# Altova MapForce 2025 Professional Edition

## Manual del usuario y referencia

# Altova MapForce 2025 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: 2025

© 2019-2025 Altova GmbH

# Contenido


## 1 Introducción 12

1.1 Novedades 13
- 1.1.1 Versión 2025 13
- 1.1.2 Versión 2024 13
- 1.1.3 Versión 2023 14
- 1.1.4 Versión 2022 15
- 1.1.5 Versión 2021 16

1.2 ¿Qué es MapForce? 17
- 1.2.1 Asignación: origen y destino 18
- 1.2.2 Tipos de asignaciones 19
- 1.2.3 Lenguajes de transformación 19
- 1.2.4 Integración con productos de Altova 20

1.3 Interfaz del usuario 22
- 1.3.1 Barras de herramientas 23
- 1.3.2 Ventanas 23
- 1.3.3 Ventana Mensajes 27
- 1.3.4 Paneles 28

## 2 Aspectos fundamentales de las asignaciones de datos 32

2.1 Componentes 34
- 2.1.1 Agregar componentes 38
- 2.1.2 Componentes básicos 41
- 2.1.3 Rutas de acceso de archivos 44

2.2 Conexiones 48
- 2.2.1 Tipos de conexión 52
- 2.2.2 Configuración de la conexión 59
- 2.2.3 Menú contextual de las conexiones 60
- 2.2.4 Conexiones defectuosas 62

2.2.5 Conservar conexiones tras eliminación de componentes ... 63
2.3 Procedimientos y funciones generales ... 66
2.3.1 Validación ... 66
2.3.2 Generación de código ... 68
2.3.3 Características de la vista Texto ... 71
2.3.4 Búsquedas en la vista Texto ... 74
2.3.5 Configuración de la asignación ... 78
2.4 Reglas y estrategias básicas ... 80
2.4.1 Secuencias ... 81
2.4.2 Contexto y orden de procesamiento ... 82
2.4.3 Contexto primario ... 85
2.4.4 Contexto de prioridad ... 89
2.4.5 Varios componentes de destino ... 94
2.5 Proyectos ... 98
2.5.1 Aspectos básicos de un proyecto ... 98
2.5.2 Configuración de proyectos ... 100
2.5.3 Carpetas de proyecto ... 101

## 3 Tutoriales 103

3.1 De esquema a esquema ... 104
3.1.1 Crear y guardar diseños ... 106
3.1.2 Agregar un componente de origen ... 107
3.1.3 Agregar un componente de destino ... 108
3.1.4 Conectar origen y destino ... 110
3.1.5 Vista previa del resultado de la asignación ... 112
3.2 Varios archivos de origen a un solo destino ... 114
3.2.1 Preparar el diseño de la asignación ... 115
3.2.2 Agregar segundo archivo de origen ... 116
3.2.3 Configurar componentes de destino ... 117
3.2.4 Conectar varios orígenes a un destino ... 117
3.3 Asignación en cadena ... 119
3.3.1 Preparar el diseño de la asignación ... 119
3.3.2 Configurar el segundo archivo de destino ... 120
3.3.3 Conectar componentes de destino ... 121

3.3.4 Filtrar datos ... 122
3.3.5 Previsualizar y guardar resultados ... 125
3.4 Varios archivos de origen a varios archivos de destino ... 128
3.4.1 Configurar el componente de entrada ... 130
3.4.2 Configurar el componente de destino, parte 1 ... 131
3.4.3 Configurar el componente de destino, parte 2 ... 134

4 Componentes estructurales 137
4.1 XML y esquemas XML ... 138
4.1.1 Configuración de componentes XML ... 139
4.1.2 Tipos derivados ... 144
4.1.3 Valores NULL ... 146
4.1.4 Comentarios e instrucciones de procesamiento ... 149
4.1.5 Secciones CDATA ... 150
4.1.6 Comodines: xs:any / xs:anyAttribute ... 151
4.1.7 Espacios de nombres personalizados ... 154
4.1.8 Gestor de esquemas ... 157
4.2 Bases de datos ... 174
4.2.1 Conectarse a un origen de datos ... 178
4.2.2 Procedimientos generales ... 246
4.2.3 Acciones de tabla de BD ... 274
4.2.4 Panel Consulta de BD ... 294
4.2.5 Asignaciones entre datos XML y campos de BD ... 304
4.2.6 Procedimientos almacenados ... 315
4.3 Archivos CSV y archivos de texto ... 344
4.3.1 Ejemplo: asignar archivos CSV a XML ... 344
4.3.2 Ejemplo: recorrer elementos ... 346
4.3.3 Ejemplo: crear jerarquías a partir de archivos CSV y FLF ... 349
4.3.4 Opciones de configuración de componentes CSV ... 353
4.3.5 Asignar archivos FLF a bases de datos ... 357
4.3.6 Opciones de configuración de componentes FLF ... 361

5 Componentes de transformación 368

5.1 Entrada simple ... 369
5.1.1 Agregar componentes de entrada simples ... 370
5.1.2 Configurar componentes de entrada simples ... 371
5.1.3 Crear un valor de entrada predeterminado ... 373
5.1.4 Ejemplo: usar nombres de archivo como parámetros de asignación ... 374
5.2 Salida simple ... 380
5.2.1 Agregar componentes de salida simples ... 381
5.2.2 Ejemplo: vista previa de resultados de una función ... 382
5.3 Variables ... 384
5.3.1 Agregar variables ... 386
5.3.2 Contexto y ámbito de las variables ... 390
5.3.3 Ejemplo: contar filas de tabla de BD ... 393
5.3.4 Ejemplo: filtrar y numerar nodos ... 394
5.3.5 Ejemplo: crear grupos y subgrupos de registros ... 395
5.4 Componentes de combinación ... 397
5.4.1 Agregar condiciones de combinación ... 399
5.4.2 Combinar tres o más estructuras ... 402
5.4.3 Ejemplo: combinar estructuras XML ... 403
5.4.4 Combinar datos de BD ... 408
5.5 Componentes de ordenación ... 426
5.5.1 Ordenar según varias claves ... 428
5.5.2 Ordenar con variables ... 430
5.6 Filtros y condiciones ... 432
5.6.1 Ejemplo: filtrar nodos ... 434
5.6.2 Ejemplo: devolver un valor de forma condicional ... 436
5.6.3 Filtrar y ordenar datos de BD ... 437
5.7 Asignación de valores ... 445
5.7.1 Ejemplo: reemplazar días de la semana ... 450
5.7.2 Ejemplo: reemplazar puestos de trabajo ... 453
5.8 Excepciones ... 457
5.8.1 Ejemplo: excepción en la condición Greater than ... 458
5.8.2 Ejemplo: excepción cuando un nodo no existe ... 458
5.9 Funciones ... 461
5.9.1 Fundamentos de las funciones ... 461

5.9.2 Gestionar bibliotecas de funciones ... 464
5.9.3 Valores predeterminados y funciones de nodo ... 469
5.9.4 Funciones definidas por el usuario ... 485
5.9.5 Funciones personales ... 507
5.9.6 Expresiones regulares ... 540
5.9.7 Referencia de la biblioteca de funciones ... 544

## 6 Asignaciones avanzadas 762

6.1 Asignar nombres de nodos ... 763
6.1.1 Obtener acceso al nombre de los nodos ... 764
6.1.2 Obtener acceso a determinado tipo de nodos ... 771
6.1.3 Ejemplo: asignar nombres de elemento a valores de atributo ... 776
6.1.4 Ejemplo: agrupar y filtrar nodos según su nombre ... 779
6.2 Procesar archivos por lotes ... 784
6.2.1 Ejemplo: dividir un archivo XML en varios archivos ... 787
6.2.2 Ejemplo: dividir una tabla de BD en varios archivos XML ... 788
6.3 Analizar y serializar cadenas de texto ... 791
6.3.1 Información sobre el componente de análisis/serialización ... 791
6.3.2 Ejemplo: serializar datos a una cadena (de XML a BD) ... 794

## 7 Documentación de la asignación 799

7.1 Hojas de estilos predefinidas de StyleVision ... 801
7.2 Hojas de estilos personalizadas ... 806

## 8 Depuración 808

8.1 Preparar la depuración ... 811
8.2 Información sobre el modo depurador ... 812
8.3 Agregar y quitar puntos de interrupción ... 815
8.4 Ventana Valores ... 818
8.5 Ventana Contexto ... 820
8.6 Ventana Puntos de interrupción ... 822
8.7 Vista previa parcial de resultados generados ... 824
8.8 Ver el valor actual de un conector ... 825

8.9 Retroceder en el pasado reciente ... 826
8.10 Ver historial de valores procesados por un conector ... 827
8.11 Establecer el contexto en un valor ... 829

## 9 Automatizar asignaciones con productos de Altova 830

9.1 Automatización con RaptorXML Server ... 831
9.2 Automatización con MapForce Server ... 832
9.3 Preparar asignaciones para ejecutarlas en servidores ... 833
9.4 Compilar asignaciones en archivos de ejecución de MapForce Server ... 839
9.5 Implementar asignaciones en FlowForce Server ... 842
9.6 Paneles de resultados de StyleVision ... 847
9.7 Interfaz de la línea de comandos de MapForce ... 851

## 10 Recursos globales de Altova 856

10.1 Configurar recursos globales - parte 1 ... 857
10.2 Configurar recursos globales - parte 2 ... 859
10.3 Archivos XML como recursos globales ... 863
10.4 Carpetas como recursos globales ... 865
10.5 Bases de datos como recursos globales ... 867
10.6 Resultados de transformación como recursos globales ... 869
10.7 Recursos globales en entornos de ejecución ... 873
10.7.1 Recursos globales en el código generado ... 873
10.7.2 Recursos globales en MapForce Server ... 874
10.7.3 Recursos globales en FlowForce Server ... 874

## 11 Catálogos en MapForce 877

11.1 Funcionamiento de los catálogos ... 878
11.2 Estructura de los catálogos de MapForce ... 880
11.3 Personalizar catálogos ... 882
11.4 Variables de entorno ... 884

## 12 Complemento de MapForce para Visual Studio 886

## 13 Complemento de MapForce para Eclipse 889

13.1 Instalación del complemento de MapForce para Eclipse ... 890
13.2 Perspectiva MapForce ... 892
13.3 Acceso a menús y funciones frecuentes ... 895
13.4 Trabajar con asignaciones y proyectos ... 899
13.4.1 Crear un proyecto MapForce/Eclipse ... 899
13.4.2 Crear asignaciones nuevas ... 901
13.4.3 Importar asignaciones a un proyecto de Eclipse ... 903
13.4.4 Configurar la generación automática de código MapForce ... 906
13.5 Ampliar el complemento de MapForce para Eclipse ... 909

## 14 Generador de código 911

14.1 Generar, compilar y ejecutar código ... 913
14.2 Integrar código generado ... 919
14.2.1 Modificar datos de entrada/salida, definir gestión de errores ... 919
14.2.2 Cambiar tipo de datos de entrada/salida ... 922
14.2.3 Generar código a partir de esquemas XML o DTD ... 928
14.2.4 Clases generadas (C++) ... 972
14.2.5 Clases generadas (C#) ... 988
14.2.6 Clases generadas (Java) ... 1004
14.2.7 Referencia de la SPL ... 1020

## 15 API COM de MapForce 1040

## 16 ActiveX Integration 1041

16.1 Prerequisites ... 1042
16.2 Adding the ActiveX Controls to the Toolbox ... 1044
16.3 Integration at Application Level ... 1046
16.4 Integration at Document Level ... 1049
16.5 ActiveX Integration Examples ... 1052
16.5.1 C# ... 1052

16.5.2 Java ... 1059
16.5.3 VB.NET ... 1068
16.6 Command Reference ... 1072
16.6.1 "File" Menu ... 1072
16.6.2 "Edit" Menu ... 1073
16.6.3 "Insert" Menu ... 1074
16.6.4 "Project" Menu ... 1074
16.6.5 "Component" Menu ... 1075
16.6.6 "Connection" Menu ... 1076
16.6.7 "Function" Menu ... 1077
16.6.8 "Output" Menu ... 1077
16.6.9 "Debug" Menu ... 1078
16.6.10 "View" Menu ... 1079
16.6.11 "Tools" Menu ... 1079
16.6.12 "Window" Menu ... 1080
16.6.13 "Help" Menu ... 1080
16.7 Object Reference ... 1082
16.7.1 MapForceCommand ... 1082
16.7.2 MapForceCommands ... 1084
16.7.3 MapForceControl ... 1085
16.7.4 MapForceControlDocument ... 1093
16.7.5 MapForceControlPlaceHolder ... 1099
16.7.6 Enumerations ... 1102

## 17 Comandos de menú 1103

17.1 Archivo ... 1104
17.2 Edición ... 1107
17.3 Insertar ... 1108
17.4 Proyecto ... 1111
17.5 Componente ... 1113
17.6 Conexión ... 1115
17.7 Función ... 1116
17.8 Resultados ... 1117
17.9 Depuración ... 1119

17.10 Vista ... 1121
17.11 Herramientas ... 1123
17.11.1 Personalizar menús ... 1124
17.11.2 Personalizar teclas de acceso rápido ... 1125
17.11.3 Opciones ... 1128
17.12 Ventanas ... 1142
17.13 Ayuda ... 1143

## 18 Anexos 1148

18.1 Notas sobre compatibilidad ... 1149
18.1.1 Orígenes y destinos de datos compatibles ... 1149
18.1.2 Funciones compatibles en el código generado ... 1150
18.2 Información sobre motores ... 1153
18.2.1 Información sobre motores XSLT y XQuery ... 1153
18.2.2 Funciones XSLT y XPath/XQuery ... 1159
18.3 Datos técnicos ... 1258
18.3.1 Requisitos de SO y memoria ... 1258
18.3.2 Motores XSLT y XQuery de Altova ... 1258
18.3.3 Compatibilidad con Unicode ... 1259
18.3.4 Uso de Internet ... 1259
18.4 Información sobre licencias ... 1261
18.4.1 Distribución electrónica de software ... 1261
18.4.2 Activación del software y medición de licencias ... 1262
18.4.3 Contrato de licencia para el usuario final ... 1263
18.4.4 Empaquetar archivos de licencia con el instalador de MapForce ... 1263

## Índice 1264

# 1 Introducción

Altova MapForce 2025 Professional Edition es una potente herramienta de transformación de datos y ETL para proyectos de integración de datos. MapForce es una aplicación de 32/64 bits para Windows compatible con Windows 10, Windows 11 y Windows Server 2016 o superior. Las ediciones Enterprise y Professional son compatibles con plataformas de 64 bits.

MapForce le permite realizar conversiones de datos entre prácticamente cualquier tipo de formato. MapForce dispone de una interfaz gráfica 22 que incluye numerosas opciones para administrar, visualizar, manipular y ejecutar asignaciones de datos independientes y proyectos de asignación de datos complejos. Para la transformación de datos, MapForce ofrece una amplia biblioteca de funciones de procesamiento y conversión de datos 461 con las que puede filtrar y manipular los datos según las necesidades de su proyecto de integración de datos.

En cuanto haya terminado de diseñar su asignación de datos, podrá previsualizar el resultado en un panel separado y guardarlo en la ubicación deseada. Además, puede generar código 68 para ejecutarlo en una ubicación externa.

También puede ampliar la funcionalidad de MapForce integrándolo con otros productos de Altova:

- Puede ejecutar sus asignaciones de datos con MapForce Server. Esto le ayudará a automatizar las operaciones empresariales que requieren transformaciones de datos reiterativas. MapForce Server incluye un potente motor de transformación de datos y puede realizar conversiones de datos de cualquier tipo. Es importante destacar que se trata de un servidor multiplataforma que está disponible en Windows, macOS y Linux.
- RaptorXML Server es un motor ultrarrápido que valida sus instancias.
- FlowForce Server le ayuda a automatizar sus tareas y le permite ejecutar sus asignaciones de datos como trabajos programados.
- StyleVision Server genera resultados en HTML, RTF, PDF y Word.
- Puede utilizar StyleVision para diseñar hojas de estilo StyleVision Power Stylesheets que permiten a StyleVision Server generar resultados en múltiples formatos.
- DatabaseSpy es una herramienta versátil que le permite diseñar, editar y consultar bases de datos.
- XMLSpy es especialmente útil a la hora de editar sus archivos de asignación de datos. Algunos cuadros de diálogo de MapForce le permiten abrir archivos directamente en XMLSpy.
- También puede utilizar MapForce como complemento de Microsoft Visual Studio y Eclipse. Esto le permite acceder a la funcionalidad de MapForce sin salir de su entorno de desarrollo preferido.

*Última actualización: 20 marzo 2025*

 *© 2019-2025 Altova GmbH*

# 1.1 Novedades

En esta sección, se describen las características y funciones de cada nueva versión de MapForce. Para más detalles, consulte la sección correspondiente.

## 1.1.1 Versión 2025

### Versión 2025 Release 2

- Compatibilidad con las API de Shopify/GraphQL (*Enterprise Edition*).
- Compatibilidad con MySQL 9, DB2 12.1 y SQLite 3.47.2 (*Professional y Enterprise Edition*). Para saber más, consulte el apartado Bases de datos 174.
- Mejoras y actualizaciones internas.

### Versión 2025

- Compatibilidad con la especificación OpenAPI 2.0, 3.0 y 3.1 para API HTTP (*Enterprise Edition*).
- Ahora PDF Extractor le permite buscar texto tanto en la interfaz de usuario como en tiempo de ejecución (*Enterprise Edition*).
- Ahora es posible generar automáticamente mensajes EDIFACT CONTRL que se envían en respuesta a los mensajes EDIFACT recibidos (*Enterprise Edition*).
- Compatibilidad con UN/EDIFACT 2023A (*Enterprise Edition*).
- Compatibilidad con SWIFT 2024 (*Enterprise Edition*).
- Compatibilidad con Azure CosmosDB (*Enterprise Edition*). Para saber más, consulte el apartado Bases de datos 174.
- Mejoras y actualizaciones internas.

## 1.1.2 Versión 2024

### Versión 2024 Release 2

- Ahora es posible acceder a varios tutoriales en vídeo en el proyecto **`MapForceExamples`** (*ediciones Enterprise y Professional*). Además, puede añadir sus propios enlaces a recursos externos. Para más detalles consulte el apartado Aspectos básicos de un proyecto 100.
- Al implementar su asignación en FlowForce Server, puede elegir anexar los archivos de asignación para recuperarlos después. Esto evitará que pierda sus archivos de asignación y le permitirá descargarlos en cualquier momento (*ediciones Professional y Enterprise*). Para más información consulte el apartado Implementar asignaciones en FlowForce Server 842.
- Los componentes de BD ahora pueden compartir la misma conexión de base de datos en tiempo de ejecución (*ediciones Enterprise y Professional*). Para más detalles consulte el apartado Configurar componentes de BD 258.
- Ahora es posible crear una asignación a partir de valores de tipos de enumeración en XML (*todas las ediciones*) y componentes XBRL (*Enterprise Edition*). Gracias a esta función le resultará más fácil asignar valores de enumeración: Ambos lados de la asignación de valores se rellenan previamente con

todos los valores de enumeración y sólo tendrá que revisar y editar los valores relevantes de la asignación. Para más información consulte el apartado Asignación de valores 449.

- La generación de código C# es compatible con NET 8.0 (*ediciones Professional y Enterprise*). Para más información consulte el apartado Generación de código 68.
- Compatibilidad con mensajes FORTRAS EDI (*Enterprise Edition*).
- Compatibilidad con PostgreSQL 16, MySQL 8.2, MySQL 8.3, MariaDB 11.2, SQLite 3.45 (*ediciones Professional y Enterprise*). Para más detalles consulte el apartado Bases de datos 174.
- Mejoras y actualizaciones internas.

### Versión 2024

- Otra utilidad nueva de MapForce que ya está disponible es PDF Extractor (*Enterprise Edition*). PDF Extractor sirve para crear plantillas de extracción PDF que puede importar a MapForce y utilizar como componentes de origen en las asignaciones de datos.
- En MapForce ahora es posible crear asignaciones de datos basadas en IA (*Enterprise Edition*). MapForce permite crear llamadas a servicios web (de tipo REST) a una API, como la API OpenAI, la API OpenAI de Azure, los servicios de IA de AWS, etc.
- Compatibilidad con SWIFT 2023 (*Enterprise Edition*).
- Ahora está disponible una nueva función de suspensión, **`sleep`**, que permite pasar datos tras un retraso especificado (*Ediciones Enterprise y Professional*). Para más información, consulte `sleep` 694.
- Las bases de datos MySQL y MariaDB ahora tienen compatibilidad nativa (*ediciones Professional y Enterprise*). Para más información sobre bases de datos compatibles, consulte el apartado Bases de datos 174.
- Se ha mejorado y ampliado la funcionalidad de las conexiones de secundarios equivalentes, incorporando nuevas opciones de combinación. Para más detalles, consulte Conectar los secundarios equivalentes 55.
- Además del tipo de concesión *Código de autorización*, los tipos *Credenciales de cliente* y *Credenciales de contraseña del propietario de recursos* ahora también son compatibles con las credenciales OAuth (*Enterprise Edition*).
- Mejoras y actualizaciones internas.

## 1.1.3 Versión 2023

### Versión 2023 Release 2

- Ahora puede generar la declaración `standalone="yes"` en la declaración XML de los archivos XML de destino. Para más detalles consulte Configuración de componentes XML 142.
- Hemos reorganizado la Ayuda 1143 para que pueda acceder a la Ayuda en línea de forma predeterminada, aunque sigue teniendo la opción de usar el archivo PDF local 1128.
- Ahora puede añadir comentarios de estilo post-it a una asignación. Para más información consulte Comentarios 36.
- Hemos añadido opciones de configuración de red 1138.
- Ahora MapForce también es compatible con mensajes VDA EDI (*Enterprise Edition*).
- Mejoras y actualizaciones internas.

### Versión 2023


© 2019-2025 Altova GmbH

- Hemos añadido compatibilidad con estos temas: *Clásico*, *Ligero* y *Oscuro*. Visite el apartado Ventanas 1142 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 889.
- 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 494 basados en BD y variables 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 174.
- Compatibilidad con el Gestor de esquemas XML 157, 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.

## 1.1.4 Versión 2022

### Versión 2022 Release 2

- Actualizaciones y mejoras internas
- Hemos ampliado la compatibilidad con Eclipse a las versiones: 2021- 12, 2021- 09; 2021- 06; 2021- 03 (*ediciones Professional y Enterprise*). Para más detalles, consulte al apartado Complemento de MapForce para Ecplise 889.
- El complemento de MapForce para Visual Studio y la generación de código son compatibles con Visual Studio 2022 (*ediciones Professional y Enterprise*). Para más información, véase los apartados Complemento de MapForce para Visual Studio 886 y Generación de código.
- La generación de código es compatible con NET 6.0 (*ediciones Professional y Enterprise*). Para más información consulte el apartado Generación de código.
- 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 174.
- Ahora se puede generar una vista previa de las imágenes en la ventana **Proyecto** (*ediciones Professional y Enterprise*). Para más información, véase el apartado Aspectos básicos de un proyecto de datos 100.
- Ahora es posible crear indicadores de tramitación que son conformes con la EBA para los componentes XBRL de destino (*Enterprise Edition*).

### Versión 2022

- Actualizaciones y mejoras internas
- Hemos ampliado la compatibilidad con Eclipse a las versiones: 2021- 09; 2021- 06; 2021- 03; 2020- 12 (*ediciones Professional y Enterprise*). Para más detalles, consulte al apartado Complemento de MapForce para Ecplise 889.

- Las conexiones de copia total 57 ahora también son compatibles con JSON. Esta funcionalidad solo está disponible para tipos JSON compatibles (*Enterprise Edition*).
- En StyleVision existe un panel de resultados nuevo 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 (*ediciones Professional y Enterprise*). Para más información, consulte Paneles de resultados de StyleVision 847.
- Compatibilidad con JSON Schema en variables y parámetros 492 (*Enterprise Edition*).
- Compatibilidad con bases de datos NoSQL: MongoDB y CouchDB (*Enterprise Edition*). Para más información sobre la compatibilidad con bases de datos consulte Bases de datos 174.
- Ahora hay disponible una biblioteca de funciones `bson` que permite crear y manipular tipos BSON (*Enterprise Edition*).
- Compatibilidad con los directorios EDIFACT/ONU D.20B y D.21A.
- Compatibilidad con SWIFT 2021.

## 1.1.5 Versión 2021

### Versión 2021 Release 3

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

### Versión 2021 Release 2

- Ahora se puede usar XSLT 3.0 como lenguaje de asignación, véase Generar código XSLT 68. Además, MapForce ahora incluye funciones integradas nuevas compatibles con XSLT 3.0, véase la referencia de la biblioteca de funciones 544.
- Al generar código en C# puede seleccionar como destino .NET Core 3.1 y .NET 5.0 desde las opciones de generación de código (además de la compatibilidad que ya existe con proyectos .NET Framework) . Para más detalles consulte Generar código C#.
- 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 662 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 combinar varias tablas o vistas de BD con componentes de combinación SQL en una asignación puede usar tanto combinación externa izquierda como combinación interna; consulte Combinaciones en modo SQL 411.
- Otras mejoras y actualizaciones internas.

 © 2019-2025 Altova GmbH

# 1.2 ¿Qué es MapForce?

**Sitio web de Altova:** Herramienta de asignación de datos

MapForce es una potente herramienta gráfica que permite convertir e integrar cualquier combinación de datos. Consulte Asignación: origen y destino 18 para ver una lista completa de los formatos disponibles. Una asignación consiste en uno o más orígenes de datos y uno o más destinos de datos 34. La asignación puede incluir uno o más componentes de transformación 35 con los que cuenta un amplio número de opciones de procesamiento y filtrado. Para saber más sobre diferentes tipos de asignaciones, consulte Tipos de asignaciones 19 y Tutoriales 103.

Para poder ejecutar una asignación debe indicar una estructura de datos para 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.

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

## Modelo abstracto

El modelo abstracto que ve a continuación ilustra una de las transformaciones de datos básicas que puede realizar con MapForce. El esquema de origen describe la estructura de la instancia de origen. El esquema de destino describe la estructura de la instancia de destino. En función de sus necesidades los esquemas de origen y de destino pueden tener la misma estructura o estructuras distintas. Al conectar el origen y el destino, la asignación genera código de transformación (en el lenguaje de transformación 19 seleccionado) que lee datos de la instancia de origen y los escribe en la de destino. Para ver cómo se implementa este modelo de transformación de datos en un ejemplo concreto, consulte el Tutorial 1 104.

En situaciones reales, puede mezclar y unir cualquier combinación de fuentes de datos (p.ej., XML, EDI y archivos de texto) y asignarlos a cualquier combinación de destinos de datos (p.ej., una base de datos y un archivo Excel).

### Notas generales

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

- `C:\Usuario\<usuario>\Documentos\Altova\MapForce2025\MapForceExamples`
- `C:\Usuario\<usuario>\Documentos\Altova\MapForce2025\MapForceExamples\Tutorial`
- `C:\Usuario\<usuario>\Documentos\Altova\MapForce2025\MapForceExamples\Tutorial\BasicTutorials`

### En esta sección

En esta sección encontrará:


- Asignación: origen y destino 18
- Tipos de asignaciones 19
- Lenguajes de transformación 19
- Integración con productos de Altova 20


## 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)
- 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)
- 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
- Bases de datos NoSQL
- Archivos binarios (contenido BLOB sin formato)
- Estándares EDI
- Archivos JSON
- Archivos de Microsoft Excel 2007 y posteriores
- Archivos de instancia y taxonomías XBRL
- Protocol Buffers

 © 2019-2025 Altova GmbH

- Archivos PDF basados en plantillas PDF creadas en PDF Extractor (sólo se pueden usar como fuentes de datos)

### 1.2.2 Tipos de asignaciones

**Sitio web de Altova:** Vídeos de demostración de MapForce

El grado de complejidad de una asignación puede variar en función de sus necesidades o requisitos empresariales: Por ejemplo, puede que necesite configurar su asignación para (i) leer datos de un solo origen y escribir estos datos en varios destinos o para (ii) combinar datos de varios orígenes en un solo destino. Se pueden usar distintas estructuras de datos como origen y destino: p.ej. archivos XML, bases de datos, archivos EDI, etc. Para obtener más información sobre formatos de origen y destino consulte Asignación: origen y destino 18

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

- Asignar un archivo de origen a un archivo de destino. Visite Tutorial 1 104 para obtener más información.
- Combinar varios orígenes de datos en un único destino. Visite Tutorial 2 114 para obtener más información.
- Asignar datos de un solo origen al primer archivo de destino y, a continuación, filtrar los datos de forma que sólo un subconjunto de estos se asigne al segundo archivo de destino. Consulte Tutorial 3 119.
- Asignar varios archivos de origen a varios archivos de destino. Consulte Tutorial 4 128.

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 más información sobre los términos y las funcionalidades básicas de MapForce, consulte Fundamentos de la asignación de datos 32 e Interfaz del usuario 22.

*Proyectos (ediciones Professional y Enterprise)*
Para administrar y acceder a los diseños de las asignaciones de datos de forma más fácil puede organizarlos en proyectos. Además de generar código para cada asignación individual en el proyecto, podrá generar código de programa para el proyecto entero. Para más detalles consulte el apartado Proyectos 98.

### 1.2.3 Lenguajes de transformación

En MapForce los lenguajes de transformación sirven para generar el código de transformación que ejecuta las asignaciones. Puede seleccionar o modificar el lenguaje de transformación en cualquier momento. También puede generar código de programa con el comando de menú **Archivo | Generar código en** o **Archivo | Generar código en el lenguaje seleccionado** y usar ese código para llevar a cabo transformaciones de datos fuera de MapForce. Para más información consulte Generación de código 68.

Según la edición de MapForce puede elegir uno de estos lenguajes para la transformación de datos:

| Basic Edition | ediciones Enterprise y Professional |
|---|---|
| • XSLT 1.0<br>• XSLT 2.0<br>• XSLT 3.0 | • XSLT 1.0<br>• XSLT 2.0<br>• XSLT 3.0<br>• BUILT-IN<br>• XQuery<br>• Java<br>• C#<br>• C++ |

Si selecciona XSLT 1-3 o XQuery como lenguaje de transformación, podrá ver el código de transformación en un panel separado de MapForce.

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 1149.

Durante el proceso de diseño o la vista previa, MapForce valida constantemente la integridad de los esquemas y de las transformaciones. Si hay errores de validación, los muestra en la ventana Mensajes 27. Esto puede resultar muy útil ya que permite revisar y corregir esos errores al instante.

*BUILT-IN*
Cuando se selecciona el lenguaje de transformación integrado BUILT-IN, MapForce utiliza su motor de transformación nativo para ejecutar las asignaciones. MapForce también usa implícitamente esta opción cada vez que se genera la vista previa de los resultados de asignaciones que usan Java, C# o C++ como lenguaje de transformación.

El motor integrado 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 el lenguaje de transformación integrado 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 1149. Además, si selecciona el motor integrado como lenguaje de transformación, puede automatizar sus asignaciones en MapForce Server. Para obtener más información consulte Automatizar asignaciones con productos de Altova 830.

## 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*).

 © 2019-2025 Altova GmbH

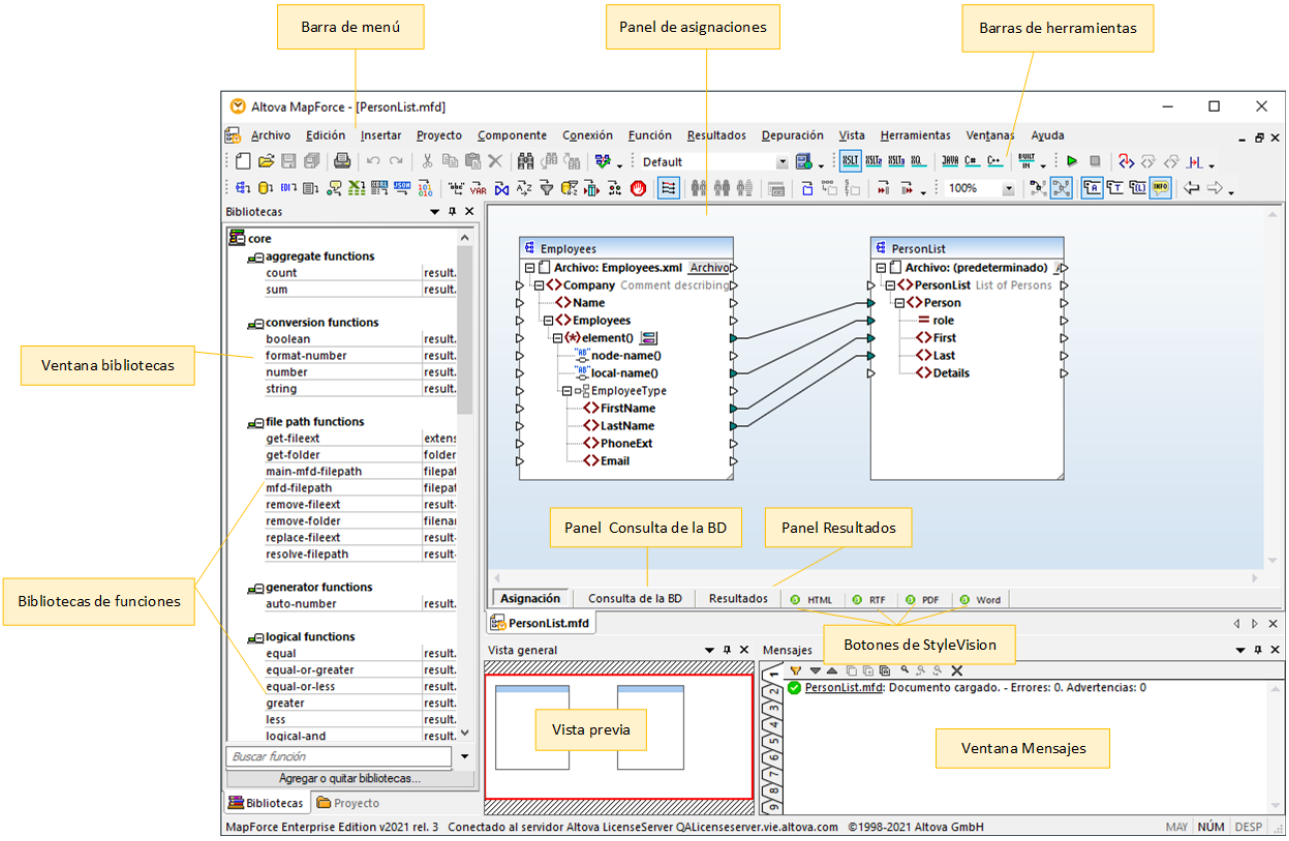
*XMLSpy*
Si tiene XMLSpy instalado en el mismo equipo que MapForce, podrá abrir y editar cualquier tipo de archivo compatible en XMLSpy desde MapForce directamente Por ejemplo, si hace clic con el botón derecho en un componente XML de MapForce, el menú contextual incluye el comando **Componente | Editar 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 con productos de Altova 830. Para obtener más información sobre MapForce como complemento consulte Complemento de MapForce para Visual Studio 886 y Complemento de MapForce para Eclipse 889.

