPROFILE shell variable. The path needs to be adapted to major
release versions.
objWshShell = WScript.CreateObject("WScript.Shell");
majorVersionYear = objMapForce.MajorVersion + 1998
strExampleFolder = objWshShell.ExpandEnvironmentStrings("%USERPROFILE%") + "\\Documents\
\Altova\\MapForce" + majorVersionYear + "\\MapForceExamples\\";

objMapForce.Documents.OpenDocument(strExampleFolder + "CompletePO.mfd");
//objMapForce.Documents.OpenDocument(strExampleFolder + "Altova_Hierarchical_DB.mfd");
objMapForce.Documents.NewDocument();

// **************************** code snippet for "Simple Document Access"
*************************

// **************************** code snippet for "Iteration"
************************************

objText = "";
// go through all open documents using a JScript Enumerator and generate c++ code
for (var iterDocs = new Enumerator(objMapForce.Documents); !iterDocs.atEnd();
iterDocs.moveNext())
{
   objText += "Generated c++ code result for document " + iterDocs.item().Name + " :\n";
   objErrorMarkers = iterDocs.item().generateCodeEx(1); // ENUMProgrammingLanguage.eCpp =
1

   bSuccess = true;
   for (var iterErrorMarkers = new
Enumerator(objErrorMarkers); !iterErrorMarkers.atEnd(); iterErrorMarkers.moveNext())
   {
      bSuccess = false;
      objText += "\t" + iterErrorMarkers.item().Text + "\n";
```

```
   }

   if (bSuccess)
       objText += "\tCode generation completed successfully.\n";

   objText += "\n";
}

WScript.Echo(objText);

// go through all open documents using index-based access to the document collection
for (i = objMapForce.Documents.Count; i > 0; i--)
   objMapForce.Documents.Item(i).Close();


// **************************** code snippet for "Iteration"
************************************

//objMapForce.Visible = false;      // will shutdown application if it has no more COM
connections
objMapForce.Visible = true;   // will keep application running with UI visible
```

El fragmento de código anterior está disponible como archivo de ejemplo (*véase* *Ejemplos en JScript* 1240). Para ejecutar el script, inícielo desde la línea de comandos o desde el explorador de Windows.

16.6.4 Ejemplo: generación de código

El siguiente fragmento de código en JScript muestra cómo cargar un documento ya existente y generar distintos topos de código de asignación para él.

```
// ------------------- begin JScript example ---------------------
// Generate Code for existing mapping.
// works with Windows scripting host.

// ----------------- helper function ------------------
function Exit(strErrorText)
{
   WScript.Echo(strErrorText);
   WScript.Quit(-1);
}

function ERROR(strText, objErr)
{
   if (objErr != null)
      Exit ("ERROR: (" + (objErr.number & 0xffff) + ")" + objErr.description + " - " +
strText);
   else
      Exit ("ERROR: " + strText);
}
// ---------------------------------------------------

// ----------------- MAIN ------------------
```

```
// ----- create the Shell and FileSystemObject of the windows scripting
try
{
   objWshShell = WScript.CreateObject("WScript.Shell");
   objFSO = WScript.CreateObject("Scripting.FileSystemObject");
}
catch(err)
   { Exit("Can't create WScript.Shell object"); }

// ----- open MapForce or access running instance and make it visible
try
{
   objMapForce = WScript.GetObject ("", "MapForce.Application");
   objMapForce.Visible = true;      // remove this line to perform background processing
}
catch(err) { WScript.Echo ("Can't access or create MapForce.Application"); }

// ----- open an existing mapping. adapt this to your needs!
objMapForce.OpenDocument(objFSO.GetAbsolutePathName ("Test.mfd"));

// ----- access the mapping to have access to the code generation methods
var objDoc = objMapForce.ActiveDocument;

// ----- set the code generation output properties and call the code generation methods.
// ----- adapt the output directories to your needs
try
{
   // ----- code generation uses some of these options
   var objOptions = objMapForce.Options;

   // ----- generate XSLT -----
   objOptions.XSLTDefaultOutputDirectory = "C:\\test\\TestCOMServer\\XSLT";
   objDoc.GenerateXSLT();

   // ----- generate Java Code -----
   objOptions.CodeDefaultOutputDirectory = "C:\\test\\TestCOMServer\\Java";
   objDoc.GenerateJavaCode();

   // ----- generate CPP Code, use same cpp code options as the last time -----
   objOptions.CodeDefaultOutputDirectory = "C:\\test\\TestCOMServer\\CPP";
   objDoc.GenerateCppCode();

   // ----- generate C# Code, use options C# code options as the last time -----
   objOptions.CodeDefaultOutputDirectory = "C:\\test\\TestCOMServer\\CHash";
   objDoc.GenerateCHashCode();
}
catch (err)
   { ERROR ("while generating XSL or program code", err); }


// hide MapForce to allow it to shut down
objMapForce.Visible = false;

// -------------------- end example ---------------------
```

16.6.5 Ejemplo: ejecución de asignaciones

El siguiente ejemplo en JScript muestra cómo cargar un documento ya existente con una asignación simple, acceder a sus componentes, establecer nombres de archivo de instancia entrada y de salida, y ejecutar la asignación.

```
/*
   This sample file performs the following operations:

   Load existing MapForce mapping document.
   Find source and target component.
   Set input and output instance filenames.
   Execute the transformation.

   Works with Windows scripting host.
*/

// ---- general helpers ------------------------------

function Exit( message )
{
   WScript.Echo( message );
   WScript.Quit(-1);
}

function ERROR( message, err )
{
   if( err != null )
      Exit( "ERROR: (" + (err.number & 0xffff) + ") " + err.description + " - " + message
);
   else
      Exit( "ERROR: " + message );
}



// ---- MapForce constants -----------------------

var eComponentUsageKind_Unknown     = 0;
var eComponentUsageKind_Instance    = 1;
var eComponentUsageKind_Input       = 2;
var eComponentUsageKind_Output      = 3;



// ---- MapForce helpers -----------------------

// Searches in the specified mapping for a component by name and returns it.
// If not found, throws an error.
function FindComponent( mapping, component_name )
{
   var components = mapping.Components;
   for( var i = 0 ; i < components.Count ; ++i )
```

 © 2016-2022 Altova GmbH

```
   {
      var component = components.Item( i + 1 );
      if( component.Name == component_name )
         return component;
   }
   throw new Error( "Cannot find component with name " + component_name );
}



// Browses components in a mapping and returns the first one found acting as
// source component (i.e. having connections on its right side).
function GetFirstSourceComponent( mapping )
{
   var components = mapping.Components;
   for( var i = 0 ; i < components.Count ; ++i )
   {
      var component = components.Item( i + 1 );
      if( component.UsageKind == eComponentUsageKind_Instance &&
         component.HasOutgoingConnections )
      {
         return component;
      }
   }
   throw new Error( "Cannot find a source component" );
}

// Browses components in a mapping and returns the first one found acting as
// target component (i.e. having connections on its left side).
function GetFirstTargetComponent( mapping )
{
   var components = mapping.Components;
   for( var i = 0 ; i < components.Count ; ++i )
   {
      var component = components.Item( i + 1 );
      if( component.UsageKind == eComponentUsageKind_Instance &&
         component.HasIncomingConnections )
      {
         return component;
      }
   }
   throw new Error( "Cannot find a target component" );
}



function IndentTextLines( s )
{
   return "\t" + s.replace( /\n/g, "\n\t" );
}

function GetAppoutputLineFullText( oAppoutputLine )
{
   var s = oAppoutputLine.GetLineText();
```

```
   var oAppoutputChildLines = oAppoutputLine.ChildLines;
   var i;

   for( i = 0 ; i < oAppoutputChildLines.Count ; ++i )
   {
      oAppoutputChildLine = oAppoutputChildLines.Item( i + 1 );
      sChilds = GetAppoutputLineFullText( oAppoutputChildLine );
      s += "\n" + IndentTextLines( sChilds );
   }

   return s;
}

// Create a nicely formatted string from AppOutputLines
function GetResultMessagesString( oAppoutputLines )
{
   var s1 = "Transformation result messages:\n";
   var oAppoutputLine;
   var i;

   for( i = 0 ; i < oAppoutputLines.Count ; ++i )
   {
      oAppoutputLine = oAppoutputLines.Item( i + 1 );
      s1 += GetAppoutputLineFullText( oAppoutputLine );
      s1 += "\n";
   }

   return s1;
}



// ---- MAIN -------------------------------------

var wshShell;
var fso;
var mapforce;

// create the Shell and FileSystemObject of the windows scripting system
try
{
   wshShell = WScript.CreateObject( "WScript.Shell" );
   fso = WScript.CreateObject( "Scripting.FileSystemObject" );
}
catch( err )
   { ERROR( "Can't create windows scripting objects", err ); }

// open MapForce or access currently running instance
try
{
   mapforce = WScript.GetObject( "", "MapForce.Application" );
}
catch( err )
   { ERROR( "Can't access or create MapForce.Application", err ); }
```


© 2016-2022 Altova GmbH

```
try
{
   // Make MapForce UI visible. This is an API requirement for output generation.
  mapforce.Visible = true;

   // open an existing mapping.
   // **** adjust the examples path to your needs ! **************
   var sMapForceExamplesPath = fso.BuildPath(
            wshShell.SpecialFolders( "MyDocuments" ),
            "Altova\\MapForce2023\\MapForceExamples" );
   var sDocFilename = fso.BuildPath( sMapForceExamplesPath, "PersonList.mfd" );
   var doc = mapforce.OpenDocument( sDocFilename );

   // Find existing components by name in the main mapping.
   // Note, the names of components may not be unique as a schema component's name
   // is derived from its schema file name.
   var source_component = FindComponent( doc.MainMapping, "Employees" );
   var target_component = FindComponent( doc.MainMapping, "PersonList" );
   // If you do not know the names of the components for some reason, you could
   // use the following functions instead of FindComponent.
   //var source_component = GetFirstSourceComponent( doc.MainMapping );
   //var target_component = GetFirstTargetComponent( doc.MainMapping );

   // specify the desired input and output files.
   source_component.InputInstanceFile = fso.BuildPath( sMapForceExamplesPath,
"Employees.xml" );
   target_component.OutputInstanceFile = fso.BuildPath( sMapForceExamplesPath,
"test_transformation_results.xml" );

   // Perform the transformation.
   // You can use doc.GenerateOutput() if you do not need result messages.
   // If you have a mapping with more than one target component and you want
   // to execute the transformation only for one specific target component,
   // call target_component.GenerateOutput() instead.
   var result_messages = doc.GenerateOutputEx();

   var summary_info =
         "Transformation performed from " + source_component.InputInstanceFile + "\n" +
         "to " + target_component.OutputInstanceFile + "\n\n" +
         GetResultMessagesString( result_messages );
   WScript.Echo( summary_info );
}
catch( err )
{
   ERROR( "Failure", err );
}
```

16.6.6 Ejemplo: ayuda con el proyecto

El siguiente ejemplo en JScript muestra cómo usar la API de MapForce para automatizar tareas pertenecientes a proyectos de MapForce. Antes de ejecutar el ejemplo, asegúrese de que edita la variable

`strSamplePath` para que apunte a la carpeta **MapForceExamples** de su instalación de MapForce: **`C:\Usuarios\<usuario>\Documentos\Altova\MapForce2023\MapForceExamples`**.

Para ejecutar con éxito todas las operaciones del siguiente ejemplo necesitará la versión Enterprise de MapForce. Si está ejecutando la versión Professional, bloquee con comentarios las líneas que insertan el proyecto de servicio web.

```
// ///////////// global variables /////////////////
var objMapForce = null;
var objWshShell = null;
var objFSO = null;

// !!! adapt the following path to your needs. !!!
var strSamplePath = "C:\\Users\\<username>\\Documents\\Altova\\MapForce2023\
\MapForceExamples\\";

// ////////////////////////// Helpers ////////////////////////////////

function Exit(strErrorText)
{
   WScript.Echo(strErrorText);
   WScript.Quit(-1);
}

function ERROR(strText, objErr)
{
   if (objErr != null)
      Exit ("ERROR: (" + (objErr.number & 0xffff) + ")" + objErr.description + " - " +
strText);
   else
      Exit ("ERROR: " + strText);
}

function CreateGlobalObjects ()
{
   // the Shell and FileSystemObject of the windows scripting host often useful
   try
   {
      objWshShell = WScript.CreateObject("WScript.Shell");
      objFSO = WScript.CreateObject("Scripting.FileSystemObject");
   }
   catch(err)
      { Exit("Can't create WScript.Shell object"); }

   // create the MapForce connection
   // if there is a running instance of MapForce (that never had a connection) - use it
   // otherwise, we automatically create a new instance
   try
   {
      objMapForce = WScript.GetObject("", "MapForce.Application");
   }
   catch(err)
   {
      { Exit("Can't access or create MapForce.Application"); }
```

 © 2016-2022 Altova GmbH

```
   }
}

// ------------------------------------------------------------
// print project tree items and their properties recursively.
// ------------------------------------------------------------
function PrintProjectTree( objProjectItemIter, strTab )
{
   while ( ! objProjectItemIter.atEnd() )
   {
      // get current project item
      objItem = objProjectItemIter.item();

      try
      {
         // ----- print common properties
         strGlobalText += strTab + "[" + objItem.Kind + "]" + objItem.Name + "\n";

         // ----- print code generation properties, if available
         try
         {
            if ( objItem.CodeGenSettings_UseDefault )
               strGlobalText += strTab + "  Use default code generation settings\n";
            else
               strGlobalText += strTab + "  code generation language is " +
                                                   objItem.CodeGenSettings_Language +
                                 " output folder is " +
objItem.CodeGenSettings_OutputFolder + "\n";
         }
         catch( err ) {}

         // ----- print WSDL settings, if available
         try
         {
            strGlobalText += strTab + "  WSDL File is " + objItem.WSDLFile +
                              " Qualified Name is " + objItem.QualifiedName + "\n";
         }
         catch( err ) {}
      }
      catch( ex )
         { strGlobalText += strTab + "[" + objItem.Kind + "]\n" }

      // ---- recurse
      PrintProjectTree( new Enumerator( objItem ), strTab + '  ' );

      objProjectItemIter.moveNext();
   }
}

// ------------------------------------------------------------
// Load example project installed with MapForce.
// ------------------------------------------------------------
function LoadSampleProject()
{
```

```
   // close open project
   objProject = objMapForce.ActiveProject;
   if ( objProject != null )
      objProject.Close();

   // open sample project and iterate through it.
   objProject = objMapForce.OpenProject(strSamplePath + "MapForceExamples.mfp");
   // dump properties of all project items
   strGlobalText = '';
   PrintProjectTree( new Enumerator (objProject), ' ' )
   WScript.Echo( strGlobalText );

   objProject.Close();
}

// ----------------------------------------------------------
// Create a new project with some folders, mappings and a
// Web service project.
// ----------------------------------------------------------
function CreateNewProject()
{
   try
   {
      // create new project and specify file to store it.
      objProject = objMapForce.NewProject(strSamplePath + "Sample.mfp");

      // create a simple folder structure
      objProject.CreateFolder( "New Folder 1");
      objFolder1 = objProject.Item(0);
      objFolder1.CreateFolder( "New Folder 2");
      objFolder2 = ( new Enumerator( objFolder1 ) ).item();   // an alternative to
Item(0)

      // add two different mappings to folder structure
      objFolder1.AddFile( strSamplePath + "DB_Altova_SQLXML.mfd");
      objMapForce.Documents.OpenDocument(strSamplePath + "InspectionReport.mfd");
      objFolder2.AddActiveFile();

      // override code generation settings for this folder
      objFolder2.CodeGenSettings_UseDefault = false;
      objFolder2.CodeGenSettings_OutputFolder = strSamplePath + "SampleOutput"
      objFolder2.CodeGenSettings_Language = 1;      //C++

      // insert Web service project based on a wsdl file from the installed examples
      objProject.InsertWebService( strSamplePath + "TimeService/TimeService.wsdl",
                                   "{http://www.Nanonull.com/TimeService/}TimeService",
                           "TimeServiceSoap",
                           true );
      objProject.Save();
      if ( ! objProject.Saved )
         WScript.Echo("problem occurred when saving project");

      // dump project tree
      strGlobalText = '';
      PrintProjectTree( new Enumerator (objProject), ' ' )
```

 © 2016-2022 Altova GmbH

```
      WScript.Echo( strGlobalText );
   }
   catch (err)
   { ERROR("while creating new project", err ); }
}

// ---------------------------------------------------------
// Generate code for a project's sub-tree. Mix default code
// generation parameters and overloaded parameters.
// ---------------------------------------------------------
function GenerateCodeForNewProject()
{
   // since the Web service project contains only initial mappings,
   // we generate code only for our custom folder.
   // code generation parameters from project are used for Folder1,
   // whereas Folder2 provides overwritten values.
   objFolder = objProject.Item(0);
   objFolder1.GenerateCode();
}

// ////////////////////////// MAIN ////////////////////////////////

CreateGlobalObjects();
objMapForce.Visible = true;

LoadSampleProject();
CreateNewProject();
GenerateCodeForNewProject();

// uncomment to shut down application when script ends
// objMapForce.Visible = false;
```

16.7 Referencia de objetos

Esta sección sirve de referencia para los objetos de la API de MapForce Los objetos se describen de forma genérica, ya que la API se puede usar con prácticamente cualquier lenguaje que admita llamadas a objetos COM. Para ver ejemplos de lenguajes específicos, consulte.

- Ejemplo de proyecto en C# 1232
- Ejemplo de proyecto en Java 1236
- Ejemplos en JScript 1240

La referencia de la API contiene dos secciones principales que describen, respectivamente, la interfaz y los tipos de enumeración que se usan en la API. Los valores de enumeración contienen un nombre de cadena y un valor numérico. Si su entorno de programación no admite enumeraciones, use los valores numéricos.

En NET, por cada interfaz de la interfaz de automatización COM de MapForce existe una clase NET del mismo nombre. Asimismo, los tipos COM se convierten en los tipos NET correspondientes. Por ejemplo, un tipo como `Long` en la API COM aparecería como `System.Int32` en NET:

En Java se dan algunas variaciones de sintaxis:

- **Clases y nombres de clases**. Por cada interfaz de la interfaz de automatización de MapForce existe una clase en Java con el nombre de la interfaz.
- **Nombres de métodos**. Los nombres de métodos de la interfaz en Java son los mismos que los de las interfaces COM pero empiezan con minúscula para ajustarse a las convenciones de nomenclatura en Java. Para acceder a las propiedades COM se pueden usar métodos en Java que añaden a la propiedad los prefijos get y set. Si una propiedad no admite el acceso de escritura, entonces no se puede usar ningún método de acceso o "setter". Por ejemplo, para la propiedad `Name` de la interfaz `Document` se pueden usar los métodos en Java `getName` y `setName`.
- **Enumeraciones**. Por cada enumeración definida en la interfaz de automatización hay definida una enumeración en Java con el mismo nombre y los mismos valores.
- **Eventos y controladores de eventos**. Por cada interfaz de la interfaz de automatización que admite eventos existe una interfaz en Java con el mismo nombre, al que se añade 'Event'. Para simplificar el exceso de eventos individuales, la API cuenta con una clase en Java con implementaciones predeterminadas para todos los eventos. El nombre de esta clase en Java es el de la interfaz del evento, al que se añade 'DefaultHandler'. Por ejemplo:

```
Application // Java class to access the application
ApplicationEvents // Events interface for the application
ApplicationEventsDefaultHandler // Default handler for "ApplicationEvents"
```

16.7.1 Interfaces

16.7.1.1 Application

La interfaz `Application` corresponde a un objeto de aplicación de MapForce. También es el principal punto de acceso para la aplicación MapForce. Esta interfaz es el punto de partida para realizar cualquier otra operación con MapForce o para obtener o crear otros objetos de automatización relacionados con MapForce. Para

 © 2016-2022 Altova GmbH

obtener información sobre cómo crear una instancia del objeto Application, consulte el apartado Accessing the API[1226].

Propiedades para navegar en el modelo del objeto:

- `Application`
- `Parent`
- `Options`
- `Project`
- `Documents`

Estado de la aplicación:

- `Visible`
- `Name`
- `Quit`
- `Status`
- `WindowHandle`

Diseños de MapForce:

- `NewDocument`
- `OpenDocument`
- `OpenURL`
- `ActiveDocument`

Proyectos de MapForce:

- `NewProject`
- `OpenProject`
- `ActiveProject`

Generación de código en MapForce:

- `HighlightSerializedMarker`

Recursos globales:

- `GlobalResourceConfig`
- `GlobalResourceFile`

Información de la versión:

- `Edition`
- `IsAPISupported`
- `MajorVersion`
- `MinorVersion`

Propiedades

Nombre	Descripción
`ActiveDocument`[1259]	De solo lectura.

Nombre	Descripción
	Devuelve el objeto de automatización del documento activo en ese momento. Esta propiedad devuelve lo mismo que `Documents.ActiveDocument`.
`ActiveProject` 1259	De solo lectura. Devuelve el objeto de automatización del proyecto activo en ese momento.
`Application` 1260	De solo lectura. Obtiene el objeto de nivel superior de la aplicación.
`Documents` 1260	De solo lectura. Devuelve una colección de todos los documentos abiertos en ese momento.
`Edition` 1261	De solo lectura. Devuelve la edición de la aplicación, por ejemplo "Altova MapForce Enterprise Edition" para esa edición.
`GlobalResourceConfig` 1261	Obtiene o indica el nombre del archivo de configuración activo para los recursos globales. Por defecto, este archivo se llama **GlobalResources.xml**. El archivo de configuración se puede renombrar y guardar en cualquier ubicación. Eso significa que puede tener varios archivos XML para los recursos globales. Sin embargo, solo uno de esos archivos XML para los recursos globales puede estar activo en cada sesión y por cada aplicación, que solo podrá usar las definiciones que contenga el archivo activo.
`GlobalResourceFile` 1262	Obtiene o indica el archivo de definición de los recursos globales. Por defecto, este archivo se llama GlobalResources.xml.
`IsAPISupported` 1262	De solo lectura. Devuelve true si la API es compatible con esta versión de MapForce.
`LibraryImports` 1262	De solo lectura. Obtiene las bibliotecas importadas. En la interfaz gráfica del usuario de MapForce estas se corresponden con entradas de la ventana **Gestionar bibliotecas** añadidas a nivel de la aplicación.
`MajorVersion` 1263	De solo lectura. Obtiene el número principal de la versión de MapForce. La versión se empieza a calcular en 1998 y aumenta con cada año. Por ejemplo, la versión principal para el release 2016 es "18".
`MinorVersion` 1263	De solo lectura. El número complementario del número principal de la versión del producto, por ejemplo 2 para 2006 R2 SP1.

 © 2016-2022 Altova GmbH

Nombre	Descripción
Name 1264	De solo lectura. El nombre de la aplicación.
Options 1264	De solo lectura. Esta propiedad da acceso a las opciones que configuran la generación de código.
Parent 1265	De solo lectura. El objeto de nivel superior según el modelo del objeto.
ServicePackVersion 1265	De solo lectura. El service pack de la versión del producto, por ejemplo 1 para 2016 R2 SP1.
Status 1266	De solo lectura. El estado de la aplicación. Es uno de los valores de la enumeración `ENUMApplicationStatus`.
Visible 1266	**True** si MapForce se muestra en pantalla (aunque puede que esté oculto por otras aplicaciones o minimizado). **False** si MapForce está oculto. El valor predeterminado para MapForce es false cuando este se inicia de forma automática debido a una solicitud del servidor de automatización `Application`. En todos los demás casos la propiedad se inicia como **true**. Se entiende que una instancia visible de la aplicación la controla el usuario (y puede que clientes conectados mediante la interfaz de automatización). La aplicación solo se detendrá por solicitud explícita de un usuario. Para cerrar una instancia de la aplicación, cambie su visibilidad a false y elimine todas las referencias a esta instancia dentro de su programa. La instancia de la aplicación se cerrará automáticamente cuando ningún cliente COM haga referencia a ella.
WindowHandle 1267	De solo lectura. Obtiene la aplicación Windows Handle.

Métodos

Nombre	Descripción
HighlightSerializedMarker 1267	Use este método para resaltar la ubicación en un archivo de asignación previamente seriado. Primero se cargará el documento correspondiente si no lo ha hecho ya. En `Document.GenerateCodeEx` se explica el método para obtener un indicador seriado.

Nombre	Descripción
NewDocument 1268	Crea un documento nuevo vacío. El documento recién abierto se convierte en el ActiveDocument. Este método es una forma abreviada de Documents.NewDocument.
NewProject 1268	Crea un proyecto nuevo vacío. El proyecto actual está cerrado. El proyecto nuevo es accesible desde ActiveProject.
NewWebServiceProject 1269	Crea un servicio web nuevo vacío. El proyecto nuevo es accesible desde ActiveProject. Este método solo está disponible en MapForce Enterprise Edition.
OpenDocument 1269	Carga un documento previamente guardado y continúa trabajando en él. El documento que acaba de abrir se convierte en el ActiveDocument. Este método es una forma abreviada de Documents.OpenDocument.
OpenProject 1270	Abre un proyecto de MapForce ya existente (*.mfp). El proyecto actual está cerrado. El proyecto que acaba de crear es accesible desde ActiveProject.
OpenURL 1270	Carga un archivo previamente guardado desde una ubicación URL. Permite indicar nombre de usuario y contraseña.
Quit 1271	Se desconecta de MapForce para permitir que la aplicación se cierre. El uso de este método es opcional, ya que MapForce controla todas las conexiones COM externas y reconoce de forma automática cuando alguna se desconecta. Para obtener más información sobre el cierre automático, consulte la propiedad Visible.

Eventos

Nombre	Descripción
OnDocumentOpened 1272	Este evento se desencadena cuando se abre un documento, sea nuevo o ya existente. El evento de cierre correspondiente es Document.OnDocumentClosed.
OnProjectOpened 1272	Este evento se desencadena cuando se carga un proyecto en la aplicación. El evento de cierre correspondiente es Project.OnProjectClosed.
OnShutdown 1272	Este evento se desencadena cuando se cierra la aplicación.

© 2016-2022 Altova GmbH

16.7.1.1.1 Propiedades

16.7.1.1.1.1 ActiveDocument

Devuelve el objeto de automatización del documento activo en ese momento. Esta propiedad devuelve lo mismo que `Documents.ActiveDocument`.

Firma

```
ActiveDocument : Document
```

Firma genérica

Errores

Código de error	Descripción
1000	El objeto de aplicación ya no es válido.
1001	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.1.1.2 ActiveProject

Devuelve el objeto de automatización del proyecto activo en ese momento.

Firma

```
ActiveProject : Project
```

Firma genérica

Errores

Código de error	Descripción
1000	El objeto de aplicación ya no es válido.
1001	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.1.1.3 Application

Obtiene el objeto de nivel superior de la aplicación.

Firma

```
Application : Application
```

Firma genérica

Errores

Código de error	Descripción
1000	El objeto de aplicación ya no es válido.
1001	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.1.1.4 Documents

Devuelve una colección de todos los documentos abiertos en ese momento.

Firma

```
Documents : Documents
```

Firma genérica

Errores

Código de error	Descripción
1000	El objeto de aplicación ya no es válido.
1001	La dirección indicada para el parámetro de retorno no es válida.

© 2016-2022 Altova GmbH

16.7.1.1.1.5 Edition

Devuelve la edición de la aplicación, por ejemplo "Altova MapForce Enterprise Edition" para esa edición.

Firma

```
Edition : String
```

Firma genérica

Errores

Código de error	Descripción
1000	El objeto de aplicación ya no es válido.
1001	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.1.1.6 GlobalResourceConfig

Obtiene o indica el nombre del archivo de configuración activo para los recursos globales. Por defecto, este archivo se llama **GlobalResources.xml**.

El archivo de configuración se puede renombrar y guardar en cualquier ubicación. Eso significa que puede tener varios archivos XML para los recursos globales. Sin embargo, solo uno de esos archivos XML para los recursos globales puede estar activo en cada sesión y por cada aplicación, que solo podrá usar las definiciones que contenga el archivo activo.

Firma

```
GlobalResourceConfig : String
```

Firma genérica

Errores

Código de error	Descripción
1000	El objeto de aplicación ya no es válido.
1001	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.1.1.7 GlobalResourceFile

Obtiene o indica el archivo de definición de los recursos globales. Por defecto, este archivo se llama GlobalResources.xml.

Firma

```
GlobalResourceFile : String
```

Firma genérica

Errores

Código de error	Descripción
1000	El objeto de aplicación ya no es válido.
1001	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.1.1.8 IsAPISupported

Devuelve true si la API es compatible con esta versión de MapForce.

Firma

```
IsAPISupported : Boolean
```

Firma genérica

Errores

Código de error	Descripción
1001	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.1.1.9 LibraryImports

Obtiene las bibliotecas importadas. En la interfaz gráfica del usuario de MapForce estas se corresponden con entradas de la ventana **Gestionar bibliotecas** añadidas a nivel de la aplicación.

Firma

```
LibraryImports : LibraryImports
```


© 2016-2022 Altova GmbH

Firma genérica

Errores

Código de error	Descripción
1200	El objeto ya no es válido.
1201	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.1.1.10 MajorVersion

Obtiene el número principal de la versión de MapForce. La versión se empieza a calcular en 1998 y aumenta con cada año. Por ejemplo, la versión principal para el release 2016 es "18".

Firma

```
MajorVersion : Long
```

Firma genérica

Errores

Código de error	Descripción
1000	El objeto de aplicación ya no es válido.
1001	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.1.1.11 MinorVersion

El número complementario del número principal de la versión del producto, por ejemplo 2 para 2006 R2 SP1.

Firma

```
MinorVersion : Long
```

Firma genérica

Errores

Código de error	Descripción
1000	El objeto de aplicación ya no es válido.

Código de error	Descripción
1001	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.1.1.12 Name

El nombre de la aplicación.

Firma

```
Name : String
```

Firma genérica

Errores

Código de error	Descripción
1000	El objeto de aplicación ya no es válido.
1001	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.1.1.13 Options

Esta propiedad da acceso a las opciones que configuran la generación de código.

Firma

```
Options : Options
```

Firma genérica

Errores

Código de error	Descripción
1000	El objeto de aplicación ya no es válido.
1001	La dirección indicada para el parámetro de retorno no es válida.

© 2016-2022 Altova GmbH

16.7.1.1.1.14 Parent

El objeto de nivel superior según el modelo del objeto.

Firma

```
Parent : Application
```

Firma genérica

Errores

Código de error	Descripción
1000	El objeto de aplicación ya no es válido.
1001	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.1.1.15 ServicePackVersion

El service pack de la versión del producto, por ejemplo 1 para 2016 R2 SP1.

Firma

```
ServicePackVersion : Long
```

Firma genérica

Errores

Código de error	Descripción
1000	El objeto de aplicación ya no es válido.
1001	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.1.1.16 Status

El estado de la aplicación. Es uno de los valores de la enumeración `ENUMApplicationStatus`.

Firma

```
Status : ENUMApplicationStatus 1412
```

Firma genérica

Errores

Código de error	Descripción
1001	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.1.1.17 Visible

True si MapForce se muestra en pantalla (aunque puede que esté oculto por otras aplicaciones o minimizado).

False si MapForce está oculto. El valor predeterminado para MapForce es false cuando este se inicia de forma automática debido a una solicitud del servidor de automatización `Application`. En todos los demás casos la propiedad se inicia como **true**.

Se entiende que una instancia visible de la aplicación la controla el usuario (y puede que clientes conectados mediante la interfaz de automatización). La aplicación solo se detendrá por solicitud explícita de un usuario. Para cerrar una instancia de la aplicación, cambie su visibilidad a false y elimine todas las referencias a esta instancia dentro de su programa. La instancia de la aplicación se cerrará automáticamente cuando ningún cliente COM haga referencia a ella.

Firma

```
Visible : Boolean
```

Firma genérica

Errores

Código de error	Descripción
1000	El objeto de aplicación ya no es válido.
1001	La dirección indicada para el parámetro de retorno no es válida.

© 2016-2022 Altova GmbH

16.7.1.1.1.18 WindowHandle

Obtiene la aplicación Windows Handle.

Firma

```
WindowHandle : Long
```

Firma genérica

Errores

Código de error	Descripción
1000	El objeto de aplicación ya no es válido.
1001	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.1.2 Métodos

16.7.1.1.2.1 HighlightSerializedMarker

Use este método para resaltar la ubicación en un archivo de asignación previamente seriado. Primero se cargará el documento correspondiente si no lo ha hecho ya. En `Document.GenerateCodeEx` se explica el método para obtener un indicador seriado.

Firma

```
HighlightSerializedMarker(in i_strSerializedMarker:String) -> Void
```

Firma genérica

Parámetros

Nombre	Tipo	Descripción
i_strSerializedMarker	`String`	El objeto `ErrorMarker` que se quiere resaltar. Use `ErrorMaker.Serialized` para obtener este valor.

Errores

Código de error	Descripción
1000	El objeto de aplicación ya no es válido.
1001	La dirección indicada para el parámetro de retorno no es válida.
1007	La cadena enviada en **i_strSerializedMarker** no se reconoce como indicador seriado de MapForce.
1008	El indicador apunta a una ubicación que ya no es válida.

16.7.1.1.2.2 NewDocument

Crea un documento nuevo vacío. El documento recién abierto se convierte en el `ActiveDocument`. Este método es una forma abreviada de `Documents.NewDocument`.

Firma

```
NewDocument() -> Document
```

Firma genérica

Errores

Código de error	Descripción
1000	El objeto de aplicación ya no es válido.
1001	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.1.2.3 NewProject

Crea un proyecto nuevo vacío. El proyecto actual está cerrado. El proyecto nuevo es accesible desde `ActiveProject`.

Firma

```
NewProject() -> Project
```

Firma genérica

© 2016-2022 Altova GmbH

Errores

Código de error	Descripción
1000	El objeto de aplicación ya no es válido.
1001	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.1.2.4 NewWebServiceProject

Crea un servicio web nuevo vacío. El proyecto nuevo es accesible desde `ActiveProject`. Este método solo está disponible en MapForce Enterprise Edition.

Firma

```
NewWebServiceProject() -> Project
```

Firma genérica

Errores

Código de error	Descripción
1004	Error al crear el nuevo proyecto.
1005	Edición de MapForce no válida.

16.7.1.1.2.5 OpenDocument

Carga un documento previamente guardado y continúa trabajando en él. El documento que acaba de abrir se convierte en el `ActiveDocument`. Este método es una forma abreviada de `Documents.OpenDocument`.

Firma

```
OpenDocument(in i_strFileName:String) -> Document
```

Firma genérica

Parámetros

Nombre	Tipo	Descripción
i_strFileName	`String`	La ruta del documento que se quiere abrir.

Errores

Código de error	Descripción
1000	El objeto de aplicación ya no es válido.
1001	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.1.2.6 OpenProject

Abre un proyecto de MapForce ya existente (*.mfp). El proyecto actual está cerrado. El proyecto que acaba de crear es accesible desde `ActiveProject`.

Firma

```
OpenProject(in i_strFileName:String) -> Project
```

Firma genérica

Parámetros

Nombre	Tipo	Descripción
i_strFileName	`String`	La ruta del proyecto que se quiere abrir.

Errores

Código de error	Descripción
1000	El objeto de aplicación ya no es válido.
1001	La dirección indicada para el parámetro de retorno no es válida.
1002	El nombre de archivo indicado no es válido.

16.7.1.1.2.7 OpenURL

Carga un archivo previamente guardado desde una ubicación URL. Permite indicar nombre de usuario y contraseña.

Firma

```
OpenURL(in strURL:String, in strUser:String, in strPassword:String) -> Void
```


© 2016-2022 Altova GmbH

Firma genérica

Parámetros

Nombre	Tipo	Descripción
strURL	`String`	La URL desde la que se debe cargar el documento.
strUser	`String`	El nombre de usuario necesario para acceder a la URL.
strPassword	`String`	La contraseña necesaria para acceder a la URL.

Errores

Código de error	Descripción
1000	El objeto de aplicación ya no es válido.
1001	La dirección indicada para el parámetro de retorno no es válida.
1002	La URL indicada no es válida.
1006	Error al abrir el archivo URL.

16.7.1.1.2.8 Quit

Se desconecta de MapForce para permitir que la aplicación se cierre. El uso de este método es opcional, ya que MapForce controla todas las conexiones COM externas y reconoce de forma automática cuando alguna se desconecta. Para obtener más información sobre el cierre automático, consulte la propiedad `Visible`.

Firma

```
Quit() -> Void
```

Firma genérica

Errores

Código de error	Descripción
1000	El objeto de aplicación ya no es válido.
1001	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.1.3 Eventos

16.7.1.1.3.1 OnDocumentOpened

Este evento se desencadena cuando se abre un documento, sea nuevo o ya existente. El evento de cierre correspondiente es `Document.OnDocumentClosed`.

Firma

```
OnDocumentOpened(in i_ipDocument:Document) : Void
```

16.7.1.1.3.2 OnProjectOpened

Este evento se desencadena cuando se carga un proyecto en la aplicación. El evento de cierre correspondiente es `Project.OnProjectClosed`.

Firma

```
OnProjectOpened(in i_ipProject:Project) : Void
```

16.7.1.1.3.3 OnShutdown

Este evento se desencadena cuando se cierra la aplicación.

Firma

```
OnShutdown : Void
```

16.7.1.2 AppOutputLine

Representa una línea de mensaje. A diferencia de `ErrorMarker`, su estructura es más detallada y puede contener una colección de líneas secundarias, en cuyo caso forma una estructura de líneas de mensaje.

Propiedades para navegar en el modelo del objeto:

- `Application`
- `Parent`

Línea de acceso:

- `GetLineSeverity`
- `GetLineSymbol`
- `GetLineText`

© 2016-2022 Altova GmbH

- GetLineTextEx
- GetLineTextWithChildren
- GetLineTextWithChildrenEx

Una sola AppOutputLine consiste en una o más sublíneas. Sublínea de acceso:

- GetLineCount

Una sublínea consiste en una o más celdas. Celda de acceso:

- GetCellCountInLine
- GetCellIcon
- GetCellSymbol
- GetCellText
- GetCellTextDecoration
- GetIsCellText

Bajo una AppOutputLine puede haber ninguna, una o más líneas secundarias que también son de tipo AppOutputLine, con lo que se forma una estructura en árbol.

Línea secundaria de acceso:

- ChildLines

Propiedades

Nombre	Descripción
Application 1274	De solo lectura. Obtiene el objeto de nivel superior de la aplicación.
ChildLines 1275	De solo lectura. Devuelve una colección de las líneas secundarias directas de la línea activa.
Parent 1275	De solo lectura. El objeto de nivel superior según el modelo del objeto.

Métodos

Nombre	Descripción
GetCellCountInLine 1276	Obtiene el número de celdas de la sublínea que indica nLine en la AppOutputLine activa.
GetCellIcon 1276	Obtiene el icono de la celda indicada por nCell en la sublínea activa de AppOutputLine indicada por nLine
GetCellSymbol 1277	Obtiene el símbolo de la celda indicada por nCell en la sublínea activa de AppOutputLine indicada por nLine
GetCellText 1277	Obtiene el texto de la celda indicada por nCell en la sublínea activa de AppOutputLine indicada por nLine

Nombre	Descripción
GetCellTextDecoration 1278	Obtiene el formato de la celda de texto indicada por nCell en la sublínea activa AppOutputLine indicada por nLine. Este puede ser uno de los valores ENUMAppOutputLine_TextDecoration.
GetIsCellText 1279	Devuelve true si la celda indicada por nCell en la sublínea activa de AppOutputLine indicada por nLine es una celda de texto.
GetLineCount 1279	Obtiene el número de sublíneas que componen la línea activa en ese momento.
GetLineSeverity 1280	Obtiene la severidad de la línea. Este puede ser uno de los valores ENUMAppOutputLine_Severity.
GetLineSymbol 1280	Obtiene el símbolo asignado a toda la línea.
GetLineText 1281	Obtiene el contenido de la línea como texto.
GetLineTextEx 1281	Obtiene el contenido de la línea como texto usando la parte y la línea especificadas.
GetLineTextWithChildren 1282	Obtiene como texto el contenido de la línea, incluidas todas las líneas secundarias y descendientes.
GetLineTextWithChildrenEx 1282	Obtiene como texto el contenido de la línea, incluidas todas las líneas secundarias y descendientes, usando la parte, la línea, la pestaña y los separadores de elementos que se hayan especificado.

16.7.1.2.1 Propiedades

16.7.1.2.1.1 Application

Obtiene el objeto de nivel superior de la aplicación.

Firma

```
Application : Application
```

Firma genérica

Errores

Código de error	Descripción
4100	El objeto ya no es válido.

© 2016-2022 Altova GmbH

Código de error	Descripción
4101	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.2.1.2 ChildLines

Devuelve una colección de las líneas secundarias directas de la línea activa.

Firma

```
ChildLines : AppOutputLines
```

Firma genérica

Errores

Código de error	Descripción
4100	El objeto ya no es válido.
4101	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.2.1.3 Parent

El objeto de nivel superior según el modelo del objeto.

Firma

```
Parent : AppOutputLines
```

Firma genérica

Errores

Código de error	Descripción
4100	El objeto ya no es válido.
4101	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.2.2 Métodos

16.7.1.2.2.1 GetCellCountInLine

Obtiene el número de celdas de la sublínea que indica `nLine` en la `AppOutputLine` activa.

Firma

```
GetCellCountInLine(in nLine:Long) -> Long
```

Firma genérica

Parámetros

Nombre	Tipo	Descripción
nLine	**Long**	Indica el índice de base cero de la línea.

Errores

Código de error	Descripción
4100	El objeto ya no es válido.
4101	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.2.2.2 GetCellIcon (obsolete)

Obtiene el icono de la celda indicada por `nCell` en la sublínea activa de `AppOutputLine` indicada por `nLine`

Firma

```
GetCellIcon(in nLine:Long, in nCell:Long) -> Long
```

Firma genérica

Parámetros

Nombre	Tipo	Descripción
nLine	**Long**	
nCell	**Long**	

© 2016-2022 Altova GmbH

Errores

Código de error	Descripción
4100	El objeto ya no es válido.
4101	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.2.2.3 GetCellSymbol

Obtiene el símbolo de la celda indicada por `nCell` en la sublínea activa de `AppOutputLine` indicada por `nLine`

Firma

```
GetCellSymbol(in nLine:Long, in nCell:Long) -> AppOutputLineSymbol
```

Firma genérica

Parámetros

Nombre	Tipo	Descripción
nLine	**Long**	Indica el índice de base cero de la línea.
nCell	**Long**	Indica el índice de base cero de la celda.

Errores

Código de error	Descripción
4100	El objeto ya no es válido.
4101	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.2.2.4 GetCellText

Obtiene el texto de la celda indicada por `nCell` en la sublínea activa de `AppOutputLine` indicada por `nLine`

Firma

```
GetCellText(in nLine:Long, in nCell:Long) -> String
```

Firma genérica

Parámetros

Nombre	Tipo	Descripción
nLine	**Long**	Indica el índice de base cero de la línea.
nCell	**Long**	Indica el índice de base cero de la celda.

Errores

Código de error	Descripción
4100	El objeto ya no es válido.
4101	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.2.2.5 GetCellTextDecoration

Obtiene el formato de la celda de texto indicada por `nCell` en la sublínea activa `AppOutputLine` indicada por `nLine`. Este puede ser uno de los valores `ENUMAppOutputLine_TextDecoration`.

Firma

```
GetCellTextDecoration(in nLine:Long, in nCell:Long) -> Long
```

Firma genérica

Parámetros

Nombre	Tipo	Descripción
nLine	**Long**	Indica el índice de base cero de la línea.
nCell	**Long**	Indica el índice de base cero de la celda.

Errores

Código de error	Descripción
4100	El objeto ya no es válido.
4101	La dirección indicada para el parámetro de retorno no es válida.

© 2016-2022 Altova GmbH

16.7.1.2.2.6 GetIsCellText

Devuelve true si la celda indicada por `nCell` en la sublínea activa de `AppOutputLine` indicada por `nLine` es una celda de texto.

Firma

```
GetIsCellText(in nLine:Long, in nCell:Long) -> Boolean
```

Firma genérica

Parámetros

Nombre	Tipo	Descripción
nLine	**Long**	Indica el índice de base cero de la línea.
nCell	**Long**	Indica el índice de base cero de la celda.

Errores

Código de error	Descripción
4100	El objeto ya no es válido.
4101	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.2.2.7 GetLineCount

Obtiene el número de sublíneas que componen la línea activa en ese momento.

Firma

```
GetLineCount() -> Long
```

Firma genérica

Errores

Código de error	Descripción
4100	El objeto ya no es válido.
4101	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.2.2.8 GetLineSeverity

Obtiene la severidad de la línea. Este puede ser uno de los valores `ENUMAppOutputLine_Severity`.

Firma

```
GetLineSeverity() -> Long
```

Firma genérica

Errores

Código de error	Descripción
4100	El objeto ya no es válido.
4101	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.2.2.9 GetLineSymbol

Obtiene el símbolo asignado a toda la línea.

Firma

```
GetLineSymbol() -> AppOutputLineSymbol
```

Firma genérica

Errores

Código de error	Descripción
4100	El objeto ya no es válido.
4101	La dirección indicada para el parámetro de retorno no es válida.

© 2016-2022 Altova GmbH

16.7.1.2.2.10 GetLineText

Obtiene el contenido de la línea como texto.

Firma

```
GetLineText() -> String
```

Firma genérica

Errores

Código de error	Descripción
4100	El objeto ya no es válido.
4101	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.2.2.11 GetLineTextEx

Obtiene el contenido de la línea como texto usando la parte y la línea especificadas.

Firma

```
GetLineTextEx(in psTextPartSeperator:String, in psLineSeperator:String) -> String
```

Firma genérica

Parámetros

Nombre	Tipo	Descripción
psTextPartSeperator	`String`	
psLineSeperator	`String`	

Errores

Código de error	Descripción
4100	El objeto ya no es válido.
4101	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.2.2.12 GetLineTextWithChildren

Obtiene como texto el contenido de la línea, incluidas todas las líneas secundarias y descendientes.

Firma

```
GetLineTextWithChildren() -> String
```

Firma genérica

Errores

Código de error	Descripción
4100	El objeto ya no es válido.
4101	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.2.2.13 GetLineTextWithChildrenEx

Obtiene como texto el contenido de la línea, incluidas todas las líneas secundarias y descendientes, usando la parte, la línea, la pestaña y los separadores de elementos que se hayan especificado.

Firma

```
GetLineTextWithChildrenEx(in psPartSep:String, in psLineSep:String, in psTabSep:String,
in psItemSep:String) -> String
```

Firma genérica

Parámetros

Nombre	Tipo	Descripción
psPartSep	String	
psLineSep	String	
psTabSep	String	
psItemSep	String	

© 2016-2022 Altova GmbH

Errores

Código de error	Descripción
4100	El objeto ya no es válido.
4101	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.3 AppOutputLines

Representa una colección de líneas de mensaje `AppOutputLine`.

Propiedades para navegar en el modelo del objeto:

- `Application`
- `Parent`

Se repiten en la colección:

- `Count`
- `Item`

Propiedades

Nombre	Descripción
`Application` 1284	De solo lectura. Obtiene el objeto de nivel superior de la aplicación.
`Count` 1284	De solo lectura. Obtiene el número de líneas que hay en la colección.
`Item` 1285	De solo lectura. Obtiene la línea del índice `n` de la colección. El índice empieza en el 1.
`Parent` 1285	De solo lectura. El objeto de nivel superior según el modelo del objeto.

16.7.1.3.1 Propiedades

16.7.1.3.1.1 Application

Obtiene el objeto de nivel superior de la aplicación.

Firma

```
Application : Application
```

Firma genérica

Errores

Código de error	Descripción
4000	El objeto ya no es válido.
4001	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.3.1.2 Count

Obtiene el número de líneas que hay en la colección.

Firma

```
Count : Integer
```

Firma genérica

Errores

Código de error	Descripción
4000	El objeto ya no es válido.
4001	La dirección indicada para el parámetro de retorno no es válida.

© 2016-2022 Altova GmbH

16.7.1.3.1.3 Item

Obtiene la línea del índice `n` de la colección. El índice empieza en el 1.

Firma

```
Item(in n:Integer) : AppOutputLine
```

Firma genérica

Errores

Código de error	Descripción
4000	El objeto ya no es válido.
4001	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.3.1.4 Parent

El objeto de nivel superior según el modelo del objeto.

Firma

```
Parent : AppOutputLine
```

Firma genérica

Errores

Código de error	Descripción
4000	El objeto ya no es válido.
4001	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.4 AppOutputLineSymbol

Un `AppOutputLineSymbol` representa un enlace en una línea de mensaje `AppOutputLine` en la que se puede hacer clic desde la ventana de mensajes de MapForce. Se aplica a una celda de una `AppOutputLine` o a toda la línea.

Propiedades para navegar en el modelo del objeto:

- `Application`

- `Parent`

Acceso a los métodos AppOutputLineSymbol:

- `GetSymbolHREF`
- `GetSymbolID`
- `IsSymbolHREF`

Propiedades

Nombre	Descripción
Application 1286	De solo lectura. Obtiene el objeto de nivel superior de la aplicación.
Parent 1287	De solo lectura. El objeto de nivel superior según el modelo del objeto.

Métodos

Nombre	Descripción
GetSymbolHREF 1287	Si el símbolo es de tipo URL, entonces devuelve la URL como una cadena.
GetSymbolID 1288	Obtiene el ID del símbolo.
IsSymbolHREF 1288	Devuelve true si el símbolo es de tipo URL.

16.7.1.4.1 Propiedades

16.7.1.4.1.1 Application

Obtiene el objeto de nivel superior de la aplicación.

Firma

```
Application : Application
```

Firma genérica

Errores

Código de error	Descripción
4200	El objeto ya no es válido.

© 2016-2022 Altova GmbH

Código de error	Descripción
4201	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.4.1.2 Parent

El objeto de nivel superior según el modelo del objeto.

Firma

```
Parent : Application
```

Firma genérica

Errores

Código de error	Descripción
4200	El objeto ya no es válido.
4201	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.4.2 Métodos

16.7.1.4.2.1 GetSymbolHREF

Si el símbolo es de tipo URL, entonces devuelve la URL como una cadena.

Firma

```
GetSymbolHREF() -> String
```

Firma genérica

Errores

Código de error	Descripción
4200	El objeto ya no es válido.
4201	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.4.2.2 GetSymbolID

Obtiene el ID del símbolo.

Firma

```
GetSymbolID() -> Long
```

Firma genérica

Errores

Código de error	Descripción
4200	El objeto ya no es válido.
4201	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.4.2.3 IsSymbolHREF

Devuelve true si el símbolo es de tipo URL.

Firma

```
IsSymbolHREF() -> Boolean
```

Firma genérica

Errores

Código de error	Descripción
4200	El objeto ya no es válido.
4201	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.5 Component

Un `Component` representa un componente de MapForce.

Para navegar por el control, use las propiedades `Application` y `Parent`.

Propiedades de los componentes:

- `HasIncomingConnections`

 © 2016-2022 Altova GmbH

- HasOutgoingConnections
- CanChangeInputInstanceFile
- CanChangeOutputInstanceFile
- ComponentName.
- ID
- IsParameterInputRequired
- IsParameterSequence
- Name
- Preview
- Schema
- SubType
- Type

Propiedades relacionadas con instancias:

- InputInstanceFile
- OutputInstanceFile

Puntos de datos:

- GetRootDatapoint

Ejecución:

- GenerateOutput

Propiedades

Nombre	Descripción
Application 1291	De solo lectura. Obtiene el objeto de nivel superior de la aplicación.
CanChangeInputInstanceFile 1292	De solo lectura. Indica si se puede modificar el nombre del archivo de instancia de entrada. Devuelve false si el componente tiene un nodo de nombre de archivo y este nodo tiene una conexión en su lado izquierdo (entrada); de lo contrario, devuelve true. Si el componente no tiene un nodo de nombre de archivo, devuelve false.
CanChangeOutputInstanceFile 1292	De solo lectura. Indica si se puede modificar el nombre del archivo de instancia de salida. Devuelve false si el componente tiene un nodo de nombre de archivo y este nodo tiene una conexión en su lado izquierdo (entrada); de lo contrario, devuelve true. Si el componente no tiene un nodo de nombre de archivo, devuelve false.
ComponentName 1293	Obtiene o indica el nombre del componente.

Nombre	Descripción
HasIncomingConnections 1293	De solo lectura. Indica si el componente tiene conexiones entrantes (en su lado izquierdo), sin incluir el nodo de nombre de archivo. Una conexión entrante en el nodo de nombre de archivo no tiene efecto en el valor devuelto.
HasOutgoingConnections 1294	De solo lectura. Indica si el componente tiene alguna conexión saliente (en su lado derecho).
ID 1294	De solo lectura. Obtiene el ID del componente.
InputInstanceFile 1295	Obtiene o indica el archivo de instancia de entrada del componente.
IsParameterInputRequired 1295	Averigua o define si el componente del parámetro de entrada necesita una conexión entrante en el componente de llamada de función de la función definida por el usuario en que se encuentra este componente de parámetro de entrada. Esta propiedad funciona solo para componentes de parámetro de entrada.
IsParameterSequence 1296	Averigua o define si el componente de parámetro de entrada o el de de salida admiten secuencias. Esta propiedad solo funciona para componentes de parámetro de entrada o de salida.
Name 1296	De solo lectura. Obtiene el nombre del componente.
OutputInstanceFile 1297	Obtiene o indica el archivo de instancia de salida del componente. No devuelve ningún dato al intentar acceder al `OutputInstanceFile` de un componente mediante la API si el conector "Archivo" del componente se ha conectado a otro elemento en la asignación.
Parent 1297	De solo lectura. El objeto de nivel superior según el modelo del objeto.
Preview 1297	Averigua o define si el componente es el componente actual de vista previa. Esta propiedad funciona solo para componentes que son componentes de destino en la asignación del documento principal. Solo un componente de destino de la asignación principal puede ser el componente de vista previa cada vez. Al definir esta propiedad, solo es posible establecerla en true. Esto implica que la propiedad `Preview` de todos los demás componentes cambiará a false.

© 2016-2022 Altova GmbH

Nombre	Descripción
	Si solo existe un único componente de destino en la asignación principal, este será el componente de vista previa.
Schema 1298	De solo lectura. Obtiene el nombre de archivo del esquema del componente.
SubType 1299	De solo lectura. Obtiene el subtipo del componente.
Type 1299	De solo lectura. Obtiene el tipo del componente.
UsageKind 1299	De solo lectura. Obtiene el tipo de uso del componente.

Métodos

Nombre	Descripción
GenerateOutput 1300	Genera los archivos de salida definidos en la asignación solo para el componente actual y usa para ello un lenguaje de asignación interno de MapForce. Los nombres de los archivos de salida están definidos como una propiedad del componente actual, que es el elemento de salida en la asignación de este proceso de generación.
GetRootDatapoint 1300	Obtiene puntos de datos raíz en el lado izquierdo (entrada) o derecho (salida) de un componente. El objeto `Datapoint` ofrece otros métodos para acceder a secundarios y descendientes.

16.7.1.5.1 Propiedades

16.7.1.5.1.1 Application

Obtiene el objeto de nivel superior de la aplicación.

Firma

```
Application : Application
```

Firma genérica

Errores

Código de error	Descripción
1200	El objeto ya no es válido.
1201	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.5.1.2 CanChangeInputInstanceFile

Indica si se puede modificar el nombre del archivo de instancia de entrada.

Devuelve false si el componente tiene un nodo de nombre de archivo y este nodo tiene una conexión en su lado izquierdo (entrada); de lo contrario, devuelve true. Si el componente no tiene un nodo de nombre de archivo, devuelve false.

Firma

```
CanChangeInputInstanceFile : Boolean
```

Firma genérica

Errores

Código de error	Descripción
1200	El objeto ya no es válido.
1201	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.5.1.3 CanChangeOutputInstanceFile

Indica si se puede modificar el nombre del archivo de instancia de salida.

Devuelve false si el componente tiene un nodo de nombre de archivo y este nodo tiene una conexión en su lado izquierdo (entrada); de lo contrario, devuelve true.
Si el componente no tiene un nodo de nombre de archivo, devuelve false.

Firma

```
CanChangeOutputInstanceFile : Boolean
```

Firma genérica

© 2016-2022 Altova GmbH

Errores

Código de error	Descripción
1200	El objeto ya no es válido.
1201	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.5.1.4 ComponentName

Obtiene o indica el nombre del componente.

Firma

```
ComponentName : String
```

Firma genérica

Errores

Código de error	Descripción
1200	El objeto ya no es válido.
1201	La dirección indicada para el parámetro de retorno no es válida.
1246	El componente no admite que se establezca su nombre.
1247	El nombre del componente no es válido.

16.7.1.5.1.5 HasIncomingConnections

Indica si el componente tiene conexiones entrantes (en su lado izquierdo), sin incluir el nodo de nombre de archivo. Una conexión entrante en el nodo de nombre de archivo no tiene efecto en el valor devuelto.

Firma

```
HasIncomingConnections : Boolean
```

Firma genérica

Errores

Código de error	Descripción
1200	El objeto ya no es válido.
1201	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.5.1.6 HasOutgoingConnections

Indica si el componente tiene alguna conexión saliente (en su lado derecho).

Firma

```
HasOutgoingConnections : Boolean
```

Firma genérica

Errores

Código de error	Descripción
1200	El objeto ya no es válido.
1201	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.5.1.7 ID

Obtiene el ID del componente.

Firma

```
ID : Long
```

Firma genérica

Errores

Código de error	Descripción
1200	El objeto ya no es válido.
1201	La dirección indicada para el parámetro de retorno no es válida.

© 2016-2022 Altova GmbH

16.7.1.5.1.8 InputInstanceFile

Obtiene o indica el archivo de instancia de entrada del componente.

Firma

```
InputInstanceFile : String
```

Firma genérica

Errores

Código de error	Descripción
1200	El objeto ya no es válido.
1201	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.5.1.9 IsParameterInputRequired

Averigua o define si el componente del parámetro de entrada necesita una conexión entrante en el componente de llamada de función de la función definida por el usuario en que se encuentra este componente de parámetro de entrada. Esta propiedad funciona solo para componentes de parámetro de entrada.

Firma

```
IsParameterInputRequired : Boolean
```

Firma genérica

Errores

Código de error	Descripción
1200	El objeto ya no es válido.
1201	La dirección indicada para el parámetro de retorno no es válida.
1232	Esta operación solo funciona para componentes de parámetros de entrada.
1240	No está permitido cambiar el documento. Es solo de lectura.

16.7.1.5.1.10 IsParameterSequence

Averigua o define si el componente de parámetro de entrada o el de de salida admiten secuencias. Esta propiedad solo funciona para componentes de parámetro de entrada o de salida.

Firma

```
IsParameterSequence : Boolean
```

Firma genérica

Errores

Código de error	Descripción
1200	El objeto ya no es válido.
1201	La dirección indicada para el parámetro de retorno no es válida.
1233	Esta operación solo funciona para componentes de parámetros de entrada o de salida.
1240	No está permitido cambiar el documento. Es solo de lectura.

16.7.1.5.1.11 Name

Obtiene el nombre del componente.

Firma

```
Name : String
```

Firma genérica

Errores

Código de error	Descripción
1200	El objeto ya no es válido.
1201	La dirección indicada para el parámetro de retorno no es válida.

© 2016-2022 Altova GmbH

16.7.1.5.1.12 OutputInstanceFile

Obtiene o indica el archivo de instancia de salida del componente.

No devuelve ningún dato al intentar acceder al `OutputInstanceFile` de un componente mediante la API si el conector "Archivo" del componente se ha conectado a otro elemento en la asignación.

Firma

```
OutputInstanceFile : String
```

Firma genérica

Errores

Código de error	Descripción
1200	El objeto ya no es válido.
1201	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.5.1.13 Parent

El objeto de nivel superior según el modelo del objeto.

Firma

```
Parent : Mapping
```

Firma genérica

Errores

Código de error	Descripción
1200	El objeto ya no es válido.
1201	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.5.1.14 Preview

Averigua o define si el componente es el componente actual de vista previa.

Esta propiedad funciona solo para componentes que son componentes de destino en la asignación del documento principal. Solo un componente de destino de la asignación principal puede ser el componente de vista previa cada vez.

Al definir esta propiedad, solo es posible establecerla en true. Esto implica que la propiedad `Preview` de todos los demás componentes cambiará a false.

Si solo existe un único componente de destino en la asignación principal, este será el componente de vista previa.

Firma

```
Preview : Boolean
```

Firma genérica

Errores

Código de error	Descripción
1200	El objeto ya no es válido.
1201	La dirección indicada para el parámetro de retorno no es válida.
1234	Solo se puede definir un componente de la asignación principal como componente de vista previa.
1235	Un componente no se puede definir como de no vista previa. Defina otro componente como componente de vista previa.

16.7.1.5.1.15 Schema

Obtiene el nombre de archivo del esquema del componente.

Firma

```
Schema : String
```

Firma genérica

Errores

Código de error	Descripción
1200	El objeto ya no es válido.
1201	La dirección indicada para el parámetro de retorno no es válida.

© 2016-2022 Altova GmbH

16.7.1.5.1.16 SubType

Obtiene el subtipo del componente.

Firma

```
SubType : ENUMComponentSubType 1414
```

Firma genérica

Errores

Código de error	Descripción
1200	El objeto ya no es válido.
1201	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.5.1.17 Type

Obtiene el tipo del componente.

Firma

```
Type : ENUMComponentType 1415
```

Firma genérica

16.7.1.5.1.18 UsageKind

Obtiene el tipo de uso del componente.

Firma

```
UsageKind : ENUMComponentUsageKind 1415
```

Firma genérica

Errores

Código de error	Descripción
1200	El objeto ya no es válido.

Código de error	Descripción
1201	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.5.2 Métodos

16.7.1.5.2.1 GenerateOutput

Genera los archivos de salida definidos en la asignación solo para el componente actual y usa para ello un lenguaje de asignación interno de MapForce. Los nombres de los archivos de salida están definidos como una propiedad del componente actual, que es el elemento de salida en la asignación de este proceso de generación.

Firma

```
GenerateOutput(out pbError:Boolean) -> AppOutputLines
```

Firma genérica

Parámetros

Nombre	Tipo	Descripción
pbError	`Boolean`	Este es un parámetro solo de salida. Solo recibirá un valor si el lenguaje de llamada admite parámetros de salida. Si no, el valor que envíe aquí permanecerá igual cuando termine la función.

Errores

Código de error	Descripción
1200	El objeto ya no es válido.
1201	La dirección indicada para el parámetro de retorno no es válida.
1248	Solo se admite la generación de archivos de salida si la interfaz gráfica del usuario está visible.

16.7.1.5.2.2 GetRootDatapoint

Obtiene puntos de datos raíz en el lado izquierdo (entrada) o derecho (salida) de un componente. El objeto `Datapoint` ofrece otros métodos para acceder a secundarios y descendientes.

 © 2016-2022 Altova GmbH

Firma

```
GetRootDatapoint(in side:ENUMComponentDatapointSide[1414], in strNamespace:String, in
strLocalName:String, in strParameterName:String) -> Datapoint
```

Firma genérica

Parámetros

Nombre	Tipo	Descripción
side	ENUMComponentDatapointSide [1414]	El parámetro "side" indica si se debe obtener un punto de datos de entrada o de salida.
strNamespace	String	El espacio de nombre y el nombre local especificados indican el nombre del nodo cuyo punto de datos se debe obtener. Para componentes con información estructural, como los componentes de esquema, deberá indicar el espacio de nombres junto con el nombre local o puede enviar una cadena vacía para el espacio de nombres. Los componentes basados en archivos, como los componentes de esquema, contienen en su raíz un nodo de espacio de nombres. Aquí `GetRootDatapoint` solo puede encontrar el nodo de nombre de archivo. Tendrá que enviar el espacio de nombres `"http://www.altova.com/mapforce»` y el nombre local `"FileInstance»` para obtener un punto de datos de este nodo.
strLocalName	String	Véase más arriba.
strParameterName	String	El parámetro indicado debe ser una cadena vacía a no ser que el componente en cuestión sea un componente de llamada de función. Como una función definida por el usuario puede contener parámetros de entrada o de salida de la misma estructura, el

Nombre	Tipo	Descripción
		componente de llamada de función que llama a esta función definida por el usuario puede tener más de un nodo raíz con el mismo espacio de nombre y el mismo nombre local. Estos solo se diferencian por sus nombres de parámetro, que en realidad son los nombres de los correspondientes componentes de parámetro en la asignación de la función definida por el usuario. Sin embargo, no es obligatorio especificar el nombre del parámetro. En ese caso, el método devolverá el primer punto de datos raíz que coincida con el nombre de espacios y el nombre local especificados.

Errores

Código de error	Descripción
1200	El objeto ya no es válido.
1201	La dirección indicada para el parámetro de retorno no es válida.
1248	No se encontró el punto de datos.

16.7.1.6 Components

Representa una colección de objetos `Component`.

Para navegar por el modelo del objeto, use las siguientes propiedades:

- `Application`
- `Parent`

Para repetir en la colección:

- `Count`
- `Item`

© 2016-2022 Altova GmbH

Propiedades

Nombre	Descripción
Application 1303	De solo lectura. Obtiene el objeto de nivel superior de la aplicación.
Count 1304	De solo lectura. Obtiene el número de componentes que hay en la colección.
Item 1304	De solo lectura. Obtiene el componente del índice n de la colección. El índice empieza en el 1.
Parent 1305	De solo lectura. El objeto de nivel superior según el modelo del objeto.

16.7.1.6.1 Propiedades

16.7.1.6.1.1 Application

Obtiene el objeto de nivel superior de la aplicación.

Firma

```
Application : Application
```

Firma genérica

Errores

Código de error	Descripción
1200	El objeto ya no es válido.
1201	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.6.1.2 Count

Obtiene el número de componentes que hay en la colección.

Firma

```
Count : Integer
```

Firma genérica

Errores

Código de error	Descripción
1200	El objeto ya no es válido.
1201	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.6.1.3 Item

Obtiene el componente del índice n de la colección. El índice empieza en el 1.

Firma

```
Item(in n:Integer) : Component
```

Firma genérica

Errores

Código de error	Descripción
1200	El objeto ya no es válido.
1201	La dirección indicada para el parámetro de retorno no es válida.

© 2016-2022 Altova GmbH

16.7.1.6.1.4 Parent

El objeto de nivel superior según el modelo del objeto.

Firma

```
Parent : Mapping
```

Firma genérica

Errores

Código de error	Descripción
1200	El objeto ya no es válido.
1201	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.7 Connection

Un objeto `Connection` representa un conector entre dos componentes.

Para navegar por el modelo del objeto, use las siguientes propiedades:

- `Application`
- `Parent`

Para obtener o indicar el tipo de conexión, use `ConnectionType`.

Propiedades

Nombre	Descripción
`Application` 1306	De solo lectura. Obtiene el objeto de nivel superior de la aplicación.
`ConnectionType` 1306	Obtiene o indica el tipo de conexión.
`Parent` 1307	De solo lectura. El objeto de nivel superior según el modelo del objeto.

16.7.1.7.1 Propiedades

16.7.1.7.1.1 Application

Obtiene el objeto de nivel superior de la aplicación.

Firma

```
Application : Application
```

Firma genérica

Errores

Código de error	Descripción
2100	El objeto ya no es válido.
2101	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.7.1.2 ConnectionType

Obtiene o indica el tipo de conexión.

Firma

```
ConnectionType : ENUMConnectionType 1416
```

Firma genérica

Errores

Código de error	Descripción
2100	El objeto ya no es válido.
2101	La dirección indicada para el parámetro de retorno no es válida.
2102	No está permitido cambiar el documento. Es solo de lectura.
2103	Error al cambiar el tipo de conexión.

© 2016-2022 Altova GmbH

16.7.1.7.1.3 Parent

El objeto de nivel superior según el modelo del objeto.

Firma

```
Parent : Mapping
```

Firma genérica

Errores

Código de error	Descripción
2100	El objeto ya no es válido.
2101	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.8 Datapoint

Un objeto `Datapoint` representa un icono de entrada o salida de un componente.

Propiedades

Nombre	Descripción
`Application` 1308	De solo lectura. Obtiene el objeto de nivel superior de la aplicación.
`Parent` 1308	De solo lectura. El objeto de nivel superior según el modelo del objeto.

Métodos

Nombre	Descripción
`GetChild` 1309	Busca un punto de datos secundario directo del punto de datos actual por espacio de nombres y nombre local. Si un componente de esquema tiene elementos con contenido mixto, cada uno de ellos muestra un nodo secundario adicional, que es el nodo **text()**. Para obtener el punto de datos de un nodo **text()** debe enviar una cadena vacía a `strNamespace`, `"#text"` a `strLocalName` y `eSearchDatapointElement` a `searchFlags`.

16.7.1.8.1 Propiedades

16.7.1.8.1.1 Application

Obtiene el objeto de nivel superior de la aplicación.

Firma

```
Application : Application
```

Firma genérica

Errores

Código de error	Descripción
2000	El objeto ya no es válido.
2001	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.8.1.2 Parent

El objeto de nivel superior según el modelo del objeto.

Firma

```
Parent : Component
```

Firma genérica

Errores

Código de error	Descripción
2000	El objeto ya no es válido.
2001	La dirección indicada para el parámetro de retorno no es válida.

© 2016-2022 Altova GmbH

16.7.1.8.2 Métodos

16.7.1.8.2.1 GetChild

Busca un punto de datos secundario directo del punto de datos actual por espacio de nombres y nombre local.

Si un componente de esquema tiene elementos con contenido mixto, cada uno de ellos muestra un nodo secundario adicional, que es el nodo **text()**. Para obtener el punto de datos de un nodo **text()** debe enviar una cadena vacía a `strNamespace`, `"#text"` a `strLocalName` y `eSearchDatapointElement` a `searchFlags`.

Firma

```
GetChild(in strNamespace:String, in strLocalName:String, in
searchFlags:ENUMSearchDatapointFlags[1418]) -> Datapoint
```

Firma genérica

Parámetros

Nombre	Tipo	Descripción
strNamespace	`String`	El espacio de nombres del punto de datos secundario directo.
strLocalName	`String`	El nombre del punto de datos secundario directo.
searchFlags	`ENUMSearchDatapointFlags`[1418]	Las banderas de búsqueda se pueden enviar como combinaciones de valores (combinadas usando el operador lógico OR) de la enumeración `ENUMSearchDatapointFlags`.

Errores

Código de error	Descripción
2000	El objeto ya no es válido.
2001	La dirección indicada para el parámetro de retorno no es válida.
2002	No se encontró el punto de datos.

16.7.1.9 Document

Un objeto `Document` representa un documento de MapForce (un archivo MFD cargado). Un documento contiene una asignación principal y ninguna, una o más asignaciones locales de funciones definidas por el usuario.

Para navegar por el modelo del objeto, use las siguientes propiedades:

- `Application`
- `Parent`

Para gestionar archivos, use:

- `Activate`
- `Close`
- `FullName`
- `Name`
- `Path`
- `Saved`
- `Save`
- `SaveAs`

Para gestionar asignaciones, use:

- `MainMapping`
- `Mappings`
- `CreateUserDefinedFunction`

Para gestionar componentes, use:

- `FindComponentByID`

Para generar código, use:

- `OutputSettings_ApplicationName`
- `JavaSettings_BasePackageName`
- `GenerateCHashCode`
- `GenerateCodeEx`
- `GenerateCppCode`
- `GenerateJavaCode`
- `GenerateXQuery`
- `GenerateXSLT`
- `GenerateXSLT2`
- `GenerateXSLT3`
- `HighlightSerializedMarker`

Para ejecutar la asignación, use:

- `GenerateOutput`
- `GenerateOutputEx`

Ver acceso:

 © 2016-2022 Altova GmbH

- MapForceView

Obsoleto:

- OutputSettings_Encoding

Propiedades

Nombre	Descripción
Application 1314	De solo lectura. Obtiene el objeto de nivel superior de la aplicación.
FullName 1314	Ruta de acceso y nombre del fichero de documento.
JavaSettings_BasePackageName 1315	Indica u obtiene el nombre del paquete base usado al generar código Java. En la interfaz gráfica de usuario de MapForce, esta opción está disponible en el cuadro de diálogo **Configurar asignación** (haga clic con el botón derecho en la asignación y seleccione **Configurar asignación** del menú contextual).
LibraryImports 1315	De solo lectura. Obtiene las bibliotecas importadas. En la interfaz gráfica del usuario de MapForce estas se corresponden con entradas de la ventana **Gestionar bibliotecas** añadidas a nivel del documento.
MainMapping 1316	De solo lectura. Obtiene la asignación principal del documento.
MapForceView 1316	De solo lectura. Esta propiedad da acceso a las funciones de la vista MapForce.
Mappings 1317	De solo lectura. Devuelve una colección de las asignaciones contenidas en el documento.
Name 1317	De solo lectura. Nombre del fichero de documento sin ruta de acceso.
OutputSettings_ApplicationName 1317	Indica u obtiene el nombre de la aplicación disponible en el cuadro de diálogo **Configurar asignación** (para mostrar este cuadro de diálogo en MapForce, haga clic con el botón derecho en la asignación y seleccione **Configurar asignación** del menú contextual).
OutputSettings_Encoding 1318	Esta propiedad ya no se admite. La configuración de codificación de salida de la asignación ya no existe. Cada componente tiene su propia configuración de codificación de salida.
Parent 1318	De solo lectura. El objeto de nivel superior según el modelo del objeto.
Path 1319	De solo lectura.

Nombre	Descripción
	Ruta del fichero de documento sin nombre.
Saved 1319	De solo lectura. **True** si el documento no se ha modificado desde la última que se guardó, **false** en caso contrario.

Métodos

Nombre	Descripción
Activate 1320	Convierte este documento en el documento activo.
Close 1320	Cierra el documento sin guardarlo.
CreateUserDefinedFunction 1321	Crea una función definida por el usuario en el documento actual.
FindComponentByID 1322	Busca en todo el documento y en todas las asignaciones de este el componente con el id indicado.
GenerateCHashCode 1322	Genera el código C# que ejecutará la asignación. Usa las propiedades definidas en `Application.Options` para configurar la generación del código.
GenerateCodeEx 1323	Genera el código que ejecutará la asignación. El parámetro **i_nLanguage** indica el lenguaje de destino. El método devuelve un objeto que se puede usar para enumerar todos los mensajes creados por el generador de código. Estos son los mismos mensajes que se muestran en la ventana Mensajes de MapForce.
GenerateCppCode 1323	Genera el código C++ que ejecutará la asignación. Usa las propiedades definidas en `Application.Options` para configurar la generación del código.
GenerateJavaCode 1324	Genera el código Java que ejecutará la asignación. Usa las propiedades definidas en `Application.Options` para configurar la generación del código.
GenerateOutput 1324	Genera todos los archivos de salida definidos en la asignación usando para ello un lenguaje de asignación interno de MapForce. Los nombres de los archivos de salida están definidos como propiedades de los elementos de salida en la asignación. **Nota:** este método solo se puede usar cuando la ventana principal de MapForce (como servidor COM) está visible o incrustada en una interfaz gráfica de usuario. Si se llama al método mientras MapForce no está visible se generará un error.
GenerateOutputEx 1325	Genera todos los archivos de salida definidos en la asignación usando para ello un lenguaje de asignación interno de

 © 2016-2022 Altova GmbH

Nombre	Descripción
	MapForce. Los nombres de los archivos de salida están definidos como propiedades de los elementos de salida en la asignación. Este método es idéntico a `GenerateOutput`, salvo porque el valor que devuelve contiene los mensajes, las advertencias y los errores resultantes, organizados en estructuras en forma de árbol de `AppOutputLines`. **Nota:** este método solo se puede usar cuando la ventana principal de MapForce (como servidor COM) está visible o incrustada en una interfaz gráfica de usuario. Si se llama al método mientras MapForce no está visible se generará un error.
`GenerateXQuery` 1326	Genera código de asignación como XQuery. Usa las propiedades definidas en `Application.Options` para configurar la generación del código.
`GenerateXSLT` 1326	Genera código de asignación como XSLT. Usa las propiedades definidas en `Application.Options` para configurar la generación del código.
`GenerateXSLT2` 1327	Genera código de asignación como XSLT2. Usa las propiedades definidas en `Application.Options` para configurar la generación del código.
`GenerateXSLT3` 1327	Genera código de asignación como XSLT 3.0. Usa las propiedades definidas en `Application.Options` para configurar la generación del código.
`HighlightSerializedMarker` 1328	Use este método para resaltar la ubicación en un archivo de asignación previamente seriado. Primero se cargará el documento correspondiente si no lo ha hecho ya. En `GenerateCodeEx` se explica el método para obtener un indicador seriado.
`Save` 1328	Guarda el documento en el archivo definido por `Document.FullName`.
`SaveAs` 1329	Guarda el documento con el nombre de archivo indicado y establece `Document.FullName` en este valor si la operación se realiza con éxito.

Eventos

Nombre	Descripción
`OnDocumentClosed` 1329	Este evento se desencadena cuando se cierra un documento. No se debe llamar al objeto documento pasado al gestor de eventos. El evento de apertura correspondiente es `Application.OnDocumentOpened`.

Nombre	Descripción
OnModifiedFlagChanged 1330	Este evento se desencadena cuando cambia el estado de un documento.

16.7.1.9.1 Propiedades

16.7.1.9.1.1 Application

Obtiene el objeto de nivel superior de la aplicación.

Firma

```
Application : Application
```

Firma genérica

Errores

Código de error	Descripción
1200	El objeto ya no es válido.
1201	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.9.1.2 FullName

Ruta de acceso y nombre del fichero de documento.

Firma

```
FullName : String
```

Firma genérica

Errores

Código de error	Descripción
1200	El objeto ya no es válido.
1201	La dirección indicada para el parámetro de retorno no es válida.

© 2016-2022 Altova GmbH

16.7.1.9.1.3 JavaSettings_BasePackageName

Indica u obtiene el nombre del paquete base usado al generar código Java. En la interfaz gráfica de usuario de MapForce, esta opción está disponible en el cuadro de diálogo **Configurar asignación** (haga clic con el botón derecho en la asignación y seleccione **Configurar asignación** del menú contextual).

Firma

```
JavaSettings_BasePackageName : String
```

Firma genérica

Errores

Código de error	Descripción
1200	El objeto ya no es válido.
1201	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.9.1.4 LibraryImports

Obtiene las bibliotecas importadas. En la interfaz gráfica del usuario de MapForce estas se corresponden con entradas de la ventana **Gestionar bibliotecas** añadidas a nivel del documento.

Firma

```
LibraryImports : LibraryImports
```

Firma genérica

Errores

Código de error	Descripción
1200	El objeto ya no es válido.
1201	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.9.1.5 MainMapping

Obtiene la asignación principal del documento.

Firma

```
MainMapping : Mapping
```

Firma genérica

Errores

Código de error	Descripción
1200	El objeto ya no es válido.
1201	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.9.1.6 MapForceView

Esta propiedad da acceso a las funciones de la vista MapForce.

Firma

```
MapForceView : MapForceView
```

Firma genérica

Errores

Código de error	Descripción
1200	El objeto ya no es válido.
1201	La dirección indicada para el parámetro de retorno no es válida.

© 2016-2022 Altova GmbH

16.7.1.9.1.7 Mappings

Devuelve una colección de las asignaciones contenidas en el documento.

Firma

```
Mappings : Mappings
```

Firma genérica

Errores

Código de error	Descripción
1200	El objeto ya no es válido.
1201	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.9.1.8 Name

Nombre del fichero de documento sin ruta de acceso.

Firma

```
Name : String
```

Firma genérica

Errores

Código de error	Descripción
1200	El objeto ya no es válido.
1201	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.9.1.9 OutputSettings_ApplicationName

Indica u obtiene el nombre de la aplicación disponible en el cuadro de diálogo **Configurar asignación** (para mostrar este cuadro de diálogo en MapForce, haga clic con el botón derecho en la asignación y seleccione **Configurar asignación** del menú contextual).

Firma

```
OutputSettings_ApplicationName : String
```

Firma genérica

Errores

Código de error	Descripción
1200	El objeto ya no es válido.
1201	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.9.1.10 OutputSettings_Encoding (obsolete)

Esta propiedad ya no se admite. La configuración de codificación de salida de la asignación ya no existe. Cada componente tiene su propia configuración de codificación de salida.

Firma

```
OutputSettings_Encoding : String
```

Firma genérica

16.7.1.9.1.11 Parent

El objeto de nivel superior según el modelo del objeto.

Firma

```
Parent : Documents
```

Firma genérica

Errores

Código de error	Descripción
1200	El objeto ya no es válido.
1201	La dirección indicada para el parámetro de retorno no es válida.

© 2016-2022 Altova GmbH

16.7.1.9.1.12 Path

Ruta del fichero de documento sin nombre.

Firma

```
Path : String
```

Firma genérica

Errores

Código de error	Descripción
1200	El objeto ya no es válido.
1201	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.9.1.13 Saved

True si el documento no se ha modificado desde la última que se guardó, **false** en caso contrario.

Firma

```
Saved : Boolean
```

Firma genérica

Errores

Código de error	Descripción
1200	El objeto ya no es válido.
1201	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.9.2 Métodos

16.7.1.9.2.1 Activate

Convierte este documento en el documento activo.

Firma

```
Activate() -> Void
```

Firma genérica

Errores

Código de error	Descripción
1200	El objeto ya no es válido.

16.7.1.9.2.2 Close

Cierra el documento sin guardarlo.

Firma

```
Close() -> Void
```

Firma genérica

Errores

Código de error	Descripción
1200	El objeto ya no es válido.
1201	La dirección indicada para el parámetro de retorno no es válida.

© 2016-2022 Altova GmbH

16.7.1.9.2.3 CreateUserDefinedFunction

Crea una función definida por el usuario en el documento actual.

Firma

```
CreateUserDefinedFunction(in strFunctionName:String, in strLibraryName:String, in
strSyntax:String, in strDetails:String, in bInlinedUse:Boolean) -> Mapping
```

Firma genérica

Parámetros

Nombre	Tipo	Descripción
strFunctionName	`String`	El nombre de la función.
strLibraryName	`String`	El nombre de la biblioteca a la que pertenece esta función.
strSyntax	`String`	Una cadena que describe la sintaxis de esta función (solo con fines informativos).
strDetails	`String`	Una descripción de esta función.
bInlinedUse	`Boolean`	Marcador binario que indica si la función tiene un uso inline.

Errores

Código de error	Descripción
1200	El objeto ya no es válido.
1201	La dirección indicada para el parámetro de retorno no es válida.
1208	Error al crear la función definida por el usuario.
1209	No está permitido cambiar el documento. Es solo de lectura.

16.7.1.9.2.4 FindComponentByID

Busca en todo el documento y en todas las asignaciones de este el componente con el id indicado.

Firma

```
FindComponentByID(in nID:Unsigned Long) -> Component
```

Firma genérica

Parámetros

Nombre	Tipo	Descripción
nID	`Unsigned Long`	El ID del componente que hay que buscar.

Errores

Código de error	Descripción
1200	El objeto ya no es válido.
1201	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.9.2.5 GenerateCHashCode

Genera el código C# que ejecutará la asignación. Usa las propiedades definidas en `Application.Options` para configurar la generación del código.

Firma

```
GenerateCHashCode() -> Void
```

Firma genérica

Errores

Código de error	Descripción
1200	El objeto ya no es válido.
1201	La dirección indicada para el parámetro de retorno no es válida.
1205	Error durante la generación del código.

© 2016-2022 Altova GmbH

16.7.1.9.2.6 GenerateCodeEx

Genera el código que ejecutará la asignación. El parámetro **i_nLanguage** indica el lenguaje de destino. El método devuelve un objeto que se puede usar para enumerar todos los mensajes creados por el generador de código. Estos son los mismos mensajes que se muestran en la ventana Mensajes de MapForce.

Firma

```
GenerateCodeEx(in i_nLanguage:ENUMProgrammingLanguage[1416]) -> ErrorMarkers
```

Firma genérica

Parámetros

Nombre	Tipo	Descripción
i_nLanguage	ENUMProgrammingLanguage[1416]	Indica el lenguaje de generación de código de destino.

Errores

Código de error	Descripción
1200	El objeto ya no es válido.
1201	La dirección indicada para el parámetro de retorno no es válida.
1205	Error durante la generación del código.

16.7.1.9.2.7 GenerateCppCode

Genera el código C++ que ejecutará la asignación. Usa las propiedades definidas en `Application.Options` para configurar la generación del código.

Firma

```
GenerateCppCode() -> Void
```

Firma genérica

Errores

Código de error	Descripción
1200	El objeto ya no es válido.

Código de error	Descripción
1201	La dirección indicada para el parámetro de retorno no es válida.
1205	Error durante la generación del código.

16.7.1.9.2.8 GenerateJavaCode

Genera el código Java que ejecutará la asignación. Usa las propiedades definidas en `Application.Options` para configurar la generación del código.

Firma

```
GenerateJavaCode() -> Void
```

Firma genérica

Errores

Código de error	Descripción
1200	El objeto ya no es válido.
1201	La dirección indicada para el parámetro de retorno no es válida.
1205	Error durante la generación del código.

16.7.1.9.2.9 GenerateOutput

Genera todos los archivos de salida definidos en la asignación usando para ello un lenguaje de asignación interno de MapForce. Los nombres de los archivos de salida están definidos como propiedades de los elementos de salida en la asignación.

Nota: este método solo se puede usar cuando la ventana principal de MapForce (como servidor COM) está visible o incrustada en una interfaz gráfica de usuario. Si se llama al método mientras MapForce no está visible se generará un error.

Firma

```
GenerateOutput() -> Void
```

Firma genérica

© 2016-2022 Altova GmbH

Errores

Código de error	Descripción
1200	El objeto ya no es válido.
1201	La dirección indicada para el parámetro de retorno no es válida.
1206	Error al ejecutar el algoritmo de la asignación.
1210	Solo se admite la generación de archivos de salida si la interfaz gráfica del usuario está visible.

16.7.1.9.2.10 GenerateOutputEx

Genera todos los archivos de salida definidos en la asignación usando para ello un lenguaje de asignación interno de MapForce. Los nombres de los archivos de salida están definidos como propiedades de los elementos de salida en la asignación. Este método es idéntico a `GenerateOutput`, salvo porque el valor que devuelve contiene los mensajes, las advertencias y los errores resultantes, organizados en estructuras en forma de árbol de `AppOutputLines`.

Nota: este método solo se puede usar cuando la ventana principal de MapForce (como servidor COM) está visible o incrustada en una interfaz gráfica de usuario. Si se llama al método mientras MapForce no está visible se generará un error.

Firma

```
GenerateOutputEx() -> AppOutputLines
```

Firma genérica

Errores

Código de error	Descripción
1200	El objeto ya no es válido.
1201	La dirección indicada para el parámetro de retorno no es válida.
1206	Error al ejecutar el algoritmo de la asignación.
1210	Solo se admite la generación de archivos de salida si la interfaz gráfica del usuario está visible.

16.7.1.9.2.11 GenerateXQuery

Genera código de asignación como XQuery. Usa las propiedades definidas en `Application.Options` para configurar la generación del código.

Firma

```
GenerateXQuery() -> Void
```

Firma genérica

Errores

Código de error	Descripción
1200	El objeto ya no es válido.
1201	La dirección indicada para el parámetro de retorno no es válida.
1204	Error al generar el código XSLT/XSLT2/XSLT3/XQuery.

16.7.1.9.2.12 GenerateXSLT

Genera código de asignación como XSLT. Usa las propiedades definidas en `Application.Options` para configurar la generación del código.

Firma

```
GenerateXSLT() -> Void
```

Firma genérica

Errores

Código de error	Descripción
1200	El objeto ya no es válido.
1201	La dirección indicada para el parámetro de retorno no es válida.
1204	Error al generar el código XSLT/XSLT2/XSLT3/XQuery.

© 2016-2022 Altova GmbH

16.7.1.9.2.13 GenerateXSLT2

Genera código de asignación como XSLT2. Usa las propiedades definidas en `Application.Options` para configurar la generación del código.

Firma

```
GenerateXSLT2() -> Void
```

Firma genérica

Errores

Código de error	Descripción
1200	El objeto ya no es válido.
1201	La dirección indicada para el parámetro de retorno no es válida.
1204	Error al generar el código XSLT/XSLT2/XSLT3/XQuery.

16.7.1.9.2.14 GenerateXSLT3

Genera código de asignación como XSLT 3.0. Usa las propiedades definidas en `Application.Options` para configurar la generación del código.

Firma

```
GenerateXSLT3() -> Void
```

Firma genérica

Errores

Código de error	Descripción
1200	El objeto ya no es válido.
1201	La dirección indicada para el parámetro de retorno no es válida.
1204	Error al generar el código XSLT/XSLT2/XSLT3/XQuery.

16.7.1.9.2.15 HighlightSerializedMarker

Use este método para resaltar la ubicación en un archivo de asignación previamente seriado. Primero se cargará el documento correspondiente si no lo ha hecho ya. En `GenerateCodeEx` se explica el método para obtener un indicador seriado.

Firma

```
HighlightSerializedMarker(in i_strSerializedMarker:String) -> Void
```

Firma genérica

Parámetros

Nombre	Tipo	Descripción
i_strSerializedMarker	`String`	El objeto `ErrorMarker` que se quiere resaltar. Use `ErrorMaker.Serialized` para obtener este valor.

Errores

Código de error	Descripción
1000	El objeto ya no es válido.
1001	La dirección indicada para el parámetro de retorno no es válida.
1007	La cadena enviada en `i_strSerializedMarker` no se reconoce como indicador seriado de MapForce.
1008	El indicador apunta a una ubicación que ya no es válida.

16.7.1.9.2.16 Save

Guarda el documento en el archivo definido por `Document.FullName`.

Firma

```
Save() -> Void
```

Firma genérica

© 2016-2022 Altova GmbH

Errores

Código de error	Descripción
1200	El objeto ya no es válido.
1201	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.9.2.17 SaveAs

Guarda el documento con el nombre de archivo indicado y establece `Document.FullName` en este valor si la operación se realiza con éxito.

Firma

```
SaveAs(in i_strFileName:String) -> Void
```

Firma genérica

Parámetros

Nombre	Tipo	Descripción
i_strFileName	`String`	Indica la ruta en la que se debe guardar el documento.

Errores

Código de error	Descripción
1200	El objeto ya no es válido.
1201	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.9.3 Eventos

16.7.1.9.3.1 OnDocumentClosed

Este evento se desencadena cuando se cierra un documento. No se debe llamar al objeto documento pasado al gestor de eventos. El evento de apertura correspondiente es `Application.OnDocumentOpened`.

Firma

```
OnDocumentClosed(in i_ipDocument:Document) : Void
```

16.7.1.9.3.2 OnModifiedFlagChanged

Este evento se desencadena cuando cambia el estado de un documento.

Firma

```
OnModifiedFlagChanged(in i_bIsModified:Boolean) : Void
```

16.7.1.10 Documents

Representa una colección de objetos `Document`.

Propiedades para navegar en el modelo del objeto:

- `Application`
- `Parent`

Abre y crea asignaciones:

- `OpenDocument`
- `NewDocument`

Se repiten en la colección:

- `Count`
- `Item`
- `ActiveDocument`

Propiedades

Nombre	Descripción
`ActiveDocument` 1331	De solo lectura. Obtiene el documento activo. Si no hay ningún documento abierto, el valor devuelto es `null`.
`Application` 1331	De solo lectura. Obtiene el objeto de nivel superior de la aplicación.
`Count` 1332	De solo lectura. Obtiene el número de documentos que hay en la colección.
`Item` 1332	De solo lectura. Obtiene el documento del índice `n` de la colección. El índice empieza en el 1.
`Parent` 1333	De solo lectura. El objeto de nivel superior según el modelo del objeto.

© 2016-2022 Altova GmbH

Métodos

Nombre	Descripción
NewDocument 1333	Crea un nuevo documento, lo agrega al final de la colección y lo convierte en el documento activo.
OpenDocument 1333	Abre un documento de asignación ya existente (*.mfd). Agrega el documento recién abierto al final de la colección y lo convierte en el documento activo.

16.7.1.10.1 Propiedades

16.7.1.10.1.1 ActiveDocument

Obtiene el documento activo. Si no hay ningún documento abierto, el valor devuelto es `null`.

Firma

```
ActiveDocument : Document
```

Firma genérica

Errores

Código de error	Descripción
1600	El objeto ya no es válido.
1601	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.10.1.2 Application

Obtiene el objeto de nivel superior de la aplicación.

Firma

```
Application : Application
```

Firma genérica

Errores

Código de error	Descripción
1600	El objeto ya no es válido.
1601	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.10.1.3 Count

Obtiene el número de documentos que hay en la colección.

Firma

```
Count : Integer
```

Firma genérica

Errores

Código de error	Descripción
1600	El objeto ya no es válido.
1601	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.10.1.4 Item

Obtiene el documento del índice n de la colección. El índice empieza en el 1.

Firma

```
Item(in n:Integer) : Document
```

Firma genérica

Errores

Código de error	Descripción
1600	El objeto ya no es válido.
1601	La dirección indicada para el parámetro de retorno no es válida.

© 2016-2022 Altova GmbH

16.7.1.10.1.5 Parent

El objeto de nivel superior según el modelo del objeto.

Firma

```
Parent : Application
```

Firma genérica

Errores

Código de error	Descripción
1600	El objeto ya no es válido.
1601	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.10.2 Métodos

16.7.1.10.2.1 NewDocument

Crea un nuevo documento, lo agrega al final de la colección y lo convierte en el documento activo.

Firma

```
NewDocument() -> Document
```

Firma genérica

Errores

Código de error	Descripción
1600	El objeto ya no es válido.
1601	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.10.2.2 OpenDocument

Abre un documento de asignación ya existente (*.mfd). Agrega el documento recién abierto al final de la colección y lo convierte en el documento activo.

Firma

```
OpenDocument(in strPath:String) -> Document
```

Firma genérica

Parámetros

Nombre	Tipo	Descripción
strPath	`String`	La ruta de acceso al archivo de asignación.

Errores

Código de error	Descripción
1600	El objeto ya no es válido.
1601	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.11 ErrorMarker

Representa una línea simple de mensaje. Al contrario que los objetos `AppOutputLine`, los marcadores de error no tienen una estructura jerárquica.

Propiedades para navegar en el modelo del objeto:

- `Application`
- `Parent`

Acceso a la información del mensaje:

- `DocumentFileName`
- `ErrorLevel`
- `Highlight`
- `Serialization`
- `Text`

Propiedades

Nombre	Descripción
`Application` 1335	De solo lectura. Obtiene el objeto de nivel superior de la aplicación.
`DocumentFileName` 1336	De solo lectura.

© 2016-2022 Altova GmbH

Nombre	Descripción
	Obtiene el nombre del archivo de asignación al que está asociado el marcador de error.
ErrorLevel 1336	De solo lectura. Obtiene la severidad del error.
Parent 1337	De solo lectura. El objeto de nivel superior según el modelo del objeto.
Serialization 1337	De solo lectura. Seria el marcador de error en una cadena. Usa esta cadena en llamadas a `Application.HighlightSerializedMarker` o a `Document.HighlightSerializedMarker` para resaltar en la asignación el elemento marcador. La cadena se puede volver a usar en otras instancias de MapForce o de su control.
Text 1338	De solo lectura. Obtiene el texto del mensaje.

Métodos

Nombre	Descripción
Highlight 1338	Resalta el elemento al que está asociado el marcador de error. Primero se abrirá el documento correspondiente si no lo está ya.

16.7.1.11.1 Propiedades

16.7.1.11.1.1 Application

Obtiene el objeto de nivel superior de la aplicación.

Firma

```
Application : Application
```

Firma genérica

Errores

Código de error	Descripción
1900	El objeto ya no es válido.

Código de error	Descripción
1901	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.11.1.2 DocumentFileName

Obtiene el nombre del archivo de asignación al que está asociado el marcador de error.

Firma

```
DocumentFileName : String
```

Firma genérica

Errores

Código de error	Descripción
1900	El objeto ya no es válido.
1901	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.11.1.3 ErrorLevel

Obtiene la severidad del error.

Firma

```
ErrorLevel : ENUMCodeGenErrorLevel 1414
```

Firma genérica

Errores

Código de error	Descripción
1900	El objeto ya no es válido.
1901	La dirección indicada para el parámetro de retorno no es válida.

© 2016-2022 Altova GmbH

16.7.1.11.1.4 Parent

El objeto de nivel superior según el modelo del objeto.

Firma

```
Parent : ErrorMarkers
```

Firma genérica

Errores

Código de error	Descripción
1900	El objeto ya no es válido.
1901	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.11.1.5 Serialization

Seria el marcador de error en una cadena. Usa esta cadena en llamadas a `Application.HighlightSerializedMarker` o a `Document.HighlightSerializedMarker` para resaltar en la asignación el elemento marcador. La cadena se puede volver a usar en otras instancias de MapForce o de su control.

Firma

```
Serialization : String
```

Firma genérica

Errores

Código de error	Descripción
1900	El objeto ya no es válido.
1901	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.11.1.6 Text

Obtiene el texto del mensaje.

Firma

```
Text : String
```

Firma genérica

Errores

Código de error	Descripción
1900	El objeto ya no es válido.
1901	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.11.2 Métodos

16.7.1.11.2.1 Highlight

Resalta el elemento al que está asociado el marcador de error. Primero se abrirá el documento correspondiente si no lo está ya.

Firma

```
Highlight() -> Void
```

Firma genérica

Errores

Código de error	Descripción
1900	El objeto ya no es válido.
1901	La dirección indicada para el parámetro de retorno no es válida.
1008	El indicador apunta a una ubicación que ya no es válida.

© 2016-2022 Altova GmbH

16.7.1.12 ErrorMarkers

Representa una colección de objetos `ErrorMarker`.

Propiedades para navegar en el modelo del objeto:

- `Application`
- `Parent`

Se repiten en la colección:

- `Count`
- `Item`

Propiedades

Nombre	Descripción
`Application` 1339	De solo lectura. Obtiene el objeto de nivel superior de la aplicación.
`Count` 1340	De solo lectura. Obtiene el número de marcadores de error que hay en la colección.
`Item` 1340	De solo lectura. Obtiene el marcador de error del índice `n` de la colección. El índice empieza en el 1.
`Parent` 1341	De solo lectura. El objeto de nivel superior según el modelo del objeto.

16.7.1.12.1 Propiedades

16.7.1.12.1.1 Application

Obtiene el objeto de nivel superior de la aplicación.

Firma

```
Application : Application
```

Firma genérica

Errores

Código de error	Descripción
1800	El objeto ya no es válido.
1801	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.12.1.2 Count

Obtiene el número de marcadores de error que hay en la colección.

Firma

```
Count : Integer
```

Firma genérica

Errores

Código de error	Descripción
1800	El objeto ya no es válido.
1801	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.12.1.3 Item

Obtiene el marcador de error del índice `n` de la colección. El índice empieza en el 1.

Firma

```
Item(in n:Integer) : ErrorMarker
```

Firma genérica

Errores

Código de error	Descripción
1800	El objeto ya no es válido.
1801	La dirección indicada para el parámetro de retorno no es válida.

© 2016-2022 Altova GmbH

16.7.1.12.1.4 Parent

El objeto de nivel superior según el modelo del objeto.

Firma

```
Parent : Application
```

Firma genérica

Errores

Código de error	Descripción
1800	El objeto ya no es válido.
1801	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.13 LibraryImport

Un `LibraryImport` representa un archivo de biblioteca importado (una entrada de la ventana **Gestionar bibliotecas**).

Propiedades

Nombre	Descripción
`Application` 1342	De solo lectura. Obtiene el objeto de nivel superior de la aplicación.
`Parent` 1342	De solo lectura. Obtiene el objeto de nivel superior según el modelo del objeto.
`Path` 1343	De solo lectura. Obtiene la ruta de la biblioteca importada.
`SaveRelativePath` 1343	Al guardar el documento esta propiedad especifica si la ruta de la biblioteca debería guardarse como absoluta o como relativa. Si se establece en **true**, la ruta de la biblioteca será relativa al documento. Si se establece en false, la ruta de la biblioteca será absoluta. No recomendamos que use esta propiedad para averiguar si una ruta es absoluta o relativa porque esa ruta puede haber cambiado (sea a través de la interfaz del usuario o mediante la API) desde que se cargó el documento desde el archivo .mfd.

Nombre	Descripción
	Si establece esta propiedad (a través de la interfaz del usuario o con la API), la ventana "Gestionar bibliotecas" muestra de inmediato el estado correcto de la ruta en la interfaz del usuario. De forma interna, sin embargo, la ruta`Path` del objeto de biblioteca importada `ImportedLibrary` no cambiará hasta que no se guarde el documento. Las bibliotecas importadas globalmente no se pueden guardar con una ruta relativa. Eso solo puede hacerlo con las bibliotecas importadas a nivel del documento.

16.7.1.13.1 Propiedades

16.7.1.13.1.1 Application

Obtiene el objeto de nivel superior de la aplicación.

Firma

```
Application : Application
```

Firma genérica

Errores

Código de error	Descripción
2500	El objeto ya no es válido.
2501	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.13.1.2 Parent

Obtiene el objeto de nivel superior según el modelo del objeto.

Firma

```
Parent : LibraryImports
```

Firma genérica

© 2016-2022 Altova GmbH

Errores

Código de error	Descripción
2500	El objeto ya no es válido.
2501	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.13.1.3 Path

Obtiene la ruta de la biblioteca importada.

Firma

```
Path : String
```

Firma genérica

Errores

Código de error	Descripción
2500	El objeto ya no es válido.
2501	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.13.1.4 SaveRelativePath

Al guardar el documento esta propiedad especifica si la ruta de la biblioteca debería guardarse como absoluta o como relativa. Si se establece en **true**, la ruta de la biblioteca será relativa al documento. Si se establece en false, la ruta de la biblioteca será absoluta.

No recomendamos que use esta propiedad para averiguar si una ruta es absoluta o relativa porque esa ruta puede haber cambiado (sea a través de la interfaz del usuario o mediante la API) desde que se cargó el documento desde el archivo .mfd.

Si establece esta propiedad (a través de la interfaz del usuario o con la API), la ventana "Gestionar bibliotecas" muestra de inmediato el estado correcto de la ruta en la interfaz del usuario. De forma interna, sin embargo, la ruta`Path` del objeto de biblioteca importada `ImportedLibrary` no cambiará hasta que no se guarde el documento.

> Las bibliotecas importadas globalmente no se pueden guardar con una ruta relativa. Eso solo puede hacerlo con las bibliotecas importadas a nivel del documento.

Firma

```
SaveRelativePath : Boolean
```

Firma genérica

Errores

Código de error	Descripción
2500	El objeto ya no es válido.
2501	La dirección indicada para el parámetro de retorno no es válida.
2502	Las importaciones globales no se pueden guardar con una ruta relativa.

16.7.1.14 LibraryImports

Representa una colección de bibliotecas importadas (objetos `LibraryImport`). Usar las propiedades `Application` y `Parent` para navegar en el modelo del objeto. Usar las propiedades `Count` e `Item` para repetir en la colección. Puede obtener esta colección de dos maneras:

- De forma local (a nivel del documento) con la propiedad `Document.LibraryImports` o
- De forma global (a nivel de la aplicación) con la propiedad `Application.LibraryImports`.

> Si obtiene la colección `LibraryImports` desde el objeto de la aplicación, la propiedad `Parent` de la colección será null.

Propiedades

Nombre	Descripción
`Application` 1345	De solo lectura. Obtiene el objeto de nivel superior de la aplicación.
`Count` 1346	De solo lectura. Obtiene la suma de todos los objetos `LibraryImport` en esta colección.
`Item` 1346	De solo lectura. Obtiene una entrada de biblioteca del índice *n* de la colección. El índice es de base uno.
`Parent` 1346	De solo lectura. Obtiene el documento de nivel superior para las bibliotecas importadas a nivel local. Si obtiene la colección

© 2016-2022 Altova GmbH

Nombre	Descripción
	`LibraryImports` desde el objeto de la aplicación, la propiedad `Parent` de la colección será null.

Métodos

Nombre	Descripción
Add 1347	Añade una biblioteca nueva a este objeto `LibraryImports`. La biblioteca nueva tendrá la ruta indicada por el parámetro **i_strFileName**.
Find 1348	Devuelve una referencia de biblioteca en función de la ruta de acceso al archivo de la biblioteca.
Remove 1349	Elimina una referencia a una biblioteca de la ventana **Gestionar bibliotecas**

16.7.1.14.1 Propiedades

16.7.1.14.1.1 Application

Obtiene el objeto de nivel superior de la aplicación.

Firma

```
Application : Application
```

Firma genérica

Errores

Código de error	Descripción
2400	El objeto ya no es válido.
2401	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.14.1.2 Count

Obtiene la suma de todos los objetos `LibraryImport` en esta colección.

Firma

```
Count : Integer
```

Firma genérica

Errores

Código de error	Descripción
2400	El objeto ya no es válido.
2401	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.14.1.3 Item

Obtiene una entrada de biblioteca del índice *n* de la colección. El índice es de base uno.

Firma

```
Item(in n:Integer) : LibraryImport
```

Firma genérica

Errores

Código de error	Descripción
2400	El objeto ya no es válido.
2401	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.14.1.4 Parent

Obtiene el documento de nivel superior para las bibliotecas importadas a nivel local. Si obtiene la colección `LibraryImports` desde el objeto de la aplicación, la propiedad `Parent` de la colección será null.

Firma

```
Parent : Document
```


© 2016-2022 Altova GmbH

Firma genérica

Errores

Código de error	Descripción
2400	El objeto ya no es válido.
2401	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.14.2 Métodos

16.7.1.14.2.1 Add

Añade una biblioteca nueva a este objeto `LibraryImports`. La biblioteca nueva tendrá la ruta indicada por el parámetro **i_strFileName**.

Firma

```
Add(in i_strFileName:String) -> LibraryImport
```

Firma genérica

Parámetros

Nombre	Tipo	Descripción
i_strFileName	`String`	Indica la ruta de la biblioteca que se quiere añadir. Esta parte puede ser absoluta o relativa a la asignación en función del estado en el que se pasó al objeto. Cuando se guarda el documento, la ruta se convierte en relativa si el marcador `LibraryImport.SaveRelativePath` se establece en **true**; de lo contrario la ruta es absoluta.

Errores

Código de error	Descripción
2400	El objeto ya no es válido.

Código de error	Descripción
2401	La dirección indicada para el parámetro de retorno no es válida.
2402	Error al añadir la biblioteca.

16.7.1.14.2.2 Find

Devuelve una referencia de biblioteca en función de la ruta de acceso al archivo de la biblioteca.

Firma

```
Find(in i_strFileName:String) -> LibraryImport
```

Firma genérica

Parámetros

Nombre	Tipo	Descripción
i_strFileName	`String`	La ruta del archivo de biblioteca que hay que buscar. En el caso de las bibliotecas importadas de forma local, puede indicar la ruta de acceso al archivo de biblioteca como absoluta o como relativa (al contrario que con el método `Remove`, que necesita que indique la ruta exacta). En el caso de las bibliotecas importadas globalmente, la ruta siempre debe ser absoluta y no puede ser relativa.

Errores

Código de error	Descripción
2400	El objeto ya no es válido.
2401	La dirección indicada para el parámetro de retorno no es válida.

© 2016-2022 Altova GmbH

16.7.1.14.2.3 Remove

Elimina una referencia a una biblioteca de la ventana **Gestionar bibliotecas**

Firma

```
Remove(in i_strFileName:String) -> Void
```

Firma genérica

Parámetros

Nombre	Tipo	Descripción
i_strFileName	`String`	La ruta del archivo de biblioteca que hay que eliminar. Observe que la ruta debe reflejar el estado actual (más reciente) del objeto `LibraryImport`. Recuerde que la ruta puede ser relativa o absoluta y que puede haber cambiado si guardó el documento en función del marcados `LibraryImport.SaveRelativePath`. Por tanto, si el objeto `LibraryImport` contiene actualmente una ruta relativa, entonces también debería indicar una ruta relativa como valor del parámetro. De lo contrario, el método `Remove` porque no encuentra la biblioteca. Esto solo afecta a las bibliotecas importadas localmente. En el caso de las bibliotecas importadas globalmente, la ruta siempre debe ser absoluta y no puede ser relativa.

Errores

Código de error	Descripción
2400	El objeto ya no es válido.

16.7.1.15 MapForceView

Representa la vista activa de un documento en la pestaña Asignación de MapForce. Un documento tiene una sola vista de MapForce (`MapForceView`) que muestra la asignación activa en ese momento.

Propiedades para navegar en el modelo del objeto:

- `Application`
- `Parent`

Ver propiedades de activación y de vista:

- `Active`
- `ShowItemTypes`
- `ShowLibraryInFunctionHeader`
- `HighlightMyConnections`
- `HighlightMyConnectionsRecursivly`

Propiedades relativas a la asignación

- `ActiveMapping`
- `ActiveMappingName`

Agregar elementos:

- `InsertWSDLCall`
- `InsertXMLFile`
- `InsertXMLSchema`
- `InsertXMLSchemaWithSample`

Propiedades

Nombre	Descripción
`Active` 1351	Use esta propiedad para consultar si la vista de la asignación es la vista activa o para establecer esta vista como vista activa.
`ActiveMapping` 1352	Obtiene o indica la asignación activa en ese momento en el documento al que pertenece esta vista de MapForce (`MapForceView`).
`ActiveMappingName` 1352	Obtiene por nombre la asignación activa en ese momento en el documento al que pertenece esta vista de MapForce (MapForceView) o la indica.
`Application` 1353	De solo lectura. Obtiene el objeto de nivel superior de la aplicación.
`HighlightMyConnections` 1353	Esta propiedad indica si solo se deben resaltar las conexiones del elemento seleccionado.

© 2016-2022 Altova GmbH

Nombre	Descripción
HighlightMyConnectionsRecursively 1353	Esta propiedad indica si solo se deben resaltar las conexiones que provienen directa o indirectamente del elemento seleccionado.
Parent 1354	De solo lectura. El objeto de nivel superior según el modelo del objeto.
ShowItemTypes 1354	Esta propiedad indica si se deben mostrar los tipos de los elementos en el diagrama de la asignación.
ShowLibraryInFunctionHeader 1355	Esta propiedad indica si el nombre de la biblioteca de funciones debe formar parte de los nombres de las funciones.

Métodos

Nombre	Descripción
InsertWSDLCall 1355	Añade un nuevo componente de llamada WSDL a la asignación.
InsertXMLFile 1356	`MapForceView.InsertXMLFile` está obsoleto. Use `Mapping.InsertXMLFile` en su lugar.
InsertXMLSchema 1356	`MapForceView.InsertXMLSchema` está obsoleto. Use `Mapping.InsertXMLSchema` en su lugar.
InsertXMLSchemaWithSample 1357	`MapForceView.InsertXMLSchemaWithSample` está obsoleto. Use `Mapping.InsertXMLFile` en su lugar. Tenga en cuenta que `Mapping.InsertXMLFile` no requiere un parámetro para enviar el elemento raíz. El elemento raíz se indica automáticamente como nombre de elemento raíz del archivo XML.

16.7.1.15.1 Propiedades

16.7.1.15.1.1 Active

Use esta propiedad para consultar si la vista de la asignación es la vista activa o para establecer esta vista como vista activa.

Firma

```
Active : Boolean
```

Firma genérica

Errores

Código de error	Descripción
1300	El objeto ya no es válido.
1301	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.15.1.2 ActiveMapping

Obtiene o indica la asignación activa en ese momento en el documento al que pertenece esta vista de MapForce (`MapForceView`).

Firma

```
ActiveMapping : Mapping
```

Firma genérica

Errores

Código de error	Descripción
1300	El objeto ya no es válido.
1301	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.15.1.3 ActiveMappingName

Obtiene por nombre la asignación activa en ese momento en el documento al que pertenece esta vista de MapForce (MapForceView) o la indica.

Firma

```
ActiveMappingName : String
```

Firma genérica

Errores

Código de error	Descripción
1300	El objeto ya no es válido.
1301	La dirección indicada para el parámetro de retorno no es válida.

© 2016-2022 Altova GmbH

16.7.1.15.1.4 Application

Obtiene el objeto de nivel superior de la aplicación.

Firma

```
Application : Application
```

Firma genérica

Errores

Código de error	Descripción
1300	El objeto ya no es válido.
1301	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.15.1.5 HighlightMyConnections

Esta propiedad indica si solo se deben resaltar las conexiones del elemento seleccionado.

Firma

```
HighlightMyConnections : Boolean
```

Firma genérica

Errores

Código de error	Descripción
1300	El objeto ya no es válido.
1301	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.15.1.6 HighlightMyConnectionsRecursively

Esta propiedad indica si solo se deben resaltar las conexiones que provienen directa o indirectamente del elemento seleccionado.

Firma

```
HighlightMyConnectionsRecursively : Boolean
```

Firma genérica

Errores

Código de error	Descripción
1300	El objeto ya no es válido.
1301	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.15.1.7 Parent

El objeto de nivel superior según el modelo del objeto.

Firma

```
Parent : Document
```

Firma genérica

Errores

Código de error	Descripción
1300	El objeto ya no es válido.
1301	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.15.1.8 ShowItemTypes

Esta propiedad indica si se deben mostrar los tipos de los elementos en el diagrama de la asignación.

Firma

```
ShowItemTypes : Boolean
```

Firma genérica

Errores

Código de error	Descripción
1300	El objeto ya no es válido.

© 2016-2022 Altova GmbH

Código de error	Descripción
1301	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.15.1.9 ShowLibraryInFunctionHeader

Esta propiedad indica si el nombre de la biblioteca de funciones debe formar parte de los nombres de las funciones.

Firma

```
ShowLibraryInFunctionHeader : Boolean
```

Firma genérica

Errores

Código de error	Descripción
1300	El objeto ya no es válido.
1301	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.15.2 Métodos

16.7.1.15.2.1 InsertWSDLCall

Añade un nuevo componente de llamada WSDL a la asignación.

Firma

```
InsertWSDLCall(in i_strWSDLFileName:String) -> Void
```

Firma genérica

Parámetros

Nombre	Tipo	Descripción
i_strWSDLFileName	`String`	Indica la ruta de acceso al archivo WSDL que se va a añadir a la asignación.

Errores

Código de error	Descripción
1300	El objeto ya no es válido.
1301	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.15.2.2 InsertXMLFile (obsolete)

`MapForceView.InsertXMLFile` está obsoleto. Use `Mapping.InsertXMLFile` en su lugar.

Firma

```
InsertXMLFile(in i_strFileName:String, in i_strXMLRootName:String) -> Void
```

Firma genérica

Parámetros

Nombre	Tipo	Descripción
i_strFileName	`String`	
i_strXMLRootName	`String`	

16.7.1.15.2.3 InsertXMLSchema (obsolete)

`MapForceView.InsertXMLSchema` está obsoleto. Use `Mapping.InsertXMLSchema` en su lugar.

Firma

```
InsertXMLSchema(in i_strSchemaFileName:String, in i_strXMLRootName:String) -> Void
```

Firma genérica

Parámetros

Nombre	Tipo	Descripción
i_strSchemaFileName	`String`	
i_strXMLRootName	`String`	

© 2016-2022 Altova GmbH

16.7.1.15.2.4 InsertXMLSchemaWithSample (obsolete)

`MapForceView.InsertXMLSchemaWithSample` está obsoleto. Use `Mapping.InsertXMLFile` en su lugar. Tenga en cuenta que `Mapping.InsertXMLFile` no requiere un parámetro para enviar el elemento raíz. El elemento raíz se indica automáticamente como nombre de elemento raíz del archivo XML.

Firma

```
InsertXMLSchemaWithSample(in i_strSchemaFileName:String, in i_strXMLExampleFile:String,
in i_strXMLRootName:String) -> Void
```

Firma genérica

Parámetros

Nombre	Tipo	Descripción
i_strSchemaFileName	`String`	
i_strXMLExampleFile	`String`	
i_strXMLRootName	`String`	

16.7.1.16 Mapping

Un objeto `Mapping` representa una asignación en un documento, sea la asignación principal o una asignación local de funciones definidas por el usuario.

Propiedades para navegar en el modelo del objeto:

- `Application`
- `Parent`

Propiedades de la asignación:

- `IsMainMapping`
- `Name`

Componentes de la asignación:

- `Components`

Añadir elementos:

- `CreateConnection`
- `InsertFunctionCall`
- `InsertXMLFile`
- `InsertXMLSchema`
- `InsertXMLSchemaInputParameter`

- `InsertXMLSchemaOutputParameter`

Propiedades

Nombre	Descripción
`Application` 1360	De solo lectura. Obtiene el objeto de nivel superior de la aplicación.
`Components` 1360	De solo lectura. Devuelve una colección de todos los componentes en la asignación actual.
`IsMainMapping` 1360	De solo lectura. Indica si la asignación actual es la asignación principal del documento en el que esta se encuentra. **True** significa que es la asignación principal. **False** significa que es una función definida por el usuario (UDF por sus siglas en inglés).
`Name` 1361	De solo lectura. El nombre de la asignación o de la función definida por el usuario (UDF).
`Parent` 1361	De solo lectura. El objeto de nivel superior según el modelo del objeto.

Métodos

Nombre	Descripción
`CreateConnection` 1362	Crea una conexión entre los dos puntos de datos dados (`DatapointFrom` y `DatapointTo`). Esta conexión no se puede crear si `DatapointFrom` no es un punto de datos en el lado de salida, si `DatapointTo` no es un punto de datos en el lado de entrada o si ya existe una conexión entre estos dos puntos de datos.
`InsertFunctionCall` 1363	Inserta un componente de llamada a una función en la asignación actual. Los nombres de biblioteca y función especificados indican la función o función definida por el usuario a la que hay que llamar.
`InsertXMLFile` 1363	Añade un nuevo componente de esquema XML a la asignación. La estructura interna del componente la determina el esquema al que se hace referencia en el archivo XML indicado (`i_strFileName`) o, si el archivo XML no hace referencia a un

 © 2016-2022 Altova GmbH

Nombre	Descripción
	archivo de esquema, un archivo de esquema especificado por separado (`i_strSchemaFileName`). Si el archivo XML hace referencia a un archivo de esquema, entonces se ignora el parámetro `i_strSchemaFileName`. El elemento raíz del archivo XML se usará en el componente. El archivo XML especificado se usa como archivo de entrada de muestra para evaluar la asignación.
`InsertXMLSchema` 1364	Añade un nuevo componente de esquema XML a la asignación. La estructura interna del componente la determina el archivo de esquema indicado en el primer parámetro. El segundo parámetro define el elemento raíz de este esquema si hay más de un candidato. Si el elemento raíz dado es una cadena vacía y hay más candidatos disponibles aparecerá un cuadro de diálogo emergente **Seleccione un elemento raíz** si MapForce es visible. Si MapForce no es visible no aparecerá ningún cuadro de diálogo y solo se devolverá un error. No hay ningún archivo XML de entrada de muestra asignado a este componente.
`InsertXMLSchemaInputParameter` 1365	Inserta un componente de parámetro de entrada de esquema XML en la asignación actual. La asignación actual tiene que ser una función definida por el usuario. Intentar insertarlo (el parámetro de entrada del esquema) en la asignación principal generará un error.
`InsertXMLSchemaOutputParameter` 1366	Inserta un componente de parámetro de salida de esquema XML en la asignación actual. La asignación actual tiene que ser una función definida por el usuario. Intentar insertarlo (el parámetro de salida del esquema) en la asignación principal generará un error.

16.7.1.16.1 Propiedades

16.7.1.16.1.1 Application

Obtiene el objeto de nivel superior de la aplicación.

Firma

```
Application : Application
```

Firma genérica

Errores

Código de error	Descripción
1200	El objeto ya no es válido.
1201	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.16.1.2 Components

Devuelve una colección de todos los componentes en la asignación actual.

Firma

```
Components : Components
```

Firma genérica

Errores

Código de error	Descripción
1200	El objeto ya no es válido.
1201	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.16.1.3 IsMainMapping

Indica si la asignación actual es la asignación principal del documento en el que esta se encuentra.

True significa que es la asignación principal.

 © 2016-2022 Altova GmbH

False significa que es una función definida por el usuario (UDF por sus siglas en inglés).

Firma

```
IsMainMapping : Boolean
```

Firma genérica

Errores

Código de error	Descripción
1200	El objeto ya no es válido.
1201	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.16.1.4 Name

El nombre de la asignación o de la función definida por el usuario (UDF).

Firma

```
Name : String
```

Firma genérica

Errores

Código de error	Descripción
1200	El objeto ya no es válido.
1201	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.16.1.5 Parent

El objeto de nivel superior según el modelo del objeto.

Firma

```
Parent : Document
```

Firma genérica

Errores

Código de error	Descripción
1200	El objeto ya no es válido.
1201	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.16.2 Métodos

16.7.1.16.2.1 CreateConnection

Crea una conexión entre los dos puntos de datos dados (`DatapointFrom` y `DatapointTo`).

Esta conexión no se puede crear si `DatapointFrom` no es un punto de datos en el lado de salida, si `DatapointTo` no es un punto de datos en el lado de entrada o si ya existe una conexión entre estos dos puntos de datos.

Firma

```
CreateConnection(in DatapointFrom:Datapoint, in DatapointTo:Datapoint) -> Connection
```

Firma genérica

Parámetros

Nombre	Tipo	Descripción
DatapointFrom	`Datapoint`	El punto de datos a partir del cual se debe crear la conexión.
DatapointTo	`Datapoint`	El punto de datos de destino.

Errores

Código de error	Descripción
1200	El objeto ya no es válido.
1201	La dirección indicada para el parámetro de retorno no es válida.
1240	No está permitido cambiar el documento. Es solo de lectura.
1241	Error al crear la conexión.

© 2016-2022 Altova GmbH

16.7.1.16.2.2 InsertFunctionCall

Inserta un componente de llamada a una función en la asignación actual.

Los nombres de biblioteca y función especificados indican la función o función definida por el usuario a la que hay que llamar.

Firma

```
InsertFunctionCall(in strFunctionName:String, in strLibraryName:String) -> Component
```

Firma genérica

Parámetros

Nombre	Tipo	Descripción
strFunctionName	`String`	El nombre de la función que se debe insertar.
strLibraryName	`String`	El nombre de la biblioteca que se debe insertar.

Errores

Código de error	Descripción
1200	El objeto ya no es válido.
1201	La dirección indicada para el parámetro de retorno no es válida.
1240	No está permitido cambiar el documento. Es solo de lectura.
1242	Error al crear el componente de llamada a una función.

16.7.1.16.2.3 InsertXMLFile

Añade un nuevo componente de esquema XML a la asignación.

La estructura interna del componente la determina el esquema al que se hace referencia en el archivo XML indicado (`i_strFileName`) o, si el archivo XML no hace referencia a un archivo de esquema, un archivo de esquema especificado por separado (`i_strSchemaFileName`).

Si el archivo XML hace referencia a un archivo de esquema, entonces se ignora el parámetro `i_strSchemaFileName`.

El elemento raíz del archivo XML se usará en el componente.

El archivo XML especificado se usa como archivo de entrada de muestra para evaluar la asignación.

Firma

```
InsertXMLFile(in i_strFileName:String, in i_strSchemaFileName:String) -> Component
```

Firma genérica

Parámetros

Nombre	Tipo	Descripción
i_strFileName	`String`	La ruta del archivo XML de instancia que se quiere añadir.
i_strSchemaFileName	`String`	La ruta del archivo de definición de esquema XML que se quiere añadir.

Errores

Código de error	Descripción
1200	El objeto ya no es válido.
1201	La dirección indicada para el parámetro de retorno no es válida.
1240	No está permitido cambiar el documento. Es solo de lectura.
1244	Error al crear el componente.

16.7.1.16.2.4 InsertXMLSchema

Añade un nuevo componente de esquema XML a la asignación.

La estructura interna del componente la determina el archivo de esquema indicado en el primer parámetro.

El segundo parámetro define el elemento raíz de este esquema si hay más de un candidato.

Si el elemento raíz dado es una cadena vacía y hay más candidatos disponibles aparecerá un cuadro de diálogo emergente **Seleccione un elemento raíz** si MapForce es visible. Si MapForce no es visible no aparecerá ningún cuadro de diálogo y solo se devolverá un error.

No hay ningún archivo XML de entrada de muestra asignado a este componente.

Firma

```
InsertXMLSchema(in i_strSchemaFileName:String, in i_strXMLRootName:String) -> Component
```

 © 2016-2022 Altova GmbH

Firma genérica

Parámetros

Nombre	Tipo	Descripción
i_strSchemaFileName	`String`	La ruta del archivo de definición de esquema XML que se quiere añadir.
i_strXMLRootName	`String`	El elemento raíz del esquema (aplicable cuando el esquema tiene más de un elemento raíz).

Errores

Código de error	Descripción
1200	El objeto ya no es válido.
1201	La dirección indicada para el parámetro de retorno no es válida.
1240	No está permitido cambiar el documento. Es solo de lectura.
1244	Error al crear el componente.

16.7.1.16.2.5 InsertXMLSchemaInputParameter

Inserta un componente de parámetro de entrada de esquema XML en la asignación actual.

La asignación actual tiene que ser una función definida por el usuario. Intentar insertarlo (el parámetro de entrada del esquema) en la asignación principal generará un error.

Firma

```
InsertXMLSchemaInputParameter(in strParamName:String, in strSchemaFileName:String, in
strXMLRootElementName:String) -> Component
```

Firma genérica

Parámetros

Nombre	Tipo	Descripción
strParamName	`String`	El nombre del componente de parámetro de entrada

Nombre	Tipo	Descripción
strSchemaFileName	`String`	La ruta del archivo de definición de esquema XML que se quiere añadir.
strXMLRootElementName	`String`	El elemento raíz del esquema (aplicable cuando el esquema tiene más de un elemento raíz). Si el elemento raíz dado es una cadena vacía y hay más candidatos disponibles aparecerá un cuadro de diálogo emergente **Seleccione un elemento raíz** si MapForce es visible. Si MapForce no es visible no aparecerá ningún cuadro de diálogo y solo se devolverá un error.

Errores

Código de error	Descripción
1200	El objeto ya no es válido.
1201	La dirección indicada para el parámetro de retorno no es válida.
1240	No está permitido cambiar el documento. Es solo de lectura.
1243	Error al crear el componente de parámetro.
1245	No se admite esta operación en la asignación principal.

16.7.1.16.2.6 InsertXMLSchemaOutputParameter

Inserta un componente de parámetro de salida de esquema XML en la asignación actual.

La asignación actual tiene que ser una función definida por el usuario. Intentar insertarlo (el parámetro de salida del esquema) en la asignación principal generará un error.

Firma

```
InsertXMLSchemaOutputParameter(in strParamName:String, in strSchemaFileName:String, in
strXMLRootElementName:String) -> Component
```

Firma genérica

 © 2016-2022 Altova GmbH

Parámetros

Nombre	Tipo	Descripción
strParamName	`String`	El nombre del componente de parámetro de salida que se debe crear.
strSchemaFileName	`String`	La ruta del archivo de definición de esquema XML que se quiere añadir.
strXMLRootElementName	`String`	El elemento raíz del esquema (aplicable cuando el esquema tiene más de un elemento raíz). Si el elemento raíz dado es una cadena vacía y hay más candidatos disponibles aparecerá un cuadro de diálogo emergente **Seleccione un elemento raíz** si MapForce es visible. Si MapForce no es visible no aparecerá ningún cuadro de diálogo y solo se devolverá un error.

Errores

Código de error	Descripción
1200	El objeto ya no es válido.
1201	La dirección indicada para el parámetro de retorno no es válida.
1240	No está permitido cambiar el documento. Es solo de lectura.
1243	Error al crear el componente de parámetro.
1245	No se admite esta operación en la asignación principal.

16.7.1.17 Mappings

Representa una colección de objetos `Mapping`.

Propiedades para navegar en el modelo del objeto:

- `Application`
- `Parent`

Se repiten en la colección:

- Count
- Item

Propiedades

Nombre	Descripción
Application 1368	De solo lectura. Obtiene el objeto de nivel superior de la aplicación.
Count 1369	De solo lectura. Obtiene el número de asignaciones que hay en la colección.
Item 1369	De solo lectura. Obtiene la asignación del índice n de la colección. El índice empieza en el 1.
Parent 1370	De solo lectura. El objeto de nivel superior según el modelo del objeto.

16.7.1.17.1 Propiedades

16.7.1.17.1.1 Application

Obtiene el objeto de nivel superior de la aplicación.

Firma

```
Application : Application
```

Firma genérica

Errores

Código de error	Descripción
1200	El objeto ya no es válido.
1201	La dirección indicada para el parámetro de retorno no es válida.

© 2016-2022 Altova GmbH

16.7.1.17.1.2 Count

Obtiene el número de asignaciones que hay en la colección.

Firma

```
Count : Integer
```

Firma genérica

Errores

Código de error	Descripción
1200	El objeto ya no es válido.
1201	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.17.1.3 Item

Obtiene la asignación del índice n de la colección. El índice empieza en el 1.

Firma

```
Item(in n:Integer) : Mapping
```

Firma genérica

Errores

Código de error	Descripción
1200	El objeto ya no es válido.
1201	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.17.1.4 Parent

El objeto de nivel superior según el modelo del objeto.

Firma

```
Parent : Document
```

Firma genérica

Errores

Código de error	Descripción
1200	El objeto ya no es válido.
1201	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.18 Options

Este objeto da acceso a todas las opciones de MapForce disponibles en el cuadro de diálogo **Herramientas | Opciones**.

Propiedades para navegar en el modelo del objeto:

- `Application`
- `Parent`

Opciones generales:

- `ShowLogoOnPrint`
- `ShowLogoOnStartup`
- `UseGradientBackground`

Opciones de generación de código:

- `DefaultOutputEncoding`
- `DefaultOutputByteOrder`
- `DefaultOutputByteOrderMark`
- `XSLTDefaultOutputDirectory`
- `CodeDefaultOutputDirectory`
- `CPPSettings_DOMType`
- `CPPSettings_GenerateVC6ProjectFile`
- `CppSettings_GenerateVSProjectFile`
- `CPPSettings_LibraryType`
- `CPPSettings_UseMFC`
- `CSharpSettings_ProjectType`

 © 2016-2022 Altova GmbH

Propiedades

Nombre	Descripción
Application 1372	De solo lectura. Obtiene el objeto de nivel superior de la aplicación.
CodeDefaultOutputDirectory 1372	Indica el directorio de destino en el que se encuentran los archivos generados por `Document.GenerateCppCode`, `Document.GenerateJavaCode` y `Document.GenerateCHashCode`.
CPPSettings_DOMType 1373	Indica el tipo DOM que usa `Document.GenerateCppCode`.
CPPSettings_GenerateVC6ProjectFile 1373	Indica si `Document.GenerateCppCode` debe generar archivos de proyecto de VisualC++ 6.0.
CppSettings_GenerateVSProjectFile 1374	Indica en qué versión de Visual Studio debe generar `Document.GenerateCppCode` los archivos de proyecto.
CPPSettings_LibraryType 1374	Indica el tipo de biblioteca que usa `Document.GenerateCppCode`.
CPPSettings_UseMFC 1375	Indica si el código C++ generado por `Document.GenerateCppCode` debe usar el soporte de MFC.
CSharpSettings_ProjectType 1375	Indica el tipo de proyecto C# que usa `Document.GenerateCHashCode`.
DefaultOutputByteOrder 1376	Orden de bytes para la codificación usada para los archivos de salida.
DefaultOutputByteOrderMark 1376	Indica si se debe incluir una marca BOM (marca de orden de bytes) en la codificación del archivo de los archivos de salida.
DefaultOutputEncoding 1377	Codificación usada para los archivos de salida.
GenerateWrapperClasses 1377	Indica si también deben generarse clases envolventes al generar código.
JavaSettings_ApacheAxisVersion 1378	Esta propiedad está obsoleta.
Parent 1378	De solo lectura. El objeto de nivel superior según el modelo del objeto.
ShowLogoOnPrint 1378	Esconde o muestra el logo de MapForce en las copias impresas.
ShowLogoOnStartup 1379	Esconde o muestra el logo de MapForce al arrancar la aplicación.
UseGradientBackground 1379	Indica u obtiene el modo de color de fondo para una ventana de asignación.

Nombre	Descripción
XSLTDefaultOutputDirectory 1380	Indica el directorio de destino en el que se encuentran los archivos generados por `Document.GenerateXSLT`.

16.7.1.18.1 Propiedades

16.7.1.18.1.1 Application

Obtiene el objeto de nivel superior de la aplicación.

Firma

```
Application : Application
```

Firma genérica

Errores

Código de error	Descripción
1400	El objeto ya no es válido.
1401	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.18.1.2 CodeDefaultOutputDirectory

Indica el directorio de destino en el que se encuentran los archivos generados por `Document.GenerateCppCode`, `Document.GenerateJavaCode` y `Document.GenerateCHashCode`.

Firma

```
CodeDefaultOutputDirectory : String
```

Firma genérica

Errores

Código de error	Descripción
1400	El objeto ya no es válido.
1401	La dirección indicada para el parámetro de retorno no es válida.

© 2016-2022 Altova GmbH

16.7.1.18.1.3 CPPSettings_DOMType

Indica el tipo DOM que usa `Document.GenerateCppCode`.

Firma

```
CPPSettings_DOMType : ENUMDOMType 1416
```

Firma genérica

Errores

Código de error	Descripción
1400	El objeto ya no es válido.
1401	La dirección indicada para el parámetro de retorno no es válida.
1402	El valor del parámetro está fuera de intervalo.
1403	El valor del parámetro ya no está disponible.

16.7.1.18.1.4 CPPSettings_GenerateVC6ProjectFile (obsolete)

Indica si `Document.GenerateCppCode` debe generar archivos de proyecto de VisualC++ 6.0.

Firma

```
CPPSettings_GenerateVC6ProjectFile : Boolean
```

Firma genérica

Errores

Código de error	Descripción
1400	El objeto ya no es válido.
1401	La dirección indicada para el parámetro de retorno no es válida.
1402	El valor del parámetro está fuera de intervalo.
1403	El valor del parámetro ya no está disponible.

16.7.1.18.1.5 CppSettings_GenerateVSProjectFile

Indica en qué versión de Visual Studio debe generar `Document.GenerateCppCode` los archivos de proyecto.

Firma

```
CppSettings_GenerateVSProjectFile : ENUMProjectType 1417
```

Firma genérica

Errores

Código de error	Descripción
1400	El objeto ya no es válido.
1401	La dirección indicada para el parámetro de retorno no es válida.
1402	El valor del parámetro está fuera de intervalo.
1403	El valor del parámetro ya no está disponible.

16.7.1.18.1.6 CPPSettings_LibraryType

Indica el tipo de biblioteca que usa `Document.GenerateCppCode`.

Firma

```
CPPSettings_LibraryType : ENUMLibType 1416
```

Firma genérica

Errores

Código de error	Descripción
1400	El objeto ya no es válido.
1401	La dirección indicada para el parámetro de retorno no es válida.

© 2016-2022 Altova GmbH

16.7.1.18.1.7 CPPSettings_UseMFC

Indica si el código C++ generado por `Document.GenerateCppCode` debe usar el soporte de MFC.

Firma

```
CPPSettings_UseMFC : Boolean
```

Firma genérica

Errores

Código de error	Descripción
1400	El objeto ya no es válido.
1401	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.18.1.8 CSharpSettings_ProjectType

Indica el tipo de proyecto C# que usa `Document.GenerateCHashCode`.

Firma

```
CSharpSettings_ProjectType : ENUMProjectType 1417
```

Firma genérica

Errores

Código de error	Descripción
1400	El objeto ya no es válido.
1401	La dirección indicada para el parámetro de retorno no es válida.
1402	El valor del parámetro está fuera de intervalo.
1403	El valor del parámetro ya no está disponible.

16.7.1.18.1.9 DefaultOutputByteOrder

Orden de bytes para la codificación usada para los archivos de salida.

Firma

```
DefaultOutputByteOrder : String
```

Firma genérica

Errores

Código de error	Descripción
1400	El objeto ya no es válido.
1401	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.18.1.10 DefaultOutputByteOrderMark

Indica si se debe incluir una marca BOM (marca de orden de bytes) en la codificación del archivo de los archivos de salida.

Firma

```
DefaultOutputByteOrderMark : Boolean
```

Firma genérica

Errores

Código de error	Descripción
1400	El objeto ya no es válido.
1401	La dirección indicada para el parámetro de retorno no es válida.

© 2016-2022 Altova GmbH

16.7.1.18.1.11 DefaultOutputEncoding

Codificación usada para los archivos de salida.

Firma

```
DefaultOutputEncoding : String
```

Firma genérica

Errores

Código de error	Descripción
1400	El objeto ya no es válido.
1401	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.18.1.12 GenerateWrapperClasses

Indica si también deben generarse clases envolventes al generar código.

Firma

```
GenerateWrapperClasses : Boolean
```

Firma genérica

Errores

Código de error	Descripción
1400	El objeto ya no es válido.
1401	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.18.1.13 JavaSettings_ApacheAxisVersion (obsolete)

Esta propiedad está obsoleta.

Firma

```
JavaSettings_ApacheAxisVersion : ENUMApacheAxisVersion 1412
```

Firma genérica

16.7.1.18.1.14 Parent

El objeto de nivel superior según el modelo del objeto.

Firma

```
Parent : Application
```

Firma genérica

Errores

Código de error	Descripción
1400	El objeto ya no es válido.
1401	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.18.1.15 ShowLogoOnPrint

Esconde o muestra el logo de MapForce en las copias impresas.

Firma

```
ShowLogoOnPrint : Boolean
```

Firma genérica

Errores

Código de error	Descripción
1400	El objeto ya no es válido.

© 2016-2022 Altova GmbH

Código de error	Descripción
1401	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.18.1.16 ShowLogoOnStartup

Esconde o muestra el logo de MapForce al arrancar la aplicación.

Firma

```
ShowLogoOnStartup : Boolean
```

Firma genérica

Errores

Código de error	Descripción
1400	El objeto ya no es válido.
1401	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.18.1.17 UseGradientBackground

Indica u obtiene el modo de color de fondo para una ventana de asignación.

Firma

```
UseGradientBackground : Boolean
```

Firma genérica

Errores

Código de error	Descripción
1400	El objeto ya no es válido.
1401	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.18.1.18 XSLTDefaultOutputDirectory

Indica el directorio de destino en el que se encuentran los archivos generados por `Document.GenerateXSLT`.

Firma

```
XSLTDefaultOutputDirectory : String
```

Firma genérica

Errores

Código de error	Descripción
1400	El objeto ya no es válido.
1401	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.19 Project

Un objeto `Project` representa un proyecto y su estructura de elementos del proyecto en MapForce.

Propiedades para navegar en el modelo del objeto:

- `Application`
- `Parent`

Para gestionar archivos, use:

- `FullName`
- `Name`
- `Path`
- `Saved`
- `Save`
- `Close`

Para navegar por la estructura del proyecto, use:

- `Count`
- `Item`
- `_NewEnum`

Para modificar la estructura, use:

- `AddActiveFile`
- `AddFile`
- `InsertWebService` (solo en la edición Enterprise)
- `CreateFolder`

 © 2016-2022 Altova GmbH

Para generar código, use:

- Output_Folder
- Output_Language
- Output_TextEncoding
- Java_BasePackageName
- GenerateCode
- GenerateCodeEx
- GenerateCodeIn
- GenerateCodeInEx

Para ver ejemplos de cómo usar las propiedades y los métodos de la lista anterior, consulte el ejemplo: Tareas del proyecto 1249. Tenga en cuenta que para realizar operaciones que incluyan servicios web es necesario tener instalada la edición MapForce Enterprise.

Propiedades

Nombre	Descripción
_NewEnum 1383	De solo lectura. Esta propiedad admite enumeraciones estándar propias de cada lenguaje.
Application 1384	De solo lectura. Obtiene el objeto de aplicación de nivel superior.
Count 1385	De solo lectura. Obtiene el número de secundarios del elemento raíz del proyecto. Para ver ejemplos, consulte `Item` o `_NewEnum`
FullName 1385	Ruta de acceso y nombre del fichero de proyecto.
Item 1386	De solo lectura. Devuelve el secundario en la posición *n* de la raíz del proyecto. El índice es de base uno (el primer índice es 1). El máximo índice válido es `Count`. Para ver una alternativa consulte `_NewEnum`.
Java_BasePackageName 1386	Indica u obtiene el nombre base de los paquetes Java que se van a generar. Esta propiedad solo se usa al generar código Java.
Name 1387	De solo lectura. Nombre del fichero de proyecto sin ruta de acceso.
Output_Folder 1387	Indica u obtiene la carpeta de salida predeterminada mediante `GenerateCode` o `GenerateCodeIn`. Los elementos del proyecto pueden sobrescribir este valor en su propiedad `CodeGenSettings_OutputFolder` si `CodeGenSettings_UseDefault` es false.
Output_Language 1388	Indica u obtiene el lenguaje predeterminado para generar código al usar `GenerateCode`. Los elementos del proyecto pueden sobrescribir este valor en su propiedad

Nombre	Descripción
	`CodeGenSettings_OutputLanguage` si `CodeGenSettings_UseDefault` es false.
`Output_TextEncoding` 1388	Indica u obtiene la codificación de texto que se usa al generar código basado en XML.
`Parent` 1389	De solo lectura. El objeto de nivel superior según el modelo del objeto.
`Path` 1389	De solo lectura. Ruta del fichero de proyecto sin nombre.
`Saved` 1390	De solo lectura. **True** si el proyecto no se ha modificado desde la última que se guardó, **false** en caso contrario.

Métodos

Nombre	Descripción
`AddActiveFile` 1390	Añade el documento activo a la carpeta de la asignación de la raíz del proyecto.
`AddFile` 1391	Añade el documento indicado a la carpeta de la asignación de la raíz del proyecto.
`Close` 1391	Cierra el proyecto sin guardarlo.
`CreateFolder` 1392	Crea una nueva carpeta como secundario del elemento raíz del proyecto.
`GenerateCode` 1392	Genera código para todos los elementos del proyecto. El lenguaje del código y la ubicación de salida las determinan las propiedades del proyecto y sus elementos.
`GenerateCodeEx` 1393	Genera código para todos los elementos del proyecto. El lenguaje del código y la ubicación de salida las determinan las propiedades del proyecto y sus elementos. Devuelve un objeto que se puede iterar en todos los mensajes emitidos por el proceso de generación de código. Los mensajes son los mismos que los que se muestran en la ventana Mensajes de MapForce.
`GenerateCodeIn` 1393	Genera código para todos los elementos del proyecto en el lenguaje indicado. La ubicación de salida la determinan las propiedades del proyecto y sus elementos.
`GenerateCodeInEx` 1394	Genera código para todos los elementos del proyecto en el lenguaje indicado. La ubicación de salida la determinan las propiedades del proyecto y sus elementos. Devuelve un objeto que se puede iterar en todos los mensajes emitidos por el

© 2016-2022 Altova GmbH

Nombre	Descripción
	proceso de generación de código. Los mensajes son los mismos que los que se muestran en la ventanas **Mensajes** de MapForce.
`InsertWebService` 1394	Inserta un nuevo proyecto de servicio en la carpeta Servicio web del proyecto. Si **i_bGenerateMappings** se encuentra en true, entonces se generan automáticamente documentos de asignación iniciales para todos los puertos.
`Save` 1395	Guarda el proyecto en el archivo definido por `FullName`.

Eventos

Nombre	Descripción
`OnProjectClosed` 1396	Este evento se desencadena cuando se cierra el proyecto. No se debe llamar al objeto proyecto pasado al gestor de eventos. El evento de apertura correspondiente es `Application.OnProjectOpened`.

16.7.1.19.1 Propiedades

16.7.1.19.1.1 _NewEnum

Esta propiedad admite enumeraciones estándar propias de cada lenguaje.

Firma

```
_NewEnum : IUnknown
```

Firma genérica

Errores

Código de error	Descripción
1500	El objeto ya no es válido.

Ejemplos

```
// ------------------------------------------------------------
// JScript sample - enumeration of a project's project items.
function AllChildrenOfProjectRoot()
{
```

```
    objProject = objMapForce.ActiveProject;
    if ( objProject != null )
    {

for ( objProjectIter = new Enumerator(objProject); ! objProjectIter.atEnd(); objProjectIt
er.moveNext() )
        {
            objProjectItem = objProjectIter.item();

            // do something with project item here
        }
    }
}
```

```
// ------------------------------------------------------------
// JScript sample - iterate all project items, depth first.
function IterateProjectItemsRec(objProjectItemIter)
{
    while ( ! objProjectItemIter.atEnd() )
    {
        objProjectItem = objProjectItemIter.item();
        // do something with project item here

        IterateProjectItemsRec( new Enumerator(objProjectItem) );

        objProjectItemIter.moveNext();
    }
}
function IterateAllProjectItems()
{
    objProject = objMapForce.ActiveProject;
    if ( objProject != null )
    {
        IterateProjectItemsRec( new Enumerator(objProject) );
    }
}
```

16.7.1.19.1.2 Application

Obtiene el objeto de aplicación de nivel superior.

Firma

```
Application : Application
```

Firma genérica

 © 2016-2022 Altova GmbH

Errores

Código de error	Descripción
1500	El objeto ya no es válido.
1501	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.19.1.3 Count

Obtiene el número de secundarios del elemento raíz del proyecto. Para ver ejemplos, consulte `Item` o `_NewEnum`

Firma

```
Count : Integer
```

Firma genérica

Errores

Código de error	Descripción
1500	El objeto ya no es válido.

16.7.1.19.1.4 FullName

Ruta de acceso y nombre del fichero de proyecto.

Firma

```
FullName : String
```

Firma genérica

Errores

Código de error	Descripción
1500	El objeto ya no es válido.
1501	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.19.1.5 Item

Devuelve el secundario en la posición *n* de la raíz del proyecto. El índice es de base uno (el primer índice es 1). El máximo índice válido es `Count`. Para ver una alternativa consulte `_NewEnum`.

Firma

```
Item(in n:Integer) : ProjectItem
```

Firma genérica

Errores

Código de error	Descripción
1500	El objeto ya no es válido.

Ejemplos

```
// ----------------------------------------------------------
// JScript code snippet - enumerate children using Count and Item.
for( nItemIndex = 1; nItemIndex <= objProject.Count; nItemIndex++ )
{
    objProjectItem = objProject.Item(nItemIndex);
    // do something with project item here
}
```

16.7.1.19.1.6 Java_BasePackageName

Indica u obtiene el nombre base de los paquetes Java que se van a generar. Esta propiedad solo se usa al generar código Java.

Firma

```
Java_BasePackageName : String
```

Firma genérica

Errores

Código de error	Descripción
1500	El objeto ya no es válido.

© 2016-2022 Altova GmbH

Código de error	Descripción
1501	El nombre de paquete indicado no es válido. La dirección indicada para el parámetro de retorno no es válida.

16.7.1.19.1.7 Name

Nombre del fichero de proyecto sin ruta de acceso.

Firma

```
Name : String
```

Firma genérica

Errores

Código de error	Descripción
1500	El objeto ya no es válido.
1501	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.19.1.8 Output_Folder

Indica u obtiene la carpeta de salida predeterminada mediante `GenerateCode` o `GenerateCodeIn`. Los elementos del proyecto pueden sobrescribir este valor en su propiedad `CodeGenSettings_OutputFolder` si `CodeGenSettings_UseDefault` es false.

Firma

```
Output_Folder : String
```

Firma genérica

Errores

Código de error	Descripción
1500	El objeto ya no es válido.
1501	El nombre de carpeta indicado no es válido. La dirección indicada para el parámetro de retorno no es válida.

16.7.1.19.1.9 Output_Language

Indica u obtiene el lenguaje predeterminado para generar código al usar `GenerateCode`. Los elementos del proyecto pueden sobrescribir este valor en su propiedad `CodeGenSettings_OutputLanguage` si `CodeGenSettings_UseDefault` es false.

Firma

```
Output_Language : ENUMProgrammingLanguage 1416
```

Firma genérica

Errores

Código de error	Descripción
1500	El objeto ya no es válido.
1501	El lenguaje indicado no es válido. La dirección indicada para el parámetro de retorno no es válida.

16.7.1.19.1.10 Output_TextEncoding

Indica u obtiene la codificación de texto que se usa al generar código basado en XML.

Firma

```
Output_TextEncoding : String
```

Firma genérica

Errores

Código de error	Descripción
1500	El objeto ya no es válido.
1501	La codificación de texto indicada no es válida. La dirección indicada para el parámetro de retorno no es válida.

© 2016-2022 Altova GmbH

16.7.1.19.1.11 Parent

El objeto de nivel superior según el modelo del objeto.

Firma

```
Parent : Application
```

Firma genérica

Errores

Código de error	Descripción
1500	El objeto ya no es válido.
1501	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.19.1.12 Path

Ruta del fichero de proyecto sin nombre.

Firma

```
Path : String
```

Firma genérica

Errores

Código de error	Descripción
1500	El objeto ya no es válido.
1501	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.19.1.13 Saved

True si el proyecto no se ha modificado desde la última que se guardó, **false** en caso contrario.

Firma

```
Saved : Boolean
```

Firma genérica

Errores

Código de error	Descripción
1500	El objeto ya no es válido.
1501	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.19.2 Métodos

16.7.1.19.2.1 AddActiveFile

Añade el documento activo a la carpeta de la asignación de la raíz del proyecto.

Firma

```
AddActiveFile() -> ProjectItem
```

Firma genérica

Errores

Código de error	Descripción
1500	El objeto ya no es válido.
1501	La dirección indicada para el parámetro de retorno no es válida.
1503	No hay ningún documento activo disponible.
1504	El documento activo debe contener un nombre de ruta para que se pueda añadir al proyecto.
1705	La asignación no se puede asignar al proyecto. Puede que ya exista en la carpeta de destino.

© 2016-2022 Altova GmbH

16.7.1.19.2.2 AddFile

Añade el documento indicado a la carpeta de la asignación de la raíz del proyecto.

Firma