# 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 sólo 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:

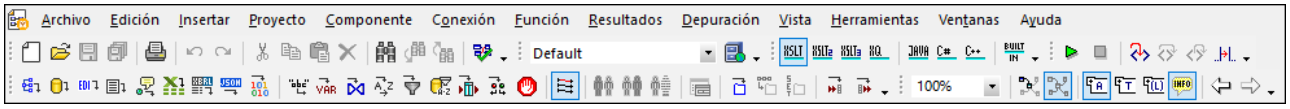

- Barras de herramientas 23
- Ventanas 23
- Ventana Mensajes 27
- Paneles 28

## 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

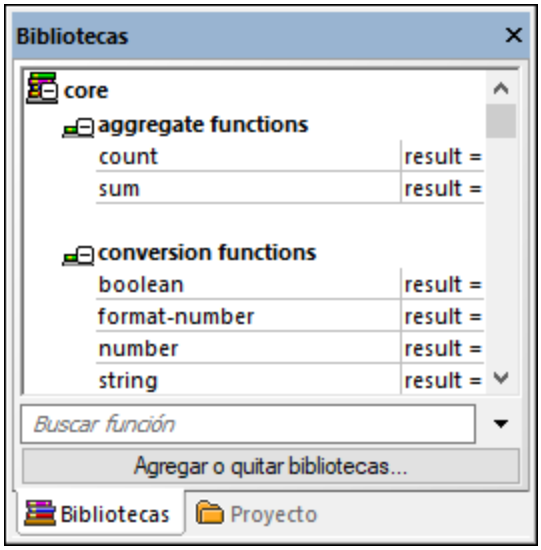
La barra de estado está situada en la parte inferior de la ventana de la interfaz y muestra información general sobre MapForce. Si pasa el puntero del ratón sobre los botones verá aparecer información rápida sobre ellos. Si usa la versión de 64 bits de MapForce, el nombre de la aplicación que aparece 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 19. Además, en esta ventana aparecen también las funciones definidas por el usuario que se hayan creado y las bibliotecas externas que se hayan importado.

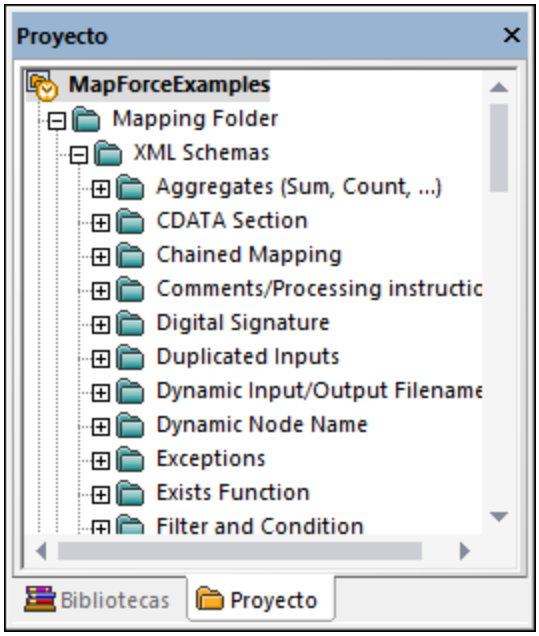



Para buscar funciones por nombre o por su descripción, escriba el valor de búsqueda en el cuadro de texto situado en la parte inferior de la ventana **Bibliotecas**. Para buscar todas las instancias de una función dentro de la asignación activa basta con hacer clic con el botón derecho en la función 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. Véase Funciones 461 para obtener más información.

## Ventana de proyecto

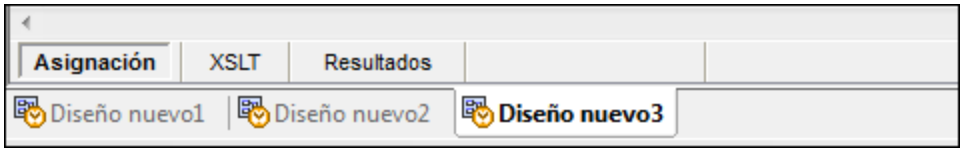
MapForce ofrece una interfaz compatible con múltiples documentos y permite agrupar las asignaciones en proyectos. La ventana **Proyecto** muestra todos los archivos y todas las carpetas del proyecto. La extensión de los archivos de proyecto de MapForce es **`*.mfp`**. Cuando quiera buscar una asignación dentro de un proyecto, haga clic dentro de la ventana **Proyecto** y pulse **Ctrl+F**. Para obtener más información consulte la sección Proyectos de asignación de datos 98.

 © 2019-2025 Altova GmbH




## 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).



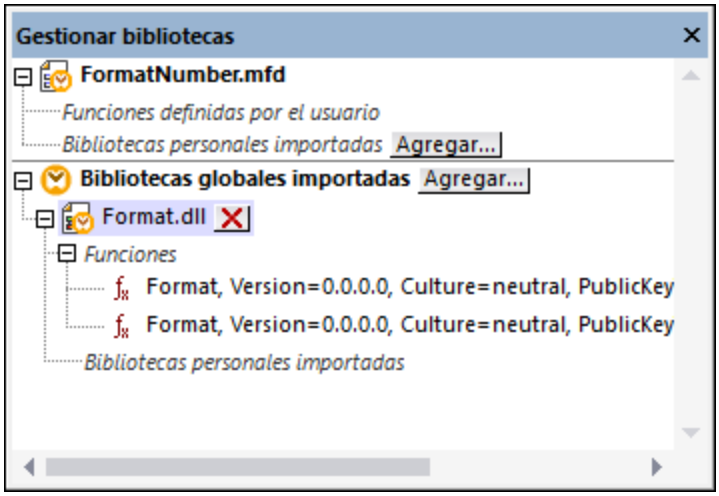

Puede acceder a las opciones de configuración de la ventana con el comando de menú **Ventanas | Ventanas**. El cuadro de diálogo "Ventanas" permite llevar a cabo distintas acciones, como activar, guardar, cerrar o minimizar las ventanas de asignación abiertas. Para seleccionar varias ventanas en el cuadro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



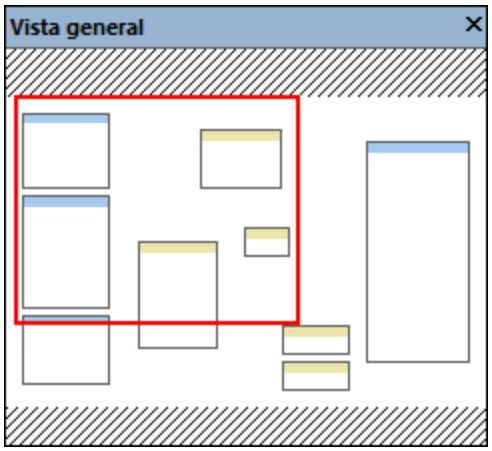

Puede elegir si quiere ver solamente las funciones definidas por el usuario del archivo de asignación activo o si quiere ver las de todos los archivos de asignación abiertos. Para ver las funciones y bibliotecas importadas para todas las asignaciones abiertas actualmente haga clic dentro de la ventana 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 464.

## Ventana Vista general

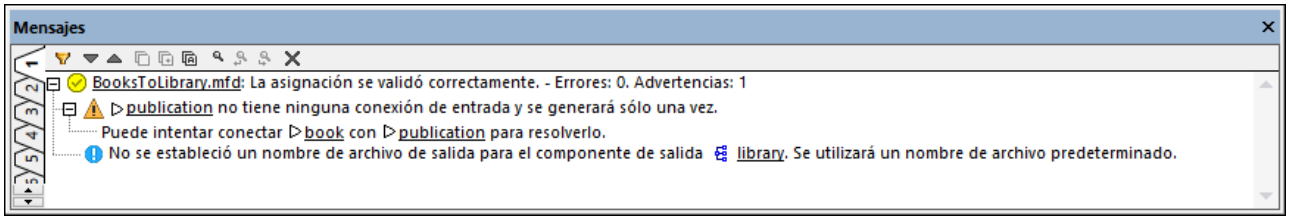
La ventana **Vista general** ofrece una perspectiva general del panel Asignación 28. Desde esta ventana podrá navegar más fácilmente por el área de asignación cuando la asignación sea muy grande. Para navegar por la asignación basta con arrastrar el rectángulo rojo por la ventana.




 © 2019-2025 Altova GmbH

## 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 66. 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**:

| Icono | Descripción |
|---|---|
| | Filtrar los mensajes según el nivel de gravedad: información, errores y advertencias. Seleccione **Activar todos** para incluir todos los niveles de gravedad (opción predeterminada).<br><br>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. |
| | Borrar 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

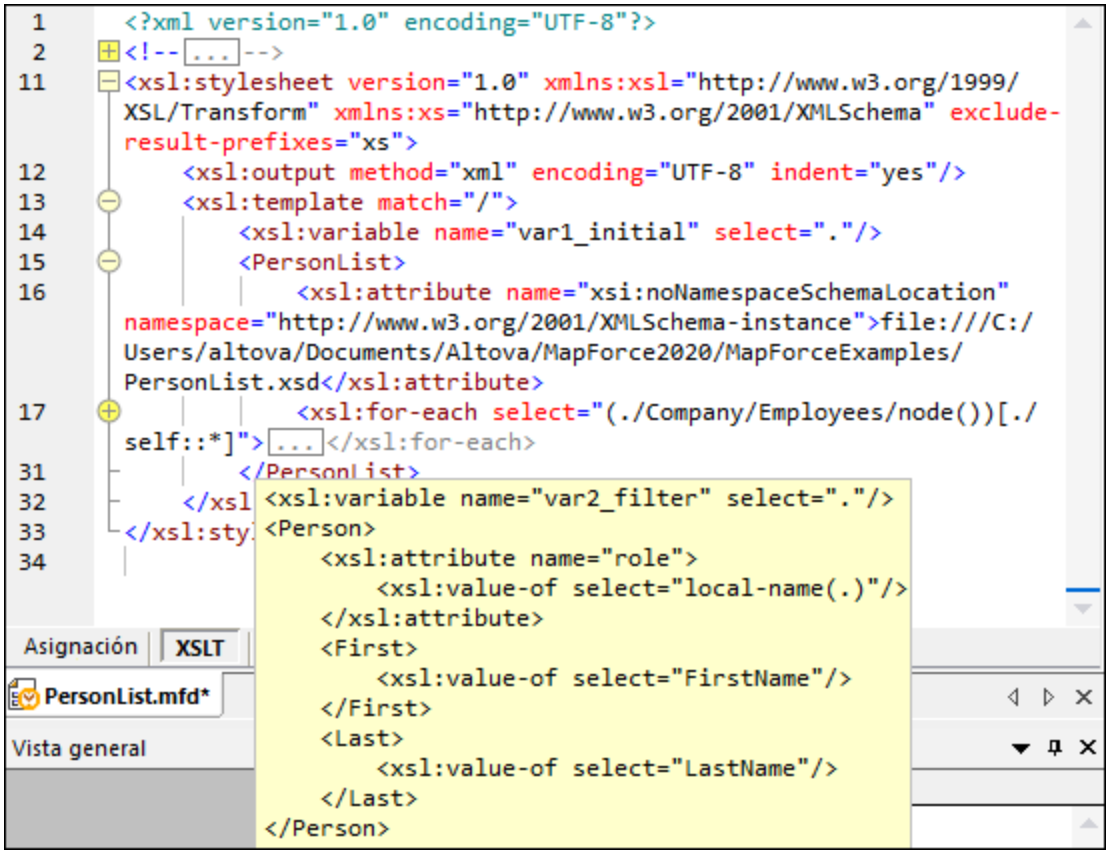
El panel **Asignación** es el área de trabajo donde se diseñan las asignaciones 66. En este panel puede añadir componentes de asignación (archivos, esquemas, constantes, variables, etc.) desde el menú **Insertar**. Para más información consulte Agregar componentes a la asignación 38. También puede arrastrar funciones desde la ventana **Bibliotecas** hasta el panel **Asignación**. Para más información consulte Funciones 462.

### 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 19 XSLT (XSLT, XSLT 2 o XSLT3) y después haga clic en la pestaña del mismo nombre.

 © 2019-2025 Altova GmbH

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.



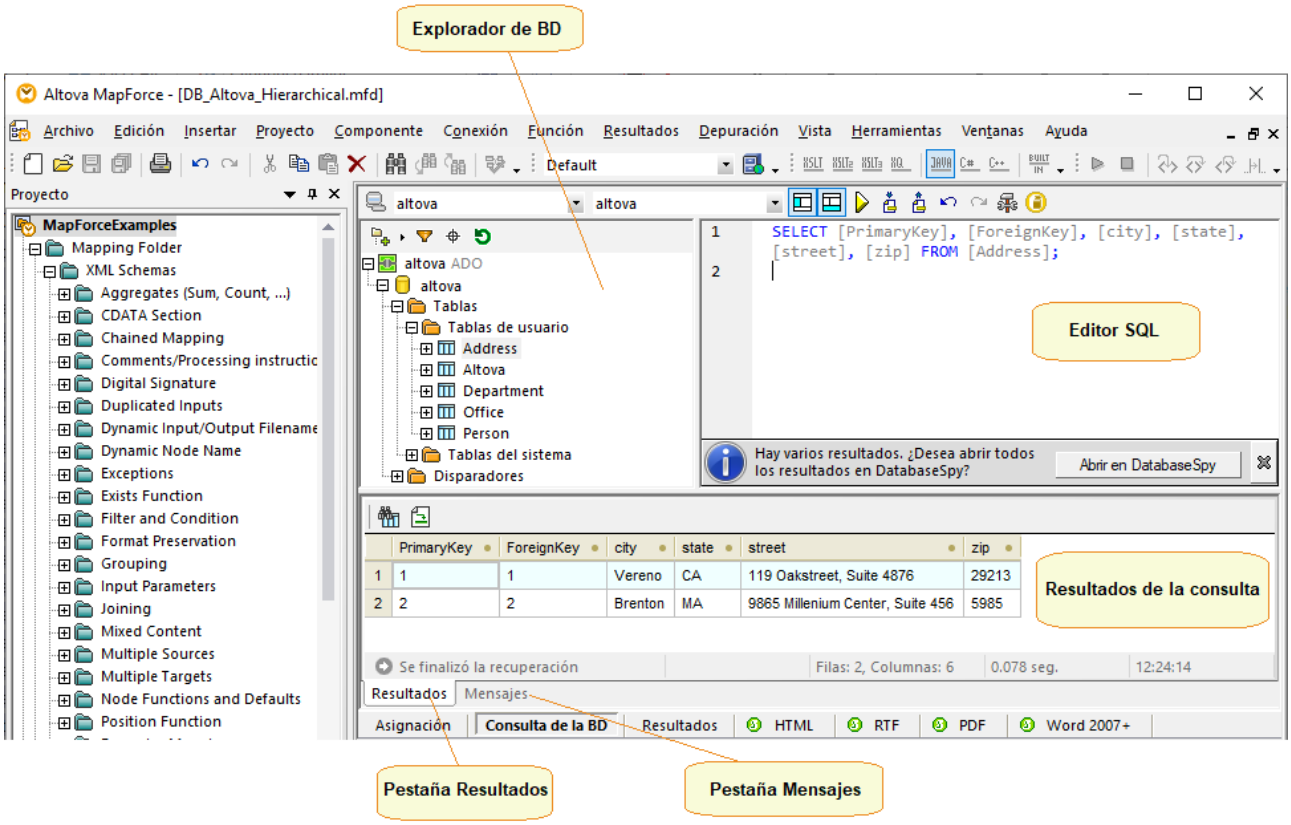

Para configurar las opciones de la vista, como la sangría, los marcadores de final de línea, etc. Haga clic con el botón derecho en el panel y seleccione **Configurar la vista Texto** en el menú contextual. También puede hacer clic en (**Configurar la vista Texto**) en la barra de herramientas.

## 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 294.




## 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).

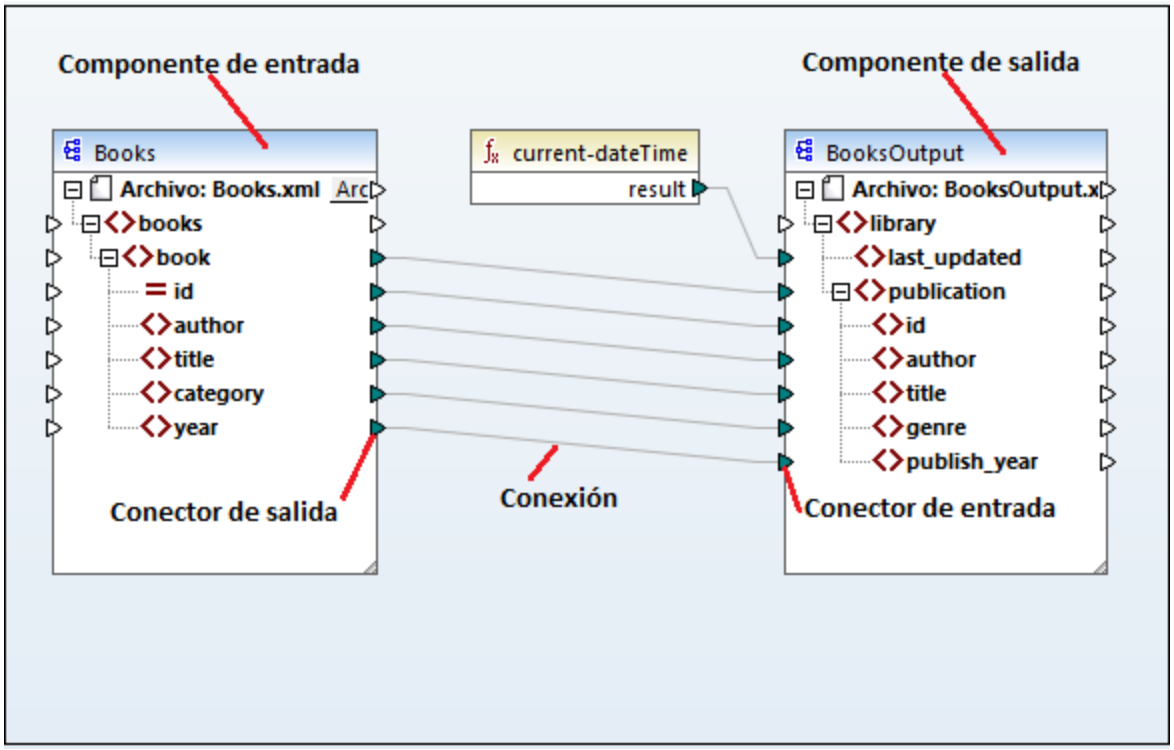
## Panel Resultados de StyleVision (ediciones Enterprise y Professional)

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 (hojas de estilo StyleVision Power) diseñados en StyleVision y asignados a un componente de asignación en MapForce.

# 2 Aspectos fundamentales de las 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**. Luego, seleccione **Asignación** y haga clic en **Aceptar**. El paso siguiente es agregar componentes 38 a la asignación y trazar las conexiones 48 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

 © 2019-2025 Altova GmbH

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 137 y de transformación 368

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 119. Solamente pueden ser componentes de paso a través los componentes estructurales.

En el segundo grupo (componentes estructurales y de transformación)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 34
.

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

*Conexión*
Una conexión es la línea que se puede dibujar para unir dos conectores. Al dibujar la línea y crear la conexión, estamos dando a MapForce instrucciones de transformar lo datos de una forma determinada, como por ejemplo leer datos de un documento XML y escribirlos en otro documento XML.

## Temas de esta sección

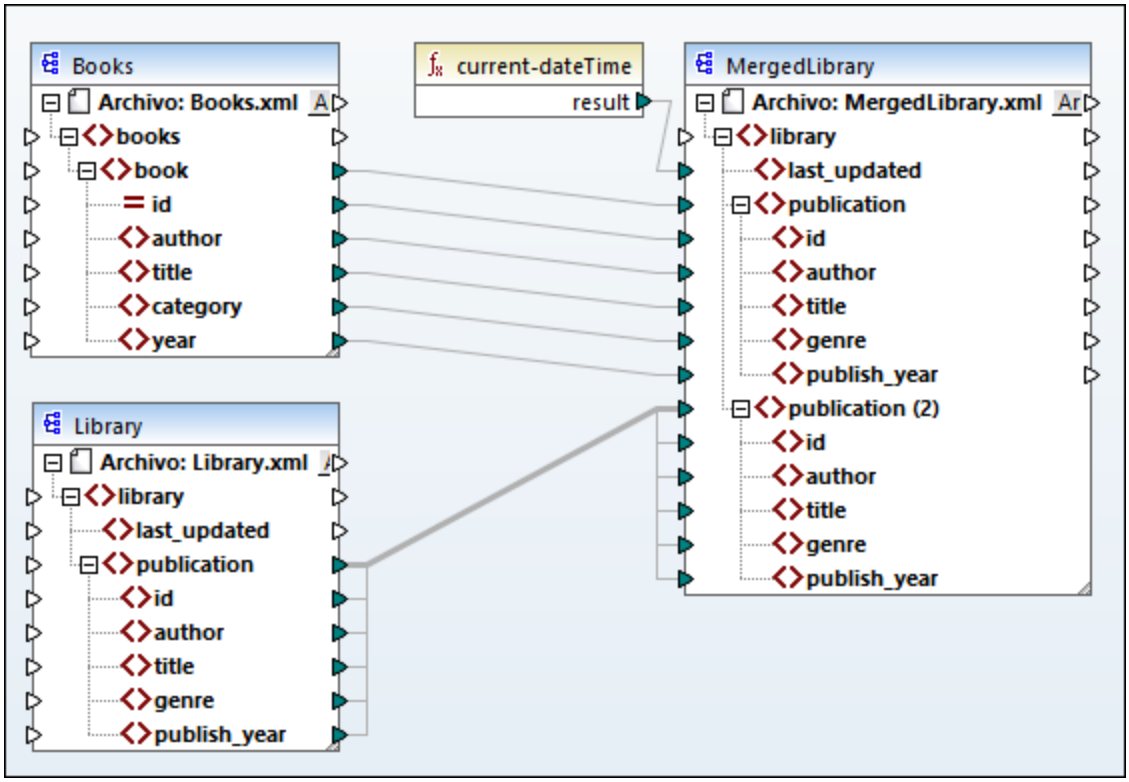
En esta sección explicamos las tareas y los conceptos más comunes de MapForce. Hemos dividido la sección en varios apartados:


- Componentes 34
- Conexiones 48
- Procedimientos y funciones generales 66
- Reglas y estrategias básicas 80
- Proyectos 98

# 2.1 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 32. Además de componentes estructurales y de transformación, también puede añadir comentarios a las asignaciones (*véase Comentarios más abajo*).

*Ejemplo*
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

Un componente estructural representa 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 137. Los componentes estructurales pueden leer datos de ciertos orígenes de datos, escribir datos en otros y 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.

 © 2019-2025 Altova GmbH

| 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 EDI (*edición Enterprise*) |
| | Componente XBRL (*edición Enterprise*) |
| | Protocol Buffers (*edición Enterprise*) |
| | PDF (*edición Enterprise*) |

## Componentes de transformación

Con los componentes de transformación puede transformar datos 461, almacenar resultados intermedios de una asignación para procesarlos más adelante, reemplazar un valor por otro 445, ordenar 426, agrupar 595, combinar 397 y filtrar 432 datos. También puede agregar excepciones 457, 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 |
| | Componente de ordenación |
| | Función integrada |
| | Función definida por el usuario |
| | Componente WHERE/ORDER de SQL/NoSQL (*ediciones Professional y Enterprise*) |
| | Componente de asignación de valores |
| | Variable |

| Icono | Descripción |
|---|---|
| | Función de servicio web (*edición Enterprise*) |
| | Excepción (*ediciones Professional y Enterprise*) |
| | Constante |
| | Condición If-Else |
| | Componente de combinación (*ediciones Professional y Enterprise*) |

## Comentarios

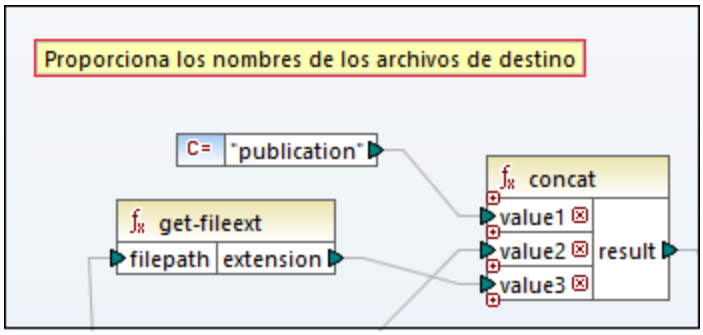
MapForce permite añadir comentarios como componentes independientes y también como notas bajo componentes que ya existan. El texto de los comentarios no solo está disponible en las asignaciones, sino que también se puede añadir a la documentación de la asignación 799. En la documentación de la asignación que se genera, los comentarios de los componentes se añaden directamente bajo el componente correspondiente y los comentarios de los componentes son parte de la sección *Componentes restantes*.

*Comentarios de componentes*

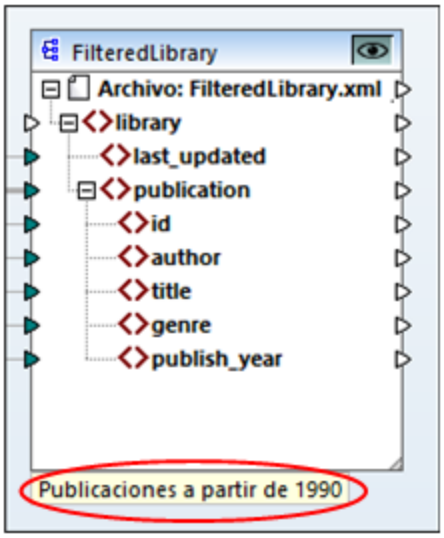
Los comentarios de los componentes son cajas autónomas en las que se visualizan varias líneas de texto y que no se pueden conectar a ningún otro componente. Para añadir un comentario a un componente tiene varias opciones:

- Seleccione el comando de la barra de herramientas, que abre un cuadro de diálogo donde puede escribir el comentario.
- Seleccione el comando de menú **Insertar | Comentario**, que abre un cuadro de diálogo donde puede escribir el comentario.
- Haga doble clic en un área vacía de la asignación, escriba el símbolo #, escriba un comentario y pulse la tecla **Entrar**. El símbolo # no aparece en la caja del comentario.

Para mover un comentario, arrástrelo a la ubicación deseada. Para eliminar un comentario, haga clic en su caja y pulse la tecla **Suprimir**. Más abajo puede ver un ejemplo de un comentario de componente (*rectángulo rojo*).



*Comentarios de componentes*
Además de los comentarios autónomos, también puede añadir comentarios a componentes que ya existan. Estos comentarios aparecen bajo el componente (*óvalo rojo*).




Para añadir un comentario a un componente que ya exista también tiene varias opciones:

- Haga clic con el botón derecho dentro del componente y seleccione **Editar comentario** en el menú contextual. Se abre un cuadro de diálogo en el que puede introducir un comentario.
- Seleccione el componente al que quiere añadir el comentario. Después seleccione **Editar comentario** en el menú **Componente**. Se abre un cuadro de diálogo en el que puede introducir un comentario.

Puede limitar los comentarios de componentes a un número concreto de líneas, que puede definir en el menú **Herramientas | Opciones | General | Visualización de asignaciones**. Para más información consulte Opciones 1128.

Para eliminar un comentario de un componente:

- Haga doble clic en el comentario, borre todo el texto y pulse la tecla **Entrar**.
- Haga clic con el botón derecho en el comentario o dentro del componente, seleccione **Editar comentario** del menú contextual, borre el texto y haga clic en **Aceptar**.

*Editar comentarios*
Puede editar los dos tipos de comentarios de varias formas:

- Haga doble clic en el texto del componente y edite el texto directamente en la caja. Pulse la tecla **Entrar**.
- Haga clic con el botón derecho en la caja del comentario, edite el texto en el cuadro de diálogo **Editar comentario** y haga clic en **Aceptar**. También puede acceder al cuadro de diálogo **Editar comentarios** haciendo clic con el botón derecho dentro del componente y seleccionando la opción **Editar comentario** en el menú contextual.

## En esta sección

Esta sección sobre componentes se divide en distintos apartados:

- Agregar componentes 38
- Componentes básicos 41
- Rutas de acceso de archivos 44


## 2.1.1 Agregar componentes

En este apartado explicamos cómo agregar componentes a una asignación. Para agregar un componente primero debe crear un diseño de asignación nuevo 32 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.
- 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 34. 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 137. Para más información sobre los componentes de MapForce que se usan para almacenar datos temporalmente o para transformarlos consulte Diseño de asignaciones 368.

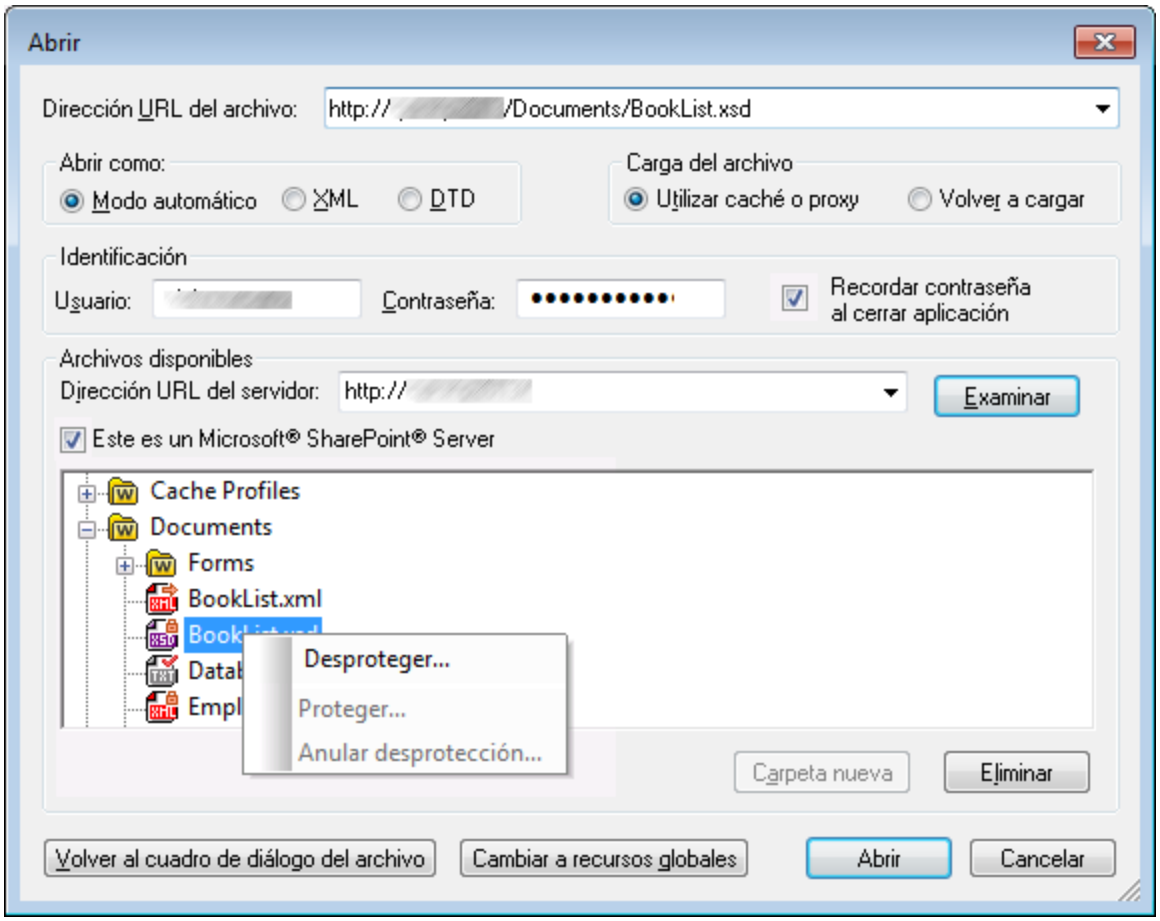
Si quiere agregar un componente estructural a una asignación, puede elegir entre agregar un archivo local, un componente desde una URL o uno de la lista de recursos globales (*véanse las subsecciones siguientes*).

### Agregar componentes desde una URL

Sólo se pueden agregar componentes desde una URL a componente de origen 32. Los protocolos compatibles son HTTP, HTTPS y FTP. En función del tipo de estructura de datos, las instrucciones sobre cómo añadir un componente desde una URL pueden variar. Para la mayoría de las estructuras de datos sirven estas instrucciones:

1. Seleccione componente que quiere agregar (p.ej. **Archivo o esquema XML**).
2. Haga clic en **Cambiar a URL** en el cuadro de diálogo **Abrir**.
3. Introduzca la URL en la caja de texto *Dirección URL del archivo* y haga clic en **Abrir** (*imagen siguiente*).

 © 2019-2025 Altova GmbH




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

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

- *Cargar archivos:* 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. Seleccione *Volver a cargar* para que el archivo se cargue de nuevo cada vez que abra la asignación.

- *Identificación:* 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*.

- *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 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**.

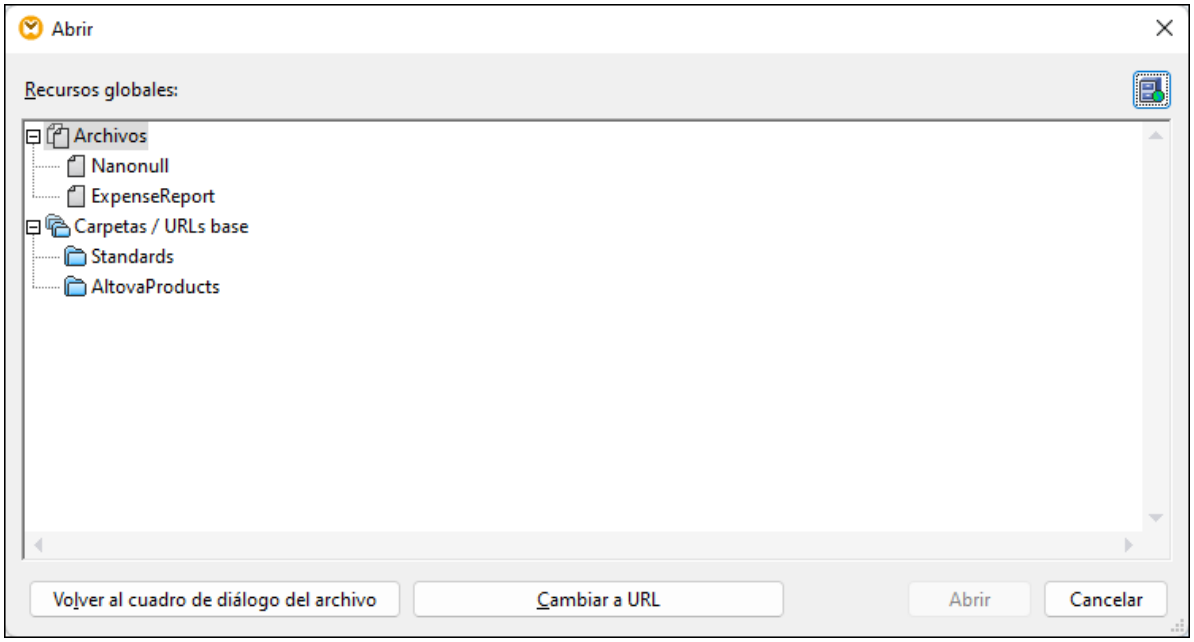
- *Cambiar al cuadro de diálogo Archivo:* Haga clic en este botón para ir al cuadro de diálogo en el que puede seleccionar un archivo local.

- *Cambiar a recursos globales:* Haga clic en este botón para ir al cuadro de diálogo en el que puede seleccionar un recurso global.

## Agregar recursos globales

Puede añadir a la asignación cualquier base de datos (*ediciones Professional y Enterprise*), archivo o carpeta que haya definido como recurso global. Para más información, consulte Recursos globales de Altova 856. En función del tipo de estructura de datos con el que esté trabajando, las instrucciones sobre cómo añadir un componente desde una URL pueden variar. Para la mayoría de las estructuras de datos sirven estas instrucciones:

1. Seleccione el componente que quiere agregar (p.ej. **Archivo o esquema XML**).
2. Haga clic en **Cambiar a URL** en el cuadro de diálogo **Abrir**.
3. Seleccione uno de los recursos de la lista y haga clic en **Abrir** (*imagen siguiente*).




Si quiere agregar, editar y eliminar recursos globales haga clic en el icono **Administrar recursos globales** (*en un círculo rojo*). El cuadro de diálogo **Abrir** para recursos globales permite volver a cambiar al cuadro de diálogo de archivos locales (**Cambiar al cuadro de diálogo Archivo**) o abrir un archivo desde una URL (**Cambiar a URL**).

Si creó una BD como recurso global y la quiere añadir a una asignación, siga los pasos que se explican en Recursos globales 206.

 © 2019-2025 Altova GmbH

## 2.1.2 Componentes básicos

En este apartado explicamos cómo configurar, buscar y manipular componentes estructurales [34]. 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**.

Tenga en cuenta que las opciones disponibles dependen del tipo de componente. Estos son los tipos de componentes:

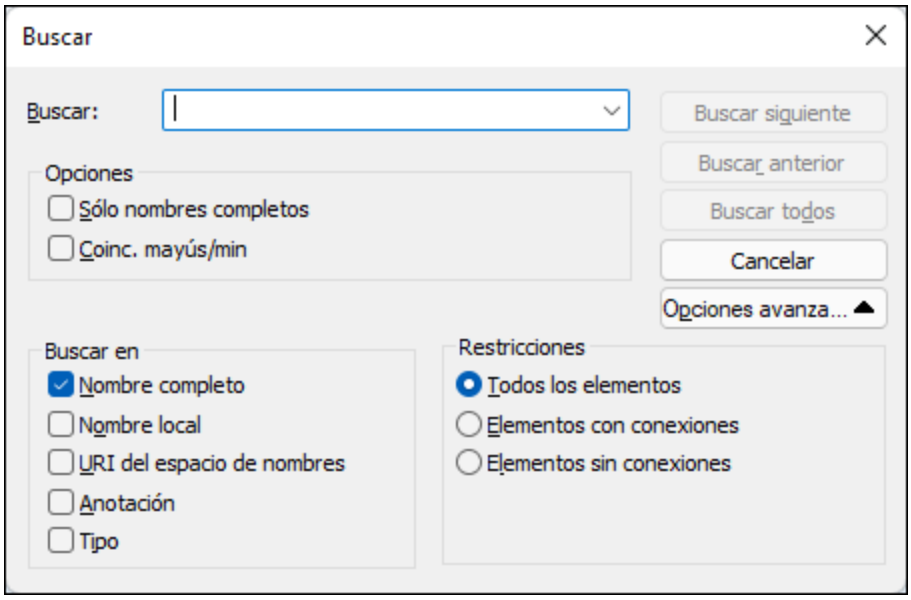
- Configuración de componentes XML [139]
- Configuración de componentes de base de datos [251]
- Configuración de componentes CSV [353]
- Configuración de componentes FLF (de longitud fija) [361]

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 [784]. Este botón también permite configurar las opciones para analizar cadenas de texto y serializar datos en cadenas de texto [791].

### Buscar dentro de los componentes

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

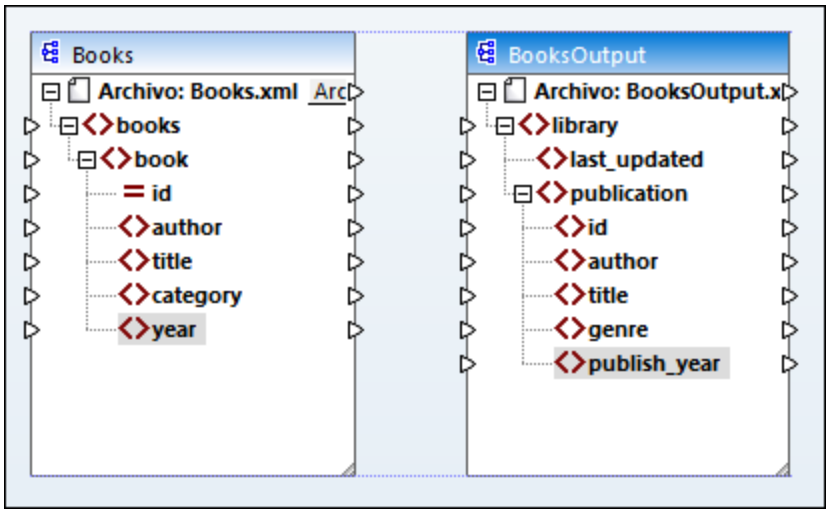
1. Haga clic en el componente donde quiere buscar y pulse **Ctrl+F**.
2. Introduzca un término de búsqueda y haga clic en Buscar siguiente/anterior/todos (*imagen siguiente*).




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**.

 © 2019-2025 Altova GmbH

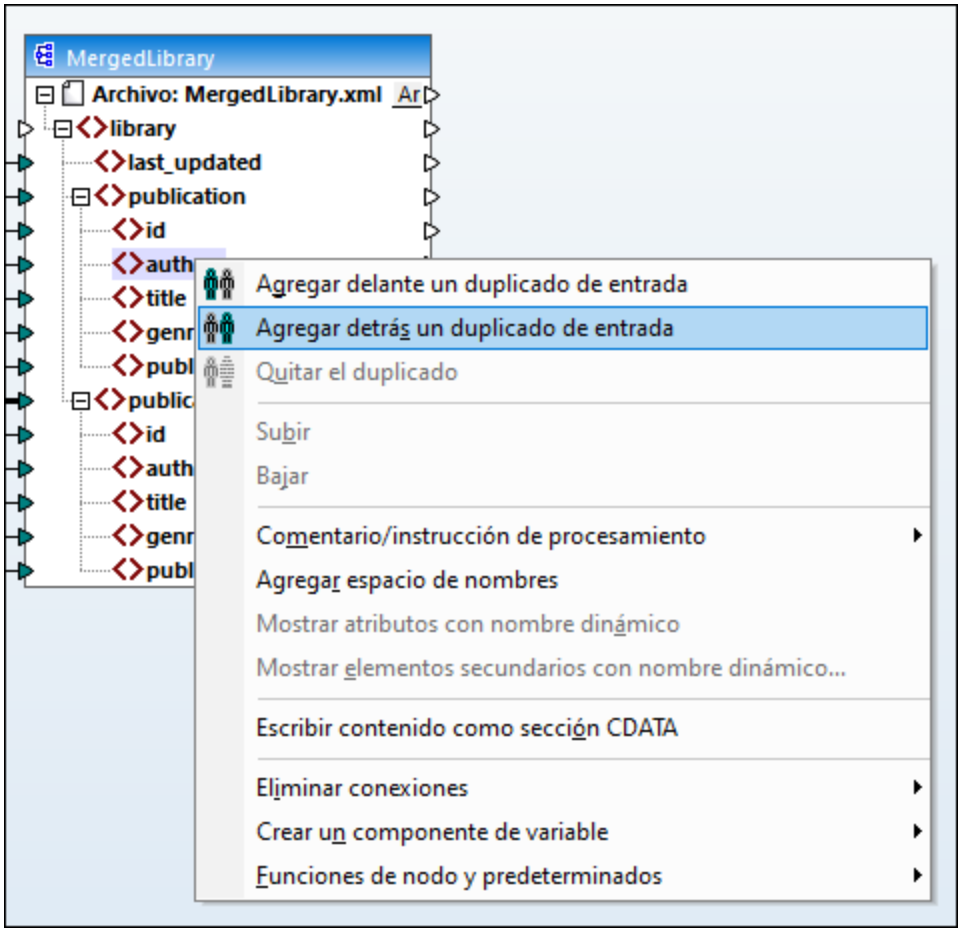
## Duplicar elementos de entrada

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 114.

*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.

## 2.1.3 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 44
- Rutas de acceso según el entorno de ejecución 46


### 2.1.3.1 Rutas de acceso absolutas y relativas

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 datos (p. ej. XML, JSON, CSV, etc.)
- Archivos de esquema (relevantes para los componentes con esquemas)
- Archivos utilizados por parámetros complejos de entrada o salida de funciones definidas por el usuario y de variables de tipo complejo
- Archivos StyleVision Power Stylesheet (**`*.sps`**), que se utilizan para dar formato a los datos para archivos de salida como PDF, HTML y Word (*ediciones Professional y Enterprise*)
- Archivos de base de datos (*ediciones Professional y Enterprise*)
- Archivos de esquema a los que hacen referencia componentes de base de datos que admiten campos XML (*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*
Si añade o modifica una referencia de archivo en una asignación y no se puede resolver la ruta de acceso, MapForce emite un mensaje de advertencia. No obstante, pueden darse referencias de ruta de acceso rotas en estos casos:

 © 2019-2025 Altova GmbH

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

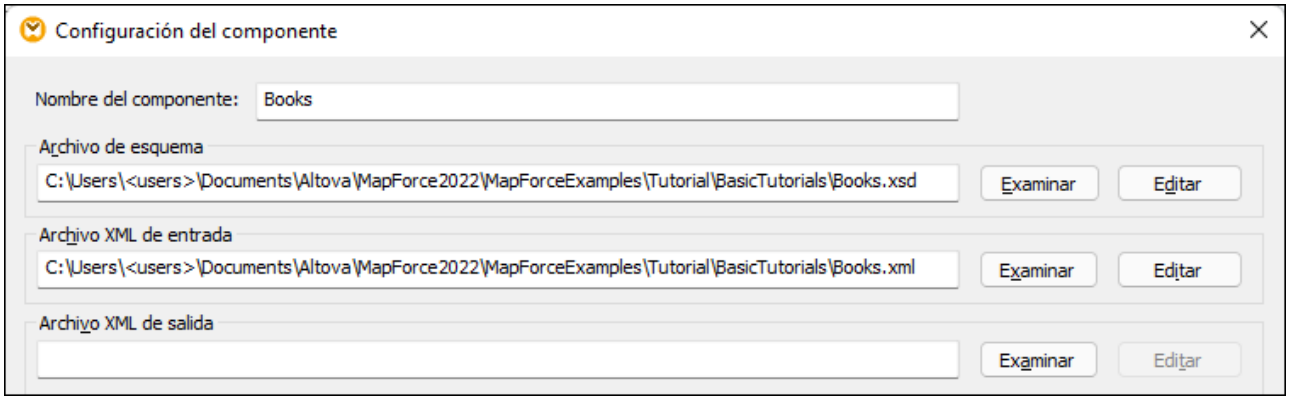
Las referencias rotas a rutas de acceso provocan que MapForce resalte el componente afectado en color rojo. En casos así, la solución consiste en hacer doble clic en el título del componente y actualizar las referencias de ruta de acceso rotas en el cuadro de diálogo **Configuración del componente**. Consulte también Cambiar configuración de los componentes 41.

## Ejemplo 1: 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 32 y agregue un componente estructural 38: 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.

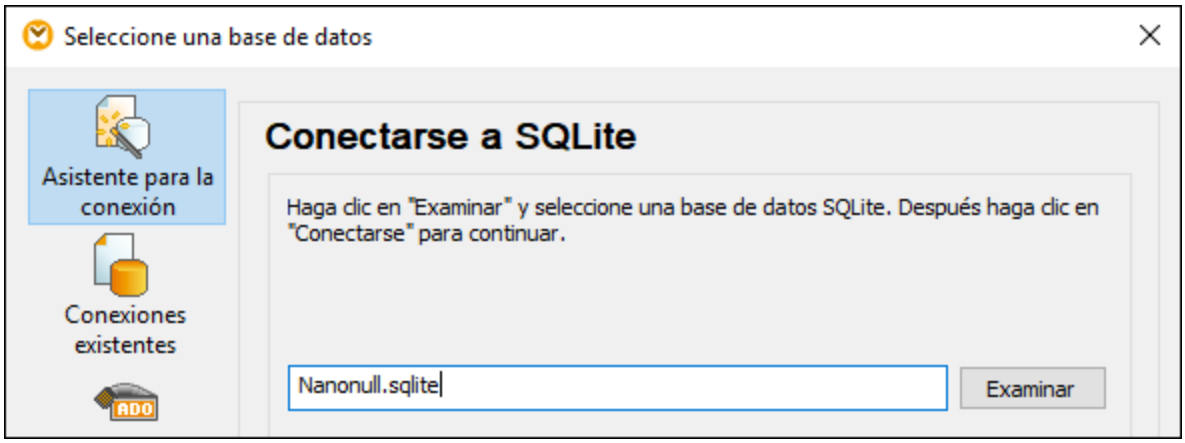
**Nota:** Las rutas que hacen referencia a unidades no locales o usan una URL no se pueden convertir en relativas.




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.

### Ejemplo 2: componente de BD (ediciones Professional y Enterprise)

Si agrega a la asignación un archivo de BD 179, 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 78. Para más información consulte Rutas de acceso según el entorno de ejecución 46.

## 2.1.3.2 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:

| 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**). |

 © 2019-2025 Altova GmbH

| Lenguaje de destino | Ruta base |
|---|---|
| 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 78.

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 464.

** 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 78.
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.

# 2.2 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.

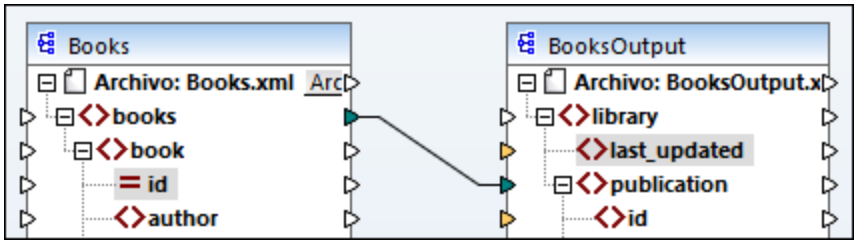
## Crear, copiar y eliminar conexiones

Para crear una conexión entre dos elementos, pulse y mantenga pulsado el conector de salida 33 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 43. 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.

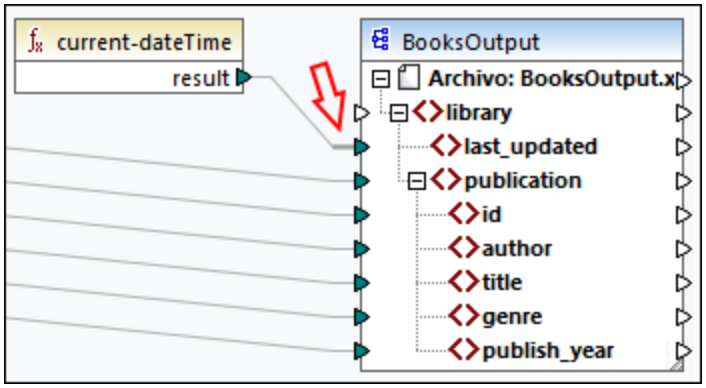
Si prefiere deshabilitar este tipo de notificaciones:

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**.

 © 2019-2025 Altova GmbH

## Mover conexiones

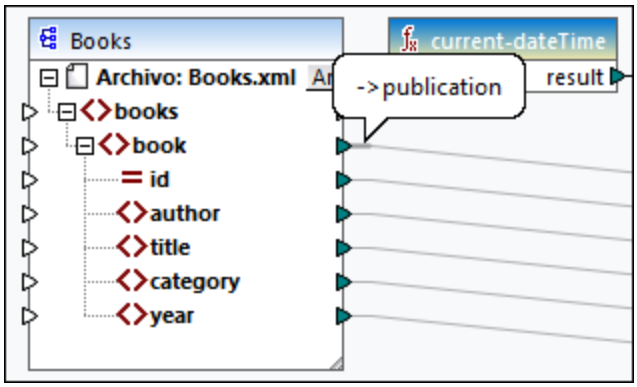
Para mover una conexión a otro nodo pulse y mantenga pulsada la parte gruesa que hay al final de la conexión (*véase la imagen de más abajo* ) y arrástrela hasta su destino.




## 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:

- 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 la conexión 59.

## 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*

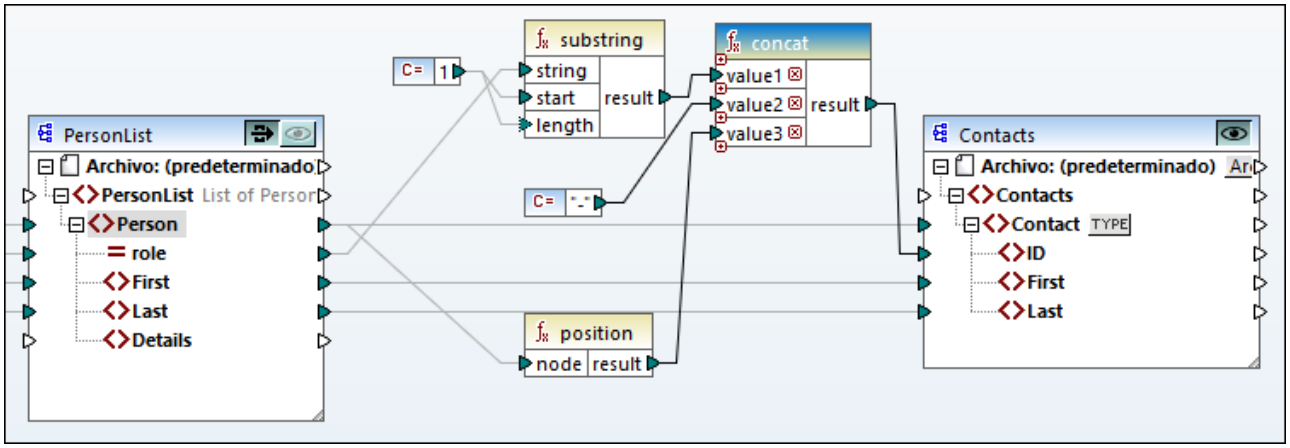
El botón para **conexiones de origen a destino** se habilita solo si pulsa el botón para **conexiones directas**. Si pulsa el botón para **conexiones de origen a destino** puede hacer un seguimiento de las conexiones del componente seleccionado en ese momento, incluidas las conexiones directas y las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

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

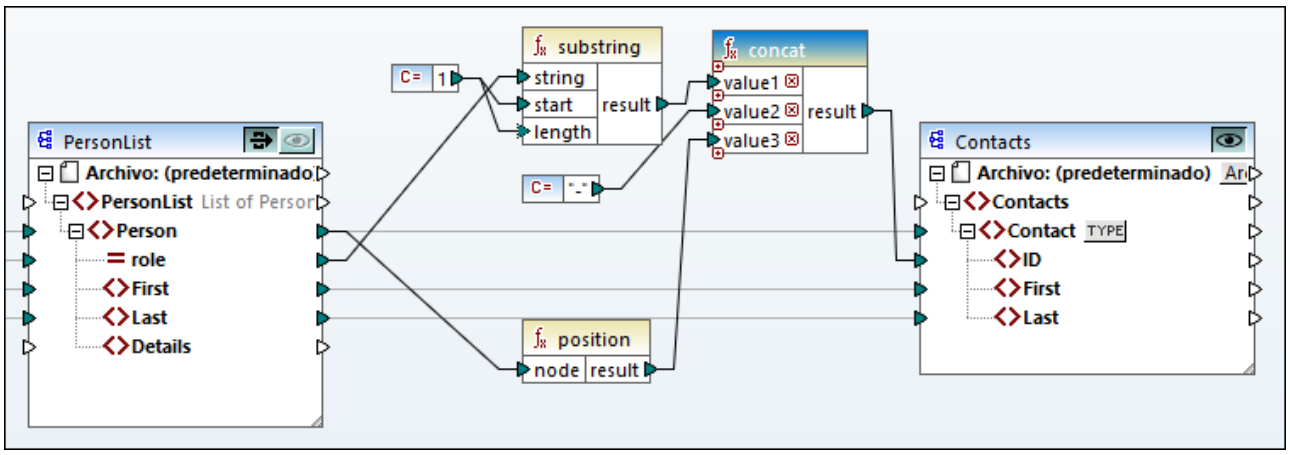
*Ejemplo*

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.

 © 2019-2025 Altova GmbH




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 [35] (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 [34] (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 [52]
- Configuración de las conexiones [59]

- Menú contextual de las conexiones 60
- Conexiones defectuosas 62
- Conservar conexiones tras eliminación de componentes 63

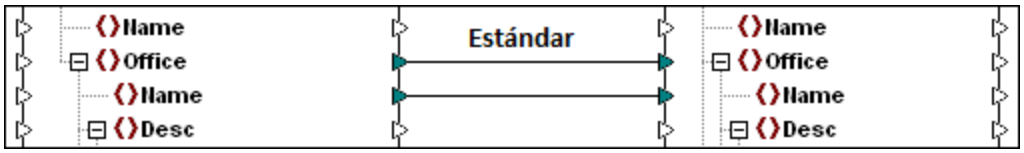

## 2.2.1 Tipos de conexión

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

- Conexiones basadas en el destino 52 (estándar)
- Conexiones basadas en el origen 52 (contenido mixto)
- Conectar los secundarios equivalentes 55
- Conexiones de copia total 57 (copiar los elementos secundarios)

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

Las conexiones basadas en el destino y conexiones basadas en el origen 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 destino 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 el origen 52: 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.

## 2.2.1.1 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.

 © 2019-2025 Altova GmbH

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

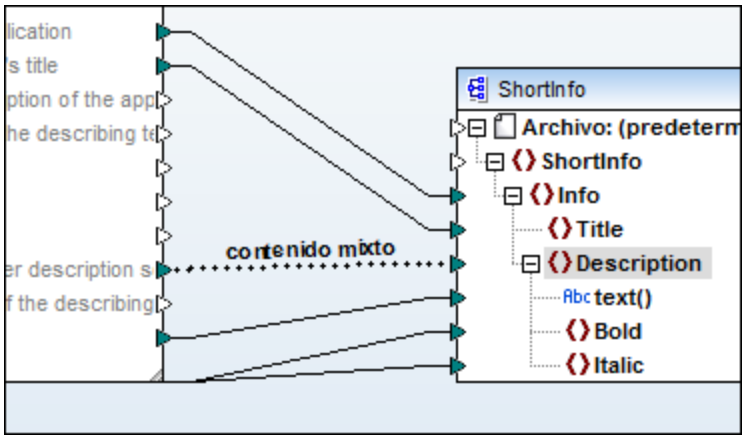
**Nota:** Para que acepten contenido mixto los componente de destino deben tener nodos de contenido mixto.

## 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`**, **`Tut-Person.xsd`** y **`Tut-OrgChart.xsd`**, que encontrará en la carpeta Tutoriales 18.

*Instancia XML de origen*

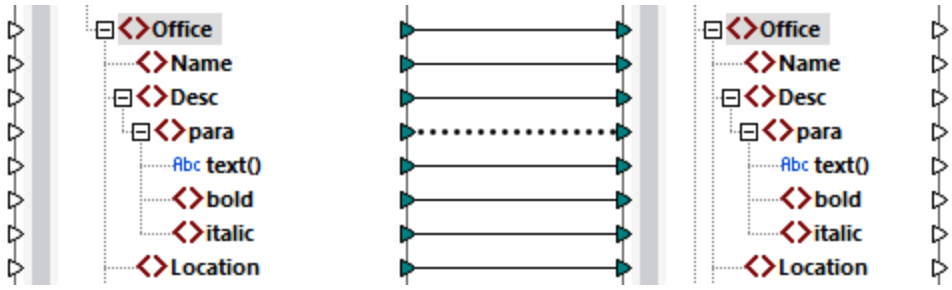
A continuación puede ver un extracto de **`Tut-OrgChart.xml`**. En este ejemplo nos centraremos en el elemento de contenido mixto `<para>` y en sus nodos secundarios `<bold>` e `<italic>`. El elemento **`<para>`** también contiene una instrucción de procesamiento (`<?sort alpha-ascending?>`) y un comentario (`<!--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 55 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.




*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`** de la carpeta **`Tutoriales`**.
2. Pulse el botón de la barra de herramientas (Conectar automáticamente los secundarios equivalentes 55). Desmarque la casilla **Crear conexiones de copia total** en la configuración de las conexiones de secundarios equivalentes 55. Esto evita que MapForce cree conexiones de copia total 57 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.

 © 2019-2025 Altova GmbH

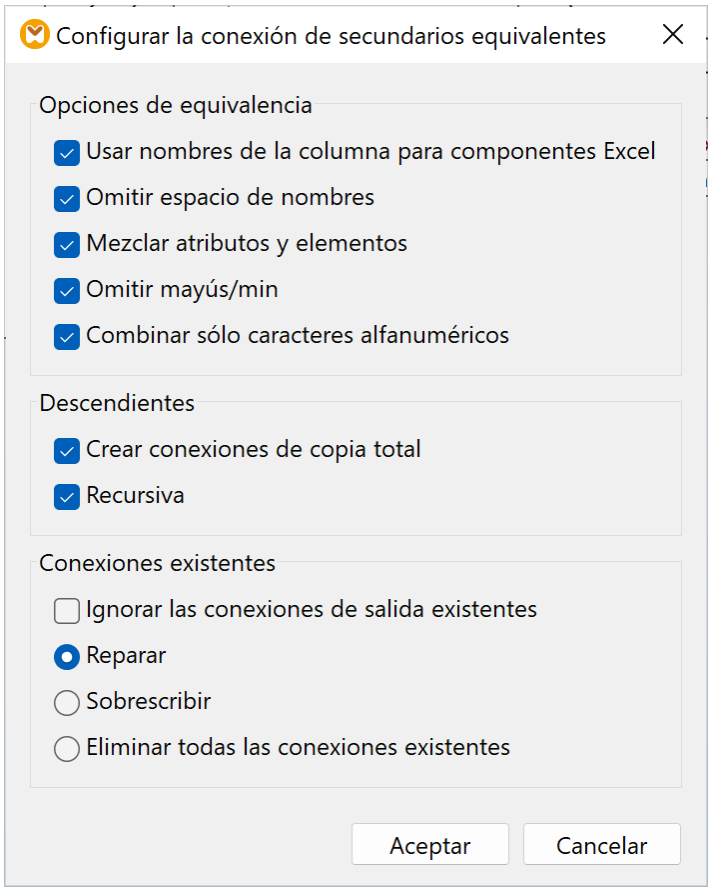
## 2.2.1.2 Conexiones de secundarios equivalentes

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. También puede ir al menú **Conexión** y hacer clic en **Configurar la conexión de secundarios equivalentes**. Así, se abrirá el cuadro de diálogo **Configurar la conexión de secundarios equivalentes** (*ver imagen siguiente*).



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 pulsado el botón de la barra de herramientas (**Alternar la conexión automática de secundarios**).

*Opciones de equivalencia*
La sección *Opciones de equivalencia* permite relajar criterios de coincidencia y definir cómo compara nombres de nodos. Estas son las opciones disponibles:

- *Usar nombres de la columna para componentes Excel:* esta opción sólo se aplica a componentes de Excel (*Enterprise Edition*). Esta opción significa que para la comparación se utilizarán los nombres de columna definidos por el usuario (p.ej., `Company`) en lugar de los nombres de referencia de columna (p.ej., A, B, C). Los nombres de columna definidos por el usuario se establecen en el cuadro de diálogo **Seleccionar rango de celdas** y aparecen como anotaciones en un componente de Excel.
- *Omitir espacio de nombres:* los secundarios equivalentes se conectan independientemente de los espacios de nombres de los nodos secundarios.
- *Mezclar atributos y elementos:* esta opción permite crear conexiones entre atributos y elementos que tienen el mismo nombre. Por ejemplo, si existen dos nodos `ID` se crea una conexión aunque uno sea un elemento y el otro un atributo.
- *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.
- *Combinar sólo caracteres alfanuméricos:* si esta opción está activa, se compararán sólo dígitos y letras. Otros caracteres, como espacios, comas, puntos, etc. se descartarán antes de la comparación.

*Descendientes*
La sección *Descendientes* describe cómo proceder con nodos descendientes/secundarios. Estas son las opciones disponibles:

- *Crear conexiones de copia total:* esta casilla está activada por defecto. Es una opción que crea (si es posible) conexiones de copia total 57 que crean asignaciones de datos entre nodos que tienen nodos secundarios que son muy parecidos o idénticos. Una conexión de copia total, representada por una línea gruesa, evita el desorden y facilita la comprensión de la asignación.
- *Recursiva:* 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.

*Conexiones existentes*
La sección *Conexiones existentes* especifica qué se puede hacer con conexiones existentes. Estas son las opciones disponibles:

- *Omitir las conexiones de salida existentes:* esta opción crea más conexiones para todos los secundarios equivalentes, aunque ya tengan conexiones de salida.
- *Reparar:* 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.

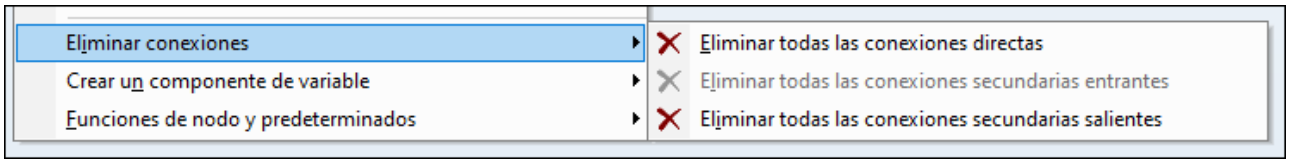
## Eliminar conexiones en grupo

Si quiere eliminar conexiones en grupo siga estos pasos:

1. Haga clic con el botón derecho en un nombre de nodo dentro del componente.

 © 2019-2025 Altova GmbH

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. Elimina todas las conexiones secundarias salientes del nodo principal seleccionado.

## 2.2.1.3 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. Sólo 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.

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:

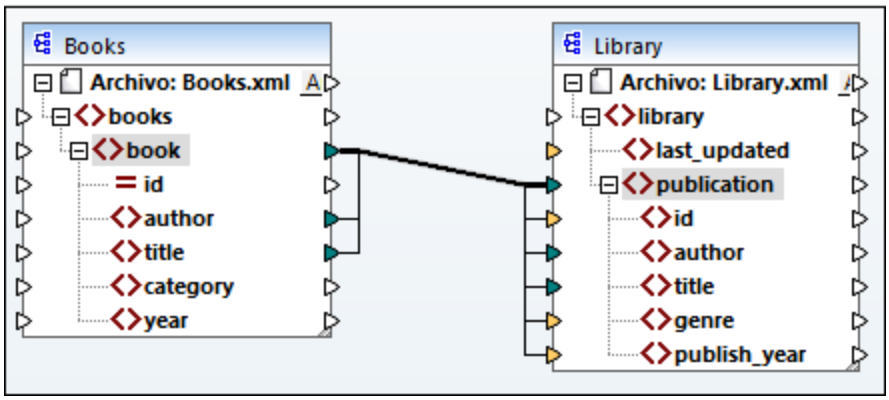
1. Vaya al menú **Conexión**.
2. Haga clic en **Configurar la conexión de secundarios equivalentes**.
3. Marque la casilla **Crear conexiones de copia total** y haga clic en **Aceptar**.
4. Pulse el botón **Alternar la conexión automática de secundarios** de la barra de herramientas. También puede ir 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 la carpeta de Tutoriales básicos [18].
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 [110].
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 sólo 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 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 [43] sólo 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 [52] . Si prefiere eliminar la conexión de copia total por completo haga clic en **Eliminar conexiones secundarias**.

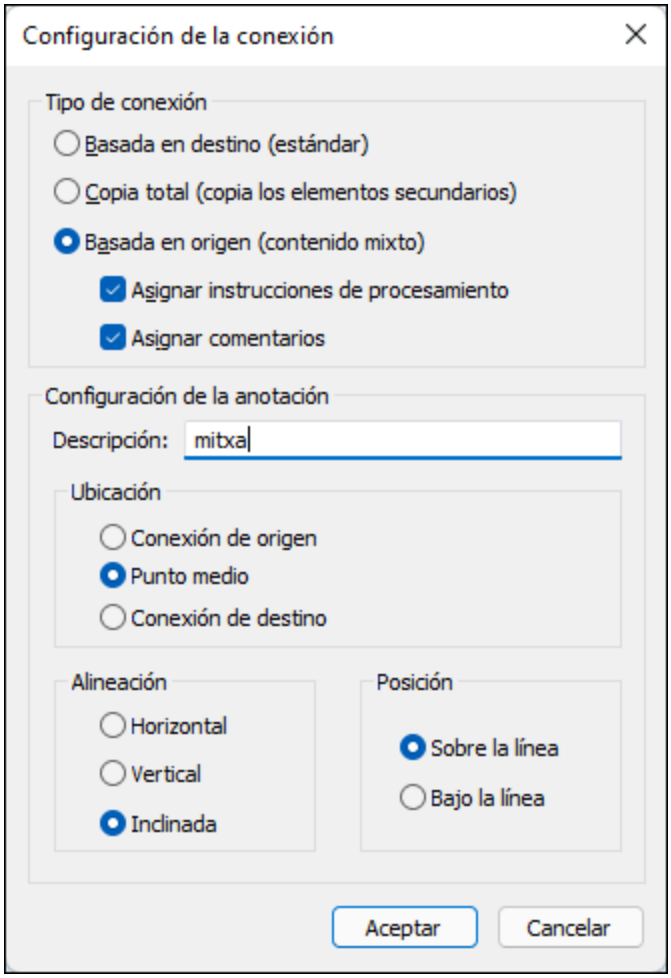
**Importante**

Cuando se crea una conexión de copia total entre un esquema y un parámetro de una función definida por el usuario [485], los dos componentes deben basarse en el mismo esquema. Sin embargo, no es necesario que los dos tengan el mismo elemento raíz.