```
AddFile(in i_strFileName:String) -> ProjectItem
```

Firma genérica

Parámetros

Nombre	Tipo	Descripción
i_strFileName	`String`	Indica la ruta del documento que se quiere añadir.

Errores

Código de error	Descripción
1500	El objeto ya no es válido.
1501	La dirección indicada para el parámetro de retorno no es válida.
1705	La asignación no se puede asignar al proyecto. El archivo no existe o no es una asignación de MapForce. Puede que este archivo ya esté asignado a la carpeta de destino.

16.7.1.19.2.3 Close

Cierra el proyecto sin guardarlo.

Firma

```
Close() -> Void
```

Firma genérica

Errores

Código de error	Descripción
1500	El objeto ya no es válido.

16.7.1.19.2.4 CreateFolder

Crea una nueva carpeta como secundario del elemento raíz del proyecto.

Firma

```
CreateFolder(in i_strFolderName:String) -> ProjectItem
```

Firma genérica

Parámetros

Nombre	Tipo	Descripción
i_strFolderName	`String`	El nombre de la carpeta que se quiere crear.

Errores

Código de error	Descripción
1500	El objeto ya no es válido.
1501	El nombre de carpeta o la dirección indicados para el parámetro de retorno son inválidos.

16.7.1.19.2.5 GenerateCode

Genera código para todos los elementos del proyecto. El lenguaje del código y la ubicación de salida las determinan las propiedades del proyecto y sus elementos.

Firma

```
GenerateCode() -> Void
```

Firma genérica

Errores

Código de error	Descripción
1500	El objeto ya no es válido.
1706	Error durante la generación del código.

© 2016-2022 Altova GmbH

16.7.1.19.2.6 GenerateCodeEx

Genera código para todos los elementos del proyecto. El lenguaje del código y la ubicación de salida las determinan las propiedades del proyecto y sus elementos. Devuelve un objeto que se puede iterar en todos los mensajes emitidos por el proceso de generación de código. Los mensajes son los mismos que los que se muestran en la ventana Mensajes de MapForce.

Firma

```
GenerateCodeEx() -> ErrorMarkers
```

Firma genérica

Errores

Código de error	Descripción
1500	El objeto ya no es válido.
1501	La dirección indicada para el parámetro de retorno no es válida.
1706	Error durante la generación del código.

16.7.1.19.2.7 GenerateCodeIn

Genera código para todos los elementos del proyecto en el lenguaje indicado. La ubicación de salida la determinan las propiedades del proyecto y sus elementos.

Firma

```
GenerateCodeIn(in i_nLanguage:ENUMProgrammingLanguage[1416]) -> Void
```

Firma genérica

Parámetros

Nombre	Tipo	Descripción
i_nLanguage	ENUMProgrammingLanguage[1416]	Indica el lenguaje de programación en el que se debe generar el código.

Errores

Código de error	Descripción
1500	El objeto ya no es válido.
1706	Error durante la generación del código.

16.7.1.19.2.8 GenerateCodeInEx

Genera código para todos los elementos del proyecto en el lenguaje indicado. La ubicación de salida la determinan las propiedades del proyecto y sus elementos. Devuelve un objeto que se puede iterar en todos los mensajes emitidos por el proceso de generación de código. Los mensajes son los mismos que los que se muestran en la ventanas **Mensajes** de MapForce.

Firma

```
GenerateCodeInEx(in i_nLanguage:ENUMProgrammingLanguage[1416]) -> ErrorMarkers
```

Firma genérica

Parámetros

Nombre	Tipo	Descripción
i_nLanguage	ENUMProgrammingLanguage[1416]	Indica el lenguaje de programación en el que se debe generar el código.

Errores

Código de error	Descripción
1500	El objeto ya no es válido.
1501	La dirección indicada para el parámetro de retorno no es válida.
1706	Error durante la generación del código.

16.7.1.19.2.9 InsertWebService

Inserta un nuevo proyecto de servicio en la carpeta Servicio web del proyecto. Si **i_bGenerateMappings** se encuentra en true, entonces se generan automáticamente documentos de asignación iniciales para todos los puertos.

© 2016-2022 Altova GmbH

Firma

```
InsertWebService(in i_strWSDLFile:String, in i_strService:String, in i_strPort:String, in
i_bGenerateMappings:Boolean) -> ProjectItem
```

Firma genérica

Parámetros

Nombre	Tipo	Descripción
i_strWSDLFile	`String`	Indica la ruta del archivo WSDL que se quiere añadir.
i_strService	`String`	Indica el nombre del servicio web que se quiere añadir.
i_strPort	`String`	Indica el puerto del servicio web que se quiere añadir.
i_bGenerateMappings	`Boolean`	Si este parámetro se encuentra en true, entonces se generan automáticamente documentos de asignación iniciales para todos los puertos.

Errores

Código de error	Descripción
1500	El objeto ya no es válido.
1501	El archivo WSDL no se puede encontrar o es inválido. El nombre del servicio o del puerto es inválido. La dirección indicada para el parámetro de retorno no es válida.
1503	La edición actual no admite esta operación.

16.7.1.19.2.10 Save

Guarda el proyecto en el archivo definido por `FullName`.

Firma

```
Save() -> Void
```

Firma genérica

Errores

Código de error	Descripción
1500	El objeto ya no es válido.
1502	No se puede guardar en el archivo.

16.7.1.19.3 Eventos

16.7.1.19.3.1 OnProjectClosed

Este evento se desencadena cuando se cierra el proyecto. No se debe llamar al objeto proyecto pasado al gestor de eventos. El evento de apertura correspondiente es `Application.OnProjectOpened`.

Firma

```
OnProjectClosed(in i_ipProject:Project) : Void
```

16.7.1.20 ProjectItem

Un objeto `ProjectItem` representa un elemento de una estructura de proyecto.

Propiedades para navegar en el modelo del objeto:

- `Application`
- `Parent`

Para navegar por la estructura del proyecto, use:

- `Count`
- `Item`
- `_NewEnum`

Propiedades de los elementos del proyecto:

- `Kind`
- `Name`
- `WSDLFile` (solo disponible para elementos de proyectos de servicios web)
- `QualifiedName` (solo disponible para elementos de proyectos de servicios web)

Para modificar la estructura del proyecto, use:

- `AddActiveFile` (solo disponible para elementos de la carpeta)
- `AddFile` (solo disponible para elementos de la carpeta)
- `CreateFolder` (solo disponible para elementos de la carpeta)
- `CreateMappingForProject` (solo disponible para operaciones de servicios web)

© 2016-2022 Altova GmbH

- Remove

Para acceder a documentos, use:

- Open (solo disponible para elementos de asignación y operaciones de servicios web)

Para generar código, use:

- CodeGenSettings_UseDefault
- CodeGenSettings_OutputFolder
- CodeGenSettings_Language
- GenerateCode
- GenerateCodeEx
- GenerateCodeIn
- GenerateCodeInEx

Para ver ejemplos de cómo usar las propiedades y los métodos de la lista anterior, consulte el ejemplo: Tareas del proyecto[1249]. Las operaciones con servicios web requieren la edición MapForce Enterprise.

Propiedades

Nombre	Descripción
_NewEnum[1400]	De solo lectura. Esta propiedad admite enumeraciones estándar propias de cada lenguaje. Para ver ejemplos, consulte Project.Item o Project._NewEnum
Application[1400]	De solo lectura. Obtiene el objeto de aplicación de nivel superior.
CodeGenSettings_Language[1401]	Mediante GenerateCode o Project.GenerateCode, indica u obtiene el lenguaje que se debe usar. Esta propiedad se consulta solo si CodeGenSettings_UseDefault se encuentra en false.
CodeGenSettings_OutputFolder[1401]	Mediante GenerateCode, GenerateCodeIn, Project.GenerateCode o Project.GenerateCodeIn, obtiene o indica qué directorio de salida debe usarse. Esta propiedad se consulta solo si CodeGenSettings_UseDefault se encuentra en false.
CodeGenSettings_UseDefault[1402]	Obtiene o indica si el directorio de salida y el lenguaje del código se usan según lo definido en (a) las carpetas superiores o (b) la raíz del proyecto. Esta propiedad se usa con llamadas a GenerateCode, GenerateCodeIn, Project.GenerateCode y Project.GenerateCodeIn. Si esta propiedad se encuentra en false, entonces los valores de CodeGenSettings_OutputFolder y CodeGenSettings_Language se usan para generar código para este elemento del proyecto.
Count[1402]	De solo lectura.

Nombre	Descripción
	Obtiene el número de secundarios de este elemento del proyecto. Consulte también `Item`. Para ver ejemplos, consulte `Project.Item` o `Project._NewEnum`
`Item` 1403	De solo lectura. Devuelve el secundario en la posición `n` de este elemento del proyecto. El índice es de base uno (el primer índice es 1). El máximo índice válido es `ProjectItem.Count` 1. Para ver una alternativa consulte `ProjectItem._NewEnum`. Para ver ejemplos, consulte `Project.Item` o `Project._NewEnum`
`Kind` 1403	De solo lectura. Obtiene el tipo de elemento del proyecto. Algunas propiedades solo están disponibles en y algunos métodos solo se pueden aplicar a ciertos tipos de elementos de proyecto. Las descripciones de los métodos y las propiedades contiene información acerca de estas restricciones.
`Name` 1403	Obtiene o indica el nombre de un elemento del proyecto. El nombre de la mayoría de los elementos es solo de lectura. Una excepción son las carpetas creadas por los usuarios, cuyos nombres se pueden modificar.
`Parent` 1404	De solo lectura. Obtiene el proyecto del que este elemento es secundario. Tiene el mismo efecto que `Application.ActiveProject`.
`QualifiedName` 1404	De solo lectura. Obtiene el nombre completo de un elemento de un servicio web.
`WSDLFile` 1405	De solo lectura. Obtiene el nombre del archivo WSDL que define el servicio web que contiene el elemento de proyecto actual.

Métodos

Nombre	Descripción
`AddActiveFile` 1405	Añade el documento activo en ese momento a este elemento de proyecto si es un elemento secundario válido. En caso contrario añade ese documento a la carpeta de asignación de la raíz del proyecto.
`AddFile` 1406	Añade el documento indicado a este elemento de proyecto si es un elemento secundario válido. En caso contrario añade ese documento a la carpeta de asignación de la raíz del proyecto.
`CreateFolder` 1407	Crea una nueva carpeta como secundario de este elemento de proyecto.

© 2016-2022 Altova GmbH

Nombre	Descripción
CreateMappingForProject 1407	Crea un documento inicial de asignación para una operación de servicio web y lo guarda en **i_strFileName**. Al usar `Project.InsertWebService` puede usar la bandera **i_bGenerateMappings** para que MapForce genere automáticamente asignaciones iniciales para todos los puertos.
GenerateCode 1408	Genera código para este elemento de proyecto y sus secundarios. El lenguaje del código y la ubicación de salida las determinan `CodeGenSettings_UseDefault`, `CodeGenSettings_Language` y `CodeGenSettings_OutputFolder`. Los secundarios de este elemento de proyecto pueden tener su propia configuración de propiedades para la generación de código.
GenerateCodeEx 1408	Genera código para este elemento de proyecto y sus secundarios. El lenguaje del código y la ubicación de salida las determinan `CodeGenSettings_UseDefault`, `CodeGenSettings_Language` y `CodeGenSettings_OutputFolder`. Los secundarios de este elemento de proyecto pueden tener su propia configuración de propiedades para la generación de código.
GenerateCodeIn 1409	Genera código en el lenguaje indicado para el elementos del proyecto y sus secundarios. La ubicación de salida la determinan `CodeGenSettings_UseDefault` y `CodeGenSettings_OutputFolder`. Los secundarios de este elemento de proyecto pueden tener su propia configuración de propiedades para la generación de código.
GenerateCodeInEx 1410	Genera código en el lenguaje indicado para el elementos del proyecto y sus secundarios. La ubicación de salida la determinan `CodeGenSettings_UseDefault` y `CodeGenSettings_OutputFolder`. Los secundarios de este elemento de proyecto pueden tener su propia configuración de propiedades para la generación de código. Devuelve un objeto que se puede iterar en todos los mensajes emitidos por el proceso de generación de código. Los mensajes son los mismos que los que se muestran en la ventana Mensajes de MapForce.
Open 1410	Abre el elemento de proyecto como documento o convierte en activo el documento correspondiente si ya está abierto. El elemento de proyecto debe ser una asignación de MapForce o, solo en el caso de la edición Enterprise, una operación de servicio web.
Remove 1411	Elimina este elemento y todos sus secundarios de la estructura del proyecto.

Eventos

Nombre	Descripción
OnModifiedFlagChanged 1411	Se da cuando cambia el estado de modificación de `ProjectItem`.
OnProjectClosed 1411	Este evento se desencadena cuando se cierra el proyecto. No se debe llamar al objeto proyecto pasado al gestor de eventos. El evento de apertura correspondiente es `Application.OnProjectOpened`.

16.7.1.20.1 Propiedades

16.7.1.20.1.1 _NewEnum

Esta propiedad admite enumeraciones estándar propias de cada lenguaje. Para ver ejemplos, consulte `Project.Item` o `Project._NewEnum`

Firma

```
_NewEnum : IUnknown
```

Firma genérica

Errores

Código de error	Descripción
1700	El objeto ya no es válido.

16.7.1.20.1.2 Application

Obtiene el objeto de aplicación de nivel superior.

Firma

```
Application : Application
```

Firma genérica

© 2016-2022 Altova GmbH

Errores

Código de error	Descripción
1700	El objeto ya no es válido.
1701	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.20.1.3 CodeGenSettings_Language

Mediante `GenerateCode` o `Project.GenerateCode`, indica u obtiene el lenguaje que se debe usar. Esta propiedad se consulta solo si `CodeGenSettings_UseDefault` se encuentra en false.

Firma

```
CodeGenSettings_Language : ENUMProgrammingLanguage 1416
```

Firma genérica

Errores

Código de error	Descripción
1700	El objeto ya no es válido.
1701	El lenguaje o la dirección indicados para el parámetro de retorno son inválidos.

16.7.1.20.1.4 CodeGenSettings_OutputFolder

Mediante `GenerateCode`, `GenerateCodeIn`, `Project.GenerateCode` o `Project.GenerateCodeIn`, obtiene o indica qué directorio de salida debe usarse. Esta propiedad se consulta solo si `CodeGenSettings_UseDefault` se encuentra en false.

Firma

```
CodeGenSettings_OutputFolder : String
```

Firma genérica

Errores

Código de error	Descripción
1700	El objeto ya no es válido.

Código de error	Descripción
1701	Se indicaron una carpeta de salida o una dirección inválidas para el parámetro de retorno.

16.7.1.20.1.5 CodeGenSettings_UseDefault

Obtiene o indica si el directorio de salida y el lenguaje del código se usan según lo definido en (a) las carpetas superiores o (b) la raíz del proyecto. Esta propiedad se usa con llamadas a `GenerateCode`, `GenerateCodeIn`, `Project.GenerateCode` y `Project.GenerateCodeIn`. Si esta propiedad se encuentra en false, entonces los valores de `CodeGenSettings_OutputFolder` y `CodeGenSettings_Language` se usan para generar código para este elemento del proyecto.

Firma

```
CodeGenSettings_UseDefault : Boolean
```

Firma genérica

Errores

Código de error	Descripción
1700	El objeto ya no es válido.
1701	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.20.1.6 Count

Obtiene el número de secundarios de este elemento del proyecto. Consulte también `Item`. Para ver ejemplos, consulte `Project.Item` o `Project._NewEnum`

Firma

```
Count : Integer
```

Firma genérica

Errores

Código de error	Descripción
1700	El objeto ya no es válido.

© 2016-2022 Altova GmbH

16.7.1.20.1.7 Item

Devuelve el secundario en la posición `n` de este elemento del proyecto. El índice es de base uno (el primer índice es 1). El máximo índice válido es `ProjectItem.Count` 1. Para ver una alternativa consulte `ProjectItem._NewEnum`. Para ver ejemplos, consulte `Project.Item` o `Project._NewEnum`

Firma

```
Item(in n:Integer) : ProjectItem
```

Firma genérica

Errores

Código de error	Descripción
1700	El objeto ya no es válido.

16.7.1.20.1.8 Kind

Obtiene el tipo de elemento del proyecto. Algunas propiedades solo están disponibles en y algunos métodos solo se pueden aplicar a ciertos tipos de elementos de proyecto. Las descripciones de los métodos y las propiedades contiene información acerca de estas restricciones.

Firma

```
Kind : ENUMProjectItemType 1417
```

Firma genérica

Errores

Código de error	Descripción
1700	El objeto ya no es válido.
1701	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.20.1.9 Name

Obtiene o indica el nombre de un elemento del proyecto. El nombre de la mayoría de los elementos es solo de lectura. Una excepción son las carpetas creadas por los usuarios, cuyos nombres se pueden modificar.

Firma

```
Name : String
```

Firma genérica

Errores

Código de error	Descripción
1700	El objeto ya no es válido.
1701	La dirección indicada para el parámetro de retorno no es válida.
1702	No se puede cambiar el nombre de este elemento del proyecto.

16.7.1.20.1.10 Parent

Obtiene el proyecto del que este elemento es secundario. Tiene el mismo efecto que `Application.ActiveProject`.

Firma

```
Parent : Project
```

Firma genérica

Errores

Código de error	Descripción
1700	El objeto ya no es válido.
1701	La dirección indicada para el parámetro de retorno no es válida.

16.7.1.20.1.11 QualifiedName

Obtiene el nombre completo de un elemento de un servicio web.

Firma

```
QualifiedName : String
```

Firma genérica

© 2016-2022 Altova GmbH

Errores

Código de error	Descripción
1700	El objeto ya no es válido.
1701	La dirección indicada para el parámetro de retorno no es válida.
1702	El elemento de proyecto no forma parte de un servicio web.

16.7.1.20.1.12 WSDLFile

Obtiene el nombre del archivo WSDL que define el servicio web que contiene el elemento de proyecto actual.

Firma

```
WSDLFile : String
```

Firma genérica

Errores

Código de error	Descripción
1700	El objeto ya no es válido.
1701	La dirección indicada para el parámetro de retorno no es válida.
1702	El elemento de proyecto no forma parte de un servicio web.

16.7.1.20.2 Métodos

16.7.1.20.2.1 AddActiveFile

Añade el documento activo en ese momento a este elemento de proyecto si es un elemento secundario válido. En caso contrario añade ese documento a la carpeta de asignación de la raíz del proyecto.

Firma

```
AddActiveFile() -> ProjectItem
```

Firma genérica

Errores

Código de error	Descripción
1700	El objeto ya no es válido.
1701	El nombre del archivo está vacío. La dirección indicada para el parámetro de retorno no es válida.
1703	No hay ningún documento activo disponible.
1704	El documento activo debe contener un nombre de ruta para que se pueda añadir al proyecto.
1705	La asignación no se puede asignar al proyecto. El archivo no existe o no es una asignación de MapForce. Puede que este archivo ya esté asignado a la carpeta de destino.

16.7.1.20.2.2 AddFile

Añade el documento indicado a este elemento de proyecto si es un elemento secundario válido. En caso contrario añade ese documento a la carpeta de asignación de la raíz del proyecto.

Firma

```
AddFile(in i_strFilePath:String) -> ProjectItem
```

Firma genérica

Parámetros

Nombre	Tipo	Descripción
i_strFilePath	`String`	La ruta del documento que quiere abrir.

Errores

Código de error	Descripción
1700	El objeto ya no es válido.
1701	El nombre del archivo está vacío. La dirección indicada para el parámetro de retorno no es válida.
1705	La asignación no se puede asignar al proyecto. El archivo no existe o no es una asignación de MapForce. Puede que este archivo ya esté asignado a la carpeta de destino.

© 2016-2022 Altova GmbH

16.7.1.20.2.3 CreateFolder

Crea una nueva carpeta como secundario de este elemento de proyecto.

Firma

```
CreateFolder(in i_strFolderName:String) -> ProjectItem
```

Firma genérica

Parámetros

Nombre	Tipo	Descripción
i_strFolderName	`String`	El nombre de la carpeta que se quiere crear.

Errores

Código de error	Descripción
1700	El objeto ya no es válido.
1701	El nombre de carpeta o la dirección indicados para el parámetro de retorno son inválidos.
1702	El elemento de proyecto no admite secundarios.

16.7.1.20.2.4 CreateMappingForProject

Crea un documento inicial de asignación para una operación de servicio web y lo guarda en **i_strFileName**. Al usar `Project.InsertWebService` puede usar la bandera **i_bGenerateMappings** para que MapForce genere automáticamente asignaciones iniciales para todos los puertos.

Firma

```
CreateMappingForProject(in i_strFileName:String) -> ProjectItem
```

Firma genérica

Parámetros

Nombre	Tipo	Descripción
i_strFileName	`String`	Indica la ruta en la que se debe

Nombre	Tipo	Descripción
		guardar la asignación.

Errores

Código de error	Descripción
1700	El objeto ya no es válido.
1701	La dirección indicada para el parámetro de retorno no es válida.
1707	No se puede crear una nueva asignación. El elemento de proyecto no admite la creación automática de asignaciones iniciales o ya existe una asignación.
1708	La edición actual no admite esta operación.

16.7.1.20.2.5 GenerateCode

Genera código para este elemento de proyecto y sus secundarios. El lenguaje del código y la ubicación de salida las determinan `CodeGenSettings_UseDefault`, `CodeGenSettings_Language` y `CodeGenSettings_OutputFolder`. Los secundarios de este elemento de proyecto pueden tener su propia configuración de propiedades para la generación de código.

Firma

```
GenerateCode() -> Void
```

Firma genérica

Errores

Código de error	Descripción
1700	El objeto ya no es válido.
1706	Error durante la generación del código.

16.7.1.20.2.6 GenerateCodeEx

Genera código para este elemento de proyecto y sus secundarios. El lenguaje del código y la ubicación de salida las determinan `CodeGenSettings_UseDefault`, `CodeGenSettings_Language` y `CodeGenSettings_OutputFolder`. Los secundarios de este elemento de proyecto pueden tener su propia configuración de propiedades para la generación de código.

© 2016-2022 Altova GmbH

Firma

```
GenerateCodeEx() -> ErrorMarkers
```

Firma genérica

Errores

Código de error	Descripción
1700	El objeto ya no es válido.
1701	La dirección indicada para el parámetro de retorno no es válida.
1706	Error durante la generación del código.

16.7.1.20.2.7 GenerateCodeIn

Genera código en el lenguaje indicado para el elementos del proyecto y sus secundarios. La ubicación de salida la determinan `CodeGenSettings_UseDefault` y `CodeGenSettings_OutputFolder`. Los secundarios de este elemento de proyecto pueden tener su propia configuración de propiedades para la generación de código.

Firma

```
GenerateCodeIn(in i_nLanguage:ENUMProgrammingLanguage[1416]) -> Void
```

Firma genérica

Parámetros

Nombre	Tipo	Descripción
i_nLanguage	ENUMProgrammingLanguage[1416]	Indica el lenguaje de programación para la generación de código.

Errores

Código de error	Descripción
1700	El objeto ya no es válido.
1701	El lenguaje indicado no es válido.
1706	Error durante la generación del código.

16.7.1.20.2.8 GenerateCodeInEx

Genera código en el lenguaje indicado para el elementos del proyecto y sus secundarios. La ubicación de salida la determinan `CodeGenSettings_UseDefault` y `CodeGenSettings_OutputFolder`. Los secundarios de este elemento de proyecto pueden tener su propia configuración de propiedades para la generación de código.

Devuelve un objeto que se puede iterar en todos los mensajes emitidos por el proceso de generación de código. Los mensajes son los mismos que los que se muestran en la ventana Mensajes de MapForce.

Firma

```
GenerateCodeInEx(in i_nLanguage:ENUMProgrammingLanguage[1416]) -> ErrorMarkers
```

Firma genérica

Parámetros

Nombre	Tipo	Descripción
i_nLanguage	ENUMProgrammingLanguage[1416]	Indica el lenguaje de programación para la generación de código.

Errores

Código de error	Descripción
1700	El objeto ya no es válido.
1701	El lenguaje o la dirección indicados para el parámetro de retorno son inválidos.
1706	Error durante la generación del código.

16.7.1.20.2.9 Open

Abre el elemento de proyecto como documento o convierte en activo el documento correspondiente si ya está abierto. El elemento de proyecto debe ser una asignación de MapForce o, solo en el caso de la edición Enterprise, una operación de servicio web.

Firma

```
Open() -> Document
```

Firma genérica

© 2016-2022 Altova GmbH

Errores

Código de error	Descripción
1700	El objeto ya no es válido.
1701	La dirección indicada para el parámetro de retorno no es válida.
1702	El elemento de proyecto no hace referencia a un archivo de asignación de MapForce.
1708	La edición actual no admite esta operación.

16.7.1.20.2.10 Remove

Elimina este elemento y todos sus secundarios de la estructura del proyecto.

Firma

```
Remove() -> Void
```

Firma genérica

Errores

Código de error	Descripción
1700	El objeto ya no es válido.

16.7.1.20.3 Eventos

16.7.1.20.3.1 OnModifiedFlagChanged

Se da cuando cambia el estado de modificación de `ProjectItem`.

Firma

```
OnModifiedFlagChanged(in i_bIsModified:Boolean) : Void
```

16.7.1.20.3.2 OnProjectClosed

Este evento se desencadena cuando se cierra el proyecto. No se debe llamar al objeto proyecto pasado al gestor de eventos. El evento de apertura correspondiente es `Application.OnProjectOpened`.

Firma

```
OnProjectClosed(in i_ipProject:Project) : Void
```

16.7.2 Enumeraciones

16.7.2.1 ENUMApacheAxisVersion (obsolete)

Este tipo de enumeración está obsoleto.

Miembros

eApacheAxisVersion_Axis = 1

eApacheAxisVersion_Axis2 = 2

16.7.2.2 ENUMApplicationStatus

Valores de enumeración para indicar el estado de la aplicación.

Miembros

eApplicationRunning = 0

eApplicationAfterLicenseCheck = 1

eApplicationBeforeLicenseCheck = 2

eApplicationConcurrentLicenseCheckFailed = 3

eApplicationProcessingCommandLine = 4

16.7.2.3 ENUMAppOutputLine_Severity

Valores de enumeración para identificar la severidad de una `AppOutputLine`.

Miembros

eSeverity_Undefined = -1

eSeverity_Info = 0

eSeverity_Warning = 1

eSeverity_Error = 2

© 2016-2022 Altova GmbH

eSeverity_CriticalError = 3

eSeverity_Success = 4

eSeverity_Summary = 5

eSeverity_Progress = 6

eSeverity_DataEdit = 7

eSeverity_ParserInfo = 8

eSeverity_PossibleInconsistencyWarning = 9

eSeverity_Message = 10

eSeverity_Document = 11

eSeverity_Rest = 12

eSeverity_NoSelect = 13

eSeverity_Select = 14

eSeverity_Autoinsertion = 15

eSeverity_GlobalResources_DefaultWarning = 16

eSeverity_XPath_Styles_Changed = 17

eSeverity_XPath_Styles_Unchanged = 18

eSeverity_XPath_Styles_Skipped = 19

eSeverity_XPath_ComboBox_Values_Changed = 20

eSeverity_XPath_ComboBox_Values_Unchanged = 21

eSeverity_XPath_ComboBox_Values_Skipped = 22

eSeverity_XPath_Assertions_Changed = 23

eSeverity_XPath_Assertions_Unchanged = 24

eSeverity_XPath_Assertions_Skipped = 25

16.7.2.4 ENUMAppOutputLine_TextDecoration

Valores de enumeración para los distintos tipos de formato del texto de una `AppOutputLine`.

Miembros

eTextDecorationDefault = 0

eTextDecorationBold = 1

eTextDecorationDebugValues = 2

eTextDecorationDB_ObjectName = 3

eTextDecorationDB_ObjectLink = 4

eTextDecorationDB_ObjectKind = 5

eTextDecorationDB_TimeoutValue = 6

eTextDecorationFind_MatchingString = 7

eTextDecorationValidation_Speclink = 8

eTextDecorationValidation_ErrorPosition = 9

eTextDecorationValidation_UnkownParam = 10

16.7.2.5 ENUMCodeGenErrorLevel

Valores de enumeración para identificar la severidad de los mensajes de generación de código.

Miembros

eCodeGenErrorLevel_Information = 0

eCodeGenErrorLevel_Warning = 1

eCodeGenErrorLevel_Error = 2

eCodeGenErrorLevel_Undefined = 3

16.7.2.6 ENUMComponentDatapointSide

Valores de enumeración para indicar el lado de un punto de datos en su componente. Véase también `Component.GetRootDatapoint`.

Miembros

eDatapointSideInput = 0

eDatapointSideOutput = 1

16.7.2.7 ENUMComponentSubType

Valores de enumeración para indicar subtipos de los componentes.

 © 2016-2022 Altova GmbH

Miembros

eComponentSubType_None = 0

eComponentSubType_Text_EDI = 1

eComponentSubType_Text_Flex = 2

eComponentSubType_Text_CSVFLF = 3

16.7.2.8 ENUMComponentType

Valores de enumeración para indicar tipos de los componentes.

Miembros

eComponentType_Unknown = 0

eComponentType_XML = 1

eComponentType_DB = 2

eComponentType_Text = 3

eComponentType_Excel = 4

eComponentType_WSDL = 5

eComponentType_XBRL = 6

eComponentType_Input = 7

eComponentType_JSON = 8

16.7.2.9 ENUMComponentUsageKind

Valores de enumeración para indicar tipo de uso de los componentes.

Miembros

eComponentUsageKind_Unknown = 0

eComponentUsageKind_Instance = 1

eComponentUsageKind_Input = 2

eComponentUsageKind_Output = 3

eComponentUsageKind_Variable = 4

eComponentUsageKind_String = 5

16.7.2.10 ENUMConnectionType

Valores de enumeración para indicar el tipo de una conexión. Véase también `Connection.ConnectionType`.

Miembros

eConnectionTypeTargetDriven = 0

eConnectionTypeSourceDriven = 1

eConnectionTypeCopyAll = 2

16.7.2.11 ENUMDOMType

Valores de enumeración para especificar el tipo de DOM usado por el código de asignación C++ generado.

NOTA: El valor `eDOMType_xerces` está obsoleto. `eDOMType_xerces3` indica que se usa Xerces 3.x. En este contexto, obsoleto significa que este valor no es compatible y no se debería usar.

Miembros

eDOMType_xerces = 1 (obsolete)

eDOMType_xerces3 = 2

eDOMType_msxml6 = 3

16.7.2.12 ENUMLibType

Valores de enumeración para especificar el tipo de biblioteca usado por el código de asignación C++ generado.

Miembros

eLibType_static = 0

eLibType_dll = 1

16.7.2.13 ENUMProgrammingLanguage

Valores de enumeración para seleccionar un lenguaje de programación.

Miembros

eUndefinedLanguage = -1

© 2016-2022 Altova GmbH

eJava = 0

eCpp = 1

eCSharp = 2

eXSLT = 3

eXSLT2 = 4

eXQuery = 5

eXSLT3 = 6

16.7.2.14 ENUMProjectItemType

Enumeración para identificar los distintos tipos de elementos de proyecto que pueden ser secundarios de `Project` o `ProjectItems` de tipo carpeta. Véase también `ProjectItem.Kind`.

Miembros

eProjectItemType_MappingFolder = 0

eProjectItemType_Mapping = 1

eProjectItemType_WebServiceFolder = 2

eProjectItemType_WebServiceRoot = 3

eProjectItemType_WebServiceService = 4

eProjectItemType_WebServicePort = 5

eProjectItemType_WebServiceOperation = 6

eProjectItemType_ExternalFolder = 7

eProjectItemType_LibraryFolder = 8

eProjectItemType_ResourceFolder = 9

eProjectItemType_VirtualFolder = 10

eProjectItemType_Count = 11

eProjectItemType_Invalid = -1

16.7.2.15 ENUMProjectType

Valores de enumeración para seleccionar un tipo de proyecto para los códigos de asignación C# y C++ generados.

Miembros

eVisualStudio2010Project = 6

eVisualStudio2013Project = 7

eVisualStudio2015Project = 8

eVisualStudio2017Project = 9

eVisualStudio2019Project = 10

eDotNetCore3_1 = 11

eDotNet5_0 = 12

16.7.2.16 ENUMSearchDatapointFlags

Valores de enumeración usados como banderas de bits; se pueden usar como combinación de banderas al buscar un punto de datos. Véase también `GetChild`.

Miembros

eSearchDatapointElement = 1

eSearchDatapointAttribute = 2

16.7.2.17 ENUMViewMode

Valores de enumeración para seleccionar una vista de MapForce.

Miembros

eMapForceView = 0

eXSLView = 1

eOutputView = 2

© 2016-2022 Altova GmbH

17 ActiveX Integration

The MapForce user interface and the functionality described in this section can be integrated into custom applications that can consume ActiveX controls. ActiveX technology enables a wide variety of languages to be used for integration, such as C++, C#, VB.NET, HTML. (Note that ActiveX components integrated in HTML must be run with Microsoft Internet Explorer versions and platforms that support ActiveX). All components are full OLE Controls. Integration into Java is provided through wrapper classes.

> To integrate the ActiveX controls into your custom code, the MapForce Integration Package must be installed (see https://www.altova.com/components/download). Ensure that you install MapForce first, and then the MapForce Integration Package. Other prerequisites apply, depending on language and platform (see Prerequisites 1420).

You can flexibly choose between two different levels of integration: application level and document level.

Integration at application level means embedding the complete interface of MapForce (including its menus, toolbars, panes, etc) as an ActiveX control into your custom application. For example, in the most simple scenario, your custom application could consist of only one form that embeds the MapForce graphical user interface. This approach is easier to implement than integration at document level but may not be suitable if you need flexibility to configure the MapForce graphical user interface according to your custom requirements.

Integration at document level means embedding MapForce into your own application piece-by-piece. This includes implementing not only the main MapForce control but also the main document editor window, and, optionally, any additional windows. This approach provides greater flexibility to configure the GUI, but requires advanced interaction with ActiveX controls in your language of choice.

The sections Integration at the Application Level 1424 and Integration at Document Level 1427 describe the key steps at these respective levels. The ActiveX Integration Examples 1431 section provides examples in C#, HTML, and Java. Looking through these examples will help you to make the right decisions quickly. The Object Reference 1459 section describes all COM objects that can be used for integration, together with their properties and methods.

For information about using MapForce as a Visual Studio plug-in, see MapForce in Visual Studio 999.

17.1 Prerequisites

To integrate the MapForce ActiveX control into a custom application, the following must be installed on your computer:

- MapForce
- The MapForce Integration Package, available for download at https://www.altova.com/components/download

To integrate the 64-bit ActiveX control, install the 64-bit versions of MapForce and MapForce Integration Package. For applications developed under Microsoft .NET platform with Visual Studio, both the 32-bit and 64-bit versions of MapForce and MapForce Integration Package must be installed, as explained below.

Microsoft .NET (C#, VB.NET) with Visual Studio

To integrate the MapForce ActiveX control into a 32-bit application developed under Microsoft .NET, the following must be installed on your computer:

- Microsoft .NET Framework 4.0 or later
- Visual Studio 2012/2013/2015/2017/2019/2022
- MapForce 32-bit and MapForce Integration Package 32-bit
- The ActiveX controls must be added to the Visual Studio toolbox (see Adding the ActiveX Controls to the Toolbox 1422).

If you want to integrate the 64-bit ActiveX control, the following prerequisites apply in addition to the ones above:

- MapForce 32-bit and MapForce Integration Package 32-bit must still be installed (this is required to provide the 32-bit ActiveX control to the Visual Studio designer, since Visual Studio runs on 32-bit)
- MapForce 64-bit and MapForce Integration Package 64-bit must be installed (provides the actual 64-bit ActiveX control to your custom application at runtime)
- In Visual Studio, create a 64-bit build configuration and build your application using this configuration. For an example, see Running the Sample C# Solution 1431.

Java

To integrate the MapForce ActiveX control into Java application using the Eclipse development environment, the following must be installed on your computer:

- Java Runtime Environment (JRE) or Java Development Kit (JDK) 7 or later
- Eclipse
- MapForce and MapForce Integration Package

Note: To run the 64-bit version of the MapForce ActiveX control, use a 64-bit version of Eclipse, as well as the 64-bit version of MapForce and the MapForce Integration Package.

MapForce integration and deployment on client computers

If you create a .NET application and intend to distribute it to other clients, you will need to install the following on the client computer(s):

 © 2016-2022 Altova GmbH

- MapForce
- The MapForce Integration Package
- The custom integration code or application.

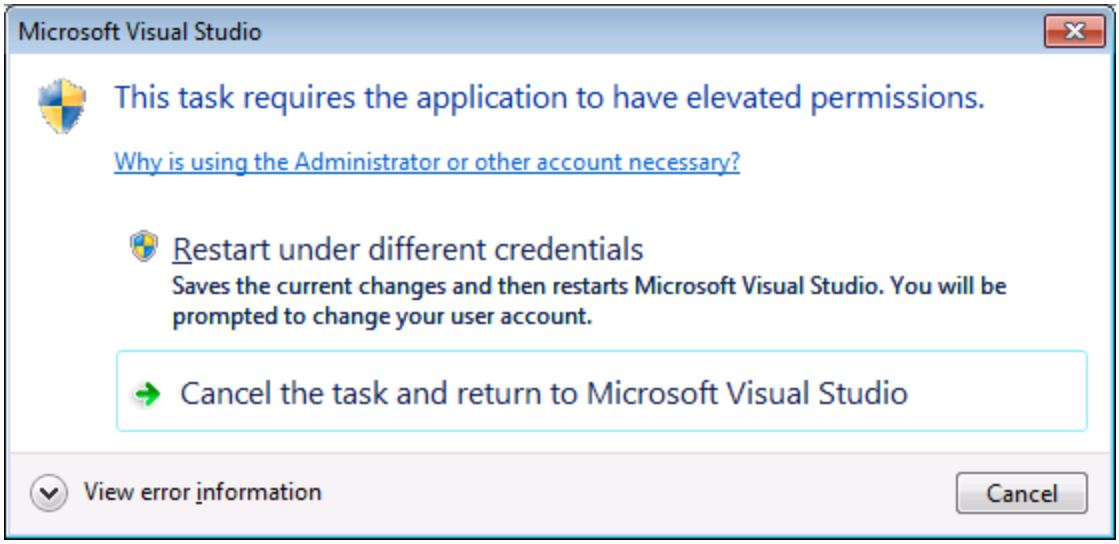
17.2 Adding the ActiveX Controls to the Toolbox

To use the MapForce ActiveX controls in an application developed with Visual Studio, the controls must first be added to the Visual Studio Toolbox, as follows:

1. On the **Tools** menu of Visual Studio, click **Choose Toolbox Items**.
2. On the **COM Components** tab, select the check boxes next to the MapForceControl, MapForceControl Document, and MapForceControl Placeholder.

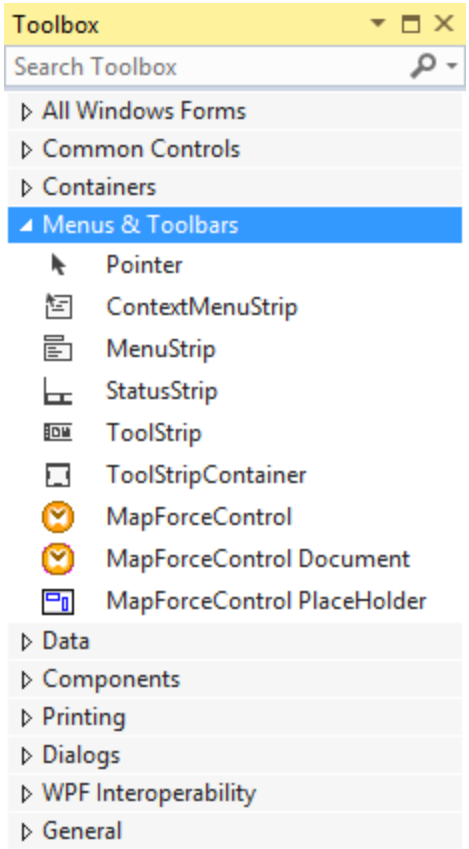
In case the controls above are not available, follow the steps below:

1. On the **COM Components** tab, click **Browse**, and select the **MapForceControl.ocx** file from the MapForce installation folder. Remember that the MapForce Integration Package must be installed; otherwise, this file is not available, see Prerequisites 1420.
2. If prompted to restart Visual Studio with elevated permissions, click **Restart under different credentials**.



If the steps above were successful, the MapForce ActiveX controls become available in the Visual Studio Toolbox.

© 2016-2022 Altova GmbH



Note: For an application-level integration, only the **MapForceControl** ActiveX control is used (see Integration at Application Level 1424). The **MapForceControl Document** and **MapForceControl Placeholder** controls are used for document-level integration (see Integration at Document Level 1427).

17.3 Integration at Application Level

Integration at application level allows you to embed the complete interface of MapForce into a window of your application. With this type of integration, you get the whole user interface of MapForce, including all menus, toolbars, the status bar, document windows, and helper windows. Customization of the application's user interface is restricted to what MapForce provides. This includes rearrangement and resizing of helper windows and customization of menus and toolbars.

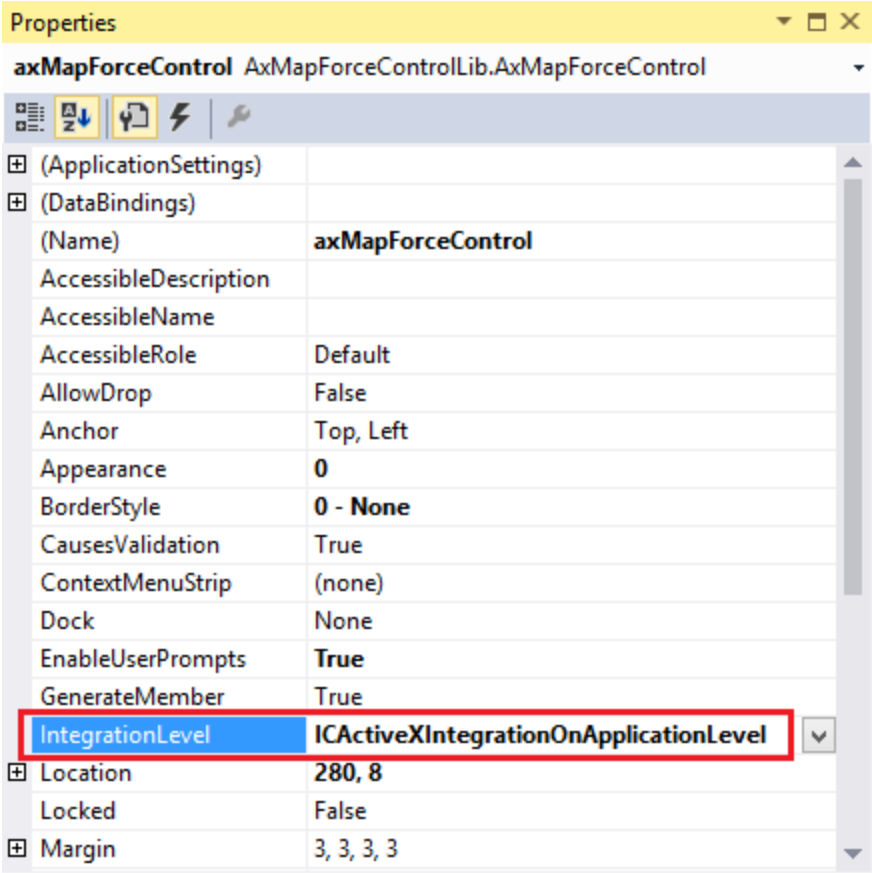
The only ActiveX control you need to integrate is `MapForceControl`[1462]. Do not instantiate or access `MapForceControlDocument`[1470] or `MapForceControlPlaceHolder`[1476] ActiveX controls when integrating at application-level.

If you have any initialization to do or if you want to automate some behaviour of MapForce, use the properties, methods, and events described for `MapForceControl`[1462]. Consider using `MapForceControl.Application`[1463] for more complex access to MapForce functionality.

In C# or VB.NET with Visual Studio, the steps to create a basic, one-form application which integrates the MapForce ActiveX controls at application level are as follows:

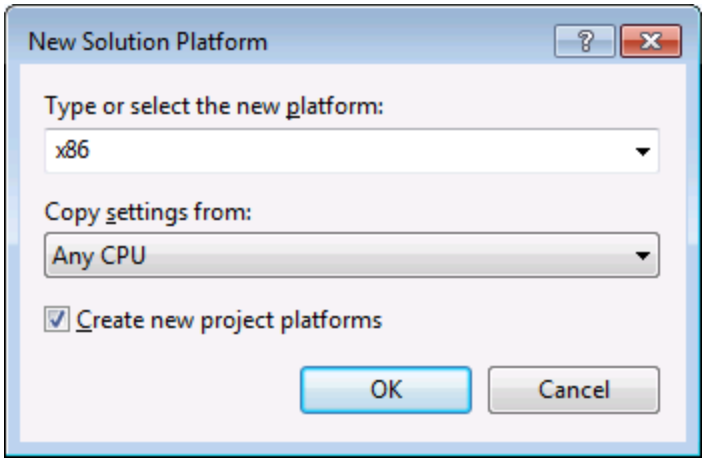
1. Check that all prerequisites are met (see Prerequisites[1420]).
2. Create a new Visual Studio Windows Forms project with a new empty form.
3. If you have not done that already, add the ActiveX controls to the toolbox (see Adding the ActiveX Controls to the Toolbox[1422]).
4. Drag the **MapForceControl** from the toolbox onto your new form.
5. Select the **MapForceControl** on the form, and, in the Properties window, set the **IntegrationLevel** property to **ICActiveXIntegrationOnApplicationLevel**.

© 2016-2022 Altova GmbH



6. Create a build platform configuration that matches the platform under which you want to build (x86, x64). Here is how you can create the build configuration:

   a. Right-click the solution in Visual Studio, and select **Configuration Manager**.
   b. Under **Active solution platform**, select **New...** and then select the x86 or x64 configuration (in this example, **x86**).



You are now ready to build and run the solution in Visual Studio. Remember to build using the configuration that matches your target platform (x86, x64).

© 2016-2022 Altova GmbH

17.4 Integration at Document Level

Compared to integration at application level, integration at document level is a more complex, yet more flexible way to embed MapForce functionality into your application by means of ActiveX controls. With this approach, your code can access selectively the following parts of the MapForce user interface:

- Document editing window
- Project window
- Libraries window
- Overview window
- Messages window

As mentioned in Integration at Application Level 1424, for an ActiveX integration at application level, only one control is required, namely the **MapForceControl**. However, for an ActiveX integration at document level, MapForce functionality is provided by the following ActiveX controls:

- MapForceControl 1462
- MapForceControl Document 1470
- MapForceControl Placeholder 1476

These controls are supplied by the **MapForceControl.ocx** file available in the application installation folder of MapForce. When you develop the ActiveX integration with Visual Studio, you will need to add these controls to the Visual Studio toolbox (see Adding the ActiveX Controls to the Toolbox 1422).

The basic steps to integrate the ActiveX controls at document level into your application are as follows:

1. First, instantiate **MapForceControl** in your application. Instantiating this control is mandatory; it enables support for the **MapForceControl Document** and **MapForceControl Placeholder** controls mentioned above. It is important to set the `IntegrationLevel` 1464 property to **ICActiveXIntegrationOnDocumentLevel** (or "1"). To hide the control from the user, set its **Visible** property to **False**.

Note: When integrating at document level, do not use the **Open** method of the **MapForceControl**; this might lead to unexpected results. Use the corresponding open methods of **MapForceControl Document** and **MapForceControl PlaceHolder** instead.

2. Create at least one instance of **MapForceControl Document** in your application. This control supplies the document editing window of MapForce to your application and can be instantiated multiple times if necessary.

   Use the method **Open** to load any existing file. To access document-related functionality, use the **Path** and **Save** or methods and properties accessible via the property **Document**.

Note: The control does not support a read-only mode. The value of the property **ReadOnly** is ignored.

3. Optionally, add to your application the **MapForceControl Placeholder** control for each additional window (other than the document window) that must be available to your application.

   Instances of **MapForceControl PlaceHolder** allow you to selectively embed additional windows of MapForce into your application. The window kind (for example, Project window) is defined by the

property **PlaceholderWindowID**. Therefore, to set the window kind, set the property **PlaceholderWindowID**. For valid window identifiers, see MapForceControlPlaceholderWindow[1479].

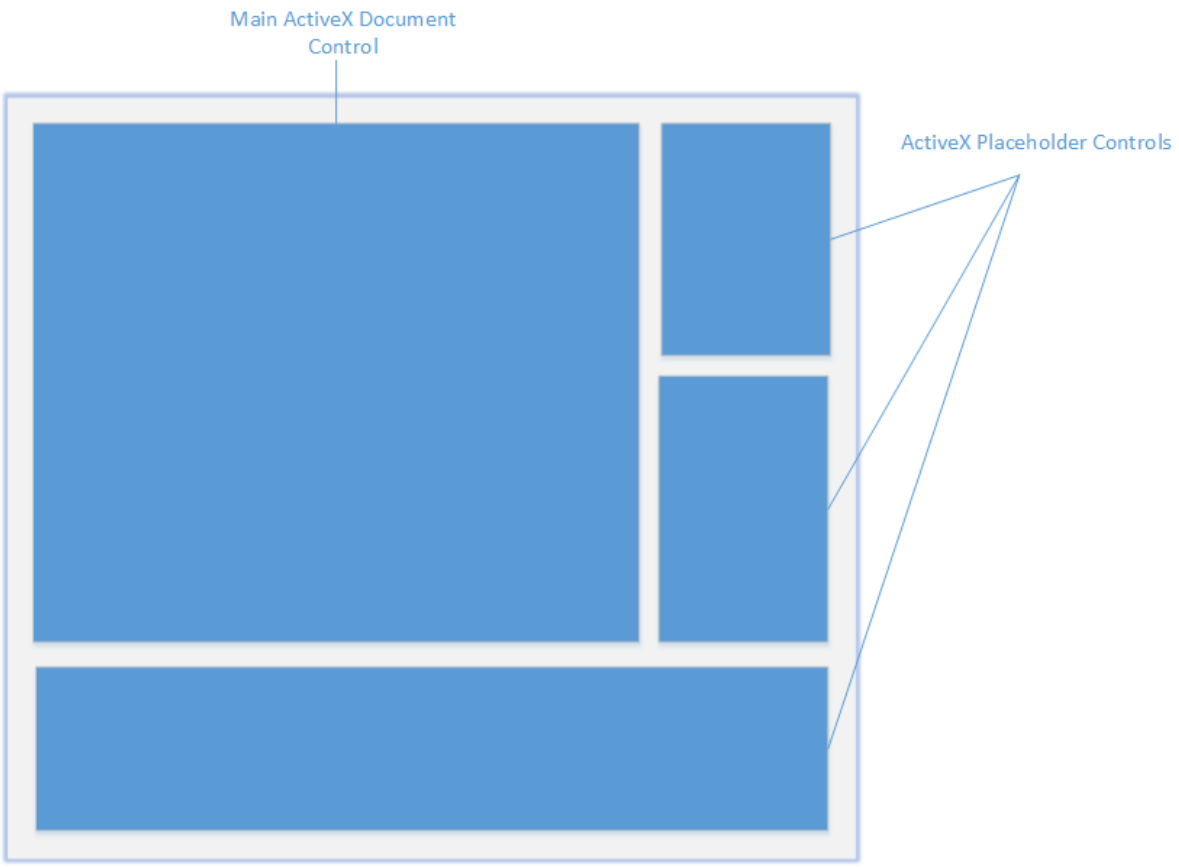
Note: Use only one **MapForceControl PlaceHolder** for each window identifier.

For placeholder controls that select the MapForce project window, additional methods are available. Use **OpenProject** to load a MapForce project. Use the property Project and the methods and properties from the MapForce automation interface to perform any other project related operations.

For example, in C# or VB.NET with Visual Studio, the steps to create a basic, one-form application which integrates the MapForce ActiveX controls at document level could be similar to those listed below. Note that your application may be more complex if necessary; however, the instructions below are important to understand the minimum requirements for an ActiveX integration at document level.

1. Create a new Visual Studio Windows Forms project with a new empty form.
2. If you have not done that already, add the ActiveX controls to the toolbox (see Adding the ActiveX Controls to the Toolbox[1422]).
3. Drag the **MapForceControl**[1462] from the toolbox onto your new form.
4. Set the **IntegrationLevel** property of the **MapForceControl** to **ICActiveXIntegrationOnDocumentLevel**, and the **Visible** property to **False**. You can do this either from code or from the **Properties** window.
5. Drag the **MapForceControl Document**[1470] from the toolbox onto the form. This control provides the main document window of MapForce to your application, so you may need to resize it to a reasonable size for a document.
6. Optionally, add one or more **MapForceControl Placeholder**[1476] controls to the form (one for each additional window type that your application needs, for example, the **Project** window). You will typically want to place such additional placeholder controls either below or to the right or left of the main document control, for example:

© 2016-2022 Altova GmbH



7. Set the **PlaceholderWindowID** property of each **MapForceControl Placeholder** control to a valid window identifier. For the list of valid values, see MapForceControlPlaceholderWindow[1479].
8. Add commands to your application (at minimum, you will need to open, save and close documents), as shown below.

Querying MapForce Commands

When you integrate at document level, no MapForce menu or toolbar is available to your application. Instead, you can retrieve the required commands, view their status, and execute them programmatically, as follows:

- To retrieve all available commands, use the `CommandsList`[1463] property of the **MapForceControl**.
- To retrieve commands organized according to their menu structure, use the `MainMenu`[1464] property.
- To retrieve commands organized by the toolbar in which they appear, use the `Toolbars`[1465] property.
- To send commands to MapForce, use the `Exec`[1466] method.
- To query if a command is currently enabled or disabled, use the `QueryStatus`[1467] method.

This enables you to flexibly integrate MapForce commands into your application's menus and toolbars.

Your installation of MapForce also provides you with command label images used within MapForce. See the folder **<ApplicationFolder>\Examples\ActiveX\Images** of your MapForce installation for icons in GIF format. The file names correspond to the command names as they are listed in the Command Reference[1450] section.

General considerations

To automate the behaviour of MapForce, use the properties, methods, and events described for the **MapForceControl**[1462], **MapForceControl Document**[1470], and **MapForceControl Placeholder**[1476].

For more complex access to MapForce functionality, consider using the following properties:

- `MapForceControl.Application`[1463]
- `MapForceControlDocument.Document`[1471]
- `MapForceControlPlaceHolder.Project`[1477]

These properties give you access to the MapForce automation interface (MapForceAPI)

Note: To open a document, always use `MapForceControlDocument.Open`[1473] or `MapForceControlDocument.New`[1472] on the appropriate document control. To open a project, always use `MapForceControlPlaceHolder.OpenProject`[1477] on a placeholder control embedding a MapForce project window.

For examples that show how to instantiate and access the necessary controls in different programming environments, see ActiveX Integration Examples[1431].

© 2016-2022 Altova GmbH

17.5 ActiveX Integration Examples

This section contains examples of MapForce document-level integration using different container environments and programming languages. Source code for all examples is available in the folder `<ApplicationFolder>\Examples\ActiveX` of your MapForce installation.

17.5.1 C#

A basic ActiveX integration example solution for C# and Visual Studio is available in the folder `<ApplicationFolder>\Examples\ActiveX\C#`. Before you compile the source code and run the sample, make sure that all prerequisites are met (see Running the Sample C# Solution 1431).

17.5.1.1 Running the Sample C# Solution

The sample Visual Studio solution available in the folder **<ApplicationFolder>\Examples\ActiveX\C#** illustrates how to consume the MapForce ActiveX controls. Before attempting to build and run this solution, note the following steps:

Step 1: Check the prerequisites

Visual Studio 2010 or later is required to open the sample solution. For the complete list of prerequisites, see Prerequisites 1420.

Step 2: Copy the sample to a directory where you have write permissions

To avoid running Visual Studio as an Administrator, copy the source code to a directory where you have write permissions, instead of running it from the default location.

Step 3: Check and set all required control properties

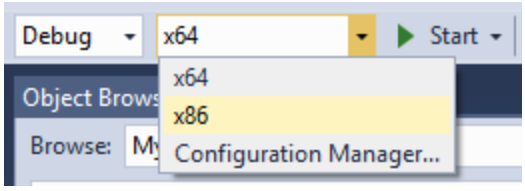
The sample application contains one instance of MapForceControlDocument 1470 and several instances of MapForceControlPlaceHolder 1476 controls. Double-check that the following properties of these controls are set as shown in the table below:

Control name	Property	Property value
axMapForceControl	IntegrationLevel	ICActiveXIntegrationOnDocumentLevel
axMapForceControlLibrary	PlaceholderWindowID	0
axMapForceControlOutput	PlaceholderWindowID	2
axMapForceControlPreview	PlaceholderWindowID	1

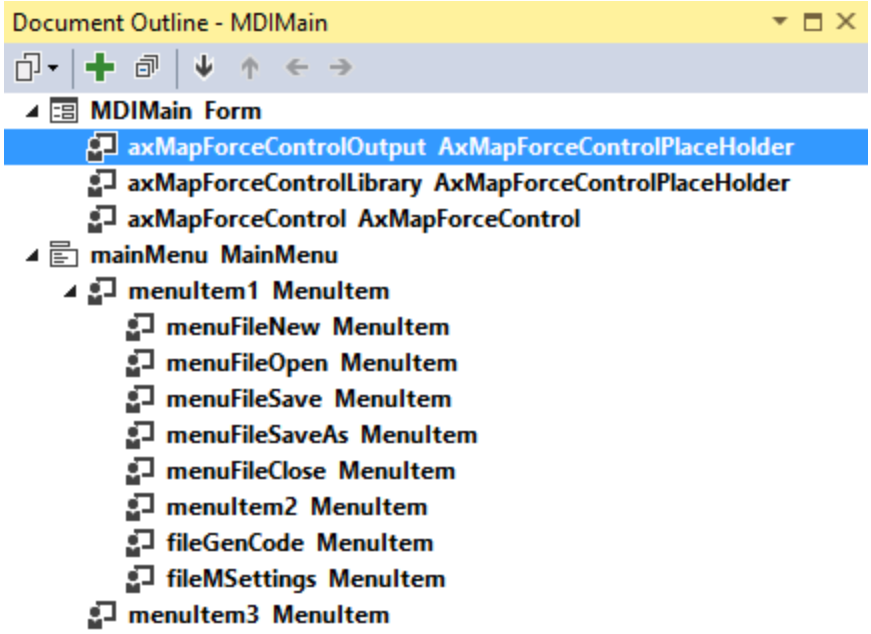
Here is how you can view or set the properties of an ActiveX control:

1. Open the **MDIMain.cs** form in the designer window.

Note: On 64-bit Windows, it may be necessary to change the build configuration of the Visual Studio solution to "x86" **before** opening the designer window. If you need to build the sample as a 64-bit application, see Prerequisites 1420.

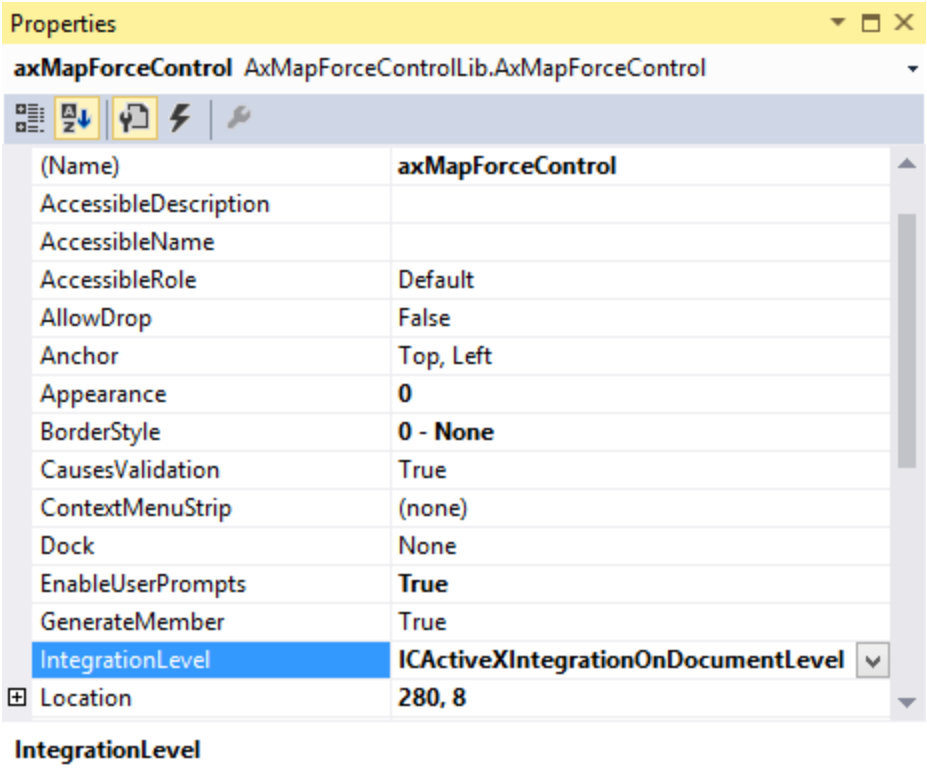


2. Open the **Document Outline** window of Visual Studio (On the **View** menu, click **Other Windows | Document Outline**).



3. Click an ActiveX control in the **Document Outline** window, and edit its required property in the **Properties** window, for example:

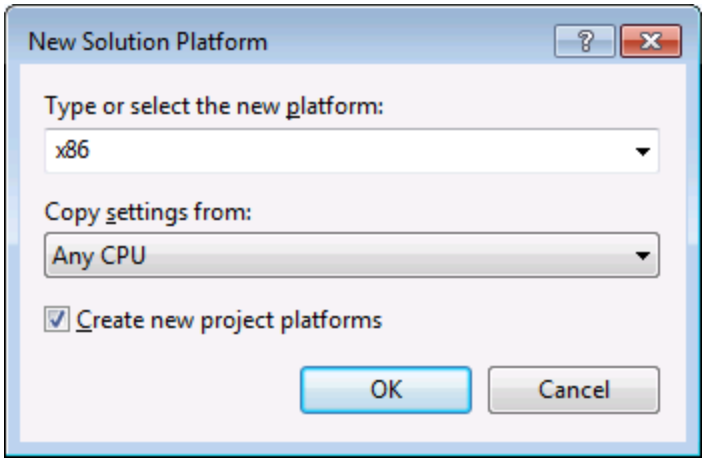
 © 2016-2022 Altova GmbH



Step 4: Set the build platform

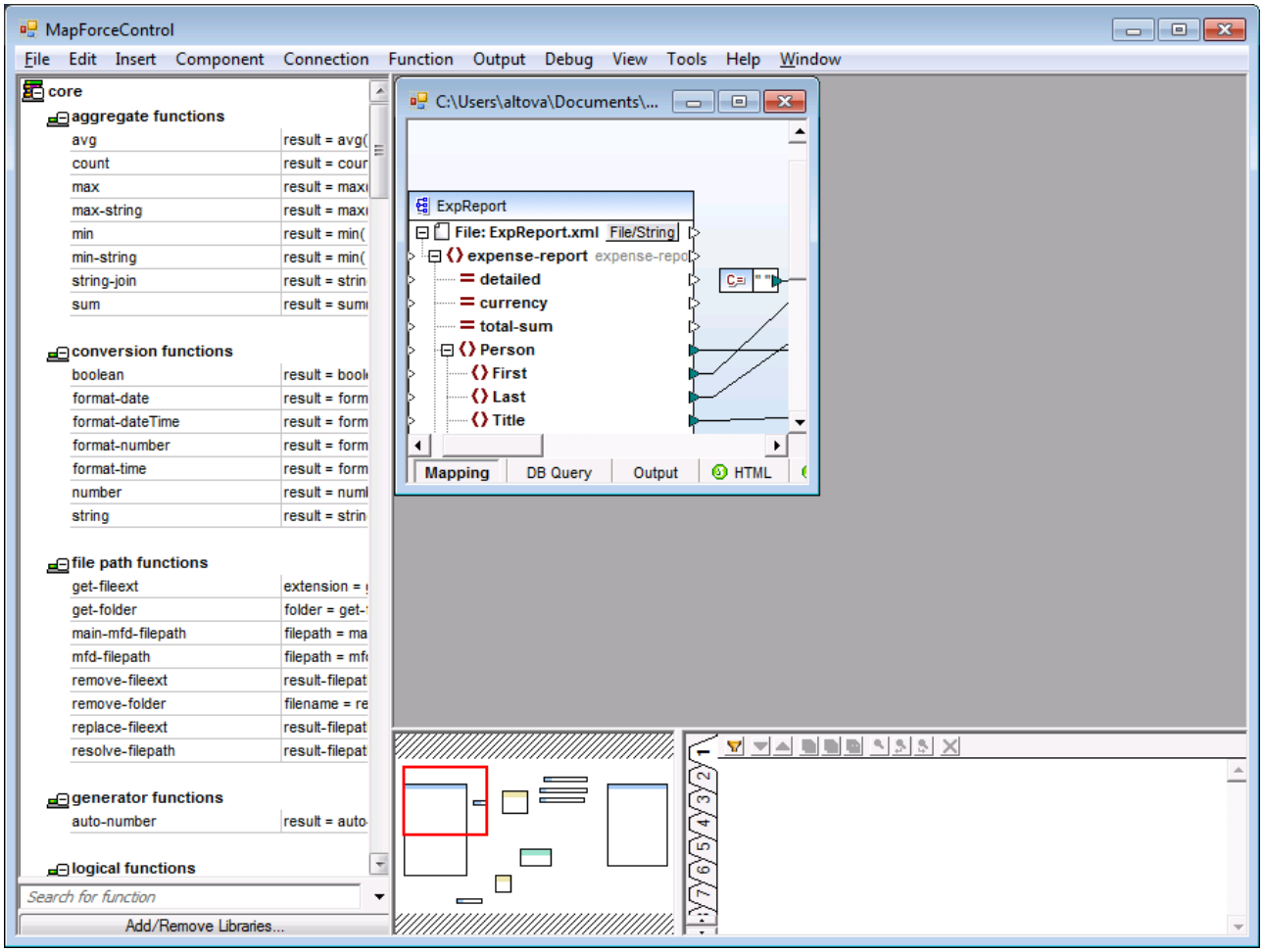
- Create a build platform configuration that matches the platform under which you want to build (x86, x64). Here is how you can create the build configuration:

  a. Right-click the solution in Visual Studio, and select **Configuration Manager**.
  b. Under **Active solution platform**, select **New...** and then select the x86 or x64 configuration (in this example, **x86**).



You are now ready to build and run the solution in Visual Studio. Remember to build using the configuration that matches your target platform (x86, x64); otherwise, runtime errors might occur.

On running the sample, the main MDI Frame window is displayed. Use **File | Open** to open a mapping file (for example, `MarketingExpenses.mfd`, which is in the MapForce examples folder). The file is loaded and displayed in a new document child window:



After you load the document, you can execute commands against the active document using the menu. Context menus are also available. You can also load additional documents. Save any modifications using the **File | Save** command.

17.5.1.2 Retrieving Command Information

The MapForceControl gives access to all commands of MapForce through its `CommandsList` , `MainMenu`, and `Toolbars` properties. The example project available in the folder **<ApplicationFolder>\Examples\ActiveX\C#** uses the `MainMenu` property to create the MapForce menu structure dynamically.

The code that gets the menu commands can be found in the `MDIMain` method in `MDIMain.cs` file:

```
public MDIMain()
{
```

```
    // ...

    // Get the MainMenu property of the control and create the menu structure from it.
    MFLib.MapForceCommand objCommand = this.axMapForceControl.MainMenu;
    InsertMenuStructure(mainMenu, objCommand);
}
```

In the code listing above, `mainMenu` is the existing static menu of the main MDI Frame window. If you open the **MDIMain.cs** form in the Visual Studio Designer, you will notice that this menu contains two menu items: **File** and **Window**.

MDIMain.cs

The method `InsertMenuStructure` takes as parameters the `mainMenu` and the `objCommand` objects (the former is the existing static menu, while the latter contains the full menu structure retrieved from the MapForce ActiveX control). The retrieved MapForce menu structure is then merged into the existing static menu. Note that the menus **File**, **Project**, and **Window** are not added dynamically. This is intentional, because these menus deal with actively open documents, and they would require code which is beyond the scope of this example. The basic file management commands (create, open, save, bring into focus) are handled by the existing static menus **File** and **Window**. All other menus are inserted dynamically based on the information taken from the `MainMenu` property of the ActiveX control. The new menus are inserted after "File" but before "Window", i.e. starting at menu index 1.

The method `InsertMenuStructure` iterates through all top-level menus found in `MapForceCommand` object and adds a new menu item for each. Since each top-level menu has its own child menu items, a call to the method `InsertMenuCommand` takes place for each encountered child menu item. Furthermore, since each child menu item can have its own children menu items, and so on, the `InsertMenuCommand` method recurses into itself until no more child menu items exist.

The commands added dynamically are instances of the class `CustomMenuItem`, which is defined in `CustomMenuItem.cs`. This class is derived from `System.Windows.Forms.MenuItem` class and has an additional member to store the MapForce command ID.

```
public class CustomMenuItem : System.Windows.Forms.MenuItem
{
        public int m_MapForceCmdID;
}
```

All dynamically added commands (except those that are containers for other commands) get the same event handler `AltovaMenuItem_Click` which does the processing of the command:

```
private void AltovaMenuItem_Click(object sender, EventArgs e)
{
  if(sender.GetType() == System.Type.GetType("MapForceApplication.CustomMenuItem"))
  {
    CustomMenuItem   customItem = (CustomMenuItem)sender;
    ProcessCommand(customItem.m_MapForceCmdID);
  }
}
```

If the command is a container for other commands (that is, if it has child commands), it gets the event handler `AltovaSubMenu_Popup`. This handler queries the status of each child command and enables or disables it as required. This ensures that each command is enabled only when that is meaningful (for example, the **File | Save** menu item should be disabled if there is no active document open).

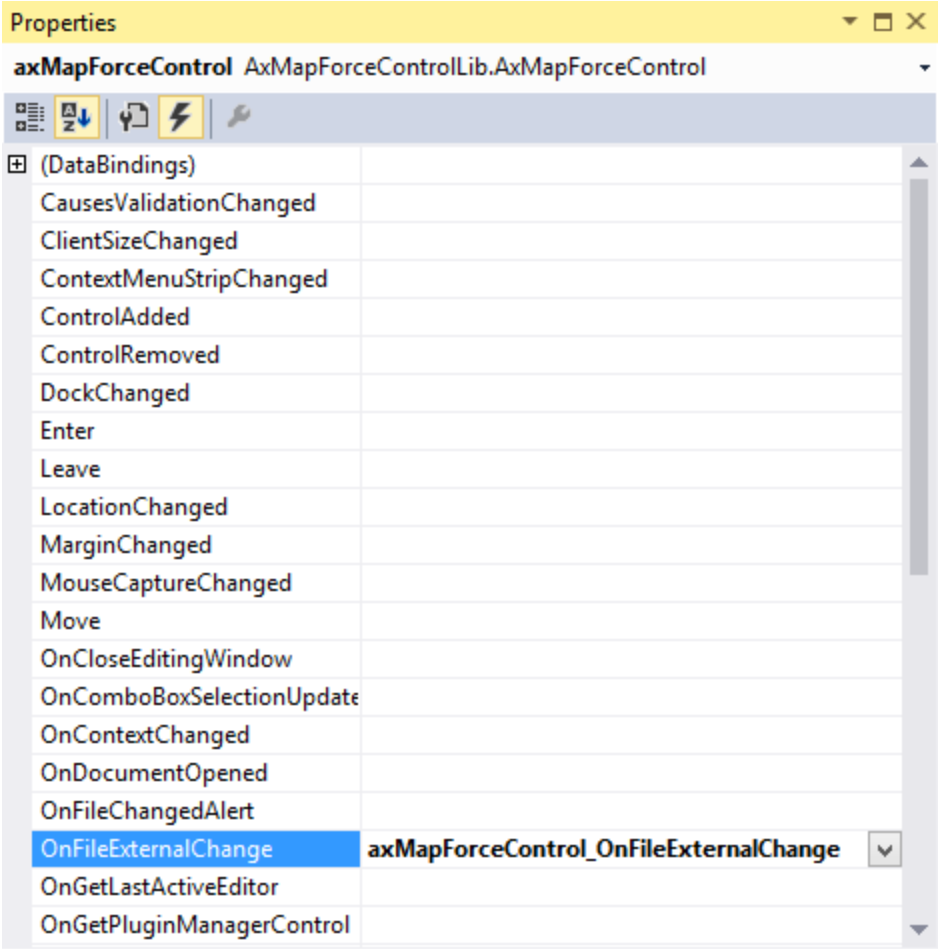
The method `ProcessCommand` delegates the execution either to the MapForceControl itself or to any active MapForce document loaded in a `MapForceControlDocument` control. This is necessary because the MapForceControl has no way to know which document is currently active in the hosting application.