 © 2019-2025 Altova GmbH

## 2.2.2 Configuración de la conexión

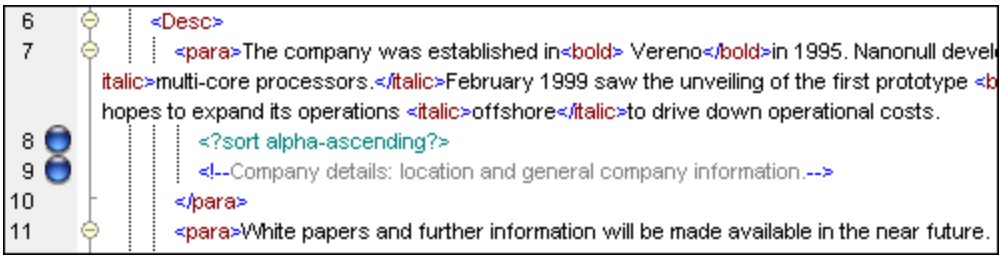
Este cuadro de diálogo sirve para configurar conexiones. Para abrirlo haga doble clic en una conexión. También puede hacer clic con el botón derecho en la conexión y seleccionar **Propiedades** en el menú contextual. Los ajustes se dividen en dos partes: tipos de conexión y configuración de la anotación.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




Tipos de conexión

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

- Basada en destino (estándar) 52: estas conexiones se pueden usar en casi todas las asignaciones.
- Copia total (copia lo elementos secundarios) 57: 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) 52: 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*).




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:

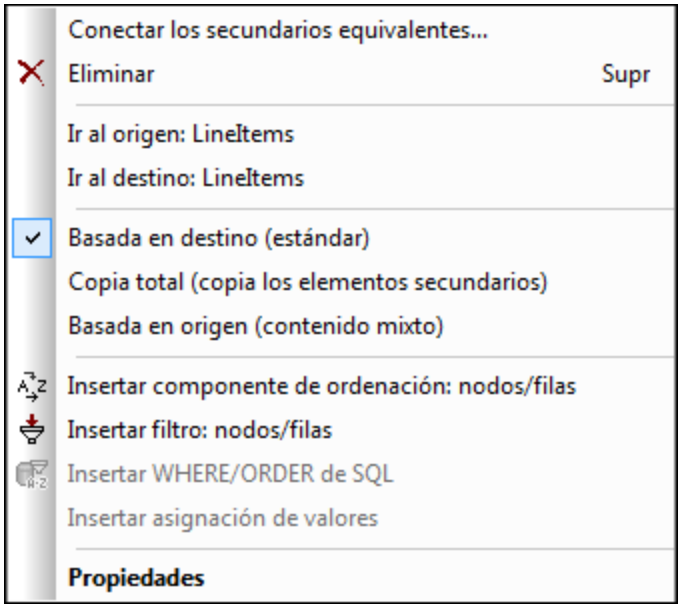
1. Haga clic con el botón derecho en la conexión y seleccione **Propiedades** del menú contextual. O, si lo prefiere, haga doble clic en la conexión.
2. Introduzca el nombre de la conexión seleccionada en el campo **Descripción**. Esto habilita todas las conexiones de la sección **Configuración de la 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.

## 2.2.3 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:

 © 2019-2025 Altova GmbH




Para más información siga leyendo.
*Configuración general*

- *Conectar los secundarios equivalentes:* Abre el cuadro de diálogo Conectar los secundarios equivalentes 55. 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 33 de la conexión seleccionada.
- *Ir al destino: <nombre del elemento>:* Resalta el conector de entrada 33 de la conexión seleccionada.

*Tipos de conexión*
Consulte los detalles sobre los tipos de conexión en Tipos de conexión 52 y Configuración de las conexiones 59.

*Comandos de inserción*

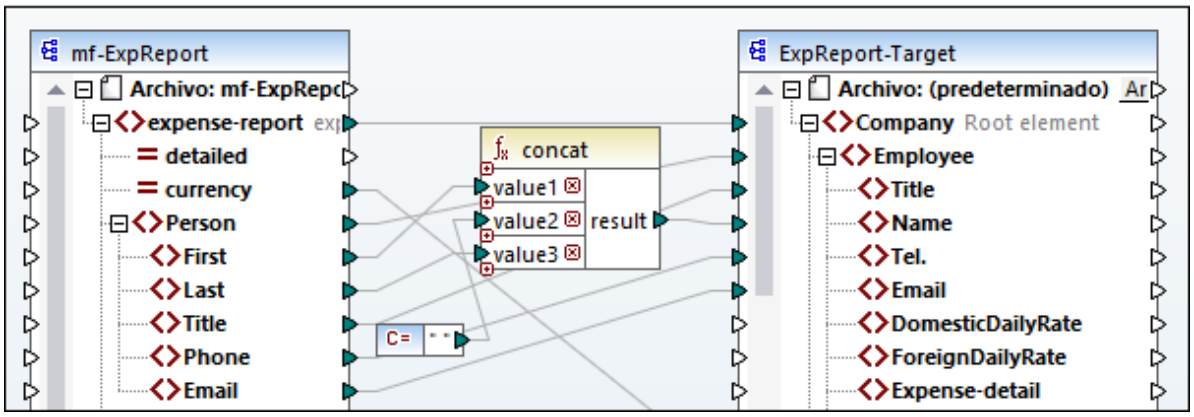
- *Insertar componente de ordenación: nodos/filas:* Agrega un componente de ordenación 426 entre un nodo de origen y uno de destino.
- *Insertar filtro: nodos/filas:* Agrega un componente de filtrado 432 entre un nodo de origen y uno de destino.
- *Insertar componente WHERE/ORDER de SQL/NoSQL:* Agrega 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) 437.
- *Insertar asignación de valores:* Agrega una asignación de valores 445 entre un nodo de origen y uno de destino.

*Propiedades*
Abre el cuadro de diálogo Configuración de las conexiones 59.

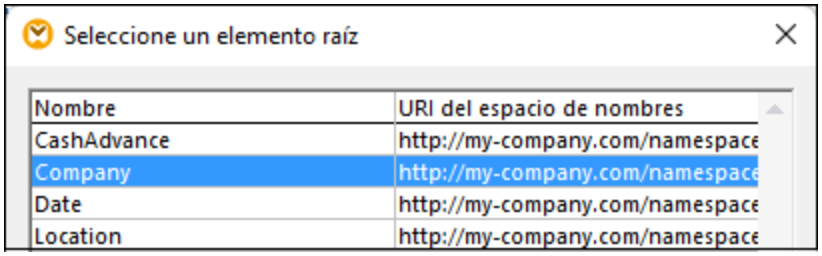
## 2.2.4 Conexiones defectuosas

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 Tutoriales 18. A continuación puede ver parte de esta.

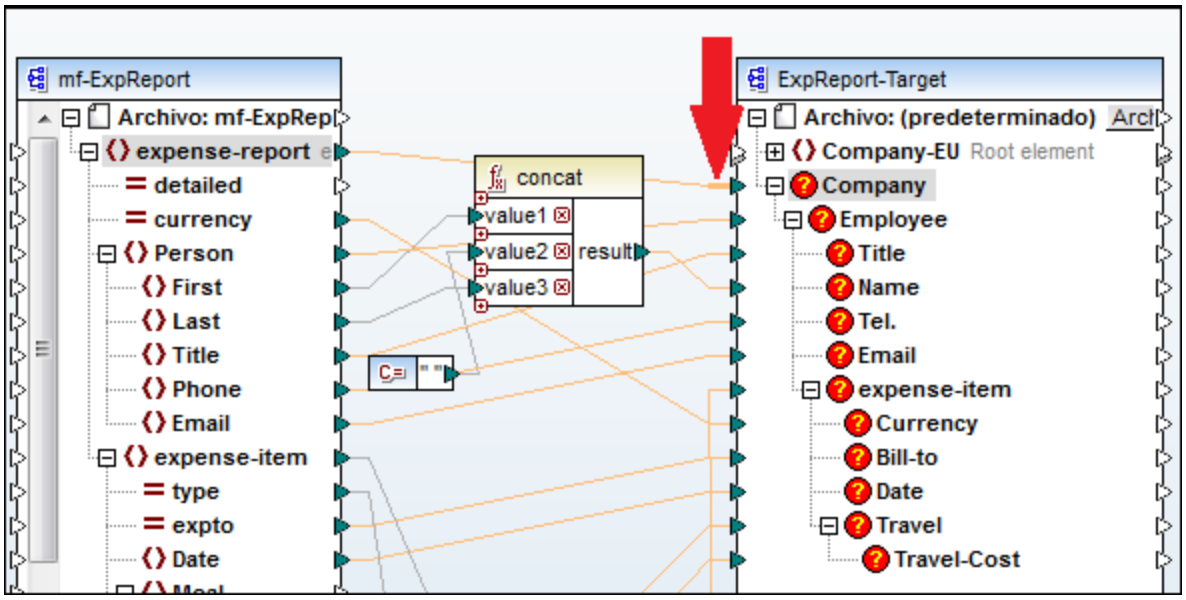



2. Abra **`ExpReport-Target.xsd`** en un editor (p.ej. Altova XMLSpy) y cambie el elemento raíz `Company` en el esquema de destino a `Company-EU`. No es necesario que cierre MapForce.
3. Una vez haya editado el elemento raíz del esquema de destino MapForce le informará de que se han cambiado archivos. 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`.

 © 2019-2025 Altova GmbH




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 41. 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.

## 2.2.5 Conservar conexiones tras eliminación de componentes

MapForce permite conservar conexiones (secundarias) aunque se hayan eliminado varios componentes de transformación 34, como variables, filtros o componentes de ordenación, asignaciones de valores, componentes de entrada simples o componentes SQL/NoSQL-WHERE/ORDER. Las conexiones pueden ser

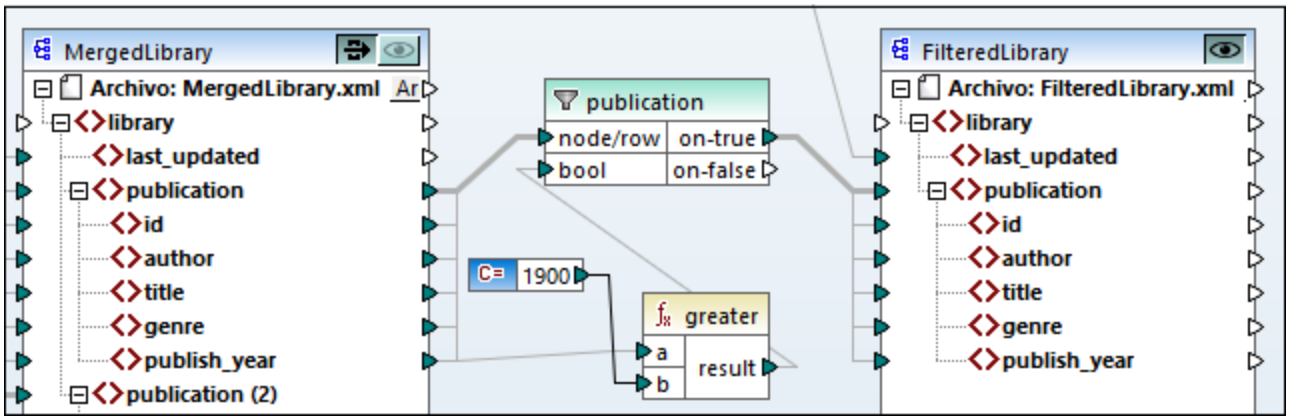
individuales o múltiples. Mantener conexiones puede ser especialmente útil si tiene varias conexiones secundarias, ya que no tiene que restaurar cada una de ellas manualmente después de eliminar el componente de transformación en cuestión. Para habilitar esta característica vaya a **Herramientas | Opciones | 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 18:

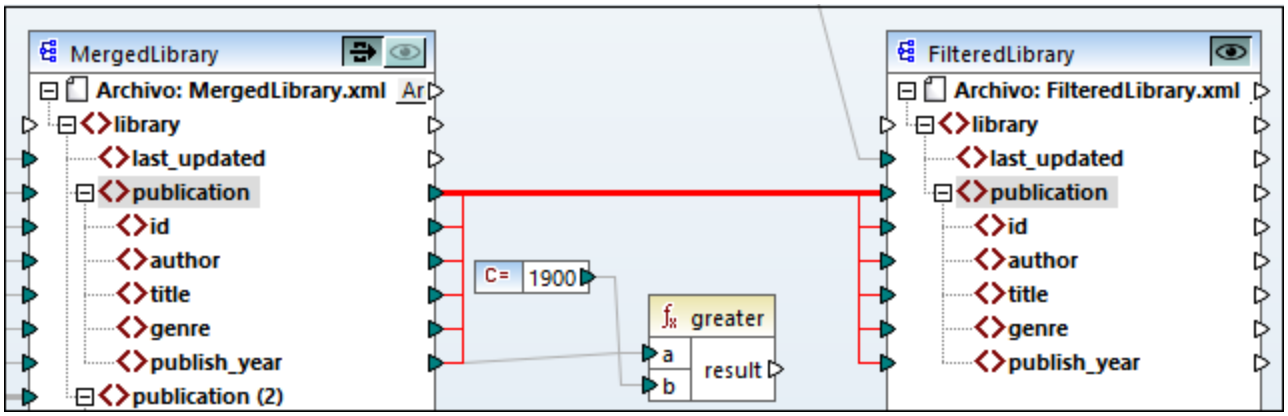
*Antes de eliminar el componente*

En la imagen siguiente se ve que existen conexiones de copia total 57 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.

 © 2019-2025 Altova GmbH

# 2.3 Procedimientos y funciones 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 66
- Generación de código 68
- Características de la vista Texto 71
- Búsqueda en la vista Texto 74
- Configuración de la asignación 78


## 2.3.1 Validación

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 27. La ventana Mensajes también permite ejecutar acciones relacionadas con los mensajes 27. 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 35:

- 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

 © 2019-2025 Altova GmbH

externa. Para generar la vista previa de los resultados MapForce ejecuta la asignación y rellena el panel Resultados 30.

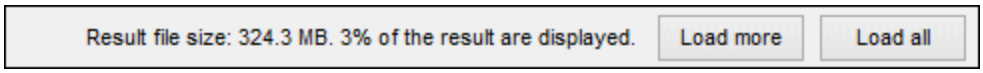
Una vez disponibles en el panel **Resultados**, los datos se pueden validar y guardar. También puede usar el comando **Buscar** (**Ctrl+F**) para encontrar un patrón de texto en el archivo de salida. Para más información consulte Búsqueda en la vista Texto 74. Los mensajes de error, advertencia e información relacionados con la ejecución de la asignación aparecen en la ventana Mensajes 27.

Si selecciona C++, C# o Java (*ediciones Professional y Enterprise*) como lenguajes de transformación 19, 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.

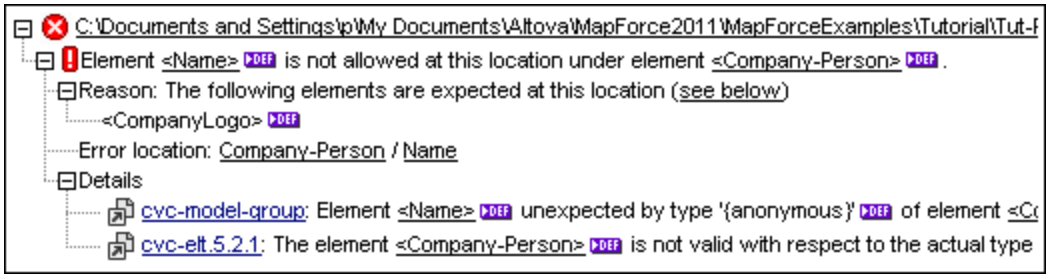
*Cargar opciones*
A la hora de generar una previa de archivos de gran tamaño, MapForce limita la cantidad de datos que se presentan en el panel **Resultados**. En concreto se presenta solamente parte del archivo y aparece el botón **Cargar más** en la parte inferior del panel (*imagen siguiente*). Al 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 1128.




*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 **Validar el archivo de salida** del menú **Resultados**.

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**.




## 2.3.2 Generación de código

El generador de código es una función integrada en MapForce que le permite generar código a partir de archivos de asignación. Puede utilizar el código generado para ejecutar sus asignaciones de datos fuera de MapFprce, lo que le permitirá automatizar sus operaciones de asignación. Estos son los lenguajes de transformación de datos 19 en los que puede generar código:

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

Puede generar código a partir de un solo diseño de asignación (**`.mfd`**) o a partir de un proyecto de asignación (**`.mfp`**). Sin embargo, sólo puede generar código a partir de un proyecto en las ediciones Enterprise y Professional. Para más detalles siga leyendo.

*Puntos importantes*
A la hora de generar código debe tener en cuenta los siguientes aspectos:

- Algunas funciones de MapForce no son compatibles con el código de programa generado. Para más detalles consulte el apartado Funciones compatibles en el código generado 1150.
- Para obtener más información sobre la gestión de rutas de acceso en el código generado consulte el apartado Rutas de acceso según el entorno de ejecución 46.
- *Las ediciones Enterprise y Professional:* En la sección *Generación* del cuadro de diálogo **Opciones** puede configurar las opciones generales de generación de código. Para más detalles consulte el apartado Generación 1131.
- *Las ediciones Enterprise y Professional:* La compatibilidad con las conexiones a bases de datos varía según la plataforma, y hay tipos de conexión que no son compatibles con todas las plataformas. Si su asignación se conecta a una base de datos, elija una conexión de base de datos que sea compatible con el entorno de destino para el que genera el código. Para obtener más información, consulte Asignaciones de bases de datos en distintos entornos de ejecución 175.

*Información de compatibilidad*
En la siguiente tabla, puede ver un resumen de la compatibilidad con los lenguajes de programación C++, C# y Java.

 © 2019-2025 Altova GmbH

| 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<br><br>Marcos de destino:<br>• .NET Framework<br>• .NET Core 3.1<br>• NET 5.0<br>• NET 6.0<br>• NET 8.0 | Java SE JDK 8, 11, 17, 21 (incluido OpenJDK)<br>Eclipse 4.4 o superior<br>Apache Ant |
| **Implementaciones XML DOM** | MSXML 6,0<br>Apache Xerces 3 | System.Xml | JAXP |
| **API de la base de datos** | ADO | ADO.NET | JDBC |

## Generar código a partir de una asignación de datos

Para generar código a partir de un diseño de asignación (**`.mfd`**), siga estos pasos:

1. En el cuadro de diálogo **Opciones** (para C# y C++) o en las opciones de la asignación 78, vaya a la sección *Generación* y seleccione las opciones de generación de código relevantes. Para más detalles sobre la configuración de la generación de código en el cuadro de diálogo **Opciones**, véase el apartado Generación 1131.
2. Haga clic en **Archivo | Generar código en** y seleccione el lenguaje de transformación correspondiente. Otra opción es seleccionar **Archivo | Generar código en el idioma seleccionado**. En este caso, el código se generará en el idioma que haya seleccionado en la barra de herramientas.
3. Seleccione el directorio de destino para los archivos que se deben generar y haga clic en **Aceptar** para confirmar. MapForce genera el código y muestra el resultado de la operación en la ventana Mensajes 27.

## Generar código a partir de un proyecto (ediciones Professional y Enterprise)

Puede generar código C# a partir de proyectos de asignación (**`.mfp`**) que consistan en varios archivos de diseño de asignación (**`.mfd`**). Tenga en cuenta que 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 1150.

Para generar código a partir de un proyecto de asignación, siga las instrucciones a continuación.

1. Abra el proyecto de asignación correspondiente a partir del cual quiere generar el código.
2. Haciendo clic con el botón derecho en el nombre del proyecto en la ventana Proyecto y entonces eligiendo el comando de menú **Propiedades** del menú contextual. Otra opción es hacer clic en el nombre del proyecto y seleccionar el comando **Propiedades** del menú **Proyecto**.
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 **Generar código para todo el proyecto**.

Independientemente del idioma seleccionado en el cuadro de diálogo **Propiedades del proyecto**, siempre puede optar por generar el código del proyecto en un idioma diferente, seleccionando el comando de menú **Proyecto | Generar código en | <idioma>**.

Los progresos y resultados de la generación de código aparecen en la ventana Mensajes. Por defecto, el nombre de la aplicación que se ha generado es igual al nombre de proyecto. Si el nombre del proyecto contiene espacios, éstos se convertirán en guiones bajos en el código generado. El código se genera por defecto en el directorio donde está ubicado el proyecto de MapForce, en el subdirectorio **`output`**.

Puede modificar el directorio de salida y/o el nombre del proyecto en el cuadro de diálogo **Propiedades del proyecto**. Si su proyecto de MapForce tiene carpetas, podrá cambiar las opciones de generación de código para cada una de las carpetas por separado: Haga clic con el botón derecho en una carpeta de interés y seleccione **Propiedades** del menú contextual. Sino todas las carpetas del proyecto heredan por defecto las opciones de generación de código del nivel superior. Para obtener más información sobre los proyectos, su configuración y los procedimientos relacionados con ellos, consulte el apartado Proyectos 98.

## Siguientes pasos

Los siguientes pasos varían en función del lenguaje de transformación que haya seleccionado para la generación de código. Si ha generado código en XSLT 1-3 o XQuery, el siguiente paso será ejecutar la transformación desde la línea de comandos (*ver detalles más abajo*).

Si ha generado código Java, C# o C++, los siguientes pasos consistirán en compilar y ejecutar el código generado. Para más información sobre estos procedimientos, consulte Generador de código 911. También puede modificar el código Java, C# y C++ generado e integrarlo en su código personalizado. Para más detalles consulte Integrar código generado 919.

*Código XSLT y XQuery*
Una vez generado el código XSLT 1-3, la carpeta de destino incluirá los siguientes archivos:

1. Un archivo de transformación XSLT que tiene este formato: **`<Mapping>MapTo<TargetFileName>.xslt`**. `<Mapping>` es el valor del campo *ApplicationName* en la configuración de la asignación 78. `<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 41 y Rutas de acceso según el entorno de ejecución 47.
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.

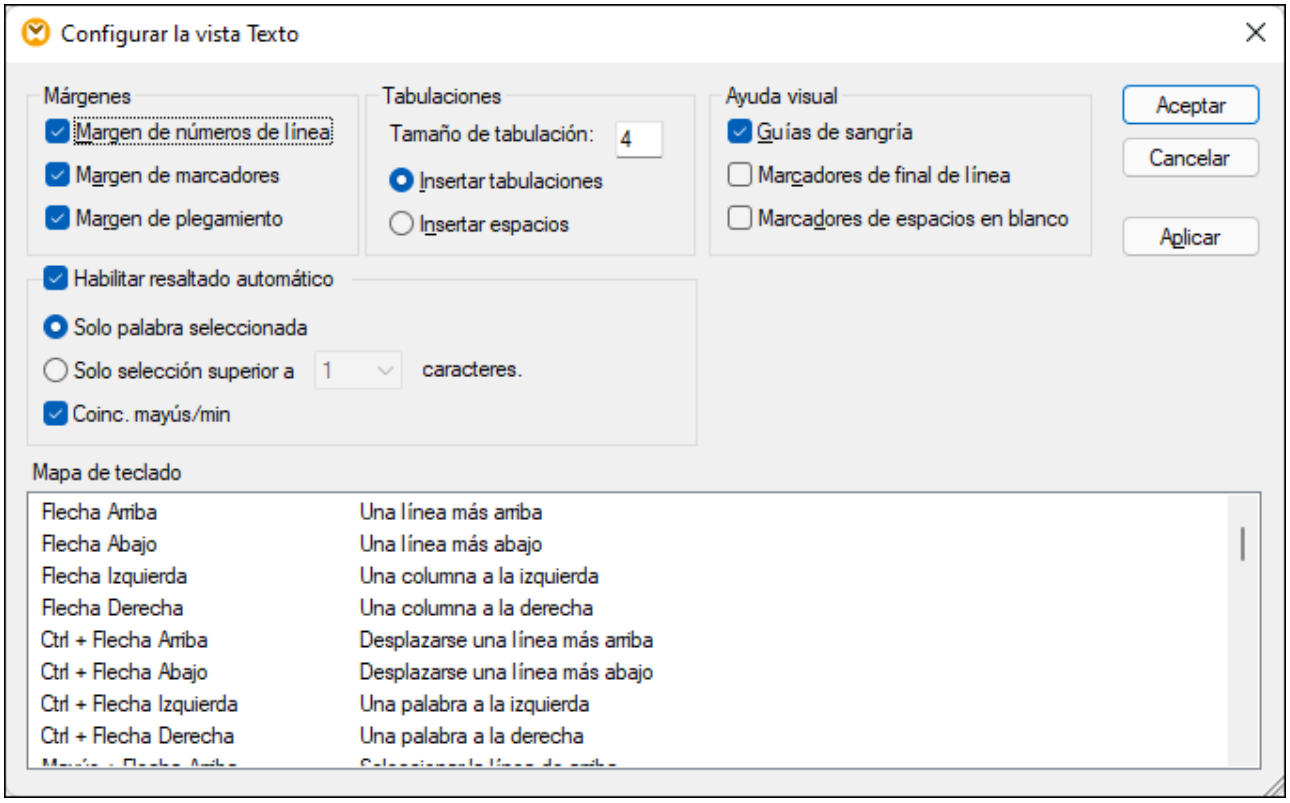
Si se trata de una asignación encadenada 119, se generará un archivo de transformación diferente para cada componente de destino.

La generación de código XQuery es similar a la generación de código XSLT, con la diferencia de que los archivos de transformación tienen una extensión **`.xq`** y el siguiente formato: **`<Mapping>MapTo<TargetFileName>.xq`**.

 © 2019-2025 Altova GmbH

## 2.3.3 Características de la vista Texto

Los paneles Resultados 30, XQuery 29 y XSLT 28 ofrecen varias funciones visuales para facilitar la consulta y edición del texto, como p.ej. los márgenes, el resaltado de texto, los guías de sangría, los marcadores de final de línea y los marcadores de espacios en blanco. 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 sólo 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.

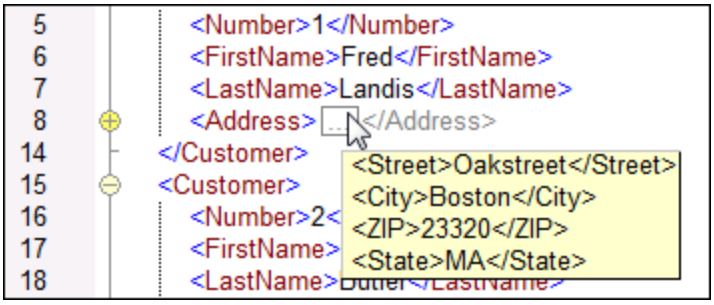
```
6      <LastName>Little</LastName>
7      <Address>
8        <Street>Long Way</Street>
9        <City>Los-Angeles</City>
10       <ZIP>34424</ZIP>
11       <State>CA</State>
12     </Address>
13   </Customer>
14   <LineItems>
15     <LineItem> ... </LineItem>
24     <LineItem> ... </LineItem>
33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

 © 2019-2025 Altova GmbH




## Habilitar el resaltado automático

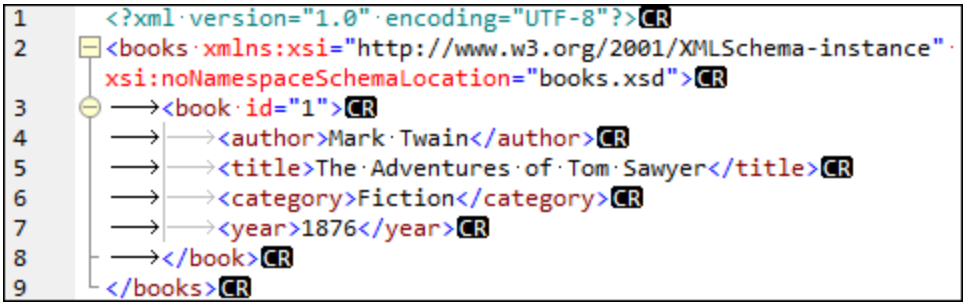
La opción **Habilitar el resaltado automático** permite visualizar todas las coincidencias del texto seleccionado. La selección se resalta en azul pálido y las coincidencias se resaltan en naranja pálido. La selección y sus coincidencias también se indican con cuadrados grises en la barra de desplazamiento, 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

*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.

## 2.3.4 Búsquedas en la vista Texto

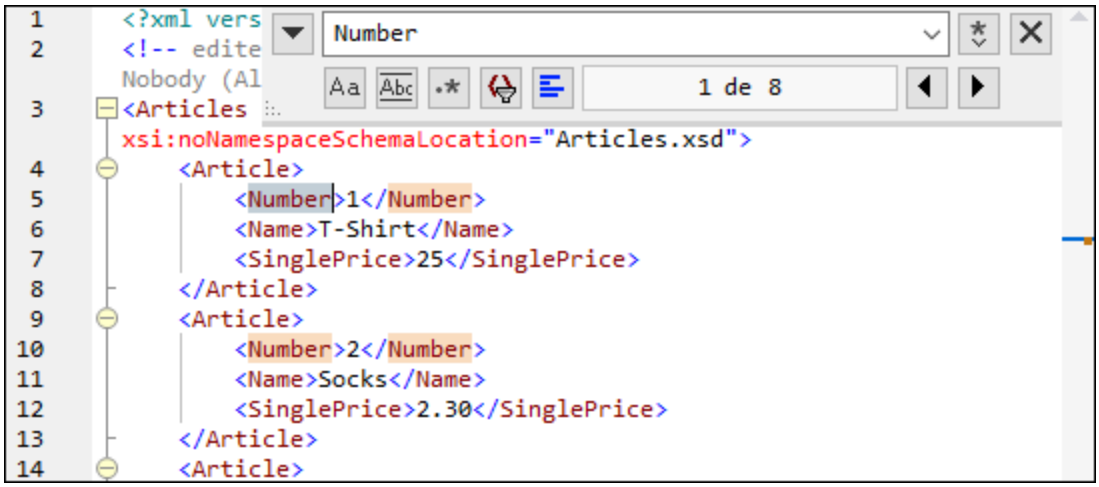
Puede realizar búsquedas en el texto de los paneles **Resultados**, **XQuery** y **XSLT** con varias opciones y ayudas visuales.

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.

 © 2019-2025 Altova GmbH




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:

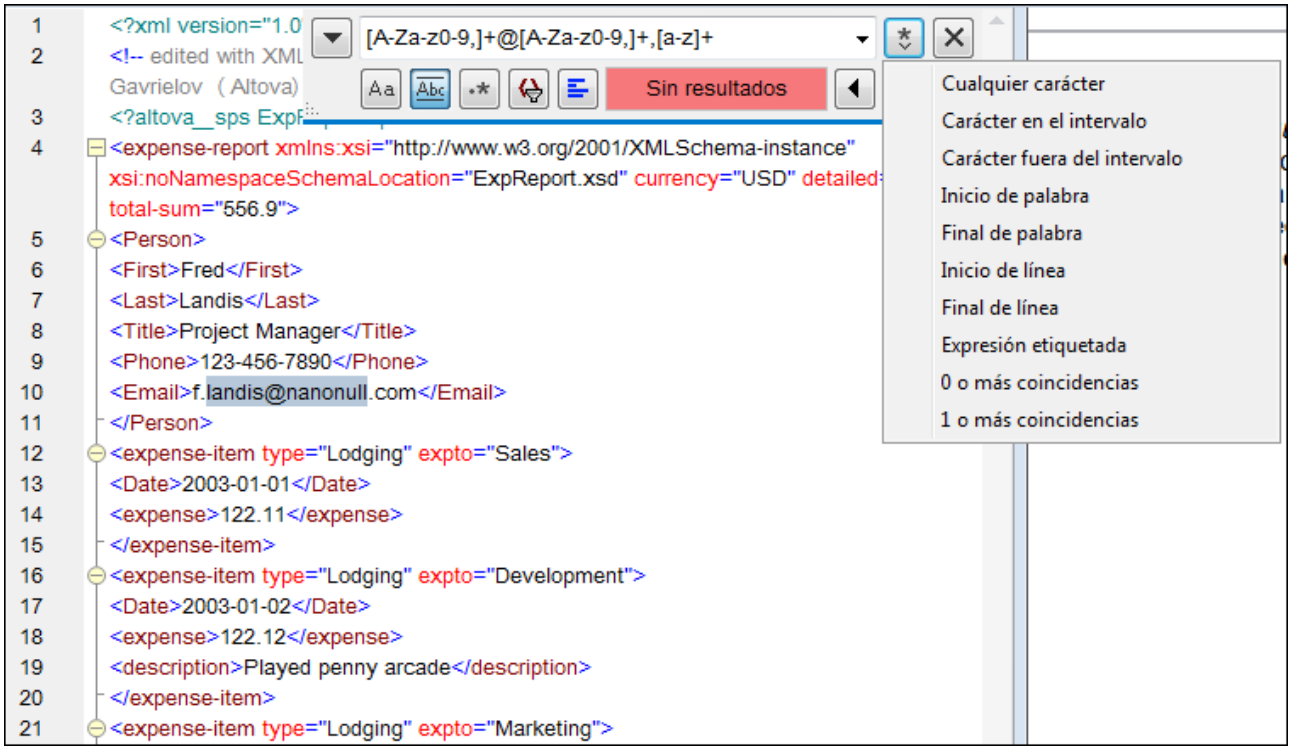
| Opción | Icono | Descripción |
|---|---|---|
| **Coinc. mayús/min** | Aa | La búsqueda tiene en cuenta las mayúsculas y minúsculas a la hora de realizar la búsqueda (`Address` no es lo mismo que `address`). |
| **Sólo palabras completas** | Abc | Solo se consideran coincidencias las palabras completas. |
| **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 |

| | | |
|---|---|---|
| | | 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) |

 © 2019-2025 Altova GmbH

| | |
|---|---|
| | 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.<br><br>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

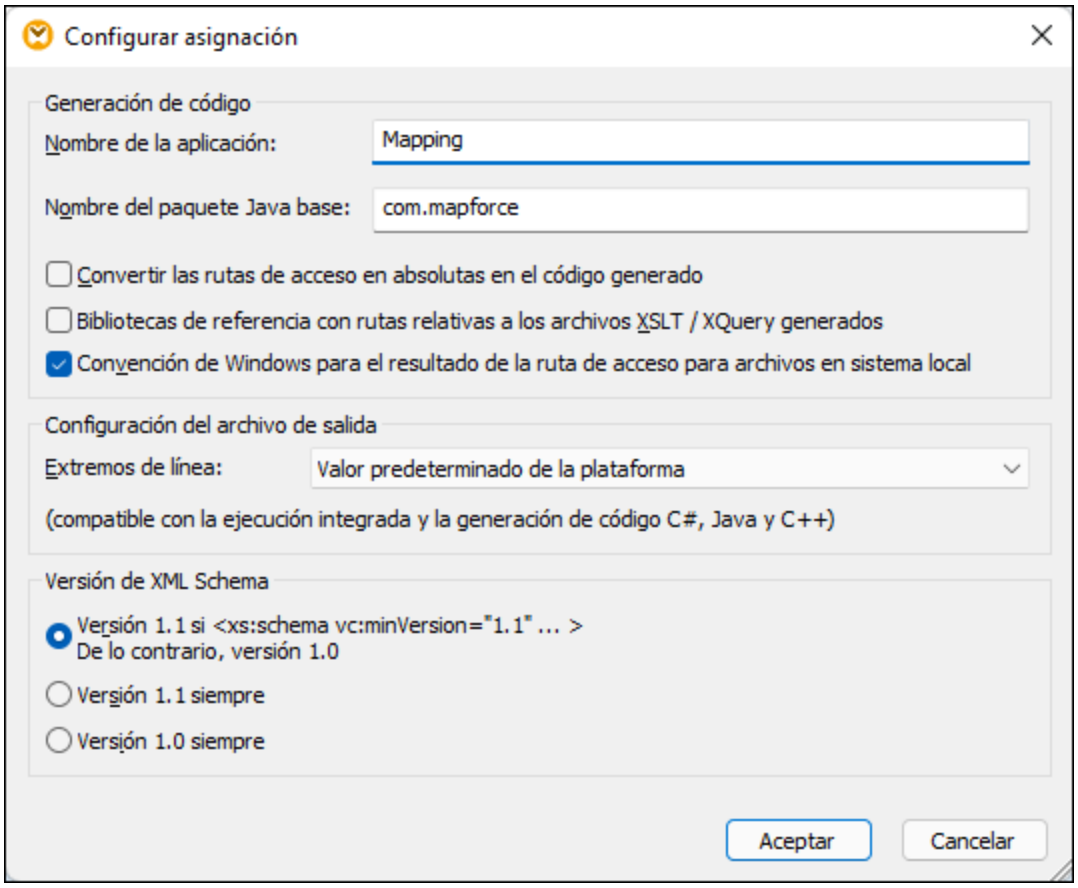
Puede buscar cualquiera de los caracteres especiales siguientes dentro del texto si habilita la opción **Usar expresiones regulares**:

- `\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**.

## 2.3.5 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 **Configuración de la asignación**. También puede hacer clic con el botón derecho en un área vacía del panel de asignación y seleccionar **Configuración de la 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 (*sólo disponible en las ediciones MapForce Professional y MapForce Enterprise*).
  - *Nombre del paquete Java base (sólo 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

 © 2019-2025 Altova GmbH

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 46 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 (*sólo 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 47.

- *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 (*sólo 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 (disponible en las ediciones Professional y Enterprise)

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 macOS 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 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 (edición 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.

## 2.4 Reglas y estrategias básicas

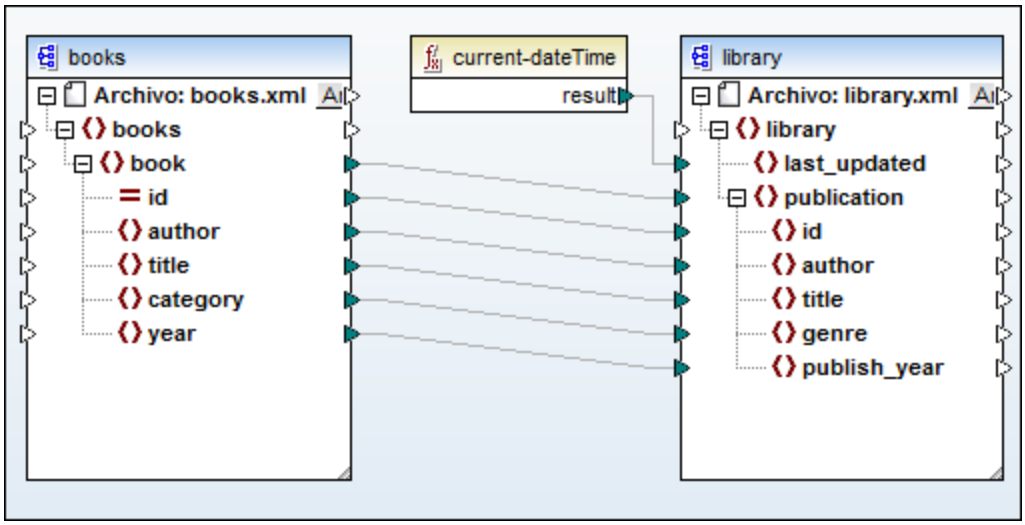
Por lo general, MapForce asigna los datos de manera intuitiva, pero puede que encuentre situaciones en las que los resultados contengan demasiados o demasiados pocos elementos. Este apartado describe las principales reglas de asignación de datos y le ayuda a evitar situaciones en las que la asignación produce un resultado no deseado debido a conexiones o contextos incorrectos.

Para ser válida, una asignación de datos debe incluir al menos un componente de entrada conectado con al menos un componente de destino. Un componente de entrada lee datos, normalmente de un archivo o de una base de datos. Un componente de destino escribe datos, normalmente en un archivo o en una base de datos.

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 eficiencia del algoritmo de asignación depende principalmente de estas conexiones. También puede adaptar algunas opciones a nivel de asignación 78, de componente 41 o incluso de conexión 52, pero son las *conexiones* de la asignación las que determinan cómo se procesan los datos.

### Reglas de asignación

Este subapartado describe las principales normas de procesamiento de datos. Para illustrar las reglas, utilizaremos la asignación de datos que se muestra a continuación.




*Regla 1*

Por cada elemento de origen, se crea un elemento en el destino. **Esta es la principal regla de asignación de datos en MapForce.** En 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 Secuencias 81).

*Regla 2*

MapForce siempre inicia el procesamiento desde el *elemento (nodo) raíz de destino*. Aquí es donde comienza realmente la ejecución de una asignación de datos. Tras procesar el nodo de destino, MapForce avanza por la jerarquía.

 © 2019-2025 Altova GmbH

*Regla 3*
Cada conexión crea el *contexto actual de asignación*. El contexto determina qué datos están disponibles *para el nodo de destino actual y sus descendientes*. El contexto, por tanto, determina qué elementos de origen se copian realmente del componente de origen al 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 el apartado Contexto y orden de procesamiento 82.

*Regla 4*
Cuando se traza una conexión desde un elemento de origen a un elemento de destino, la asignación lee los datos de ese elemento de origen, pasa los datos al elemento de destino y genera datos de instancia en el formato de destino.

El elemento de origen puede aparecer cero, una o múltiples veces en el archivo de instancia de origen. Independientemente del número de apariciones, MapForce siempre muestra únicamente jerarquías de esquemas en los componentes estructurales (por ejemplo, XML). Esa es la razón por la que, por ejemplo, el componente de origen de la asignación anterior solo tiene un elemento book.

## 2.4.1 Secuencias

Comprender las secuencias es esencial para dominar las técnicas de asignación de datos en MapForce. La regla principal para las secuencias es la siguiente:

> *Si se conecta una secuencia de más de un nodo a un nodo de destino, se crea un bucle que genera tantos nodos de destino como nodos de origen haya.*

Si hay un filtro a mitad de camino, MapForce comprueba si hay al menos una condición booleana en la secuencia que dé como resultado true y asigna los datos que cumplen esta condición al destino.

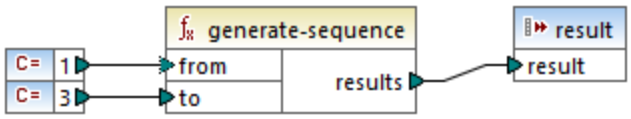
Si la secuencia está vacía no se genera nada en el lado de destino. Por ejemplo, si el componente de destino es un documento XML y la secuencia de origen está vacía, entonces no se crearía ningún elemento XML en el componente de destino.

### Excepciones

La regla sobre secuencias tiene algunas excepciones (*ver más abajo*).

*Excepción 1*
Si el elemento de destino es un elemento raíz XML, se crea una sola vez. Si conecta a él una secuencia, el resultado puede no ser válido según el esquema. 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.

*Excepción 2*
Si el elemento de destino acepta solamente un valor, se crea una sola vez. Los ejemplos de elementos que solo aceptan un valor son: atributos XML, campos de base de datos y componentes de salida simples. 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 de salida simple que acepta solamente un valor. Los otros dos valores se ignoran.




*Excepción 3*
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, de acuerdo con al esquema, debe existir) y crea un solo elemento en el componente de destino. Para deshabilitar este comportamiento, desactive la casilla *Optimización de procesamiento de datos de entrada basada en minOccurs/maxOccurs* de la configuración del componente 139.

### Funciones

Si una función no secuencial recibe una secuencia como entrada, entonces se llama a la función tantas veces como se indique yla función produce tantos resultados como elementos haya en la secuencia. Si una función recibe una secuencia vacía como entrada, devuelve un resultado vacío y, en consecuencia, no produce ningún resultado.

*Funciones que devuelven un valor aunque reciban una secuencia vacía*
Sin embargo, hay algunas categorías o funciones que, por motivos de diseño, devuelven un valor incluso aunque reciban 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 sustituir un valor ausente, añada la función **`substitute-missing`** a la asignación y sustituya el valor ausente por un valor sustituto de su elección. Alternativamente, puede conseguir el mismo resultado utilizando valores predeterminados y funciones de nodo 469.

*Funciones con varias entradas*
Las funciones pueden tener distintos elementos de entrada. Si se conecta una secuencia a cada entrada, se obtiene un producto cartesiano de todas las entradas (un conjunto de todos los pares ordenados que pueden crearse combinando elementos de dos o más conjuntos). Esto es, por lo general, un resultado no deseado. Para evitarlo, conecte solo una secuencia a una función con varios parámetros. Los demás parámetros deben conectarse a elementos «únicos» de los primarios o de otros componentes.

## 2.4.2 Contexto y orden de procesamiento

El contexto es uno de los aspectos más cruciales en la ejecución de las asignaciones de datos. Comprender el contexto es clave para lograr nuestro objetivo en el resultado. El contexto determina qué datos están disponibles *para el nodo de destino actual y sus descendientes* y qué elementos de origen se seleccionan realmente y se copian del componente de origen al de destino.

MapForce siempre establece el contexto actual empezando por el *(nodo) elemento raíz de destino*. Aquí es donde comienza realmente la ejecución de una asignación de datos. Tras procesar el nodo de destino,

 © 2019-2025 Altova GmbH

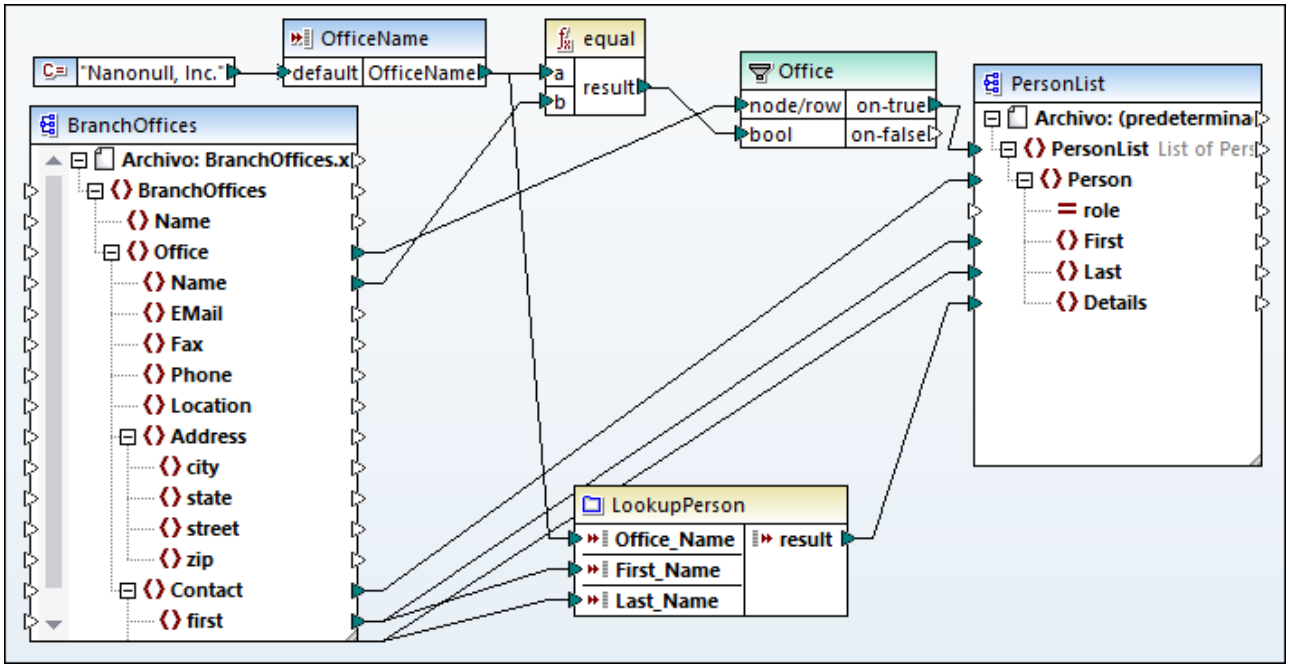
MapForce avanza por la jerarquía. Para cada nodo de destino, las conexiones se rastrean hasta los nodos de origen, a través de cualquier función intermedia.

Para cada nodo de destino, se establece un nuevo contexto que contiene inicialmente todos los nodos de origen actuales del nodo primario y todos los resultados de las funciones. Mientras MapForce evalúa las conexiones que conducen directa o indirectamente al nodo de destino, se agregan al contexto fuentes y resultados de funciones adicionales. De este modo, los nodos de destino del mismo nivel son independientes entre sí, pero tienen acceso a todos los orígenes de sus nodos primarios.

Para repasar las reglas fundamentales de la ejecución de asignaciones de datos, consulte Reglas y estrategias básicas 80.

## Ejemplo

Para poner en práctica los principios descritos anteriormente, analizaremos la siguiente asignación de datos: **`MapForceExamples\PersonListByBranchOffice.mfd`** (*imagen siguiente*). El objetivo principal de la asignación de datos es obtener una lista de empleados con sus datos de una oficina determinada.




*Estructuras de origen y de destino*

En nuestro ejemplo, la asignación contiene un componente XML de origen y un componente XML de destino. Además, hay un componente de entrada simple que también actúa como origen y suministra el nombre de la oficina cuyos empleados nos interesan. La asignación de datos también incluye varios componentes de transformación, como un filtro y algunas funciones.

Es importante destacar que MapForce siempre muestra únicamente jerarquías de esquemas en los componentes estructurales (por ejemplo, XML). Por ejemplo, el componente `BranchOffices` solo muestra un elemento `Contact`, a pesar de que el archivo de instancia de origen **`BranchOffices.xml`** contiene varios nodos `Contact` con contenido diferente y en nodos primarios `Office` diferentes. A continuación, puede ver un extracto de **`BranchOffices.xml`**.

```
<BranchOffices xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="BranchOffices.xsd">
      <Name>Nanonull</Name>
      <Office>
            <Name>Nanonull, Inc.</Name>
            <EMail>office@nanonull.com</EMail>
            <Fax>+1 (321) 555 5155 - 9</Fax>
            <Phone>+1 (321) 555 5155</Phone>
            <Address>
                  <city>Vereno</city>
                  <state>CA</state>
                  <street>119 Oakstreet, Suite 4876</street>
                  <zip>29213</zip>
            </Address>
            <Contact>
                  <first>Vernon</first>
                  <last>Callaby</last>
            </Contact>
            <Contact>...</Contact>
            <Contact>...</Contact>
            <Contact>...</Contact>
            <...>
      </Office>
      <Office>
            <Name>Nanonull Partners, Inc.</Name>
            <EMail>nextoffice@nanonull.com</EMail>
            <Fax>+1 (927) 555 1845</Fax>
            <Phone>+1 (927) 555 0094</Phone>
            <Address>
                  <city>Brenton</city>
                  <state>MA</state>
                  <street>9865 Millenium Center, Suite 456</street>
                  <zip>5985</zip>
            </Address>
            <Contact>
                  <first>Steve</first>
                  <last>Meier</last>
            </Contact>
            <Contact>...</Contact>
            <Contact>...</Contact>
            <...>
      </Office>
</BranchOffices>
```

*Contexto y orden de procesamiento*

En nuestra asignación de datos, el procesamiento comienza a partir del nodo raíz de destino llamado `PersonList`. Podemos rastrear la conexión con el filtro y la función `equal` hasta su elemento de origen: el elemento `Office`.

***Una sola oficina como contexto para el destino***

Como nuestra asignación de datos tiene un filtro, solo los datos que cumplan la condición booleana se asignarán al destino. Esta condición solo se cumple una vez, porque solo hay una oficina llamada Nanonull, Inc. en el archivo XML de origen. En consecuencia, la conexión desde la lista `PersonList` a través del filtro hasta el nodo `Office` define una única oficina como contexto para todo el documento de destino. Esto

 © 2019-2025 Altova GmbH

significa que todas las conexiones posteriores a los descendientes del elemento raíz se verán afectadas por el filtro. Además, solo tendrán acceso a los datos de la oficina Nanonull, Inc. y no existirá ninguna otra oficina en el contexto actual.

***Una persona por cada contacto***

La siguiente conexión es entre los nodos `Contact` y `Person`. Según la regla general de asignación de datos 80, esta conexión creará un elemento `Person` de destino por cada `Contact` de origen. En cada iteración, esta conexión establece un `Contact` actual para las conexiones con los secundarios del elemento `Person`. Por lo tanto, las conexiones secundarias (`first` con `First`, `last` con `Last`) seleccionarán el nombre y el apellido del elemento `Contact` *actual* y los escribirán en el destino.

***Función definida por el usuario***

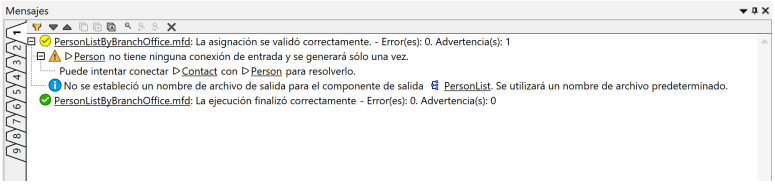
La asignación también incluye una función definida por el usuario 485 llamada `LookupPerson`. La función definida por el usuario también se ejecuta en el contexto de cada `Person`, debido a la conexión primaria entre `Contact` y `Person`. Cada vez que un elemento `Person` nuevo se crea en el lado de destino, se llama a la función para que rellene el elemento `Details` de esa persona.

La función toma tres parámetros de entrada. El parámetro `OfficeName` lee los datos del componente de entrada simple. 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 del 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 se configura la función **`LookupPerson`**, consulte Implementación de la búsqueda. 503

***Si los nodos primarios no están conectados***

Si no existiera una conexión entre `Contact` y `Person`, entonces la asignación crearía solamente un elemento `Person` con varios nodos `First`, `Last` y `Details`. En casos como este, MapForce muestra una advertencia en la ventana Mensajes (*imagen siguiente*) y sugiere conectar los nodos primarios para resolver el problema.

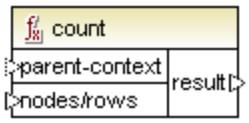



## 2.4.3 Contexto primario

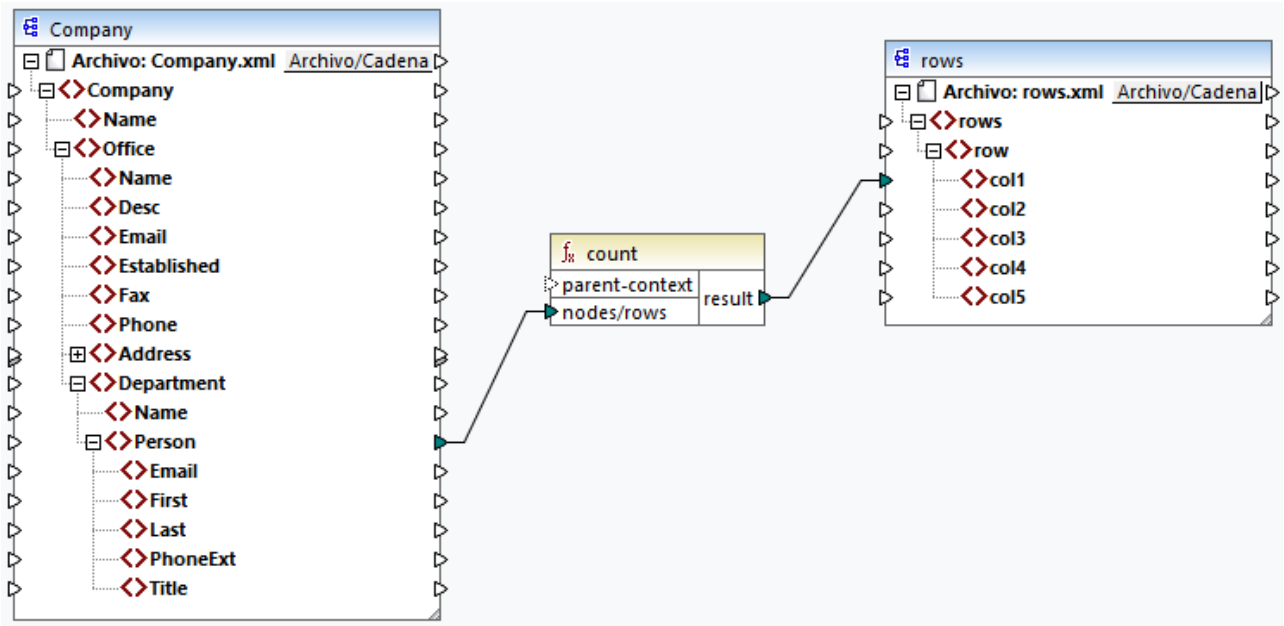
Algunos componentes de asignación de datos tienen un elemento **parent-context** opcional.

**parent-context** es un argumento opcional de algunas funciones de agregado de la biblioteca de funciones **core** de MapForce (como, por ejemplo, de las funciones **`min`**, **`max`**, **`avg`** y **`count`**). En un componente de origen que tiene varias secuencias jerárquicas, el contexto primario determina en qué conjunto de nodos debe operar la función.

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: las funciones agregadas, las variables y los componentes de combinación.




Para ver un ejemplo de la utilidad de cambiar el contexto primario, abra esta asignación: **<Documentos>\Altova\MapForce2025\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 |

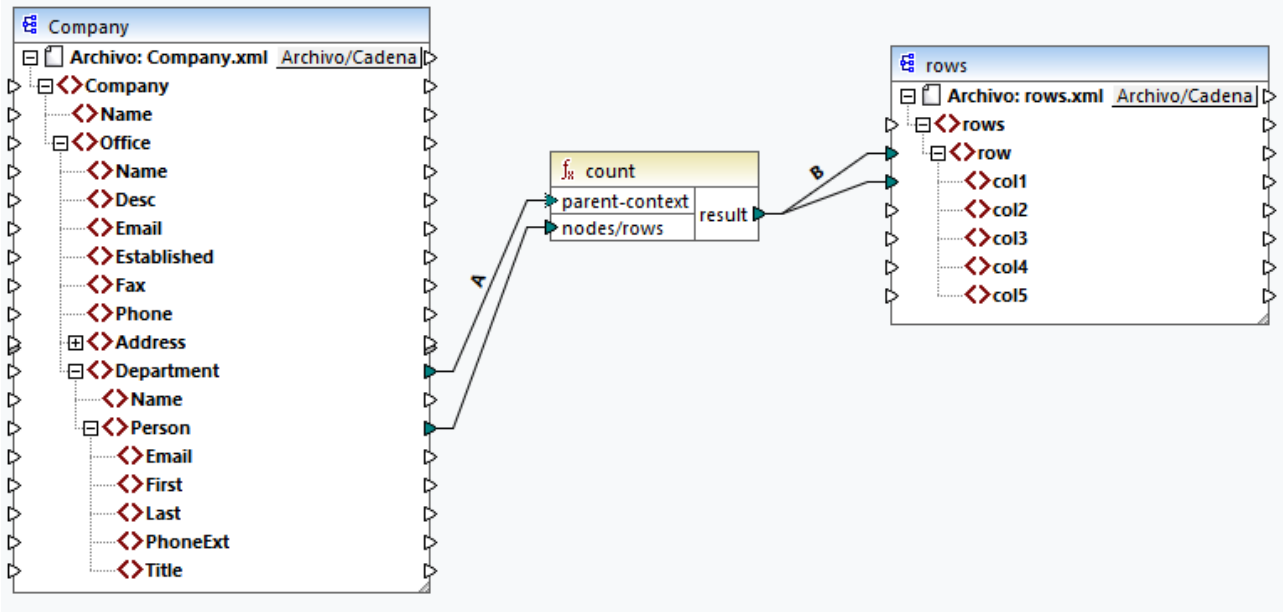
 © 2019-2025 Altova GmbH

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 de origen.

La asignación funciona así:

- Como es habitual, la ejecución de 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 de nivel 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 primario 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:

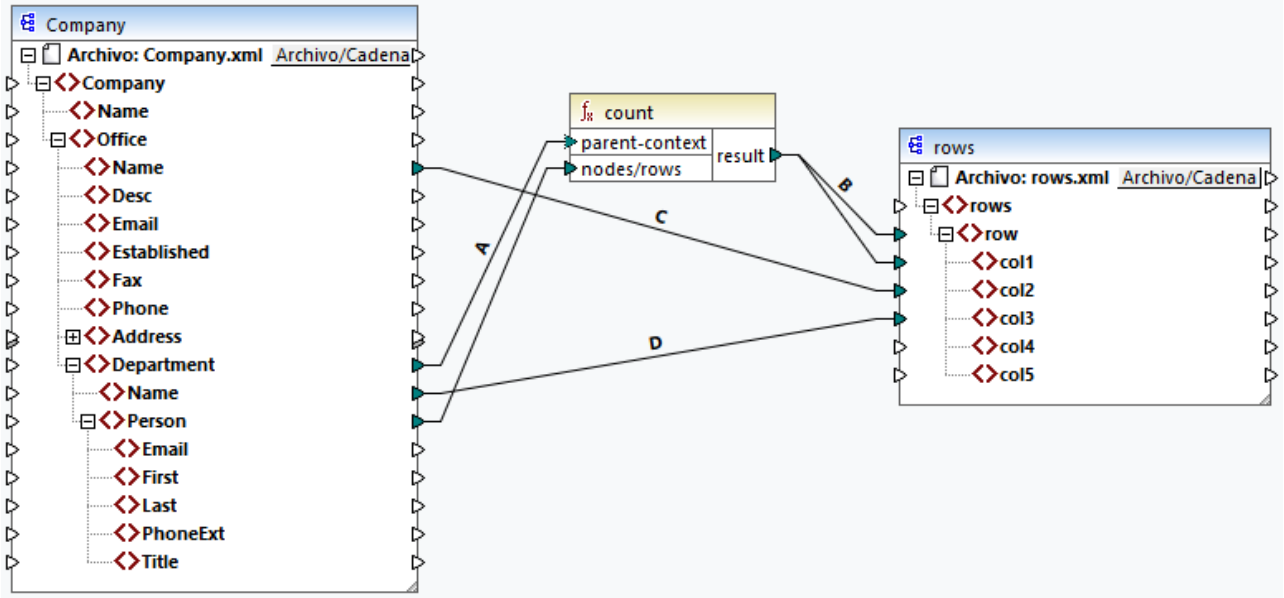



En la asignación anterior la conexión A cambia el contexto primario de la función **count** a **Department**. Como resultado, la función contará el número de personas de cada departamento. Es importante recordar que la función ahora devolverá una *secuencia* de resultados en lugar de un solo resultado 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. Ahora, el resultado de la asignación de datos ha cambiado como corresponde (observe que los números corresponden exactamente al recuento de personas en 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>
```

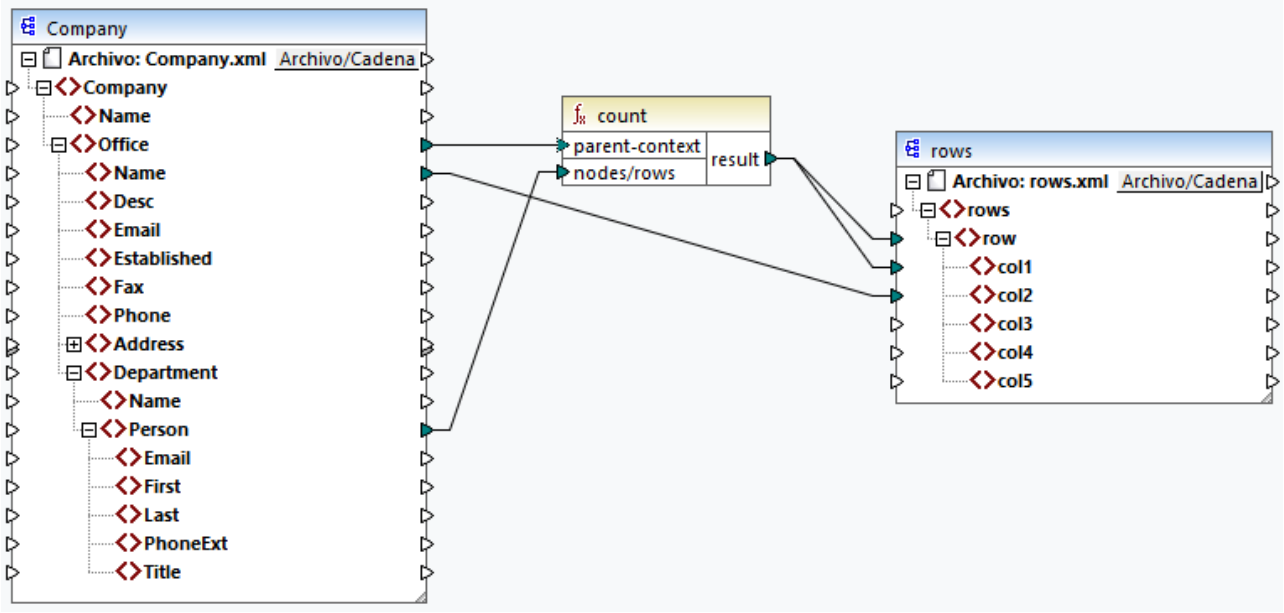
Si tenemos en cuenta que la asignación actual crea una fila por cada departamento, puede optar por copiar también el nombre de la oficina y el del departamento en el archivo de destino; para ello debe trazar las conexiones C y D:




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 número de personas de cada oficina, conecte el contexto primario de la función **count** al elemento **Office** del componente de origen.

 © 2019-2025 Altova GmbH




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>
```

## 2.4.4 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 llevar 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 tanto *a* como *b* reciben una secuencia de valores, entonces MapForce procesa así:

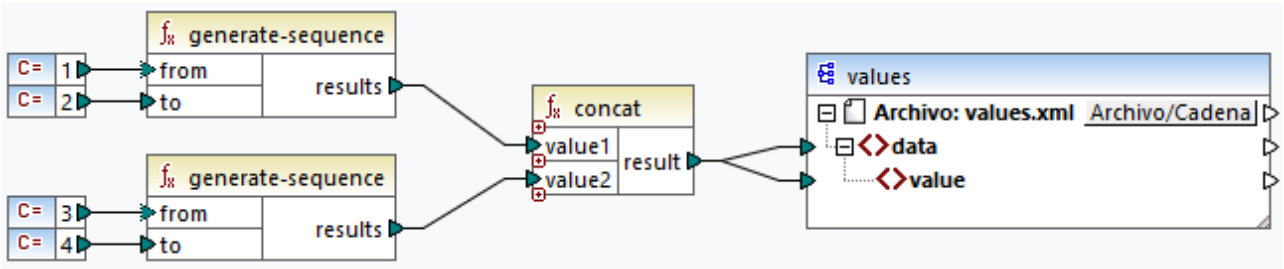
- Por cada instancia de *a*

- o Por cada instancia de *b*
  - ¿Es *a* igual a *b*?

Como puede ver en el ejemplo anterior, cada instancia de *b* se evalúa en el contexto de cada instancia de *a*. El contexto de prioridad permite alterar el proceso lógico para que cada instancia de *a* se evalúe en el contexto de cada instancia de *b*. En otras palabras, permite cambiar el bucle interno por el externo:

- Por cada instancia de *b*
  - o Por cada instancia 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 combina valores en este orden:

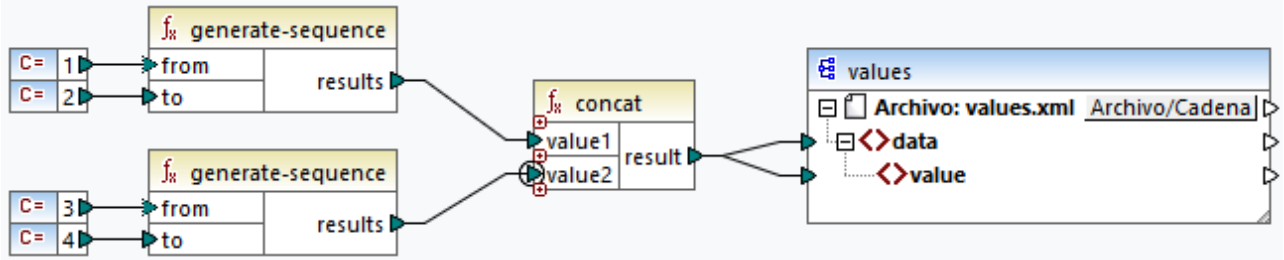
- 1 con 3
- 1 con 4
- 2 con 3
- 2 con 4

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. A continuación, puede ver la entrada del contexto de prioridad marcada con un círculo.

 © 2019-2025 Altova GmbH




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". Por consiguiente, el resultado pasa a ser:

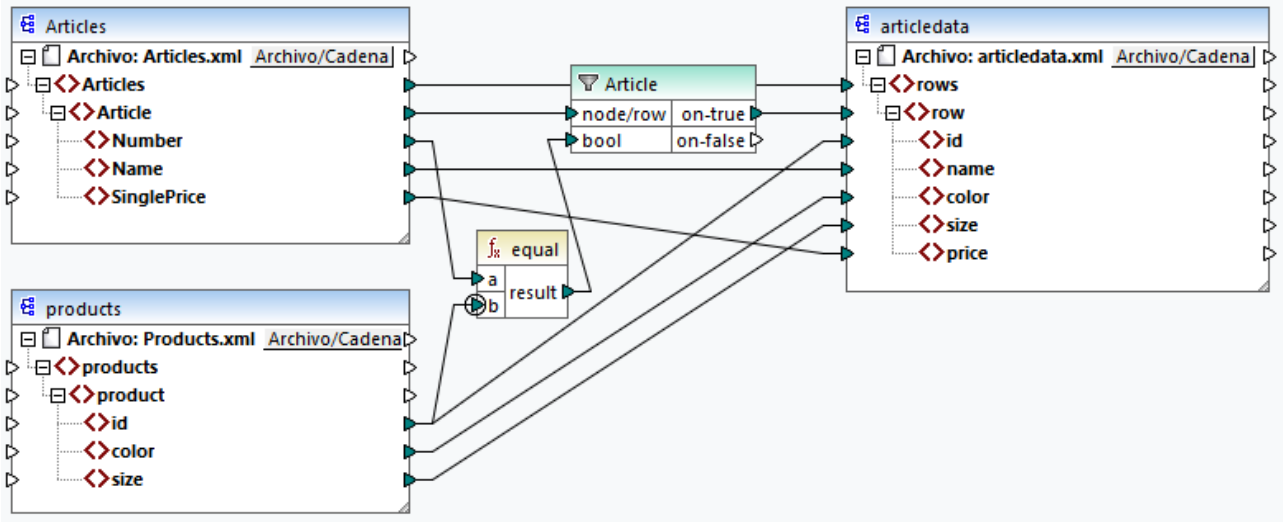
```
<data>
    <value>13</value>
    <value>23</value>
    <value>14</value>
    <value>24</value>
</data>
```

Hasta ahora solo hemos visto la parte teórica del contexto de prioridad. Para ver un ejemplo práctico consulte Ejemplo: filtrar con el contexto de prioridad 91

## 2.4.4.1 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 siguiente ruta:
**<Documentos>\Altova\MapForce2025\MapForceExamples\Tutorial\FilterWithPriority.mfd**.

**Nota:** esta asignación usa componentes XML, pero la misma lógica que se describe a continuación se aplica a todos los demás tipos de componentes de MapForce, incluidos los componentes EDI, JSON, etc. Para las bases de datos, es aconsejable realizar el filtrado utilizando componentes SQL WHERE 437 en lugar de filtros estándar.




El objetivo de esta asignación es copiar datos 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 artículos correspondiente. Tenga en cuenta que cada registro de **Artículos.xml** tiene un **Number** y cada registro de **Products.xml** tiene un **id**. Si estos dos valores son iguales, entonces todos los demás valores (**Name**, **SinglePrice**, **color**, **size**) deben copiarse en la misma **fila** del destino.

Para conseguir este objetivo se ha añadido un filtro. Cada filtro requiere una condición booleana como entrada. Solo aquellos nodos/filas que cumplan la condición se copiarán en el destino. Para ello, existe una función `equal` en la asignación de datos. 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 daría lugar a 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 de datos, 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 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:

 © 2019-2025 Altova GmbH

```
<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: Usar el contexto de prioridad

El problema anterior se soluciona añadiendo un contexto de prioridad a la función que calcula 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 procedente de **Products.xml** tiene prioridad. Esto se traduce en la siguiente lógica de asignación:

- Por cada producto se rellena la entrada **b** de la función `equal` (en otras palabras, **b** tiene prioridad). 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.