```
private void ProcessCommand(int nID)
{
   MapForceDoc docMapForce = GetCurrentMapForceDoc();

   if(docMapForce != null)
      docMapForce.axMapForceControlDoc.Exec(nID);
   else
      axMapForceControl.Exec(nID);
}
```

17.5.1.3 Handling Events

Because all events in the MapForce library are based on connection points, you can use the C# delegate mechanism to provide the custom event handlers. You will always find a complete list of events on the property page of each control of the MapForce library. The image below shows the events of the main MapForceControl:

© 2016-2022 Altova GmbH



As you can see, the example project only overrides the `OnFileExternalChange` event. The creation of the C# delegate is done for you by the C# Framework. All you need to do is fill in the empty event handler.

For example, the handler implementation shown below turns off any file reloading and displays a message box to inform the user that a file loaded by the MapForceControl has been changed from outside:

```
private void axMapForceControl_OnFileExternalChange(object sender,
AxMapForceControlLib._DMapForceControlEvents_OnFileExternalChangeEvent e)
{
    MessageBox.Show("Attention: The file " + e.strPath + " has been changed from
outside\nbut reloading is turned off in the sample application!");

    // This turns off any file reloading:
    e.varRet = false;
}
```

17.5.2 Java

MapForce ActiveX components can be accessed from Java code. Java integration is provided by the libraries listed below. These libraries are available in the folder `<ApplicationFolder>\Examples\JavaAPI` of your MapForce installation, after you have installed both MapForce and the MapForce Integration Package (see also Prerequisites[1420]).

- `AltovaAutomation.dll`: a JNI wrapper for Altova automation servers (in case of the 32-bit installation of MapForce)
- `AltovaAutomation_x64.dll`: a JNI wrapper for Altova automation servers (in case of the 64-bit installation of MapForce)
- `AltovaAutomation.jar`: Java classes to access Altova automation servers
- `MapForceActiveX.jar`: Java classes that wrap the MapForce ActiveX interface
- `MapForceActiveX_JavaDoc.zip`: a Javadoc file containing help documentation for the Java interface

Note: In order to use the Java ActiveX integration, the .dll and .jar files must be included in the Java class search path.

Example Java project

An example Java project is supplied with your product installation. You can test the Java project and modify and use it as you like. For more details, see Example Java Project[1439].

Rules for mapping the ActiveX Control names to Java

For the documentation of ActiveX controls, see Object Reference[1459]. Note that the object naming conventions are slightly different in Java compared to other languages. Namely, the rules for mapping between the ActiveX controls and the Java wrapper are as follows:

- ***Classes and class names***
  For every component of the MapForce ActiveX interface a Java class exists with the name of the component.

- ***Method names***
  Method names on the Java interface are the same as used on the COM interfaces but start with a small letter to conform to Java naming conventions. To access COM properties, Java methods that prefix the property name with get and set can be used. If a property does not support write-access, no setter method is available. Example: For the `IntegrationLevel` property of the MapForceControl, the Java methods `getIntegrationLevel` and `setIntegrationLevel` are available.

- ***Enumerations***
  For every enumeration defined in the ActiveX interface, a Java enumeration is defined with the same name and values.

- ***Events and event handlers***
  For every interface in the automation interface that supports events, a Java interface with the same name plus 'Event' is available. To simplify the overloading of single events, a Java class with default implementations for all events is provided. The name of this Java class is the name of the event interface plus '`DefaultHandler`'. For example:
  `MapForceControl`: Java class to access the application
  `MapForceControlEvents`: Events interface for the MapForceControl

© 2016-2022 Altova GmbH

`MapForceControlEventsDefaultHandler`: Default handler for `MapForceControlEvents`

Exceptions to mapping rules

There are some exceptions to the rules listed above. These are listed below:

Interface	Changes in Java class
`MapForceControlDocument`, method `New`	Renamed to `newDocument`
`MapForceControlDocument`, method `OpenDocument`	Removed. Use the `Open` method
`MapForceControlDocument`, method `NewDocument`	Removed. Use the `newDocument` method
`MapForceControlDocument`, method `SaveDocument`	Removed. Use the `Save` method

This section

This section shows how some basic MapForce ActiveX functionality can be accessed from Java code. It is organized into the following sub-sections:

- Example Java Project 1439
- Creating the ActiveX Controls 1441
- Loading Data in the Controls 1442
- Basic Event Handling 1442
- Menus 1443
- UI Update Event Handling 1445
- Creating a MapForce Mapping Table 1445

17.5.2.1 Example Java Project

The MapForce installation package contains an example Java project, located in the ActiveX Examples folder of the application folder: `<ApplicationFolder>\Examples\ActiveX\Java\`.

The Java example shows how to integrate the MapForceControl in a common desktop application created with Java. You can test it directly from the command line using the batch file `BuildAndRun.bat`, or you can compile and run the example project from within Eclipse. See below for instructions on how to use these procedures.

File list

The Java examples folder contains all the files required to run the example project. These files are listed below:

`.classpath`	Eclipse project helper file
`.project`	Eclipse project file
`AltovaAutomation.dll`	Java-COM bridge: DLL part (for the 32-bit installation)
`AltovaAutomation_x64.dll`	Java-COM bridge: DLL part (for the 64-bit installation)

`AltovaAutomation.jar`	Java-COM bridge: Java library part
`BuildAndRun.bat`	Batch file to compile and run example code from the command line prompt. Expects folder where Java Virtual Machine resides as parameter.
`MapForceActiveX.jar`	Java classes of the MapForce ActiveX control
`MapForceActiveX_JavaDoc.zip`	Javadoc file containing help documentation for the Java API
`MapForceContainer.java`	Java example source code
`MapForceContainerEventHandler.java`	Java example source code
`MapForceTable.java`	Java example source code

What the example does

The example places one MapForce document editor window, the MapForce project window, the MapForce library window and the MapForce validation window in an AWT frame window. It reads out the main menu defined for MapForce and creates an AWT menu with the same structure. You can use this menu or the project window to open and work with files in the document editor.

You can modify the example in any way you like.

The following specific features are described in code listings:

- Creating the ActiveX Controls 1441: Starts MapForce, which is registered as an automation server, or activates MapForce if it is already running.
- Loading Data in the Controls 1442: Locates one of the example documents installed with MapForce and opens it.
- Basic Event Handling 1442: Changes the view of all open documents to Text View. The code also shows how to iterate through open documents.
- Menus 1443: Validates the active document and shows the result in a message box. The code shows how to use output parameters.
- UI Update Event Handling 1445: Shows how to handle MapForce events.
- Creating a MapForce Mapping Table 1445: Shows how to create a MapForce mapping table and prepare it for modal activation.

Updating the path to the Examples folder

Before running the provided sample, you may need to edit the **MapForceContainer.java** file. Namely, check that the following path refers to the actual folder where the MapForce example files are stored on your operating system:

```
// Locate samples installed with the product.
final String strExamplesFolder = System.getenv( "USERPROFILE" ) + "\\Documents\\Altova\
\MapForce2023\\MapForceExamples\\";
```

Running the example from the command line

To run the example from the command line:

 © 2016-2022 Altova GmbH

1. Check that all prerequisites are met (see Prerequisites[1420]).
2. Open a command prompt window, change the current directory to the sample Java project folder, and type:

```
buildAndRun.bat "<Path-to-the-Java-bin-folder>"
```

3. Press **Enter**.

The Java source in `MapForceContainer.java` will be compiled and then executed.

Compiling and running the example in Eclipse

To import the sample Java project into Eclipse:

1. Check that all prerequisites are met (see Prerequisites[1420]).
2. On the **File** menu, click **Import**.
3. Select **Existing Projects into Workspace**, and browse for the Eclipse project file located at `<ApplicationFolder>\Examples\ActiveX\Java\`. Since you may not have write-access in this folder, it is recommended to select the **Copy projects into workspace** check box on the Import dialog box.

To run the example application, right-click the project in Package Explorer and select the command **Run as | Java Application**.

Help for Java API classes is available through comments in code as well as the Javadoc view of Eclipse. To enable the Javadoc view in Eclipse, select the menu command **Window | Show View | JavaDoc**.

17.5.2.2 Creating the ActiveX Controls

The code listing below show how ActiveX controls can be created. The constructors will create the Java wrapper objects. Adding these Canvas-derived objects to a panel or to a frame will trigger the creation of the wrapped ActiveX object.

```
01   /**
02    * MapForce manager control - always needed
03    */
04   public static MapForceControl         mapForceControl = null;
05
06   /**
07    * MapForceDocument editing control
08    */
09   public static MapForceControlDocument     mapForceDocument = null;
10
11   /**
12    * Tool windows - MapForce place-holder controls
13    */
14   private static MapForceControlPlaceHolder   mapForceProjectToolWindow = null;
15   private static MapForceControlPlaceHolder   mapForceValidationToolWindow = null;
16   private static MapForceControlPlaceHolder   mapForceLibraryToolWindow = null;
17
```

```
18   // Create the MapForce ActiveX control; the parameter determines that we want
     // to place document controls and place-holder controls individually.
19   // It gives us full control over the menu, as well.
20      mapForceControl = new MapForceControl(
        ICActiveXIntegrationLevel.ICActiveXIntegrationOnDocumentLevel.getValue(), false );
21
22      mapForceDocument = new MapForceControlDocument();
23      frame.add( mapForceDocument, BorderLayout.CENTER );
24
25
26   // Create a project window and open the sample project in it
27      mapForceProjectToolWindow = new MapForceControlPlaceHolder(
        MapForceControlPlaceholderWindow.MapForceXProjectWindow.getValue(),
        strExamplesFolder + "MapForceExamples.mfp" ) ;
28      mapForceProjectToolWindow.setPreferredSize( new Dimension( 200, 200 ) );
```

17.5.2.3 Loading Data in the Controls

The code listing below show how data can be loaded in the ActiveX controls.

```
1    // Locate samples installed with the product.
2       final String strExamplesFolder = System.getenv( "USERPROFILE" ) +
        "\\Documents\\Altova\\MapForce2022\\MapForceExamples\\";
3       mapForceProjectToolWindow = new
MapForceControlPlaceHolder( MapForceControlPlaceholderWindow.MapForceXProjectWindow.getValu
e(), strExamplesFolder + "MapForceExamples.mfp" ) ;
```

17.5.2.4 Basic Event Handling

The code listing below shows how basic events can be handled. When calling the MapForceControl's `open` method, or when trying to open a file via the menu or Project tree, the `onOpenedOrFocused` event is sent to the attached event handler. The basic handling for this event is opening the file by calling the MapForceDocumentControl's `open` method.

```
01       // Open the Marketing file when button is pressed
02       btnMarkExp.addActionListener( new ActionListener() {
03         public void actionPerformed(ActionEvent e) {
04           try {
05             // Instruct the Document control to open the file - avoid calling the open
method of MapForceControl (see help)
06             mapForceDocument.open( strExamplesFolder + "MarketingExpenses.mfd" );
07             mapForceDocument.requestFocusInWindow();
08           } catch (AutomationException e1) {
09             e1.printStackTrace();
10           }
11         }
12       } );
13       public void onOpenedOrFocused( String i_strFileName, boolean
i_bOpenWithThisControl, boolean i_bFileAlreadyOpened ) throws AutomationException
14   {
15     // Handle the New/Open events coming from the Project tree or from the menus
16     if ( !i_bFileAlreadyOpened )
```

```
17      {
18        // This is basically an SDI interface, so open the file in the already existing
document control
19        try {
20          MapForceContainer.mapForceDocument.open( i_strFileName );
21          MapForceContainer.mapForceDocument.requestFocusInWindow();
22        } catch (Exception e) {
23          e.printStackTrace();
24        }
25      }
26    }
```

17.5.2.5 Menus

The code listing below shows how menu items can be created. Each `MapForceCommand` object gets a corresponding `MenuItem` object, with the `ActionCommand` set to the ID of the command. The actions generated by all menu items are handled by the same function, which can perform specific handlings (like reinterpreting the closing mechanism) or can delegate the execution to the `MapForceControl` object by calling its `exec` method. The `menuMap` object that is filled during menu creation is used later (see section UI Update Event Handling 1445).

```
01
02        // Load the file menu when the button is pressed
03        btnMenu.addActionListener( new ActionListener() {
04          public void actionPerformed(ActionEvent e) {
05            try {
06              // Create the menubar that will be attached to the frame
07              MenuBar mb = new MenuBar();
08              // Load the main menu's first item - the File menu
09              MapForceCommand xmlSpyMenu =
mapForceControl.getMainMenu().getSubCommands().getItem( 0 );
10              // Create Java menu items from the Commands objects
11              Menu fileMenu = new Menu();
12              handlerObject.fillMenu( fileMenu, xmlSpyMenu.getSubCommands() );
13              fileMenu.setLabel( xmlSpyMenu.getLabel().replace( "&", "" ) );
14              mb.add( fileMenu );
15              frame.setMenuBar( mb );
16              frame.validate();
17            } catch (AutomationException e1) {
18              e1.printStackTrace();
19            }
20            // Disable the button when the action has been performed
21            ((AbstractButton) e.getSource()).setEnabled( false );
22          }
23        } ) ;
24    /**
25     * Populates a menu with the commands and submenus contained in an MapForceCommands
object
26     */
27    public void fillMenu(Menu newMenu, MapForceCommands mapForceMenu) throws
AutomationException
28    {
29      // For each command/submenu in the mapForceMenu
30      for ( int i = 0 ; i < mapForceMenu.getCount() ; ++i  )
31      {
```

```
32        MapForceCommand mapForceCommand = mapForceMenu.getItem( i );
33        if ( mapForceCommand.getIsSeparator() )
34          newMenu.addSeparator();
35        else
36        {
37          MapForceCommands subCommands = mapForceCommand.getSubCommands();
38          // Is it a command (leaf), or a submenu?
39          if ( subCommands.isNull() || subCommands.getCount() == 0 )
40          {
41            // Command -> add it to the menu, set its ActionCommand to its ID and store it
in the menuMap
42            MenuItem mi = new MenuItem( mapForceCommand.getLabel().replace( "&", "" ) );
43            mi.setActionCommand( "" + mapForceCommand.getID() );
44            mi.addActionListener( this );
45            newMenu.add( mi );
46            menuMap.put( mapForceCommand.getID(), mi );
47          }
48          else
49          {
50            // Submenu -> create submenu and repeat recursively
51            Menu newSubMenu = new Menu();
52            fillMenu( newSubMenu, subCommands );
53            newSubMenu.setLabel( mapForceCommand.getLabel().replace( "&", "" ) );
54            newMenu.add( newSubMenu );
55          }
56        }
57      }
58    }
59    /**
60     * Action handler for the menu items
61     * Called when the user selects a menu item; the item's action command corresponds to
the command table for MapForce
62     */
63    public void actionPerformed( ActionEvent e )
64    {
65      try
66      {
67        int iCmd = Integer.parseInt( e.getActionCommand() );
68        // Handle explicitly the Close commands
69        switch ( iCmd )
70        {
71          case 57602:       // Close
72          case 34050:       // Close All
73            MapForceContainer.initMapForceDocument();
74            break;
75          default:
76            MapForceContainer.mapForceControl.exec( iCmd );
77            break;
78        }
79      }
80      catch ( Exception ex )
81      {
82        ex.printStackTrace();
83      }
84
85    }
```


© 2016-2022 Altova GmbH

17.5.2.6 UI Update Event Handling

The code listing below shows how a UI-Update event handler can be created.

```
01 /**
02    * Call-back from the MapForceControl.
03    * Called to enable/disable commands
04    */
05   @Override
06   public void onUpdateCmdUI() throws AutomationException
07   {
08     // A command should be enabled if the result of queryStatus contains the Supported
(1) and Enabled (2) flags
09     for ( java.util.Map.Entry<Integer, MenuItem> pair : menuMap.entrySet() )
10
      pair.getValue().setEnabled( MapForceContainer.mapForceControl.queryStatus( pair.getKe
y() ) > 2 );
11   }
12  /**
13    * Call-back from the MapForceControl.
14    * Usually called while enabling/disabling commands due to UI updates
15    */
16   @Override
17   public boolean onIsActiveEditor( String i_strFilePath ) throws AutomationException
18   {
19     try {
20       return
MapForceContainer.mapForceDocument.getDocument().getFullName().equalsIgnoreCase( i_strFileP
ath );
21     } catch ( Exception e ) {
22       return false;
23     }
24   }
```

17.5.2.7 Listing the Properties of a MapForce Mapping

The listing below shows how a Mapping object in MapForce can be loaded as a table and prepared for modal activation.

```
01 //access MapForce Java-COM bridge
02 import com.altova.automation.MapForce.*;
03 import com.altova.automation.MapForce.Component;
04 import com.altova.automation.MapForce.Enums.ENUMComponentUsageKind;
05
06 //access AWT and Swing components
07 import java.awt.*;
08 import javax.swing.*;
09 import javax.swing.table.*;
10
11
12 /**
13  * A simple example of a table control loading the structure from a Mapping object.
```

```
14  * The class receives an Mapping object, loads its components in a JTable, and prepares
15  * for modal activation.
16  *
17  * Feel free to modify and extend this sample.
18  *
19  * @author Altova GmbH
20  */
21 class MapForceTable extends JDialog
22 {
23   /**
24    * The table control
25    */
26   private JTable myTable;
27
28   /**
29    * Constructor that prepares the modal dialog containing the filled table control
30    * @param mapping The data to be displayed in the table
31    * @param parent  Parent frame
32    */
33   public MapForceTable( Mapping mapping, Frame parent )
34   {
35     // Construct the modal dialog
36     super( parent, "MapForce component table", true );
37     // Build up the tree
38     fillTable( mapping );
39     // Arrange controls in the dialog
40     setContentPane( new JScrollPane( myTable ) );
41   }
42
43   /**
44    * Loads the components of a Mapping object in the table
45    * @param mapping Source data
46    */
47   private void fillTable( Mapping mapping)
48   {
49     try
50     {
51       // count how many Instance components do we have
52       int size = 0;
53       for (Component comp : mapping.getComponents())
54         if ( comp.getUsageKind() ==
ENUMComponentUsageKind.eComponentUsageKind_Instance )
55           ++size;
56
57       // Prepare data
58       final String[] columnNames = { "Component", "Has inputs", "Has outputs", "Input
file", "Output file", "Schema" };
59       final Object[][] data = new Object[size ][ 7 ] ;
60       int index = 0 ;
61       for (Component comp : mapping.getComponents())
62         if ( comp.getUsageKind() ==
ENUMComponentUsageKind.eComponentUsageKind_Instance )
63         {
64           int i = 0;
65           data[ index ][ i++ ] = comp.getName() ;
66           data[ index ][ i++ ] = new Boolean( comp.getHasIncomingConnections() );
67           data[ index ][ i++ ] = new Boolean( comp.getHasOutgoingConnections() );
68           data[ index ][ i++ ] = comp.getInputInstanceFile();
```

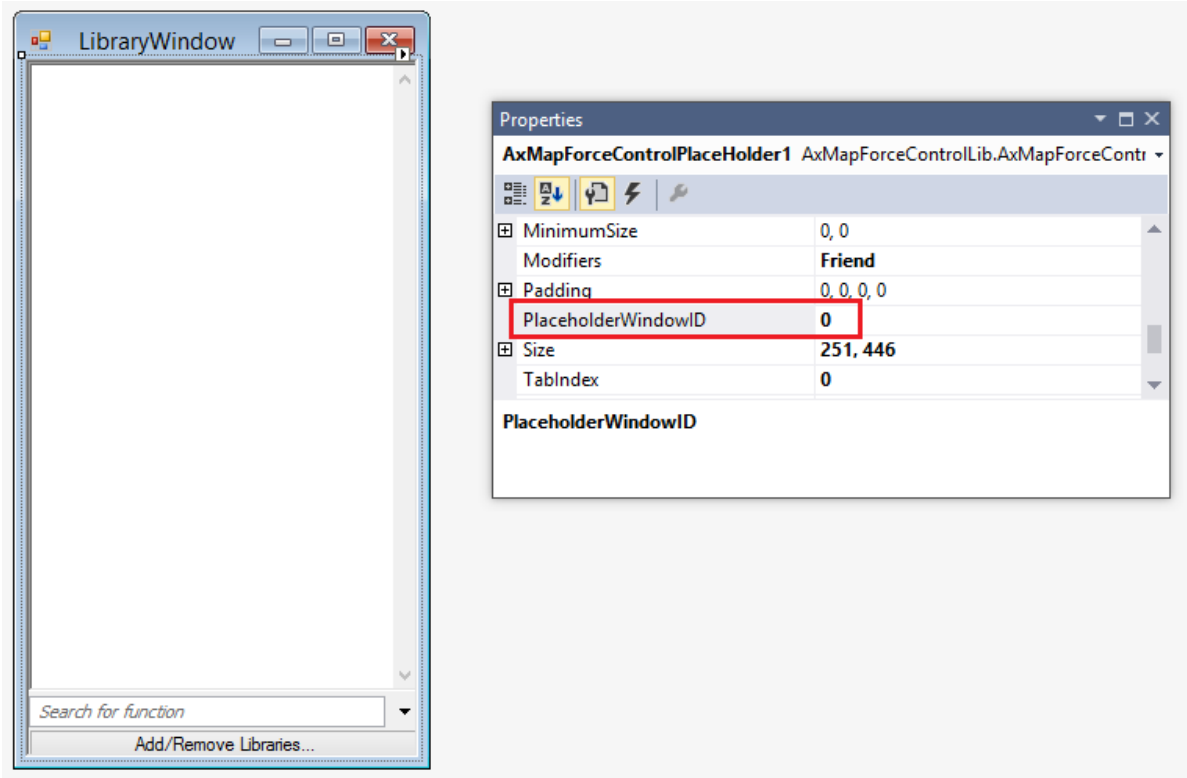

© 2016-2022 Altova GmbH

```
69          data[ index ][ i++ ] = comp.getOutputInstanceFile();
70          data[ index++ ][ i ] = comp.getSchema() ;
71        }
72
73      // Set up table
74      myTable = new JTable( new AbstractTableModel() {
75          public String getColumnName(int col) { return columnNames[col]; }
76          public int getRowCount() { return data.length; }
77          public int getColumnCount() { return columnNames.length; }
78          public Object getValueAt(int row, int col) { return data[row][col]; }
79          public boolean isCellEditable(int row, int col) { return false; }
80          public Class getColumnClass(int c) { return getValueAt(0, c).getClass(); }

81      } );
82
83      // Set width
84      for( index = 0 ; index < columnNames.length ; ++index )
85        myTable.getColumnModel().getColumn( index ).setMinWidth( 80 );
86      myTable.getColumnModel().getColumn( 5 ).setMinWidth( 400 );
87    }
88    catch (Exception e)
89    {
90      e.printStackTrace();
91    }
92  }
93
94 }
```

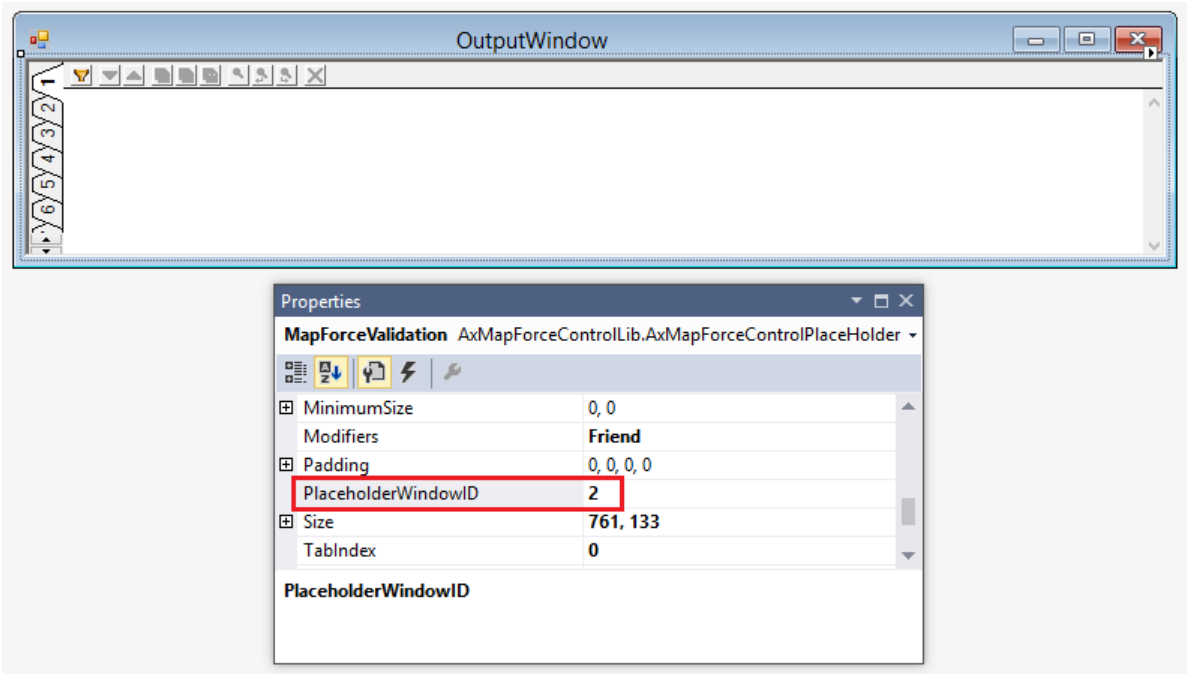
17.5.3 VB.NET

Source code which illustrates integration of MapForceControl into a VB.NET application can be found in the folder `<ApplicationFolder>\Examples\ActiveX\VB.NET` of your MapForce installation. The solution consists of three windows, as follows:

1. **MainWindow.vb** - the main document window, which also includes a basic application menu.
2. **LibraryWindow.vb** - the Library window. The contents of this window is populated by a Placeholder control which has the **PlaceholderWindowID** property set to 0 (this value instructs the control to display specifically the Library window).



3. **OutputWindow.vb** - the Messages (Output) window. The contents of this window is populated by a Placeholder control which has the **PlaceholderWindowID** property set to 2 (this value instructs the control to display specifically the Output window).



 © 2016-2022 Altova GmbH

Before attempting to build and run this solution, note the following steps:

Step 1: Check the prerequisites

For the list of prerequisites, see Prerequisites 1420.

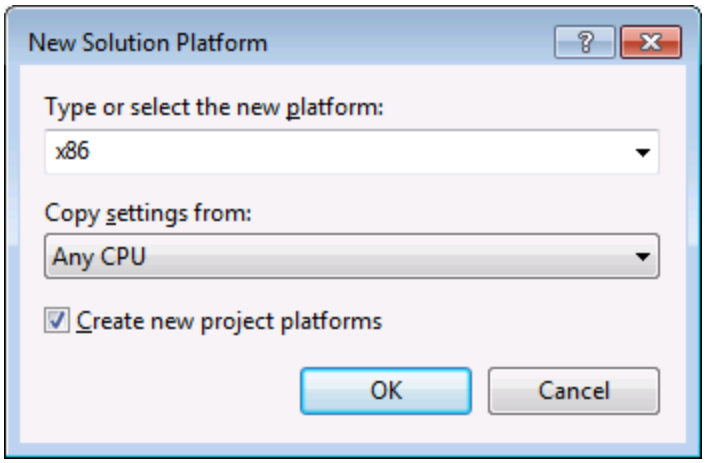
Step 2: Copy the sample to a directory where you have write permissions

To avoid running Visual Studio as an Administrator, copy the source code to a directory where you have write permissions, instead of running it from the default location.

Step 3: Set the build platform

- Create a build platform configuration that matches the platform under which you want to build (x86, x64). Here is how you can create the build configuration:

  a. Right-click the solution in Visual Studio, and select **Configuration Manager**.
  b. Under **Active solution platform**, select **New...** and then select the x86 or x64 configuration (in this example, **x86**).



You are now ready to build and run the solution in Visual Studio. Remember to build using the configuration that matches your target platform (x86, x64); otherwise, runtime errors might occur.

17.6 Command Reference

This section lists the names and identifiers of all menu commands that are available within MapForce. Every sub-section lists the commands from the corresponding top-level menu of MapForce. The command tables are organized as follows:

- The "Menu Item" column shows the command's menu text as it appears in MapForce, to make it easier for you to identify the functionality behind the command.
- The "Command Name" column specifies the string that can be used to get an icon with the same name from **ActiveX\Images** folder of the MapForce installation directory.
- The "ID" column shows the numeric identifier of the column that must be supplied as argument to methods which execute or query this command.

To execute a command, use the `MapForceControl.Exec`[1466] or the `MapForceControlDocument.Exec`[1472] methods. To query the status of a command, use the `MapForceControl.QueryStatus`[1467] or `MapForceControlDocument.QueryStatus`[1473] methods.

Depending on the edition of MapForce you have installed, some of these commands might not be supported.

17.6.1 "File" Menu

The "File" menu has the following commands:

Menu item	Command name	ID
New...	ID_FILE_NEW	57600
Open...	ID_FILE_OPEN	57601
Save	ID_FILE_SAVE	57603
Save As...	ID_FILE_SAVE_AS	57604
Save All	ID_FILE_SAVEALL	32377
Reload	IDC_FILE_RELOAD	32467
Close	ID_WINDOW_CLOSE	32453
Close All	ID_WINDOW_CLOSEALL	32454
Print...	ID_FILE_PRINT	57607
Print Preview	ID_FILE_PRINT_PREVIEW	57609
Print Setup...	ID_FILE_PRINT_SETUP	57606
Validate Mapping	ID_MAPPING_VALIDATE	32347
Mapping Settings	ID_MAPPING_SETTINGS	32396
Generate Code in Selected Language	ID_FILE_GENERATE_SELECTED_CODE	32362

 © 2016-2022 Altova GmbH

Menu item	Command name	ID
XSLT 1.0	ID_FILE_GENERATEXSLT	32360
XSLT 2.0	ID_FILE_GENERATEXSLT2	32361
XQuery	ID_FILE_GENERATEXQUERY	32359
Java	ID_FILE_GENERATEJAVACODE	32358
C# (Sharp)	ID_FILE_GENERATECSCODE	32357
C++	ID_FILE_GENERATECPPCODE	32356
Compile to MapForce Server Execution File...	ID_FILE_CREATE_SERVER_EXECUTION_FILE	32517
Deploy to FlowForce Server...	ID_FILE_DEPLOY_MAPPING	32506
Generate Documentation...	ID_FILE_GENERATE_DOCUMENTATION	32468
Recent File	ID_FILE_MRU_FILE1	57616
Exit	ID_APP_EXIT	57665

17.6.2 "Edit" Menu

The "Edit" menu has the following commands:

Menu item	Command name	ID
Undo	ID_EDIT_UNDO	57643
Redo	ID_EDIT_REDO	57644
Find...	ID_EDIT_FIND	57636
Find Next	ID_EDIT_FINDNEXT	32349
Find Previous	ID_EDIT_FINDPREV	32350
Cut	ID_EDIT_CUT	57635
Copy	ID_EDIT_COPY	57634
Paste	ID_EDIT_PASTE	57637
Delete	ID_EDIT_CLEAR	57632
Select All	ID_EDIT_SELECT_ALL	57642

17.6.3 "Insert" Menu

The "Insert" menu has the following commands:

Menu item	Command name	ID
XML Schema/File...	ID_INSERT_XSD	32393
Database...	ID_INSERT_DATABASE	32389
EDI...	ID_INSERT_EDI	32390
Text File...	ID_INSERT_TXT	32392
Web Service Function...	ID_INSERT_WEBSERVICE_FUNCTION	32319
Excel 2007+ File...	ID_INSERT_EXCEL	32376
XBRL Document...	ID_INSERT_XBRL	32469
JSON Schema/File...	ID_INSERT_JSON	32531
Insert Input...	ID_FUNCTION_INSERT_INPUT	32383
Insert Output...	ID_FUNCTION_INSERT_OUTPUT	32402
Constant...	ID_INSERT_CONSTANT	32388
Variable...	ID_INSERT_VARIABLE	32500
Join	ID_INSERT_JOIN	32581
Sort: Nodes/Rows	ID_INSERT_SORT	32444
Filter: Nodes/Rows	ID_INSERT_FILTER	32391
SQL-WHERE/ORDER	ID_INSERT_SQLWHERE_CONDITION	32351
Value-Map	ID_INSERT_VALUEMAP	32354
IF-Else Condition	ID_INSRT_CONDITION	32394
Exception	ID_INSERT_EXCEPTION	32311

17.6.4 "Project" Menu

The "Project" menu has the following commands:

Menu item	Command name	ID
Reload Project	ID_PROJECT_RELOAD	32476

© 2016-2022 Altova GmbH

Menu item	Command name	ID
Close Project	ID_FILE_CLOSEPROJECT	32355
Save Project	ID_FILE_SAVEPROJECT	32378
Add Files to Project...	ID_PROJECT_ADDFILESTOPROJECT	32420
Add Active File to Project	ID_PROJECT_ADDACTIVEFILETOPROJECT	32419
Create Folder...	ID_PROJECT_CREATE_FOLDER	32310
Open Mapping	ID_PROJECT_OPEN_MAPPING	32307
Create Mapping for Operation...	ID_PROJECT_CREATE_MAPPING_FOR_OPERATION	32399
Add Mapping File for Operation...	ID_PROJECT_ADD_MAPPING	32309
Insert Web Service...	ID_PROJECT_INSERT_WEBSERVICE	32306
Open File in XMLSpy	ID_PROJECT_OPEN_IN_XMLSPY	32305
Generate Code for Entire Project	ID_PROJECT_GENERATE_ALL	32303
XSLT 1.0	ID_PROJECT_GENERATEXSLTCODE_ENTIRE	32408
XSLT 2.0	ID_PROJECT_GENERATEXSLT2CODE_ENTIRE	32409
XQuery	ID_PROJECT_GENERATEXQUERYCODE_ENTIRE	32410
Java	ID_PROJECT_GENERATEJAVACODE_ENTIRE	32411
C# (Sharp)	ID_PROJECT_GENERATECSCODE_ENTIRE	32412
C++	ID_PROJECT_GENERATECPPCODE_ENTIRE	32413
Properties	ID_PROJECT_PROPERTIES	32404
Recent Project	ID_FILE_MRU_PROJECT1	32364

17.6.5 "Component" Menu

The "Component" menu has the following commands:

Menu item	Command name	ID
Change Root Element...	ID_COMPONENT_CHANGEROOTELEMENT	32334
Edit Schema Definition in XMLSpy	ID_COMPONENT_EDIT_SCHEMA	32337

Menu item	Command name	ID
Edit FlexText Configuration	ID_COMPONENT_EDIT_MFT	32301
Add/Remove/Edit Database Objects...	ID_COMPONENT_SELECTTABLES	32346
Create Mapping to EDI X12 997	ID_COMPONENT_CREATE_MAPPING_TO_997	32483
Create Mapping to EDI X12 999	ID_COMPONENT_CREATE_MAPPING_TO_999	32484
Refresh	IDC_COMMAND_REFRESH_COMPONENT	32373
Add Duplicate Input Before	ID_COMPONENT_CREATE_DUPLICATE_ICON_BEFORE	32503
Add Duplicate Input After	ID_COMPONENT_CREATE_DUPLICATE_ICON	32335
Remove Duplicate	ID_COMPONENT_REMOVE_DUPLICATE_ICON	32339
Add Comment Before	ID_COMPONENT_ADD_COMMENT_BEFORE	32518
Add Comment After	ID_COMPONENT_ADD_COMMENT_AFTER	32519
Add Processing Instruction Before...	ID_COMPONENT_ADD_PI_BEFORE	32520
Add Processing Instruction After...	ID_COMPONENT_ADD_PI_AFTER	32521
Edit Processing Instruction Name...	ID_COMPONENT_EDIT_PI	32524
Delete Comment/Processing Instruction	ID_COMPONENT_REMOVE_COMMENT_PI	32522
Write Content as CDATA Section	ID_COMPONENT_TOGGLE_CDATA	32525
Database Table Actions	ID_POPUP_DATABASETABLEACTIONS	32400
Query Database...	ID_QUERY_DATABASE	32341
Align Tree Left	ID_COMPONENT_LEFTALIGNTREE	32338
Align Tree Right	ID_COMPONENT_RIGHTALIGNTREE	32340
Properties	ID_COMPONENT_PROPERTIES	32336

17.6.6 "Connection" Menu

The "Connection" menu has the following commands:

Menu item	Command name	ID
Auto Connect Matching Children	ID_CONNECTION_AUTOCONNECTCHILDREN	32342

© 2016-2022 Altova GmbH

Menu item	Command name	ID
Settings for Connect Matching Children	ID_CONNECTION_SETTINGS	32344
Connect Matching Children...	ID_CONNECTION_MAPCHILDELEMENTS	32343
Target Driven (Standard)	ID_POPUP_NORMALCONNECTION	32401
Copy-All (Copy Child Items)	ID_POPUP_NORMALWITHCHILDREN_CONNECTION	32460
Source Driven (Mixed Content)	ID_POPUP_ORDERBYSOURCECONNECTION	32403
Properties	ID_POPUP_CONNECTION_SETTINGS	32398

17.6.7 "Function" Menu

The "Function" menu has the following commands:

Menu item	Command name	ID
Create User-Defined Function...	ID_FUNCTION_CREATE_EMPTY	32380
Create User-Defined Function from Selection...	ID_FUNCTION_CREATE_FROM_SELECTION	32381
Function Settings	ID_FUNCTION_SETTINGS	32387
Remove Function	ID_FUNCTION_REMOVE	32385
Insert Input...	ID_FUNCTION_INSERT_INPUT	32383
Insert Output...	ID_FUNCTION_INSERT_OUTPUT	32402

17.6.8 "Output" Menu

The "Output" menu has the following commands:

Menu item	Command name	ID
XSLT 1.0	ID_SELECT_LANGUAGE_XSLT	32433
XSLT 2.0	ID_SELECT_LANGUAGE_XSLT2	32434
XQuery	ID_SELECT_LANGUAGE_XQUERY	32432
Java	ID_SELECT_LANGUAGE_JAVA	32431
C# (Sharp)	ID_SELECT_LANGUAGE_CSHARP	32430
C++	ID_SELECT_LANGUAGE_CPP	32429

Menu item	Command name	ID
Built-In Execution Engine	ID_SELECT_LANGUAGE_BUILTIN	32490
Validate Output File	ID_XML_VALIDATE	32458
Save Output File...	IDC_FILE_SAVEGENERATEDOUTPUT	32321
Save All Output Files...	IDC_FILE_SAVEALLGENERATEDOUTPUT	32374
Regenerate Output	ID_REGENERATE_PREVIEW_OUTPUT	32480
Run SQL-Script	ID_TRANSFORM_RUN_SQL	32442
Insert/Remove Bookmark	ID_TOGGLE_BOOKMARK	32317
Next Bookmark	ID_GOTONEXTBOOKMARK	32315
Previous Bookmark	ID_GOTOPREVBOOKMARK	32314
Remove All Bookmarks	ID_REMOVEALLBOOKMARKS	32313
Pretty-Print XML Text	ID_PRETTY_PRINT_OUTPUT	32363
Text View Settings	ID_TEXTVIEWSETTINGSDIALOG	32472

17.6.9 "Debug" Menu

The "Debug" menu has the following commands:

Menu item	Command name	ID
Start Debugging	ID_DEBUG_START	32540
Stop Debugging	ID_DEBUG_STOP	32541
Step Into	ID_DEBUG_STEP_INTO	32545
Step Over	ID_DEBUG_STEP_OVER	32551
Step Out	ID_DEBUG_STEP_OUT	32552
Minimal Step	ID_DEBUG_STEP_NEXT_TRACE	32554

17.6.10 "View" Menu

The "View" menu has the following commands:

Menu item	Command name	ID
Show Annotations	ID_SHOW_ANNOTATION	32435

© 2016-2022 Altova GmbH

Menu item	Command name	ID
Show Types	ID_SHOW_TYPES	32437
Show Library in Function Header	ID_VIEW_SHOWLIBRARYINFUNCTIONHEADER	32448
Show Tips	ID_SHOW_TIPS	32436
XBRL Display Options	ID_VIEW_XBRL_DISPLAY_OPTIONS	32473
Show Selected Component Connectors	ID_VIEW_AUTOHIGHLIGHTCOMPONENTCONNECTIONS	32443
Show Connectors from Source to Target	ID_VIEW_RECURSIVEAUTOHIGHLIGHT	32447
Zoom...	ID_VIEW_ZOOM	32451
Back	ID_CMD_BACK	32479
Forward	ID_CMD_FORWARD	32478
Status Bar	ID_VIEW_STATUS_BAR	59393
Library Window	ID_VIEW_LIBRARY_WINDOW	32445
Messages	ID_VIEW_VALIDATION_OUTPUT	32450
Overview	ID_VIEW_OVERVIEW_WINDOW	32446
Project Window	ID_VIEW_PROJECT_WINDOW	32302
Values	ID_DEBUG_VIEW_VALUES_WINDOW	32544
Context	ID_DEBUG_VIEW_CONTEXT_WINDOW	32546
Breakpoints	ID_DEBUG_VIEW_DEBUGPOINTS_WINDOW	32547

17.6.11 "Tools" Menu

The "Tools" menu has the following commands:

Menu item	Command name	ID
Global Resources	IDC_GLOBALRESOURCES	37401
<plugin not loaded>	IDC_GLOBALRESOURCES_SUBMENUENTRY1	37408
Create Reversed Mapping	ID_CREATE_REVERSED_MAPPING	32489
Customize...	IDC_APP_TOOLS_CUSTOMIZE	32959
Options...	ID_TOOLS_OPTIONS	32441

17.6.12 "Window" Menu

The "Window" menu has the following commands:

Menu item	Command name	ID
Cascade	ID_WINDOW_CASCADE	57650
Tile Horizontal	ID_WINDOW_TILE_HORZ	57651
Tile Vertical	ID_WINDOW_TILE_VERT	57652

17.6.13 "Help" Menu

The "Help" menu has the following commands:

Menu item	Command name	ID
Table of Contents...	IDC_HELP_CONTENTS	32966
Index...	IDC_HELP_INDEX	32967
Search...	IDC_HELP_SEARCH	32969
Software Activation...	IDC_ACTIVATION	32970
Order Form...	IDC_OPEN_ORDER_PAGE	32971
Registration...	IDC_REGISTRATION	32972
Check for Updates...	IDC_CHECK_FOR_UPDATES	32973
MapForce Product Comparison...	IDC_PRODUCT_COMPARISON	32955
Support Center...	IDC_OPEN_SUPPORT_PAGE	32961
FAQ on the Web...	IDC_OPEN_FAQ_PAGE	32962
Download Components and Free Tools...	IDC_OPEN_COMPONENTS_PAGE	32963
MapForce on the Internet..	IDC_OPEN_HOME_PAGE	32964
MapForce Training...	IDC_OPEN_TRAINING_PAGE	32965
About MapForce...	ID_APP_ABOUT	57664

© 2016-2022 Altova GmbH

17.7 Object Reference

Objects:
MapForceCommand 1459
MapForceCommands 1461
MapForceControl 1462
MapForceControlDocument 1470
MapForceControlPlaceHolder 1476

To give access to standard MapForce functionality, objects of the **MapForce automation interface** can be accessed as well. See MapForceControl.Application 1463, MapForceControlDocument.Document 1471 and MapForceControlPlaceHolder.Project 1477 for more information.

17.7.1 MapForceCommand

Properties:
`ID` 1460
`Label` 1460
`Name` 1460
`IsSeparator` 1460
`ToolTip` 1461
`StatusText` 1461
`Accelerator` 1459
`SubCommands` 1461

Description:
A command object can be one of the following: an executable command, a command container (for example, a menu, submenu, or toolbar), or a menu separator. To determine what kind of information is stored in the current `Command` object, query its `ID`, `IsSeparator`, and `SubCommands` properties, as follows.

The Command object is...	When...
An executable command	• `ID` is greater than zero • `IsSeparator` is false • `SubCommands` is empty
A command container	• `ID` is zero • `IsSeparator` is false • `SubCommands` contains a collection of `Command` objects.
Separator	• `ID` is zero • `IsSeparator` is true

17.7.1.1 Accelerator

Property: `Accelerator` as string

Description:
Returns the accelerator key defined for the command. If the command has no accelerator key assigned, this property returns the empty string. The string representation of the accelerator key has the following format:

```
[ALT+][CTRL+][SHIFT+]key
```

Where `key` is converted using the Windows Platform SDK function `GetKeyNameText`.

17.7.1.2 ID

Property: `ID` as long

Description:
This property gets the unique identifier of the command. A command's ID is required to execute the command (using `Exec`[1466]) or query its status (using `QueryStatus`[1467]). If the command is a container for other commands (for example, a top-level menu), or a separator, the ID is 0.

17.7.1.3 IsSeparator

Property: `IsSeparator` as boolean

Description:
The property returns `true` if the command object is a menu separator; `false` otherwise. See also Command[1459].

17.7.1.4 Label

Property: `Label` as string

Description:
This property gets the text of the command as it is displayed in the graphical user interface of MapForce. If the command is a separator, "Label" is an empty string. This property may also return an empty string for some toolbar commands that do not have any GUI text associated with them.

17.7.1.5 Name

Property: `Name` as string

Description:
This property gets the unique name of the command. This value can be used to get the icon file of the command, where it is available. The available icon files can be found in the folder **<ApplicationFolder>\Examples\ActiveX\Images** of your MapForce installation.

© 2016-2022 Altova GmbH

17.7.1.6 StatusText

Property: `Label` as string

Description:
The status text is the text shown in the status bar of MapForce when the command is selected. It applies only to command objects that are not separators or containers of other commands; otherwise, the property is an empty string.

17.7.1.7 SubCommands

Property: `SubCommands` as `Commands` 1461

Description:
The `SubCommands` property gets the collection of `Command` 1459 objects that are sub-commands of the current command. The property is applicable only to commands that are containers for other commands (menus, submenus, or toolbars). Such container commands have the `ID` set to 0, and the `IsSeparator` property set to `false`.

17.7.1.8 ToolTip

Property: `ToolTip` as string

Description:
This property gets the text that is shown as a tool-tip for each command. If the command does not have a tooltip text, the property returns an empty string.

17.7.2 MapForceCommands

Properties:
`Count` 1461
`Item` 1462

Description:
Collection of `Command` 1459 objects to get access to command labels and IDs of the MapForceControl. Those commands can be executed with the `Exec` 1466 method and their status can be queried with `QueryStatus` 1467.

17.7.2.1 Count

Property: `Count` as long

Description:
Number of `Command` 1459 objects on this level of the collection.

17.7.2.2 Item

Property: `Item` (n as long) as `Command` 1459

Description:
Gets the command with the index n in this collection. Index is 1-based.

17.7.3 MapForceControl

Properties:
`IntegrationLevel` 1464
`Appearance` 1463
`Application` 1463
`BorderStyle` 1463
`CommandsList` 1463
`EnableUserPrompts` 1464
`MainMenu` 1464
`Toolbars` 1465

Methods:
`Open` 1466
`Exec` 1466
`QueryStatus` 1467

Events:
`OnUpdateCmdUI` 1469
`OnOpenedOrFocused` 1469
`OnCloseEditingWindow` 1467
`OnFileChangedAlert` 1468
`OnContextChanged` 1468
`OnDocumentOpened` 1468
`OnValidationWindowUpdated` 1469

This object is a complete ActiveX control and should only be visible if the MapForce library is used in the Application Level mode.

```
CLSID: A38637E9-5759-4456-A167-F01160CC22C1
ProgID: Altova.MapForceControl
```

17.7.3.1 Properties

The following properties are defined:

`IntegrationLevel` 1464
`EnableUserPrompts` 1464
`Appearance` 1463
`BorderStyle` 1463

Command related properties:

 © 2016-2022 Altova GmbH

CommandsList 1463
MainMenu 1464
Toolbars 1465

Access to MapForceAPI:
Application 1463

17.7.3.1.1 Appearance

Property: `Appearance` as short

Dispatch Id: *-520*

Description:
A value not equal to `0` displays a client edge around the control. Default value is `0`.

17.7.3.1.2 Application

Property: `Application` as Application

Dispatch Id: *1*

Description:
The `Application` property gives access to the `Application` object of the complete MapForce automation server API. The property is read-only.

17.7.3.1.3 BorderStyle

Property: `BorderStyle` as short

Dispatch Id: *-504*

Description:
A value of `1` displays the control with a thin border. Default value is `0`.

17.7.3.1.4 CommandsList

Property: `CommandList` as Commands 1461 (read-only)

Dispatch Id: *1004*

Description:
This property returns a flat list of all commands defined available with MapForceControl. To get commands organized according to their menu structure, use MainMenu 1464. To get toolbar commands, use Toolbars 1465.

```
public void GetAllMapForceCommands()
{
    // Get all commands from the MapForce ActiveX control assigned to the current form
    MapForceControlLib.MapForceCommands commands = this.axMapForceControl1.CommandList;
    // Iterate through all commands
    for (int i = 0; i < commands.Count; i++)
    {
      // Get each command by index and output it to the console
      MapForceControlLib.MapForceCommand cmd = axMapForceControl1.CommandList[i];
      Console.WriteLine("{0} {1} {2}", cmd.ID, cmd.Name, cmd.Label.Replace("&", ""));
    }
}
```

C# example

17.7.3.1.5 EnableUserPrompts

Property: `EnableUserPrompts` as boolean

Dispatch Id: *1006*

Description:
Setting this property to *false*, disables user prompts in the control. The default value is *true*.

17.7.3.1.6 IntegrationLevel

Property: `IntegrationLevel` as `ICActiveXIntegrationLevel` 1478

Dispatch Id: *1000*

Description:
The `IntegrationLevel` property determines the operation mode of the control. See also Integration at Application Level 1424 and Integration at Document Level 1427 for more information.

Note: It is important to set this property immediately after the creation of the `MapForceControl` object.

17.7.3.1.7 MainMenu

Property: `MainMenu` as Command 1459 (read-only)

Dispatch Id: *1003*

Description:
This property provides information about the structure and commands available in the MapForceControl main menu, as a `Command` object. The `Command` object contains all available submenus of MapForce (for example "File", "Edit", "View" etc.). To access the submenu objects, use the `SubCommands` property of the `MainMenu` property. Each submenu is also a `Command` object. For each submenu, you can then further iterate through their

 © 2016-2022 Altova GmbH

SubCommands property in order to get their corresponding child commands and separators (this technique may be used, for example, to create the application menu programmatically). Note that some menu commands act as containers ("parents") for other menu commands, in which case they also have a SubCommands property. To get the structure of all menu commands programmatically, you will need a recursive function, as illustrated for C# in Retrieving Command Information[1434].

```
public void GetMapForceMenus()
{
    // Get the main menu from the MapForce ActiveX control assigned to the current form
    MapForceControlLib.MapForceCommand mainMenu = this.axMapForceControl1.MainMenu;

    // Loop through entries of the main menu (e.g. File, Edit, etc.)
    for (int i = 0; i < mainMenu.SubCommands.Count; i++)
    {
      MapForceControlLib.MapForceCommand menu = mainMenu.SubCommands[i];
      Console.WriteLine("{0} menu has {1} children items (including separators)",
menu.Label.Replace("&", ""), menu.SubCommands.Count);
    }
}
```

C# example

17.7.3.1.8 Toolbars

Property: Toolbars as Commands[1461] (read-only)

Dispatch Id: *1005*

Description:
This property provides information about the structure of MapForceControl toolbars, as a Command object. The Command object contains all available toolbars of MapForce. To access the toolbars, use the SubCommands property of the Toolbars property. Each toolbar is also a Command object. For each toolbar, you can then further iterate through their SubCommands property in order to get their commands (this technique may be used, for example, to create the application's toolbars programmatically).

```
public void GetMapForceToolbars()
{
    // Get the application toolbars from the MapForce ActiveX control assigned to the
current form
    MapForceControlLib.MapForceCommands toolbars = this.axMapForceControl1.Toolbars;

    // Iterate through all toolbars
    for (int i = 0; i < toolbars.Count; i++)
    {
      MapForceControlLib.MapForceCommand toolbar = toolbars[i];
      Console.WriteLine();
      Console.WriteLine("The toolbar \"{0}\" has the following commands:",
toolbar.Label);

      // Iterate through all commands of this toolbar
```

```
      for (int j = 0; j < toolbar.SubCommands.Count; j++)
      {
         MapForceControlLib.MapForceCommand cmd = toolbar.SubCommands[j];
         // Output only command objects that are not separators
         if (!cmd.IsSeparator)
         {
            Console.WriteLine("{0}, {1}, {2}", cmd.ID, cmd.Name, cmd.Label.Replace("&",
""));
         }
      }
    }
}
```

C# example

17.7.3.2 Methods

The following methods are defined:

Open 1466
Exec 1466
QueryStatus 1467

17.7.3.2.1 Exec

Method: `Exec` (`nCmdID` as long) as boolean

Dispatch Id: *6*

Description:
This method calls the MapForce command with the ID `nCmdID`. If the command can be executed, the method returns `true`. To get a list of all available commands, use `CommandsList` 1463. To retrieve the status of any command, use `QueryStatus` 1467.

17.7.3.2.2 Open

Method: `Open` (`strFilePath` as string) as boolean

Dispatch Id: *5*

Description:
The result of the method depends on the extension passed in the argument `strFilePath`. If the file extension is `.sps`, a new document is opened. If the file extension is `.svp`, the corresponding project is opened. If a different file extension is passed into the method, the control tries to load the file as a new component into the active document.

Do not use this method to load documents or projects when using the control in document-level integration mode. Instead, use `MapForceControlDocument.Open` 1473 and `MapForceControlPlaceHolder.OpenProject` 1477.

© 2016-2022 Altova GmbH

17.7.3.2.3 QueryStatus

Method: `QueryStatus` (`nCmdID` as long) as long

Dispatch Id: *7*

Description:
`QueryStatus` returns the enabled/disabled and checked/unchecked status of the command specified by `nCmdID`. The status is returned as a bit mask.

Bit	Value	Name	Meaning
0	1	Supported	Set if the command is supported.
1	2	Enabled	Set if the command is enabled (can be executed).
2	4	Checked	Set if the command is checked.

This means that if `QueryStatus` returns `0` the command ID is not recognized as a valid MapForce command. If `QueryStatus` returns a value of `1` or `5`, the command is disabled.

17.7.3.3 Events

The MapForceControl ActiveX control provides the following connection point events:

`OnUpdateCmdUI` 1469
`OnOpenedOrFocused` 1469
`OnCloseEditingWindow` 1467
`OnFileChangedAlert` 1468
`OnContextChanged` 1468

`OnDocumentOpened` 1468
`OnValidationWindowUpdated` 1469

17.7.3.3.1 OnCloseEditingWindow

Event: `OnCloseEditingWindow` (i_strFilePath as String) as boolean

Dispatch Id: *1002*

Description:
This event is triggered when MapForce needs to close an already open document. As an answer to this event, clients should close the editor window associated with *i_strFilePath*. Returning *true* from this event indicates that the client has closed the document. Clients can return *false* if no specific handling is required and MapForceControl should try to close the editor and destroy the associated document control.

17.7.3.3.2 OnContextChanged

Event: `OnContextChanged` (`i_strContextName` as String, `i_bActive` as bool) as bool

Dispatch Id: *1004*

Description:
This event is not used in MapForce

17.7.3.3.3 OnDocumentOpened

Event: `OnDocumentOpened` (`objDocument` as Document)

Dispatch Id: *1*

Description:
This event is triggered whenever a document is opened. The argument `objDocument` is a `Document` object from the MapForce automation interface and can be used to query for more details about the document, or perform additional operations. When integrating on document-level, it is often better to use the event `MapForceControlDocument.OnDocumentOpened` 1475 instead.

17.7.3.3.4 OnFileChangedAlert

Event: `OnFileChangedAlert` (`i_strFilePath` as String) as bool

Dispatch Id: *1001*

Description:
This event is triggered when a file loaded with MapForceControl is changed on the hard disk by another application. Clients should return true, if they handled the event, or false, if MapForce should handle it in its customary way, i.e. prompting the user for reload.

17.7.3.3.5 OnLicenseProblem

Event: `OnLicenseProblem` (`i_strLicenseProblemText` as String)

Dispatch Id: *1005*

Description:
This event is triggered when MapForceControl detects that no valid license is available for this control. In case of restricted user licenses this can happen some time after the control has been initialized. Integrators should use this event to disable access to this control's functionality. After returning from this event, the control will block access to its functionality (e.g. show empty windows in its controls and return errors on requests).

© 2016-2022 Altova GmbH

17.7.3.3.6 OnOpenedOrFocused

Event: `OnOpenedOrFocused` (`i_strFilePath` as String, `i_bOpenWithThisControl` as bool)

Dispatch Id: *1000*

Description:
When integrating at application level, this event informs clients that a document has been opened, or made active by MapForce.

When integrating at document level, this event instructs the client to open the file `i_strFilePath` in a document window. If the file is already open, the corresponding document window should be made the active window.

if `i_bOpenWithThisControl` is `true`, the document must be opened with MapForceControl, since internal access is required. Otherwise, the file can be opened with different editors.

17.7.3.3.7 OnToolWindowUpdated

Event: `OnToolWindowUpdated(pToolWnd` as long `)`

Dispatch Id: *1006*

Description:
This event is triggered when the tool window is updated.

17.7.3.3.8 OnUpdateCmdUI

Event: `OnUpdateCmdUI ()`

Dispatch Id: *1003*

Description:
Called frequently to give integrators a good opportunity to check status of MapForce commands using `MapForceControl.QueryStatus`[1467]. Do not perform long operations in this callback.

17.7.3.3.9 OnValidationWindowUpdated

Event: `OnValidationWindowUpdated` ()

Dispatch Id: *3*

Description:
This event is triggered whenever the validation output window is updated with new information.

17.7.4 MapForceControlDocument

Properties:
Appearance 1471
BorderStyle 1471
Document 1471
IsModified 1471
Path 1471
ReadOnly 1472

Methods:
Exec 1472
New 1472
Open 1473
QueryStatus 1473
Reload 1473
Save 1473
SaveAs 1474

Events:
OnDocumentOpened 1475
OnDocumentClosed 1474
OnModifiedFlagChanged 1475
OnContextChanged 1474
OnFileChangedAlert 1475
OnActivate 1474

If the MapForceControl is integrated in the Document Level mode each document is displayed in an own object of type `MapForceControlDocument`. The `MapForceControlDocument` contains only one document at the time but can be reused to display different files one after another.

This object is a complete ActiveX control.

```
CLSID: DFBB0871-DAFE-4502-BB66-08CEB7DF5255
ProgID: Altova.MapForceControlDocument
```

17.7.4.1 Properties

The following properties are defined:

ReadOnly 1472
IsModified 1471
Path 1471
Appearance 1471
BorderStyle 1471

Access to MapForceAPI:
Document 1471

© 2016-2022 Altova GmbH

17.7.4.1.1 Appearance

Property: `Appearance` as short

Dispatch Id: *-520*

Description:
A value not equal to `0` displays a client edge around the document control. Default value is `0`.

17.7.4.1.2 BorderStyle

Property: `BorderStyle` as short

Dispatch Id: *-504*

Description:
A value of `1` displays the control with a thin border. Default value is `0`.

17.7.4.1.3 Document

Property: `Document` as Document

Dispatch Id: *1*

Description:
The `Document` property gives access to the `Document` object of the MapForce automation server API. This interface provides additional functionality which can be used with the document loaded in the control. The property is read-only.

17.7.4.1.4 IsModified

Property: `IsModified` as boolean (read-only)

Dispatch Id: *1006*

Description:
`IsModified` is *true* if the document content has changed since the last open, reload or save operation. It is *false*, otherwise.

17.7.4.1.5 Path

Property: `Path` as string

Dispatch Id: *1005*

Description:
Sets or gets the full path name of the document loaded into the control.

17.7.4.1.6 ReadOnly

Property: `ReadOnly` as boolean

Dispatch Id: *1007*

Description:
Using this property you can turn on and off the read-only mode of the document. If `ReadOnly` is `true` it is not possible to do any modifications.

17.7.4.2 Methods

The following methods are defined:

Document handling:
`New` 1472
`Open` 1473
`Reload` 1473
`Save` 1473
`SaveAs` 1474

Command Handling:
`Exec` 1472
`QueryStatus` 1473

17.7.4.2.1 Exec

Method: `Exec` (`nCmdID` as long) as boolean

Dispatch Id: *8*

Description:
`Exec` calls the MapForce command with the ID `nCmdID`. If the command can be executed, the method returns `true`. This method should be called only if there is currently an active document available in the application.

To get commands organized according to their menu structure, use the `MainMenu` 1464 property of MapForceControl. To get toolbar commands, use the `Toolbars` 1465 property of the MapForceControl.

17.7.4.2.2 New

Method: `New` () as boolean

Dispatch Id: *1000*

© 2016-2022 Altova GmbH

Description:
This method initializes a new document inside the control.

17.7.4.2.3 Open

Method: `Open` (`strFileName` as string) as boolean

Dispatch Id: *1001*

Description:
`Open` loads the file `strFileName` as the new document into the control.

17.7.4.2.4 QueryStatus

Method: `QueryStatus` (`nCmdID` as long) as long

Dispatch Id: *9*

Description:
`QueryStatus` returns the enabled/disabled and checked/unchecked status of the command specified by `nCmdID`. The status is returned as a bit mask.

Bit	Value	Name	Meaning
0	1	Supported	Set if the command is supported.
1	2	Enabled	Set if the command is enabled (can be executed).
2	4	Checked	Set if the command is checked.

This means that if `QueryStatus` returns `0` the command ID is not recognized as a valid MapForce command. If `QueryStatus` returns a value of `1` or `5` the command is disabled. The client should call the `QueryStatus` method of the document control if there is currently an active document available in the application.

17.7.4.2.5 Reload

Method: `Reload` () as boolean

Dispatch Id: *1002*

Description:
`Reload` updates the document content from the file system.

17.7.4.2.6 Save

Method: `Save` () as boolean

Dispatch Id: *1003*

Description:
`Save` saves the current document at the location `Path` 1471.

17.7.4.2.7 SaveAs

Method: `SaveAs` (`strFileName` as string) as boolean

Dispatch Id: *1004*

Description:
`SaveAs` sets `Path` 1471 to *strFileName* and then saves the document to this location.

17.7.4.3 Events

The MapForceControlDocument ActiveX control provides following connection point events:

`OnDocumentOpened` 1475
`OnDocumentClosed` 1474
`OnModifiedFlagChanged` 1475
`OnContextChanged` 1474
`OnFileChangedAlert` 1475
`OnActivate` 1474
`OnSetEditorTitle` 1476

17.7.4.3.1 OnActivate

Event: `OnActivate` ()

Dispatch Id: *1005*

Description:
This event is triggered when the document control is activated, has the focus, and is ready for user input.

17.7.4.3.2 OnContextChanged

Event: `OnContextChanged` (`i_strContextName` as String, `i_bActive` as bool) as bool

Dispatch Id: *1004*

Description: None

17.7.4.3.3 OnDocumentClosed

Event: `OnDocumentClosed` (`objDocument` as Document)

© 2016-2022 Altova GmbH

Dispatch Id: *1001*

Description:
This event is triggered whenever the document loaded into this control is closed. The argument `objDocument` is a `Document` object from the MapForce automation interface and should be used with care.

17.7.4.3.4 OnDocumentOpened

Event: `OnDocumentOpened` (`objDocument` as Document)

Dispatch Id: *1000*

Description:
This event is triggered whenever a document is opened in this control. The argument `objDocument` is a `Document` object from the MapForce automation interface, and can be used to query for more details about the document, or perform additional operations.

17.7.4.3.5 OnDocumentSaveAs

Event: `OnContextDocumentSaveAs` (`i_strFileName` as String)

Dispatch Id: *1007*

Description:
This event is triggered when this document gets internally saved under a new name.

17.7.4.3.6 OnFileChangedAlert

Event: `OnFileChangedAlert` () as bool

Dispatch Id: *1003*

Description:
This event is triggered when the file loaded into this document control is changed on the hard disk by another application. Clients should return true, if they handled the event, or false, if MapForce should handle it in its customary way, i.e. prompting the user for reload.

17.7.4.3.7 OnModifiedFlagChanged

Event: `OnModifiedFlagChanged` (`i_bIsModified` as boolean)

Dispatch Id: *1002*

Description:
This event gets triggered whenever the document changes between modified and unmodified state. The parameter *i_bIsModifed* is *true* if the document contents differs from the original content, and *false*, otherwise.

17.7.4.3.8 OnSetEditorTitle

Event: OnSetEditorTitle ()

Dispatch Id: *1006*

Description:
This event is being raised when the contained document is being internally renamed.

17.7.5 MapForceControlPlaceHolder

Properties available for all kinds of placeholder windows:
`PlaceholderWindowID` 1477

Properties for project placeholder window:
`Project` 1477

Methods for project placeholder window:
`OpenProject` 1477
`CloseProject` 1477

The `MapForceControlPlaceHolder` control is used to show the additional MapForce windows like Overview, Library or Project window. It is used like any other ActiveX control and can be placed anywhere in the client application.

```
CLSID: FDEC3B04-05F2-427d-988C-F03A85DE53C2
ProgID: Altova.MapForceControlPlaceHolder
```

17.7.5.1 Properties

The following properties are defined:

`PlaceholderWindowID` 1477

Access to MapForceAPI:
`Project` 1477

17.7.5.1.1 Label

Property: `Label` as String (read-only)

Dispatch Id: *1001*

Description:
This property gives access to the title of the placeholder. The property is read-only.

 © 2016-2022 Altova GmbH

17.7.5.1.2 PlaceholderWindowID

Property: `PlaceholderWindowID as MapForceControlPlaceholderWindow`[1479]

Dispatch Id: *1*

Description:
This property specifies which MapForce window should be displayed in the client area of the control. The `PlaceholderWindowID` can be set at any time to any valid value of the `MapForceControlPlaceholderWindow`[1479] enumeration. The control changes its state immediately and shows the new MapForce window.

17.7.5.1.3 Project

Property: `Project` as Project (read-only)

Dispatch Id: *2*

Description:
The `Project` property gives access to the `Project` object of the MapForce automation server API. This interface provides additional functionality which can be used with the project loaded into the control. The property will return a valid project interface only if the placeholder window has `PlaceholderWindowID`[1477] with a value of `MapForceXProjectWindow (=3)`. The property is read-only.

17.7.5.2 Methods

The following method is defined:

`OpenProject`[1477]
`CloseProject`[1477]

17.7.5.2.1 OpenProject

Method: `OpenProject` (`strFileName` as `string`) as `boolean`
Dispatch Id: *3*
Description:
`OpenProject` loads the file `strFileName` as the new project into the control. The method will fail if the placeholder window has a **`PlaceholderWindowID`**[1477] different to `MapForceXProjectWindow (=3)`.

17.7.5.2.2 CloseProject

Method: `CloseProject ()`

Dispatch Id: *4*

Description:
`CloseProject` closes the project loaded by the control. The method will fail if the placeholder window has a `PlaceholderWindowID`[1477] different to `MapForceXProjectWindow (=3)`.

17.7.5.3 Events

The MapForceControlPlaceholder ActiveX control provides following connection point events:

`OnModifiedFlagChanged`[1478]

17.7.5.3.1 OnModifiedFlagChanged

Event: `OnModifiedFlagChanged` (`i_bIsModified` as boolean)

Dispatch Id: *1*

Description:
This event gets triggered only for placeholder controls with a `PlaceholderWindowID`[1477] of `MapForceXProjectWindow (=3)`. The event is fired whenever the project content changes between modified and unmodified state. The parameter *i_bIsModifed* is *true* if the project contents differs from the original content, and *false*, otherwise.

17.7.5.3.2 OnSetLabel

Event: `OnSetLabel(i_strNewLabel` as string`)`

Dispatch Id: *1000*

Description:
Raised when the title of the placeholder window is changed.

17.7.6 Enumerations

The following enumerations are defined:

`ICActiveXIntegrationLevel`[1478]
`MapForceControlPlaceholderWindow`[1479]

17.7.6.1 ICActiveXIntegrationLevel

Possible values for the IntegrationLevel[1464] property of the MapForceControl.

```
ICActiveXIntegrationOnApplicationLevel  = 0
ICActiveXIntegrationOnDocumentLevel     = 1
```


© 2016-2022 Altova GmbH

17.7.6.2 MapForceControlPlaceholderWindow

This enumeration contains the list of the supported additional MapForce windows.