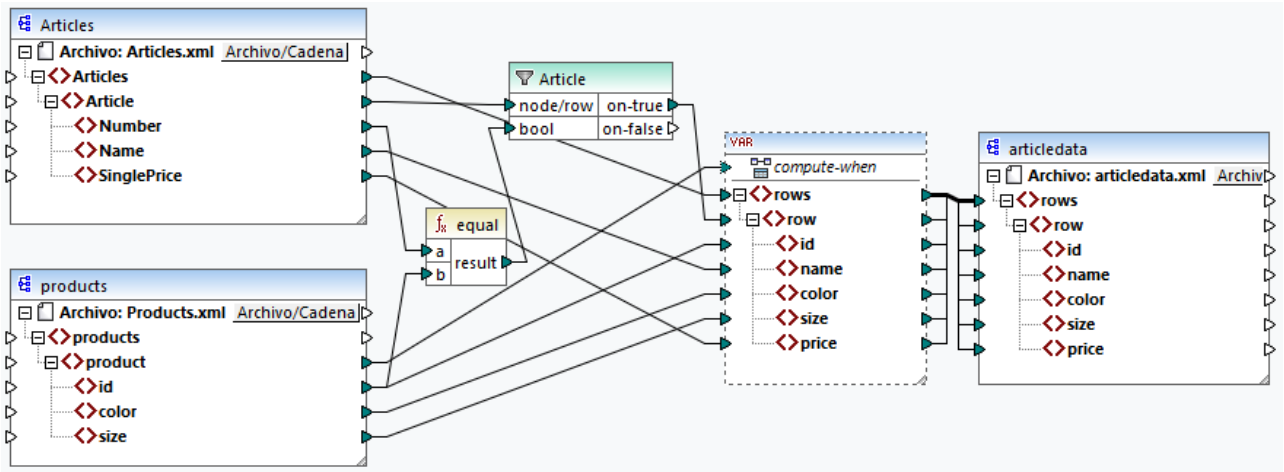
La lógica de asignación anterior produce un resultado correcto, por ejemplo:

```
<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 **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.

## 2.4.5 Varios componentes de destino

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 a la vez en MapForce: el que indique al hacer clic en 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.

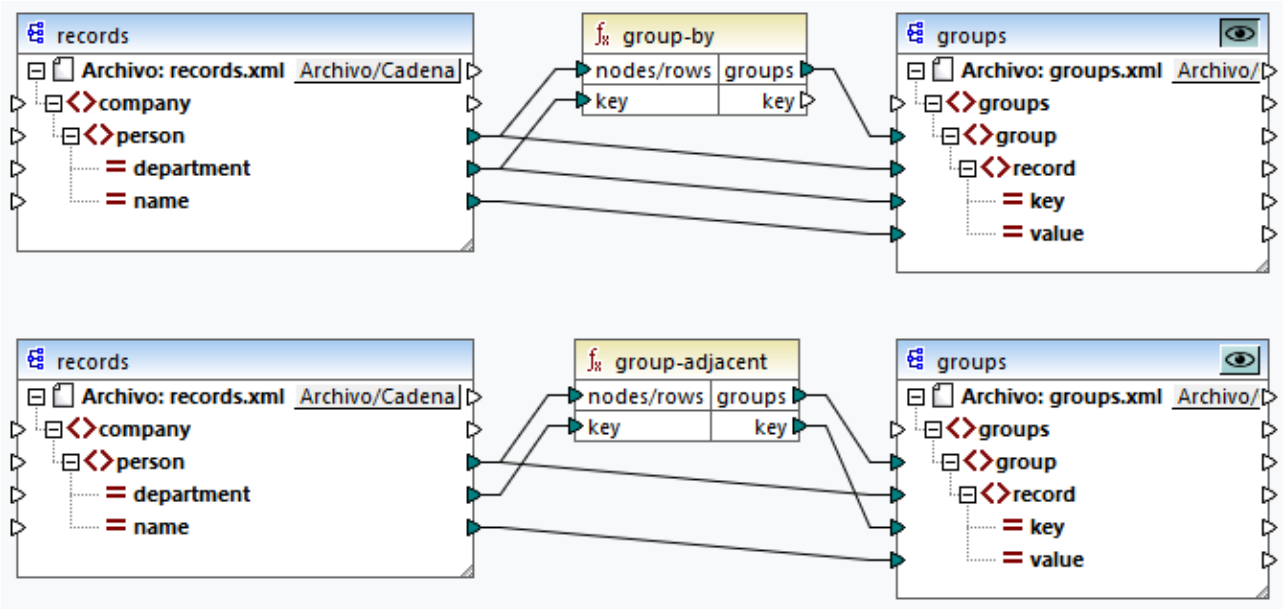
Por defecto, los componentes de destino se procesan de arriba a abajo y de izquierda a derecha. Si lo necesita, puede modificar este orden cambiando la posición de los componentes de destino en la ventana de asignación. El punto de referencia es la esquina superior izquierda de cada uno de los componentes. Tenga en cuenta que:

- Si dos componentes tienen la misma posición vertical, tendrá prioridad el componente situado más arriba.
- Si dos componentes tienen la misma posición horizontal, tendrá prioridad el componente situado más a la izquierda.

 © 2019-2025 Altova GmbH

- 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.

Para ver un ejemplo abra la asignación de ejemplo: **<Documentos>\Altova\MapForce2025\MapForceExamples\Tutorial\GroupingFunctions.mfd**. Esta asignación está compuesta por varios componentes de origen y varios de destino (en la imagen se ve solamente un fragmento).




Conforme a las reglas, el orden de procesamiento predeterminado 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:

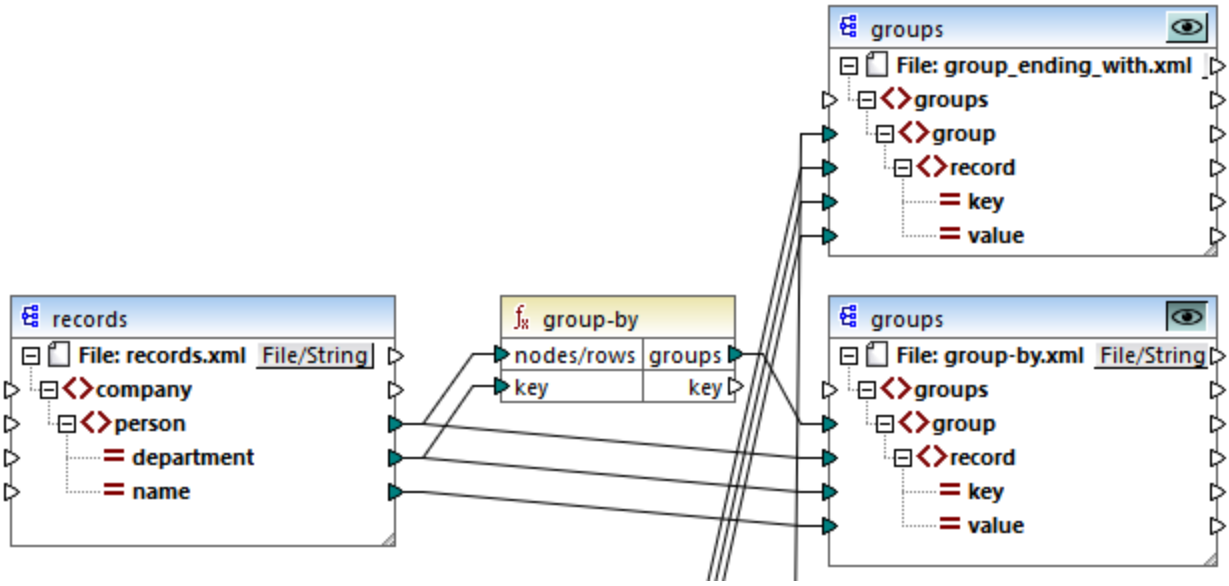
1. En el menú **Archivo**, haga clic en **Generar código en | XSLT 2.0**.
2. Cuando la aplicación lo pida, seleccione un directorio de destino para el código generado.

Una vez generado el código, 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%
```

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).

 © 2019-2025 Altova GmbH

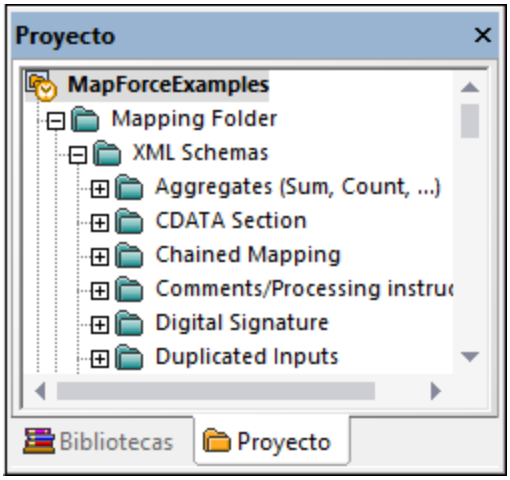
## Asignaciones encadenadas

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.

## 2.5 Proyectos

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 99 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. 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.

### 2.5.1 Aspectos básicos de un proyecto

The subsections below will help you get started with a project. Procedures associated with projects can be broadly divided into (i) creating a project, (ii) organizing a project, and (iii) performing different actions. See information about these procedures in the subsections below.

#### New project

To create a new project, take the steps below:

1. Click the button in the toolbar. Alternatively, go to the **File** menu and click **New**.
2. Select **Project File** and click **OK**.
3. Enter the project name in the **Save Project As** dialog box and click **Save**. The new project is now displayed in the **Project** window.

To close a project, go to the **Project** menu and click **Close Project**.

 © 2019-2025 Altova GmbH

## Project organization

*Add a mapping to a project*

If you want to add a currently open mapping to a project, do one of the following:

- Go to the **Project** menu and click **Add Active File to Project**.
- Right-click the relevant project in the **Project** window and select **Add Active File to Project** from the context menu.

To add existing mapping files to a project, do one of the following:

- Go to the **Project** menu and click **Add Files to Project**.
- Right-click the relevant project in the **Project** window and select **Add Files to Project**.

**Tip:** If you want to add multiple files, hold the **Ctrl** key while selecting the files in the **Open** dialog box.

*Delete a file from a project*

To remove a file or folder from a project, do one of the following:

- Choose the file you would like to delete in the **Project** window. Right-click the file and select **Delete** from the context menu.
- Select the relevant file in the **Project** window and press **Delete**.

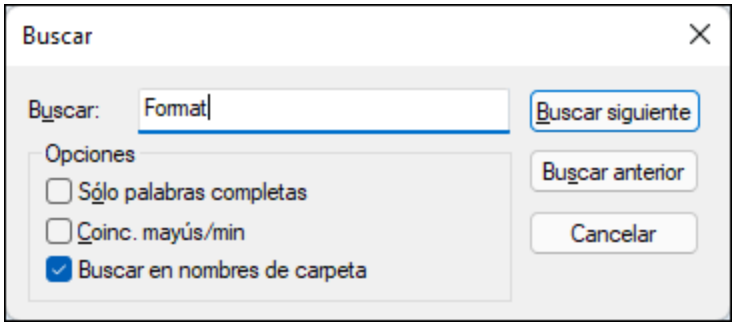
MapForce project files have a `.mfp` extension. You can open existing MapForce projects in the same way as mappings: Go to the **File** menu and click **Open**. By default, when you run MapForce for the first time, you will see the `MapForceExamples.mfp` project in the **Project** window.

## Project-related actions

*Search a project*

To search a project for files, follow the instruction below:

1. In the **Project** window, click the project or folder to be searched.
2. Press **Ctrl + F**. The **Find** dialog box allows you to define your search options. For example, if you want to include folder names in the search, select the **Find in folder names** option (*see screenshot below*).




*Generate code for your project*

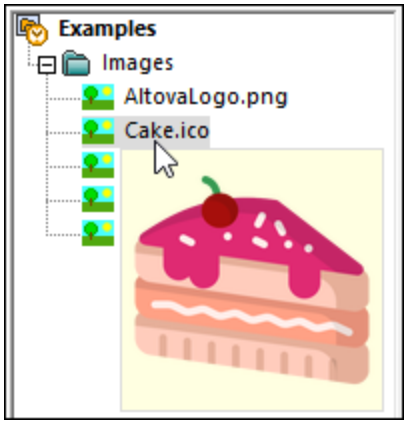
In projects, you can generate code for (i) individual mappings, (ii) a specific folder, or (iii) the entire project.

To generate code for a mapping or folder in your project, right-click the relevant mapping or folder and select **Generate Code** or **Generate code in**. If you select **Generate Code**, the code will be generated in the

language specified in the project settings [100]. You can also choose to generate code in one of the languages available in your MapForce edition. For more information, see Code Generation [68].

To generate code for the entire project, go to the **Project** menu and select **Generate Code for Entire Project**. Alternatively, right-click the name of the project in the **Project** window and select **Generate Code**. The code will be generated in the language specified in the project settings [100]. For the entire project, you can also select a language to generate code in in the **Project** menu or in the project's context menu. The choice of languages depends on your MapForce edition. For more information, see Code Generation [68].

*Preview images*
The **Project** window allows you to preview images of the following formats: **`.png`**, **`.jpeg`**, **`.gif`**, **`.bmp`**, **`.tiff`**, and **`.ico`** (*see screenshot below*). Double-clicking an image file will open it in an external application, which depends on file association in Windows.




*Watch video tutorials and add Web links*
Besides multiple sample files, the **`MapForceExamples`** project also contains links to various video tutorials on the Altova website. Double-clicking any link will open the corresponding page in your default Web browser.

You can also insert your own external links in any of the following ways:

- By right-clicking the project name or a folder of interest and selecting **Create Web Link** from the context menu.
- By selecting the project name or a folder of interest and then clicking **Create Web Link** in the **Project** menu.

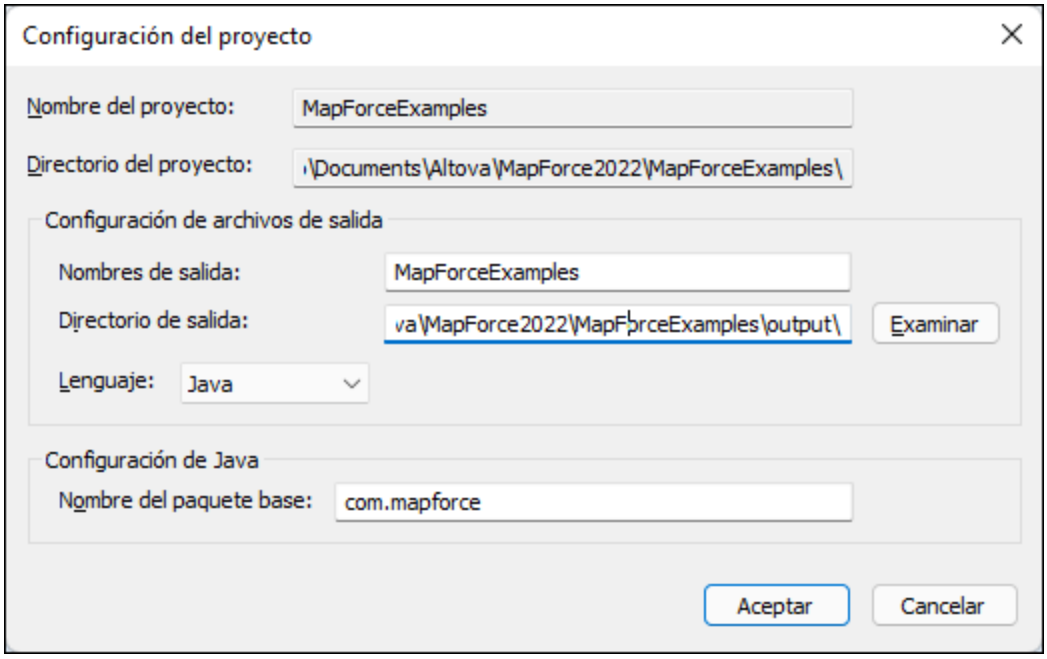
Either method causes the **Web Link Properties** dialog to pop up. Type the link's name that will be visible in the interface and the URL of this resource. You also need to select an icon for your resource: as a general Web link or as a link to a video. You can always change the location of any link, by dragging the link to the desired location. You can also use the combination of the **Ctrl + C** and **Ctrl + V** keys to copy a link and paste it into the desired location.

## 2.5.2 Configuración de 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*):

 © 2019-2025 Altova GmbH

- 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**.



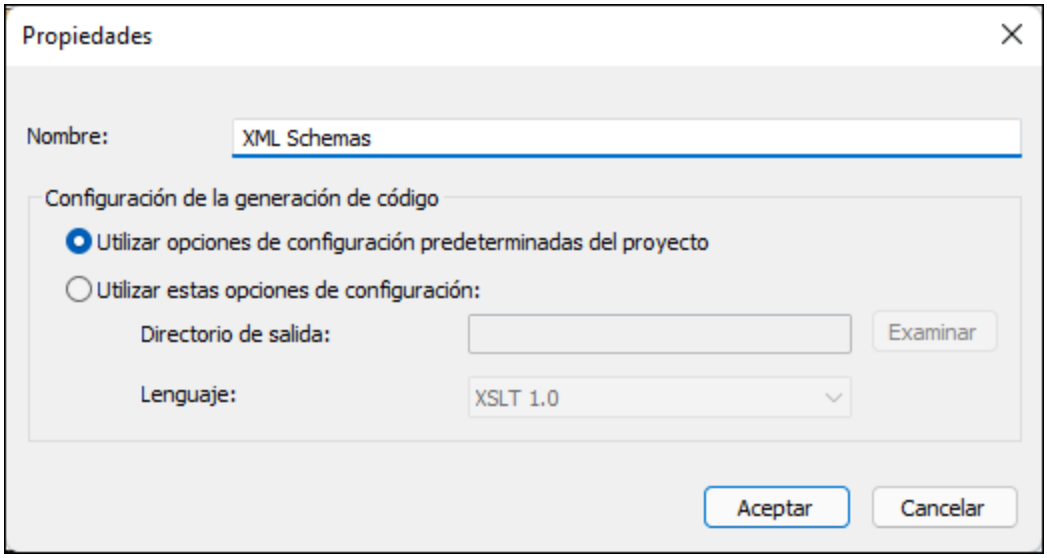

A continuación mostramos las opciones disponibles. (recuerde que el nombre y el directorio del proyecto no se pueden cambiar después de crear el proyecto):

- *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. Para más información sobre cómo generar código, consulte el apartado Generación de código 68.
- *Nombre del paquete base:* Esta opción sólo es relevante si seleccionó el lenguaje Java. Define el nombre del paquete base en el proyecto de Java generado.

## 2.5.3 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 100 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.

 © 2019-2025 Altova GmbH

# 3 Tutoriales

**Sitio web de Altova:** Vídeos de demostración de MapForce

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 18. 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 104 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 114 aprenderá a combinar los datos de distintos archivos XML de origen con los de un único archivo de destino.
*Asignaciones encadenadas*
En este tutorial 119 crearemos una asignación simple como la del segundo 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 128 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.

# 3.1 De esquema a esquema

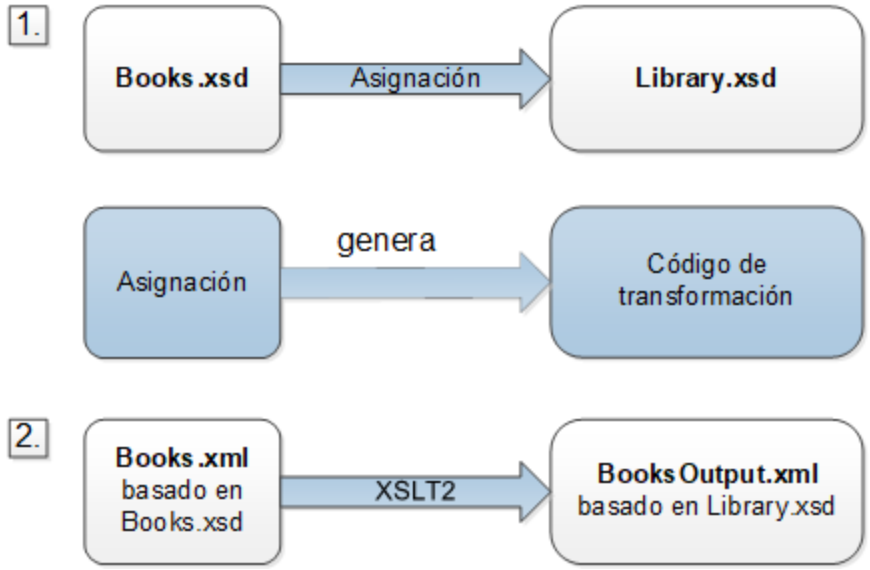
En este tutorial se explica cómo crear una asignación básica. El objetivo es transferir datos de un archivo XML A que tenga asignado un esquema XML A y pasarlos a un archivo XML B que tenga asignado un esquema B. Estos son los pasos que seguiremos a grandes rasgos:

1. Como estamos usando dos estructuras de datos, vamos a crear dos componentes (*origen* y *destino*) en el diseño de la asignación.
2. La transformación de un documento en otro se lleva a cabo usando un lenguaje de transformación 19 que tendremos que elegir.
3. Después tendremos que conectar los nodos de origen con los nodos de destino deseados. Estas conexiones son las que constituyen la asignación y determinan a qué nodo de destino está asignado cada nodo de origen.
4. Como resultado de la asignación obtenemos el documentos XML de destino, que es válido conforme al esquema de destino.
5. Por último, guardamos el archivo XML de salida.

Para más información sobre transformaciones de datos consulte el modelo abstracto que ve más abajo.

## Modelo abstracto

El modelo abstracto que ve a continuación ilustra una transformación de datos del tutorial:




La asignación tiene un origen y un destino. El esquema de origen (**`Books.xsd`**) describe la estructura del archivo de instancia de origen (**`Books.xml`**). El esquema de destino (**`Library.xsd`**) describe la estructura del archivo de instancia de destino (**`BooksOutput.xml`**). Al conectar los nodos del componente de origen a los del componente de destino la asignación genera código de transformación en XSLT 3.0. El código de transformación lee código de **`Books.xml`** y escribe esos datos en **`BooksOutput.xml`**.

 © 2019-2025 Altova GmbH

## Archivos de origen y de destino

El fragmento de código siguiente contiene algunos de los datos de muestra de **`Books.xml`** que se usan como datos de origen.

```
<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>
```

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.

## 3.1.1 Crear y guardar diseños

En este apartado explicamos cómo crear un diseño nuevo, seleccionar un lenguaje de transformación, así como 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 19. En este tutorial hemos seleccionado XSLT3. Puede seleccionar este lenguaje de transformación de dos maneras diferentes:

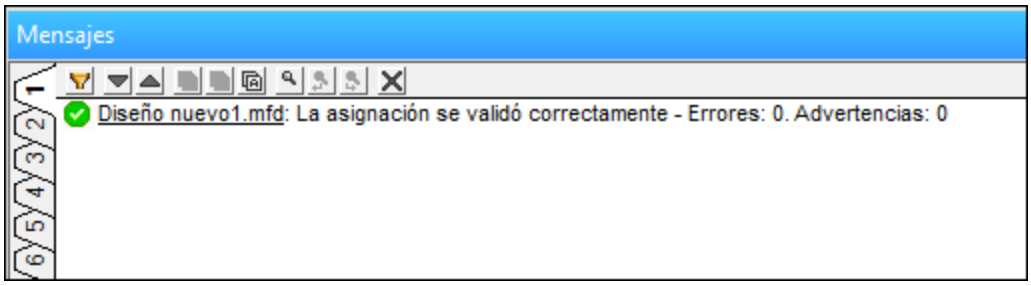
- En la barra de herramientas, haga clic en **XSLT3**.
- Abra el menú **Resultados** y haga clic en **XSLT 3.0**.

### 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 .

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 .

 © 2019-2025 Altova GmbH

Hemos guardado la asignación creada en este tutorial como **`Tut1_OneToOne.mfd`**.

## 3.1.2 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.

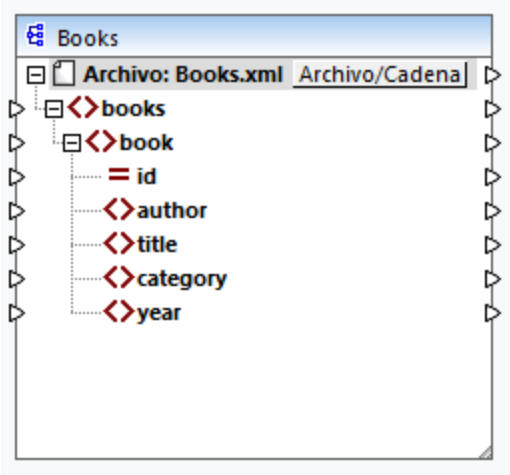
Si selecciona una de las dos primeras opciones, el cuadro de diálogo **Insertar un archivo de esquema XML** le sugerirá que elija entre un paquete de esquema estándar y un archivo remoto o local. En nuestros tutoriales usamos únicamente archivos locales. Para más información sobre cómo añadir archivos XML, consulte el apartado XML y esquemas XML 138.

Si se crea un componente a partir de un archivo XSD, la aplicación le pedirá que seleccione un archivo XML para usarlo como archivo de datos del componente. Si se crea un componente a partir de un archivo XML, 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 sugerirá generar uno para ese componente.

Dado que primero añadimos el archivo de esquema, MapForce sugiere que añadimos también un archivo XML 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.

### 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 32. 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:




El nombre del encabezado (Books) 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 y otras características del componente haga doble clic en el encabezado del componente. Esto abre el cuadro de diálogo Configuración del componente 139.

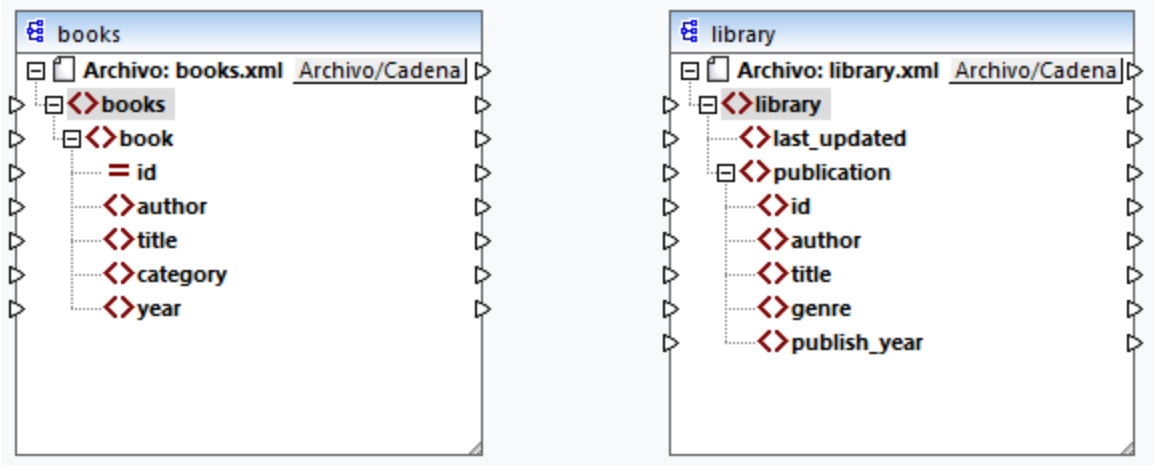
El nodo de nivel superior del componente (**`Archivo: Books.xml`**) representa el nombre del archivo de instancia XML. Los elementos XML de la estructura están representados por el icono . Los atributos XML están representados por el icono . Los triángulos pequeños que aparecen a ambos lados del componente 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*.

Para más información sobre componentes, conexiones, procedimientos y funciones generales, consulte el apartado Fundamentos de la asignación de datos 32.

## 3.1.3 Agregar un componente de destino

El paso siguiente es agregar un componente de destino y configurarlo. Agregue el archivo de destino **`Library.xsd`** a la asignación. Haga clic en **Omitir** cuando MapForce le pida que indique un archivo de instancia. En este punto la asignación tiene este aspecto:

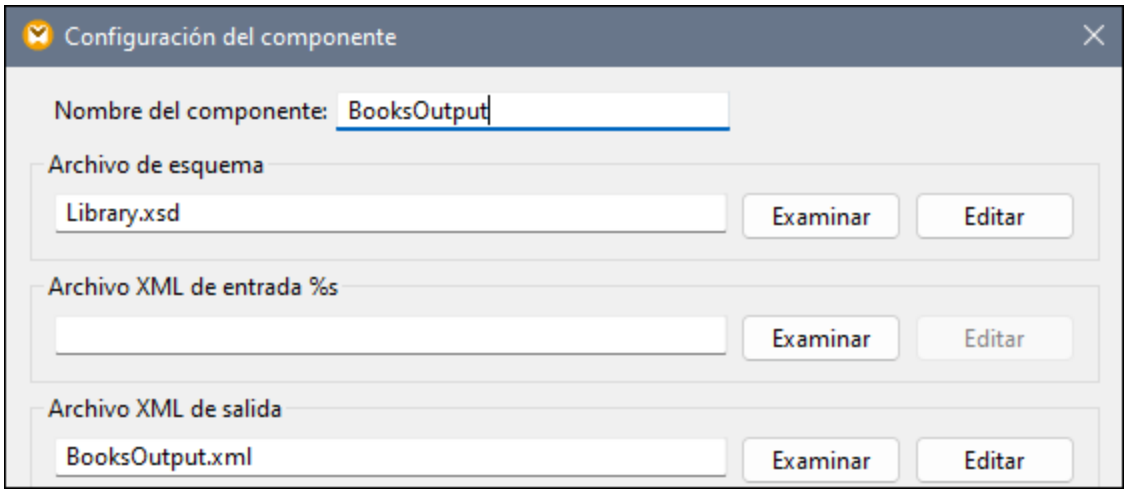
 © 2019-2025 Altova GmbH



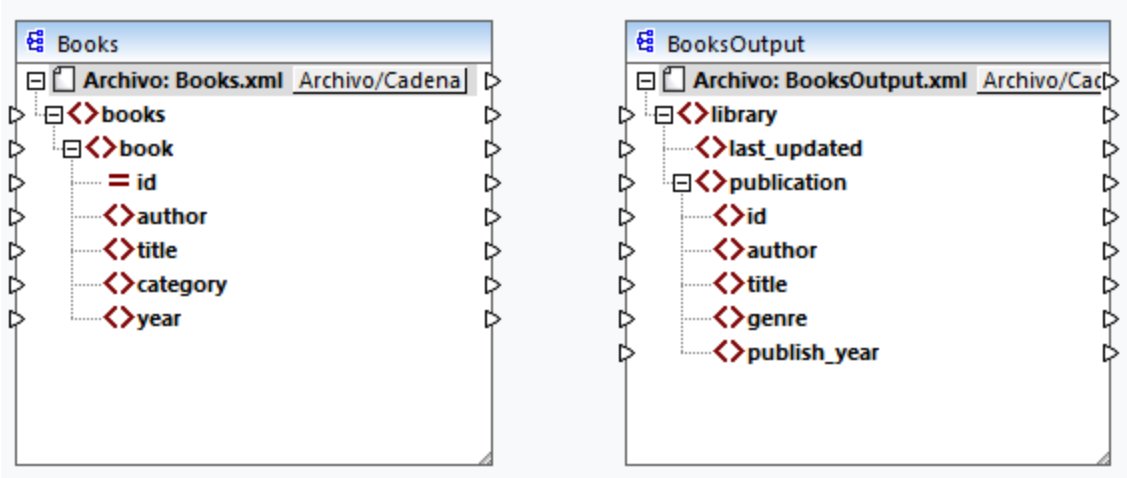

Como se puede ver, al abrir **`Library.xsd`**, este aparece como archivo XML en el componente. En realidad, lo que hace MapForce es crear una referencia al archivo XML llamada **`Library.xml`**, pero sin crear ningún archivo XML de verdad. Por tanto, el componente de destino ahora mismo tiene un esquema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

## Configurar componentes

Ahora cambiamos el nombre del componente de destino a **`BooksOutput.xml`**. Es decir que el archivo de salida se llamará **`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 139 (*ver imagen siguiente*), donde vamos a cambiar el nombre del archivo de destino:




La asignación diseñar ahora tiene este aspecto:

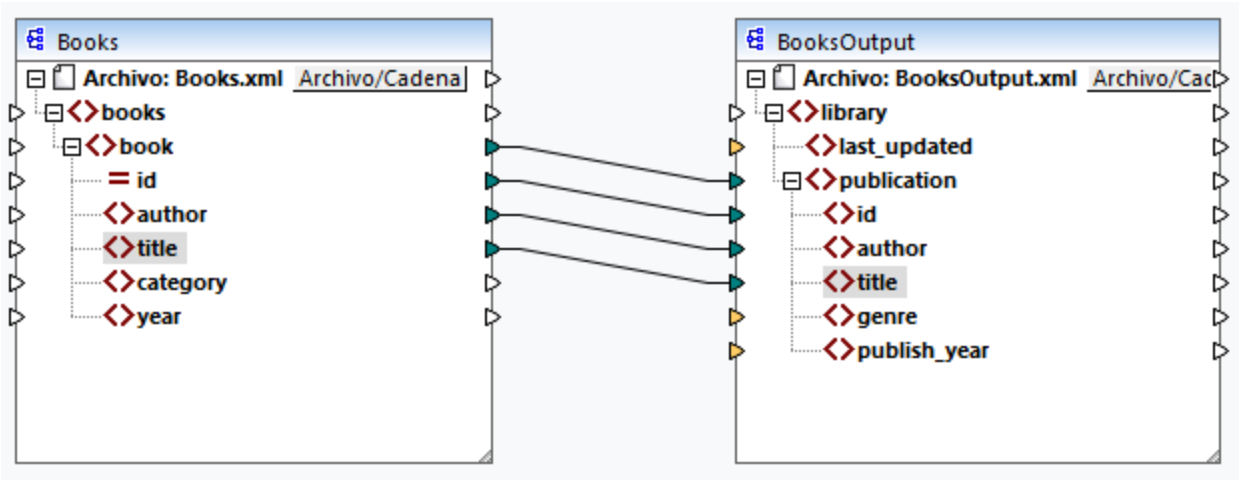



## 3.1.4 Conectar origen y destino

En este paso vamos a asignar los datos del archivo de origen al archivo de destino. También proporcionaremos información sobre la fecha y hora actuales con la función XPath `current-dateTime` 703.

### 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 pulse y mantenga en el conector de salida (el pequeño triángulo) a la derecha del elemento `<book>` y arrastre la línea hasta el conector de entrada del elemento `<publication>` en el componente de destino. Con esta acción MapForce puede conectar automáticamente todos los nodos secundarios de `<book>` del archivo de origen a los nodos 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* 55 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**.

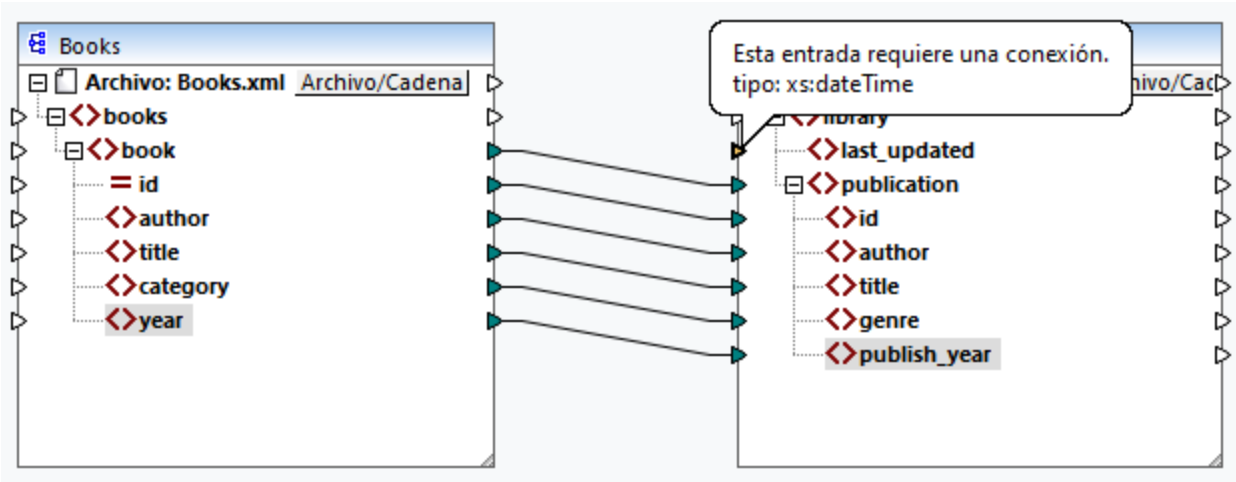
## Conectar elementos obligatorios

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:

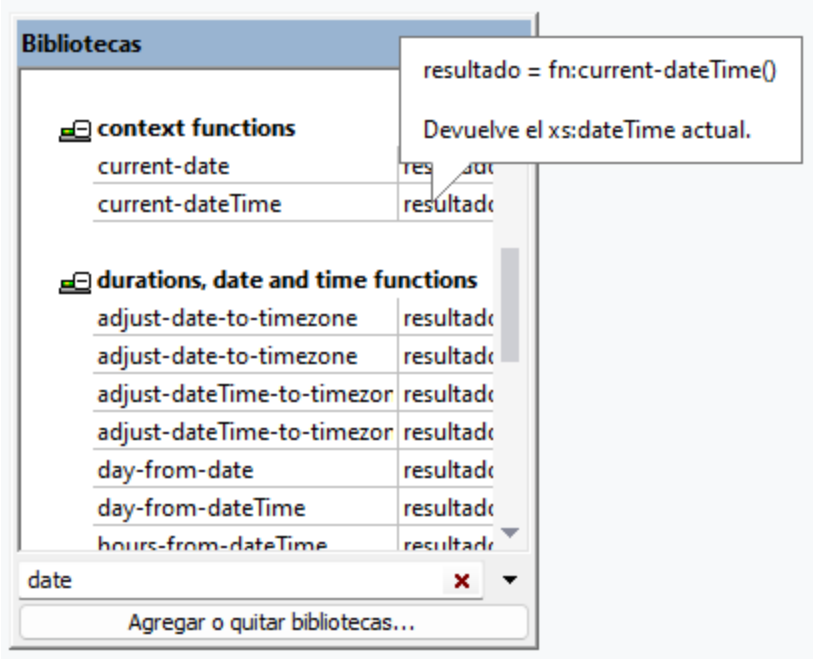
- Conecte el elemento `<category>` del archivo de origen con el elemento `<genre>` del componente de destino.
- Conecte el elemento `<year>` del archivo de origen con el elemento `<publish_year>` del componente de destino.

## Agregar fecha y hora actuales

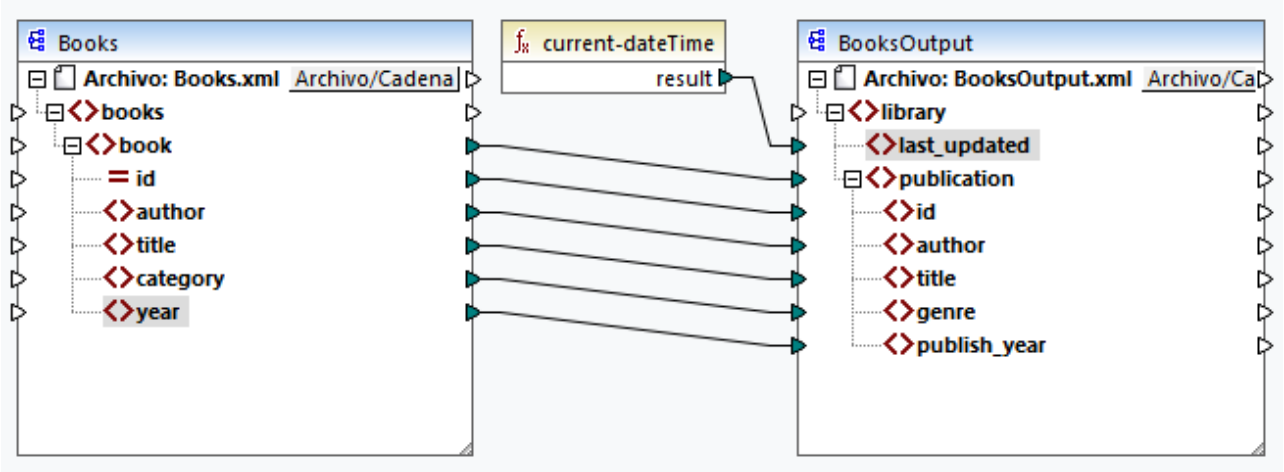
Por último debe indicar un valor para el elemento `<last_updated>`. Si pasa el cursor del ratón por encima de este conector de entrada puede ver que es un elemento de tipo `xs:dateTime` (*imagen siguiente*). Para poder ver esa información, pulse el botón de la barra de herramientas . Si pulsa el botón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




Para obtener la fecha y hora actuales puede usar la función **`current-dateTime`**. Para encontrarla empiece a teclearla en la caja de texto que hay en la parte inferior de la ventana Bibliotecas 23 (*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. Después, conecte su conector de salida con el conector de entrada del elemento `<last_updated>` (*ver imagen siguiente*).




Ahora puede validar y guardar la asignación, como explicamos en Crear y guardar diseños 106.

## 3.1.5 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 (*ver imagen siguiente*).

 © 2019-2025 Altova GmbH

```
1   <?xml version="1.0" encoding="UTF-8"?>
2   <library xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:n
    Documents/Altova/MapForce2024/MapForceExamples/Tutorial/BasicTutoria
3       <last_updated>2023-08-23T15:50:38+02:00</last_updated>
4       <publication>
5           <id>1</id>
6           <author>Mark Twain</author>
7           <title>The Adventures of Tom Sawyer</title>
8           <genre>Fiction</genre>
9           <publish_year>1876</publish_year>
10      </publication>
11      <publication>
12          <id>2</id>
13          <author>Franz Kafka</author>
14          <title>The Metamorphosis</title>
15          <genre>Fiction</genre>
16          <publish_year>1912</publish_year>
17      </publication>
```

Asignación | XSLT3 | Consulta de la BD | **Resultados**

Tut1-OneToOne.mfd*

*Guardar resultados*

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.

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.

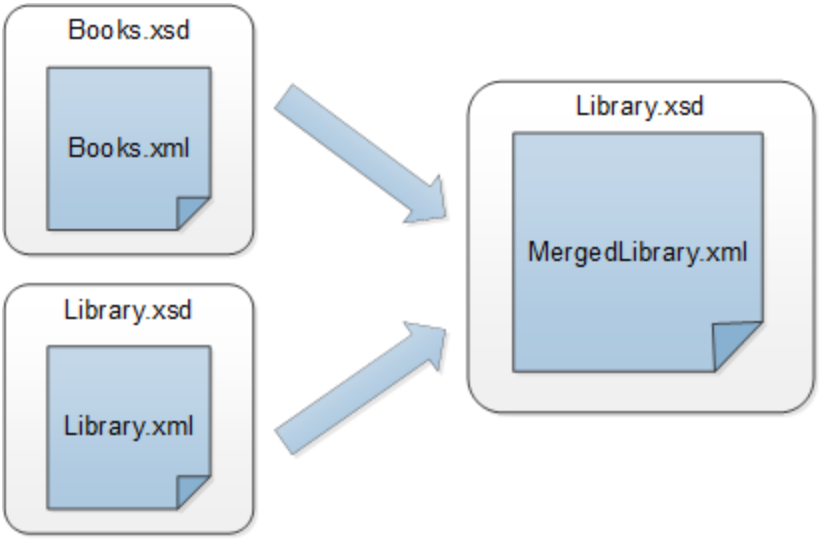
*Vista previa del código generado*

También puede acceder a la vista previa del código XSLT generado que lleva a cabo la transformación. Para previsualizar el código, abra el panel XSLT3 que hay en la parte inferior de la ventana Asignación. Para generar el código XSLT3 y guardarlo en un archivo seleccione el elemento de menú **Archivo | Generar código en | XSLT 3.0**. 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.

## 3.2 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 128. 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** que usaremos 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** 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>
```

 © 2019-2025 Altova GmbH

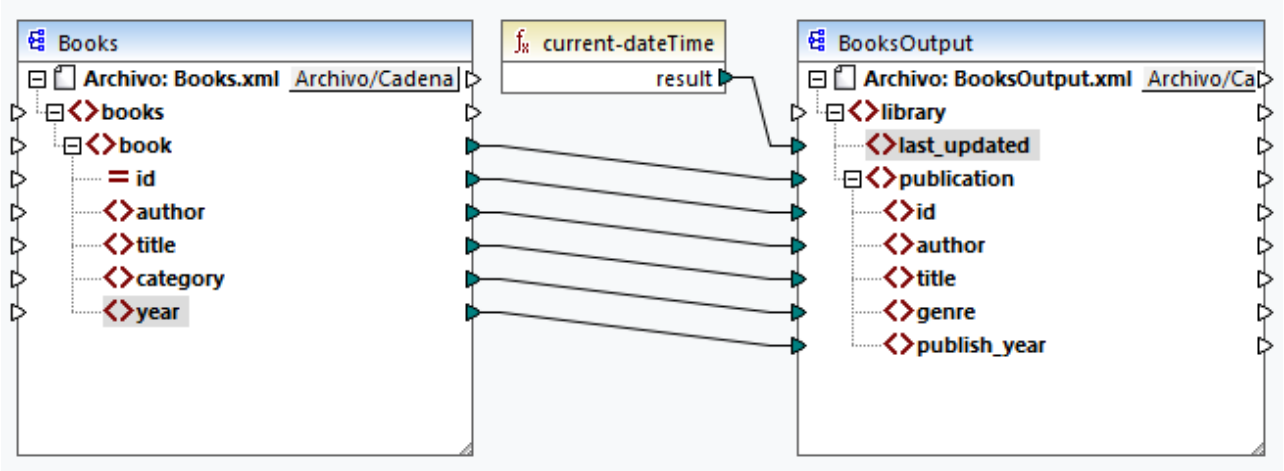
Este es el aspecto que queremos que tengan los datos combinados en el archivo de destino **MergedLibrary.xml**:

```
<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.

## 3.2.1 Preparar el diseño de la asignación

El punto de partida de este tutorial es la asignación **Tut1_OneToOne.mfd** (*ver imagen siguiente*) que hemos diseñado en el primer tutorial 104. Antes de realizar cualquier cambio en la asignación, asegúrese de guardar este diseño con un nuevo nombre en la carpeta **Tutoriales básicos**.

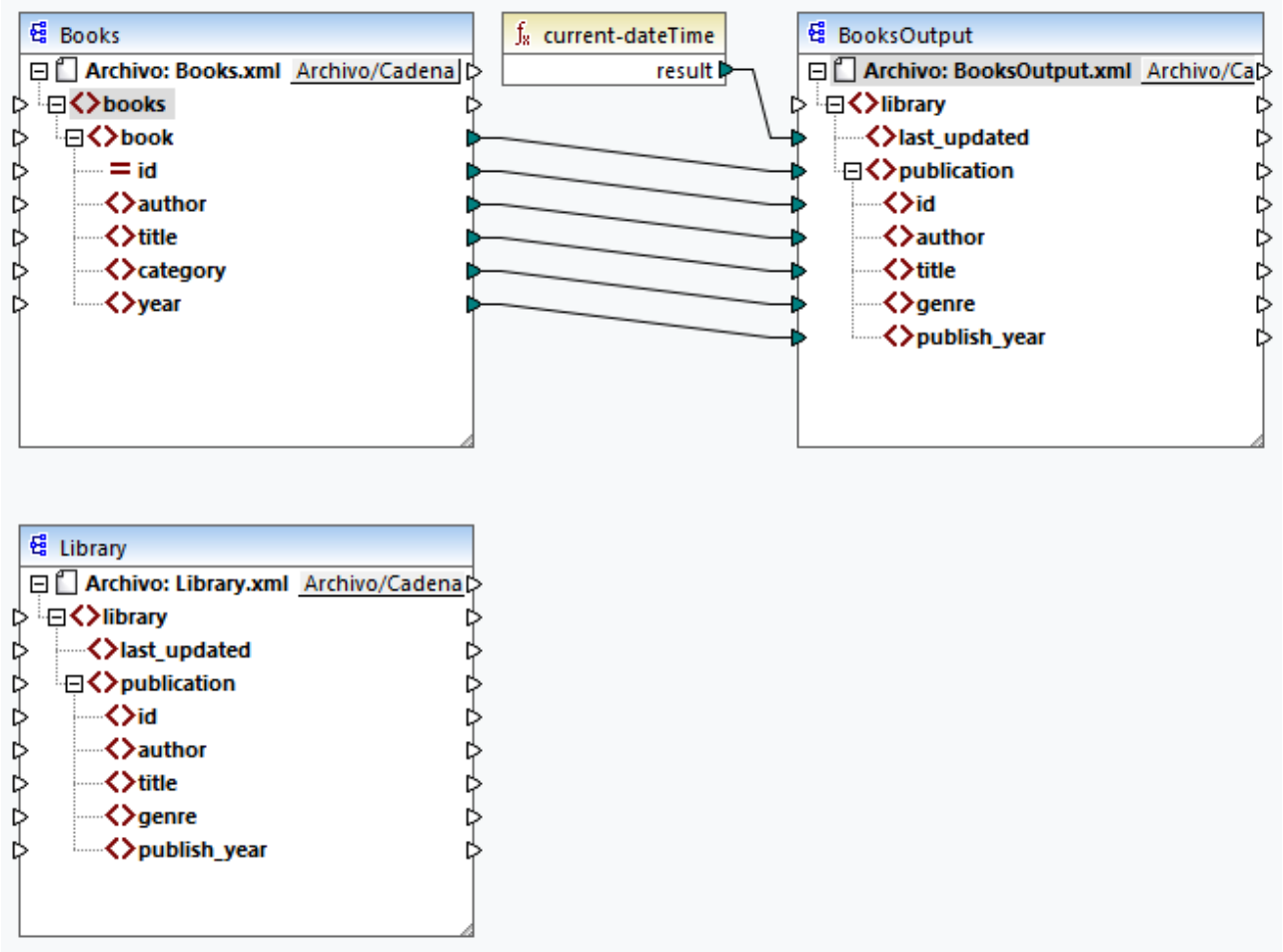


## 3.2.2 Agregar segundo archivo de origen

El paso siguiente consiste en agregar el segundo archivo de origen: Insertar el archivo **`Library.xml`** en la asignación. Ya que se hace referencia el esquema (**`Library.xsd`**) en este archivo XML, no es necesario añadir el archivo del esquema en otro paso separado. Para comprobar si la referencia al esquema es correcta, abra la configuración de los componentes 139.

Ahora pulse y mantenga pulsado el encabezado del componente nuevo y póngalo bajo el componente `Books`. 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.

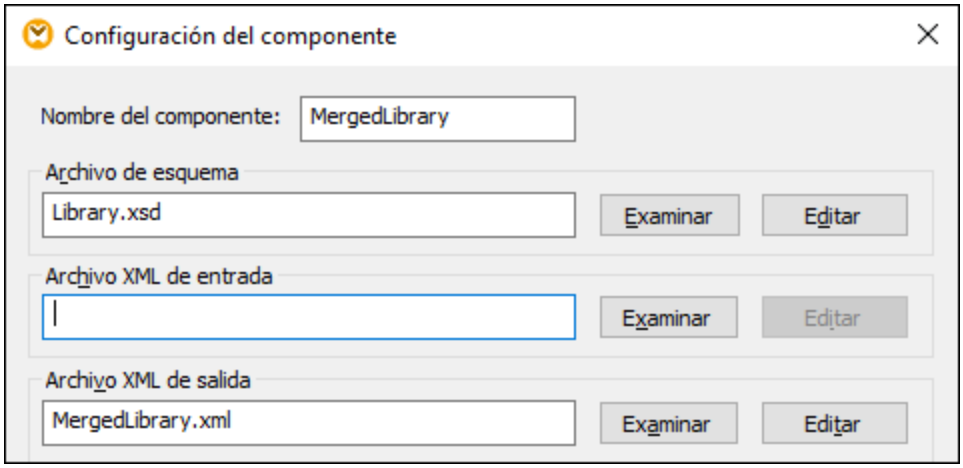
En este punto la asignación tiene este aspecto:




 © 2019-2025 Altova GmbH

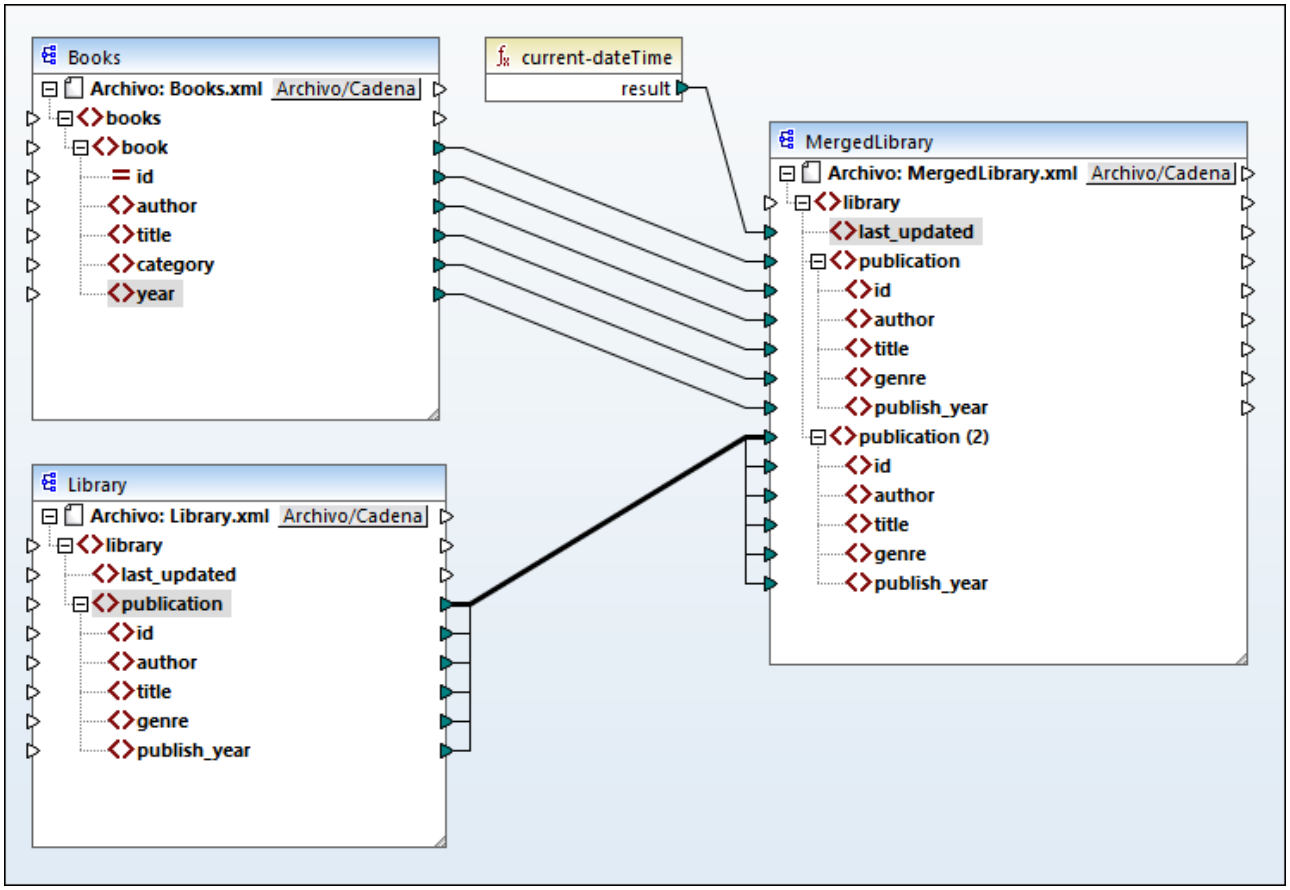
## 3.2.3 Configurar componentes de destino

En este punto la asignación tiene dos componentes de origen (`Books` y `Library`) y un componente de destino (`BooksOutput`). Para asegurar la coherencia y evitar malentendidos, tendremos que cambiar la configuración del componente `BookOutput`. Haga doble clic en el encabezado del componente. Esto abre el cuadro de diálogo Configuración del componente 139. Configura las opciones tal y como se explica a continuación.




## 3.2.4 Conectar varios orígenes a un destino

El último paso consiste en conectar el segundo componente de origen (`Library`) con el componente de destino (`MergedLibrary`). Para ello, conecte el elemento `<publication>` en **`Library.xml`** con el elemento `<publication>` en **`MergedLibrary.xml`**. Puesto que el conector de entrada de destino ya tiene una conexión, MapForce le pedirá que reemplace o duplique la entrada. 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 segundo componente de origen. La asignación ahora tiene este aspecto:




En la imagen anterior se observa que el elemento `publication` del componente de destino se ha duplicado. El nodo nuevo `publication(2)` aceptará los datos de **`Library.xml`**. 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.

*Conexión de copia total*

Los elementos secundarios de ambos elementos `publication` en los dos componentes, es decir, tanto en `Library` como en `MergedLibrary`, tienen los mismos nombres y tipos de datos. Por tanto, estos elementos están conectados con una línea gruesa. Este tipo de conexión se llama *conexión de copia total* 57 y facilita la comprensión de la asignación.

*Vista previa del resultado*

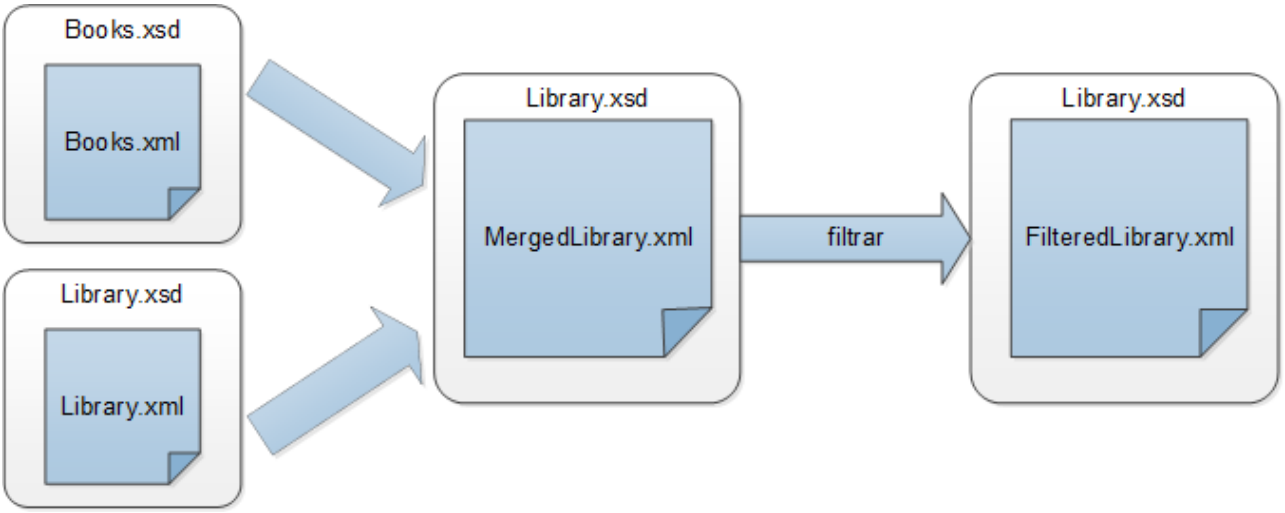
Abra la pestaña Resultados para ver el resultado. Observará que los datos de **`Books.xml`** y de **`Library.xml`** ahora se han combinado en el archivo nuevo **`MergedLibrary.xml`**. Hemos guardado el diseño de asignación de este tutorial como **`Tut2_MultipleToOne.mfd`**. Esta es la asignación que usaremos como punto de partida en el tutorial siguiente 119.

 © 2019-2025 Altova GmbH

## 3.3 Asignación en cadena

**Sitio web de Altova:** Vídeo de tutorial sobre las asignaciones encadenadas

En este tutorial aprenderá a trabajar con varios componentes de destino. El objetivo de este tutorial es combinar los datos de dos archivos de origen en un único archivo de destino para después filtrar los datos resultantes y pasarlos a un segundo archivo de destino. En la imagen siguiente puede ver un modelo abstracto de la transformación de datos de este tutorial.



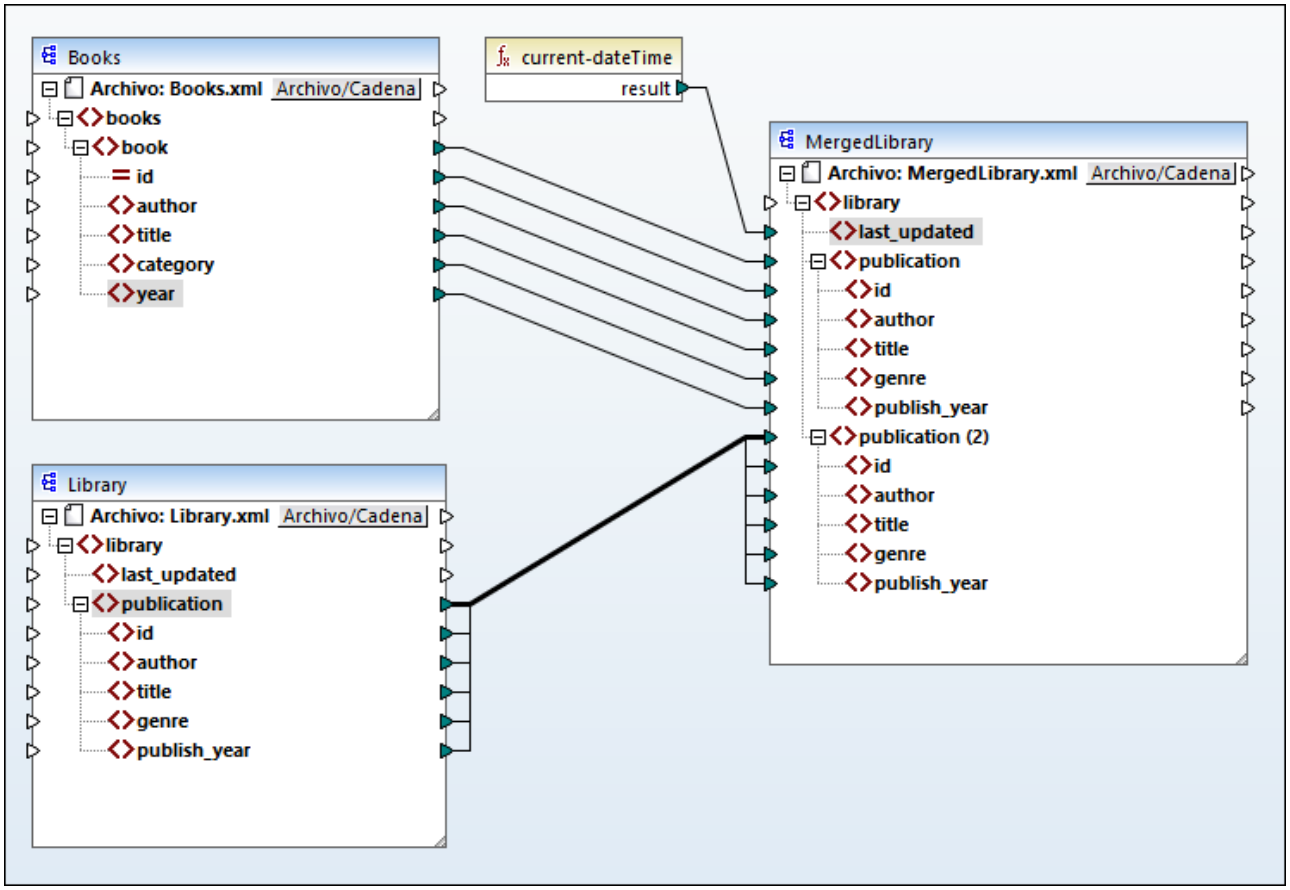

En este diagrama puede ver que primero se combinan los datos de los dos archivos de origen (**`Books.xml`** y **`Library.xml`**) en un único archivo de destino llamado **`MergedLibrary.xml`**. Después los datos se transforman con una función de filtro y se pasan al siguiente componente, que es **`FilteredLibrary.xml`**. Recuerde que **`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*. En las asignaciones encadenadas, puede previsualizar y guardar uno o varios resultados intermedios de la asignación (en nuestro caso es **`MergedLibrary.xml`**) y el resultado del último componente de destino (en nuestro caso es **`FilteredLibrary.xml`**).

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

Para ver otro ejemplo de una asignación encadenada, véase el tutorial en vídeo anexado en la parte superior de la página.

### 3.3.1 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[114]. Antes de realizar cualquier cambio en la asignación, asegúrese de guardar este diseño con un nuevo nombre a la carpeta **`Tutoriales básicos`**.



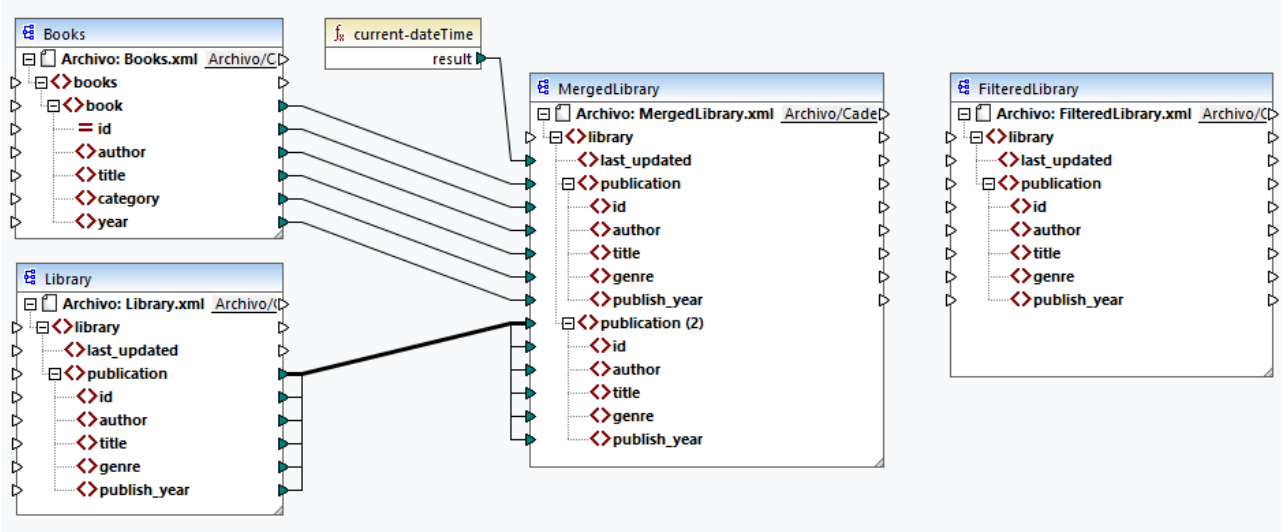

## 3.3.2 Configurar el segundo archivo de destino

Ahora tenemos que añadir y configurar el segundo archivo de destino que contiene solamente un subconjunto de los datos de `publicación` provenientes de **`MergedLibrary.xml`**.