```
MapForceXNoWindow                 = -1
MapForceXLibraryWindow            = 0
MapForceXOverviewWindow           = 1
MapForceXValidationWindow         = 2
MapForceXProjectWindow            = 3
MapForceXDebuggerValuesWindow     = 4
MapForceXDebuggerContextWindow    = 5
MapForceXDebuggerPointsWindow     = 6
```

18 Anexos

Estos anexos contienen datos técnicos sobre MapForce e información importante sobre las licencias. También contiene unan lista de términos clave específicos de MapForce y los productos relacionados con MapForce. Esta sección se divide en los apartados siguientes:

- Notas sobre compatibilidad 1481
- Información sobre motores 1485
- Datos técnicos 1590
- Información sobre licencias 1593

© 2016-2022 Altova GmbH

18.1 Notas sobre compatibilidad

MapForce es una aplicación de 32/64 bits para Windows compatible con los sistemas operativos siguientes:

- Windows 7 SP1 con actualización de la plataforma, Windows 8, Windows 10, Windows 11
- Windows Server 2008 R2 SP1 con actualización de la plataforma o superior

Las ediciones Enterprise y Professional son compatibles con plataformas de 64 bits.

MapForce también puede utilizarse como complemento para estos entornos de desarrollo integrado:

- Visual Studio 2012/2013/2015/2017/2019/2022 Véase Complemento de UModel para Visual Studio 999.
- Eclipse 2022-09, 2022-06, 2022-03, 2021-12. Véase Complemento de MapForce para Eclipse 1002.

También puede integrar MapForce con los productos de Microsoft Office:

- Puede usar MapForce para asignar datos a y desde bases de datos Access. Para ver qué versiones son compatibles consulte Bases de datos SQL 159.
- Puede generar la documentación de la asignación en Word 2000 o versiones posteriores. Véase Generar la documentación de la asignación 915.

18.1.1 Orígenes y destinos de datos compatibles

Al cambiar el lenguaje de transformación de una asignación de MapForce, es posible que algunas funciones no sean compatibles con ese lenguaje en concreto Esta tabla recoge, ordenados por lenguaje, todos los orígenes y destinos de datos compatibles con **MapForce Professional Edition**.

Notas:

- *Built-in* permite ejecutar la asignación haciendo clic en la pestaña **Resultados** de MapForce o bien con MapForce Server.

Formato de la asignación		XSLT 1.0	XSLT 2.0	XSLT 3.0	XQuery	C++	C#	Java	BUILT-IN
XML[1]		●	●	●	●	●	●	●	●
CSV y texto						●	●	●	●
Archivos binarios									●
Bases de datos[2]	ADO					●	●		●
	ADO.NET						●		●
	JDBC							●	●

Formato de la asignación		XSLT 1.0	XSLT 2.0	XSLT 3.0	XQuery	C++	C#	Java	BUILT-IN
	Native SQLite								●
	Native PostgreSQL								●
	ODBC					●	●	●	●

Notas al pie:

1. XML con procesamiento de firmas digitales solo es compatible con MapForce Enterprise Edition si se usa el lenguaje de transformación BUILT-IN.
2. Pueden existir limitaciones en función del tipo de base de datos y del entorno de destino. Para más información consulte Asignaciones de BD en distintos entornos de ejecución 160.

18.1.2 Funciones compatibles en el código generado

En esta tabla se recogen las funciones relevantes para la generación de código y el alcance de la compatibilidad con MapForce **Professional Edition** para cada lenguaje.

Función	XSLT 1.0	XSLT 2.0	XSLT 3.0	XQuery	C++	C#	Java	BUILT-IN
Pasar parámetros a la asignación 477	●	●	●	●	●	●	●	●
Suministrar los nombres de los archivos de entrada dinámicamente desde la asignación 892	●	●	●	●	●	●	●	●
Suministrar nombres de archivos comodín como entrada de la asignación 892 1		●	●	●	●	●	●	●
Generar los nombres de los archivos de salida dinámicamente desde la asignación 892		●	●		●	●	●	●

© 2016-2022 Altova GmbH

Función	XSLT 1.0	XSLT 2.0	XSLT 3.0	XQuery	C++	C#	Java	BUILT-IN
Obtener valores de cadena de una asignación 488	●	●	●		●	●	●	●
Variables 492		●	●	●	●	●	●	●
Ordenar componentes 505		●	●	●				●
Agrupar funciones 511		●	●		●	●	●	●
Filtros 543	●	●	●	●	●	●	●	●
Combinar componentes 517								●
Componentes Asignación de valores 533	●	●	●	●	●	●	●	●
Valores predeterminados y funciones 558								●
Asignaciones: excepciones 529		●	●	●	●	●	●	●
Redistribuir y serializar cadenas de texto 903 2								●
Nombres de nodo dinámicos 850		●	●	●	●	●	●	●
BD: inserciones masivas 320								●
Instrucciones SQL SELECT para BD sin parámetros de entrada 359					●	●	●	●
Instrucciones SQL SELECT para BD con parámetros de entrada 359								●
BD: procedimientos almacenados 395								●

Función	XSLT 1.0	XSLT 2.0	XSLT 3.0	XQuery	C++	C#	Java	BUILT-IN
Gestionar excepciones de BD 422 3					●	●	●	●
Seguimiento de BD y registro de errores 435								●
Generar archivos de ejecución de MapForce Server 955								●
Implementar asignaciones en FlowForce Server 958								●
Leer datos de archivos binarios 471								●
Escribir datos en archivos binarios 471								●

Notas al pie:

1. XSLT 2.0, XSLT 3.0 y XQuery utilizan la función `fn:collection`. La implementación en los motores XSLT 2.0, XSLT 3.0 y XQuery de Altova resuelve comodines. Los demás motores pueden comportarse de modo distinto.
2. Para JSON, la redistribución y serialización también se admite en Java y C#.
3. Se pueden gestionar las excepciones de BD si el lenguaje de la asignación es compatible con el controlador de la BD conectada actualmente, como se indica en la tabla anterior.

© 2016-2022 Altova GmbH

18.2 Información sobre motores

Esta sección ofrece información sobre las características de implementación del validador XML de Altova y de los motores XSLT 1.0, XSLT 2.0 y XQuery de Altova.

18.2.1 Información sobre motores XSLT y XQuery

Los motores XSLT y XQuery de MapForce siguen las especificaciones del W3C y, por tanto, son más estrictos que otros motores anteriores de Altova, como los de las versiones antiguas de XMLSpy. Por consiguiente, MapForce señala algunos errores leves que antes no se notificaban en la versión anterior de estos motores.

Por ejemplo:

- Se notifica un error de tipo (`err:XPTY0018`) si el resultado de un operador de ruta de acceso contiene tanto nodos como no nodos.
- Se notifica un error de tipo (`err:XPTY0019`) si `E1` en una expresión XPath `E1/E2` no da como resultado una secuencia de nodos.

Si encuentra este tipo de errores, modifique el documento XSLT/XQuery o el documento de instancia según corresponda.

Esta sección describe características relacionadas con la implementación de los motores e incluye estos apartados:

- XSLT 1.0 1485
- XSLT 2.0 1485
- XQuery 1.0 1487

18.2.1.1 XSLT 1.0

El motor XSLT 1.0 de MapForce cumple con la recomendación XSLT 1.0 del 16 de noviembre de 1999 y con la recomendación XPath 1.0 del 16 de noviembre de 1999, ambas del W3C.

Nota sobre la implementación

Cuando el atributo `method` de `xsl:output` tiene el valor HTML o si selecciona de forma predeterminada el formato de salida HTML, los caracteres especiales del archivo XML o XSLT se insertan en el documento HTML como referencias de caracteres HTML. Por ejemplo, el carácter U+00A0 (la referencia de carácter hexadecimal para un espacio de no separación) se inserta en el código HTML como referencia de carácter (` ` o ` `) o como referencia de entidad (` `).

18.2.1.2 XSLT 2.0

Temas de este apartado:

- Especificaciones con las que cumple el motor 1486
- Compatibilidad con versiones antiguas 1486
- Espacios de nombres 1486
- Compatibilidad con esquemas 1487
- Comportamiento propio de esta implementación 1487

Especificaciones

El motor XSLT 2.0 de MapForce cumple con la recomendación XSLT 2.0 del 23 de enero de 2007 y la recomendación XPath 2.0 del 14 de diciembre de 2010, ambas del W3C.

Compatibilidad con versiones antiguas

El motor XSLT 2.0 es compatible con versiones previas. Esto solamente es relevante cuando se utiliza el motor XSLT 2.0 para procesar una hoja de estilos XSLT 1.0. Tenga en cuenta que los resultados obtenidos con el motor XSLT 1.0 pueden ser diferentes a los obtenidos con el motor XSLT 2.0 en modo de compatibilidad con versiones antiguas.

Espacios de nombres

En su hoja de estilos XSLT 2.0 debe declarar estos espacios de nombres para poder usar los constructores de tipo y las funciones disponibles en XSLT 2.0. Los prefijos que aparecen a continuación son los que se suelen usar, pero puede usar otros prefijos si quiere.

Espacio de nombres	Prefijo	URI del espacio de nombres
Tipos XML Schema	`xs:`	`http://www.w3.org/2001/XMLSchema`
Funciones XPath 2.0	`fn:`	`http://www.w3.org/2005/xpath-functions`

Estos espacios de nombres se suelen declarar en el elemento `xsl:stylesheet` o en el elemento `xsl:transform`:

```
<xsl:stylesheet version="2.0"
   xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
   xmlns:xs="http://www.w3.org/2001/XMLSchema"
   xmlns:fn="http://www.w3.org/2005/xpath-functions"
   ...
</xsl:stylesheet>
```

Es necesario tener en cuenta que:

- El motor XSLT 2.0 utiliza el espacio de nombres Funciones XPath 2.0 y XQuery 1.0 como **espacio de nombres de funciones predeterminado**. Esto significa que puede usar funciones XPath 2.0 y XSLT 2.0 en su hoja de estilos sin prefijos. Si declara el espacio de nombres Funciones XPath 2.0 en su hoja de estilos con un prefijo, podrá usar el prefijo asignado en la declaración.
- Cuando se usan constructores de tipo y tipos del espacio de nombres XML Schema, el prefijo utilizado en la declaración de espacio de nombres se debe usar en la llamada al constructor de tipo (por ejemplo, `xs:date`).
- Algunas funciones XPath 2.0 se llaman igual que algunos tipos de datos de XML Schema. Por ejemplo, las funciones XPath `fn:string` y `fn:boolean` y los tipos de datos de XML Schema

 © 2016-2022 Altova GmbH

`xs:string` y `xs:boolean`. Por tanto, si usa la expresión `string('Hello')`, la expresión se evalúa como `fn:string('Hello')` y no como `xs:string('Hello')`.

Compatibilidad con esquemas

El motor XSLT 2.0 está preparado para esquemas de modo que puede usar tipos de esquema definidos por el usuario y la instrucción `xsl:validate`.

Comportamiento propio de esta implementación

Más abajo puede ver cómo se ocupa el motor XSLT 2.0 de algunos aspectos del comportamiento de las funciones XSLT 2.0 relacionadas con esta implementación.

`xsl:result-document`
También son compatibles estas codificaciones específicas de Altova: `x-base16tobinary` y `x-base64tobinary`.

`function-available`
Esta función mira si hay funciones del ámbito disponibles (funciones XSLT, XPath y de extensión).

`unparsed-text`
El atributo `href` acepta (i) rutas de acceso relativas para archivos que estén en la carpeta del URI base y (ii) rutas de acceso absolutas con o sin el protocolo `file://`. También son compatibles estas codificaciones específicas de Altova: `x-binarytobase16` y `x-binarytobase64`.

`unparsed-text-available`
El atributo `href` acepta (i) rutas de acceso relativas para archivos que estén en la carpeta del URI base y (ii) rutas de acceso absolutas con o sin el protocolo `file://`. También son compatibles estas codificaciones específicas de Altova: `x-binarytobase16` y `x-binarytobase64`.

Nota: estos valores de codificación estaban implementados en el ya descatalogado AltovaXML pero ya no se utilizan (son obsoletos): `base16tobinary`, `base64tobinary`, `binarytobase16` y `binarytobase64`.

18.2.1.3 XQuery 1.0

Temas de este apartado:

- Especificaciones con las que cumple el motor 1488
- Compatibilidad con esquemas 1488
- Codificación 1488
- Espacios de nombres 1488
- Fuentes XML y validación 1489
- Comprobación de tipos estática y dinámica 1489
- Módulos biblioteca 1489
- Funciones externas 1489
- Intercalaciones 1489
- Precisión de datos numéricos 1490
- Compatibilidad con instrucciones XQuery 1490

Especificaciones compatibles

El motor XQuery 1.0 de MapForce cumple con la recomendación XQuery 1.0 del 14 de diciembre de 2010 del W3C. El estándar XQuery concede libertad a la hora de implementar muchas características. A continuación explicamos cómo se implementaron estas características en el motor XQuery 1.0 de MapForce.

Compatibilidad con esquemas

El motor XQuery 1.0 está preparado para esquemas.

Codificación

El motor XQuery 1.0 es compatible con las codificaciones de caracteres UTF-8 y UTF-16.

Espacios de nombres

Se predefinen estos URI de espacios de nombres y sus enlaces asociados.

Espacio de nombres	Prefijo	URI del espacio de nombres
Tipos XML Schema	`xs:`	`http://www.w3.org/2001/XMLSchema`
Schema instance	`xsi:`	`http://www.w3.org/2001/XMLSchema-instance`
Funciones integradas	`fn:`	`http://www.w3.org/2005/xpath-functions`
Funciones locales	`local:`	`http://www.w3.org/2005/xquery-local-functions`

Es importante tener en cuenta que:

- El motor XQuery 1.0 entiende que los prefijos de la tabla anterior están enlazados con los correspondientes espacios de nombres.
- Como el espacio de nombres de funciones integradas (ver tabla) es el espacio de nombres de funciones predeterminado de XQuery, no es necesario usar el prefijo `fn:` cuando se invocan funciones integradas (por ejemplo, `string("Hello")` llamará a la función `fn:string`). No obstante, el prefijo `fn:` se puede utilizar para llamar a una función integrada sin necesidad de declarar el espacio de nombres en el prólogo de la consulta (por ejemplo: `fn:string("Hello")`).
- Puede cambiar el espacio de nombres de funciones predeterminado declarando la expresión `default function namespace` en el prólogo de la consulta.
- Cuando use tipos del espacio de nombres XML Schema, puede usar el prefijo `xs:` sin necesidad de declarar los espacios de nombres de forma explícita ni enlazar estos prefijos a los espacios de nombres en el prólogo de la consulta. (Ejemplo: `xs:date` y `xs:yearMonthDuration`.) Si quiere usar otros prefijos para el espacio de nombres de XML Schema, estos se deben declarar en el prólogo de la consulta. (Ejemplo: `declare namespace alt = "http://www.w3.org/2001/XMLSchema"; alt:date("2004-10-04")`.)
- Recuerde que los tipos de datos `untypedAtomic`, `dayTimeDuration` y `yearMonthDuration` se movieron del espacio de nombres XPath Datatypes al espacio de nombres XML Schema (es decir, ahora es `xs:yearMonthDuration`.)

Si se asignaron mal los espacios de nombres para funciones, constructores de tipo, pruebas de nodo, etc., se emite un error. Sin embargo, recuerde que algunas funciones se llaman igual que los tipos de datos de

 © 2016-2022 Altova GmbH

esquema (p. ej. `fn:string` y `fn:boolean`.) (Se definen `xs:string` y `xs:boolean`.) El prefijo del espacio de nombres determina si se usa la función o el constructor de tipo.

Documento XML de origen y validación

Los documentos XML que se utilizan para ejecutar un documento XQuery con el motor XQuery 1.0 deben tener un formato XML correcto. Sin embargo, no es necesario que sean válidos con respecto a un esquema XML. Si el archivo no es válido, el archivo no válido se carga sin información de esquema. Si el archivo XML está asociado a un esquema externo y es válido con respecto a dicho esquema, se genera información posterior a la validación de esquema, que se utilizará para evaluar la consulta.

Comprobación de tipos estática y dinámica

En la fase de análisis estático se revisan aspectos de la consulta como la sintaxis, si existen referencias externas (p. ej. para módulos), si las funciones y variables que se invocan están definidas, etc. Si se detecta un error en la fase de análisis estático, se notifica y la ejecución se interrumpe.

La comprobación dinámica de tipos se realiza en tiempo de ejecución, cuando la consulta se ejecuta. Si un tipo no es compatible con los requisitos de una operación, se emite un error. Por ejemplo, la expresión `xs:string("1") + 1` devuelve un error porque la operación de suma no se puede llevar a cabo en un operando de tipo `xs:string`.

Módulos biblioteca

Los módulos biblioteca almacenan funciones y variables para poder volver a utilizarlas. El motor XQuery 1.0 es compatible con el uso de módulos almacenados en un **solo archivo XQuery externo**. Dicho archivo de módulo debe incluir una declaración `module` en su prólogo que apunte a un espacio de nombres de destino Por ejemplo:

```
module namespace libns="urn:module-library";
declare variable $libns:company := "Altova";
declare function libns:webaddress() { "http://www.altova.com" };
```

Todas las funciones y variables declaradas en el módulo pertenecen al espacio de nombres asociado al módulo. El módulo se importa en un archivo XQuery con la instrucción `import module` del prólogo de la consulta. La instrucción `import module` solamente importa funciones y variables declaradas directamente en el archivo de módulo biblioteca. Por ejemplo:

```
import module namespace modlib = "urn:module-library" at "modulefilename.xq";
if        ($modlib:company = "Altova")
then       modlib:webaddress()
else       error("No match found.")
```

Funciones externas

Las funciones externas son incompatibles con el motor XQuery 1.0, es decir, todas las expresiones que usen la palabra clave `external`. Por ejemplo:

```
declare function hoo($param as xs:integer) as xs:string external;
```

Intercalaciones

La intercalación predeterminada es la intercalación de puntos de código Unicode, que compara las cadenas de texto según sus puntos de código Unicode. Otras intercalaciones compatibles son las intercalaciones ICU que se enumeran aquí[1490]. Para usar una intercalación concreta, indique su URI tal y como aparece en la lista de intercalaciones compatibles[1490]. Las comparaciones de cadenas de texto, incluidas las comparaciones para las funciones `fn:max` y `fn:min`, se harán según la intercalación especificada. Si no se indica la opción de intercalación, se utiliza la intercalación de puntos de código Unicode predeterminada.

Precisión de tipos numéricos

- El tipo de datos `xs:integer` es de precisión arbitraria, es decir, puede representar un número de dígitos cualquiera.
- El tipo de datos `xs:decimal` tiene un límite de 20 dígitos después del punto decimal.
- Los tipos de datos `xs:float` y `xs:double` tienen una precisión limitada de 15 dígitos.

Compatibilidad con instrucciones XQuery

La instrucción `Pragma` no es compatible. Si se encuentra, se ignora y en su lugar se evalúa la expresión de reserva.

18.2.2 Funciones XSTL y XPath/XQuery

Esta sección enumera las funciones de extensión de Altova y otras funciones de extensión que se pueden utilizar con expresiones XPath y XQuery. Las funciones de extensión de Altova se pueden usar con los motores XSLT y XQuery de Altova y ofrecen algunas funciones más aparte de las que están disponibles en las bibliotecas de funciones definidas en los estándares del W3C.

Aspectos generales

Es necesario tener en cuenta estos puntos generales:

- A las funciones de las bibliotecas de funciones `core` definidas en las especificaciones W3C se les puede llamar sin un prefijo. Esto se debe a que los motores XSLT y XQuery leen funciones sin prefijo como si pertenecieran a un espacio de nombres de funciones predeterminado, que es el que se especifica en las especificaciones de las funciones XPath y XQuery `http://www.w3.org/2005/xpath-functions`. Si este espacio de nombres se declara explícitamente en un documento XSLT o XQuery, el prefijo utilizado en la declaración de espacio de nombres también se puede usar en el nombre de las funciones.
- Por lo general, si una función espera como argumento una secuencia de un elemento y se suministra una secuencia de más de un elemento, entonces se devuelve un error.
- Todas las comparaciones de cadena se realizan usando la intercalación de puntos de código Unicode.
- Los resultados que son QName se serializan de esta forma `[prefijo:]nombrelocal`.

Precisión de xs:decimal

El término *precisión* hace referencia al número de dígitos del número y la especificación exige un mínimo de 18 dígitos. Para las operaciones de división que dan un resultado de tipo `xs:decimal`, la precisión es de 19 dígitos después del punto decimal sin redondear.

© 2016-2022 Altova GmbH

Uso horario implícito
Cuando hay que comparar dos valores `date`, `time` o `dateTime`, es necesario conocer el uso horario de los valores que se deben comparar. Cuando el uso horario no se conoce de forma explícita, se usa el uso horario implícito, que se toma del reloj del sistema. Para probar cuál es su valor puede utilizar la función `fn:implicit-timezone()`.

Intercalaciones
La intercalación predeterminada es la intercalación de puntos de código Unicode, que compara las cadenas de texto según sus puntos de código Unicode. El motor usa el algoritmo de intercalación Unicode. Otras intercalaciones compatibles son las intercalaciones ICU que aparecen más abajo. Para usar una intercalación indique su URI tal y como aparece en la tabla más abajo. Las comparaciones de cadenas de texto (incluidas las que usan las funciones `fn:max` y `fn:min`) se harán según la intercalación especificada. Si no se especifica la opción de intercalación, se usa la intercalación predeterminada de puntos de código Unicode.

Idioma	Identificadores URI
`da`: danés	`da_DK`
`de`: alemán	`de_AT, de_BE, de_CH, de_DE, de_LI, de_LU`
`en`: inglés	`en_AS, en_AU, en_BB, en_BE, en_BM, en_BW, en_BZ, en_CA, en_GB, en_GU, en_HK, en_IE, en_IN, en_JM, en_MH, en_MP, en_MT, en_MU, en_NA, en_NZ, en_PH, en_PK, en_SG, en_TT, en_UM, en_US, en_VI, en_ZA, en_ZW`
`es`: español	`es_419, es_AR, es_BO, es_CL, es_CO, es_CR, es_DO, es_EC, es_ES, es_GQ, es_GT, es_HN, es_MX, es_NI, es_PA, es_PE, es_PR, es_PY, es_SV, es_US, es_UY, es_VE`
`fr`: francés	`fr_BE, fr_BF, fr_BI, fr_BJ, fr_BL, fr_CA, fr_CD, fr_CF, fr_CG, fr_CH, fr_CI, fr_CM, fr_DJ, fr_FR, fr_GA, fr_GN, fr_GP, fr_GQ, fr_KM, fr_LU, fr_MC, fr_MF, fr_MG, fr_ML, fr_MQ, fr_NE, fr_RE, fr_RW, fr_SN, fr_TD, fr_TG`
`it`: italiano	`it_CH, it_IT`
`ja`: jaoonés	`ja_JP`
`nb`: noruego bokmål	`nb_NO`
`nl`: holandés	`nl_AW, nl_BE, nl_NL`
`nn`: noruego nynorsk	`nn_NO`
`pt`: portugués	`pt_AO, pt_BR, pt_GW, pt_MZ, pt_PT, pt_ST`
`ru`: ruso	`ru_MD, ru_RU, ru_UA`
`sv`: sueco	`sv_FI, sv_SE`

Eje del espacio de nombres
El eje del espacio de nombres se dejó de utilizar en XPath 2.0. Sin embargo, los ejes de espacio de nombres son compatibles con la aplicación. Para acceder a la información sobre el espacio de nombres con mecanismos de XPath 2.0, utilice las funciones `fn:in-scope-prefixes()`, `fn:namespace-uri()` y

`fn:namespace-uri-for-prefix()`.

18.2.2.1 Funciones de extensión de Altova

Puede utilizar todas las funciones de extensión de Altova en sus expresiones XPath/XQuery. Con ellas conseguirá funciones adicionales no disponibles en la biblioteca de funciones estándar de XPath, XQuery y XSLT. Las funciones de extensión de Altova están en el **espacio de nombres** **`http://www.altova.com/xslt-extensions`** y en esta sección se presentan con el prefijo, que se supone estará enlazado al espacio de nombres señalado. Tenga en cuenta que en futuras versiones del producto algunas funciones pueden dejar de ser compatibles o su comportamiento puede cambiar. Por tanto, consulte siempre la documentación del producto para conocer el funcionamiento de estas funciones en cada versión del producto.

Las funciones definidas en las especificaciones XPath/XQuery Functions del W3C se pueden usar en (i) expresiones XPath en contextos XSLT y en (ii) expresiones XQuery en documentos XQuery. En esta documentación las funciones que se pueden usar en el primer contexto (XPath en XSLT) llevan el símbolo **XP** y se les llama funciones XPath. Las funciones que se pueden usar en contextos XQuery llevan el símbolo **XQ** y funcionan como funciones XQuery. Las especificaciones XSLT del W3C también definen funciones que se pueden usar en expresiones XPath en documentos XSLT. Estas funciones llevan el símbolo **XSLT** y se les denomina funciones XSLT. Por cada función se indica en qué versión de XPath/XQuery y XSLT se puede usar (ver símbolos más abajo). Las funciones de las bibliotecas de funciones XPath/XQuery y XSLT aparecen sin prefijo. Las funciones de extensión de otras bibliotecas, como las funciones de extensión de Altova, aparecen con un prefijo.

Funciones XPath (en expresiones XPath en XSLT):	**XP1** **XP2** **XP3.1.1**
Funciones XSLT (en expresiones XPath en XSLT):	**XSLT1** **XSLT2** **XSLT3**
Funciones XQuery (en expresiones XQuery en XQuery):	**XQ1** **XQ3.1**

Funciones XSLT 1493

Las funciones XSLT solo se pueden utilizar en expresiones XPath en un contexto XSLT (igual que las funciones XSLT 2.0 `current-group()` o `key()`). Estas funciones no están pensadas para contextos no XSLT (p. ej. contextos XQuery) y, por tanto, no funcionarán en contextos que no sean XSLT. Recuerde que las funciones XSLT para XBRL solamente se pueden utilizar con ediciones de los productos de Altova compatibles con XBRL.

Funciones XPath/XQuery

Las funciones XPath/XQuery se pueden utilizar en expresiones XPath, en contextos XSLT y en expresiones XQuery:

- Funciones de fecha y hora 1496
- Funciones de geoubicación 1514
- Funciones relacionadas con imágenes 1525
- Funciones numéricas 1530

 © 2016-2022 Altova GmbH

- Funciones de secuencia 1552
- Funciones de cadena 1561
- Funciones varias 1568

18.2.2.1.1 Funciones XSLT

Las **funciones de extensión XSLT** pueden utilizarse en expresiones XPath en contextos XSLT y no funcionan en contextos que no sean XSLT (por ejemplo, en contextos XQuery).

Nota sobre el nombre de las funciones y lenguajes

Puede utilizar todas las funciones de extensión de Altova en sus expresiones XPath/XQuery. Con ellas conseguirá funciones adicionales no disponibles en la biblioteca de funciones estándar de XPath, XQuery y XSLT. Las funciones de extensión de Altova están en el **espacio de nombres** **`http://www.altova.com/xslt-extensions`** y en esta sección se presentan con el prefijo, que se supone estará enlazado al espacio de nombres señalado. Tenga en cuenta que en futuras versiones del producto algunas funciones pueden dejar de ser compatibles o su comportamiento puede cambiar. Por tanto, consulte siempre la documentación del producto para conocer el funcionamiento de estas funciones en cada versión del producto.

Funciones XPath (en expresiones XPath en XSLT):	XP1 XP2 XP3.1.1
Funciones XSLT (en expresiones XPath en XSLT):	XSLT1 XSLT2 XSLT3
Funciones XQuery (en expresiones XQuery en XQuery):	XQ1 XQ3.1

Funciones generales

distinct-nodes [altova:]

`altova:distinct-nodes(`*`node()*`***`)`** como **`node()*`** XSLT1 XSLT2 XSLT3

Toma un conjunto de nodos como entrada y devuelve el mismo conjunto menos los nodos que tengan el mismo valor (es decir, devuelve los nodos que son únicos). La comparación se hace con la función XPath/XQuery `fn:deep-equal`.

Ejemplo

- **`altova:distinct-nodes`**`(country)` devuelve todos los nodos secundarios `country` excepto los que tengan el mismo valor.

evaluate [altova:]

`altova:evaluate(ExpresiónXPath` *`como xs:string`*`[, ValorDe$p1, ... ValorDe$pN])` XSLT1 XSLT2 XSLT3

Toma una expresión XPath, pasada como cadena, como argumento obligatorio. Devuelve el resultado de la expresión evaluada. Por ejemplo, **`evaluate('//Name[1]')`** devuelve el contenido del primer elemento `Name` del documento. Observe que para pasar la expresión `//Name[1]` como cadena basta con ponerla

entre comillas simples.

La función `altova:evaluate` puede tomar más argumentos, que son los valores de las variables del ámbito que se llaman `p1`, `p2`, `p3`... `pN`. Recuerde que (i) las variables deben definirse con nombres de tipo `pX`, siendo `X` un entero; (ii) los argumentos de la función `altova:evaluate` (*ver firma más abajo*), a partir del segundo argumento, ofrecen los valores de las variables, correspondiendo la secuencia de argumentos a la secuencia numérica de variables: `p1` corresponde a `pN` y el segundo argumento será el valor de la variable `p1`, el tercer argumento al de la variable `p2`, y así sucesivamente; (iii) los valores de las variables deben ser de tipo `item*`.

Ejemplo

```
<xsl:variable name="xpath" select="'$p3, $p2, $p1'" />
<xsl:value-of select="altova:evaluate($xpath, 10, 20, 'hi')" />
```

da el resultado `"hi 20 10"`

En el ejemplo anterior puede observar que:

- El segundo argumento de la expresión `altova:evaluate` es el valor asignado a la variable `$p1`, el tercer argumento es el valor asignado a la variable `$p2` y así sucesivamente.
- Observe que el cuarto argumento de la función es un valor de cadena porque va entre comillas simples.
- El atributo `select` del elemento `xs:variable` suministra la expresión XPath. Como esta expresión debe ser de tipo `xs:string`, se pone entre comillas simples.

Más ejemplos

- ```
  <xsl:variable name="xpath" select="'$p1'" />
  <xsl:value-of select="altova:evaluate($xpath, //Name[1])" />
  ```
  El resultado es el valor del primer elemento `Name`.

- ```
  <xsl:variable name="xpath" select="'$p1'" />
  <xsl:value-of select="altova:evaluate($xpath, '//Name[1]')" />
  ```
  El resultado es "`//Name[1]`*"*

La función de extensión `altova:evaluate()` es muy práctica cuando una expresión XPath de la hoja de estilos XSLT contiene partes que se deben evaluar de forma dinámica. Por ejemplo, imagine que el usuario selecciona un criterio de ordenación y este criterio se almacena en el atributo `UserReq/@sortkey`. En la hoja de estilos podría tener esta expresión:

```
<xsl:sort select="altova:evaluate(../UserReq/@sortkey)" order="ascending"/>
```

La función `altova:evaluate()` lee el atributo `sortkey` del elemento secundario `UserReq` del primario del nodo de contexto. Imagine que el valor del atributo `sortkey` es `Price`. En ese caso, la función `altova:evaluate()` devuelve `Price`, que se convierte en el valor del atributo `select`:

```
<xsl:sort select="Price" order="ascending"/>
```

Si esta instrucción `sort` aparece dentro del contexto de un elemento llamado `Order`, entonces los elementos `Order` se ordenan según el valor de los secundarios `Price`. Otra opción es que, si el valor de `@sortkey` fuera `Date`, por ejemplo, entonces los elementos `Order` se ordenarían según el valor de los

 © 2016-2022 Altova GmbH

secundarios `Date`. Es decir, el criterio de ordenación para `Order` se selecciona del atributo `sortkey` en tiempo de ejecución. Esto no sería posible con una expresión como:

```
<xsl:sort select="../UserReq/@sortkey" order="ascending"/>
```

En este caso, el criterio de ordenación sería el propio atributo `sortkey`, no `Price` ni `Date` (ni otro contenido actual de `sortkey`).

Nota: el contexto estático incluye espacios de nombres, tipos y funciones (pero no variables) del entorno de llamada. El URI base y el espacio de nombres predeterminado se heredan.

Más ejemplos

- Variables estáticas: `<xsl:value-of select="$i3, $i2, $i1" />`
  El resultado es los valores de las tres variables.

- Expresión XPath dinámica con variables dinámicas:
  ```
  <xsl:variable name="xpath" select="'$p3, $p2, $p1'" />
  <xsl:value-of select="altova:evaluate($xpath, 10, 20, 30)" />
  ```
  El resultado es "30 20 10"

- Expresión XPath dinámica sin variables dinámicas:
  ```
  <xsl:variable name="xpath" select="'$p3, $p2, $p1'" />
  <xsl:value-of select="altova:evaluate($xpath)" />
  ```
  Error: no se definió la variable para $p3.

encode-for-rtf [altova:]

altova:encode-for-rtf(entrada *como xs:string*, **conservarEspaciosEnBlanco** *como xs:boolean*, **conservarLíneasNuevas** *como xs:boolean*) como **xs:string** XSLT2 XSLT3

Convierte la cadena de entrada en código para RTF. Los espacios en blanco y las líneas nuevas se conservan o no dependiendo del valor booleano especificado para los correspondientes parámetros.

[Subir 1493]

Funciones XBRL

Las funciones XBRL de Altova solo funcionan en las ediciones de los productos de Altova que son compatibles con XBRL.

xbrl-footnotes [altova:]

altova:xbrl-footnotes(*node()***)** como **node()*** XSLT2 XSLT3

Toma un nodo como argumento de entrada y devuelve el conjunto de nodos de nota al pie XBRL al que hace referencia el nodo de entrada.

xbrl-labels [altova:]

altova:xbrl-labels(*xs:QName*, *xs:string***)** como **node()*** XSLT2 XSLT3

Toma dos argumentos de entrada: un nombre de nodo y la ubicación del archivo de taxonomía en el que está el nodo. La función devuelve los nodos de etiqueta XBRL asociados al nodo de entrada.

[Subir 1493]

18.2.2.1.2 Funciones XPath/XQuery: de fecha y hora

Las funciones de extensión de fecha y hora de Altova se pueden usar en expresiones XPath y XQuery y permiten procesar datos almacenados en tipos de datos XML Schema de fecha y hora. Estas funciones se pueden usar con los **motores XPath 3.0 y XQuery 3.0** de Altova y están disponibles en contextos XPath/XQuery.

Nota sobre el nombre de las funciones y lenguajes

Puede utilizar todas las funciones de extensión de Altova en sus expresiones XPath/XQuery. Con ellas conseguirá funciones adicionales no disponibles en la biblioteca de funciones estándar de XPath, XQuery y XSLT. Las funciones de extensión de Altova están en el **espacio de nombres** `http://www.altova.com/xslt-extensions` y en esta sección se presentan con el prefijo, que se supone estará enlazado al espacio de nombres señalado. Tenga en cuenta que en futuras versiones del producto algunas funciones pueden dejar de ser compatibles o su comportamiento puede cambiar. Por tanto, consulte siempre la documentación del producto para conocer el funcionamiento de estas funciones en cada versión del producto.

Funciones XPath (en expresiones XPath en XSLT):	XP1 XP2 XP3.1.1
Funciones XSLT (en expresiones XPath en XSLT):	XSLT1 XSLT2 XSLT3
Funciones XQuery (en expresiones XQuery en XQuery):	XQ1 XQ3.1

Funciones agrupadas según su funcionalidad

- Agregar una duración a xs:dateTime y devolver xs:dateTime 1497
- Agregar una duración a xs:date y devolver xs:date 1499
- Agregar una duración a xs:time y devolver xs:time 1501
- Recuperar duraciones y aplicarles formato 1500
- Quitar la zona horaria de las funciones que generan la fecha/hora actual 1502
- Devolver el número de días, horas, minutos y segundos de duraciones 1503
- Devolver el día de la semana de una fecha como número entero 1505
- Devolver el número de semana de una fecha como número entero 1506
- Generar la fecha, la hora y el tipo de duración a partir de los componentes léxicos de cada tipo 1507
- Construir un tipo date, dateTime o a partir de la cadena de entrada 1509
- Funciones para calcular la edad 1510
- Funciones para calcular el tiempo Unix 1512

Funciones por orden alfabético

 © 2016-2022 Altova GmbH

altova:add-days-to-date 1499
altova:add-days-to-dateTime 1497
altova:add-hours-to-dateTime 1497
altova:add-hours-to-time 1501
altova:add-minutes-to-dateTime 1497
altova:add-minutes-to-time 1501
altova:add-months-to-date 1499
altova:add-months-to-dateTime 1497
altova:add-seconds-to-dateTime 1497
altova:add-seconds-to-time 1501
altova:add-years-to-date 1499
altova:add-years-to-dateTime 1497
altova:age 1510
altova:age-details 1510
altova:build-date 1507
altova:build-duration 1507
altova:build-time 1507
altova:current-dateTime-no-TZ 1502
altova:current-date-no-TZ 1502
altova:current-time-no-TZ 1502
altova:date-no-TZ 1502
altova:dateTime-from-epoch 1512
altova:dateTime-from-epoch-no-TZ 1512
altova:dateTime-no-TZ 1502
altova:days-in-month 1503
altova:epoch-from-dateTime 1512
altova:hours-from-dateTimeDuration-accumulated 1503
altova:minutes-from-dateTimeDuration-accumulated 1503
altova:seconds-from-dateTimeDuration-accumulated 1503
altova:format-duration 1500
altova:parse-date 1509
altova:parse-dateTime 1509
altova:parse-duration 1500
altova:parse-time 1509
altova:time-no-TZ 1502
altova:weekday-from-date 1505
altova:weekday-from-dateTime 1505
altova:weeknumber-from-date 1506
altova:weeknumber-from-dateTime 1506

[**Subir** 1496]

Agregar una duración a xs:dateTime XP3.1 XQ3.1

Estas funciones sirven para agregar una duración a **`xs:dateTime`** y devuelven **`xs:dateTime`**. El tipo `xs:dateTime` tiene el formato `SSAA-MM-DDThh:mm:ss.sss`. Se trata de la concatenación de los formatos `xs:date` y `xs:time` separados por la letra `T`. Si quiere puede usar un sufijo de zona horaria (por ejemplo **`+01:00`**).

▼ add-years-to-dateTime [altova:]

`altova:add-years-to-dateTime(FechaHora` *`as xs:dateTime`*, **`Años`** *`as xs:integer`***`)`** como **`xs:dateTime`** XP3.1 XQ3.1

Añade una duración en años un valor de fecha y hora. El segundo argumento es el número de años que

se debe añadir al valor de fecha y hora dado como primer argumento. El resultado es de tipo `xs:dateTime`.

Ejemplos

- **altova:add-years-to-dateTime**(xs:dateTime("2014-01-15T14:00:00"), 10) devuelve 2024-01-15T14:00:00
- **altova:add-years-to-dateTime**(xs:dateTime("2014-01-15T14:00:00"), -4) devuelve 2010-01-15T14:00:00

add-months-to-dateTime [altova:]

altova:add-months-to-dateTime(FechaHora *as xs:dateTime*, **Meses** *as xs:integer***)** como **xs:dateTime** XP3.1 XQ3.1

Añade una duración en meses a un valor de fecha y hora. El segundo argumento es el número de meses que se debe añadir al valor de fecha y hora dado como primer argumento. El resultado es de tipo `xs:dateTime`.

Ejemplos

- **altova:add-months-to-dateTime**(xs:dateTime("2014-01-15T14:00:00"), 10) devuelve 2014-11-15T14:00:00
- **altova:add-months-to-dateTime**(xs:dateTime("2014-01-15T14:00:00"), -2) devuelve 2013-11-15T14:00:00

add-days-to-dateTime [altova:]

altova:add-days-to-dateTime(FechaHora *as xs:dateTime*, **Días** *as xs:integer***)** como **xs:dateTime** XP3.1 XQ3.1

Añade una duración en días a un valor de fecha y hora. El segundo argumento es el número de días que se deben añadir al valor de fecha y hora dado como primer argumento. El resultado es de tipo `xs:dateTime`.

Ejemplos

- **altova:add-days-to-dateTime**(xs:dateTime("2014-01-15T14:00:00"), 10) devuelve 2014-01-25T14:00:00
- **altova:add-days-to-dateTime**(xs:dateTime("2014-01-15T14:00:00"), -8) devuelve 2014-01-07T14:00:00

add-hours-to-dateTime [altova:]

altova:add-hours-to-dateTime(FechaHora *as xs:dateTime*, **Horas** *as xs:integer***)** como **xs:dateTime** XP3.1 XQ3.1

Añade una duración en horas a un valor de fecha y hora. El segundo argumento es el número de horas que se deben añadir al valor de fecha y hora dado como primer argumento. El resultado es de tipo `xs:dateTime`.

Ejemplos

- **altova:add-hours-to-dateTime**(xs:dateTime("2014-01-15T13:00:00"), 10) devuelve 2014-

© 2016-2022 Altova GmbH

01-15T23:00:00
- **altova:add-hours-to-dateTime**(xs:dateTime("2014-01-15T13:00:00"), -8) devuelve 2014-01-15T05:00:00

add-minutes-to-dateTime [altova:]

altova:add-minutes-to-dateTime(FechaHora *as xs:dateTime*, **Minutos** *as xs:integer*) como **xs:dateTime** XP3.1 XQ3.1
Añade una duración en minutos a un valor de fecha y hora. El segundo argumento es el número de minutos que se debe añadir al valor de fecha y hora dado como primer argumento. El resultado es de tipo xs:dateTime.

Ejemplos

- **altova:add-minutes-to-dateTime**(xs:dateTime("2014-01-15T14:10:00"), 45) devuelve 2014-01-15T14:55:00
- **altova:add-minutes-to-dateTime**(xs:dateTime("2014-01-15T14:10:00"), -5) devuelve 2014-01-15T14:05:00

add-seconds-to-dateTime [altova:]

altova:add-seconds-to-dateTime(FechaHora *as xs:dateTime*, **Segundos** *as xs:integer*) como **xs:dateTime** XP3.1 XQ3.1
Añade una duración en segundos a un valor de fecha y hora. El segundo argumento es el número de segundos que se debe añadir al valor de fecha y hora dado como primer argumento. El resultado es de tipo xs:dateTime.

Ejemplos

- **altova:add-seconds-to-dateTime**(xs:dateTime("2014-01-15T14:00:10"), 20) devuelve 2014-01-15T14:00:30
- **altova:add-seconds-to-dateTime**(xs:dateTime("2014-01-15T14:00:10"), -5) devuelve 2014-01-15T14:00:05

[Subir 1496]

Agregar una duración a xs:date XP3.1 XQ3.1

Estas funciones agregan una duración a **xs:date** y devuelven **xs:date**. El tipo xs:date tiene el formato SSAA-MM-DD.

add-years-to-date [altova:]

altova:add-years-to-date(Fecha *as xs:date*, **Años** *as xs:integer*) como **xs:date** XP3.1 XQ3.1
Añade una duración en años a una fecha. El segundo parámetro es el número de años que se debe añadir a la fecha dada como primer argumento. El resultado es de tipo xs:date.

Ejemplos

- **altova:add-years-to-date**(xs:date("2014-01-15"), 10) devuelve 2024-01-15

- **altova:add-years-to-date**(xs:date("2014-01-15"), -4) devuelve 2014-01-15

add-months-to-date [altova:]

altova:add-months-to-date(Fecha *as xs:date*, **Meses** *as xs:integer*) como **xs:date** XP3.1 XQ3.1
Añade una duración en meses a una fecha. El segundo argumento es el número de meses que se debe añadir a la fecha dada como primer argumento. El resultado es de tipo xs:date.

Ejemplos

- **altova:add-months-to-date**(xs:date("2014-01-15"), 10) devuelve 2014-11-15
- **altova:add-months-to-date**(xs:date("2014-01-15"), -2) devuelve 2013-11-15

add-days-to-date [altova:]

altova:add-days-to-date(Fecha *as xs:date*, **Días** *as xs:integer*) como **xs:date** XP3.1 XQ3.1
Añade una duración en días a una fecha. El segundo argumento es el número de días que se deben añadir a la fecha dad como primer argumento. El resultado es de tipo xs:date.

Ejemplos

- **altova:add-days-to-date**(xs:date("2014-01-15"), 10) devuelve 2014-01-25
- **altova:add-days-to-date**(xs:date("2014-01-15"), -8) devuelve 2014-01-07

[Subir 1496]

Recuperar duraciones y aplicarles formato XP3.1 XQ3.1

Estas funciones analizan la entrada **xs:duration** o **xs:string** y devuelven, respectivamente, **xs:string** o **xs:duration**.

format-duration [altova:]

altova:format-duration(Duración *como xs:duration*, **Imagen** *como xs:string*) como **xs:string** XP3.1 XQ3.1
Aplica formato a una duración, que se suministra como primer argumento, en base a la cadena de imagen dada como segundo argumento. El resultado es una cadena de texto cuyo formato se ajusta a la cadena de imagen.

Ejemplos

- **altova:format-duration**(xs:duration("P2DT2H53M11.7S"), "Días:[D01] Horas:[H01] Minutos:[m01] Segundos:[s01] Fracciones:[f0]") devuelve "Días:02 Horas:02 Minutos:53 Segundos:11 Fracciones:7"
- **altova:format-duration**(xs:duration("P3M2DT2H53M11.7S"), "Meses:[M01] Días:[D01] Horas:[H01] Minutos:[m01]") devuelve "Meses:03 Días:02 Horas:02 Minutos:53"

parse-duration [altova:]

 © 2016-2022 Altova GmbH

altova:parse-duration(CadenaEntrada *como xs:string*, **Imagen** *como xs:string*) como **xs:duration** XP3.1 XQ3.1
Toma una cadena con patrón como primer argumento y una cadena de imagen como segundo argumento. La cadena de entrada se analiza en base a la cadena de imagen y se devuelve un `xs:duration`.

Ejemplos

- **altova:parse-duration**("Días:02 Horas:02 Minutos:53 Segundos:11 Fracciones:7", "Días:[D01] Horas:[H01] Minutes:[m01] Segundos:[s01] Fracciones:[f0]") devuelve "P2DT2H53M11.7S"
- **altova:parse-duration**("Meses:03 Días:02 Horas:02 Minutos:53 Segundos:11 Fracciones:7", "Meses:[M01] Días:[D01] Horas:[H01] Minutos:[m01]") devuelve "P3M2DT2H53M"

[Subir 1496]

Agregar una duración a xs:time XP3.1 XQ3.1

Estas funciones agregan una duración a **xs:time** y devuelven **xs:time**. El tipo `xs:time` tiene un formato léxico de este tipo `hh:mm:ss.sss`. Si quiere, puede añadir un sufijo de zona horaria. La letra `Z` indica (UTC). Las demás zonas horarias se representan con la diferencia que hay entre ellas y la zona UTC: `+hh:mm` o `-hh:mm`. Si falta el valor de zona horaria, se entiende que se desconoce (no se da por hecho que es UTC)

add-hours-to-time [altova:]

altova:add-hours-to-time(Hora *as xs:time*, **Horas** *as xs:integer*) como **xs:time** XP3.1 XQ3.1
Añade una duración en horas a una hora. El segundo argumento es el número de horas que se debe añadir a la hora dada como primer argumento. El resultado es de tipo `xs:time`.

Ejemplos

- **altova:add-hours-to-time**(xs:time("11:00:00"), 10) devuelve 21:00:00
- **altova:add-hours-to-time**(xs:time("11:00:00"), -7) devuelve 04:00:00

add-minutes-to-time [altova:]

altova:add-minutes-to-time(Hora *as xs:time*, **Minutos** *as xs:integer*) como **xs:time** XP3.1 XQ3.1
Añade una duración en minutos a una hora. El segundo argumento es el número de minutos que se debe añadir a la hora dada como primer argumento. El resultado es de tipo `xs:time`.

Ejemplos

- **altova:add-minutes-to-time**(xs:time("14:10:00"), 45) devuelve 14:55:00
- **altova:add-minutes-to-time**(xs:time("14:10:00"), -5) devuelve 14:05:00

add-seconds-to-time [altova:]

altova:add-seconds-to-time(Hora *as xs:time*, **Segundos** *as xs:integer*) como **xs:time** XP3.1 XQ3.1
Añade una duración en segundos a una hora. El segundo argumento es el número de segundos que se debe añadir a la hora dada como primer argumento. El resultado es de tipo `xs:time`. El componente

Segundos puede estar comprendido entre `0` y `59.999`.

Ejemplos

- **altova:add-seconds-to-time**(xs:time("14:00:00"), 20) devuelve 14:00:20
- **altova:add-seconds-to-time**(xs:time("14:00:00"), 20.895) devuelve 14:00:20.895

[Subir 1496]

Quitar la parte de zona horaria de los tipos de datos date/time XP3.1 XQ3.1

Estas funciones quitan la zona horaria de los valores **xs:dateTime**, **xs:date** o **xs:time** actuales. Tenga en cuenta que la diferencia entre `xs:dateTime` y `xs:dateTimeStamp` es que en esta última la parte de zona horaria es obligatoria (mientras que en la primera es opcional). Es decir, el formato de un valor `xs:dateTimeStamp` puede ser `SSAA-MM-DDThh:mm:ss.sss±hh:mm` o `SSAA-MM-DDThh:mm:ss.sssZ`. Si la fecha y la hora se leen del reloj del sistema como `xs:dateTimeStamp`, la función `current-dateTime-no-TZ()` se puede usar para quitar la zona horaria.

current-date-no-TZ [altova:]

altova:current-date-no-TZ() como **xs:date** XP3.1 XQ3.1
Esta función no toma ningún argumento. Quita la parte de zona horaria de la función `current-date()` (que es la fecha actual según el reloj del sistema) y devuelve un valor de tipo `xs:date`.

Ejemplos

Si la fecha actual es **2014-01-15+01:00**:

- **altova:current-date-no-TZ**() devuelve 2014-01-15

current-dateTime-no-TZ [altova:]

altova:current-dateTime-no-TZ() como **xs:dateTime** XP3.1 XQ3.1
Esta función no toma ningún argumento. Quita la parte de zona horaria de `current-dateTime()` (que es la fecha y hora actual según el reloj del sistema) y devuelve un valor de tipo `xs:dateTime`.

Ejemplos

Si la fecha y hora actual es **2014-01-15T14:00:00+01:00**:

- **altova:current-dateTime-no-TZ**() devuelve 2014-01-15T14:00:00

current-time-no-TZ [altova:]

altova:current-time-no-TZ() **asxs:time** XP3.1 XQ3.1
Esta función no toma ningún argumento. Quita la parte de zona horaria de `current-time()` (que es la hora actual según el reloj del sistema) y devuelve un valor de tipo `xs:time`.

Ejemplos

© 2016-2022 Altova GmbH

Si la hora actual es **14:00:00+01:00**:

- **altova:current-time-no-TZ**() devuelve 14:00:00

date-no-TZ [altova:]

altova:date-no-TZ(InputDate *as xs:date***)** como **xs:date** XP3.1 XQ3.1
Esta función toma un argumento xs:date, del que elimina la parte timezone y devuelve un valor xs:date. Observe que la fecha permanece intacta.

Ejemplos

- **altova:date-no-TZ**(xs:date("2014-01-15+01:00")) devuelve 2014-01-15

dateTime-no-TZ [altova:]

altova:dateTime-no-TZ(InputDateTime *as xs:dateTime***)** como **xs:dateTime** XP3.1 XQ3.1
Esta función toma un argumento xs:dateTime, del que elimina la parte timezone, y devuelve un valor xs:dateTime. Observe que tanto la fecha como la hora permanecen intactas.

Ejemplos

- **altova:dateTime-no-TZ**(xs:date("2014-01-15T14:00:00+01:00")) devuelve 2014-01-15T14:00:00

time-no-TZ [altova:]

altova:time-no-TZ(HoraEntrada *como xs:time***)** como **xs:time** XP3.1 XQ3.1
Esta función toma un argumento xs:time, quita la parte de la zona horaria y devuelve un valor xs:time. Tenga en cuenta que la hora no se modifica.

Ejemplos

- **altova:time-no-TZ**(xs:time("14:00:00+01:00")) devuelve 14:00:00

[Subir 1496]

Devolver el número de días, horas, minutos y segundos de duraciones XP3.1 XQ3.1

Estas funciones devuelven el número de días en un mes y el número de horas, minutos y segundos de las duraciones correspondientes.

days-in-month [altova:]

altova:days-in-month(Year *as xs:integer*, **Month** *as xs:integer***)** como **xs:integer** XP3.1 XQ3.1
Devuelve el número de días en el mes indicado. El mes se indica con los argumentos Year y Month.

Ejemplos

- **altova:days-in-month**(2018, 10) devuelve 31
- **altova:days-in-month**(2018, 2) devuelve 28
- **altova:days-in-month**(2020, 2) devuelve 29

hours-from-dayTimeDuration-accumulated

altova:hours-from-dayTimeDuration-accumulated(DayAndTime *como xs:duration***)** como **xs:integer** XP3.1 XQ3.1

Devuelve el número total de horas de la duración enviada por el argumento `DayAndTime` (que es de tipo `xs:duration`). Las horas de los componentes `Day` y `Time` se agregan juntos para dar como resultado un número entero. Una hora nueva son 60 minutos enteros. Las duraciones negativas dan como resultado un valor de hora negativo.

Ejemplos

- **altova:hours-from-dayTimeDuration-accumulated**(xs:duration("P5D") devuelve 120, que es el número total de horas en 5 días.
- **altova:hours-from-dayTimeDuration-accumulated**(xs:duration("P5DT2H") devuelve 122, que es el número total de horas en 5 días más 2 horas.
- **altova:hours-from-dayTimeDuration-accumulated**(xs:duration("P5DT2H60M") devuelve 123, que es el número total de horas en 5 días más 2 horas y 60 mins.
- **altova:hours-from-dayTimeDuration-accumulated**(xs:duration("P5DT2H119M") devuelve 123, que es el número total de horas en 5 días más 2 horas y 119 mins.
- **altova:hours-from-dayTimeDuration-accumulated**(xs:duration("P5DT2H120M") devuelve 124, que es el número total de horas en 5 días más 2 horas y 120 mins.
- **altova:hours-from-dayTimeDuration-accumulated**(xs:duration("-P5DT2H") devuelve -122

minutes-from-dayTimeDuration-accumulated

altova:minutes-from-dayTimeDuration-accumulated(DayAndTime *como xs:duration***)** como **xs:integer** XP3.1 XQ3.1

Devuelve el número total de minutos de la duración enviada por el argumento `DayAndTime` (que es de tipo `xs:duration`). Los minutos de los componentes `Day` y `Time` se agregan juntos para dar como resultado un número entero. Las duraciones negativas dan como resultado un valor de minutos negativo.

Ejemplos

- **altova:minutes-from-dayTimeDuration-accumulated**(xs:duration("PT60M") devuelve 60
- **altova:minutes-from-dayTimeDuration-accumulated**(xs:duration("PT1H") devuelve 60, que es el número total de minutos en 1 hora.
- **altova:minutes-from-dayTimeDuration-accumulated**(xs:duration("PT1H40M") devuelve 100
- **altova:minutes-from-dayTimeDuration-accumulated**(xs:duration("P1D") devuelve 1440, que es el número total de minutos en 1 día.
- **altova:minutes-from-dayTimeDuration-accumulated**(xs:duration("-P1DT60M") devuelve -1500

seconds-from-dayTimeDuration-accumulated

altova:seconds-from-dayTimeDuration-accumulated(DayAndTime *como xs:duration***)** como

© 2016-2022 Altova GmbH

xs:integer XP3.1 XQ3.1
Devuelve el número total de segundos de la duración enviada por el argumento `DayAndTime` (que es de tipo `xs:duration`). Los segundos de los componentes Day y Time se agregan juntos para dar como resultado un número entero. Las duraciones negativas dan como resultado un valor de segundos negativo.

Ejemplos

- **altova:seconds-from-dayTimeDuration-accumulated**(xs:duration("PT1M") devuelve 60, que es el número total de segundos en 1 minuto.
- **altova:seconds-from-dayTimeDuration-accumulated**(xs:duration("PT1H") devuelve 3600, que es el número total de segundos en 1 hora.
- **altova:seconds-from-dayTimeDuration-accumulated**(xs:duration("PT1H2M") devuelve 3720
- **altova:seconds-from-dayTimeDuration-accumulated**(xs:duration("P1D") devuelve 86400, que es el número total de segundos en 1 día.
- **altova:seconds-from-dayTimeDuration-accumulated**(xs:duration("-P1DT1M") devuelve -86460

Obtener el día de la semana de **xs:dateTime** o **xs:date** XP3.1 XQ3.1

Estas funciones obtienen el día de la semana (como entero) de `xs:dateTime` o `xs:date`. Los días de la semana se numeran del `1` al `7` (usando el formato EE UU, es decir `Domingo =1`). En el formato europeo la semana empieza el lunes (es decir, `Lunes=1`). Para establecer el formato EE UU (`Domingo=1`) use el entero `0` allí donde se acepte un entero para indicar el formato.

weekday-from-dateTime [altova:]

altova:weekday-from-dateTime(DateTime *como xs:dateTime*) **como xs:integer** XP3.1 XQ3.1
Toma una fecha como único argumento y devuelve el día de la semana de la fecha dada como número entero. Los días de la semana se numeran del 1 al 7 empezando por `Domingo=1`. Si necesita usar el formato europeo (donde `Lunes=1`), utilice la otra firma de esta función (*ver más abajo*).

Ejemplos

- **altova:weekday-from-dateTime**(xs:dateTime("2014-02-03T09:00:00")) devuelve 2, lo cual significa "Lunes"..

altova:weekday-from-dateTime(DateTime *como xs:dateTime*, **Formato** *como xs:integer*) **como xs:integer** XP3.1 XQ3.1
Toma una fecha como primer argumento y devuelve el día de la semana de la fecha dada como número entero. Si el segundo argumento (número entero) es `0`, entonces los días de la semana se numeran del `1` al `7` empezando por `Domingo=1`. Si el segundo argumento es un entero distinto de `0`, entonces `Lunes=1`. Si falta el segundo argumento, la función se lee como en la firma anterior (*ver más arriba*).

Ejemplos

- **altova:weekday-from-dateTime**(xs:dateTime("2014-02-03T09:00:00"), 1) devuelve 1, lo cual significa "Lunes"
- **altova:weekday-from-dateTime**(xs:dateTime("2014-02-03T09:00:00"), 4) devuelve 1, lo cual significa "Lunes"
- **altova:weekday-from-dateTime**(xs:dateTime("2014-02-03T09:00:00"), 0) devuelve 2, lo cual significa "Lunes"

weekday-from-date [altova:]

altova:weekday-from-date(Date *como xs:date*) **como xs:integer** XP3.1 XQ3.1
Toma una fecha como único argumento y devuelve el día de la semana de la fecha dada como número entero. Los días de la semana se numeran del 1 al 7 empezando por `Domingo=1`. Si necesita usar el formato europeo (donde `Lunes=1`), utilice la otra firma de esta función (*ver más abajo*).

Ejemplos

- `altova:weekday-from-date(xs:date("2014-02-03+01:00"))` devuelve `2`, lo cual significa lunes.

altova:weekday-from-date(Date *como xs:date*, **Formato** *como xs:integer*) **como xs:integer** XP3.1 XQ3.1
Toma una fecha como primer argumento y devuelve el día de la semana de la fecha dada como número entero. Si el segundo argumento (`Formato`) es `0`, entonces los días de la semana se numeran del `1` al `7` empezando por `Domingo=1`. Si el segundo argumento es un entero distinto de `0`, entonces `Lunes=1`. Si falta el segundo argumento, la función se lee como en la firma anterior (*ver más arriba*).

Ejemplos

- `altova:weekday-from-date(xs:date("2014-02-03"), 1)` devuelve `1`, lo cual significa "Lunes"
- `altova:weekday-from-date(xs:date("2014-02-03"), 4)` devuelve `1`, lo cual significa "Lunes"
- `altova:weekday-from-date(xs:date("2014-02-03"), 0)` devuelve `2`, lo cual significa "Lunes".

[Subir 1496]

Devolver el número de semana de xs:dateTime o xs:date XP2 XQ1 XP3.1 XQ3.1

Estas funciones devuelven el número de semana (como número entero) de `xs:dateTime` o `xs:date`. El número de la semana está disponible en el formato de calendario estadounidense, europeo e islámico. La razón de que los números de semana difieran en cada uno de estos calendarios es que en cada uno de ellos se considera un día diferente para el inicio de la semana (p. ej. en el formato estadounidense el primer día de la semana es el domingo).

weeknumber-from-date [altova:]

altova:weeknumber-from-date(Fecha *como xs:date*, **Calendario** *como xs:integer*) como **xs:integer** XP2 XQ1 XP3.1 XQ3.1
Devuelve como número entero el número de semana del argumento **Fecha** dado. El segundo argumento (**Calendario**) indica el sistema de calendario que se debe seguir.
Estos son los valores permitidos para el argumento **Calendario**:

- **0 = Calendario estadounidense** (la semana comienza el domingo)
- **1 = Calendario estándar ISO o europeo** (la semana comienza el lunes)
- **2 = Calendario islámico** (la semana comienza el sábado)

El valor predeterminado es **0**.

Ejemplos

- `altova:weeknumber-from-date(xs:date("2014-03-23"), 0)` devuelve `13`
- `altova:weeknumber-from-date(xs:date("2014-03-23"), 1)` devuelve `12`

 © 2016-2022 Altova GmbH

- **altova:weeknumber-from-date**(xs:date("2014-03-23"), 2) devuelve 13
- **altova:weeknumber-from-date**(xs:date("2014-03-23")) devuelve 13

El día de la fecha de los ejemplos anteriores (2014-03-23) es un domingo. Por tanto, en este caso, el calendario estadounidense y el islámico van una semana por delante del calendario europeo.

weeknumber-from-dateTime [altova:]

altova:weeknumber-from-dateTime(FechaHora *como xs:dateTime*, **Calendario** *como xs:integer*) como **xs:integer** XP2 XQ1 XP3.1 XQ3.1

Devuelve como entero el día de la semana del argumento **FechaHora** dado. El segundo argumento (**Calendario**) indica el sistema de calendario que se debe seguir.
Estos son los valores permitidos para el argumento **Calendario**:

- **0 = Calendario estadounidense** (la semana comienza el domingo)
- **1 = Calendario estándar ISO o europeo** (la semana comienza el lunes)
- **2 = Calendario islámico** (la semana comienza el sábado)

El valor predeterminado es **0**.

Ejemplos

- **altova:weeknumber-from-dateTime**(xs:dateTime("2014-03-23T00:00:00"), 0) devuelve 13
- **altova:weeknumber-from-dateTime**(xs:dateTime("2014-03-23T00:00:00"), 1) devuelve 12
- **altova:weeknumber-from-dateTime**(xs:dateTime("2014-03-23T00:00:00"), 2) devuelve 13
- **altova:weeknumber-from-dateTime**(xs:dateTime("2014-03-23T00:00:00")) devuelve 13

El día de dateTime de los ejemplos anteriores (2014-03-23T00:00:00) es un domingo. Por tanto, en este caso, el calendario estadounidense y el islámico van una semana por delante del calendario europeo.

[Subir 1496]

Generar tipos de datos de fecha, hora y duración a partir de sus componentes léxicos XP3.1 XQ3.1

Estas funciones toman los componentes léxicos de los tipos de datos xs:date, xs:time y xs:duration como argumentos de entrada y los combinan para generar el tipo de datos correspondiente.

build-date [altova:]

altova:build-date(Año *as xs:integer*, **Mes** *as xs:integer*, **Fecha** *as xs:integer*) como **xs:date** XP3.1 XQ3.1

Los argumentos son el año, el mes y la fecha respectivamente. Se combinan para generar un valor de tipo xs:date. Los valores de los enteros deben estar en el intervalo de esa fecha en particular. Por ejemplo, el segundo argumento (para el mes) no puede ser mayor que 12.

Ejemplos

- **altova:build-date**(2014, 2, 03) devuelve 2014-02-03

build-time [altova:]

altova:build-time(Horas *as xs:integer*, **Minutos** *as xs:integer*, **Segundos** *as xs:integer*) **como xs:time** XP3.1 XQ3.1

El primer, segundo y tercer argumentos son la hora (0 - 23), los minutos (0 - 59) y los segundos (0 - 59) respectivamente. Se combinan para generar un valor de tipo xs:time. Los valores de los enteros deben estar dentro del intervalo correcto de esa parte de tiempo concreta. Por ejemplo, el segundo argumento (Minutos) no puede ser mayor que 59. Para añadir la parte de uso horario al valor, use la firma que aparece más abajo.

Ejemplos

- **altova:build-time**(23, 4, 57) devuelve 23:04:57

altova:build-time(Horas *como xs:integer*, **Minutos** *como xs:integer*, **Segundos** *as xs:integer*, **TimeZone** *como xs:string*) **como xs:time** XP3.1 XQ3.1

El primer, segundo y tercer argumentos son la hora (0 - 23), los minutos (0 - 59) y los segundos (0 - 59) respectivamente. El cuarto argumento es una cadena de texto que indica la parte del valor de la zona horaria. Este cuarto argumento se combina para generar un valor de tipo xs:time. Los valores de los enteros deben estar dentro del intervalo correcto de esa parte de tiempo concreta. Por ejemplo, el segundo argumento (Minutos) no puede ser mayor que 59.

Ejemplos

- **altova:build-time**(23, 4, 57, '+1') devuelve 23:04:57+01:00

build-duration [altova:]

altova:build-duration(Años *as xs:integer*, **Meses** *as xs:integer*) como **xs:yearMonthDuration** XP3.1 XQ3.1

Toma dos argumentos para generar un valor de tipo xs:yearMonthDuration. El primer argumento da la parte Years del valor de duración, mientras que el segundo da la parte Months. Si el segundo (Months) es mayor o igual que 12, el entero se divide por 12. El cociente se añade al primer argumento para aportar la parte Years del valor de duración, mientras que el resto (de la división) da la parte Months. Para generar una duración de tipo xs:dayTimeDuration, consulte la firma siguiente.

Ejemplos

- **altova:build-duration**(2, 10) devuelve P2Y10M
- **altova:build-duration**(14, 27) devuelve P16Y3M
- **altova:build-duration**(2, 24) devuelve P4Y

altova:build-duration(Días *as xs:integer*, **Horas** *as xs:integer*, **Minutos** *as xs:integer*, **Segundos** *as xs:integer*) como **xs:dayTimeDuration** XP3.1 XQ3.1

Toma cuatro argumentos y los combina para generar un valor de tipo xs:dayTimeDuration. El primer argumento da la parte Days del valor de duración, el segundo, el tercero y el cuarto dan las partes Hours, Minutes y Seconds respectivamente. Los tres argumentos de tiempo se convierten a un valor equivalente en cuanto a la unidad mayor siguiente y el resultado se utiliza para calcular el valor total de la duración. Por ejemplo, 72 segundos se convierte en 1M+12S (1 minuto y 12 segundos) y este valor se usa para calcular el valor total de la duración. Para generar una duración de tipo xs:yearMonthDuration, consulte la firma anterior.

© 2016-2022 Altova GmbH

Ejemplos

- `altova:build-duration(2, 10, 3, 56)` devuelve `P2DT10H3M56S`
- `altova:build-duration(1, 0, 100, 0)` devuelve `P1DT1H40M`
- `altova:build-duration(1, 0, 0, 3600)` devuelve `P1DT1H`

[Subir 1496]

Construir tipos de datos date, dateTime y time a partir de una cadena de entrada XP2 XQ1 XP3.1 XQ3.1

Estas funciones toman cadenas como argumentos y construyen tipos de datos `xs:date`, `xs:dateTime` o `xs:time`. La cadena de entrada se analiza para los componentes del tipo de datos en función del argumento patrón dado.

parse-date [altova:]

altova:parse-date(Fecha *como xs:string*, **PatrónFecha** *como xs:string*) como **xs:date** XP2 XQ1 XP3.1 XQ3.1

Devuelve la cadena de entrada **Fecha** como valor **xs:date**. El segundo argumento (**PatrónFecha**) indica el patrón (secuencia de componentes) de la cadena de entrada. El argumento **PatrónFecha** se describe con los especificadores que aparecen a continuación y con cualquier separador de componentes (consulte los ejemplos más abajo).

D	Día
M	Mes
Y	Año

El patrón **PatrónFecha** debe coincidir con el patrón de **Fecha**. Como el resultado es de tipo **xs:date**, el resultado siempre tendrá el formato léxico **YYYY-MM-DD**.

Ejemplos

- `altova:parse-date(xs:string("09-12-2014"), "[D]-[M]-[Y]")` devuelve `2014-12-09`
- `altova:parse-date(xs:string("09-12-2014"), "[M]-[D]-[Y]")` devuelve `2014-09-12`
- `altova:parse-date("06/03/2014", "[M]/[D]/[Y]")` devuelve `2014-06-03`
- `altova:parse-date("06 03 2014", "[M] [D] [Y]")` devuelve `2014-06-03`
- `altova:parse-date("6 3 2014", "[M] [D] [Y]")` devuelve `2014-06-03`

parse-dateTime [altova:]

altova:parse-dateTime(FechaHora *como xs:string*, **PatrónFechaHora** *como xs:string*) como **xs:dateTime** XP2 XQ1 XP3.1 XQ3.1

Devuelve la cadena de entrada **FechaHora** como valor **xs:dateTime**. El segundo argumento (**PatrónFechaHora**) indica el patrón (secuencia de componentes) de la cadena de entrada. El argumento **PatrónFechaHora** se describe con los especificadores que aparecen a continuación y con cualquier separador de componentes (consulte los ejemplos más abajo).

D	Día
M	Mes
Y	Año
H	Hora
m	minutos
s	segundos

El patrón **PatrónFechaHora** debe coincidir con el patrón de **FechaHora**. Como el resultado es de tipo **xs:dateTime**, el resultado siempre tendrá el formato léxico **YYYY-MM-DDTHH:mm:ss**.

Ejemplos

- `altova:parse-dateTime(xs:string("09-12-2014 13:56:24"), "[M]-[D]-[Y] [H]:[m]:[s]")` devuelve `2014-09-12T13:56:24`
- `altova:parse-dateTime("time=13:56:24; date=09-12-2014", "time=[H]:[m]:[s]; date=[D]-[M]-[Y]")` devuelve `2014-12-09T13:56:24`

parse-time [altova:]

altova:parse-time(Hora *como xs:string*, **PatrónHora** *como xs:string*) como **xs:time** XP2 XQ1 XP3.1 XQ3.1

Devuelve la cadena de entrada **Hora** como valor **xs:time**. El segundo argumento (**PatrónHora**) indica el patrón (secuencia de componentes) de la cadena de entrada. El argumento **PatrónHora** se describe con los especificadores que aparecen a continuación y con cualquier separador de componentes (consulte los ejemplos más abajo).

H	Hora
m	minutos
s	segundos

El patrón **PatrónHora** debe coincidir con el patrón de **Hora**. Como el resultado es de tipo **xs:time**, el resultado siempre tendrá el formato léxico **HH:mm:ss**.

Ejemplos

- `altova:parse-time(xs:string("13:56:24"), "[H]:[m]:[s]")` devuelve `13:56:24`
- `altova:parse-time("13-56-24", "[H]-[m]")` devuelve `13:56:00`
- `altova:parse-time("time=13h56m24s", "time=[H]h[m]m[s]s")` devuelve `13:56:24`
- `altova:parse-time("time=24s56m13h", "time=[s]s[m]m[H]h")` devuelve `13:56:24`

[Subir 1496]

Funciones para calcular la edad XP3.1 XQ3.1

Estas funciones devuelven la edad que se calcula obteniendo la diferencia (i) entre la fecha del argumento de entrada y la fecha actual o (ii) entre las fechas de los dos argumentos de entrada. La función **age** devuelve la

edad en años, mientras que la función **age-details** devuelve la edad en forma de una secuencia de tres enteros (años, meses y días).

age [altova:]

altova:age(FechaInicio *as xs:date***)** como **xs:integer** XP3.1 XQ3.1
Devuelve un entero que es la edad *en años* de algún objeto, contando a partir de la fecha de inicio dada como argumento y hasta la fecha actual (tomada del reloj del sistema). Si el argumento de entrada es un año o más después que la fecha actual, el valor devuelto será negativo.

Ejemplos

Si la fecha actual es **2014-01-15**:

- **altova:age**(xs:date("2013-01-15")) devuelve 1
- **altova:age**(xs:date("2013-01-16")) devuelve 0
- **altova:age**(xs:date("2015-01-15")) devuelve -1
- **altova:age**(xs:date("2015-01-14")) devuelve 0

altova:age(FechaInicio *as xs:date*, **FechaFinal** *as xs:date***)** como **xs:integer** XP3.1 XQ3.1
Devuelve un entero que es la edad *en años* de algún objeto, contando a partir de la fecha de inicio dada como primer argumento y hasta la fecha dada como segundo argumento. El valor devuelto será negativo si el primer argumento es un año o más después que el segundo argumento.

Ejemplos

- **altova:age**(xs:date("2000-01-15"), xs:date("2010-01-15")) devuelve 10
- **altova:age**(xs:date("2000-01-15"), current-date()) devuelve 14 si la fecha actual es 2014-01-15
- **altova:age**(xs:date("2014-01-15"), xs:date("2010-01-15")) devuelve -4

age-details [altova:]

altova:age-details(FechaEntrada *as xs:date***)** como (**xs:integer**)* XP3.1 XQ3.1
Devuelve tres enteros que son los años, meses y días respectivamente que hay entre la fecha dada como argumento y la fecha actual (tomada del reloj del sistema). La suma del valor devuelto nos da el tiempo total transcurrido entre ambas fechas (entre la fecha dada y la fecha actual). La fecha de entrada puede tener un valor anterior o posterior a la fecha actual, pero esto no se indica en el valor devuelto por medio de un signo negativo o positivo. El valor devuelto siempre es positivo.

Ejemplos

Si la fecha actual es **2014-01-15**:

- **altova:age-details**(xs:date("2014-01-16")) devuelve (0 0 1)
- **altova:age-details**(xs:date("2014-01-14")) devuelve (0 0 1)
- **altova:age-details**(xs:date("2013-01-16")) devuelve (1 0 1)
- **altova:age-details**(current-date()) devuelve (0 0 0

altova:age-details(Fecha1 *as xs:date*, **Fecha2** *as xs:date***)** como (**xs:integer**)* XP3.1 XQ3.1
Devuelve tres enteros que son los años, meses y días que hay entre las dos fechas dadas por los

argumentos. La suma del valor devuelto nos da el tiempo total transcurrido entre las dos fechas de entrada. Da igual cuál de las dos fechas se da como primer argumento, la más antigua o la más reciente. El valor devuelto no indica si la fecha de entrada es anterior o posterior a la fecha actual. Es decir, el valor devuelto siempre es positivo.

Ejemplos

- **altova:age-details**(xs:date("2014-01-16"), xs:date("2014-01-15")) devuelve (0 0 1)
- **altova:age-details**(xs:date("2014-01-15"), xs:date("2014-01-16")) devuelve (0 0 1)

[Subir 1496]

Funciones para calcular el tiempo Unix XP3.1 XQ3.1

El tiempo Unix es una medida de tiempo que se usa en sistemas Unix. Se define como la cantidad de segundos transcurridos desde las 00:00:00 UTC del 1 de enero de 1970. Estas funciones convierten valores **xs:dateTime** en tiempo Unix y viceversa.

dateTime-from-epoch [altova:]

altova:dateTime-from-epoch(Epoch *como xs:decimal* **como xs:dateTime** XP3.1 XQ3.1

El tiempo Unix es una medida de tiempo que se usa en sistemas Unix. Se define como la cantidad de segundos transcurridos desde las 00:00:00 UTC del 1 de enero de 1970. La función **dateTime-from-epoch** devuelve el equivalente en **xs:dateTime** de un instante de tiempo Unix, lo ajusta a la zona horaria local e incluye la información de esa zona horaria en el resultado.

La función toma un argumento xs:decimal y devuelve un valor xs:dateTime que incluye una parte **TZ**, que indica la zona horaria. Para obtener el resultado se calcula el equivalente en **dateTime** UTC del instante de tiempo Unix y se añade a la zona horaria local (que se obtiene del reloj del sistema). Por ejemplo, si la función se ejecuta en un equipo cuya configuración sitúa en una zona horaria de +01:00 (con respecto a UTC), una vez se ha calculado el equivalente en **dateTime** se le añade una hora al resultado. La información de la zona horaria, que es una parte léxica opcional del resultado de **xs:dateTime**, también se incluye en el resultado **dateTime**. Compare este resultado con el de **dateTime-from-epoch-no-TZ** y consulte la función **epoch-from-dateTime**.

Ejemplos

La zona horaria local de los ejemplos siguientes es UTC +01:00. En consecuencia, el equivalente en **dateTime** UTC del instante de tiempo Unix indicado aumentará en una hora. La zona horaria se indica en el resultado.

- **altova:dateTime-from-epoch**(34) devuelve 1970-01-01T01:00:34+01:00
- **altova:dateTime-from-epoch**(62) devuelve 1970-01-01T01:01:02+01:00

dateTime-from-epoch-no-TZ [altova:]

altova:dateTime-from-epoch-no-TZ(Epoch *como xs:decimal* **como xs:dateTime** XP3.1 XQ3.1

El tiempo Unix es una medida de tiempo que se usa en sistemas Unix. Se define como la cantidad de segundos transcurridos desde las 00:00:00 UTC del 1 de enero de 1970. La función **dateTime-from-epoch-no-TZ** devuelve el equivalente en **xs:dateTime** de un instante de tiempo Unix y lo ajusta a la zona

horaria local pero no incluye la información de esa zona horaria en el resultado.

La función toma un argumento `xs:decimal` y devuelve un valor `xs:dateTime` que no incluye la parte **TZ**, que indica la zona horaria. Para obtener el resultado se calcula el equivalente en **dateTime** UTC del instante de tiempo Unix y se añade a la zona horaria local (que se obtiene del reloj del sistema). Por ejemplo, si la función se ejecuta en un equipo cuya configuración sitúa en una zona horaria de +01:00 (con respecto a UTC), una vez se ha calculado el equivalente en **dateTime** se le añade una hora al resultado. La información de la zona horaria, que es una parte léxica opcional del resultado de **xs:dateTime**, no se incluye en el resultado **dateTime**. Compare este resultado con el de **dateTime-from-epoch** y consulte la función **epoch-from-dateTime**.

Ejemplos

La zona horaria local de los ejemplos siguientes es UTC +01:00. En consecuencia, el equivalente en **dateTime** UTC del instante de tiempo Unix indicado aumentará en una hora. La zona horaria no se indica en el resultado.

- **altova:dateTime-from-epoch**(34) devuelve 1970-01-01T01:00:34
- **altova:dateTime-from-epoch**(62) devuelve 1970-01-01T01:01:02

epoch-from-dateTime [altova:]

altova:epoch-from-dateTime(dateTimeValue *como xs:dateTime*) **como xs:decimal** XP3.1 XQ3.1

El tiempo Unix es una medida de tiempo que se usa en sistemas Unix. Se define como la cantidad de segundos transcurridos desde las 00:00:00 UTC del 1 de enero de 1970. La función **epoch-from-dateTime** devuelve el equivalente en tiempo Unix del valor **xs:dateTime** que se indica en el argumento de la función. Tenga en cuenta que puede que deba generar de forma explícita el valor **xs:dateTime**. El valor **xs:dateTime** puede o no contener la parte opcional **TZ**, que indica la zona horaria.

Tanto si se indica la parte de la zona horaria como parte del argumento como si no, la diferencia que esta indica se obtiene del reloj del sistema y se resta al argumento **dateTimeValue** indicado. El resultado es el tiempo UTC a partir del cual se calcula el equivalente en tiempo Unix. Por ejemplo, si la función se ejecuta en un equipo cuya configuración sitúa en una zona horaria de +01:00 (con respecto a UTC), se resta una hora al valor **dateTimeValue** indicado antes de calcular el valor en tiempo Unix. Consulte también la función **dateTime-from-epoch**.

Ejemplos

La zona horaria local de los ejemplos siguientes es UTC +01:00. En consecuencia, se le restará una hora al valor **dateTime** indicado antes de calcular el tiempo Unix.

- **altova:epoch-from-dateTime**(xs:dateTime("1970-01-01T01:00:34+01:00")) devuelve 34
- **altova:epoch-from-dateTime**(xs:dateTime("1970-01-01T01:00:34")) devuelve 34
- **altova:epoch-from-dateTime**(xs:dateTime("2021-04-01T11:22:33")) devuelve 1617272553

[Subir 1496]

18.2.2.1.3 Funciones XPath/XQuery: de geoubicación

Las funciones de extensión XPath/XQuery de geoubicación son compatibles con la versión actual de MapForce y se pueden utilizar en (i) expresiones XPath en contextos XSLT o (ii) expresiones XQuery en documentos XQuery.

Nota sobre el nombre de las funciones y lenguajes

Puede utilizar todas las funciones de extensión de Altova en sus expresiones XPath/XQuery. Con ellas conseguirá funciones adicionales no disponibles en la biblioteca de funciones estándar de XPath, XQuery y XSLT. Las funciones de extensión de Altova están en el **espacio de nombres** **`http://www.altova.com/xslt-extensions`** y en esta sección se presentan con el prefijo, que se supone estará enlazado al espacio de nombres señalado. Tenga en cuenta que en futuras versiones del producto algunas funciones pueden dejar de ser compatibles o su comportamiento puede cambiar. Por tanto, consulte siempre la documentación del producto para conocer el funcionamiento de estas funciones en cada versión del producto.

Funciones XPath (en expresiones XPath en XSLT):	XP1 XP2 XP3.1.1
Funciones XSLT (en expresiones XPath en XSLT):	XSLT1 XSLT2 XSLT3
Funciones XQuery (en expresiones XQuery en XQuery):	XQ1 XQ3.1

format-geolocation [altova:]

`altova:format-geolocation(Latitude` *`como xs:decimal`*`, `**`Longitude`** *`como xs:decimal`*`, `**`GeolocationOutputStringFormat`** *`como xs:integer`*`)` como **`xs:string`** XP3.1 XQ3.1

Toma la latitud y la longitud como los dos primeros argumentos y da como resultado la geoubicación como cadena. El tercer argumento, **`GeolocationOutputStringFormat`**, es el formato de la cadena de resultado de la geoubicación: usa valores enteros del 1 al 4 para identificar el formato de la cadena de resultado (consulte más abajo "*Formatos de la cadena de resultado geoubicación"*). Los valores de latitud oscilan entre `+90` y `-90 (N a S)`. Los valores de longitud oscilan entre `+180` y `-180 (E a O)`.

Nota: la función `image-exif-data` y el atributo de metadatos Exif se pueden usar para suministrar las cadenas de entrada.

Ejemplos

- **`altova:format-geolocation`**`(33.33, -22.22, 4)` devuelve el `xs:string` `"33.33 -22.22"`
- **`altova:format-geolocation`**`(33.33, -22.22, 2)` devuelve el `xs:string` `"33.33N 22.22W"`
- **`altova:format-geolocation`**`(-33.33, 22.22, 2)` devuelve el `xs:string` `"33.33S 22.22E"`
- **`altova:format-geolocation`**`(33.33, -22.22, 1)` devuelve el `xs:string` `"33°19'48.00"S 22° 13'12.00"E"`

Formato de las cadenas de salida de las geoubicaciones:

A la latitud y longitud suministradas se les aplica un formato de salida de los que se indican más abajo. El formato deseado se identifica con un identificador comprendido entre `1` y `4`. Los valores de latitud pueden estar comprendidos entre `+90` y `-90` (`N` a `S`). Los valores de longitud pueden estar

 © 2016-2022 Altova GmbH

comprendidos entre `+180` y `-180` (`E` a `W`).

1
Grados, minutos y segundos decimales + orientación como sufijo (`N/S, E/W`) **`D°M'S.SS"N/S D°M'S.SS"E/W`** *Ejemplo*: **`33°55'11.11"N 22°44'66.66"W`**

2
Grados decimales + orientación como sufijo (`N/S`, `E/W`) **`D.DDN/S D.DDE/W`** *Ejemplo*: **`33.33N 22.22W`**

3
Grados, minutos y segundos decimales + prefijo (`+/-`). El signo + para (**`N/E`**) es opcional **`+/-D°M'S.SS" +/-D°M'S.SS"`** *Ejemplo*: **`33°55'11.11" -22°44'66.66"`**

4
Grados decimales + prefijo (`+/-`). El signo + para (**`N/E`**) es opcional **`+/-D.DD +/-D.DD`** *Ejemplo*: **`33.33 -22.22`**

Atributo Exif de Altova: Geolocation

El motor XPath/XQuery de Altova genera el atributo personalizado **`Geolocation`** a partir de las etiquetas de metadatos Exif estándar. Este atributo es una concatenación de cuatro etiquetas Exif (`GPSLatitude`, `GPSLatitudeRef`, `GPSLongitude`, `GPSLongitudeRef`) seguidas de unidades:

GPSLatitude	GPSLatitudeRef	GPSLongitude	GPSLongitudeRef	Geolocation
33 51 21.91	S	151 13 11.73	E	33°51'21.91"S 151° 13'11.73"E

parse-geolocation [altova:]

`altova:parse-geolocation(CadenaEntradaGeoubicación` *`como xs:string`*`)` como **`xs:decimal+`**
XP3.1 **XQ3.1**

Analiza el argumento `CadenaEntradaGeoubicación` y devuelve la latitud y la longitud (en ese orden) de la geoubicación en forma de secuencia de dos elementos `xs:decimal`. Más abajo puede ver en qué formatos se puede suministrar la cadena de entrada de la geoubicación.

Nota: la función `image-exif-data` 1525 y el atributo `@Geolocation` 1525 de los metadatos Exif se pueden utilizar para suministrar la cadena de entrada de la geoubicación (*ver ejemplos*).

Ejemplos

- **altova:parse-geolocation**("33.33 -22.22") devuelve la secuencia de dos xs:decimals (33.33, 22.22)
- **altova:parse-geolocation**("48°51'29.6""N 24°17'40.2""") devuelve la secuencia de dos xs:decimals (48.8582222222222, 24.2945)
- **altova:parse-geolocation**('48°51''29.6"N 24°17''40.2"') devuelve la secuencia de dos xs:decimals (48.8582222222222, 24.2945)
- **altova:parse-geolocation**(**image-exif-data**(//MisImágenes/Imagen20141130.01)/**@Geolocation**) devuelve una secuencia de dos xs:decimals

Formato de las cadenas de entrada de geoubicaciones:

La cadena de entrada de la geoubicación debe contener la latitud y la longitud (en ese orden) se paradas por un espacio en blanco. Ambas pueden estar en cualquier formato de los que se indican más abajo y puede combinar formatos distintos. Es decir, la latitud puede estar en un formato y la longitud en otro. Los valores de la latitud deben estar comprendidos entre +90 y -90 (N a S). Los valores de longitud deben estar comprendidos entre +180 y -180 (E a W).

Nota: si utiliza comillas simples o dobles para delimitar el argumento de la cadena de entrada, esto dará lugar a un conflicto con las comillas simples o dobles que se utilizan, respectivamente, para indicar los valores de los minutos y los segundos. Si esto ocurre, debe añadir caracteres de escape a las comillas utilizadas para los minutos y segundos (esto se hace duplicando las comillas.

- Grados, minutos y segundos decimales + orientación como sufijo (N/S, E/W)
  D°M'S.SS"N/S D°M'S.SS"W/E
  Ejemplo: **33°55'11.11"N 22°44'55.25"W**

- Grados, minutos y segundos decimales + prefijo (+/-). El signo + para (**N/E**) es opcional
  +/-D°M'S.SS" +/-D°M'S.SS"
  Ejemplo: **33°55'11.11" -22°44'55.25"**

- Grados y minutos decimales + orientación como sufijo (N/S, E/W)
  D°M.MM'N/S D°M.MM'W/E
  Ejemplo: **33°55.55'N 22°44.44'W**

- Grados y minutos decimales + prefijo (+/-). El signo + para (**N/E**) es opcional
  +/-D°M.MM' +/-D°M.MM'
  Ejemplo: **+33°55.55' -22°44.44'**

- Grados decimales + orientación como sufijo (N/S, E/W)
  D.DDN/S D.DDW/E
  Ejemplo: **33.33N 22.22W**

- Grados decimales + prefijo (+/-). El signo + para (**N/S, E/W**) es opcional
  +/-D.DD +/-D.DD
  Ejemplo: **33.33 -22.22**

Ejemplos de combinación de formatos:
33.33N -22°44'55.25"
33.33 22°44'55.25"W

© 2016-2022 Altova GmbH

33.33 22.45

Atributo Exif de Altova: Geolocation

El motor XPath/XQuery de Altova genera el atributo personalizado **Geolocation** a partir de las etiquetas de metadatos Exif estándar. Este atributo es una concatenación de cuatro etiquetas Exif (GPSLatitude, GPSLatitudeRef, GPSLongitude, GPSLongitudeRef) seguidas de unidades:

GPSLatitude	GPSLatitudeRef	GPSLongitude	GPSLongitudeRef	Geolocation
33 51 21.91	S	151 13 11.73	E	33°51'21.91"S 151° 13'11.73"E

geolocation-distance-km [altova:]

altova:geolocation-distance-km(CadenaEntradaGeoubicación-1 *como xs:string*, **CadenaEntradaGeoubicación-2** *como xs:string*) como **xs:decimal** XP3.1 XQ3.1

Calcula la distancia en km que existe entre dos geoubicaciones. El formato que puede utilizarse para dar las cadenas de entrada aparece más abajo. Los valores de latitud están comprendidos entre +90 y -90 (N a S). Los valores de longitud están comprendidos entre +180 y -180 (E a W).

Nota: la función image-exif-data 1525 y el atributo de metadatos Exif @Geolocation 1525 pueden utilizarse para suministrar las cadenas de entrada de geoubicaciones.

Ejemplos

- **altova:geolocation-distance-km**("33.33 -22.22", "48°51'29.6""N 24°17'40.2""") devuelve el xs:decimal 4183.08132372392

Formato de las cadenas de entrada de geoubicaciones:

La cadena de entrada de la geoubicación debe contener la latitud y la longitud (en ese orden) se paradas por un espacio en blanco. Ambas pueden estar en cualquier formato de los que se indican más abajo y puede combinar formatos distintos. Es decir, la latitud puede estar en un formato y la longitud en otro. Los valores de la latitud deben estar comprendidos entre +90 y -90 (N a S). Los valores de longitud deben estar comprendidos entre +180 y -180 (E a W).

Nota: si utiliza comillas simples o dobles para delimitar el argumento de la cadena de entrada, esto dará lugar a un conflicto con las comillas simples o dobles que se utilizan, respectivamente, para indicar los valores de los minutos y los segundos. Si esto ocurre, debe añadir caracteres de escape a las comillas utilizadas para los minutos y segundos (esto se hace duplicando las comillas.

- Grados, minutos y segundos decimales + orientación como sufijo (N/S, E/W)
  D°M'S.SS"N/S D°M'S.SS"W/E
  Ejemplo: **33°55'11.11"N 22°44'55.25"W**

- Grados, minutos y segundos decimales + prefijo (+/-). El signo + para (**N/E**) es opcional
  +/-D°M'S.SS" +/-D°M'S.SS"
  Ejemplo: **33°55'11.11" -22°44'55.25"**

- Grados y minutos decimales + orientación como sufijo (N/S, E/W)
  D°M.MM'N/S D°M.MM'W/E
  Ejemplo: **33°55.55'N 22°44.44'W**

- Grados y minutos decimales + prefijo (+/-). El signo + para (**N/E**) es opcional
  +/-D°M.MM' +/-D°M.MM'
  Ejemplo: **+33°55.55' -22°44.44'**

- Grados decimales + orientación como sufijo (N/S, E/W)
  D.DDN/S D.DDW/E
  Ejemplo: **33.33N 22.22W**

- Grados decimales + prefijo (+/-). El signo + para (**N/S, E/W**) es opcional
  +/-D.DD +/-D.DD
  Ejemplo: **33.33 -22.22**

Ejemplos de combinación de formatos:
33.33N -22°44'55.25"
33.33 22°44'55.25"W
33.33 22.45

Atributo Exif de Altova: Geolocation

El motor XPath/XQuery de Altova genera el atributo personalizado **Geolocation** a partir de las etiquetas de metadatos Exif estándar. Este atributo es una concatenación de cuatro etiquetas Exif (GPSLatitude, GPSLatitudeRef, GPSLongitude, GPSLongitudeRef) seguidas de unidades:

GPSLatitude	GPSLatitudeRef	GPSLongitude	GPSLongitudeRef	Geolocation
33 51 21.91	S	151 13 11.73	E	33°51'21.91"S 151° 13'11.73"E

geolocation-distance-mi [altova:]

altova:geolocation-distance-mi(CadenaEntradaGeoubicación-1 *como xs:string*, **CadenaEntradaGeoubicación-2** *como xs:string*) como **xs:decimal** XP3.1 XQ3.1
Calcula la distancia en millas que existe entre dos geoubicaciones. El formato que puede utilizarse para dar las cadenas de entrada aparece más abajo. Los valores de latitud están comprendidos entre +90 y -90 (N a S). Los valores de longitud están comprendidos entre +180 y -180 (E a W).

Nota: la función image-exif-data 1525 y el atributo de metadatos Exif @Geolocation 1525 pueden utilizarse para suministrar las cadenas de entrada de geoubicaciones.

Ejemplos

- **altova:geolocation-distance-mi**("33.33 -22.22", "48°51'29.6""N 24°17'40.2""")
  devuelve el xs:decimal 2599.40652340653

Formato de las cadenas de entrada de geoubicaciones:

© 2016-2022 Altova GmbH

La cadena de entrada de la geoubicación debe contener la latitud y la longitud (en ese orden) se paradas por un espacio en blanco. Ambas pueden estar en cualquier formato de los que se indican más abajo y puede combinar formatos distintos. Es decir, la latitud puede estar en un formato y la longitud en otro. Los valores de la latitud deben estar comprendidos entre `+90` y `-90` (`N` a `S`). Los valores de longitud deben estar comprendidos entre `+180` y `-180` (`E` a `W`).

Nota: si utiliza comillas simples o dobles para delimitar el argumento de la cadena de entrada, esto dará lugar a un conflicto con las comillas simples o dobles que se utilizan, respectivamente, para indicar los valores de los minutos y los segundos. Si esto ocurre, debe añadir caracteres de escape a las comillas utilizadas para los minutos y segundos (esto se hace duplicando las comillas.

- Grados, minutos y segundos decimales + orientación como sufijo (`N/S`, `E/W`)
  `D°M'S.SS"N/S  D°M'S.SS"W/E`
  Ejemplo: **`33°55'11.11"N  22°44'55.25"W`**

- Grados, minutos y segundos decimales + prefijo (`+/-`). El signo + para (**`N/E`**) es opcional
  `+/-D°M'S.SS"  +/-D°M'S.SS"`
  Ejemplo: **`33°55'11.11"  -22°44'55.25"`**

- Grados y minutos decimales + orientación como sufijo (`N/S`, `E/W`)
  `D°M.MM'N/S  D°M.MM'W/E`
  Ejemplo: **`33°55.55'N  22°44.44'W`**

- Grados y minutos decimales + prefijo (`+/-`). El signo + para (**`N/E`**) es opcional
  `+/-D°M.MM'  +/-D°M.MM'`
  Ejemplo: **`+33°55.55'  -22°44.44'`**

- Grados decimales + orientación como sufijo (`N/S`, `E/W`)
  `D.DDN/S  D.DDW/E`
  Ejemplo: **`33.33N  22.22W`**

- Grados decimales + prefijo (`+/-`). El signo + para (**`N/S, E/W`**) es opcional
  `+/-D.DD  +/-D.DD`
  Ejemplo: **`33.33  -22.22`**

Ejemplos de combinación de formatos:
`33.33N  -22°44'55.25"`
`33.33  22°44'55.25"W`
`33.33  22.45`

Atributo Exif de Altova: Geolocation

El motor XPath/XQuery de Altova genera el atributo personalizado **`Geolocation`** a partir de las etiquetas de metadatos Exif estándar. Este atributo es una concatenación de cuatro etiquetas Exif (`GPSLatitude`, `GPSLatitudeRef`, `GPSLongitude`, `GPSLongitudeRef`) seguidas de unidades:

GPSLatitude	GPSLatitudeRef	GPSLongitude	GPSLongitudeRef	Geolocation
33 51 21.91	S	151 13 11.73	E	33°51'21.91"S 151° 13'11.73"E

altova:geolocations-bounding-rectangle(Geolocations como *xs:sequence*, **GeolocationOutputStringFormat como** *xs:integer*) como **xs:string** XP3.1 XQ3.1
Toma una secuencia de cadenas de texto como primer argumento; cada cadena de esa secuencia es una geoubicación. La función devuelve una secuencia de dos cadenas que son, respectivamente, las coordinadas de geoubicación de la parte superior izquierda e inferior derecha de un rectángulo delimitado que tiene el tamaño exacto para contener las geoubicaciones suministradas en el primer argumento. Más abajo se enumeran los formatos en que se puede dar una cadena de entrada de geoubicación (*véase "Formato de las cadenas de entrada de geoubicaciones"*). Los valores de latitud están comprendidos entre `+90` y `-90` (`N` a `S`). Los valores de longitud están comprendidos entre `+180` y `-180` (`E` a `W`).

El segundo argumento de la función indica el formato de las dos cadenas de geoubicación de la secuencia de salida. El argumento toma un valor entero entre `1` y `4`, donde cada valor representa un formato distinto de las cadenas de entrada de geoubicaciones (*véase "Formato de las cadenas de salida de geoubicaciones"*).

Nota: la función `image-exif-data` 1525 y los atributos de metadatos Exif se pueden usar para suministrar las cadenas de entrada.

Ejemplos

- **altova:geolocations-bounding-rectangle**(("48.2143531 16.3707266", "51.50939 -0.11832"), 1) devuelve la secuencia ("51°30'33.804"N 0°7'5.952"W", "48°12'51.67116"N 16°22'14.61576"E")
- **altova:geolocations-bounding-rectangle**(("48.2143531 16.3707266", "51.50939 -0.11832", "42.5584577 -70.8893334"), 4) devuelve la secuencia ("51.50939 -70.8893334", "42.5584577 16.3707266")

Formato de las cadenas de entrada de geoubicaciones:

La cadena de entrada de la geoubicación debe contener la latitud y la longitud (en ese orden) se paradas por un espacio en blanco. Ambas pueden estar en cualquier formato de los que se indican más abajo y puede combinar formatos distintos. Es decir, la latitud puede estar en un formato y la longitud en otro. Los valores de la latitud deben estar comprendidos entre `+90` y `-90` (`N` a `S`). Los valores de longitud deben estar comprendidos entre `+180` y `-180` (`E` a `W`).

Nota: si utiliza comillas simples o dobles para delimitar el argumento de la cadena de entrada, esto dará lugar a un conflicto con las comillas simples o dobles que se utilizan, respectivamente, para indicar los valores de los minutos y los segundos. Si esto ocurre, debe añadir caracteres de escape a las comillas utilizadas para los minutos y segundos (esto se hace duplicando las comillas.

- Grados, minutos y segundos decimales + orientación como sufijo (`N/S`, `E/W`)
  D°M'S.SS"N/S D°M'S.SS"W/E
  Ejemplo: **33°55'11.11"N 22°44'55.25"W**

- Grados, minutos y segundos decimales + prefijo (`+/-`). El signo + para (**N/E**) es opcional
  +/-D°M'S.SS" +/-D°M'S.SS"
  Ejemplo: **33°55'11.11" -22°44'55.25"**

- Grados y minutos decimales + orientación como sufijo (`N/S`, `E/W`)
  D°M.MM'N/S D°M.MM'W/E
  Ejemplo: **33°55.55'N 22°44.44'W**

© 2016-2022 Altova GmbH

- Grados y minutos decimales + prefijo (+/-). El signo + para (**N/E**) es opcional
  +/-D°M.MM' +/-D°M.MM'
  Ejemplo: **+33°55.55' -22°44.44'**

- Grados decimales + orientación como sufijo (N/S, E/W)
  D.DDN/S D.DDW/E
  Ejemplo: **33.33N 22.22W**

- Grados decimales + prefijo (+/-). El signo + para (**N/S, E/W**) es opcional
  +/-D.DD +/-D.DD
  Ejemplo: **33.33 -22.22**

Ejemplos de combinación de formatos:
33.33N -22°44'55.25"
33.33 22°44'55.25"W
33.33 22.45

Formato de las cadenas de salida de las geoubicaciones:

A la latitud y longitud suministradas se les aplica un formato de salida de los que se indican más abajo. El formato deseado se identifica con un identificador comprendido entre 1 y 4. Los valores de latitud pueden estar comprendidos entre +90 y -90 (N a S). Los valores de longitud pueden estar comprendidos entre +180 y -180 (E a W).

1
Grados, minutos y segundos decimales + orientación como sufijo (N/S, E/W) **D°M'S.SS"N/S D°M'S.SS"E/W** *Ejemplo*: **33°55'11.11"N 22°44'66.66"W**

2
Grados decimales + orientación como sufijo (N/S, E/W) **D.DDN/S D.DDE/W** *Ejemplo*: **33.33N 22.22W**

3
Grados, minutos y segundos decimales + prefijo (+/-). El signo + para (**N/E**) es opcional **+/-D°M'S.SS" +/-D°M'S.SS"** *Ejemplo*: **33°55'11.11" -22°44'66.66"**

4
Grados decimales + prefijo (+/-). El signo + para (**N/E**) es opcional **+/-D.DD +/-D.DD** *Ejemplo*: **33.33 -22.22**

Atributo Exif de Altova: Geolocation

El motor XPath/XQuery de Altova genera el atributo personalizado **Geolocation** a partir de las etiquetas de metadatos Exif estándar. Este atributo es una concatenación de cuatro etiquetas Exif (GPSLatitude, GPSLatitudeRef, GPSLongitude, GPSLongitudeRef) seguidas de unidades:

GPSLatitude	GPSLatitudeRef	GPSLongitude	GPSLongitudeRef	Geolocation
33 51 21.91	S	151 13 11.73	E	33°51'21.91"S 151° 13'11.73"E

geolocation-within-polygon [altova:]

altova:geolocation-within-polygon(Geoubicación *como xs:string*, **((PuntoDePolígono** *como xs:string*)+)) como xs:boolean XP3.1 XQ3.1

Determina si **Geoubicación** (primer argumento) está dentro del área poligonal descrita por los argumentos **PuntoDePolígono**. Si los argumentos PuntoDePolígono no forman una figura cerrada (la figura se cierra cuando el primer y el último punto son el mismo), entonces el primer punto se añade implícitamente como último punto a fin de cerrar la figura. Todos los argumentos (Geoubicación y PuntoDePolígono+) se dan como cadenas de entrada de geoubicación (*formatos permitidos más abajo*). Si el argumento Geoubicación está dentro del área poligonal, entonces la función devuelve true(). De lo contrario, devuelve false(). Los valores de latitud están comprendidos entre +90 y -90 (N a S). Los valores de longitud están comprendidos entre +180 y -180 (E a W).

Nota: la función image-exif-data 1525 y el atributo de metadatos Exif @Geolocation 1525 pueden utilizarse para suministrar las cadenas de entrada de geoubicaciones.

Ejemplos

- **altova:geolocation-within-polygon(**"33 -22", ("58 -32", "-78 -55", "48 24", "58 -32")**)** devuelve true()
- **altova:geolocation-within-polygon(**"33 -22", ("58 -32", "-78 -55", "48 24")**)** devuelve true()
- **altova:geolocation-within-polygon(**"33 -22", ("58 -32", "-78 -55", "48°51'29.6""N 24°17'40.2""")**)** devuelve true()

Formato de las cadenas de entrada de geoubicaciones:

La cadena de entrada de la geoubicación debe contener la latitud y la longitud (en ese orden) se paradas por un espacio en blanco. Ambas pueden estar en cualquier formato de los que se indican más abajo y puede combinar formatos distintos. Es decir, la latitud puede estar en un formato y la longitud en otro. Los valores de la latitud deben estar comprendidos entre +90 y -90 (N a S). Los valores de longitud deben estar comprendidos entre +180 y -180 (E a W).

Nota: si utiliza comillas simples o dobles para delimitar el argumento de la cadena de entrada, esto dará lugar a un conflicto con las comillas simples o dobles que se utilizan, respectivamente, para indicar los valores de los minutos y los segundos. Si esto ocurre, debe añadir caracteres de escape a las comillas utilizadas para los minutos y segundos (esto se hace duplicando las comillas.

- Grados, minutos y segundos decimales + orientación como sufijo (N/S, E/W)

© 2016-2022 Altova GmbH

D°M'S.SS"N/S D°M'S.SS"W/E
Ejemplo: **33°55'11.11"N 22°44'55.25"W**

- Grados, minutos y segundos decimales + prefijo (+/-). El signo + para (**N/E**) es opcional
  +/-D°M'S.SS" +/-D°M'S.SS"
  Ejemplo: **33°55'11.11" -22°44'55.25"**

- Grados y minutos decimales + orientación como sufijo (N/S, E/W)
  D°M.MM'N/S D°M.MM'W/E
  Ejemplo: **33°55.55'N 22°44.44'W**

- Grados y minutos decimales + prefijo (+/-). El signo + para (**N/E**) es opcional
  +/-D°M.MM' +/-D°M.MM'
  Ejemplo: **+33°55.55' -22°44.44'**

- Grados decimales + orientación como sufijo (N/S, E/W)
  D.DDN/S D.DDW/E
  Ejemplo: **33.33N 22.22W**

- Grados decimales + prefijo (+/-). El signo + para (**N/S, E/W**) es opcional
  +/-D.DD +/-D.DD
  Ejemplo: **33.33 -22.22**

Ejemplos de combinación de formatos:
33.33N -22°44'55.25"
33.33 22°44'55.25"W
33.33 22.45

Atributo Exif de Altova: Geolocation

El motor XPath/XQuery de Altova genera el atributo personalizado **Geolocation** a partir de las etiquetas de metadatos Exif estándar. Este atributo es una concatenación de cuatro etiquetas Exif (GPSLatitude, GPSLatitudeRef, GPSLongitude, GPSLongitudeRef) seguidas de unidades:

GPSLatitude	GPSLatitudeRef	GPSLongitude	GPSLongitudeRef	Geolocation
33 51 21.91	S	151 13 11.73	E	33°51'21.91"S 151° 13'11.73"E

geolocation-within-rectangle [altova:]

altova:geolocation-within-rectangle(Geoubicación *como xs:string*, **ÁnguloRectángulo-1** *como xs:string*, **ÁnguloRectángulo-2** *como xs:string***)** como **xs:boolean** XP3.1 XQ3.1
Determina si **Geoubicación** (primer argumento) está dentro del rectángulo definido por el segundo y el tercer argumento (**ÁnguloRectángulo-1** y **ÁnguloRectángulo-2**), que indican ángulos opuestos del rectángulo. Todos los argumentos de la función se dan como cadenas de entrada de geoubicación (*formatos permitidos más abajo*). Si el argumento Geoubicación está dentro del rectángulo, entonces la función devuelve true(). De lo contrario, devuelve false(). Los valores de latitud están comprendidos entre +90 y -90 (N a S). Los valores de longitud están comprendidos entre +180 y -180 (E a W).

Nota: la función `image-exif-data` [1525] y el atributo de metadatos Exif `@Geolocation` [1525] pueden utilizarse para suministrar las cadenas de entrada de geoubicaciones.

Ejemplos

- `altova:geolocation-within-rectangle("33 -22", "58 -32", "-48 24")` devuelve `true()`
- `altova:geolocation-within-rectangle("33 -22", "58 -32", "48 24")` devuelve `false()`
- `altova:geolocation-within-rectangle("33 -22", "58 -32", "48°51'29.6""S  24° 17'40.2""")` devuelve `true()`

Formato de las cadenas de entrada de geoubicaciones:

La cadena de entrada de la geoubicación debe contener la latitud y la longitud (en ese orden) se paradas por un espacio en blanco. Ambas pueden estar en cualquier formato de los que se indican más abajo y puede combinar formatos distintos. Es decir, la latitud puede estar en un formato y la longitud en otro. Los valores de la latitud deben estar comprendidos entre `+90` y `-90` (`N` a `S`). Los valores de longitud deben estar comprendidos entre `+180` y `-180` (`E` a `W`).

Nota: si utiliza comillas simples o dobles para delimitar el argumento de la cadena de entrada, esto dará lugar a un conflicto con las comillas simples o dobles que se utilizan, respectivamente, para indicar los valores de los minutos y los segundos. Si esto ocurre, debe añadir caracteres de escape a las comillas utilizadas para los minutos y segundos (esto se hace duplicando las comillas.

- Grados, minutos y segundos decimales + orientación como sufijo (`N/S`, `E/W`)
  `D°M'S.SS"N/S  D°M'S.SS"W/E`
  Ejemplo: **`33°55'11.11"N  22°44'55.25"W`**

- Grados, minutos y segundos decimales + prefijo (`+/-`). El signo + para (**`N/E`**) es opcional
  `+/-D°M'S.SS"  +/-D°M'S.SS"`
  Ejemplo: **`33°55'11.11"  -22°44'55.25"`**

- Grados y minutos decimales + orientación como sufijo (`N/S`, `E/W`)
  `D°M.MM'N/S  D°M.MM'W/E`
  Ejemplo: **`33°55.55'N  22°44.44'W`**

- Grados y minutos decimales + prefijo (`+/-`). El signo + para (**`N/E`**) es opcional
  `+/-D°M.MM'  +/-D°M.MM'`
  Ejemplo: **`+33°55.55'  -22°44.44'`**

- Grados decimales + orientación como sufijo (`N/S`, `E/W`)
  `D.DDN/S  D.DDW/E`
  Ejemplo: **`33.33N  22.22W`**

- Grados decimales + prefijo (`+/-`). El signo + para (**`N/S, E/W`**) es opcional
  `+/-D.DD  +/-D.DD`
  Ejemplo: **`33.33  -22.22`**

Ejemplos de combinación de formatos:

`33.33N  -22°44'55.25"`
`33.33  22°44'55.25"W`
`33.33  22.45`

© 2016-2022 Altova GmbH

Atributo Exif de Altova: Geolocation

El motor XPath/XQuery de Altova genera el atributo personalizado **Geolocation** a partir de las etiquetas de metadatos Exif estándar. Este atributo es una concatenación de cuatro etiquetas Exif (GPSLatitude, GPSLatitudeRef, GPSLongitude, GPSLongitudeRef) seguidas de unidades:

GPSLatitude	GPSLatitudeRef	GPSLongitude	GPSLongitudeRef	Geolocation
33 51 21.91	S	151 13 11.73	E	33°51'21.91"S 151° 13'11.73"E

[Subir 1514]

18.2.2.1.4 Funciones XPath/XQuery: relacionadas con imágenes

Las funciones de extensión XPath/XQuery para trabajar con imágenes son compatibles con la versión actual de MapForce y se pueden utilizar en (i) expresiones XPath en contextos XSLT o (ii) expresiones XQuery en documentos XQuery.

Nota sobre el nombre de las funciones y lenguajes

Puede utilizar todas las funciones de extensión de Altova en sus expresiones XPath/XQuery. Con ellas conseguirá funciones adicionales no disponibles en la biblioteca de funciones estándar de XPath, XQuery y XSLT. Las funciones de extensión de Altova están en el **espacio de nombres** **http://www.altova.com/xslt-extensions** y en esta sección se presentan con el prefijo, que se supone estará enlazado al espacio de nombres señalado. Tenga en cuenta que en futuras versiones del producto algunas funciones pueden dejar de ser compatibles o su comportamiento puede cambiar. Por tanto, consulte siempre la documentación del producto para conocer el funcionamiento de estas funciones en cada versión del producto.

Funciones XPath (en expresiones XPath en XSLT):	XP1 XP2 XP3.1.1
Funciones XSLT (en expresiones XPath en XSLT):	XSLT1 XSLT2 XSLT3
Funciones XQuery (en expresiones XQuery en XQuery):	XQ1 XQ3.1

suggested-image-file-extension [altova:]

altova:suggested-image-file-extension(CadenaBase64 *como string*) como **string?** XP3.1 XQ3.1

Toma la codificación base64 de un archivo de imagen como argumento y devuelve la extensión de archivo de la imagen registrada en la codificación base64 de la imagen. El valor devuelto es una sugerencia basada en la información sobre el tipo de imagen disponible en la codificación. Si esta información no está disponible, entonces devuelve una cadena vacía. Esta función es muy práctica a la hora de guardar una imagen base64 como archivo y recuperar de forma dinámica una extensión de archivo adecuada.

Ejemplos

- **altova:suggested-image-file-extension**(/MisImágenes/TeléfonoMóvil/Imagen20141130.01) devuelve 'jpg'
- **altova:suggested-image-file-extension**($XML1/Personal/Persona/@photo) devuelve ''

En los ejemplos anteriores, se da por hecho que los nodos suministrados como argumento de la función contienen una imagen codificada en base64. El primer ejemplo recupera jpg como tipo de imagen y como extensión de archivo. En el segundo ejemplo, la codificación base64 dada no ofrece información sobre la extensión del archivo.

image-exif-data [altova:]

altova:image-exif-data(CadenaBinariaBase64 *como string***)** como **element?** XP3.1 XQ3.1

Toma una imagen JPEG codificada en base64 como argumento y devuelve un elemento llamado **Exif** que contiene los metadatos Exif de la imagen. Los metadatos Exif se crean como pares atributo-valor del elemento Exif. El nombre de los atributos son las etiquetas de datos Exif encontradas en la codificación base64. La lista de etiquetas Exif aparece más abajo. Si en lo datos Exif hay etiquetas de terceros, estas etiquetas y sus valores también se devuelven en un par atributo-valor. Además de las etiquetas de metadatos Exif estándar (*lista más abajo*), también se generan pares atributo-valor de Altova. Estos atributos Exif de Altova también se enumeran más abajo.

Ejemplos

- Para acceder a un atributo, utilice la función de esta manera:
  image-exif-data(//MisImágenes/Imagen20141130.01)**/@GPSLatitude**
  image-exif-data(//MisImágenes/Imagen20141130.01)**/@Geolocation**
- Para acceder a todos los atributos, utilice la función de esta manera:
  image-exif-data(//MisImágenes/Imagen20141130.01)**/@***
- Para acceder al nombre de todos los atributos, utilice esta expresión:
  for $i in image-exif-data**(//MisImágenes/Imagen20141130.01)**/@* return name($i)
  Esto es muy práctico a la hora de averiguar el nombre de los atributos que devuelve la función.

Atributo Exif de Altova: Geolocation

El motor XPath/XQuery de Altova genera el atributo personalizado **Geolocation** a partir de las etiquetas de metadatos Exif estándar. Este atributo es una concatenación de cuatro etiquetas Exif (GPSLatitude, GPSLatitudeRef, GPSLongitude, GPSLongitudeRef) seguidas de unidades:

GPSLatitude	GPSLatitudeRef	GPSLongitude	GPSLongitudeRef	Geolocation
33 51 21.91	S	151 13 11.73	E	33°51'21.91"S 151° 13'11.73"E

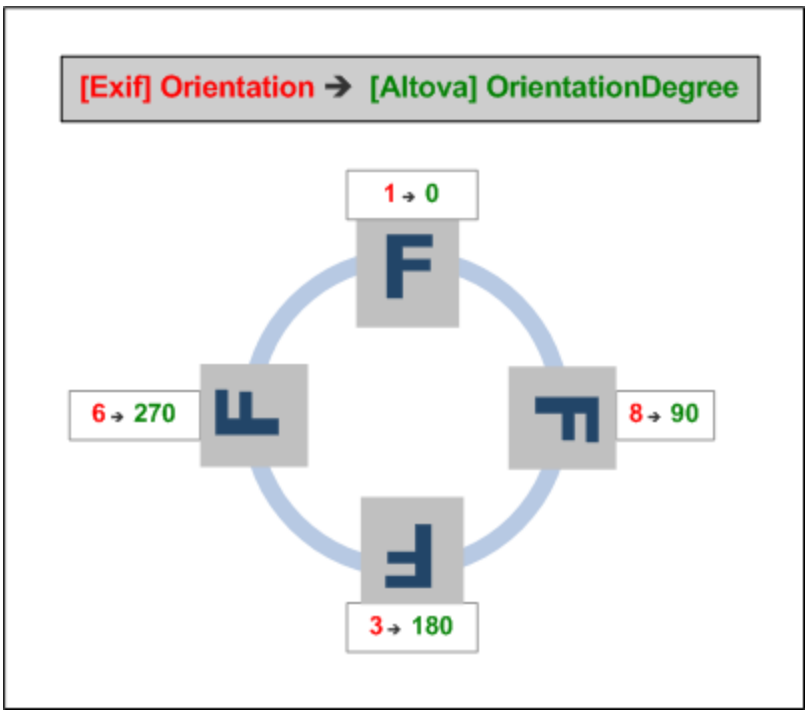
Atributo Exif de Altova: OrientationDegree

El motor XPath/XQuery de Altova genera el atributo personalizado **OrientationDegree** a partir de la etiqueta de metadatos Exif **Orientation**.

© 2016-2022 Altova GmbH

Este atributo transforma el valor entero de la etiqueta Exif `Orientation` (`1`, `8`, `3` o `6`) en el correspondiente valor en grados (`0`, `90`, `180`, `270`), tal y como describe el diagrama más abajo.

Debe tener en cuenta que los valores `2`, `4`, `5`, `7` de `Orientation` no se pueden traducir. Estas orientaciones se obtienen invirtiendo la imagen `1` en su eje central vertical para obtener la imagen con un valor de `2` e invirtiendo después esta imagen por pasos de 90 grados en el sentido de las agujas del reloj para obtener los valores de `7`, `4` y `5`, respectivamente.



Lista de etiquetas Exif estándar

- `ImageWidth`
- `ImageLength`
- `BitsPerSample`
- `Compression`
- `PhotometricInterpretation`
- `Orientation`
- `SamplesPerPixel`
- `PlanarConfiguration`
- `YCbCrSubSampling`
- `YCbCrPositioning`
- `XResolution`
- `YResolution`
- `ResolutionUnit`
- `StripOffsets`
- `RowsPerStrip`
- `StripByteCounts`
- `JPEGInterchangeFormat`
- `JPEGInterchangeFormatLength`

- TransferFunction
- WhitePoint
- PrimaryChromaticities
- YCbCrCoefficients
- ReferenceBlackWhite
- DateTime
- ImageDescription
- Make
- Model
- Software
- Artist
- Copyright

- ExifVersion
- FlashpixVersion
- ColorSpace
- ComponentsConfiguration
- CompressedBitsPerPixel
- PixelXDimension
- PixelYDimension
- MakerNote
- UserComment
- RelatedSoundFile
- DateTimeOriginal
- DateTimeDigitized
- SubSecTime
- SubSecTimeOriginal
- SubSecTimeDigitized
- ExposureTime
- FNumber
- ExposureProgram
- SpectralSensitivity
- ISOSpeedRatings
- OECF
- ShutterSpeedValue
- ApertureValue
- BrightnessValue
- ExposureBiasValue
- MaxApertureValue
- SubjectDistance
- MeteringMode
- LightSource
- Flash
- FocalLength
- SubjectArea
- FlashEnergy
- SpatialFrequencyResponse
- FocalPlaneXResolution
- FocalPlaneYResolution
- FocalPlaneResolutionUnit
- SubjectLocation
- ExposureIndex
- SensingMethod
- FileSource

© 2016-2022 Altova GmbH

- SceneType
- CFAPattern
- CustomRendered
- ExposureMode
- WhiteBalance
- DigitalZoomRatio
- FocalLengthIn35mmFilm
- SceneCaptureType
- GainControl
- Contrast
- Saturation
- Sharpness
- DeviceSettingDescription
- SubjectDistanceRange
- ImageUniqueID

- GPSVersionID
- GPSLatitudeRef
- GPSLatitude
- GPSLongitudeRef
- GPSLongitude
- GPSAltitudeRef
- GPSAltitude
- GPSTimeStamp
- GPSSatellites
- GPSStatus
- GPSMeasureMode
- GPSDOP
- GPSSpeedRef
- GPSSpeed
- GPSTrackRef
- GPSTrack
- GPSImgDirectionRef
- GPSImgDirection
- GPSMapDatum
- GPSDestLatitudeRef
- GPSDestLatitude
- GPSDestLongitudeRef
- GPSDestLongitude
- GPSDestBearingRef
- GPSDestBearing
- GPSDestDistanceRef
- GPSDestDistance
- GPSProcessingMethod
- GPSAreaInformation
- GPSDateStamp
- GPSDifferential

[Subir 1525]

18.2.2.1.5 Funciones XPath/XQuery: numéricas

Las funciones de extensión numéricas de Altova pueden utilizarse en expresiones XPath y XQuery y ofrecen funciones adicionales para el procesamiento de datos. Estas funciones se pueden usar con los motores **XPath 3.0** y **XQuery 3.0** de Altova. Están disponibles en contextos XPath/XQuery.

Nota sobre el nombre de las funciones y lenguajes

Puede utilizar todas las funciones de extensión de Altova en sus expresiones XPath/XQuery. Con ellas conseguirá funciones adicionales no disponibles en la biblioteca de funciones estándar de XPath, XQuery y XSLT. Las funciones de extensión de Altova están en el **espacio de nombres** **`http://www.altova.com/xslt-extensions`** y en esta sección se presentan con el prefijo, que se supone estará enlazado al espacio de nombres señalado. Tenga en cuenta que en futuras versiones del producto algunas funciones pueden dejar de ser compatibles o su comportamiento puede cambiar. Por tanto, consulte siempre la documentación del producto para conocer el funcionamiento de estas funciones en cada versión del producto.

Funciones XPath (en expresiones XPath en XSLT):	XP1 XP2 XP3.1.1
Funciones XSLT (en expresiones XPath en XSLT):	XSLT1 XSLT2 XSLT3
Funciones XQuery (en expresiones XQuery en XQuery):	XQ1 XQ3.1

Funciones de numeración automática

generate-auto-number [altova:]

altova:generate-auto-number(ID *como xs:string*, **EmpiezaPor** *como xs:double*, **Incremento** *como xs:double*, **RestaurarAlCambiar** *como xs:string*) como `xs:integer` XP1 XP2 XQ1 XP3.1 XQ3.1

Genera un número cada vez que se llama a la función. El primer número, que se genera cuando se llama a la función por primera vez, viene dado por el argumento `EmpiezaPor`. Cada llamada posterior genera un número nuevo, que se incrementa en función del valor especificado en el argumento `Incremento`. De hecho, la función `generate-auto-number` crea un contador llamado como indique el argumento `ID` y este contador se incrementa cada vez que se llama a la función. Si el valor del argumento `RestaurarAlCambiar` cambia con respecto al valor que tenía en la llamada anterior, entonces el valor del número que se debe generar se restablece con el valor de `EmpiezaPor`. También puede restablecer la numeración automática con la función `altova:reset-auto-number`.

Ejemplo

- **`altova:generate-auto-number`**`("ChapterNumber", 1, 1, "SomeString")`
  Devuelve un número cada vez que se llama a la función, empezando por `1` y con un incremento de `1` con cada llamada a función. Si el cuarto argumento continúa siendo `"SomeString"` en las llamadas posteriores, el incremento continuará. Cuando cambie el valor del cuarto argumento, se restaura el valor 1 del contador (llamado `ChapterNumber`). El valor de `ChapterNumber` también se puede restaurar llamando a la función `altova:reset-auto-number("ChapterNumber")`.

© 2016-2022 Altova GmbH

reset-auto-number [altova:]

altova:reset-auto-number(ID *como xs:string***)** XP1 XP2 XQ1 XP3.1 XQ3.1
Esta función restaura el número del contador de numeración automática especificado en el argumento `ID`. El número se reemplaza con el número indicado en el argumento `EmpiezaPor` de la función `altova:generate-auto-number` que creó el contador especificado en el argumento `ID`.

Ejemplos

- **altova:reset-auto-number**("ChapterNumber") restablece el número del contador de numeración automática llamado `ChapterNumber` que se creó con la función `altova:generate-auto-number`. El número se reemplaza con el valor del argumento `EmpiezaPor` de la función `altova:generate-auto-number` que creó `ChapterNumber`.

[Subir 1530]

Funciones numéricas

hex-string-to-integer [altova:]

altova:hex-string-to-integer(CadenaHex *as xs:string***)** como **xs:integer** XP3.1 XQ3.1
Toma un argumento de cadena que es el equivalente Base-16 de un entero del sistema decimal (Base-10) y devuelve un entero decimal.

Ejemplos

- **altova:hex-string-to-integer**('1') devuelve 1
- **altova:hex-string-to-integer**('9') devuelve 9
- **altova:hex-string-to-integer**('A') devuelve 10
- **altova:hex-string-to-integer**('B') devuelve 11
- **altova:hex-string-to-integer**('F') devuelve 15
- **altova:hex-string-to-integer**('G') devuelve un error
- **altova:hex-string-to-integer**('10') devuelve 16
- **altova:hex-string-to-integer**('01') devuelve 1
- **altova:hex-string-to-integer**('20') devuelve 32
- **altova:hex-string-to-integer**('21') devuelve 33
- **altova:hex-string-to-integer**('5A') devuelve 90
- **altova:hex-string-to-integer**('USA') devuelve un error

integer-to-hex-string [altova:]

altova:integer-to-hex-string(Entero *as xs:integer***)** como **xs:string** XP3.1 XQ3.1
Toma el argumento **Entero** y devuelve su equivalente Base-16 en forma de cadena.

Ejemplos

- **altova:integer-to-hex-string**(1) devuelve '1'
- **altova:integer-to-hex-string**(9) devuelve '9'
- **altova:integer-to-hex-string**(10) devuelve 'A'
- **altova:integer-to-hex-string**(11) devuelve 'B'
- **altova:integer-to-hex-string**(15) devuelve 'F'
- **altova:integer-to-hex-string**(16) devuelve '10'
- **altova:integer-to-hex-string**(32) devuelve '20'

- `altova:integer-to-hex-string(33)` devuelve `'21'`
- `altova:integer-to-hex-string(90)` devuelve `'5A'`

[Subir 1530]

Funciones de formato numérico

[Subir 1530]

18.2.2.1.6 Funciones XPath/XQuery: de esquema

Las funciones de extensión de Altova que enumeramos a continuación devuelven información del esquema. Más adelante verá descripciones de las funciones, junto con (i) ejemplos y (ii) una lista de los componentes del esquema y sus correspondientes propiedades. Estas funciones se pueden usar con los motores de Altova **XPath 3.0** y **XQuery 3.0**, y están disponibles en contextos XPath/XQuery.

Información sobre el esquema proveniente de documentos de esquema
La función **`altova:schema`** tiene dos argumentos: uno que no tiene argumentos y otro que tiene dos. La función que no tiene argumentos devuelve todo el esquema. A partir de ahí puede navegar por el esquema para encontrar los componentes que necesite. La función con dos argumentos devuelve un tipo concreto de componente al que se identifica por su QName. En ambos casos el valor de retorno es una función. Para ir al componente devuelto debe seleccionar una de sus propiedades. Si esta propiedad es un elemento no atómico (es decir, si es un componente), entonces puede seleccionar también una propiedad de este componente para seguir navegando. Si la propiedad seleccionada sí es un elemento atómico, entonces se devuelve el valor del elemento y no puede seguir navegando.

Nota: en las expresiones XPath de debe importar primero el esquema en el entorno de procesamiento (por ejemplo, XSLT), con la instrucción **`xslt:import-schema`**. En las expresiones XQuery, el esquema se debe importar de forma explícita.

Información sobre el esquema proveniente de nodos XML
La función **`altova:type`** envía el nodo de un documento XML y devuelve la información del tipo del modo desde el PSVI (Conjunto de información posterior a la validación de esquemas).

Nota sobre el nombre de las funciones y lenguajes

Puede utilizar todas las funciones de extensión de Altova en sus expresiones XPath/XQuery. Con ellas conseguirá funciones adicionales no disponibles en la biblioteca de funciones estándar de XPath, XQuery y XSLT. Las funciones de extensión de Altova están en el **espacio de nombres** **`http://www.altova.com/xslt-extensions`** y en esta sección se presentan con el prefijo, que se supone estará enlazado al espacio de nombres señalado. Tenga en cuenta que en futuras versiones del producto algunas funciones pueden dejar de ser compatibles o su comportamiento puede cambiar. Por tanto, consulte siempre la documentación del producto para conocer el funcionamiento de estas funciones en cada versión del producto.

Funciones XPath (en expresiones XPath en XSLT):	XP1 XP2 XP3.1.1

 © 2016-2022 Altova GmbH

Funciones XSLT (en expresiones XPath en XSLT):	XSLT1 XSLT2 XSLT3
Funciones XQuery (en expresiones XQuery en XQuery):	XQ1 XQ3.1

altova:schema() como (function(xs:string) como item()*)? XP3.1 XQ3.1
Devuelve el componente **schema** al completo. Para navegar por este componente seleccione una de sus propiedades.

- Si esta propiedad es un componente seleccione una de sus propiedades para navegar hasta el siguiente nivel de profundidad. Puede repetir este paso para seguir navegando por el esquema.
- Si el componente es un valor atómico se devuelve este valor y no puede seguir navegando.

Las propiedades del componente **schema** son:

```
"type definitions"
"attribute declarations"
"element declarations"
"attribute group definitions"
"model group definitions"
"notation declarations"
"identity-constraint definitions"
```

Más abajo encontrará las propiedades del resto de tipos de componente.

Nota: en las expresiones XQuery, el esquema se debe importar de forma explícita. En las expresiones XPath debe importar primero el esquema en el entorno de procesamiento, por ejemplo en XSLT con la instrucción **xslt:import**.

Ejemplos

- `import schema "" at "C:\Test\ExpReport.xsd"; for $typedef in altova:schema() ("type definitions")`
  `return $typedef ("name")` devuelve los nombres de todos los tipos simples o complejos del esquema

- `import schema "" at "C:\Test\ExpReport.xsd";`
  `altova:schema() ("type definitions")[1]("name")` devuelve el nombre del primero de los tipos simples o complejos del esquema

Componentes y sus propiedades

Assertion

Nombre de la propiedad	Tipo de la propiedad	Valor de la propiedad
kind	Cadena	"Assertion"
test	Registro de propiedades XPath	

Attribute Declaration

Nombre de la propiedad	Tipo de la propiedad	Valor de la propiedad
kind	Cadena	"Attribute Declaration"
name	Cadena	Nombre local del atributo
target namespace	Cadena	URI del espacio de nombres del atributo
type definition	Simple Type o Complex Type	
scope	Una función con propiedades ("class":"Scope", "variety": "global" o "local", "parent": el Complex Type o Attribute Group contenedor)	
value constraint	Si está presente, una función con propiedades ("class": "Value Constraint", "variety": "fixed" o "default", "value": atomic value, "lexical form": string. Tenga en cuenta que la propiedad "value" no está disponible para los tipos namespace-sensitive	
inheritable	Booleano	

Attribute Group Declaration

Nombre de la propiedad	Tipo de la propiedad	Valor de la propiedad
kind	Cadena	"Attribute Group Definition"
name	Cadena	Nombre local del grupo de atributos
target namespace	Cadena	URI del espacio de nombres del grupo de atributos
attribute uses	Secuencia de (Attribute Use)	
attribute wildcard	Comodín de atributo opcional	

Attribute Use

Nombre de la propiedad	Tipo de la propiedad	Valor de la propiedad
kind	Cadena	"Attribute Use"
required	Booleano	true si el atributo es obligatorio, false si es opcional
value constraint	Véase la declaración de atributos	
inheritable	Booleano	

© 2016-2022 Altova GmbH

Attribute Wildcard

Nombre de la propiedad	Tipo de la propiedad	Valor de la propiedad
kind	Cadena	"Wildcard"
namespace constraint	Función con propiedades ("class": "Namespace Constraint", "variety": "any"|"enumeration"|"not", "namespaces": secuencia de elementos xs:anyURI, "disallowed names": lista que contiene QNames y/o las cadenas "defined" y "definedSiblings"	
process contents	Cadena ("strict"|"lax"|"skip")	

Complex Type

Nombre de la propiedad	Tipo de la propiedad	Valor de la propiedad
kind	Cadena	"Complex Type"
name	Cadena	Nombre local del tipo (vacío si es anónimo)
target namespace	Cadena	URI del espacio de nombres del tipo (vacío si es anónimo)
base type definition	Definición del Complex Type	
final	Secuencia de cadenas ("restriction"|"extension")	
context	Secuencia vacía (not implemented)	
derivation method	Cadena ("restriction"|"extension")	
abstract	Booleano	
attribute uses	Secuencia de elementos Attribute Use	
attribute wildcard	Comodín de atributo opcional	
content type	Función con propiedades: ("class":"Content Type", "variety":string ("element-only"|"empty"|"mixed"|"simple"), particle: partícula opcional, "open content": función con propiedades ("class":"Open Content", "mode": string ("interleave"|"suffix"), "wildcard": Wildcard), "simple type definition": Simple Type)	
prohibited substitutions	Secuencia de cadenas ("restriction"|"extension")	
assertions	Secuencia de elementos Assertion	

Element Declaration

Nombre de la propiedad	Tipo de la propiedad	Valor de la propiedad
kind	Cadena	"Complex Type"
name	Cadena	Nombre local del tipo (vacío si es anónimo)
target namespace	Cadena	Namespace URI del tipo (vacío si es anónimo)
type definition	Simple Type o Complex Type	
type table	Función con propiedades ("class":"Type Table", "alternatives": secuencia de elementos Type Alternative, "default type definition": Simple Type o Complex Type)	
scope	Función con propiedades ("class":"Scope", "variety": ("global"|"local"), "parent": Complex Type opcional)	
value constraint	véase Attribute Declaration	
nillable	Booleano	
identity-constraint definitions	Secuencia de restricciones de identidad	
substitution group affiliations	Secuencia de declaraciones de elementos	
substitution group exclusions	Secuencia de cadenas ("restriction"|"extension")	
disallowed substitutions	Secuencia de cadenas ("restriction"|"extension"|"substitution")	
abstract	Booleano	

Element Wildcard

Nombre de la propiedad	Tipo de la propiedad	Valor de la propiedad
kind	Cadena	"Wildcard"
namespace constraint	Función con propiedades ("class": "Namespace Constraint", "variety": "any"|"enumeration"|"not", "namespaces": secuencia de xs:anyURI, "disallowed names": lista que contiene QNames y/o las cadenas "defined" y "definedSiblings"	
process contents	Cadena ("strict"|"lax"|"skip")	

Facet

Nombre de la	Tipo de la propiedad	Valor de la propiedad

© 2016-2022 Altova GmbH

propiedad		
kind	Cadena	El nombre de la faceta, por ejemplo "minLength" o "enumeration"
value	Depende de la faceta	El valor de la faceta
fixed	Booleano	
typed-value	Solo para facetas de enumeración, Array(xs:anyAtomicType*)	Una matriz que contiene los valores de la enumeración, cada uno de los cuales puede ser una secuencia de valores atómicos. (Nota: para la faceta de enumeración, la propiedad "value" es un secuencia de cadenas, independientemente del tipo)

Identity Constraint

Nombre de la propiedad	Tipo de la propiedad	Valor de la propiedad
kind	Cadena	"Identity-Constraint Definition"
name	Cadena	Nombre local de la restricción
target namespace	Cadena	URI del espacio de nombres de la restricción
identity-constraint category	Cadena ("key"|"unique"|"keyRef")	
selector	Registro de propiedades XPath	
fields	Secuencia de registros de propiedades XPath	
referenced key	(Solo para keyRef): Identity Constraint	La restricción clave correspondiente

Model Group

Nombre de la propiedad	Tipo de la propiedad	Valor de la propiedad
kind	Cadena	"Model Group"
compositor	Cadena ("sequence"|"choice"|"all")	
particles	Secuencia de partículas	

Model Group Definition

Nombre de la propiedad	Tipo de la propiedad	Valor de la propiedad
kind	Cadena	"Model Group Definition"
name	Cadena	Nombre local del grupo de modelos

target namespace	Cadena	URI del espacio de nombres del grupo de modelos
model group	Model Group	

Notation

Nombre de la propiedad	Tipo de la propiedad	Valor de la propiedad
kind	Cadena	"Notation Declaration"
name	Cadena	Nombre local de la notación
target namespace	Cadena	URI del espacio de nombres de la notación
system identifier	anyURI	
public identifier	Cadena	

Particle

Nombre de la propiedad	Tipo de la propiedad	Valor de la propiedad
kind	Cadena	"Particle"
min occurs	Número entero	
max occurs	Número entero o cadena ("unbounded")	
term	Element Declaration, Element Wildcard o ModelGroup	

Simple Type

Nombre de la propiedad	Tipo de la propiedad	Valor de la propiedad
kind	Cadena	"Simple Type Definition"
name	Cadena	Nombre local del tipo (vacío si es anónimo)
target namespace	Cadena	URI del espacio de nombres del tipo (vacío si es anónimo)
final	Secuencia de cadenas ("restriction"|"extension"|"list"|"union")	
context	Componente contenedor	
base type definition	Simple Type	
facets	Secuencia de facetas	
fundamental facets	Secuencia vacía (no implementada)	
variety	Cadena ("atomic"|"list"|"union")	
primitive type	Simple Type	

© 2016-2022 Altova GmbH

definition		
item type definition	(Solo para tipos de lista) Simple Type	
member type definitions	(Solo para tipos de unión) Secuencia de elementos Simple Type	

Type Alternative

Nombre de la propiedad	Tipo de la propiedad	Valor de la propiedad
kind	Cadena	"Type Alternative"
test	Registro de propiedades XPath	
type definition	Simple Type o Complex Type	

XPath Property Record

Nombre de la propiedad	Tipo de la propiedad	Valor de la propiedad
namespace bindings	Secuencia de funciones con propiedades ("prefix": string, "namespace": anyURI)	
default namespace	anyURI	
base URI	anyURI	El URI de base estático de la expresión XPath
expression	Cadena	La expresión XPath como cadena de texto

altova:schema(ComponentKind *as xs:string*, **Name** *as xs:QName*) **como (function(xs:string) como item()*)?** XP3.1 XQ3.1

Devuelve el tipo de componente que se indica en el primer argumento que tiene el mismo nombre que el que se indica en el segundo argumento. Para seguir navegando seleccione una de las propiedades del componente.

- Si esta propiedad es un componente seleccione una de sus propiedades para navegar hasta el siguiente nivel de profundidad. Puede repetir este paso para seguir navegando por el esquema.
- Si el componente es un valor atómico se devuelve este valor y no puede seguir navegando.

Nota: en las expresiones XQuery, el esquema se debe importar de forma explícita. En las expresiones XPath debe importar primero el esquema en el entorno de procesamiento, por ejemplo en XSLT con la instrucción **xslt:import**.

Ejemplos

- `import schema "" at "C:\Test\ExpReport.xsd";`
  `altova:schema("element declaration", xs:QName("OrgChart"))("type definition")("content type")("particles")[3]!.("term")("kind")`
  devuelve la propiedad `kind` del término del tercer componente `particles`. Este componente desciende de la declaración de elementos que tiene un `QName` de `OrgChart`

- ```
  import schema "" at "C:\Test\ExpReport.xsd";
  let $typedef := altova:schema("type definition", xs:QName("emailType"))
  for $facet in $typedef ("facets")
  return [$facet ("kind"), $facet("value")]
  ```
  devuelve, por cada **facet** de cada componente **emailType**, una matriz que contiene el tipo y el valor de ese elemento **facet**

Componentes y sus propiedades

Assertion

Nombre de la propiedad	Tipo de la propiedad	Valor de la propiedad
kind	Cadena	"Assertion"
test	Registro de propiedades XPath	

Attribute Declaration

Nombre de la propiedad	Tipo de la propiedad	Valor de la propiedad
kind	Cadena	"Attribute Declaration"
name	Cadena	Nombre local del atributo
target namespace	Cadena	URI del espacio de nombres del atributo
type definition	Simple Type o Complex Type	
scope	Una función con propiedades ("class":"Scope", "variety": "global" o "local", "parent": el Complex Type o Attribute Group contenedor)	
value constraint	Si está presente, una función con propiedades ("class": "Value Constraint", "variety": "fixed" o "default", "value": atomic value, "lexical form": string. Tenga en cuenta que la propiedad "value" no está disponible para los tipos namespace-sensitive	
inheritable	Booleano	

Attribute Group Declaration

Nombre de la propiedad	Tipo de la propiedad	Valor de la propiedad
kind	Cadena	"Attribute Group Definition"
name	Cadena	Nombre local del grupo de atributos

© 2016-2022 Altova GmbH
  ```

target namespace	Cadena	URI del espacio de nombres del grupo de atributos
attribute uses	Secuencia de (Attribute Use)	
attribute wildcard	Comodín de atributo opcional	

Attribute Use

Nombre de la propiedad	Tipo de la propiedad	Valor de la propiedad
kind	Cadena	"Attribute Use"
required	Booleano	true si el atributo es obligatorio, false si es opcional
value constraint	Véase la declaración de atributos	
inheritable	Booleano	

Attribute Wildcard

Nombre de la propiedad	Tipo de la propiedad	Valor de la propiedad
kind	Cadena	"Wildcard"
namespace constraint	Función con propiedades ("class": "Namespace Constraint", "variety": "any"|"enumeration"|"not", "namespaces": secuencia de elementos xs:anyURI, "disallowed names": lista que contiene QNames y/o las cadenas "defined" y "definedSiblings"	
process contents	Cadena ("strict"|"lax"|"skip")	

Complex Type

Nombre de la propiedad	Tipo de la propiedad	Valor de la propiedad
kind	Cadena	"Complex Type"
name	Cadena	Nombre local del tipo (vacío si es anónimo)
target namespace	Cadena	URI del espacio de nombres del tipo (vacío si es anónimo)
base type definition	Definición del Complex Type	
final	Secuencia de cadenas ("restriction"|"extension")	
context	Secuencia vacía (not implemented)	
derivation method	Cadena ("restriction"|"extension")	

abstract	Booleano	
attribute uses	Secuencia de elementos Attribute Use	
attribute wildcard	Comodín de atributo opcional	
content type	Función con propiedades: ("class":"Content Type", "variety":string ("element-only"|"empty"|"mixed"|"simple"), particle: partícula opcional, "open content": función con propiedades ("class":"Open Content", "mode": string ("interleave"|"suffix"), "wildcard": Wildcard), "simple type definition": Simple Type)	
prohibited substitutions	Secuencia de cadenas ("restriction"|"extension")	
assertions	Secuencia de elementos Assertion	

Element Declaration

Nombre de la propiedad	Tipo de la propiedad	Valor de la propiedad
kind	Cadena	"Complex Type"
name	Cadena	Nombre local del tipo (vacío si es anónimo)
target namespace	Cadena	Namespace URI del tipo (vacío si es anónimo)
type definition	Simple Type o Complex Type	
type table	Función con propiedades ("class":"Type Table", "alternatives": secuencia de elementos Type Alternative, "default type definition": Simple Type o Complex Type)	
scope	Función con propiedades ("class":"Scope", "variety": ("global"|"local"), "parent": Complex Type opcional)	
value constraint	véase Attribute Declaration	
nillable	Booleano	
identity-constraint definitions	Secuencia de restricciones de identidad	
substitution group affiliations	Secuencia de declaraciones de elementos	
substitution group exclusions	Secuencia de cadenas ("restriction"|"extension")	
disallowed substitutions	Secuencia de cadenas ("restriction"|"extension"|"substitution")	

 © 2016-2022 Altova GmbH

abstract	Booleano	

Element Wildcard

Nombre de la propiedad	Tipo de la propiedad	Valor de la propiedad
kind	Cadena	"Wildcard"
namespace constraint	Función con propiedades ("class": "Namespace Constraint", "variety": "any"|"enumeration"|"not", "namespaces": secuencia de xs:anyURI, "disallowed names": lista que contiene QNames y/o las cadenas "defined" y "definedSiblings"	
process contents	Cadena ("strict"|"lax"|"skip")	

Facet

Nombre de la propiedad	Tipo de la propiedad	Valor de la propiedad
kind	Cadena	El nombre de la faceta, por ejemplo "minLength" o "enumeration"
value	Depende de la faceta	El valor de la faceta
fixed	Booleano	
typed-value	Solo para facetas de enumeración, Array(xs:anyAtomicType*)	Una matriz que contiene los valores de la enumeración, cada uno de los cuales puede ser una secuencia de valores atómicos. (Nota: para la faceta de enumeración, la propiedad "value" es un secuencia de cadenas, independientemente del tipo)

Identity Constraint

Nombre de la propiedad	Tipo de la propiedad	Valor de la propiedad
kind	Cadena	"Identity-Constraint Definition"
name	Cadena	Nombre local de la restricción
target namespace	Cadena	URI del espacio de nombres de la restricción
identity-constraint category	Cadena ("key"|"unique"|"keyRef")	
selector	Registro de propiedades XPath	
fields	Secuencia de registros de propiedades XPath	

referenced key	(Solo para keyRef): Identity Constraint	La restricción clave correspondiente

Model Group

Nombre de la propiedad	Tipo de la propiedad	Valor de la propiedad
kind	Cadena	"Model Group"
compositor	Cadena ("sequence"|"choice"|"all")	
particles	Secuencia de partículas	

Model Group Definition

Nombre de la propiedad	Tipo de la propiedad	Valor de la propiedad
kind	Cadena	"Model Group Definition"
name	Cadena	Nombre local del grupo de modelos
target namespace	Cadena	URI del espacio de nombres del grupo de modelos
model group	Model Group	

Notation

Nombre de la propiedad	Tipo de la propiedad	Valor de la propiedad
kind	Cadena	"Notation Declaration"
name	Cadena	Nombre local de la notación
target namespace	Cadena	URI del espacio de nombres de la notación
system identifier	anyURI	
public identifier	Cadena	

Particle

Nombre de la propiedad	Tipo de la propiedad	Valor de la propiedad
kind	Cadena	"Particle"
min occurs	Número entero	
max occurs	Número entero o cadena ("unbounded")	
term	Element Declaration, Element Wildcard o ModelGroup	

Simple Type

Nombre de la	Tipo de la propiedad	Valor de la propiedad

© 2016-2022 Altova GmbH

propiedad		
kind	Cadena	"Simple Type Definition"
name	Cadena	Nombre local del tipo (vacío si es anónimo)
target namespace	Cadena	URI del espacio de nombres del tipo (vacío si es anónimo)
final	Secuencia de cadenas ("restriction"|"extension"|"list"|"union")	
context	Componente contenedor	
base type definition	Simple Type	
facets	Secuencia de facetas	
fundamental facets	Secuencia vacía (no implementada)	
variety	Cadena ("atomic"|"list"|"union")	
primitive type definition	Simple Type	
item type definition	(Solo para tipos de lista) Simple Type	
member type definitions	(Solo para tipos de unión) Secuencia de elementos Simple Type	

Type Alternative

Nombre de la propiedad	Tipo de la propiedad	Valor de la propiedad
kind	Cadena	"Type Alternative"
test	Registro de propiedades XPath	
type definition	Simple Type o Complex Type	

XPath Property Record

Nombre de la propiedad	Tipo de la propiedad	Valor de la propiedad
namespace bindings	Secuencia de funciones con propiedades ("prefix": string, "namespace": anyURI)	
default namespace	anyURI	
base URI	anyURI	El URI de base estático de la expresión XPath
expression	Cadena	La expresión XPath como cadena de texto

`altova:type(Node` *`as item?`*`) como (function(xs:string) como item()*)?` XP3.1 XQ3.1

La función **`altova:type`** indica un nodo de elemento o atributo de un documento XML y devuelve la información

del tipo de nodo del PSVI (Conjunto de información posterior a la validación de esquemas).

Nota: el documento XML debe tener una declaración de esquema para que se pueda hacer referencia al esquema.

Ejemplos

- ```
  for $element in //Email
  let $type := altova:type($element)
  return $type
  ```
  devuelve una función que contiene información sobre el tipo de nodo

- ```
  for $element in //Email
  let $type := altova:type($element)
  return $type ("kind")
  ```
  toma el componente de tipo del nodo (tipo simple o complejo) y devuelve el valor de la propiedad **kind** del componente

El parámetro **"_props"** devuelve las propiedades del componente seleccionado. Por ejemplo:

- ```
  for $element in //Email
  let $type := altova:type($element)
  return ($type ("kind"), $type ("_props"))
  ```
  toma el componente de tipo del nodo (tipo simple o complejo) y devuelve (i) el valor de la propiedad **kind** del componente y después (ii) las propiedades de ese componente

Componentes y sus propiedades

Assertion

Nombre de la propiedad	Tipo de la propiedad	Valor de la propiedad
kind	Cadena	"Assertion"
test	Registro de propiedades XPath	

Attribute Declaration

Nombre de la propiedad	Tipo de la propiedad	Valor de la propiedad
kind	Cadena	"Attribute Declaration"
name	Cadena	Nombre local del atributo
target namespace	Cadena	URI del espacio de nombres del atributo
type definition	Simple Type o Complex Type	
scope	Una función con propiedades ("class":"Scope", "variety": "global" o "local", "parent": el Complex Type o Attribute Group contenedor)	
value constraint	Si está presente, una función con	

 © 2016-2022 Altova GmbH
  ```

	propiedades ("class": "Value Constraint", "variety": "fixed" o "default", "value": atomic value, "lexical form": string. Tenga en cuenta que la propiedad "value" no está disponible para los tipos namespace-sensitive	
inheritable	Booleano	

Attribute Group Declaration

Nombre de la propiedad	Tipo de la propiedad	Valor de la propiedad
kind	Cadena	"Attribute Group Definition"
name	Cadena	Nombre local del grupo de atributos
target namespace	Cadena	URI del espacio de nombres del grupo de atributos
attribute uses	Secuencia de (Attribute Use)	
attribute wildcard	Comodín de atributo opcional	

Attribute Use

Nombre de la propiedad	Tipo de la propiedad	Valor de la propiedad
kind	Cadena	"Attribute Use"
required	Booleano	true si el atributo es obligatorio, false si es opcional
value constraint	Véase la declaración de atributos	
inheritable	Booleano	

Attribute Wildcard

Nombre de la propiedad	Tipo de la propiedad	Valor de la propiedad
kind	Cadena	"Wildcard"
namespace constraint	Función con propiedades ("class": "Namespace Constraint", "variety": "any"|"enumeration"|"not", "namespaces": secuencia de elementos xs:anyURI, "disallowed names": lista que contiene QNames y/o las cadenas "defined" y "definedSiblings"	
process contents	Cadena ("strict"|"lax"|"skip")	

Complex Type

Nombre de la propiedad	Tipo de la propiedad	Valor de la propiedad
kind	Cadena	"Complex Type"
name	Cadena	Nombre local del tipo (vacío si es anónimo)
target namespace	Cadena	URI del espacio de nombres del tipo (vacío si es anónimo)
base type definition	Definición del Complex Type	
final	Secuencia de cadenas ("restriction"|"extension")	
context	Secuencia vacía (not implemented)	
derivation method	Cadena ("restriction"|"extension")	
abstract	Booleano	
attribute uses	Secuencia de elementos Attribute Use	
attribute wildcard	Comodín de atributo opcional	
content type	Función con propiedades: ("class":"Content Type", "variety":string ("element-only"|"empty"|"mixed"|"simple"), particle: partícula opcional, "open content": función con propiedades ("class":"Open Content", "mode": string ("interleave"|"suffix"), "wildcard": Wildcard), "simple type definition": Simple Type)	
prohibited substitutions	Secuencia de cadenas ("restriction"|"extension")	
assertions	Secuencia de elementos Assertion	

Element Declaration

Nombre de la propiedad	Tipo de la propiedad	Valor de la propiedad
kind	Cadena	"Complex Type"
name	Cadena	Nombre local del tipo (vacío si es anónimo)
target namespace	Cadena	Namespace URI del tipo (vacío si es anónimo)
type definition	Simple Type o Complex Type	
type table	Función con propiedades ("class":"Type Table", "alternatives": secuencia de elementos Type Alternative, "default type definition": Simple Type o Complex Type)	
scope	Función con propiedades	

© 2016-2022 Altova GmbH

	("class":"Scope", "variety": ("global"|"local"), "parent": Complex Type opcional)	
value constraint	véase Attribute Declaration	
nillable	Booleano	
identity-constraint definitions	Secuencia de restricciones de identidad	
substitution group affiliations	Secuencia de declaraciones de elementos	
substitution group exclusions	Secuencia de cadenas ("restriction"|"extension")	
disallowed substitutions	Secuencia de cadenas ("restriction"|"extension"|"substitution")	
abstract	Booleano	

Element Wildcard

Nombre de la propiedad	Tipo de la propiedad	Valor de la propiedad
kind	Cadena	"Wildcard"
namespace constraint	Función con propiedades ("class": "Namespace Constraint", "variety": "any"|"enumeration"|"not", "namespaces": secuencia de xs:anyURI, "disallowed names": lista que contiene QNames y/o las cadenas "defined" y "definedSiblings"	
process contents	Cadena ("strict"|"lax"|"skip")	

Facet

Nombre de la propiedad	Tipo de la propiedad	Valor de la propiedad
kind	Cadena	El nombre de la faceta, por ejemplo "minLength" o "enumeration"
value	Depende de la faceta	El valor de la faceta
fixed	Booleano	
typed-value	Solo para facetas de enumeración, Array(xs:anyAtomicType*)	Una matriz que contiene los valores de la enumeración, cada uno de los cuales puede ser una secuencia de valores atómicos. (Nota: para la faceta de enumeración, la propiedad "value" es un secuencia de cadenas, independientemente del tipo)

Identity Constraint

Nombre de la propiedad	Tipo de la propiedad	Valor de la propiedad
kind	Cadena	"Identity-Constraint Definition"
name	Cadena	Nombre local de la restricción
target namespace	Cadena	URI del espacio de nombres de la restricción
identity-constraint category	Cadena ("key"|"unique"|"keyRef")	
selector	Registro de propiedades XPath	
fields	Secuencia de registros de propiedades XPath	
referenced key	(Solo para keyRef): Identity Constraint	La restricción clave correspondiente

Model Group

Nombre de la propiedad	Tipo de la propiedad	Valor de la propiedad
kind	Cadena	"Model Group"
compositor	Cadena ("sequence"|"choice"|"all")	
particles	Secuencia de partículas	

Model Group Definition

Nombre de la propiedad	Tipo de la propiedad	Valor de la propiedad
kind	Cadena	"Model Group Definition"
name	Cadena	Nombre local del grupo de modelos
target namespace	Cadena	URI del espacio de nombres del grupo de modelos
model group	Model Group	

Notation

Nombre de la propiedad	Tipo de la propiedad	Valor de la propiedad
kind	Cadena	"Notation Declaration"
name	Cadena	Nombre local de la notación
target namespace	Cadena	URI del espacio de nombres de la notación
system identifier	anyURI	
public identifier	Cadena	

© 2016-2022 Altova GmbH

Particle

Nombre de la propiedad	Tipo de la propiedad	Valor de la propiedad
kind	Cadena	"Particle"
min occurs	Número entero	
max occurs	Número entero o cadena ("unbounded")	
term	Element Declaration, Element Wildcard o ModelGroup	

Simple Type

Nombre de la propiedad	Tipo de la propiedad	Valor de la propiedad
kind	Cadena	"Simple Type Definition"
name	Cadena	Nombre local del tipo (vacío si es anónimo)
target namespace	Cadena	URI del espacio de nombres del tipo (vacío si es anónimo)
final	Secuencia de cadenas ("restriction"|"extension"|"list"|"union")	
context	Componente contenedor	
base type definition	Simple Type	
facets	Secuencia de facetas	
fundamental facets	Secuencia vacía (no implementada)	
variety	Cadena ("atomic"|"list"|"union")	
primitive type definition	Simple Type	
item type definition	(Solo para tipos de lista) Simple Type	
member type definitions	(Solo para tipos de unión) Secuencia de elementos Simple Type	

Type Alternative

Nombre de la propiedad	Tipo de la propiedad	Valor de la propiedad
kind	Cadena	"Type Alternative"
test	Registro de propiedades XPath	
type definition	Simple Type o Complex Type	

XPath Property Record

Nombre de la propiedad	Tipo de la propiedad	Valor de la propiedad
namespace bindings	Secuencia de funciones con propiedades ("prefix": string, "namespace": anyURI)	
default namespace	anyURI	
base URI	anyURI	El URI de base estático de la expresión XPath
expression	Cadena	La expresión XPath como cadena de texto

18.2.2.1.7 Funciones XPath/XQuery: de secuencia

Las funciones de extensión de Altova para trabajar con secuencias pueden utilizarse en expresiones XPath y XQuery y ofrecen funciones adicionales para el procesamiento de datos. Estas funciones se pueden usar con los motores **XPath 3.0** y **XQuery 3.0** de Altova. Están disponibles en contextos XPath/XQuery.

Nota sobre el nombre de las funciones y lenguajes

Puede utilizar todas las funciones de extensión de Altova en sus expresiones XPath/XQuery. Con ellas conseguirá funciones adicionales no disponibles en la biblioteca de funciones estándar de XPath, XQuery y XSLT. Las funciones de extensión de Altova están en el **espacio de nombres** `http://www.altova.com/xslt-extensions` y en esta sección se presentan con el prefijo, que se supone estará enlazado al espacio de nombres señalado. Tenga en cuenta que en futuras versiones del producto algunas funciones pueden dejar de ser compatibles o su comportamiento puede cambiar. Por tanto, consulte siempre la documentación del producto para conocer el funcionamiento de estas funciones en cada versión del producto.

Funciones XPath (en expresiones XPath en XSLT):	XP1 XP2 XP3.1.1
Funciones XSLT (en expresiones XPath en XSLT):	XSLT1 XSLT2 XSLT3
Funciones XQuery (en expresiones XQuery en XQuery):	XQ1 XQ3.1

attributes [altova:]

altova:attributes(NombreAtributo *as xs:string***)** como **attribute()*** XP3.1 XQ3.1

Devuelve todos los atributos cuyo nombre local coincida con el nombre dado como argumento de entrada (`NombreAtributo`). La búsqueda tiene en cuenta el uso de mayúsculas y minúsculas y se lleva a cabo en el eje `attribute::`.

Ejemplos

- **altova:attributes**("MiAtributo") devuelve MiAtributo()*

© 2016-2022 Altova GmbH

altova:attributes(NombreAtributo *as xs:string*, **OpcionesBúsqueda** *as xs:string*) como **attribute()*** XP3.1 XQ3.1
Devuelve todos los atributos cuyo nombre local coincida con el nombre dado como argumento de entrada (`NombreAtributo`). La búsqueda tiene en cuenta el uso de mayúsculas y minúsculas y se lleva a cabo en el eje `attribute::`. El segundo argumento es una cadena con marcas de búsqueda. Estas son las marcas disponibles:

r = habilita la búsqueda de expresiones regulares. En este caso, `NombreAtributo` debe ser una cadena de búsqueda de expresión regular;
i = la búsqueda no tiene en cuenta el uso de mayúsculas y minúsculas;
p = incluye el prefijo de espacio de nombres en la búsqueda. En este caso, `NombreAtributo` debe contener el prefijo de espacio de nombres (p. ej.: `MiAtributo`).

Las marcas pueden escribirse en cualquier orden y no hace falta utilizar todas. Si usa marcas no válidas, se genera un error. También puede usar una cadena vacía para el segundo argumento. Esto tiene el mismo efecto que usar solo el primer argumento. Sin embargo, no está permitido usar una secuencia vacía.

Ejemplos

- **altova:attributes**("MiAtributo", "rip") devuelve MiAtributo()*
- **altova:attributes**("MiAtributo", "pri") devuelve MiAtributo()*
- **altova:attributes**("MiAtributo", "") devuelve MiAtributo()*
- **altova:attributes**("MiAtributo", "Rip") devuelve un error de marca desconocida.
- **altova:attributes**("MiAtributo",) devuelve un error diciendo que falta el segundo argumento.

elements [altova:]

altova:elements(NombreElemento *as xs:string*) como **elemento()*** XP3.1 XQ3.1
Devuelve todos los elementos cuyo nombre local coincida con el nombre dado como argumento de entrada (`NombreElemento`). La búsqueda tiene en cuenta el uso de mayúsculas y minúsculas y se lleva a cabo en el eje `child::`.

Ejemplos

- **altova:elements**("MiElemento") devuelve MiElemento()*

altova:elements(NombreElemento *as xs:string*, **OpcionesBúsqueda** *as xs:string*) como **elemento()*** XP3.1 XQ3.1
Devuelve todos los elementos cuyo nombre local coincida con el nombre dado como argumento de entrada (`NombreElemento`). La búsqueda tiene en cuenta el uso de mayúsculas y minúsculas y se lleva a cabo en el eje `child::`. El segundo argumento es una cadena con marcas de búsqueda. Estas son las marcas disponibles:

r = habilita la búsqueda de expresiones regulares. En este caso, `NombreElemento` debe ser una cadena de búsqueda de expresión regular;
i = la búsqueda no tiene en cuenta el uso de mayúsculas y minúsculas;
p = incluye el prefijo de espacio de nombres en la búsqueda. En este caso, `NombreElemento` debe contener el prefijo de espacio de nombres (p. ej.: `MiElemento`).

Las marcas pueden escribirse en cualquier orden y no hace falta utilizar todas. Si usa marcas no válidas, se genera un error. También puede usar una cadena vacía para el segundo argumento. Esto tiene el mismo efecto que usar solo el primer argumento. Sin embargo, no está permitido usar una secuencia vacía.

Ejemplos

- **altova:elements**("MiElemento", "rip") devuelve MiElemento()*
- **altova:elements**("MiElemento", "pri") devuelve MiElemento()*
- **altova:elements**("MiElemento", "") devuelve MiElemento()*
- **altova:elements**("MiElemento", "Rip") devuelve un error de marca desconocida.
- **altova:elements**("MiElemento",) devuelve un error diciendo que falta el segundo argumento.

find-first [altova:]

altova:find-first((Secuencia *()**)**, (Condición(Elemento-Secuencia** *como xs:boolean***))** como **item()?** XP3.1 XQ3.1

Esta función toma dos argumentos. El primero es una secuencia de uno o varios elementos de cualquier tipo de datos. El segundo argumento, **Condición**, es una referencia a una función XPath que toma un argumento (es decir, su aridad es 1) y devuelve un valor binario. Cada elemento de **Secuencia** se envía a su vez a la función a la que se hace referencia en **Condición**. Nota: recuerde que esta función solo toma un argumento. El primer elemento de **Secuencia** que consiga que la función de **Condición** dé **true()** como resultado se devuelve como resultado de **find-first** y la iteración se detiene.

Ejemplos

- **altova:find-first**(5 to 10, function($a) {$a mod 2 = 0}) devuelve xs:integer 6
  El argumento **Condición** remite a la función inline XPath 3.0 **function()**, que declara una función inline llamada **$a** y después la define. Cada elemento del argumento Secuencia de **find-first** se envía a su vez como valor de entrada a **$a**. El valor de entrada se prueba en la condición en la definición de función ($a mod 2 = 0). El primer valor de entrada que cumpla la condición se devuelve como resultado de **find-first** (en este caso **6**).

- **altova:find-first**((1 to 10), (function($a) {$a+3=7})) devuelve xs:integer 4

Más ejemplos

Si existe el archivo **C:\Temp\Customers.xml**:

- **altova:find-first**(("C:\Temp\Customers.xml", "http://www.altova.com/index.html"), (doc-available#1)) devuelve xs:string C:\Temp\Customers.xml

Si no existe el archivo **C:\Temp\Customers.xml** pero existe **http://www.altova.com/index.html**:

- **altova:find-first**(("C:\Temp\Customers.xml", "http://www.altova.com/index.html"), (doc-available#1)) devuelve xs:string http://www.altova.com/index.html

Si no existe el archivo **C:\Temp\Customers.xml** y tampoco existe **http://www.altova.com/index.html**:

© 2016-2022 Altova GmbH

- **altova:find-first**(("C:\Temp\Customers.xml", "http://www.altova.com/index.html"), (doc-available#1)) no devuelve ningún resultado

Notas sobre los ejemplos anteriores

- La función XPath 3.0 doc-available toma un solo argumento de cadena, que se usa como URI, y devuelve true si en el URI dado se encuentra un nodo de documento. El documento que está en el URI dado debe ser un documento XML.
- La función doc-available se puede usar para **Condición**, el segundo argumento de find-first, porque solamente toma un argumento (aridad=1), porque toma un item() como entrada (una cadena que se usa como URI) y devuelve un valor binario.
- Recuerde que solamente se hace referencia a la función doc-available pero no se le llama. El sufijo #1 que se anexa a la función indica una función cuya aridad es 1. Es decir, doc-available#1 simplemente significa "*Utilizar la función doc-availabe() que tiene aridad=1, pasándole como solo argumento a su vez cada uno de los elementos de la primera secuencia."* Como resultado, se pasarán las dos cadenas a **doc-available()**, que utiliza la cadena como URI y prueba si existe un nodo de documento en el URI. Si existe, entonces doc-available() da como resultado true() y esa cadena se devuelve como resultado de la función **find-first**. Nota sobre la función doc-available(): las rutas de acceso relativas se resuelven en relación al URI base actual, que es por defecto el URI del documento XML desde el que se carga la función.

find-first-combination [altova:]

altova:find-first-combination((Sec-01 *como item()**), **(Sec-02** *como item()**), **(Condición(Elem-Sec-01, Elem-Sec-02** *como xs:boolean*)) como **item()*** XP3.1 XQ3.1

Esta función toma tres argumentos:

- Los dos primeros (**Sec-01** y **Sec-02**) son secuencias de uno o más elementos de cualquier tipo de datos.
- El tercero (**Condición**) es una referencia a una función XPath que toma dos argumentos (su aridad es 2) y devuelve un valor binario.

Los elementos de **Sec-01** y **Sec-02** se pasan en pares ordenados (cada par está formado por un elemento de cada secuencia) como argumentos de la función de **Condición**. Los pares se ordenan de la siguiente manera:

```
Si   Sec-01 = X1, X2, X3 ... Xn
Y  Sec-02 = Y1, Y2, Y3 ... Yn
Entonces (X1 Y1), (X1 Y2), (X1 Y3) ... (X1 Yn), (X2 Y1), (X2 Y2) ... (Xn Yn)
```

El primer par ordenado que consiga que la función de **Condición** dé como resultado **true()** se devuelve como resultado de **find-first-combination**. Recuerde que (i) si la función de **Condición** recorre los pares de argumentos dados y no consigue dar **true()** como resultado ni una vez, entonces **find-first-combination** devuelve *Sin resultados;* (ii) el resultado de **find-first-combination** siempre será un par de elementos (de cualquier tipo de datos) o ningún elemento.

Ejemplos

- **altova:find-first-pair**(11 to 20, 21 to 30, function($a, $b) {$a+$b = 32}) devuelve

la secuencia de `xs:integers (11, 21)`
- **`altova:find-first-pair`**`(11 to 20, 21 to 30, function($a, $b) {$a+$b = 33})` devuelve la secuencia de `xs:integers (11, 22)`
- **`altova:find-first-pair`**`(11 to 20, 21 to 30, function($a, $b) {$a+$b = 34})` devuelve la secuencia de `xs:integers (11, 23)`

find-first-pair [altova:]

altova:find-first-pair((Sec-01 *como item()**)**, (Sec-02** *como item()**)**, (Condición(Elem-Sec-01, Elem-Sec-02** *como xs:boolean*)**)** como **item()*** XP3.1 XQ3.1
Esta función toma tres argumentos:

- Los dos primeros (**Sec-01** y **Sec-02**) son secuencias de uno o más elementos de cualquier tipo de datos.
- El tercero (**Condición**) es una referencia a una función XPath que toma dos argumentos (su aridad es 2) y devuelve un valor binario.

Los elementos de **Sec-01** y **Sec-02** se pasan en pares ordenados como argumentos de la función de **Condición**. Los pares se ordenan de la siguiente manera:

```
Si   Sec-01 = X1, X2, X3 ... Xn
Y  Sec-02 = Y1, Y2, Y3 ... Yn
Entonces (X1 Y1), (X2 Y2), (X3 Y3) ... (Xn Yn)
```

El primer par ordenado que consiga que la función de **Condición** dé como resultado **true()** se devuelve como resultado de **find-first-pair**. Recuerde que (i) si la función de **Condición** recorre los pares de argumentos dados y no consigue dar **true()** como resultado ni una vez, entonces **find-first-pair** devuelve *Sin resultados;* (ii) el resultado de **find-first-pair** siempre será un par de elementos (de cualquier tipo de datos) o ningún elemento.

Ejemplos

- **`altova:find-first-pair`**`(11 to 20, 21 to 30, function($a, $b) {$a+$b = 32})` devuelve la secuencia de `xs:integers (11, 21)`
- **`altova:find-first-pair`**`(11 to 20, 21 to 30, function($a, $b) {$a+$b = 33})` devuelve *Sin resultados*

Observe que en los dos ejemplos anteriores el orden de los pares es: `(11, 21) (12, 22) (13, 23)...(20, 30)`. Por ese motivo el segundo ejemplo no obtiene resultados (porque ningún par ordenado consigue sumar `33`).

find-first-pair-pos [altova:]

altova:find-first-pair-pos((Sec-01 *como item()**)**, (Sec-02** *como item()**)**, (Condición(Elem-Sec-01, Elem-Sec-02** *como xs:boolean*)**)** como **xs:integer** XP3.1 XQ3.1
Esta función toma tres argumentos:

- Los dos primeros (**Sec-01** y **Sec-02**) son secuencias de uno o más elementos de cualquier tipo de datos.
- El tercero (**Condición**) es una referencia a una función XPath que toma dos argumentos (su

aridad es 2) y devuelve un valor binario.

Los elementos de **Sec-01** y **Sec-02** se pasan en pares ordenados como argumentos de la función de **Condición**. Los pares se ordenan de la siguiente manera:

```
Si   Sec-01 = X1, X2, X3 ... Xn
Y  Sec-02 = Y1, Y2, Y3 ... Yn
Entonces (X1 Y1), (X2 Y2), (X3 Y3) ... (Xn Yn)
```

La posición de índice del primer par ordenado que consiga que la función de **Condición** dé como resultado **true()** se devuelve como resultado de **find-first-pair-pos**. Recuerde que si la función de **Condición** recorre los pares de argumentos dados y no da como resultado **true()** ni una sola vez, entonces **find-first-pair-pos** devuelve *Sin resultados*.

Ejemplos

- **altova:find-first-pair**(11 to 20, 21 to 30, function($a, $b) {$a+$b = 32}) devuelve 1
- **altova:find-first-pair**(11 to 20, 21 to 30, function($a, $b) {$a+$b = 33}) devuelve *Sin resultados*

Observe que en los dos ejemplos anteriores el orden de los pares es: (11, 21) (12, 22) (13, 23)...(20, 30). En el primer ejemplo el primer par consigue que la función de **Condición** dé como resultado **true()** y, por tanto, se devuelve la posición de índice que tienen en la secuencia (**1**). El segundo ejemplo, sin embargo, devuelve *Sin resultados* porque ningún par consigue sumar 33.

find-first-pos [altova:]

altova:find-first-pos((Secuencia *como item()**), **(Condición(Elem-Sec** *como xs:boolean*))
como **xs:integer** XP3.1 XQ3.1

Esta función toma dos argumentos. El primer argumento es una secuencia de uno o varios elementos de cualquier tipo. El segundo argumento (**Condición**) es una referencia a una función XPath que toma un argumento (su aridad es 1) y devuelve un valor binario. Cada elemento de **Secuencia** se envía a su vez a la función a la que se hace referencia en **Condición**. (Recuerde que esta función toma un solo argumento.) El primer elemento de **Secuencia** que consiga que la función de **Condición** dé como resultado **true()** devuelve la posición de índice que tiene en **Secuencia** como resultado de **find-first-pos** y la iteración se detiene.

Ejemplos

- **altova:find-first-pos**(5 to 10, function($a) {$a mod 2 = 0}) devuelve xs:integer 2
  El argumento **Condición** hace referencia a la función inline XPath 3.0 **function()**, que declara una función inline llamada **$a** y después la define. Cada elemento del argumento **Sequence** de **find-first-pos** se pasa a su vez como valor de entrada de **$a**. El valor de entrada se prueba en la condición de la definición de función ($a mod 2 = 0). La posición de índice que tiene en la secuencia el primer valor de entrada que cumple la condición se devuelve como resultado de **find-first-pos** (en este caso es la posición de índice **2**, porque **6** es el primer valor (de la secuencia) que cumple la condición y su posición de índice en la secuencia es **2**).

- **altova:find-first-pos**((2 to 10), (function($a) {$a+3=7})) devuelve xs:integer 3

Más ejemplos
Si existe el archivo **C:\Temp\Customers.xml**:

- **altova:find-first-pos**(("C:\Temp\Customers.xml", "http://www.altova.com/index.html"), (doc-available#1)) devuelve 1

Si no existe el archivo **C:\Temp\Customers.xml** pero existe **http://www.altova.com/index.html**:

- **altova:find-first-pos**(("C:\Temp\Customers.xml", "http://www.altova.com/index.html"), (doc-available#1)) devuelve 2

Si no existe el archivo **C:\Temp\Customers.xml** y tampoco existe **http://www.altova.com/index.html**:

- **altova:find-first-pos**(("C:\Temp\Customers.xml", "http://www.altova.com/index.html"), (doc-available#1)) no devuelve ningún resultado

Notas sobre los ejemplos anteriores

- La función XPath 3.0 `doc-available` toma un solo argumento de cadena, que se usa como URI, y devuelve `true` si en el URI dado se encuentra un nodo de documento. El documento que está en el URI dado debe ser un documento XML.
- La función `doc-available` se puede usar para **Condición**, el segundo argumento de `find-first-pos`, porque solamente toma un argumento (aridad=1), porque toma un `item()` como entrada (una cadena que se usa como URI) y devuelve un valor binario.
- Recuerde que solamente se hace referencia a la función `doc-available` pero no se le llama. El sufijo `#1` que se anexa a la función indica una función cuya aridad es 1. Es decir, `doc-available#1` simplemente significa "*Utilizar la función doc-availabe() que tiene aridad=1, pasándole como solo argumento a su vez cada uno de los elementos de la primera secuencia.*" Como resultado, se pasarán las dos cadenas a **doc-available()**, que utiliza la cadena como URI y prueba si existe un nodo de documento en el URI. Si existe, entonces `doc-available()` da como resultado `true()` y esa cadena se devuelve como resultado de la función **find-first-pos**. Nota sobre la función `doc-available()`: las rutas de acceso relativas se resuelven en relación al URI base actual, que es por defecto el URI del documento XML desde el que se carga la función.

for-each-attribute-pair [altova:]

altova:for-each-attribute-pair(Seq1 *como element()?*, **Seq2** *como element()?*, **Function** *como function()*) como **item()*** XP3.1 XQ3.1

Los primeros dos argumentos identifican dos elementos cuyos atributos se usan para construir pares de atributos donde uno de los atributos del par se obtiene del primer elemento y el otro atributo del segundo elemento. Los pares de atributos se seleccionan basándose en que tienen el mismo nombre y se ordenan alfabéticamente por grupos. Si un atributo no tiene un atributo correspondiente en el otro elemento, entonces el par está "desarticulado", lo que significa que tiene un solo miembro. El elemento de la función (tercer argumento `Function`) se aplica por separado a cada par de la secuencia de pares (articulados y desarticulados) y el resultado es una secuencia de elementos.

Ejemplos

 © 2016-2022 Altova GmbH

- **altova:for-each-attribute-pair**(/Example/Test-A, /Example/Test-B, function($a, $b) {$a+b}) devuelve...

  (2, 4, 6) si
  <Test-A att1="1" att2="2" att3="3" />
  <Test-B att1="1" att2="2" att3="3" />

  (2, 4, 6) si
  <Test-A att2="2" att1="1" att3="3" />
  <Test-B att3="3" att2="2" att1="1" />

  (2, 6) si
  <Test-A att4="4" att1="1" att3="3" />
  <Test-B att3="3" att2="2" att1="1" />

  Nota: El resultado **(2, 6)** *se obtiene mediante la siguiente acción:* **(1+1, ()+2, 3+3, 4+())**. *Si uno de los operandos es la secuencia vacía, como en el caso de los elementos 2 y 4, entonces el resultado de la suma es una secuencia vacía.*

- **altova:for-each-attribute-pair**(/Example/Test-A, /Example/Test-B, concat#2) devuelve...

  (11, 22, 33) si
  <Test-A att1="1" att2="2" att3="3" />
  <Test-B att1="1" att2="2" att3="3" />

  (11, 2, 33, 4) si
  <Test-A att4="4" att1="1" att3="3" />
  <Test-B att3="3" att2="2" att1="1" />

for-each-combination [altova:]

altova:for-each-combination(PrimeraSecuencia *como item()**, **SegundaSecuencia** *como item()**, **función($i,$j){$i || $j})** como **item()*** XP3.1 XQ3.1

Los elementos de las dos secuencias en los primeros dos argumentos se combinan de forma que el primer elemento de la primera secuencia se combina, en orden, una vez con cada elemento de la segunda secuencia. La función dada como tercer argumento se aplica a cada una de las combinaciones de la secuencia resultante y da como resultado una secuencia de elementos (*véase ejemplo*).

Ejemplos

- **altova:for-each-combination**(('a', 'b', 'c'), ('1', '2', '3'), function($i, $j) {$i || $j}) devuelve ('a1', 'a2', 'a3', 'b1', 'b2', 'b3', 'c1', 'c2', 'c3')

for-each-matching-attribute-pair [altova:]

altova:for-each-matching-attribute-pair(Seq1 *como element()?*, **Seq2** *como element()?*, **Function** *como function()*) como **item()*** XP3.1 XQ3.1

Los primeros dos argumentos identifican dos elementos cuyos atributos se usan para construir pares de atributos donde un atributo de cada par se obtiene del primer elemento y el otro atributo del par se obtiene

del segundo elemento. Los pares de elementos se seleccionan basándose en que tienen el mismo nombre y se ordenan alfabéticamente por grupos. Si un atributo no tiene un atributo correspondiente en el otro elemento, entonces no se construye ningún par. El elemento de la función (tercer argumento `Function`) se aplica por separado a cada par de la secuencia de pares (articulados y desarticulados) y el resultado es una secuencia de elementos.

Ejemplos

- **`altova:for-each-matching-attribute-pair`**`(/Example/Test-A, /Example/Test-B, function($a, $b){$a+b})` devuelve...

  `(2, 4, 6)` if
  `<Test-A att1="1" att2="2" att3="3" />`
  `<Test-B att1="1" att2="2" att3="3" />`

  `(2, 4, 6)` if
  `<Test-A att2="2" att1="1" att3="3" />`
  `<Test-B att3="3" att2="2" att1="1" />`

  `(2, 6)` if
  `<Test-A att4="4" att1="1" att3="3" />`
  `<Test-B att3="3" att2="2" att3="1" />`

- **`altova:for-each-matching-attribute-pair`**`(/Example/Test-A, /Example/Test-B, concat#2)` devuelve...

  `(11, 22, 33)` if
  `<Test-A att1="1" att2="2" att3="3" />`
  `<Test-B att1="1" att2="2" att3="3" />`

  `(11, 33)` if
  `<Test-A att4="4" att1="1" att3="3" />`
  `<Test-B att3="3" att2="2" att1="1" />`

substitute-empty [altova:]

`altova:substitute-empty(PrimeraSecuencia` *`as item()*`**`, SegundaSecuencia`** *`as item()`***`)`** como **`item()*`** XP3.1 XQ3.1

Si **`PrimeraSecuencia`** está vacío, la función devuelve **`SegundaSecuencia`**. Si **`PrimeraSecuencia`** no está vacío, la función devuelve **`PrimeraSecuencia`**.

Ejemplos

- **`altova:substitute-empty`**`( (1,2,3), (4,5,6) )` devuelve `(1,2,3)`
- **`altova:substitute-empty`**`( (), (4,5,6) )` devuelve `(4,5,6)`

© 2016-2022 Altova GmbH

18.2.2.1.8 Funciones XPath/XQuery: de cadena

Las funciones de extensión de Altova para trabajar con cadenas pueden utilizarse en expresiones XPath y XQuery y ofrecen funciones adicionales para el procesamiento de datos. Estas funciones se pueden usar con los motores **XPath 3.0** y **XQuery 3.0** de Altova. Están disponibles en contextos XPath/XQuery.

Nota sobre el nombre de las funciones y lenguajes

Puede utilizar todas las funciones de extensión de Altova en sus expresiones XPath/XQuery. Con ellas conseguirá funciones adicionales no disponibles en la biblioteca de funciones estándar de XPath, XQuery y XSLT. Las funciones de extensión de Altova están en el **espacio de nombres** **http://www.altova.com/xslt-extensions** y en esta sección se presentan con el prefijo, que se supone estará enlazado al espacio de nombres señalado. Tenga en cuenta que en futuras versiones del producto algunas funciones pueden dejar de ser compatibles o su comportamiento puede cambiar. Por tanto, consulte siempre la documentación del producto para conocer el funcionamiento de estas funciones en cada versión del producto.

Funciones XPath (en expresiones XPath en XSLT):	XP1 XP2 XP3.1.1
Funciones XSLT (en expresiones XPath en XSLT):	XSLT1 XSLT2 XSLT3
Funciones XQuery (en expresiones XQuery en XQuery):	XQ1 XQ3.1

camel-case [altova:]

altova:camel-case(CadenaEntrada *como xs:string***)** como **xs:string** XP3.1 XQ3.1
Devuelve la cadena de entrada **CadenaEntrada** escrita en CamelCase. La cadena se analiza usando la expresión regular **'\s'** (que es la forma abreviada del carácter espacio en blanco). El primer carácter que no sea un espacio en blanco situado después de un espacio en blanco o de una secuencia de espacios en blanco consecutivos se pondrá en mayúsculas. El primer carácter de la cadena de salida se pondrá en mayúsculas.

Ejemplos

- **altova:camel-case**("max") devuelve Max
- **altova:camel-case**("max max") devuelve Max Max
- **altova:camel-case**("file01.xml") devuelve File01.xml
- **altova:camel-case**("file01.xml file02.xml") devuelve File01.xml File02.xml
- **altova:camel-case**("file01.xml file02.xml") devuelve File01.xml File02.xml
- **altova:camel-case**("file01.xml -file02.xml") devuelve File01.xml -file02.xml

altova:camel-case(CadenaEntrada *como xs:string*, **CaracteresDivisión** *como xs:string*, **EsExpReg** *como xs:boolean***)** como **xs:string** XP3.1 XQ3.1
Devuelve la cadena de entrada **CadenaEntrada** escrita en CamelCase usando los **CaracteresDivisón** para determinar qué caracteres desencadenan el siguiente uso de mayúsculas. El argumento **CaracteresDivisón** se usa como expresión regular **cuando EsExpReg = true()** o como caracteres planos cuando r **EsExpReg = false()**. El primer carácter de la cadensa de salida se escribe con mayúsculas.

Ejemplos

- **altova:camel-case**("setname getname", "set|get", true()) devuelve setName getName
- **altova:camel-case**("altova\documents\testcases", "\", false()) devuelve Altova\Documents\Testcases

char [altova:]

altova:char(Posición *as xs:integer***)** como **xs:string** XP3.1 XQ3.1
Devuelve una cadena que contiene el carácter que está en la posición indicada por el argumento **Posición** en la cadena que se obtiene al convertir el valor del elemento de contexto en xs:string. La cadena resultante estará vacía si en la posición indicada no existe ningún carácter.

Ejemplos

Si el elemento de contexto es **1234ABCD**:

- **altova:char**(2) devuelve 2
- **altova:char**(5) devuelve A
- **altova:char**(9) devuelve la cadena vacía
- **altova:char**(-2) devuelve la cadena vacía

altova:char(CadenaEntrada *as xs:string***, Posición** *as xs:integer***)** como **xs:string** XP3.1 XQ3.1
Devuelve una cadena que contiene el carácter que está en la posición indicada por el argumento **Posición** en la cadena dada por el argumento **CadenaEntrada**. La cadena resultante estará vacía si en la posición indicada no existe ningún carácter.

Ejemplos

- **altova:char**("2014-01-15", 5) devuelve -
- **altova:char**("USA", 1) devuelve U
- **altova:char**("USA", 1) devuelve la cadena vacía
- **altova:char**("USA", -2) devuelve la cadena vacía

create-hash-from-string[altova:]

altova:create-hash-from-string(InputString *como xs:string***)** como **xs:string** XP2 XQ1 XP3.1 XQ3.1
altova:create-hash-from-string(InputString *como xs:string***, HashAlgo** *as xs:string***)** como **xs:string** XP2 XQ1 XP3.1 XQ3.1
Genera una cadena hash a partir de InputString usando el algoritmo de hash especificado por el argumento HashAlgo. Se pueden usar los siguientes algoritmos de hash (en mayúsculas o minúsculas): **MD5**, **SHA-1**, **SHA-224**, **SHA-256**, **SHA-384**, **SHA-512**. Si no se especifica el segundo argumento (*véase la primera instrucción*) se usa el algoritmo de hash **SHA-256**.

Ejemplos

- **altova:create-hash-from-string**('abc') devuelve una cadena hash generada usando el algoritmo de hash **SHA-256**.

© 2016-2022 Altova GmbH

- **altova:create-hash-from-string**('abc', 'md5') devuelve una cadena hash generada usando el algoritmo de hash **MD5**.
- **altova:create-hash-from-string**('abc', 'MD5') devuelve una cadena hash generada usando el algoritmo de hash **MD5**.

first-chars [altova:]

altova:first-chars(X *as xs:integer***)** como **xs:string** XP3.1 XQ3.1
Devuelve una cadena que contiene los **X** primeros caracteres de la cadena que se obtiene al convertir el valor del elemento de contexto en xs:string.

Ejemplos

Si el elemento de contexto es **1234ABCD**:

- **altova:first-chars**(2) devuelve 12
- **altova:first-chars**(5) devuelve 1234A
- **altova:first-chars**(9) devuelve 1234ABCD

altova:first-chars(CadenaEntrada *as xs:string*, **X** *as xs:integer***)** como **xs:string** XP3.1 XQ3.1
Devuelve una cadena que contiene los **X** primeros caracteres de la cadena dada como argumento **CadenaEntrada**.

Ejemplos

- **altova:first-chars**("2014-01-15", 5) devuelve 2014-
- **altova:first-chars**("USA", 1) devuelve U

format-string [altova:]

altova:format-string(InputString *como xs:string*, **FormatSequence** *como item()**) como **xs:string** XP3.1 XQ3.1
La cadena de entrada (primer argumento) contiene parámetros posicionales (%1, %2, etc). Cada parámetro es reemplazado por el elemento cadena ubicado en la posición correspondiente de la secuencia de formato (enviada como segundo argumento). Por tanto, el primer elemento de la secuencia de formato reemplaza al parámetro posicional %1, el segundo elemento reemplaza a %2 y así sucesivamente. La función devuelve esta secuencia con formato que contiene los elementos de reemplazo. Si no existe una cadena para alguno de los parámetros posicionales, entonces se devuelve ese mismo parámetro posicional. Esto ocurre cuando el índice de un parámetro posicional es mayor que el número de elementos de la secuencia de formato.

Ejemplos

- **altova:format-string**('Hello %1, %2, %3', ('Jane','John','Joe')) devuelve "Hello Jane, John, Joe"
- **altova:format-string**('Hello %1, %2, %3', ('Jane','John','Joe', 'Tom')) devuelve "Hello Jane, John, Joe"
- **altova:format-string**('Hello %1, %2, %4', ('Jane','John','Joe', 'Tom')) devuelve "Hello Jane, John, Tom"
- **altova:format-string**('Hello %1, %2, %4', ('Jane','John','Joe')) devuelve "Hello

```
Jane, John, %4"
```

last-chars [altova:]

altova:last-chars(X *as xs:integer*) como **xs:string** XP3.1 XQ3.1
Devuelve una cadena que contiene los X últimos caracteres de la cadena que se obtiene al convertir el valor del elemento de contexto en `xs:string`.

Ejemplos

Si el elemento de contexto es **1234ABCD**:

- **altova:last-chars**(2) devuelve CD
- **altova:last-chars**(5) devuelve 4ABCD
- **altova:last-chars**(9) devuelve 1234ABCD

altova:last-chars(CadenaEntrada *as xs:string*, **X** *as xs:integer*) como **xs:string** XP3.1 XQ3.1
Devuelve una cadena que contiene los **X** últimos caracteres de la cadena dada como argumento **CadenaEntrada**.

Ejemplos

- **altova:last-chars**("2014-01-15", 5) devuelve 01-15-
- **altova:last-chars**("USA", 10) devuelve USA

pad-string-left [altova:]

altova:pad-string-left(CadenaParaRellenar *como xs:string*, **LongitudCadena** *como xs:integer*, **CarácterRelleno** *como xs:string*) como **xs:string** XP3.1 XQ3.1
El argumento `CarácterRelleno` es un solo carácter. Se añade a la izquierda de la cadena para aumentar el número de caracteres de la `CadenaParaRellenar`, de modo que este número equivalga al valor entero del argumento `LongitudCadena`. El argumento `LongitudCadena` puede tener cualquier valor entero (positivo o negativo), pero el relleno solo se lleva a cabo si el valor de `LongitudCadena` es mayor que el número de caracteres de `CadenaParaRellenar`. Si `CadenaParaRellenar` tiene más caracteres que el valor de `LongitudCadena`, entonces `CadenaParaRellenar` se deja como está.

Ejemplos

- **altova:pad-string-left**('AP', 1, 'Z') devuelve 'AP'
- **altova:pad-string-left**('AP', 2, 'Z') devuelve 'AP'
- **altova:pad-string-left**('AP', 3, 'Z') devuelve 'ZAP'
- **altova:pad-string-left**('AP', 4, 'Z') devuelve 'ZZAP'
- **altova:pad-string-left**('AP', -3, 'Z') devuelve 'AP'
- **altova:pad-string-left**('AP', 3, 'YZ') devuelve un error indicando que el carácter de relleno es demasiado largo.

pad-string-right [altova:]

altova:pad-string-right(CadenaParaRellenar *como xs:string*, **LongitudCadena** *como*

xs:integer, **CarácterRelleno** *como xs:string*) como **xs:string** XP3.1 XQ3.1
El argumento `CarácterRelleno` es un solo carácter. Se añade a la derecha de la cadena para aumentar el número de caracteres de la `CadenaParaRellenar`, de modo que este número equivalga al valor entero del argumento `LongitudCadena`. El argumento `LongitudCadena` puede tener cualquier valor entero (positivo o negativo), pero el relleno solo se lleva a cabo si el valor de `LongitudCadena` es mayor que el número de caracteres de `CadenaParaRellenar`. Si `CadenaParaRellenar` tiene más caracteres que el valor de `LongitudCadena`, entonces `CadenaParaRellenar` se deja como está.

Ejemplos

- **altova:pad-string-right**('AP', 1, 'Z') devuelve 'AP'
- **altova:pad-string-right**('AP', 2, 'Z') devuelve 'AP'
- **altova:pad-string-right**('AP', 3, 'Z') devuelve 'APZ'
- **altova:pad-string-right**('AP', 4, 'Z') devuelve 'APZZ'
- **altova:pad-string-right**('AP', -3, 'Z') devuelve 'AP'
- **altova:pad-string-right**('AP', 3, 'YZ') devuelve un error indicando que el carácter de relleno es demasiado largo.

repeat-string [altova:]

altova:repeat-string(CadenaEntrada *as xs:string*, **Repeticiones** *as xs:integer*) como **xs:string** XP2 XQ1 XP3.1 XQ3.1
Genera una cadena que está compuesta por el primer argumento `CadenaEntrada` repetida tantas veces como indique el argumento `Repeticiones`.

Ejemplo

- **altova:repeat-string**("Altova #", 3)
  devuelve Altova #Altova #Altova #"

substring-after-last [altova:]

altova:substring-after-last(CadenaPrincipal *as xs:string*, **CadenaPrueba** *as xs:string*) como **xs:string** XP3.1 XQ3.1
Si **CadenaPrueba** se encuentra en **CadenaPrincipal**, la función devuelve la subcadena que aparece después de **CadenaPrueba** en **CadenaPrincipal**. Si **CadenaPrueba** no está en **CadenaPrincipal**, entonces devuelve la cadena vacía. Si **CadenaPrueba** es una cadena vacía, entonces devuelve la **CadenaPrincipal** entera. Si **CadenaPrueba** aparece varias veces en **CadenaPrincipal**, la función devuelve la subcadena que aparece después de la última **CadenaPrueba**.

Ejemplos

- **altova:substring-after-last**('ABCDEFGH', 'B') devuelve 'CDEFGH'
- **altova:substring-after-last**('ABCDEFGH', 'BC') devuelve 'DEFGH'
- **altova:substring-after-last**('ABCDEFGH', 'BD') devuelve ''
- **altova:substring-after-last**('ABCDEFGH', 'Z') devuelve ''
- **altova:substring-after-last**('ABCDEFGH', '') devuelve 'ABCDEFGH'
- **altova:substring-after-last**('ABCD-ABCD', 'B') devuelve 'CD'
- **altova:substring-after-last**('ABCD-ABCD-ABCD', 'BCD') devuelve ''

substring-before-last [altova:]

altova:substring-before-last(CadenaPrincipal *as xs:string*, **CadenaPrueba** *as xs:string*) como **xs:string** XP3.1 XQ3.1

Si **CadenaPrueba** se encuentra en **CadenaPrincipal**, la función devuelve la subcadena que aparece después de **CadenaPrueba** en **CadenaPrincipal**. Si **CadenaPrueba** no está en **CadenaPrincipal**, entonces devuelve la cadena vacía. Si **CadenaPrueba** es una cadena vacía, entonces devuelve la **CadenaPrincipal** entera. Si **CadenaPrueba** aparece varias veces en **CadenaPrincipal**, la función devuelve la subcadena que aparece antes de la última **CadenaPrueba**.

Ejemplos

- **altova:substring-before-last**('ABCDEFGH', 'B') devuelve 'A'
- **altova:substring-before-last**('ABCDEFGH', 'BC') devuelve 'A'
- **altova:substring-before-last**('ABCDEFGH', 'BD') devuelve ''
- **altova:substring-before-last**('ABCDEFGH', 'Z') devuelve ''
- **altova:substring-before-last**('ABCDEFGH', '') devuelve ''
- **altova:substring-before-last**('ABCD-ABCD', 'B') devuelve 'ABCD-A'
- **altova:substring-before-last**('ABCD-ABCD-ABCD', 'ABCD') devuelve 'ABCD-ABCD-'

substring-pos [altova:]

altova:substring-pos(Cadena *as xs:string*, **CadenaBúsqueda** *as xs:string*) como **xs:integer** XP3.1 XQ3.1

Devuelve la posición de carácter de la primera instancia de **CadenaBúsqueda** en **Cadena**. La posición de carácter se devuelve como número entero. El primer carácter de **CadenaBúsqueda** tiene la posición 1. Si **CadenaBúsqueda** no aparece dentro de **Cadena**, la función devuelve el entero 0. Para buscar la segunda instancia de **CadenaBúsqueda**, etc. use la otra firma de esta función.

Ejemplos

- **altova:substring-pos**('Altova', 'to') devuelve 3
- **altova:substring-pos**('Altova', 'tov') devuelve 3
- **altova:substring-pos**('Altova', 'tv') devuelve 0
- **altova:substring-pos**('AltovaAltova', 'to') devuelve 3

altova:substring-pos(Cadena *as xs:string*, **CadenaBúsqueda** *as xs:string*, **Entero** *as xs:integer*) como **xs:integer** XP3.1 XQ3.1

Devuelve la posición de carácter de **CadenaBúsqueda** en **Cadena**. La búsqueda de **CadenaBúsqueda** empieza en la posición de carácter dada por el argumento **Entero** (es decir, no se busca en la subcadena anterior a esta posición). El entero devuelto, sin embargo, es la posición que la cadena encontrada tiene en **Cadena**. Esta firma es muy práctica si quiere buscar la segunda posición, etc. de una cadena que aparece varias veces dentro de **Cadena**. Si **CadenaBúsqueda** no aparece en **Cadena**, la función devuelve el entero 0.

Ejemplos

- **altova:substring-pos**('Altova', 'to', 1) devuelve 3
- **altova:substring-pos**('Altova', 'to', 3) devuelve 3
- **altova:substring-pos**('Altova', 'to', 4) devuelve 0
- **altova:substring-pos**('Altova-Altova', 'to', 0) devuelve 3
- **altova:substring-pos**('Altova-Altova', 'to', 4) devuelve 10

© 2016-2022 Altova GmbH

substring-pos [altova:]

altova:substring-pos(Cadena *as xs:string*, **CadenaBúsqueda** *as xs:string*) como **xs:integer** XP3.1 XQ3.1

Devuelve la posición de carácter de la primera instancia de **CadenaBúsqueda** en **Cadena**. La posición de carácter se devuelve como número entero. El primer carácter de **CadenaBúsqueda** tiene la posición 1. Si **CadenaBúsqueda** no aparece dentro de **Cadena**, la función devuelve el entero `0`. Para buscar la segunda instancia de **CadenaBúsqueda**, etc. use la otra firma de esta función.

Ejemplos

- `altova:substring-pos('Altova', 'to')` devuelve `3`
- `altova:substring-pos('Altova', 'tov')` devuelve `3`
- `altova:substring-pos('Altova', 'tv')` devuelve `0`
- `altova:substring-pos('AltovaAltova', 'to')` devuelve `3`

altova:substring-pos(Cadena *as xs:string*, **CadenaBúsqueda** *as xs:string*, **Entero** *as xs:integer*) como **xs:integer** XP3.1 XQ3.1

Devuelve la posición de carácter de **CadenaBúsqueda** en **Cadena**. La búsqueda de **CadenaBúsqueda** empieza en la posición de carácter dada por el argumento **Entero** (es decir, no se busca en la subcadena anterior a esta posición). El entero devuelto, sin embargo, es la posición que la cadena encontrada tiene en **Cadena**. Esta firma es muy práctica si quiere buscar la segunda posición, etc. de una cadena que aparece varias veces dentro de **Cadena**. Si **CadenaBúsqueda** no aparece en **Cadena**, la función devuelve el entero `0`.

Ejemplos

- `altova:substring-pos('Altova', 'to', 1)` devuelve `3`
- `altova:substring-pos('Altova', 'to', 3)` devuelve `3`
- `altova:substring-pos('Altova', 'to', 4)` devuelve `0`
- `altova:substring-pos('Altova-Altova', 'to', 0)` devuelve `3`
- `altova:substring-pos('Altova-Altova', 'to', 4)` devuelve `10`

trim-string [altova:]

altova:trim-string(CadenaEntrada *as xs:string*) como **xs:string** XP3.1 XQ3.1

Esta función toma un argumento `xs:string`, quita los espacios en blanco iniciales y finales y devuelve un `xs:string` "recortado".

Ejemplos

- `altova:trim-string("  Hello World  ")` devuelve `"Hello World"`
- `altova:trim-string("Hello World  ")` devuelve `"Hello World"`
- `altova:trim-string("  Hello World")` devuelve `"Hello World"`
- `altova:trim-string("Hello World")` devuelve `"Hello World"`
- `altova:trim-string("Hello  World")` devuelve `"Hello  World"`

trim-string-left [altova:]

altova:trim-string-left(CadenaEntrada *as xs:string*) como **xs:string** XP3.1 XQ3.1

Esta función toma un argumento `xs:string`, quita los espacios en blanco iniciales y devuelve un `xs:string` recortado por la izquierda.

Ejemplos

- `altova:trim-string-left("  Hello World  ")` devuelve `"Hello World  "`
- `altova:trim-string-left("Hello World  ")` devuelve `"Hello World  "`
- `altova:trim-string-left("  Hello World")` devuelve `"Hello World"`
- `altova:trim-string-left("Hello World")` devuelve `"Hello World"`
- `altova:trim-string-left("Hello  World")` devuelve `"Hello  World"`

trim-string-right [altova:]

altova:trim-string-right(CadenaEntrada *as xs:string***)** como **xs:string** XP3.1 XQ3.1
Esta función toma un argumento `xs:string`, quita los espacios en blanco finales y devuelve una cadena `xs:string` recortada por la derecha.

Ejemplos

- `altova:trim-string-right("  Hello World  ")` devuelve `"  Hello World"`
- `altova:trim-string-right("Hello World  ")` devuelve `"Hello World"`
- `altova:trim-string-right("  Hello World")` devuelve `"  Hello World"`
- `altova:trim-string-right("Hello World")` devuelve `"Hello World"`
- `altova:trim-string-right("Hello  World")` devuelve `"Hello  World"`

18.2.2.1.9 Funciones XPath/XQuery varias

Estas funciones de extensión XPath/XQuery generales son compatibles con la versión actual de MapForce y se pueden usar en (i) expresiones XPath en contextos XSLT o (ii) en expresiones XQuery en documentos XQuery.

Nota sobre el nombre de las funciones y lenguajes

Puede utilizar todas las funciones de extensión de Altova en sus expresiones XPath/XQuery. Con ellas conseguirá funciones adicionales no disponibles en la biblioteca de funciones estándar de XPath, XQuery y XSLT. Las funciones de extensión de Altova están en el **espacio de nombres** **http://www.altova.com/xslt-extensions** y en esta sección se presentan con el prefijo, que se supone estará enlazado al espacio de nombres señalado. Tenga en cuenta que en futuras versiones del producto algunas funciones pueden dejar de ser compatibles o su comportamiento puede cambiar. Por tanto, consulte siempre la documentación del producto para conocer el funcionamiento de estas funciones en cada versión del producto.

Funciones XPath (en expresiones XPath en XSLT):	XP1 XP2 XP3.1.1
Funciones XSLT (en expresiones XPath en XSLT):	XSLT1 XSLT2 XSLT3
Funciones XQuery (en expresiones XQuery en XQuery):	XQ1 XQ3.1

Clic para expandir/contraer

© 2016-2022 Altova GmbH

altova:decode-string(Input *as xs:base64Binary*) **como xs:string** XP3.1 XQ3.1
altova:decode-string(Input *as xs:base64Binary*, **Encoding como** *xs:string*) **como xs:string** XP3.1 XQ3.1
Descifra la entrada en base64Binary en una cadena con el cifrado que se indique. Si no se indica ninguno se usa UTF-8. Estos son los cifrados compatibles: `US-ASCII, ISO-8859-1, UTF-16, UTF-16LE, UTF-16BE, ISO-10646-UCS2, UTF-32, UTF-32LE, UTF-32BE, ISO-10646-UCS4`

Ejemplos

- **altova:decode-string**(`$XML1/MailData/Meta/b64B`) devuelve la entrada en base64Binary como cadena de texto cifrada en UTF-8
- **altova:decode-string**(`$XML1/MailData/Meta/b64B, "UTF-8"`) devuelve la entrada en base64Binary como cadena de texto cifrada en UTF-8
- **altova:decode-string**(`$XML1/MailData/Meta/b64B, "ISO-8859-1"`) devuelve la entrada en base64Binary como una cadena de texto cifrada en ISO-8859-1

encode-string [altova:]

altova:encode-string(InputString *como xs:string*) **como xs:base64Binaryinteger** XP3.1 XQ3.1
altova:encode-string(InputString *como xs:string*, **Encoding** *como xs:string*) **como xs:base64Binaryinteger** XP3.1 XQ3.1
Cifra una cadena de texto usando el cifrado que se indique. Si no se indica ninguno, entonces se usa UTF-8. La cadena cifrada se convierte en caracteres base64Binary y se devuelve el valor base64Binary convertido. De momento se admite UTF-8, pero ampliaremos la compatibilidad a: `US-ASCII, ISO-8859-1, UTF-16, UTF-16LE, UTF-16BE, ISO-10646-UCS2, UTF-32, UTF-32LE, UTF-32BE, ISO-10646-UCS4`

Ejemplos

- **altova:encode-string**(`"Altova"`) devuelve el equivalente en base64Binary de la cadena de texto cifrada en UTF-8 `"Altova"`
- **altova:encode-string**(`"Altova", "UTF-8"`) devuelve el equivalente en base64Binary de la cadena de texto cifrada en UTF-8 `"Altova"`

get-temp-folder [altova:]

altova:get-temp-folder() como `xs:string` XP2 XQ1 XP3.1 XQ3.1
Esta función no toma ningún argumento. Devuelve la ruta de acceso de la carpeta temporal del usuario actual.

Ejemplo

- **altova:get-temp-folder**()
  en un equipo Windows devuelve (más o menos) `C:\Usuarios\<Usuario>\AppData\Local\Temp\` como valor de tipo `xs:string`.

generate-guid [altova:]

altova:generate-guid() asxs:string XP2 XQ1 XP3.1 XQ3.1
Genera una cadena única de la interfaz gráfica del usuario.

Ejemplo

- **altova:generate-guid**() devuelve (por ejemplo) 85F971DA-17F3-4E4E-994E-99137873ACCD

high-res-timer [altova:]

altova:high-res-timer() como **xs:double** XP3.1 XQ3.1
Devuelve un valor de temporizador de alta resolución en segundos. La presencia de un temporizador de alta resolución en un sistema permite hacer mediciones de alta precisión si es necesario (por ejemplo, en animaciones y para precisar de forma exacta horas de ejecución de código). Esta función ofrece la resolución del temporizador de alta resolución del sistema.

Ejemplos

- **altova:high-res-timer**() devuelve algo como '1.16766146154566E6'

parse-html [altova:]

altova:parse-html(HTMLText *as xs:string***)** como **node()** XP3.1 XQ3.1
El argumento HTMLText es una cadena que contiene el texto de un documento HTML. La función crea una estructura HTML a partir de la cadena. La cadena enviada puede contener o no el elemento HTML. En ambos casos el elemento raíz de la estructura es un elemento llamado **HTML**. Asegúrese de que el código HTML de la cadena enviada es válido.

Ejemplos

- **altova:parse-html**("<html><head/><body><h1>Header</h1></body></html>") crea una estructura HTML a partir de la cadena enviada

sleep[altova:]

altova:sleep(Millisecs *como xs:integer***)** como **empty-sequence()** XP2 XQ1 XP3.1 XQ3.1
Suspende la ejecución de la operación actual durante el número de milisegundos dado por el argumento Millisecs.

Ejemplos

- **altova:sleep**(1000) suspende la ejecución de la operación actual durante 1000 millisegundos.

[Subir 1568]

18.2.2.2 Funciones de extensión varias

Los lenguajes de programación como Java y C# ofrecen varias funciones predefinidas que no están disponibles como funciones XQuery/XPath ni XSLT. Un ejemplo son las funciones matemáticas de Java sin() y cos(). Si los diseñadores de hojas de estilos XSLT y consultas XQuery tuvieran acceso a estas funciones, el área de aplicación de sus hojas de estilos y consultas aumentaría y su trabajo sería un poco más sencillo.

 © 2016-2022 Altova GmbH

Los motores XSLT y XQuery de los productos de Altova admiten el uso de funciones de extensión en Java[1571] y .NET[1580], así como scripts MSXSL para XSLT[1586].

Esta sección describe cómo usar funciones de extensión y scripts MSXSL en hojas de estilos XSLT y documentos XQuery. Las funciones de extensión pueden organizarse en varios grupos:

- Funciones de extensión Java[1571]
- Funciones de extensión .NET[1580]
- Scripts MSXSL para XSLT[1586]

En los apartados de esta sección nos ocupamos de tres aspectos fundamentales: (i) cómo se llaman las funciones en sus respectivas bibliotecas, (ii) qué reglas deben seguirse para convertir los argumentos de una llamada a función en el formato de entrada necesario de la función y (iii) qué reglas deben seguirse para la conversión del tipo devuelto.

Requisitos

Para que estas funciones de extensión funcionen es necesario tener Java Runtime Environment (para las funciones Java) y .NET Framework 2.0 o superior (para las funciones .NET) instalado en el equipo que ejecuta la transformación XSLT o XQuery.

18.2.2.2.1 Funciones de extensión Java

Puede usar una función de extensión Java dentro de una expresión XPath o XQuery para invocar un constructor Java o llamar a un método Java (estático o de instancia).

Un campo de una clase Java se trata como un método sin argumentos. Un campo puede ser estático o de instancia. Más adelante describimos cómo se accede a los campos estáticos y de instancia.

Este apartado tiene varias partes:

- Constructores Java[1577]
- Métodos estáticos y campos estáticos[1577]
- Métodos de instancia y campos de instancia[1578]
- Tipos de datos: conversión de XPath/XQuery en Java[1579]
- Tipos de datos: conversión de Java en XPath/XQuery[1580]

Tenga en cuenta que:

- si está usando un producto de escritorio de Altova, la aplicación intentará detectar automáticamente la ruta de acceso al equipo virtual Java; para ello leerá (en este orden): (i) el registro de Windows y (ii) la variable de entorno **JAVA_HOME**. También puede añadir una ruta personal en el cuadro de diálogo "Opciones" de la aplicación; esta ruta tendrá prioridad frente a cualquier otra ruta de acceso a un equipo virtual Java que se detecte automáticamente.
- si está usando un producto servidor de Altova en un equipo Windows, la ruta de acceso al equipo virtual Java se leerá primero desde el registro de Windows; si esto no ocurre se usa la variable de entorno **JAVA_HOME**.
- si está usando un producto servidor de Altova en un equipo Linux o macOS, entonces asegúrese de que la variable de entorno **JAVA_HOME** está definida correctamente y la biblioteca Java de equipos

virtuales (en Windows, el archivo **jvm.dll**) se encuentra en uno de estos directorios: **\bin\server** o **\bin\client**.

Formato de la función de extensión

La función de extensión de la expresión XPath/XQuery debe tener este formato `prefijo:nombreFunción()`.

- La parte `prefijo:` identifica la función de extensión como función Java. Lo hace asociando la función de extensión con una declaración de espacio de nombres del ámbito, cuyo URI debe empezar por `java:` (*ver ejemplos más abajo*). La declaración de espacio de nombres debe identificar una clase Java, por ejemplo: `xmlns:myns="java:java.lang.Math"`. Sin embargo, también puede ser simplemente: `xmlns:myns="java"` (sin los dos puntos), dejando la identificación de la clase Java a la parte `nombreFunción()` de la función de extensión.
- La parte `nombreFunción()`identifica el método Java al que se llama y presenta los argumentos para el método (*ver ejemplos más abajo*). Sin embargo, si el URI de espacio de nombres identificado por la parte `prefijo:` no identifica una clase Java (*ver punto anterior*), entonces la clase Java debe identificarse en la parte `nombreFunción()`, antes de la clase y separada de la clase por un punto (*ver el segundo ejemplo XSLT que aparece más abajo*).

Nota: la clase a la que se llama debe estar en la ruta de acceso de clase del equipo.

Ejemplo de código XSLT

Aquí ofrecemos dos ejemplos de cómo se puede llamar a un método estático. En el primer ejemplo, el nombre de la clase (`java.lang.Math`) se incluye en el URI de espacio de nombres y, por tanto, no puede estar en la parte `nombreFunción()`. En el segundo ejemplo, la parte `prefijo:` presenta el prefijo `java:` mientras que la parte `nombreFunción()` identifica la clase y el método.

```
<xsl:value-of xmlns:jMath="java:java.lang.Math"
              select="jMath:cos(3.14)" />

<xsl:value-of xmlns:jmath="java"
              select="jmath:java.lang.Math.cos(3.14)" />
```

El método nombrado en la función de extensión (`cos()`) debe coincidir con el nombre de un método estático público de la clase Java nombrada (`java.lang.Math`).

Ejemplo de código XQuery

Aquí puede ver un ejemplo de código XQuery similar al código XSLT anterior:

```
<cosine xmlns:jMath="java:java.lang.Math">
   {jMath:cos(3.14)}
</cosine>
```

Clases Java definidas por el usuario

Si creó sus propias clases Java, a los métodos de estas clases se les llama de otra manera, dependiendo de: (i) si a las clases se accede por medio de un archivo JAR o de un archivo de clases y (ii) si estos archivos están en el directorio actual (el directorio del documento XSLT o XQuery). Para más información consulte los

 © 2016-2022 Altova GmbH

apartados Archivos de clases definidos por el usuario[1573] y Archivos Jar definidos por el usuario[1576]. Recuerde que debe especificar las rutas de acceso de los archivos de clases que no están en el directorio actual y de todos los archivos JAR.

18.2.2.2.1.1 Archivos de clases definidos por el usuario

Si se accede a las clases por medio de un archivo de clases, entonces hay cuatro posibilidades:

- El archivo de clases está en un paquete. El archivo XSLT/XQuery está en la misma carpeta que el paquete Java. (*ver ejemplo*[1573].)
- El archivo de clases no está en un paquete. El archivo XSLT/XQuery está en la misma carpeta que el archivo de clases. (*ver ejemplo*[1574].)
- El archivo de clases está en un paquete. El archivo XSLT/XQuery está en una carpeta cualquiera. (*ver ejemplo*[1574].)
- El archivo de clases no está en un paquete. El archivo XSLT/XQuery está una carpeta cualquiera. (*ver ejemplo*[1575].)

Imaginemos que tenemos un archivo de clases que no está en un paquete y que está en la misma carpeta que el documento XSLT/XQuery. En este caso, puesto que en la carpeta se encuentran todas las clases, no es necesario especificar la ubicación del archivo. La sintaxis que se utiliza para identificar una clase es esta:

```
java:nombreClase
```

donde

`java:` indica que se está llamando a una función definida por el usuario (por defecto se cargan las clases Java del directorio actual)

`nombreClase` es el nombre de la clase del método elegido

La clase se identifica en un URI de espacio de nombres y el espacio de nombres se usa como prefijo para la llamada al método.

El archivo de clases está en un paquete. El archivo XSLT/XQuery está en la misma carpeta que el paquete Java

El código que aparece a continuación llama al método `getVehicleType()` de la clase `Car` del paquete `com.altova.extfunc`. El paquete `com.altova.extfunc` está en la carpeta `JavaProject`. El archivo XSLT también está en la carpeta `JavaProject`.

```
<xsl:stylesheet version="2.0"
       xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
       xmlns:xs="http://www.w3.org/2001/XMLSchema"
       xmlns:fn="http://www.w3.org/2005/xpath-functions"
       xmlns:car="java:com.altova.extfunc.Car" >
<xsl:output exclude-result-prefixes="fn car xsl fo xs"/>

<xsl:template match="/">
   <a>
   <xsl:value-of select="car:getVehicleType()"/>
```

```
   </a>
</xsl:template>

</xsl:stylesheet>
```

El archivo de clases está referenciado. El archivo XSLT/XQuery está en la misma carpeta que el archivo de clases

El código que aparece a continuación llama al método `getVehicleType()` de la clase `Car`. Digamos que: (i) el archivo de clases `Car` está en esta carpeta: `JavaProject/com/altova/extfunc` y que (ii) esa carpeta es la del ejemplo siguiente. El archivo XSLT también está en la carpeta `JavaProject/com/altova/extfunc`.

```
<xsl:stylesheet version="2.0"
       xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
       xmlns:xs="http://www.w3.org/2001/XMLSchema"
       xmlns:fn="http://www.w3.org/2005/xpath-functions"
       xmlns:car="java:Car" >
<xsl:output exclude-result-prefixes="fn car xsl fo xs"/>

<xsl:template match="/">
   <a>
   <xsl:value-of select="car:getVehicleType()"/>
   </a>
</xsl:template>

</xsl:stylesheet>
```

El archivo de clases está en un paquete. El archivo XSLT/XQuery está en una carpeta cualquiera

El código que aparece a continuación llama al método `getCarColor()` de la clase `Car` del paquete `com.altova.extfunc`. El paquete `com.altova.extfunc` está en la carpeta `JavaProject`. El archivo XSLT está en otra carpeta cualquiera. En este caso debe especificarse la ubicación del paquete dentro del URI como una cadena de consulta. La sintaxis es esta:

```
java:nombreClase[?ruta=uri-del-paquete]
```

donde

`java:` indica que se está llamando a una función Java definida por el usuario
`uri-del-paquete` es el URI del paquete Java
`nombreClase` es el nombre de la clase del método elegido

La clase se identifica en un URI de espacio de nombres y el espacio de nombres se usa como prefijo para la llamada al método. El ejemplo de código que aparece a continuación explica cómo se accede a un archivo de clases que está ubicado en un directorio que no es el directorio actual.

```
<xsl:stylesheet version="2.0"
       xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
```

 © 2016-2022 Altova GmbH

```
        xmlns:xs="http://www.w3.org/2001/XMLSchema"
        xmlns:fn="http://www.w3.org/2005/xpath-functions"
        xmlns:car="java:com.altova.extfunc.Car?path=file:///C:/JavaProject/" >

<xsl:output exclude-result-prefixes="fn car xsl xs"/>

<xsl:template match="/">
   <xsl:variable name="myCar" select="car:new('red')" />
   <a><xsl:value-of select="car:getCarColor($myCar)"/></a>
</xsl:template>

</xsl:stylesheet>
```

El archivo de clases no está en un paquete. El archivo XSLT/XQuery está una carpeta cualquiera

El código que aparece a continuación llama al método `getCarColor()`de la clase `Car`. Digamos que el archivo de clases `Car` está en la carpeta `C:/JavaProject/com/altova/extfunc` y que el archivo XSLT está en otra carpeta cualquiera. En este caso debe especificarse la ubicación del paquete dentro del URI como una cadena de consulta. La sintaxis es esta:

```
java:nombreClase[?ruta=<uri-del-archivoClases>]
```

donde

`java:` indica que se está llamando a una función Java definida por el usuario
`uri-del-archivoClases` es el URI de la carpeta donde se ubica el archivo de clases
`nombreClase` es el nombre de la clase del método elegido

La clase se identifica en un URI de espacio de nombres y el espacio de nombres se usa como prefijo para la llamada al método. El ejemplo de código que aparece a continuación explica cómo se accede a un archivo de clases que está ubicado en un directorio que no es el directorio actual.

```
<xsl:stylesheet version="2.0"
        xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
        xmlns:xs="http://www.w3.org/2001/XMLSchema"
        xmlns:fn="http://www.w3.org/2005/xpath-functions"
        xmlns:car="java:Car?path=file:///C:/JavaProject/com/altova/extfunc/" >

<xsl:output exclude-result-prefixes="fn car xsl xs"/>

<xsl:template match="/">
   <xsl:variable name="myCar" select="car:new('red')" />
   <a><xsl:value-of select="car:getCarColor($myCar)"/></a>
</xsl:template>

</xsl:stylesheet>
```

Nota: cuando se presenta una ruta de acceso por medio de una función de extensión, la ruta de acceso se añade al ClassLoader.

18.2.2.2.1.2 Archivos JAR definidos por el usuario

Si se accede a las clases por medio de un archivo JAR, entonces se debe especificar el URI del archivo JAR usando esta sintaxis:

```
xmlns:claseEspacioNombres="java:nombreClase?ruta=jar:uri-del-archivoJar!/"
```

Para la llamada al método se usa el prefijo del URI de espacio de nombres que identifica la clase: `claseEspacioNombres:método()`

En la sintaxis anterior:

`java:` indica que se está llamando a una función de Java
`nombreClase` es el nombre de la clase definida por el usuario
`?` es el separador entre el nombre de la clase y la ruta de acceso
`ruta=jar:` indica que se ofrece una ruta de acceso a un archivo JAR
`uri-del-archivoJar` es el URI del archivo JAR
`!/` es el delimitador final de la ruta de acceso
`claseEspacioNombres:método()` es la llamada al método

Otra opción es dar el nombre de la clase con la llamada al método. Por ejemplo:

```
xmlns:ns1="java:docx.layout.pages?
path=jar:file:///c:/projects/docs/docx.jar!/"
ns1:main()

xmlns:ns2="java?path=jar:file:///c:/projects/docs/docx.jar!/"
ns2:docx.layout.pages.main()
```

Y aquí puede ver un ejemplo de XSLT que usa un archivo JAR para llamar a una función de extensión Java:

```
<xsl:stylesheet version="2.0"
       xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
       xmlns:xs="http://www.w3.org/2001/XMLSchema"
       xmlns:fn="http://www.w3.org/2005/xpath-functions"
       xmlns:car="java?path=jar:file:///C:/test/Car1.jar!/" >
<xsl:output exclude-result-prefixes="fn car xsl xs"/>

<xsl:template match="/">
       <xsl:variable name="myCar" select="car:Car1.new('red')" />
       <a><xsl:value-of select="car:Car1.getCarColor($myCar)"/></a>
</xsl:template>

<xsl:template match="car"/>

</xsl:stylesheet>
```

Nota: cuando se presenta una ruta de acceso por medio de una función de extensión, la ruta de acceso se añade al ClassLoader.

 © 2016-2022 Altova GmbH

18.2.2.2.1.3 Constructores

Una función de extensión se puede usar para llamar a un constructor Java. A todos los constructores se les llama con la pseudofunción `new()`.

Si el resultado de una llamada a un constructor Java se puede convertir de manera implícita a tipos de datos XPath/XQuery 1580, entonces la llamada a la función de extensión Java devuelve una secuencia que es un tipo de datos XPath/XQuery. Si el resultado de una llamada a un constructor Java no se puede convertir a un tipo de datos XPath/XQuery adecuado, entonces el constructor crea un objeto Java contenido con un tipo que es el nombre de la clase que devuelve ese objeto Java. Por ejemplo, si se llama a un constructor para la clase `java.util.Date` (`java.util.Date.new()`), entonces se devuelve un objeto que tiene el tipo `java.util.Date`. Puede que el formato léxico del objeto devuelto no coincida con el formato léxico de un tipo de datos XPath y, por tanto, su valor debe convertirse al formato léxico del tipo de datos XPath pertinente y después al tipo de datos XPath.

Puede hacer dos cosas con el objeto Java creado por un constructor:

- Puede asignar el objeto a una variable:
```
<xsl:variable name="currentdate" select="date:new()"
xmlns:date="java:java.util.Date" />
```
- Puede pasar el objeto a una función de extensión (*ver métodos de instancia y campos de instancia* 1578):
```
<xsl:value-of select="date:toString(date:new())" xmlns:date="java:java.util.Date" />
```

18.2.2.2.1.4 Métodos estáticos y campos estáticos

La llamada a un método estático la hace directamente su nombre Java y se hace presentando los argumentos para el método. A los campos estáticos (es decir, los métodos que no toman argumentos), como los campos de valor constante `E` y `PI`, se accede sin especificar ningún argumento.

Ejemplos de código XSLT

Aquí puede ver varios ejemplos de cómo se llama a métodos y campos estáticos:

```
<xsl:value-of xmlns:jMath="java:java.lang.Math"
              select="jMath:cos(3.14)" />

<xsl:value-of xmlns:jMath="java:java.lang.Math"
              select="jMath:cos( jMath:PI() )" />

<xsl:value-of xmlns:jMath="java:java.lang.Math"
              select="jMath:E() * jMath:cos(3.14)" />
```

Observe que las funciones de extensión anteriores tienen el formato `prefijo:nombreFunción()`. En los tres ejemplos anteriores, el prefijo es `jMath:`, que está asociado al URI de espacio de nombres `java:java.lang.Math`. (El URI de espacio de nombres debe empezar por `java:`. En los ejemplos anteriores se extiende para contener el nombre de la clase (`java.lang.Math`).) La parte `nombreFunción()` de las funciones de extensión debe coincidir con el nombre de una clase pública (p. ej. `java.lang.Math`) seguido del

nombre de un método estático público con sus argumentos (como `cos(3.14)`) o de un campo estático público (como `PI()`).

En los tres ejemplos anteriores, el nombre de la clase se incluyó en el URI de espacio de nombres. Si no estuviera en el URI de espacio de nombres, se incluiría en la parte `nombreFunción()` de la función de extensión. Por ejemplo:

```
<xsl:value-of xmlns:java="java:"
              select="java:java.lang.Math.cos(3.14)" />
```

Ejemplo de XQuery

Un ejemplo de XQuery similar sería:

```
<cosine xmlns:jMath="java:java.lang.Math">
   {jMath:cos(3.14)}
</cosine>
```

18.2.2.2.1.5 Métodos de instancia y campos de instancia

A un método de instancia se le pasa un objeto Java como primer argumento de la llamada a método. Dicho objeto Java suele crearse usando una función de extensión (por ejemplo, una llamada a un constructor) o un parámetro o una variable de hoja de estilos. Un ejemplo de código XSLT de este tipo sería:

```
<xsl:stylesheet version="1.0" exclude-result-prefixes="date"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
  xmlns:date="java:java.util.Date"
  xmlns:jlang="java:java.lang">
  <xsl:param name="CurrentDate" select="date:new()"/>
  <xsl:template match="/">
      <enrollment institution-id="Altova School"
                date="{date:toString($CurrentDate)}"
                type="{jlang:Object.toString(jlang:Object.getClass( date:new() ))}">
      </enrollment>
  </xsl:template>
</xsl:stylesheet>
```

En el ejemplo anterior el valor del nodo `enrollment/@type` se crea de la siguiente manera:

1. Se crea un objeto con un constructor para la clase `java.util.Date` (con el constructor `date:new()`).
2. Este objeto Java se pasa como argumento del método `jlang.Object.getClass`.
3. El objeto que obtiene el método `getClass` se pasa como argumento al método `jlang.Object.toString`.

El resultado (el valor de `@type`) será una cadena con este valor: `java.util.Date`.

En teoría, un campo de instancia es diferente de un método de instancia porque al campo de instancia no se pasa como argumento un objeto Java propiamente dicho. En su lugar se pasa como argumento un parámetro o variable. Sin embargo, el parámetro o la variable puede contener el valor devuelto por un objeto Java. Por ejemplo, el parámetro `CurrentDate` toma el valor que devolvió un constructor para la clase `java.util.Date`.

 © 2016-2022 Altova GmbH

Este valor se pasa después como argumento al método de instancia `date:toString` a fin de suministrar el valor de `/enrollment/@date`.

18.2.2.2.1.6 Tipos de datos: conversión de XPath/XQuery en Java

Cuando se llama a una función Java desde dentro de una expresión XPath/XQuery, el tipo de datos de los argumentos de la función es importante a la hora de determinar a cuál de las clases Java que tienen el mismo nombre se llama.

En Java se siguen estas reglas:

- Si hay más de un método Java con el mismo nombre, pero cada método tiene un número diferente de argumentos, entonces se selecciona el método Java que mejor se ajusta al número de argumentos de la llamada a función.
- Los tipos de datos de cadena, numéricos y booleanos de XPath/XQuery (*ver lista más abajo*) se convierten de forma implícita en el tipo de datos Java correspondiente. Si el tipo XPath/XQuery suministrado se puede convertir a más de un tipo Java (p. ej. `xs:integer`), entonces se selecciona el tipo Java que se declaró para el método seleccionado. Por ejemplo, si el método Java al que se llama es `fx(decimal)` y el tipo de datos XPath/XQuery suministrado es `xs:integer`, entonces `xs:integer` se convierte en el tipo de datos Java `decimal`.

La tabla que aparece a continuación enumera las conversiones implícitas de los tipos de cadena, numéricos y booleanos XPath/XQuery en tipos de datos Java.

`xs:string`	`java.lang.String`
`xs:boolean`	`boolean` (primitivo), `java.lang.Boolean`
`xs:integer`	int, long, short, byte, float, double y sus clases contenedoras, como `java.lang.Integer`
`xs:float`	`float` (primitivo), `java.lang.Float`, `double` (primitivo)
`xs:double`	`double` (primitivo), `java.lang.Double`
`xs:decimal`	`float` (primitivo), `java.lang.Float`, `double`(primitivo), `java.lang.Double`

Los subtipos de los tipos de datos XML Schema de la tabla anterior (que se usan en XPath y XQuery) también se convierten en los tipos Java correspondientes al tipo antecesor del subtipo.

En algunos casos quizás no sea posible seleccionar el método Java correcto usando la información dada. Por ejemplo, imagine que:

- El argumento presentado es un valor `xs:untypedAtomic` de 10 y está destinado al método `mimétodo(float)`.
- Sin embargo, hay otro método en la clase que toma un argumento de otro tipo de datos: `mimétodo(double)`.

- Puesto que los métodos tienen el mismo nombre y el tipo suministrado (`xs:untypedAtomic`) se puede convertir correctamente tanto en `float` como en `double`, es posible que `xs:untypedAtomic` se convierta en `double` en lugar de en `float`.
- Por consiguiente, el método seleccionado no será el método necesario y quizás no produzca el resultado esperado. Una solución es crear un método definido por el usuario con un nombre diferente y usar ese método.

Los tipos que no aparecen en la lista anterior (p. ej. `xs:date`) no se convertirán y generarán un error. No obstante, tenga en cuenta que en algunos casos, es posible crear el tipo Java necesario usando un constructor Java.

18.2.2.2.1.7 Tipos de datos: conversión de Java en XPath/XQuery

Cuando un método Java devuelve un valor y el tipo de datos del valor es un tipo de cadena, numérico o booleano, entonces se convierte en el tipo de datos XPath/XQuery correspondiente. Por ejemplo, los tipos de datos Java `java.lang.Boolean` y `boolean` se convierten en `xsd:boolean`.

Las matrices unidimensionales devueltas por las funciones se extienden en una secuencia. Las matrices multidimensionales no se convierten y, por tanto, deberían ser contenidas.

Cuando se devuelve un objeto Java contenido o un tipo de datos que no es de cadena, numérico ni booleano, puede garantizar la conversión del tipo XPath/XQuery necesario usando primero un método Java (p. ej. `toString`) para convertir el objeto Java en una cadena. En XPath/XQuery la cadena se puede modificar para ajustarse a la representación léxica del tipo necesario y convertirse después en dicho tipo (usando la expresión `cast as`, por ejemplo).

18.2.2.2.2 Funciones de extensión .NET

Si trabaja en la plataforma .NET desde un equipo Windows, puede usar funciones de extensión escritas en cualquier lenguaje .NET (p. ej. C#). Una función de extensión .NET se puede usar dentro de una expresión XPath/XQuery para invocar un constructor, una propiedad o un método (estático o de instancia) de una clase .NET.

A una propiedad de una clase .NET se le llama usando la sintaxis `get_NombrePropiedad()`.

Este apartado tiene varias partes:

- Constructores 1583
- Métodos estáticos y campos estáticos 1583
- Métodos de instancia y campos de instancia 1584
- Tipos de datos: conversión de XPath/XQuery en .NET 1585
- Tipos de datos: conversión de .NET en XPath/XQuery 1586

Formato de la función de extensión

La función de extensión de la expresión XPath/XQuery debe tener este formato `prefijo:nombreFunción()`.

© 2016-2022 Altova GmbH

- La parte `prefijo:` está asociada a un URI que identifica la clase .NET.
- La parte `nombreFunción()` identifica el constructor, la propiedad o el método (estático o de instancia) dentro de la clase .NET y, si es necesario, suministra los argumentos.
- El URI debe empezar por `clitype:` (que identifica la función como función de extensión .NET).
- El formato `prefijo:nombreFunción()` de la función de extensión se puede usar con clases del sistema y con clases de un ensamblado cargado. No obstante, si se tiene que cargar una clase, será necesario suministrar parámetros que contengan la información necesaria.

Parámetros

Para cargar un ensamblado se usan estos parámetros:

`asm`	El nombre del ensamblado que se debe cargar.
`ver`	El número de versión (máximo cuatro enteros separados por puntos).
`sn`	El símbolo de clave del nombre seguro del ensamblado (16 dígitos hexadecimales).
`from`	Un URI que da la ubicación del ensamblado (DLL) que se debe cargar. Si el URI es relativo, es relativo al archivo XSLT o XQuery. Si está presente este parámetro, se ignoran los demás parámetros.
`partialname`	El nombre parcial del ensamblado. Se suministra a `Assembly.LoadWith.PartialName()`, que intentará cargar el ensamblado. Si está presente el parámetro `partialname`, se ignoran los demás parámetros.
`loc`	La configuración regional, por ejemplo, `en-US`. La configuración predeterminada es `neutral`.

Si el ensamblado se debe cargar desde un archivo DLL, use el parámetro `from` y omita el parámetro `sn`. Si el ensamblado se debe cargar desde el caché general de ensamblados (GAC), use el parámetro `sn` y omita el parámetro `from`.

Debe insertar un signo de interrogación final antes del primer parámetro y los parámetros deben separarse con un punto y coma (;). El nombre de parámetro da su valor con un signo igual (=), como en el ejemplo que aparece más abajo.

Ejemplos de declaraciones de espacios de nombres

Esto es un ejemplo de una declaración de espacio de nombres en XSLT que identifica la clase del sistema `System.Environment`:

```
xmlns:myns="clitype:System.Environment"
```

Esto es un ejemplo de una declaración de espacio de nombres en XSLT que identifica la clase que se debe cargar como `Trade.Forward.Scrip`:

```
xmlns:myns="clitype:Trade.Forward.Scrip?asm=forward;version=10.6.2.1"
```

Esto es un ejemplo de una declaración de espacio de nombres en XQuery que identifica la clase del sistema `MyManagedDLL.testClass`:. Existen dos tipos de clases:

1. Cuando el ensamblado se carga desde el GAC:
```
declare namespace cs="clitype:MyManagedDLL.testClass?asm=MyManagedDLL;
       ver=1.2.3.4;loc=neutral;sn=b9f091b72dccfba8";
```

2. Cuando el ensamblado se carga desde el archivo DLL (ver las referencias parciales y completas):
```
       declare namespace cs="clitype:MyManagedDLL.testClass?from=file:///C:/Altova
       Projects/extFunctions/MyManagedDLL.dll;

declare namespace cs="clitype:MyManagedDLL.testClass?from=MyManagedDLL.dll;
```

Ejemplo de código XSLT

Aquí puede ver un ejemplo de código XSLT que llama a funciones de la clase del sistema `System.Math`:

```
<xsl:stylesheet version="2.0"
   xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
   xmlns:xs="http://www.w3.org/2001/XMLSchema"
   xmlns:fn="http://www.w3.org/2005/xpath-functions">
   <xsl:output method="xml" omit-xml-declaration="yes" />
   <xsl:template match="/">
      <math xmlns:math="clitype:System.Math">
         <sqrt><xsl:value-of select="math:Sqrt(9)"/></sqrt>
         <pi><xsl:value-of select="math:PI()"/></pi>
         <e><xsl:value-of select="math:E()"/></e>
         <pow><xsl:value-of select="math:Pow(math:PI(), math:E())"/></pow>
      </math>
   </xsl:template>
</xsl:stylesheet>
```

La declaración de espacio de nombres del elemento `math` asocia el prefijo `math:` al URI `clitype:System.Math`. La parte inicial `clitype:` del URI indica que lo que sigue identifica una clase del sistema o una clase cargada. El prefijo `math:` de las expresiones XPath asocia las funciones de extensión al URI (y, por extensión, a la clase) `System.Math`. Las funciones de extensión identifican métodos en la clase `System.Math` y presenta argumentos cuando es necesario.

Ejemplo de código XQuery

Aquí puede ver un fragmento de código XQuery similar al ejemplo anterior:

```
<math xmlns:math="clitype:System.Math">
   {math:Sqrt(9)}
</math>
```

Tal y como ocurre con el código XSLT anterior, la declaración de espacio de nombres identifica la clase .NET, en este caso una clase del sistema. La expresión XQuery identifica el método al que se debe llamar y presenta el argumento.

© 2016-2022 Altova GmbH

18.2.2.2.2.1 Constructores

Una función de extensión se puede usar para llamar a un constructor .NET. A todos los constructores se les llama con la pseudofunción `new()`. Si hay más de un constructor para una clase, entonces se selecciona el constructor que más se ajusta al número de argumentos suministrados. Si no se encuentra ningún constructor que coincida con los argumentos suministrados, entonces se genera el error `"No constructor found"`.

Constructores que devuelven tipos de datos XPath/XQuery

Si el resultado de una llamada a un constructor .NET se puede convertir de forma implícita en tipos de datos XPath/XQuery [1580], entonces la función de extensión .NET devuelve una secuencia que es un tipo de datos XPath/XQuery.

Constructores que devuelven objetos .NET

Si el resultado de una llamada a un constructor .NET no se puede convertir a un tipo de datos XPath/XQuery adecuado, entonces el constructr crea un objeto .NET contenido con un tipo que es el nombre de la clase que devuelve dicho objeto. Por ejemplo, si se llama al constructor para la clase `System.DateTime` (con `System.DateTime.new()`), entonces se devuelve un objeto que tiene un tipo `System.DateTime`.

Puede que el formato léxico del objeto devuelto no coincida con el formato léxico de un tipo de datos XPath. En estos casos, el valor devuelto (i) debe convertirse al formato léxico del tipo de datos XPath pertinente y (ii) debe convertirse en el tipo de datos XPath necesario.

Se pueden hacer tres cosas con un objeto .NET creado con un constructor:

- Se puede usar dentro de una variable:
```
<xsl:variable name="currentdate" select="date:new(2008, 4, 29)"
xmlns:date="clitype:System.DateTime" />
```
- Se puede pasar a una función de extensión (ver *Métodos de instancia y campos de instancia* [1578]):
```
<xsl:value-of select="date:ToString(date:new(2008, 4, 29))"
xmlns:date="clitype:System.DateTime" />
```
- Se puede convertir en un tipo de cadena, numérico o booleano:
```
<xsl:value-of select="xs:integer(date:get_Month(date:new(2008, 4, 29)))"
xmlns:date="clitype:System.DateTime" />
```

18.2.2.2.2.2 Metodos estáticos y campos estáticos

La llamada a un método estático la hace directamente su nombre y se hace presentando los argumentos para el método. El nombre usado en la llamada debe ser el mismo que un método estático público de la clase especificada. Si el nombre del método y el número de argumentos que se dio en la llamada a función coincide con algún método de la clase, entonces los tipos de los argumentos presentados se evalúan para encontrar el resultado ideal. Si no se encuentra ninguna coincidencia, se emite un error.

Nota: un campo de una clase .NET se trata como si fuera un método sin argumentos. Para llamar a una propiedad se usa la sintaxis `get_nombrePropiedad()`.

Ejemplos

Este ejemplo de código XSLT muestra una llamada a un método con un argumento (`System.Math.Sin(arg)`):

```
<xsl:value-of select="math:Sin(30)" xmlns:math="clitype:System.Math"/>
```

Este ejemplo de código XSLT muestra una llamada a un campo (que se trata como si fuera un método sin argumentos) (`System.Double.MaxValue()`):

```
<xsl:value-of select="double:MaxValue()" xmlns:double="clitype:System.Double"/>
```

Este ejemplo de código XSLT muestra una llamada a una propiedad (la sintaxis es `get_nombrePropiedad()`) (`System.String()`):

```
<xsl:value-of select="string:get_Length('my string')"
xmlns:string="clitype:System.String"/>
```

Este ejemplo de código XQuery muestra una llamada a un método con un argumento (`System.Math.Sin(arg)`):

```
<sin xmlns:math="clitype:System.Math">
   { math:Sin(30) }
</sin>
```

18.2.2.2.2.3 Métodos de instancia y campos de instancia

Un método de instancia es un método al que se le pasa un objeto .NET como primer argumento de la llamada al método. Este objeto .NET se suele crear usando una función de extensión (por ejemplo, una llamada a un constructor) o un parámetro o una variable de una hoja de estilos. Un ejemplo de código XSLT para este tipo de método sería:

```
<xsl:stylesheet version="2.0"
   xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
   xmlns:xs="http://www.w3.org/2001/XMLSchema"
   xmlns:fn="http://www.w3.org/2005/xpath-functions">
   <xsl:output method="xml" omit-xml-declaration="yes"/>
   <xsl:template match="/">
      <xsl:variable name="releasedate"
         select="date:new(2008, 4, 29)"
         xmlns:date="clitype:System.DateTime"/>
      <doc>
         <date>
            <xsl:value-of select="date:ToString(date:new(2008, 4, 29))"
               xmlns:date="clitype:System.DateTime"/>
         </date>
         <date>
            <xsl:value-of select="date:ToString($releasedate)"
```

 © 2016-2022 Altova GmbH

```
                xmlns:date="clitype:System.DateTime"/>
            </date>
        </doc>
    </xsl:template>
</xsl:stylesheet>
```

En el ejemplo anterior, se usó un constructor `System.DateTime` (`new(2008, 4, 29)`) para crear un objeto .NET de tipo `System.DateTime`. Este objeto se creó dos veces, una vez como valor de la variable `releasedate`, y otra vez como primer y único argumento del método `System.DateTime.ToString()`. Al método de instancia `System.DateTime.ToString()` se le llama dos veces, ambas con el constructor `System.DateTime` (`new(2008, 4, 29)`) como primer y único argumento. En una de estas instancias, se usó la variable `releasedate` para obtener el objeto .NET.

Métodos de instancia y campos de instancia

La diferencia entre un método de instancia y un campo de instancia es solo teórica. En un método de instancia, se pasa directamente un objeto .NET como argumento. En un campo de instancia, se pasa un parámetro o una variable (aunque el parámetro o la variable puede contener un objeto .NET). Por ejemplo, en el código del ejemplo anterior, la variable `releasedate` contiene un objeto .NET y esta es la variable que se pasa como argumento de `ToString()` en el segundo constructor de elemento `date`. Por tanto, la instancia `ToString()` del primer elemento `date` es un método de instancia, mientras que la segunda se considera un campo de instancia. El resultado es el mismo en ambos casos.

18.2.2.2.2.4 Tipos de datos: conversión de XPath/XQuery en .NET

Cuando se usa una función de extensión .NET dentro de una expresión XPath/XQuery, los tipos de datos de los argumentos de la función son importantes para determinar a cuál de los métodos .NET que tienen el mismo nombre se está llamando.

En .NET se siguen estas normas:

- Si en una clase hay varios métodos que tienen el mismo nombre, solamente se pueden seleccionar los métodos que tienen el mismo número de argumentos que la llamada a función.
- Los tipos de datos de cadena, numéricos y booleanos XPath/XQuery (*ver lista más abajo*) se convierten de forma implícita en el tipo de datos .NET correspondiente. Si el tipo XPath/XQuery suministrado se puede convertir en más de un tipo .NET (p. ej. `xs:integer`), entonces se selecciona el tipo .NET que se declaró para el método seleccionado. Por ejemplo, si el método .NET al que se está llamando es `fx(double)` y el tipo de datos XPath/XQuery suministrado es `xs:integer`, entonces se convierte `xs:integer` en el tipo de datos .NET `double`.

La tabla que aparece a continuación enumera las conversiones implícitas de los tipos de cadena, numéricos y booleanos XPath/XQuery en tipos de datos .NET.

`xs:string`	`StringValue, string`
`xs:boolean`	`BooleanValue, bool`
`xs:integer`	`IntegerValue, decimal, long, integer, short, byte, double, float`

xs:float	FloatValue, float, double
xs:double	DoubleValue, double
xs:decimal	DecimalValue, decimal, double, float

Los subtipos de los tipos de datos XML Schema de la tabla anterior (que se usan en XPath y XQuery) también se convierten en los tipos .NET correspondientes al tipo antecesor del subtipo.

En algunos casos quizás no sea posible seleccionar el método .NET correcto usando la información dada. Por ejemplo, imagine que:

- El argumento presentado es un valor `xs:untypedAtomic` de 10 y está destinado al método `mimétodo(float)`.
- Sin embargo, hay otro método en la clase que toma un argumento de otro tipo de datos: `mimétodo(double)`.
- Puesto que los métodos tienen el mismo nombre y el tipo suministrado (`xs:untypedAtomic`) se puede convertir correctamente tanto en `float` como en `double`, es posible que `xs:untypedAtomic` se convierta en `double` en lugar de en `float`.
- Por consiguiente, el método seleccionado no será el método necesario y puede que no produzca el resultado esperado. Una solución es crear un método definido por el usuario con un nombre diferente y usar ese método.

Los tipos que no aparecen en la lista anterior (p. ej. `xs:date`) no se convertirán y generarán un error.

18.2.2.2.2.5 Tipos de datos: conversión de .NET en XPath/XQuery

Cuando un método .NET devuelve un valor y el tipo de datos del valor es un tipo de cadena, numérico o booleano, entonces se convierte en el tipo de datos XPath/XQuery correspondiente. Por ejemplo, el tipo de datos .NET `decimal` se convierte en `xsd:decimal`.

Cuando se devuelve un objeto .NET o un tipo de datos que no es de cadena, numérico ni booleano, puede garantizar la conversión del tipo XPath/XQuery necesario usando primero un método .NET (p. ej. `System.DateTime.ToString()`) para convertir el objeto .NET en una cadena. En XPath/XQuery la cadena se puede modificar para ajustarse a la representación léxica del tipo necesario y convertirse después en dicho tipo (usando la expresión `cast as`, por ejemplo).

18.2.2.2.3 Scripts MSXSL para XSLT

El elemento `<msxsl:script>` contiene funciones y variables definidas por el usuario a las que se puede llamar desde dentro de expresiones XPath en la hoja de estilos XSLT. El elemento `<msxsl:script>` es un elemento de nivel superior, es decir, debe ser un elemento secundario de `<xsl:stylesheet>` o `<xsl:transform>`.

El elemento `<msxsl:script>` debe estar en el espacio de nombres `urn:schemas-microsoft-com:xslt` (*ver ejemplo más abajo*).

 © 2016-2022 Altova GmbH

Lenguaje de scripting y espacio de nombres

El lenguaje de scripting utilizado dentro del bloque se especifica en el atributo `language` del elemento `<msxsl:script>` y el espacio de nombres que se debe usar para las llamadas a función desde expresiones XPath se identifica con el atributo `implements-prefix`:

```
<msxsl:script language="lenguaje-de-scripting" implements-prefix="prefijo-espacioNombres-
usuario">

   función-1 o variable-1
   ...
   función-n o variable-n

</msxsl:script>
```

El elemento `<msxsl:script>` interactúa con Windows Scripting Runtime, de modo que dentro del elemento `<msxsl:script>` solamente se pueden usar lenguajes que estén instalados en el equipo. Para poder usar scripts MSXSL es necesario tener **instalada la plataforma .NET Framework 2.0 (o superior)**. Por tanto, los lenguajes de scripting .NET se pueden usar dentro del elemento `<msxsl:script>`.

El atributo `language` admite los mismos valores que el atributo `language` del elemento HTML `<script>`. Si no se especifica el atributo `language`, entonce se asume Microsoft JScript por defecto.

El atributo `implements-prefix` toma un valor que es un prefijo de un espacio de nombres declarado dentro del ámbito. Este espacio de nombres suele ser un espacio de nombres de usuario que se reservó para una biblioteca de funciones. Todas las funciones y variables definidas dentro del elemento `<msxsl:script>` están en el espacio de nombres identificado por el prefijo indicado en el atributo `implements-prefix`. Cuando se llama a una función desde dentro de una expresión XPath, el nombre de función completo debe estar en el mismo espacio de nombres que la definición de función.

Ejemplo

Aquí puede ver un ejemplo de una hoja de estilos XSLT que usa una función definida dentro de un elemento `<msxsl:script>`.

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="2.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
   xmlns:xs="http://www.w3.org/2001/XMLSchema"
   xmlns:fn="http://www.w3.org/2005/xpath-functions"
   xmlns:msxsl="urn:schemas-microsoft-com:xslt"
   xmlns:user="http://mycompany.com/mynamespace">

  <msxsl:script language="VBScript" implements-prefix="user">
    <![CDATA[
    ' Input: A currency value: the wholesale price
    ' Returns: The retail price: the input value plus 20% margin,
    ' rounded to the nearest cent
    dim a as integer  = 13
    Function AddMargin(WholesalePrice) as integer
```

```
      AddMargin = WholesalePrice * 1.2 + a
    End Function
  ]]>
  </msxsl:script>

  <xsl:template match="/">
    <html>
      <body>
        <p>
          <b>Total Retail Price =
            $<xsl:value-of select="user:AddMargin(50)"/>
          </b>
          <br/>
          <b>Total Wholesale Price =
            $<xsl:value-of select="50"/>
          </b>
        </p>
      </body>
    </html>
  </xsl:template>
</xsl:stylesheet>
```

Tipos de datos

Los valores de los parámetros que se pasan dentro y fuera del bloque de script solamente pueden ser tipos de datos XPath. Esta restricción no afecta a los datos que se pasan las funciones y variables situadas dentro del bloque de script.

Ensamblados

Puede importar un ensamblado al script usando el elemento `msxsl:assembly`. El ensamblado se identifica con un nombre o un URI. El ensamblado se importa cuando se compila la hoja de estilos. Aquí puede ver cómo se usa el elemento `msxsl:assembly`:

```
<msxsl:script>
       <msxsl:assembly name="miEnsamblado.nombreEnsamblado" />
       <msxsl:assembly href="rutaDelEnsamblado" />

       ...

</msxsl:script>
```

El nombre de ensamblado puede ser un nombre completo, como:

```
"system.Math, Version=3.1.4500.1 Culture=neutral PublicKeyToken=a46b3f648229c514"
```

o un nombre abreviado, como `"miEnsamblado.Draw"`.

 © 2016-2022 Altova GmbH

Espacios de nombres

Puede declarar espacios de nombres con el elemento `msxsl:using`. Esto permite escribir las clases del ensamblado en el script sin sus espacios de nombres, lo cual le permitirá ahorrar mucho tiempo. Aquí puede ver cómo se usa el elemento `msxsl:using` para declarar espacios de nombres.

```
<msxsl:script>
       <msxsl:using namespace="ENmiEnsamblado.NombreEspaciodenombres" />

       ...

</msxsl:script>
```

El valor del atributo `namespace` es el nombre del espacio de nombres.

18.3 Datos técnicos

Esta sección incluye información sobre algunos de aspectos técnicos de su software. La información está organizada en varios apartados:

- Requisitos de OS y memoria 1590
- Motores de Altova 1590
- Compatibilidad con Unicode 1591
- Uso de Internet 1591

18.3.1 Requisitos de SO y memoria

Sistema operativo

Las aplicaciones de software de Altova están disponibles en estas plataformas:

- Windows 7 SP1 con actualización de la plataforma, Windows 8, Windows 10, Windows 11
- Windows Server 2008 R2 SP1 con actualización de la plataforma o superior

Memoria

Puesto que el software está escrito en C++ no necesita tanto espacio como un JRE y suele necesitar menos memoria que otras aplicaciones similares basadas en Java. No obstante, todos los documentos se cargan en memoria por completo, para poder analizarlos completamente y mejorar la velocidad de visualización y edición. Los requisitos de memoria, por consiguiente, aumentan en función del tamaño del documento.

Los requisitos de memoria también vienen dados por el historial de operaciones Deshacer. Cuando se cortan y pegan secciones grandes de documentos de gran tamaño, la memoria disponible se puede agotar rápidamente.

18.3.2 Motores XSLT y XQuery de Altova

Validador XML

Al abrir un documento XML, la aplicación usa su validador XML integrado para comprobar si el formato es correcto, para validar el documento en relación a un esquema (si se ha especificado uno) y para generar estructuras y conjuntos de información (infosets). El validador XML también se usa para proporcionar ayuda de edición inteligente cuando usted modifique documentos y para mostrar de forma dinámica cualquier error de validación que ocurra.

El validador XML integrado implementa la recomendación final para las especificaciones 1.0 y 1.1 del esquema de XML del W3C. Altova incorpora continuamente las recomendaciones más recientes del Grupo de Trabajo del esquema de XML del W3C a su validador XML para que los productos de Altova siempre ofrezcan el entorno de desarrollo más avanzado.

© 2016-2022 Altova GmbH

Motores XSLT y XQuery

Los productos de Altova usan los motores XSLT 1.0, XSLT 2.0, XSLT 3.0, XQuery 1.0 y XQuery 3.1 de Altova. Si alguno de estos motores está incluido en el producto, encontrará documentación específica sobre el comportamiento de cada motor en la implementación en los anexos.

Nota: Altova MapForce genera código con los motores XSLT 1.0, XSLT 2.0 y XQuery 1.0.

18.3.3 Compatibilidad con Unicode

Los productos XML de Altova son completamente compatibles con Unicode. Para editar un documento XML también necesitará una fuente compatible con los caracteres Unicode utilizados por el documento.

Tenga en cuenta que la mayoría de las fuentes contienen solamente un subconjunto muy concreto de caracteres Unicode y, por tanto, están destinadas a un sistema de escritura concreto. Si algunos caracteres aparecen desfigurados, el motivo puede ser que la fuente seleccionada no contiene los glifos necesarios. Por tanto, es recomendable tener una fuente que abarque todos los caracteres Unicode. Sobre todo si edita documentos XML en varios idiomas o sistemas de escritura. Una fuente Unicode que suele venir con los equipos Windows es la fuente Arial Unicode MS.

En la carpeta `/Examples` de la carpeta de su aplicación puede encontrar un archivo XHTML llamado `UnicodeUTF-8.html` que incluye esta frase en gran número de idiomas y sistemas de escritura diferentes:

- *When the world wants to talk, it speaks Unicode*
- *Cuando el mundo quiere conversar, habla Unicode*
- *Wenn die Welt miteinander spricht, spricht sie Unicode*
- 世界的に話すなら、Unicode です。

Abra este archivo XHTML y observe el potencial de Unicode.

18.3.4 Uso de Internet

Las aplicaciones de Altova inician conexiones a Internet en estos casos:

- Si hace clic en el botón **Solicitar una clave de evaluación GRATUITA** del cuadro de diálogo "Activación del software" (**Ayuda | Activación del software**), los campos del cuadro de diálogo de activación del software se transfieren a nuestro servidor web por medio de una conexión HTTP corriente (puerto 80) y le enviamos el código de evaluación gratuito por correo electrónico.
- En algunos productos de Altova puede abrir un archivo por Internet (**Archivo | Abrir | Cambiar a URL**). En este caso, el documento se recupera usando uno de estos protocolos y conexiones: HTTP (normalmente por el puerto 80), FTP (normalmente por el puerto 20/21) o HTTPS (normalmente por el puerto 443). También puede ejecutar un servidor HTTP en el puerto 8080. (En el cuadro de diálogo "Abrir URL", después del nombre de servidor escriba dos puntos y el número de puerto.)
- Si abre un documento XML que hace referencia a un documento DTD o esquema XML y el documento se especifica a través de una URL, el documento de esquema al que se hace referencia también se recupera a través de una conexión HTTP (puerto 80) o cualquier otro protocolo (ver punto anterior). El documento de esquema también se recupera para validar el archivo XML. Recuerde que la validación

puede realizarse automáticamente nada más abrir el documento, si seleccionó esta opción en la sección *Archivo* del cuadro de diálogo "Opciones" (**Herramientas | Opciones**).

- En las aplicaciones de Altova que trabajen con WSDL y SOAP, las conexiones a servicios web son definidas por documentos WSDL.
- Si usa el comando **Archivo | Enviar por correo electrónico** de MapForce, el texto seleccionado actualmente o el archivo se envía con el programa de correo electrónico instalado en el equipo.
- Durante la activación del software y la búsqueda de actualizaciones, tal y como se describe en el contrato de licencia de software de Altova.

© 2016-2022 Altova GmbH

18.4 Información sobre licencias

En esta sección encontrará información sobre:

- la distribución de este producto de software
- la activación del software y medición de licencias
- el contrato de licencia para el usuario final que rige el uso de este producto de software

Los términos del contrato de licencia que aceptó al instalar el producto de software son vinculantes, por lo que rogamos lea atentamente toda esta información.

Para leer los términos y condiciones de cualquiera de las licencias de Altova, consulte la página de información legal de Altova en el sitio web de Altova.

18.4.1 Distribución electrónica de software

Este producto está disponible por distribución electrónica de software, un método de distribución que ofrece ventajas únicas:

- Puede evaluar el software de forma totalmente gratuita durante 30 días antes de decidir si compra el producto (*Nota: la licencia para Altova Mobile Together Designer es gratuita*).
- Si decide comprarlo, puede hacer un pedido en línea en el sitio web de Altova y conseguir en pocos minutos el software con licencia.
- Si realiza el pedido en línea, siempre recibirá la versión más reciente de nuestro software.
- El paquete de instalación del producto incluye un sistema de ayuda en pantalla al que se puede acceder desde la interfaz de la aplicación. La versión más reciente del manual del usuario está disponible en www.altova.com (i) en formato HTML y (ii) en formato PDF para descargar e imprimir si lo desea.

Período de evaluación de 30 días

Después de descargar el producto de software, puede probarlo de forma totalmente gratuita durante un plazo de 30 días. Pasados unos 20 días, el software empieza a recordarle que no tiene una licencia. El mensaje de aviso aparece una sola vez cada vez que se inicie la aplicación. Para seguir utilizando el programa una vez pasado el plazo de 30 días, deberá comprar una licencia permanente, que se entrega en forma de código clave. Para desbloquear el producto debe introducir ese código clave en el cuadro de diálogo "Activación del software".

Las licencias de los productos pueden comprarse directamente en la tienda en línea del sitio web de Altova.

Distribuir la versión de evaluación a otros usuarios de su organización

Si desea distribuir la versión de evaluación en la red de su compañía o si desea usarlo en un PC que no está conectado a Internet, solamente puede distribuir los programas de instalación (siempre y cuando no se modifiquen de forma alguna). Todo usuario que acceda al instalador debe solicitar su propio código clave de

evaluación (de 30 días). Una vez pasado este plazo de 30 días, todos los usuarios deben comprar también una licencia para poder seguir usando el producto.

18.4.2 Activación del software y medición de licencias

Durante el proceso de activación del software de Altova, puede que la aplicación utilice su red interna y su conexión a Internet para transmitir datos relacionados con la licencia durante la instalación, registro, uso o actualización del software a un servidor de licencias operado por Altova y para validar la autenticidad de los datos relacionados con la licencia y proteger a Altova de un uso ilegítimo del software y mejorar el servicio a los clientes. La activación es posible gracias al intercambio de datos de la licencia (como el sistema operativo, la dirección IP, la fecha y hora, la versión del software, el nombre del equipo, etc.) entre su equipo y el servidor de licencias de Altova.

Su producto incluye un módulo integrado de medición de licencias que le ayudará a evitar infracciones del contrato de licencia para el usuario final. Puede comprar una licencia de un solo usuario o de varios usuarios para el producto de software y el módulo de medición de licencias se asegura de que no se utiliza un número de licencias mayor al permitido.

Esta tecnología de medición de licencias usa su red de área local (LAN) para comunicarse con las instancias de la aplicación que se ejecutan en equipos diferentes.

Licencia de un solo usuario

Cuando se inicia la aplicación, se inicia el proceso de medición de licencias y el software envía un breve datagrama de multidifusión para averiguar si hay otras instancias del producto activas en otros equipos del mismo segmento de red al mismo tiempo. Si no recibe ninguna respuesta, la aplicación abre un puerto para escuchar a otras instancias de la aplicación.

Licencia de varios usuarios

Si se usa más de una instancia de la aplicación dentro de la misma red LAN, estas instancias se comunicarán entre ellas al iniciarse. Estas instancias intercambian códigos claves para que ayudarle a no sobrepasar por error el número máximo de licencias concurrentes. Se trata de la misma tecnología de medición de licencias que suele utilizarse en Unix y en otras herramientas de desarrollo de bases de datos. Gracias a ella puede comprar licencias de varios usuarios de uso concurrente a un precio razonable.

Las aplicaciones se diseñaron de tal modo que envían pocos paquetes pequeños de red y no cargan demasiado su red. Los puertos TCP/IP (2799) utilizados por su producto de Altova están registrados oficialmente en la IANA (*para más información consulte el sitio web de la IANA www.iana.org*) y nuestro módulo de medición de licencias es una tecnología probada y eficaz.

Si usa un servidor de seguridad, puede notar las comunicaciones del puerto 2799 entre los equipos que ejecutan los productos de Altova. Si quiere, puede bloquear ese tráfico, siempre y cuando esto no resulte en una infracción del contrato de licencia.

 © 2016-2022 Altova GmbH

Nota sobre los certificados

Su aplicación de Altova contacta con el servidor de licencias de Altova (`link.altova.com`) vía HTTPS. Para esta comunicación, Altova usa un certificado SSL registrado. Si se reemplaza este certificado (por ejemplo, si lo reemplaza su departamento de informática o un organismo externo), entonces su aplicación de Altova le advertirá de que la conexión puede no ser segura. Si usa el certificado sustitutivo para iniciar la aplicación, lo hace por su cuenta y riesgo. Si ve un mensaje de advertencia de que la conexión puede no ser segura, compruebe el origen del certificado y consulte con su equipo técnico (que decidirán si se debe continuar con el reemplazo del certificado de Altova).

Si su organización necesita usar su propio certificado (por ejemplo, para monitorizar la comunicación hacia y desde equipos cliente), entonces recomendamos que instale en su red Altova LicenseServer, el software gratuito de gestión de licencias de Altova. Así, sus equipos cliente pueden seguir usando los certificados de su organización y AltovaLicenseServer puede usar el certificado de Altova cuando necesite comunicarse con Altova.

18.4.3 Contrato de licencia para el usuario final

- Encontrará el Contrato de licencia de Altova para el usuario final en: https://www.altova.com/legal/eula
- Encontrará la Política de privacidad de Altova en: https://www.altova.com/privacy

Índice

2

2005R3,
modo de compatibilidad, 1207
2023, 1246

A

A - Z,
componente de ordenación, 505
abs,
como función de MapForce (en lang | math functions), 749
como función de MapForce (en xpath2 | numeric functions), 806
Acciones,
relacionadas con conexiones, 87
acos,
como función de MapForce (en lang | math functions), 749
ActiveX,
integration at application level, 1424
integration at document level, 1427
integration prerequisites, 1420
ActiveX controls,
adding to the Visual Studio Toolbox, 1422
add,
como función de MapForce (en core | math functions), 669
ADO,
como interfaz de conexión de datos, 161
configurar una conexión, 168
ADO.NET,
configurar una conexión, 173
age,
como función de MapForce (en lang | dateTime functions), 726
Agregar,
biblioteca personal, 612
Análisis de impacto,
hoja de estilos SPS, 916
Ant,
crear un proyecto de Eclipse con, 1086
generar código Java con, 1089
Any,
xs:any, 137
API,
documentation, 1225
API de MapForce, 1225
Archivo,
Abrir, 1025
Abrir gestor de credenciales, 1025
Archivos recientes, 1025
Cerrar, 1025
Cerrar todos, 1025
como botón en componentes, 892
como botón en un componente, 79
como menú de aplicación, 1025
Compilar en MapForce Server Execution File, 1025
Configurar asignación, 1025
Configurar impresión, 1025
Generar código, 1025
Generar documentación, 1025
Guardar, 1025
Guardar como, 1025
Guardar todos, 1025
Implementar en FlowForce Server, 1025
Imprimir, 1025
Nuevo, 1025
Salir, 1025
Validar asignación, 1025
Vista previa de impresión, 1025
Volver a cargar, 1025
Archivo de biblioteca,
mff, 612
Archivo mff,
configurar, 612
Archivo plano,
asignación, 445
Archivo/Cadena,
como botón en componentes, 892
como botón en un componente, 79
Archivo: (predeterminado),
como nombre de nodo raíz, 892
Archivo: <dinámico>,
como nombre de nodo raíz, 892
Archivos CSV,
agregar o quitar campos, 454
como componente de destino, 454
como componente de origen, 454
configurar codificación, 454
vista previa, 454

© 2016-2022 Altova GmbH

Archivos de proyecto (.mfp), 118
Archivos de texto,
agregar o quitar campos, 462
asignar datos, 458
como componente de destino, 462
como componente de origen, 462
configurar codificación, 462
establecer carácter de relleno, 458
establecer el tamaño del campo fijo, 458
vista previa, 462
archivos XML,
como recursos globales, 976
generar a partir de registros de BD, 900
generar a partir de un solo origen XML, 898
Asignación,
ajustes, 115
basada en origen (contenido mixto), 91
componentes, 73
conectores, 73
conexiones, 73
crear, 73
depuración, 924
documentar, 915
esquema, 73
fundamentos, 73
hojas de estilos SPS predefinidas para la documentación, 916
partes, 73
terminología, 73
términos, 73
validar, 105
Asignación de datos,
formato de archivo plano, 445
Asignación de valores,
pasar datos sin modificarlos, 537
tabla de búsqueda, 533
tabla de búsqueda (propiedades), 539
Asignaciones,
procesamiento automatizado, 948
asin,
como función de MapForce (en lang | math functions), 750
atan,
como función de MapForce (en lang | math functions), 750
ATTLIST,
DTD y URI de espacio de nombres, 124
Automático/a,
carga de bibliotecas, 612
Automatizado,
procesamiento, 948
auto-number,
como función de MapForce (en core | generator functions), 660
avg,
función de MaForce (en core | aggregate functions), 634
Ayuda,
Acerca de MapForce, 1062
Activación del software, 1062
Buscar, 1062
Buscar actualizaciones, 1062
Centro de soporte técnico, 1062
como menú de aplicación, 1062
Contenido, 1062
Cursos de MapForce, 1062
Descargar herramientas gratis y componentes, 1062
Formulario de pedido, 1062
Índice, 1062
MapForce en Internet, 1062
Preguntas más frecuentes, 1062
Registro del software, 1062

B

Barras de herramientas,
Estado de la aplicación, 30
Herramientas, 30
Menú, 30
Basada en origen,
asignar contenido mixto, 91
Base de datos,
asignar esquema XML a un campo, 369
Codificación, 1053
Configuración del editor SQL, 1053
consulta, 380
Editor SQL, 1053
escribir archivos XML en, 373
Fuentes del editor de texto, 1053
generación de código SQL, 1053
generar valores secuenciales y únicos para, 277
generar varios archivos XML a partir de, 900
Vista de resultados, 1053
Base de datos Access,
actualización basada en condición IF, 458
Bases de datos,
actualizar datos de tabla, 291
actualizar e insertar datos de tabla, 294

Bases de datos,
agregar a la asignación, 247
combinar datos en, 294
como recursos globales, 980
compatibles, 159
ejecutar asignaciones en, 273
eliminar datos de tabla, 311
insertar datos en una tabla, 282
insertar datos en varias tablas vinculadas, 286
usar inserción masiva, 320
y contexto de asignación, 876
base-uri,
como función de MapForce function (en xpath2 | accessors library), 775
BD,
ORDER BY, 352
Biblioteca, 1221
agregar personal, 612
carga automática, 612
Biblioteca personal,
agregar, 612
Binary files,
adding to the mapping, 471
overview, 471
reading data from, 473
writing data to, 474
boolean,
como función de MapForce (en core | conversion functions), 640
Buscar,
archivos en la ventana Proyecto, 118
elementos en los componentes de asignación, 79
proyectos, 118

C

C#,
código, 1067
configuración, 1207
generar código de asignación de datos en, 1076
gestión de errores en, 1230
integrar código generado, 1100
integration of MapForce, 1431
C++,
código, 1067
configuración, 1207
generar código de asignación de datos en, 1071
gestión de errores en, 1230
integrar código generado, 1102
Cadena,
redistribuir datos de, 903
redistribuir datos desde, 903
serializar datos a, 903, 906
Campo,
clave en archivos de texto, 450
Campos vacíos,
en archivos CSV, 454
en archivos de texto planos, 462
capitalize,
como función de MapForce (en lang | string functions), 759
Carácter de relleno,
en archivos de texto planos, 462
Carpeta,
disposición en carpetas (consulta de la BD), 388
Carpetas,
como recursos globales, 978
Catálogos, 990
Catalogs,
customize, 995
environment variables, 997
in DTD, 991
in XML Schema, 991
structure, 993
CDATA, 136
ceiling,
como función de MapForce (en core | math functions), 669
char-from-code,
como función de MapForce (en core | string functions), 707
charset-decode,
como función de MapForce (en lang | string functions), 759, 761
Clave,
campos clave en archivos de texto, 450
clave de ordenación, 505
Clave de ordenación,
componente de ordenación, 505
Clave foránea,
en asignaciones de BD, 258, 267
Clave principal,
en asignaciones de BD, 258, 267
generar durante acción de inserción en BD, 282
generar para un componente de BD, 277
Code generation,
sample, 1244

© 2016-2022 Altova GmbH

code-from-char,
como función de MapForce (en core | string functions), 708
Codificación,
cambiar en archivos CSV, 454
cambiar en archivos de texto planos, 462
Código,
integrar el código de MapForce, 1097
SPL, 1209
tipos integrados, 1220
Código generado,
generar excepciones desde, 529
COM API,
documentation, 1225
Comandos de menú,
Archivo, 1025
Ayuda, 1062
Componente, 1034
Conexión, 1036
Depuración, 1040
Edición, 1028
Función, 1037
Herramientas, 1044
Herramientas | Opciones, 1049
Herramientas | Opciones | Base de datos, 1053
Herramientas | Opciones | Java, 1052
Herramientas | Opciones | Proxy de red, 1058
Herramientas | Personalizar, 1045
Herramientas | Teclado, 1047
Insertar, 1029
Personalizar, 1045, 1047
Proyecto, 1032
Resultados, 1038
Vista, 1042
Windows, 1061
Comentario de bloque, 385
Comentario de línea, 385
Comentarios,
agregar a archivos de destino, 135
Comillas,
en archivos CSV, 454
Command reference, 1450
Comodines,
xs:any - xs:anyAtrribute, 137
Compatibilidad con Unicode,
de los productos de Altova, 1591
Compatibilidad del código,
v2005R3, 1207
Compilador,
configuración, 1207
Complemento de MapForce para Eclipse,
acceso a menús y funciones frecuentes, 1008
ampliar características, 1022
cambiar a la perspectiva MapForce, 1005
configuración para la generación automática de código, 1019
crear asignaciones nuevas, 1014
crear un proyecto de MapForce/Eclipse, 1012
importar asignaciones a un proyecto de Eclipse, 1016
información, 1002, 1005, 1012
instalación, 1003
punto de extensión, 1022
trabajar con asignaciones y proyectos, 1012
Complemento de MapForce para Visual Studio,
acerca de, 999
habilitar, 999
información sobre menús y funciones, 999
trabajar con asignaciones y proyectos, 999
Complemento para Visual Studio,
ejecutar MapForce como, 999
Componente,
como menú de aplicación, 1034
ordenar datos, 505
Componente de destino,
cambiar el orden de procesamiento de, 888
Componentes,
AAlinear la estructura a la izquierda, 1034
Acciones de tabla de la base de datos, 1034
Actualizar, 1034
agregar a la asignación, 77
Agregar delante un duplicado de entrada, 1034
Agregar detrás un duplicado de entrada, 1034
Agregar, quitar o editar objetos de la base de datos, 1034
ajustes, 79
alinear, 79
Alinear la estructura a la derecha, 1034
Bases de datos, 159
buscar, 79
Cambiar de elemento raíz, 1034
cambiar la configuración, 79
comandos de menú, 1034
Consultar BD, 1034
Crear asignación a EDI X12 997, 1034
Crear asignación a EDI X12 999, 1034
de estructura, 75
Editar la configuración de FlexText, 1034
Editar la definición del esquema en XMLSpy, 1034
elementos eliminados, 101

Componentes,
eliminar, 102
Escribir contenido como secciones CDATA, 1034
esquema, 79
estructurales, 123, 124, 159
Propiedades, 1034
Quitar el duplicado, 1034
resumen, 75
transformación, 75
XML, 125
XML Schema, 125
XML y XML Schema, 125, 131
Componentes estructurales,
bases de datos, 159
Esquema XML, 124
XML, 124
XML y XML Schema, 124
Components,
Archivo de Excel 2007+, 1029
Archivo de texto, 1029
Archivo o esquema JSON, 1029
Archivo o esquema XML, 1029
Archivo Protocol Buffers, 1029
Asignación de valores, 1029
Base de datos, 1029
Combinar, 1029
Componente de entrada simple, 1029
componente de salida simple, 1029
Condición IF-Else, 1029
Constante, 1029
Documento XBRL, 1029
EDI, 1029
Excepción, 1029
Filtro: nodos/filas, 1029
Función de servicio web, 1029
Insertar componente de entrada, 1029
Insertar componente de salida, 1029
Ordenar: nodos/filas, 1029
Variable, 1029
WHERE/ORDER de SQL/NoSQL, 1029
concat,
(como función) ejemplo de uso, 458
como función de MapForce (en core | string functions), 708
Condiciones If-Else,
agregar a la asignación, 543
Conector,
ver historial de valores procesados, 934
Conexión,
como menú de aplicación, 1036
Conexión de base de datos,
configurar, 161
ejemplo de configuración, 191
iniciar el asistente, 163
reutilizar desde recursos globales, 190
Conexiones,
ajustes, 97
anotación, 97
basada en destino, 91
Basada en el destino (estándar), 1036
basada en origen, 91
Basada en origen (contenido mixto), 1036
cambiar, 87
Conectar automáticamente los secundarios equivalentes, 1036
Conectar los secundarios equivalentes, 1036
conexiones antecesoras ausentes, 87
Configurar la conexión de secundarios equivalentes, 1036
conservar conexiones tras eliminación de componentes, 102
Copia total (copia los elementos secundarios), 1036
copiar, 87
crear, 87
de copia total, 91, 95
eliminar, 87
entradas obligatorias, 87
estándar, 91
menú contextual, 99
mixta, 91
mover, 87, 101
Propiedades, 1036
reparar, 101
reparar después de editar el esquema, 101
resaltado selectivo, 87
secundarios equivalentes, 91, 94
tipos, 91, 97
ver la información rápida de las conexiones, 87
Conexiones defectuosas,
después de cambiar el esquema, 101
en archivos XML, 101
en bases de datos, 101
Configuración,
C++ y C#, 1207
de asignaciones, 115
del archivo de resultados, 115
del servicio web, 115
Configurar,
archivo mff, 612

© 2016-2022 Altova GmbH

Conservar datos,
cuando se use una asignación de valores, 537
Conservar datos sin modificarlos,
al pasar por una asignación de valores, 537
Constantes,
agregar, 550
Consulta de BD,
convertir texto en comentario, 385
marcadores, 386
Consulta de la BD,
filtrar tablas, 389
generar SQL, 388, 392
regiones, 387
contains,
como función de MapForce (en core | string functions), 710
Contenido mixto,
asignar, 91
con conexiones basadas en destino, 91
con conexiones estándar, 91
Contexto (ventana),
información, 928, 936
Contexto de asignación, 874
Contexto de prioridad, 883
ejemplo, 885
Contexto matriz,
ejemplo, 879
Contraer,
regiones, 387
Contrato de licencia para el usuario final, 1593, 1595
Controladores de BD,
resumen, 165
Controladores ODBC,
comprobar disponibilidad, 180
convert-to-utc,
como función de MapForce (en lang | dateTime functions), 726
Cortar/Copiar/Pegar/Eliminar,
Buscar, 1028
Buscar anterior, 1028
Buscar siguiente, 1028
Deshacer, 1028
Rehacer, 1028
Seleccionar todo, 1028
cos,
como función de MapForce (en lang | math functions), 751
count,
como función de MapForce (en core | aggregate functions), 635
count-substr,
como función de MapForce (en lang | string functions), 763
Crear,
regiones, 387
create-guid,
como función de MapForce (en lang | generados functions), 746
Criterio de ordenación,
cambiar, 505
CSV,
como origen de la asignación, 445
crear jerarquías (claves), 450
crear varias filas, 447
current,
como función de MapForce (en xslt | xslt functions), 835
current- dateTime,
como función de MapForce (en xpath2 | context functions), 780
current- time,
como función de MapForce (en core | context functions), 781
current-date,
como función de MapForce (en core | context functions), 780

D

date-from-datetime,
como función de MapForce (en lang | datetime functions), 727
datetime- diff,
como función de MapForce (en lang | dateTime functions), 728
datetime-add,
como función de MapForce (en lang | dateTime functions), 727
datetime-from-date-and-time,
como función de MapForce (en lang | datetime functions), 729
datetime-from-parts,
como función de MapForce (en lang | dateTime functions), 730
Datos de tabla,
ordenar, 505
day-from- duration,
como función de MapForce (en lang | dateTime functions), 732
day-from-datetime,

day-from-datetime,
como función de MapForce (en lang | datetime functions), 731
default-collation,
como función de MapForce (en xpath2 | context functions), 781
Degradado,
fondo, 1207
degrees,
como función de MapForce (en lang | math functions), 751
Delimitador,
cambiar en archivos CSV, 454
cambiar en archivos de texto planos, 462
Depuración,
como menú de aplicación, 1040
con puntos de interrupción, 924
Detener, 1040
información, 928
Iniciar, 1040
Paso a paso para salir, 1040
Paso a paso por instrucciones, 1040
Paso a paso por procedimientos, 1040
Paso mínimo, 1040
por pasos, 924
preparar, 927
restricciones, 924
Destino, 23
Disposición,
explorador, 388
distinct-values,
como función de MapForce (en core | sequence functions), 681
Distribución,
de productos de software de Altova, 1593
divide,
como función de MapForce (en core | math functions), 670
divide-integer,
como función de MapForce (en lang | math functions), 751
Dll,
configuración del compilador, 1207
document,
como función de MapForce (en xslt | xslt), 835
Documentación,
definir hojas de estilos SPS, 921
Documentación general,
hoja de estilos SPS, 916
Documentar,
asignaciones, 915
Document-level,
examples of integration of XMLSpy, 1431
DoTransform.bat,
ejecutar con RaptorXML Server, 947
DSN de archivo,
configurar, 180
DSN de sistema,
configurar, 180
DSN de usuario,
configurar, 180
DTD,
origen y destino, 124
Duplicar elementos de entrada, 50
duration- add,
como función de MapForce (en lang | dateTime functions), 732
duration- subtract,
como función de MapForce (en lang | dateTime functions), 734
Duration-from-parts,
como función de MapForce (en lang | dateTime functions), 733

E

Eclipse,
compilar código Java con, 1089
generar código de asignación de datos para, 1083
Edición,
como menú de aplicación, 1028
Editor SQL,
convertir texto en comentario, 385
crear regiones, 387
insertar comentarios, 385
insertar marcadores, 386
insertar regiones, 387
margen de marcadores, 386
quitar comentarios, 385
quitar marcadores, 386
quitar regiones, 387
usar marcadores, 386
usar regiones, 387
element-available,
como función de MapForce (en xslt | xslt functions), 836
Elementos,
ausentes, 101
Elementos ausentes, 101

Eliminación inteligente de componentes, 102
Eliminar,
elementos ausentes, 101
Eliminar los nombres de esquema de los nombres de tabla,
como opción de generación de código, 255
empty,
como función de MapForce (en lang | string functions), 764
Entrada,
duplicar, 79
Entrada de la asignación,
especificar nombre de archivo personalizado como, 897
especificar varios archivos como, 892, 895, 896
Enumerations,
in MapForceControl, 1478
equal-or-greater,
como función de MapForce (en core | logical functions), 664
equal-or-less,
como función de MapForce (en core | logical functions), 664
Errores,
errores de memoria, 77
resolución de problemas, 77
Espacios de nombres,
declarar personalizados, 140
y comodines (xs:any), 137
Esquema,
estándar del sector, 124
generador de código, 1067
generar, 124
paquete, 124
Excepción,
memoria insuficiente, 1206
Excepciones,
agregar, 529
Excepciones de BD, 422, 428
gestionar, 422, 428
registrar, 435
Registro, 439
revertir, 428
usar en asignaciones, 422
exists,
como función de MapForce (en core | sequence functions), 683
exp,
como función de MapForce (en lang | math functions), 752
Expandir,
regiones, 387
Explorador,
aplicar filtros, 389
consulta de la BD, 388
filtrado, 389
Expresiones regulares,
en asignaciones, 627
Extensiones de Altova,
funciones para gráficos, 1492

F

false,
como función de MapForce (en lang | boolean functions), 778
Fija,
archivos de longitud fija (asignación de), 445
Filas,
asignar filas en archivos de texto, 450
Filtrar,
datos de componentes, 543
de bases de datos, 356
explorador, 389
objetos de BD en la asignación, 252
objetos de la BD, 389
tablas de base de datos, 543
Filtros,
agregar a la asignación, 543
find-substring,
como función de MapForce (en lang | string functions), 764
Firebird,
conectarse por JDBC, 192
conectarse por ODBC, 193
first-items,
como función de MapForce (en core | sequence functions), 685
floor,
como función de MapForce (en core | math functions), 670
FlowForce Server,
implementar asignaciones en, 958
implementar recursos globales en, 958, 987
Recursos globales en, 987
Fondo,
degradado, 1207
format-date,
como función de MapForce (en core | conversion functions), 641
format-dateTime,

format-dateTime,
como función de MapForce (en core | conversion functions), 642
format-guid- string,
como función de MapForce (en lang | string functions), 765
format-number,
como función de MapForce (en core | conversion functions), 645, 649
format-time,
como función de MapForce (en core | conversion functions), 648
Función,
como menú de aplicación, 1037
Funciones,
agregar, 550
agregar parámetros, 550
buscar, 550
buscar en la ventana Bibliotecas, 550
buscar ocurrencias en la asignación activa, 550
constantes, 550
descripción, 550
eliminar parámetros, 550
esquema, 550
nodo, 558, 560
parámetros, 550
tipo de datos del argumento, 550
Funciones de extensión .NET,
campos de instancia, 1584
campos estáticos, 1583
constructores, 1583
conversiones de tipos de datos, 1585, 1586
métodos de instancia, 1584
métodos estáticos, 1583
para XSLT y XQuery, 1580
resumen, 1580
tipos de datos .NET en XPath/XQuery, 1586
tipos de datos XPath/XQuery en .NET, 1585
Funciones de extensión .NET para XSLT y XQuery,
ver Funciones de extensión .NET, 1580
Funciones de extensión en scripts MSXSL,
msxsl:script, 1586
Funciones de extensión Java,
archivos de clases definidos por el usuario, 1573
archivos JAR definidos por el usuario, 1576
campos de instancia, 1578
campos estáticos, 1577
constructores, 1577
conversiones de tipos de datos, 1579, 1580
métodos de instancia, 1578
métodos estáticos, 1577
para XSLT y XQuery, 1571
resumen, 1571
tipos de datos Java en XPath/XQuery, 1580
tipos de datos XPath/XQuery en Java, 1579
Funciones de extensión Java para XSLT y XQuery,
ver Funciones de extensión Java, 1571
Funciones de extensión para XSLT y XQuery, 1570
Funciones de nodo,
crear, 560
Funciones definidas por el usuario,
agregar parámetros, 581
búsqueda, 589
búsqueda recursiva, 586
copiar y pegar, 576
crear, 576
de tipo complejo, 581
de tipo simple, 581
edición, 576
ejemplo, 575
ejemplos, 586, 589
eliminar, 576
Estructuras de tipo complejo, 581
importar, 576
inline, 576
llamada, 576
llamadas recursivas, 586
navegación, 576
orden de los parámetros, 581
parámetros, 581
parámetros de entrada, 576
parámetros de salida, 576
recursivamente, 586
regular, 576
resumen, 575
ventajas de las, 575
y contexto de asignación, 876
Funciones utilizadas por, 916
function-available,
como función de MapForce (en xslt | xslt functions), 836
Functions,
Configuración de la función, 1037
Crear función definida por el usuario a partir de la selección, 1037
Crear una función definida por el usuario, 1037
Insertar componente de entrada, 1037
Insertar componente de salida, 1037

© 2016-2022 Altova GmbH

Functions,
Quitar función, 1037

G

Generación de código, 107, 115
Generador de código, 1067
Generar,
código a partir de un esquema, 1067
generate-id,
como función de MapForce (en xslt | xslt), 837
generate-sequence,
como función de MapForce (en core | sequence functions), 686
Gestión de errores,
descripción general, 1230
Gestor de esquemas,
aplicar un parche a un esquema, 150
desinstalar un esquema, 151
estado de los esquemas en el, 148
funcionamiento del, 146
help (comando ILC), 152
info (comando ILC), 153
initialize (comando ILC), 153
instalar un esquema, 150
install (comando ILC), 154
introducción, 143
introducción a la ILC, 152
list (comando ILC), 155
lista de esquemas por estado, 148
reset (comando ILC), 155
restaurar, 151
uninstall (comando ILC), 156
update (comando ILC), 157
upgrade (comando ILC), 158
Gestor de esquemas XML, 124
get-fileext,
como función de MapForce (en core | file path functions), 656
get-folder,
como función de MapForce (en core | file path functions), 656
goup-ending-with,
como función de MapForce (en core | sequence functions), 691
greater,
como función de MapForce (en core | logical functions), 665
group-adjacent,
como función de MapForce (en core | sequence functions), 687
group-by,
como función de MapForce (en core | sequence functions), 689
group-into-blocks,
como función de MapForce (en core | sequence functions), 692
group-starting-with,
como función de MapForce (en core | sequence functions), 694

H

Health Level 7,
ejemplo, 469
Herramientas, 1044
comando de menú, 1044
como menú de aplicación, 1044
Configuración activa, 1044
Crear asignación invertida, 1044
Gestor de taxonomías XBRL, 1044
Opciones, 1044
Personalizar, 1044
Recursos globales, 1044
Restaurar barras de herramientas y ventanas, 1044
Herramientas | Opciones,
Base de datos, 1049
Depurador, 1049
Edición, 1049
Generación, 1049
Generales, 1049
Java, 1049
Mensajes, 1049
Proxy de red, 1049
XBRL, 1049
HL7 2.6 a 3.x,
ejemplo, 469
Hojas de estilos,
definir para la documentación, 921
hour-from- duration,
como función de MapForce (en lang | datetime functions), 735
hour-from-datetime,
como función de MapForce (en lang | datetime functions), 735

HRESULT,
y gestión de errores, 1230
HTML,
documentación de la asignación, 915
vista previa del resultado de la asignación como, 911

I

IBM DB2,
conectarse por JDBC, 196
conectarse por ODBC, 198
escribir en campos de tipo XML, 369
leer campos de tipo XML, 369
IBM DB2 for i,
conectarse por JDBC, 204
conectarse por ODBC, 205
IBM Informix,
conectarse por JDBC, 208
Iconos,
en la ventana Mensajes de la consulta de BD, 392
en la ventana Resultados de la consulta de BD, 392
IGU, 29
implicit- timezone,
como función de MapForce (en xpath2 | context functions), 781
Información general, 1590
Información legal, 1593
Información sobre derechos de autor, 1593
Información técnica, 1590
Insertar,
comentario de bloque, 385
comentario de línea, 385
comentarios, 385
como menú de aplicación, 1029
componente WHERE de SQL, 352
marcadores, 386
regiones, 387
Instrucciones de procesamiento,
agregar a archivos de destino, 135
Instrucciones de procesamiento y comentarios,
asignar, 91
Integración,
con productos de Altova, 27
en C#, 1100
en C++, 1102
en Java, 1098
Integrar código de MapForce, 1097
Integrating,
MapForce in applications, 1419
Intercalación,
componente de ordenación, 505
intercalación local, 505
punto de código unicode, 505
Intercalación local, 505
Interfaz del usuario, 29
is-not-null,
como función de MapForce (en db functions), 723
is-null,
como función de MapForce (en db functions), 723
is-xsi-nil,
como función de MapForce (en core | node functions), 675
item-at,
como función de MapForce (en core | sequence functions), 696
items-from-till,
como función de MapForce (en core | sequence functions), 697

J

Java, 1438
código, 1067
configuración, 1207
generar código de asignación de datos en, 1083
generar código de programa, 1089
integrar código generado, 1098
ubicación de la biblioteca VM, 1052
JavaScript,
gestión de errores en, 1230
JDBC,
como interfaz de conexión de datos, 161
conectarse a Teradata, 240
configurar una conexión (Windows), 183
control de referencias en el código generado, 1087
Jerarquía,
a partir de archivos de texto, 450
JScript,
code-generation sample, 1244

L

last,
como función de MapForce (en xpath2 | context functions), 782
last-items,
como función de MapForce (en core | sequence functions), 698
leapyear,
como función de MapForce (en lang | dateTime functions), 736
left,
como función de MapForce (en lang | string functions), 766
left-trim,
como función de MapForce (en lang | string functions), 766
Lenguajes de transformación, 24
less,
como función de MapForce (en core | logical functions), 665
Lib,
configuración del compilador, 1207
Licencia, 1595
información sobre, 1593
Licencia del producto de software, 1595
Licencias, 1062
Llamar a gráficos,
hoja de estilos SPS, 916
local-name-from-QName,
como función de MapForce (en lang | QName functions), 679
log,
como función de MapForce (en lang | math functions), 752, 753
logical-and,
como función de MapForce (en core | logical functions), 666
logical-not,
como función de MapForce (en core | logical functions), 666
logical-or,
como función de MapForce (en core | logical functions), 667
logical-xor,
como función de MapForce (en lang | logical functions), 747
Lotes,
automatización de procesamiento, 948
lowercase,
como función de MapForce (en lang | string functions), 767

M

main-mfd-filepath,
como función de MapForce (en core | file path functions), 657
MapForce,
API, 1225
integration, 1419
resumen, 22
MapForce Server,
automatizar asignaciones, 948
compilar asignaciones para, 955
generar excepciones desde, 529
Recursos globales en, 987
MapForceCommand,
in MapForceControl, 1459
MapForceCommands,
in MapForceControl, 1461
MapForceControl, 1462
documentation of, 1419
examples of integration at document level, 1431
integration using C#, 1431
integration using Visual Basic, 1447
object reference, 1459
MapForceControlDocument, 1470
MapForceControlPlaceHolder, 1476
Marcadores,
insertar, 386
margen de marcadores, 386
navegar, 386
quitar, 386
MariaDB,
conectarse por ODBC, 210
match-pattern,
como función de MapForce (en lang | string functions), 768
max,
como función de MapForce (en lang | math functions), 753
max-string,
como función de MapForce (en core | aggregate functions), 636
Medición de licencias,
en los productos de Altova, 1594
Memoria,
insuficiente - excepciones, 1206
Memoria insuficiente, 1206

Mensajes,
iconos de la consulta de BD, 392
ventana de la consulta de BD, 392
MERGE,
instrucción en código SQL generado con MapForce, 294, 301
Método,
reservar nombre, 1206
mfd-filepath,
as MapForce function (in core | file path functions), 657
mff,
archivo de biblioteca, 612
archivo mff.xsd, 612
Microsoft Access,
conectarse por ADO, 168, 212
Microsoft SharePoint Server,
agregar archivos como componentes desde, 77
Microsoft SQL Server,
conectarse por ADO, 214
conectarse por ODBC, 217
millisecond-from-datetime,
como función de MapForce (en lang | datetime functions), 736
millisecond-from-duration,
como función de MapForce (en lang | datetime functions), 737
min,
como función de MapForce (en lang | math functions), 754
min-string,
como función de MapForce (en core | aggregate functions), 637, 638
minute-from- datetime,
como función de MapForce (en lang | datetime functions), 737
minute-from- duration,
como función de MapForce (en lang | datetime functions), 738
Mixto,
asignación basada en origen, 91
asignación de contenido, 91
Modelo de objetos,
Resumen, 1229
Modo,
compatible con v2005R3, 1207
Modo de compatibilidad,
v2005R3, 1207
modulus,
como función de MapForce (en core | math functions), 671
month-from- datetime,
como función de MapForce (en lang | datetime functions), 738
month-from- duration,
como función de MapForce (en lang | datetime functions), 739
Motores,
de los productos de Altova, 1590
multiply,
como función de MapForce (en core | math functions), 672
MySQL,
conectarse por ODBC, 223

N

namespace-uri-from-QName,
como función de MapForce (en lang | QName functions), 680
Navegar,
marcadores, 386
negative,
como función de MapForce (en lang | lógicas functions), 747
Nillable,
no compatible, 1206
node-name,
como función de MapForce (en core | node functions), 676
como función de MapForce (en xpath2 | accessors), 776
node-name (función),
alternativas de uso, 850
Nombre de la aplicación, 1207
Nombres de archivo,
especificar como parámetros de entrada, 897
Nombres de nodo,
asignar, 850
normalize-space,
como función de MapForce (en core | string functions), 711
Notas generales, 22
not-equal,
como función de MapForce (en core | logical functions), 663, 668
not-exists,
como función de MapForce (en core | sequence functions), 699
Novedades,
Versión 2023, 18
now,
como función de MapForce (en lang | dateTime functions), 739

© 2016-2022 Altova GmbH

NULL,
atributo, 131
gestionar en asignaciones de BD, 324
valores, 131
valores en bases de datos, 131
numeric,
como función de MapForce (en lang | logical functions), 748

O

Objeto Application, 1229
Objetos de BD,
agregar a la asignación, 252
filtrar, 252
quitar de la asignación, 252
Objetos globales,
de SPL, 1212
ODBC,
como interfaz de conexión de datos, 161
conectarse a MariaDB, 210
conectarse a Teradata, 242
configurar una conexión, 180
OLE DB,
como interfaz de conexión de datos, 161
Opciones de generación de código,
definir globalmente para todo el proyecto, 120
Oracle,
conectarse por JDBC, 225
conectarse por ODBC, 227
escribir en campos de tipo XML, 369
leer campos de tipo XML, 369
Ordenar,
bases de datos, 356
columnas en la ventana de resultados, 392
componente de ordenación, 505
datos en la ventana de resultados, 392
tablas de la consulta de la BD, 388
Ordenar datos,
componente de ordenación, 505
ORDER BY,
componente WHERE de SQL, 352
Origen, 23

P

pad-string-left,
como función de MapForce (en lang | string functions), 768
pad-string-right,
como función de MapForce (en lang | string functions), 769
Paneles,
Asignación, 35
Consulta de BD, 35
Resultados, 35
Resultados de StyleVision, 35
XQuery, 35
XSLT, 35
Paquete base, 1207
Parámetros,
pasar a la asignación, 477, 482
parse-date,
como función de MapForce (en core | conversion functions), 650
parse-dateTime,
como función de MapForce (en core | conversion functions), 651
parse-number,
como función de MapForce (en core | conversion functions), 653
parse-time,
como función de MapForce (en core | conversion functions), 655
Pasar datos,
sin modificarlos a través de una asignación de valores, 537
PDF,
documentación de la asignación, 915
vista previa del resultado de la asignación como, 911
Período de evaluación,
de los productos de software de Altova, 1593
Personalizar,
comandos, 1045
eliminar comandos, 1045
menú contextual, 1045
Menú predeterminado vs. diseño de MapForce, 1045
menús, 1045
restaurar barras de herramientas, 1045
sombras de menú, 1045
Teclado, 1047
Teclas de acceso rápido, 1047
pi,

© 2016-2022 Altova GmbH

pi,
como función de MapForce (en lang | math functions), 754
Plataformas,
para productos de Altova, 1590
Posición del depurador,
ver valor actual de, 934
position,
como función de MapForce (en core | sequence functions), 700
positive,
como función de MapForce (en lang | lógicas functions), 748
PostgreSQL,
conectarse directamente (de forma nativa), 188
conectarse por ODBC, 233
pow,
como función de MapForce (en lang | math functions), 755
Procedimientos almacenados,
agregar a una asignación, 398
llamar desde una asignación, 400, 404, 408, 413, 417
Procesador XQuery,
de los productos de Altova, 1590
Procesadores XSLT,
de los productos de Altova, 1590
Procesamiento,
automatizar asignaciones, 948
Progress OpenEdge (base de datos),
conectarse por JDBC, 234
conectarse por ODBC, 236
Projects,
Abrir archivo en XMLSpy, 1032
Abrir asignación, 1032
Agregar archivo activo al proyecto, 1032
Agregar archivo de asignación para la operación, 1032
Agregar archivos al proyecto, 1032
Cerrar, 1032
Crear asignación para la operación, 1032
Crear carpeta, 1032
Generar código en, 1032
Generar código para todo el proyecto, 1032
Guardar, 1032
Insertar servicio web, 1032
Propiedades, 1032
Volver a cargar, 1032
Propiedades,
tabla de asignación de valores, 539
Proxy de red,
configuración, 1058
configuración automática, 1058
manual, 1058
opciones, 1058
sistema, 1058
Proyecto,
abrir, 118
agregar asignaciones al, 118
ajustes, 120
buscar, 118
carpetas, 121
como menú de aplicación, 1032
crear, 118
diseños, 118
eliminar, 118
esquema, 118
generación de código, 118
nuevo, 118
opciones de generación de código, 120, 121
organización, 118
propiedades, 121
vista previa de la imagen, 118
Proyectos, 118
Punto de código,
intercalación, 505
Puntos de interrupción,
agregar, 931
eliminar, 931
información, 928
Puntos de interrupción (ventana),
información, 928, 938

Q

QName,
como función de MapForce (en lang | QName functions), 679
QName-as-string,
-as-string como función de MapForce (en lang | QName functions), 758
Quitar,
comentario de bloque, 385
comentario de línea, 385
comentarios, 385
marcadores, 386
regiones, 387

R

radians,
como función de MapForce (en lang | math functions), 755
random,
como función de MapForce (en lang | math functions), 755
RaptorXML Server,
ejecutar una transformación, 947
read-binary-file,
as MapForce function (in lang | file functions), 744
Recursos globales,
Archivo de definiciones, 970
Archivos XML como, 976
bases de datos como, 980
carpetas como, 978
configurar, 970
crear, 970
en entornos de ejecución, 986
en FlowForce Server, 987
implementar en FlowForce Server, 958, 987
introducción a, 969
Referencia, 1024
Referencia de iconos de componentes, 75
Regiones,
contraer, 387
crear, 387
expandir, 387
insertar, 387
quitar, 387
Registrar,
excepciones de BD, 435
Registro,
excepciones de BD, 439
Relaciones de BD,
conservar en asignaciones, 258
definir en asignaciones, 267
Relaciones locales,
y procedimientos almacenados, 411, 413, 417
remove- timezone,
como función de MapForce (en lang | dateTime functions), 740
remove-fileext,
como función de MapForce (en core | file path functions), 658
remove-folder,
como función de MapForce (en core | file path functions), 658
repeat- string,
como función de MapForce (en lang | string functions), 770
replace,
como función de MapForce (en lang | string functions), 770
replace-fileext,
como función de MapForce (en core | file path functions), 659
replicate-item,
como función de MapForce (en core | sequence functions), 703
replicate-sequence,
como función de MapForce (en core | sequence functions), 704
Requisitos de memoria, 1590
Reservar,
nombre de método, 1206
resolve-filepath,
como función de MapForce (en core | file path functions), 659
resolver- uri,
como función de MapForce (en xpath2 | anyURI), 777
Resultado de la asignación,
generar varios archivos como, 892, 896
Resultados, 1038
C#, 1038
C++, 1038
como menú de aplicación, 1038
Configurar la vista Texto, 1038
Ejecutar script SQL/NoSQL, 1038
guardar, 105
Guardar el archivo de salida, 1038
Guardar todos los archivos de salida, 1038
iconos de la consulta de BD, 392
Insertar o quitar marcador, 1038
Java, 1038
Marcador anterior, 1038
Marcador siguiente, 1038
Motor de ejecución integrado, 1038
Quitar todos los marcadores, 1038
Texto XML pretty-print, 1038
validar, 105
Validar archivo de salida, 1038
ventana de la consulta de BD, 392
vista previa, 105
Volver a generar archivo de salida, 1038
XQuery, 1038
XSLT 1.0, 1038

© 2016-2022 Altova GmbH

Resultados, 1038
XSLT 2.0, 1038
XSLT 3.0, 1038
Retener datos,
al pasar por una asignación de valores, 537
reversefind-substring,
como función de MapForce (en lang | string functions), 771
right,
como función de MapForce (en lang | string functions), 772
right-trim,
como función de MapForce (en lang | string functions), 772
round-half-to-even,
como función de MapForce (en xpath2 | numeric functions), 806
round-precision,
como función de MapForce (en core | math functions), 672
RTF,
documentación de la asignación, 915
vista previa del resultado de la asignación como, 911
Ruta de acceso,
convertir en rutas absolutas, 115
en el código generado, 115
Rutas de acceso de archivos,
absolutas, 82
corregir referencias a rutas de acceso rotas, 82
de bases de datos basadas en archivos, 82
en el código generado, 84
en entornos de ejecución, 84
relativas, 82
relativas y absolutas, 84
rotas, 82

S

schemanativetype, 1210
Scripts en XSLT/XQuery,
ver Funciones de extensión, 1570
Sección,
CDATA, 136
Second-from-datetime,
como función de MapForce (en lang | datetime functions), 740
second-from-duration,
como función de MapForce (en lang | datetime functions), 741
Secuencia, 872
Seleccionar,
datos de tabla (consulta de BD), 392
set-empty,
como función de MapForce (en core | sequence functions), 705
set-null,
como función de MapForce (en db functions), 723
set-xsi-nil,
como función de MapForce (en core | node functions), 677
Signo de interrogación,
elementos ausentes, 101
sin,
como función de MapForce (en lang | math functions), 756
skip-first-items,
como función de MapForce (en core | sequence functions), 706
SO,
para productos de Altova, 1590
SPL, 1209
bloques de código, 1209
condiciones, 1215
foreach, 1216
objetos globales, 1212
subrutinas, 1218
usar archivos, 1214
variables, 1211
SPS,
hojas de estilos definidas por el usuario, 921
hojas de estilos predefinidas para documentar asignaciones, 916
SQL,
cargar desde scripts, 382
combinar datos, 336
ejecutar instrucciones, 382, 384
escribir instrucciones, 383
exportar instrucciones como scripts SQL, 384
generar instrucciones, 382, 383
importar scripts SQL, 384
SQL Server,
conectarse por ADO, 168
conectarse por ADO.NET, 173
escribir en campos de tipo XML, 369
leer campos de tipo XML, 369
SQL WHERE,
componente (insertar), 352
ORDER BY, 352
SQL WHERE/ORDER,
componente de MapForce, 356
SQLite,

© 2016-2022 Altova GmbH

SQLite,
asignar datos a, 330, 906
configurar una conexión (Windows), 189
convertir rutas de acceso de BD absolutas en el código generado, 84
escribir archivos XML en, 373
sqrt,
como función de MapForce (en lang | math functions), 756
starts-with,
como función de MapForce (en core | string functions), 712
static-node-annotation,
como función de MapForce (en core | node functions), 677
static-node-name,
como función de MapForce (en core | node functions), 678
string,
como función de MapForce (en core | conversion functions), 655
como función de MapForce (en xpath2 | accessors), 776
string-as- QName,
como función de MapForce (en lang | QName functions), 758
string-compare,
como función de MapForce (en lang | string functions), 773
string-compare-ignore-case,
como función de MapForce (en lang | string functions), 774
string-join,
como función de MapForce (en core | aggregate functions), 638
string-length,
como función de MapForce (en core | string functions), 712
Stylevision,
definir hojas de estilos SPS para asignaciones, 921
substitute-missing,
como función de MapForce (en core | sequence functions), 707
substitute-missing-with-xsi-nil,
como función de MapForce (en core | node functions), 678
substitute-null,
como función de MapForce (en db functions), 724
substring-after,
como función de MapForce (en core | string functions), 713
substring-before,
como función de MapForce (en core | string functions), 714
subtract,
como función de MapForce (en core | math functions), 673
sum,
como función de MapForce (en core | aggregate functions), 639
Superposición de datos,
información, 928
Sybase,
conectarse por JDBC, 239
system-property,
como función de MapForce (en xslt | xslt functions), 837

T

Tabla,
tabla de búsqueda (asignación de valores), 533
Tabla de búsqueda,
propiedades, 539
tabla de asignación de valores, 533
tan,
como función de MapForce (en lang | math functions), 757
Teradata,
conectarse por JDBC, 240
conectarse por ODBC, 242
Texto,
archivos de texto (definir campos clave), 450
asignación de datos de archivos de texto, 445
time-from-datetime,
como función de MapForce (en lang | datetime functions), 741
timezone,
como función de MapForce (en lang | dateTime functions), 742
Tipo complejo,
ordenar, 505
Tipo simple,
ordenar, 505
Tipos,
derivados (xsi:type), 129
integrados, 1220
Tipos de asignaciones, 25
Tipos de conexión,
basada en destino, 91
basada en origen, 91
basadas en destino con contenido mixto, 91
conexiones basadas en el destino vs. conexiones basadas en el destino, 91
de copia total, 95
estándar, 91
estándar con contenido mixto, 91
mixta, 91
secundarios equivalentes, 94
Tipos derivados,

© 2016-2022 Altova GmbH

Tipos derivados,
asignaciones, 129
tokenize,
como función de MapForce (en core | string functions), 715
tokenize-by-length,
como función de MapForce (en core | string functions), 717
tokenize-regexp,
como función de MapForce (en core | string functions), 720
Transacciones de BD,
habilitar para procedimientos almacenados, 408
Transformaciones,
RaptorXML Server, 947
Transformar,
datos de entrada (asignación de valores), 533
translate (in core | string functions),
como función de MapForce, 721
Transmisión por secuencias de datos,
definición, 77
true,
como función de MapForce (en xpath2 | boolean functions), 778

U

unary-minus,
como función de MapForce (en lang | math functions), 757
Unicode,
intercalación de punto de código, 505
reemplazar caracteres especiales, 275
unparsed-entity-uri,
como función de MapForce (en xslt | xslt functions), 838
uppercase,
como función de MapForce (en lang | string functions), 775
URI,
en DTDs, 124
URI de espacio de nombres,
DTD, 124
URL,
agregar archivos como componentes desde, 77
Usar nombres de objeto relativos a un esquema predeterminado,
como opción, 255
Uso de Internet,
en los productos de Altova, 1591

V

Validar,
diseño de asignación, 105
resultados de la asignación, 105
Valores (ventana),
información, 928, 934
pestaña Contexto, 934
pestaña Historial, 934
pestaña Relacionados, 934
Valores null,
gestionar en componentes de BD, 276
Valores predeterminados, 558
crear, 560
Valores predeterminados y funciones de nodo,
anotación, 572
aplicar condicionalmente, 563
ayudas visuales, 560
casos de uso, 563
configurar, 560
configurar reglas, 560
crear, 560
filtro, 563
fractionDigits, 572
iconos de las funciones, 558
lado de entrada, 558
lado de salida, 558
length, 572
maxLength, 572
metadatos, 572
minLength, 572
precision, 572
reglas de bloqueo, 563
scale, 572
sobrescribir reglas heredadas, 563
totalDigits, 572
valores predeterminados para nodos sin conexión, 563
Variables,
agregar a la asignación, 494
cambiar el ámbito de, 498
de SPL, 1211
ejemplos de uso, 500, 501, 503
introducción, 492
Ventanas,
Administrar bibliotecas, 30

© 2016-2022 Altova GmbH

Ventanas,
Asignación, 30
Bibliotecas, 30
Messages, 34
Proyecto, 30
Ventana de asignaciones múltiples, 30
Vista general, 30
VÍNCULO EXTERNO IZQUIERDO,
en componentes de combinación, 336
VÍNCULO INTERNO,
en componentes de combinación, 336
Vista, 1042
Adelante, 1042
Administrar bibliotecas, 1042
Atrás, 1042
Barra de estado, 1042
Bibliotecas, 1042
comando de menú, 1042
como menú de aplicación, 1042
Depurar ventanas, 1042
Mensajes, 1042
Mostrar anotaciones, 1042
Mostrar biblioteca en el título de la función, 1042
Mostrar información rápida, 1042
Mostrar los conectores del componente seleccionado, 1042
Mostrar los conectores desde su origen a su destino, 1042
Mostrar tipos, 1042
Opciones de visualización XBRL, 1042
Ventana proyectos, 1042
Vista general, 1042
Zoom, 1042
Vista Explorador,
generar instrucciones SQL, 390
opciones del menú contextual, 390
Vista Texto,
ajuste automático de línea, 108
Buscar, 111
formato pretty-print, 108
guías de sangría, 108
marcadores de espacios en blanco, 108
marcadores de final de línea, 108
navegar, 108
números de línea, 108
plegamiento de código, 108
resaltado de texto, 108
ventana Información, 108
zoom, 108
Visual Basic,
gestión de errores en, 1230
integration of MapForce, 1447
Visual Studio,
adding the MapForce ActiveX Controls to the toolbox, 1422
generar código de asignación de datos para, 1071, 1076

W

WebDAV Server,
agregar archivos como componentes desde, 77
weekday,
como función de MapForce (en lang | dateTime functions), 742
weeknumber,
como función de MapForce (en lang | dateTime functions), 743
WHERE,
componente WHERE de SQL, 352
Windows,
Cascada, 1061
compatibilidad con productos de Altova, 1590
Cuadro de diálogo de Windows, 1061
En mosaico vertical/horizontal, 1061
Tema, 1061
Tema clásico, 1061
Tema ligero, 1061
Tema oscuro, 1061
Word,
documentación de la asignación, 915
Word 2007+,
vista previa del resultado de la asignación como, 911
write-binary-file,
as MapForce function (in lang | file functions), 745

X

Xerces,
compatibilidad, 1207
XML,
asignar datos CSV a XML, 445
como destino de la asignación, 445
configuración de codificación, 125
configuración del componente, 125
declaración, 125

XML,
escribir en campo de BD, 373
firma digital, 125
versión del esquema, 115
XML data,
escribir en campos de tipo XML, 369
leer campos de tipo XML, 369
XQuery,
agregar funciones personales, 600, 601
funciones de extensión, 1570
importar módulos, 601
xs: any (xs:anyAttribute), 137
xsi:nil,
como atributo en una instancia XML, 131
xsi:type,
asignación de tipos derivados, 129
XSLT,
agregar funciones personales, 594
espacio de nombres de plantilla, 594
funciones de extensión, 1570
quitar funciones personales, 594

Y

year-from- duration,
como función de MapForce (en lang | datetime functions), 744
year-from-datetime,
como función de MapForce (en lang | datetime functions), 743

Z

Z to A,
componente de ordenación, 505

© 2016-2022 Altova GmbH