### Agregar el segundo componente de destino

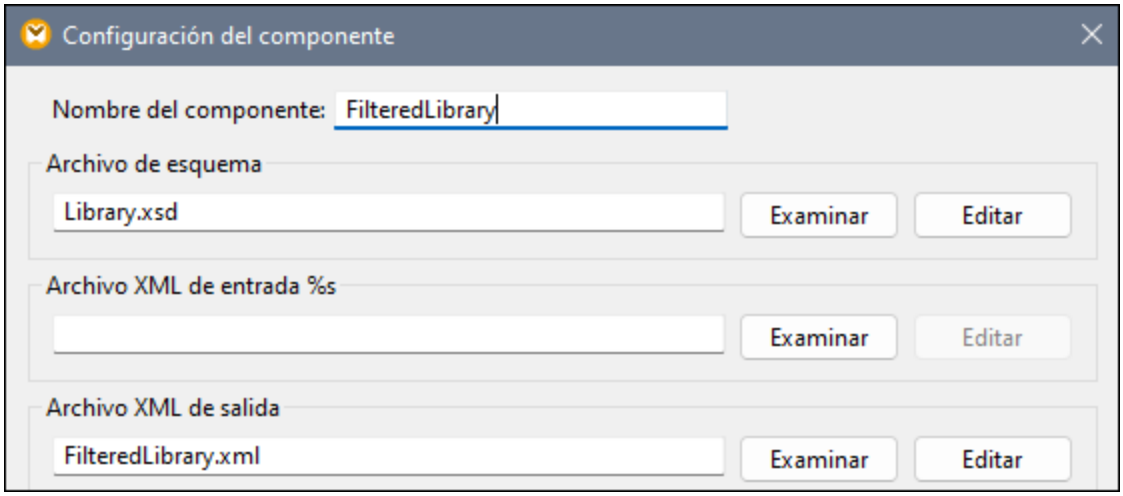
Agregue el archivo **`Library.xsd`** a la asignación y haga clic en **Omitir** cuando MapForce le pida que indique un archivo de instancia de ejemplo. El segundo componente de destino tiene estructura pero no contenido. Más adelante asignaremos los datos filtrados a este archivo de destino. El diseño de la asignación ahora tiene este aspecto:

 © 2019-2025 Altova GmbH




## Configurar el segundo componente de destino

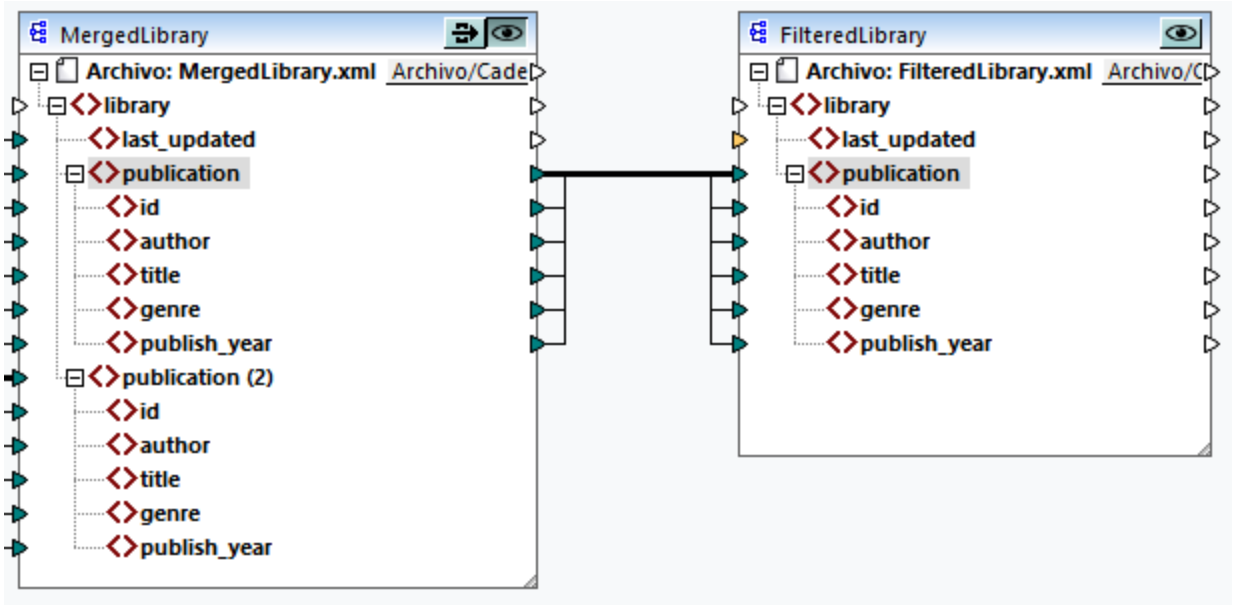
Como se ve más arriba, la asignación ahora tiene dos componente de origen (`Books` and `Library`) y dos componente de destino (`MergedLibrary` y `Library`). Para evitar malentendidos vamos a cambiar el nombre del componente que acabamos de añadir, `FilteredLibrary`. Para ello haga doble clic en el encabezado del componente que hay más a la derecha y edite su configuración 139 como sigue:




## 3.3.3 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`,MapForce le informará de que ha creado varios componentes de destino en la asignación. 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.



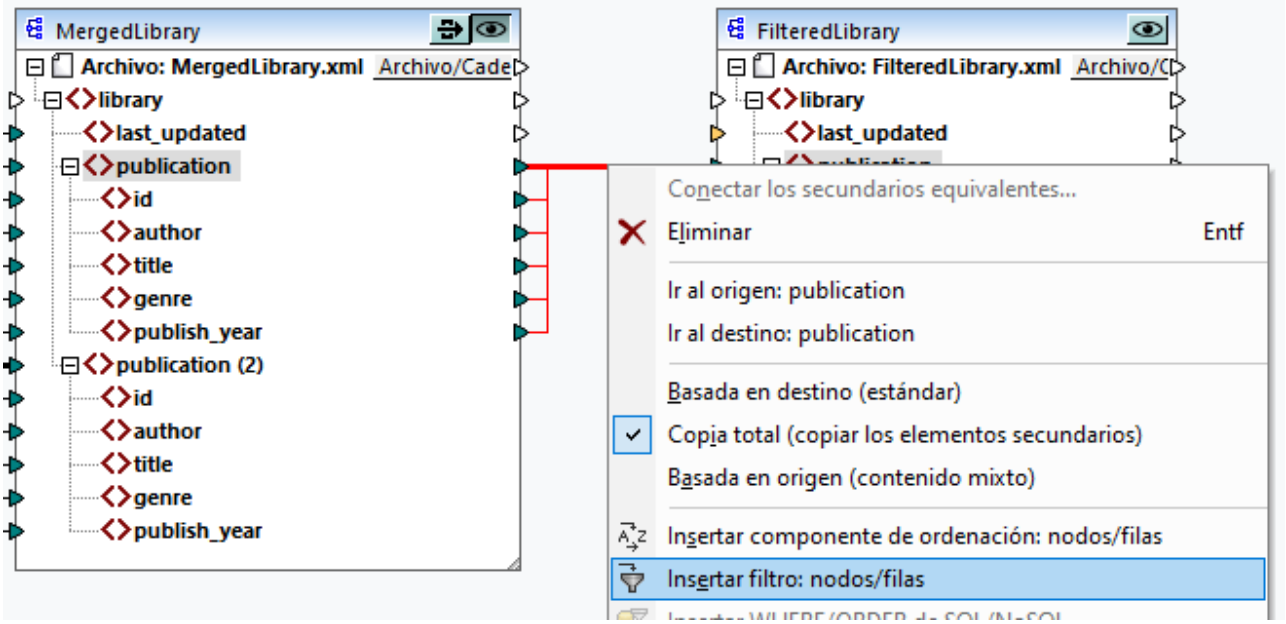

## 3.3.4 Filtrar datos

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.

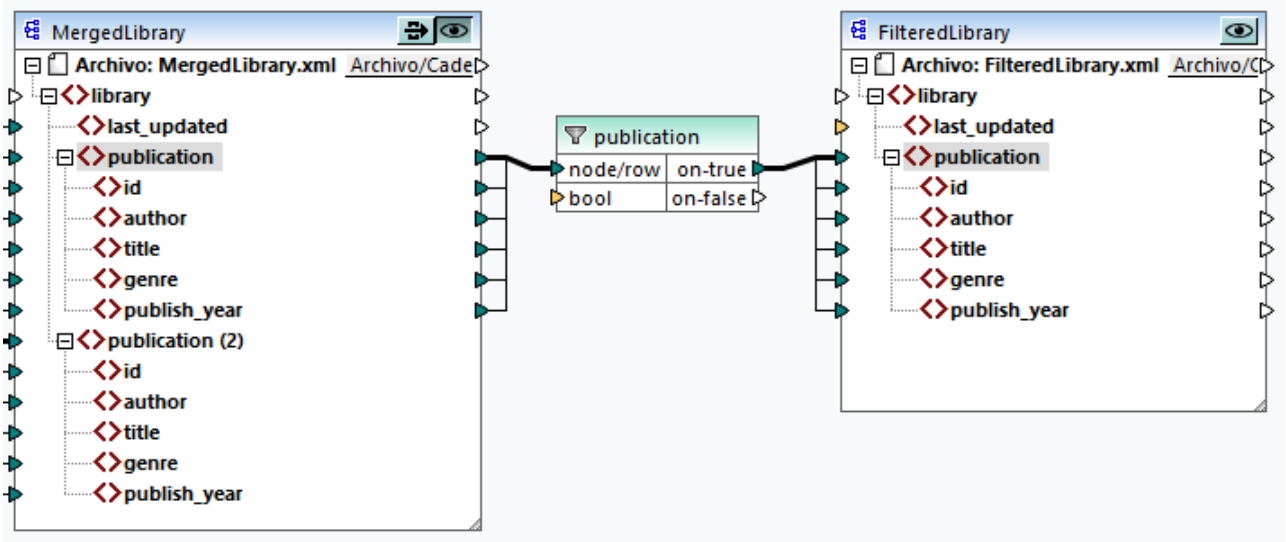
### Agregar un filtro

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 (*ver imagen siguiente*).

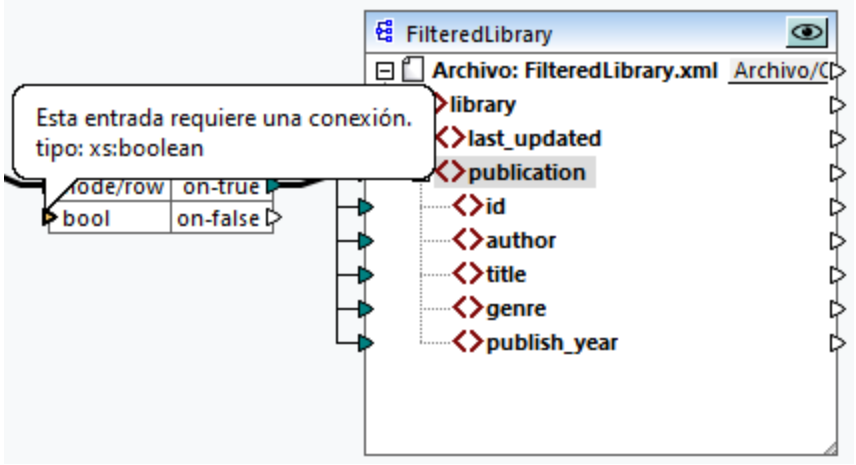
 © 2019-2025 Altova GmbH




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



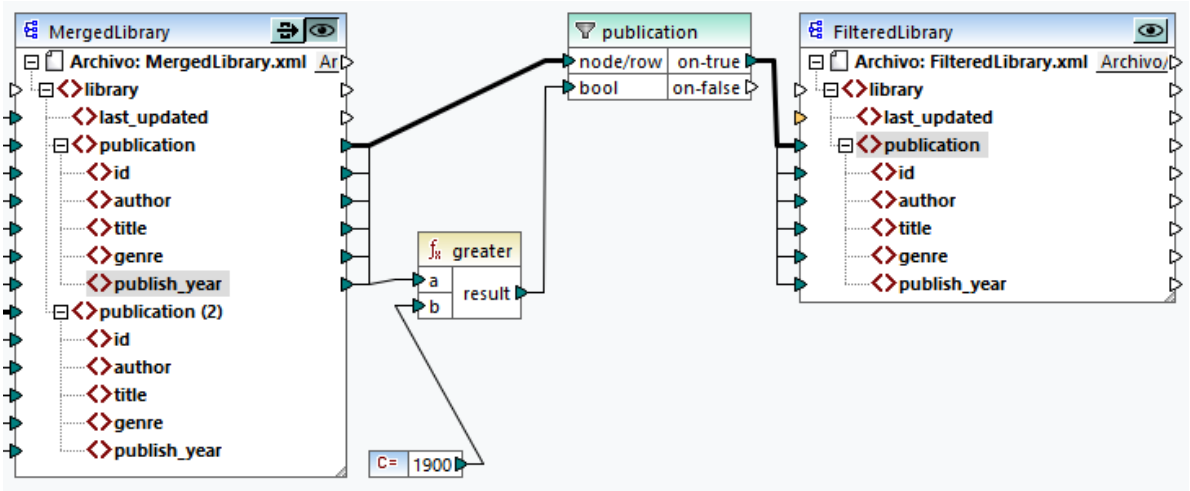

En la imagen anterior, el conector de entrada `bool` aparece en naranja, lo que significa que esta entrada es obligatoria. Si pasa el cursor del ratón por encima del conector verá que es necesario introducir un elemento de entrada de tipo `xs:boolean` (*ver imagen siguiente*). Para ver la información rápida habilítela primero haciendo clic en en la barra de herramientas (**Mostrar información rápida**).




## Sólo 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 asigna únicamente los libros que se publicaron a partir de 1900. Siga estos pasos para crear la condición:

1. Pulse **Constante** en la barra de herramientas y escriba *1900* en la barra de texto. Seleccione *Número* como tipo.
2. Agregue la función **`greater`** a la asignación.
3. Trace las conexiones de asignación a y desde la función **`greater`** como se ve en la imagen siguiente. La función **`greater`** comparará el valor del elemento `publish_year` de cada publicación con el valor de la constante. Sólo se asignarán al destino los registros de publicación cuyo año de publicación sea superior a 1900.




 © 2019-2025 Altova GmbH

## 3.3.5 Previsualizar 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, cada uno de estos componentes de destino tiene un botón (**Vista previa**). La vista previa no se puede habilitar para más de un componente al mismo tiempo. Los botones **Vista previa** permiten ver el resultado intermedio de la asignación (en nuestro ejemplo, los datos asignados al componente `MergedLibrary`), así como el resultado final de la asignación encadenada (los datos asignados al componente `FilteredLibrary`). El componente `MergedLibrary` también tiene un botón (**Paso a través**). El botón **Paso a través** controla cómo se generará el resultado (*ver detalles más abajo*).

*Previsualizar y guardar resultados intermedios y finales*
Para visualizar y guardar los resultados de los componentes `MergedLibrary` y `FilteredLibrary`, siga estos pasos:

1. Haga clic en el botón **Paso a través** del componente `MergedLibrary`.
2. Asegúrese de que el botón **Vista previa** del componente `FilteredLibrary` también está activado.
3. Abra la pestaña Resultados. Los botones **Atrás** y **Adelante** le permiten pasar de un resultado a otro.
4. Cambie al resultado que desea guardar en un archivo y haga clic en **Guardar archivo de salida** en la barra de herramientas. Si desea guardar ambos resultados, haga clic en el botón **Guardar todos los resultados generados** de la barra de herramientas.

Cuando la función del paso a través está activo, el campo *Archivo XML de entrada* del componente intermedio se desactiva automáticamente. Esto se debe a que los resultados generados al previsualizar la primera parte de la asignación entre los orígenes (`Books` y `Library`) y el primer destino (`MergedLibrary`) se utilizan por defecto como entrada cuando previsualiza la segunda parte de la asignación entre el primer destino (`MergedLibrary`) y el segundo destino (`FilteredLibrary`).

*Previsualizar y guardar resultados intermedios*
Los componentes intermedios no pueden previsualizarse cuando su botón de paso a través está activo. El botón de vista previa del componente intermedio se desactiva automáticamente porque no tiene sentido previsualizar y dejar pasar datos al mismo tiempo. Para visualizar y guardar sólo los resultados del componente intermedio (`MergedLibrary`) siga estos pasos:

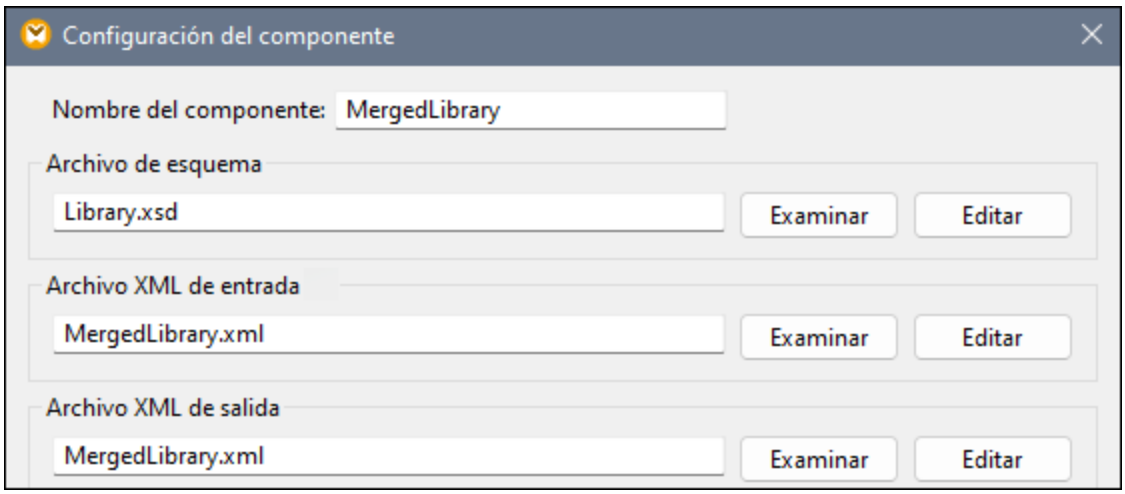
1. Desactive el botón **Paso a través** del componente `MergedLibrary` si estaba activado anteriormente.
2. Haga clic en el botón **Vista previa** del componente intermedio `MergedLibrary`.
3. Abra la pestaña Resultados.
4. Haga clic en el botón **Guardar archivo de salida** de la barra de herramientas para guardar los resultados en un archivo.

*Previsualizar y guardar resultados finales*
Para visualizar y guardar sólo los datos asignados del componente intermedio al segundo componente de destino, siga estos pasos:

1. Desactive el botón **Paso a través** del componente `MergedLibrary` si estaba activado anteriormente.
2. Haga doble clic en el encabezado del componente `MergedLibrary`.
3. Asegúrese de introducir el mismo nombre de archivo en el campo *Archivo XML de entrada* que en el campo *Archivo XML de salida* (*imagen siguiente*). Al desactivar el botón **Paso a través** puede 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*. Lo más

razonable es que el componente intermedio reciba un archivo, lo procese y lo reenvíe a la siguiente asignación, en lugar de buscar un nombre de archivo distinto.

Es importante tener el mismo archivo de entrada y salida para el componente intermedio cuando se desactiva el botón **Paso a través** del componente intermedio. Esto asegura que los resultados generados al previsualizar la primera parte de la asignación entre los orígenes (`Books` y `Library`) y el primer destino (`MergedLibrary`) se utilizan como entrada cuando previsualiza la segunda parte de la asignación entre el primer destino (`MergedLibrary`) y el segundo destino (`FilteredLibrary`). Si ejecuta su asignación con MapForce Server (*ediciones Professional y Enterprise*) o con código generado, los mismos nombres de los archivos de entrada y salida del componente intermedio garantizan que no se rompa la cadena de asignación. Tenga en cuenta que si en el componente intermedio el archivo de entrada y de salida tienen nombres *distintos* (con el botón **Paso a través** inactivo), pueden producirse errores en MapForce, en el código generado o en la ejecución con MapForce Server.




4. Haga clic en el botón **Vista previa** del componente `FilteredLibrary`.
5. Abra la pestaña Resultados.
6. Haga clic en el botón **Guardar archivo de salida** de la barra de herramientas para guardar los resultados en un archivo.

**Importante**

- 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 (*ediciones Professional y Enterprise*) pueden ser componentes intermedios pero no disponen del botón Paso a través.
- Cuando MapForce ejecuta la asignación, la configuración de la opción *Escribir directamente en archivos de salida finales* (que se configura desde **Herramientas | Opciones | Generales**[1128]) determina si los archivos de salida 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 ejecuta con MapForce o por medio de código generado, se producirían archivos en cada etapa de la cadena de asignación.

 © 2019-2025 Altova GmbH

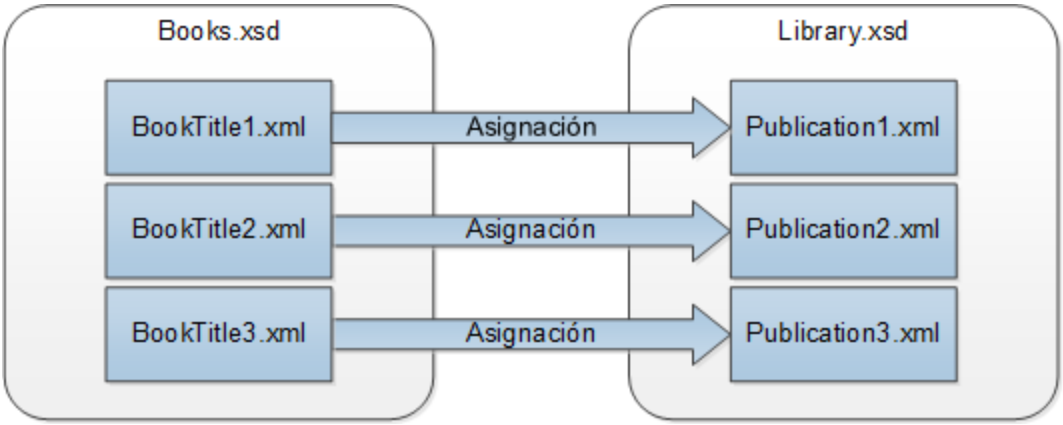
Con este paso terminamos de diseñar la asignación encadenada que produce dos archivos de salida. Hemos guardado el diseño de asignación de este tutorial como **`Tut3_ChainedMapping.mfd`**.

# 3.4 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:

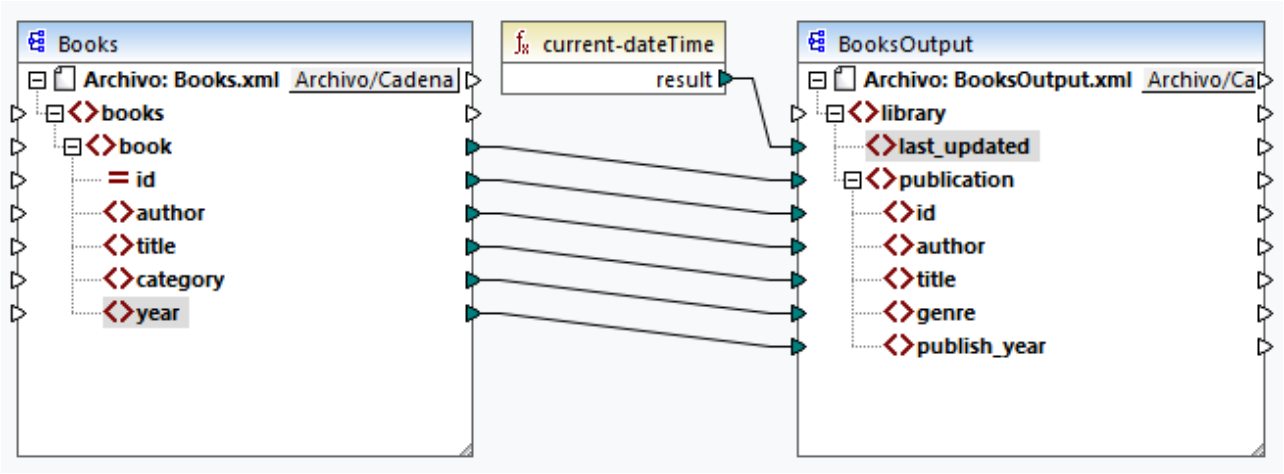
1. Leer datos de varios archivos XML ubicados en un mismo directorio. Estos archivos se basan en el mismo esquema.
2. Generar por cada archivo XML de origen un archivo XML de destino. Los archivos de destino se basarán en un nuevo esquema de destino.

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




## Resumen general

El punto de partida de este tutorial es la asignación **`Tut1_OneToOne.mfd`** del primer tutorial 104 (*ver imagen siguiente*).



*Modificar el componente de origen*
Modificaremos la configuración del componente de origen para que lea datos de distintos archivos de origen: **BookTitle1.xml**, **BookTitle2.xml** y **BookTitle3.xml**. Cada uno de los tres archivos se basa en **Books.xsd** y guarda un registro de libro (*ver a continuación*).

**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">
   <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>
```

*Modificar el componente de destino*
También modificaremos la configuración del componente de origen para que escriba datos a varios archivos de destino. Los archivos de destino se basan en el mismo esquema llamado **Library.xsd**. Los archivos de destino generados se llamarán **Publication1.xml**, **Publication2.xml** y **Publication3.xml** (*ver los extractos de código a continuación*).

**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>
```

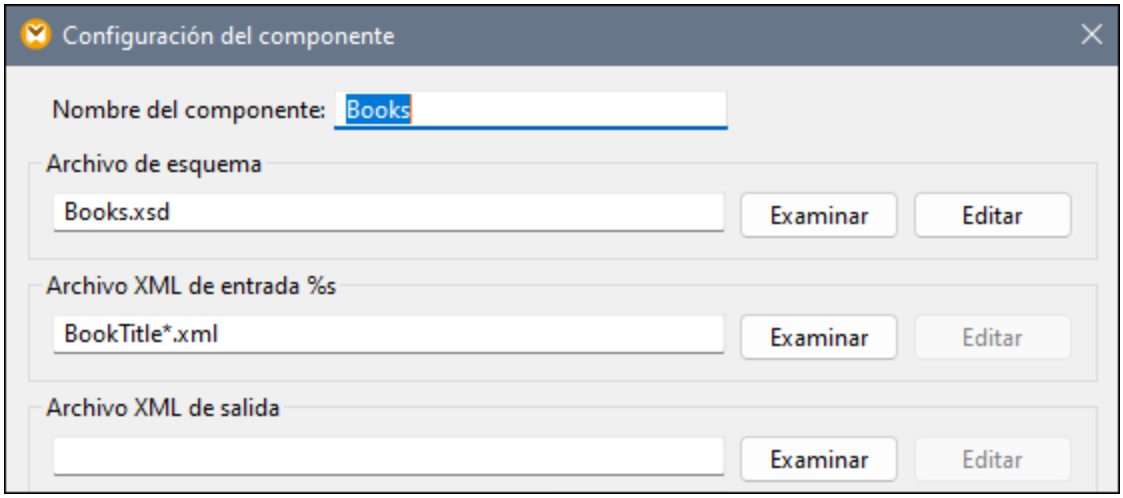
Para llevar a cabo la transformación de datos siga los pasos que describimos en los apartados siguientes.

### 3.4.1 Configurar el componente de entrada

El primer paso consiste en modificar los ajustes del componente de origen. Antes de cambiar la configuración de los componentes, asegúrese de guardar la asignación con un nuevo nombre en la carpeta **Tutoriales básicos**.

Para indicar a MapForce que debe procesar varios archivos de instancia XML, haga doble clic en el encabezado del componente e introduzca **BookTitle*.xml** en el campo *Archivo XML de entrada* (*ver imagen siguiente*). 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.

 © 2019-2025 Altova GmbH

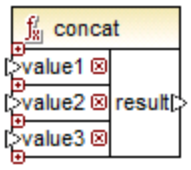



## 3.4.2 Configurar el componente de destino, parte 1

El paso siguiente es crear el nombre de archivo de los archivos de salida. Para ello vamos a usar la función **concat** 626 que combina todos los valores suministrados. 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.

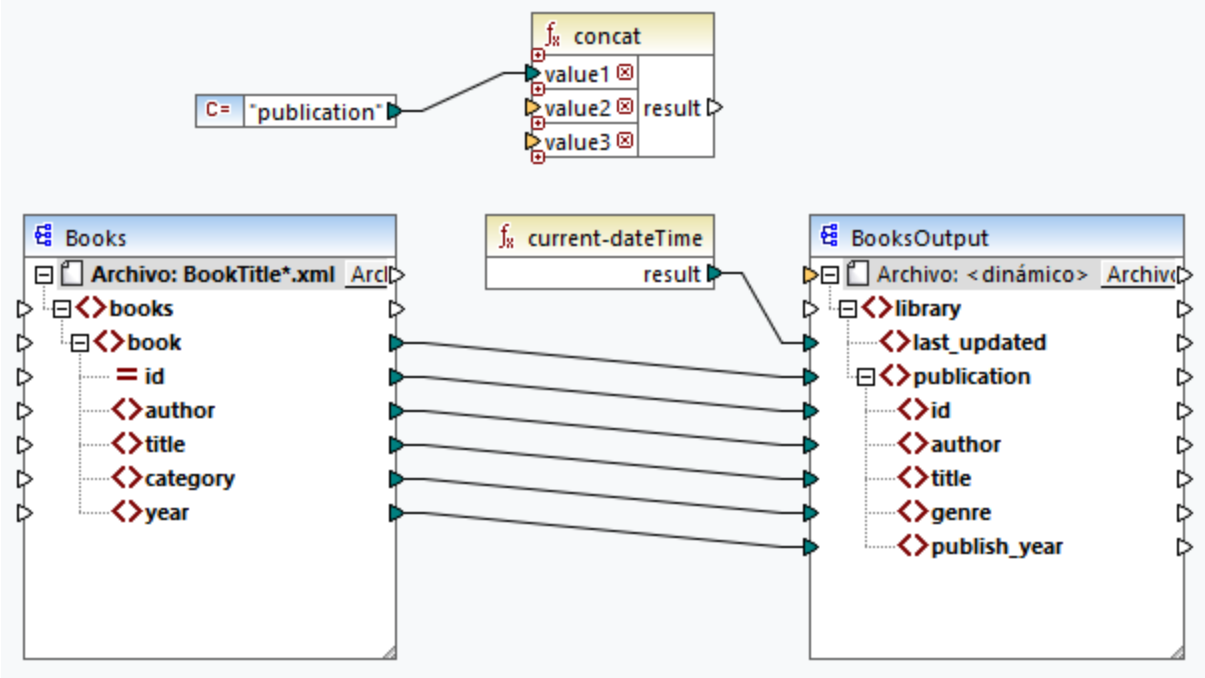
### Paso 1: Agregar la función concat

Agregue 110 la función **concat** (*ver imagen siguiente*) al área de asignación. 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**).




### Paso 2: Insertar una constante

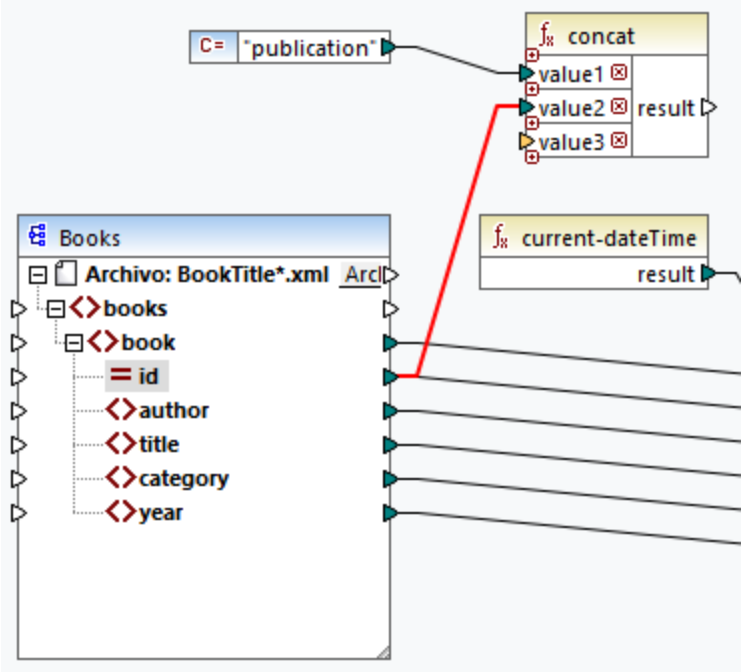
El siguiente paso consiste en agregar una constante. Cuando tenga que indicar un valor introduzca `publication` y deje la opción *String* como está. Conecte la constante con `value1` de la función **concat**, como se ve en la imagen siguiente:




## Paso 3: Suministrar Ids

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

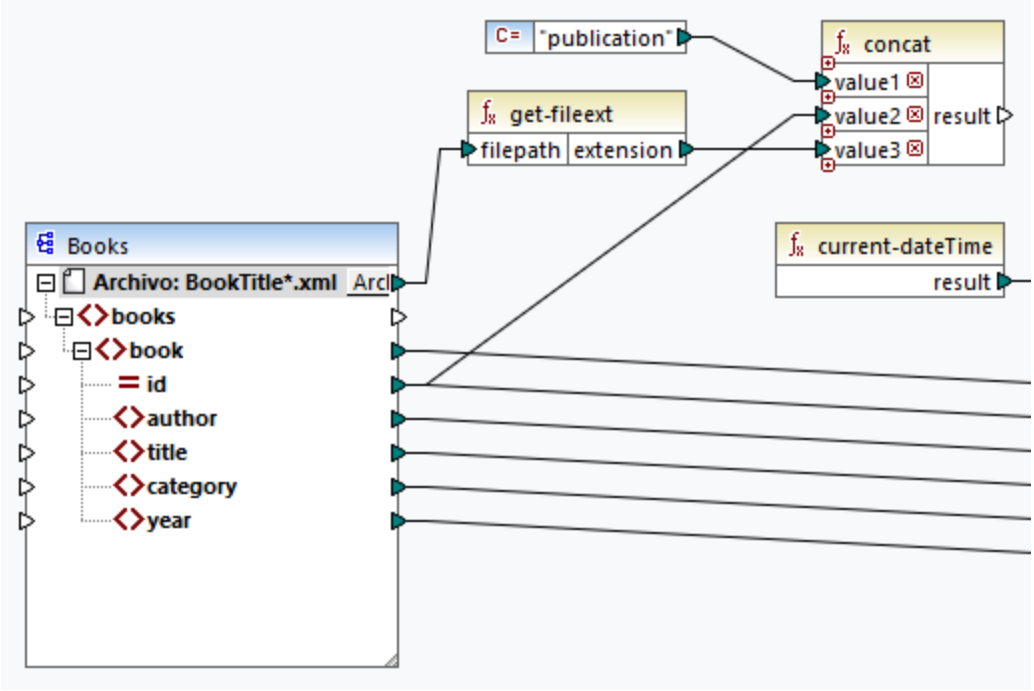
 © 2019-2025 Altova GmbH




## Paso 4: Extraer la extensión de archivo

Agregue la función **get-fileext** 570 al área de asignación. Después cree una conexión desde del nodo superior del componente de origen (**Archivo: BookTitle.xml**) hasta el parámetro `filepath` de esta función (*ver imagen siguiente*).

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.



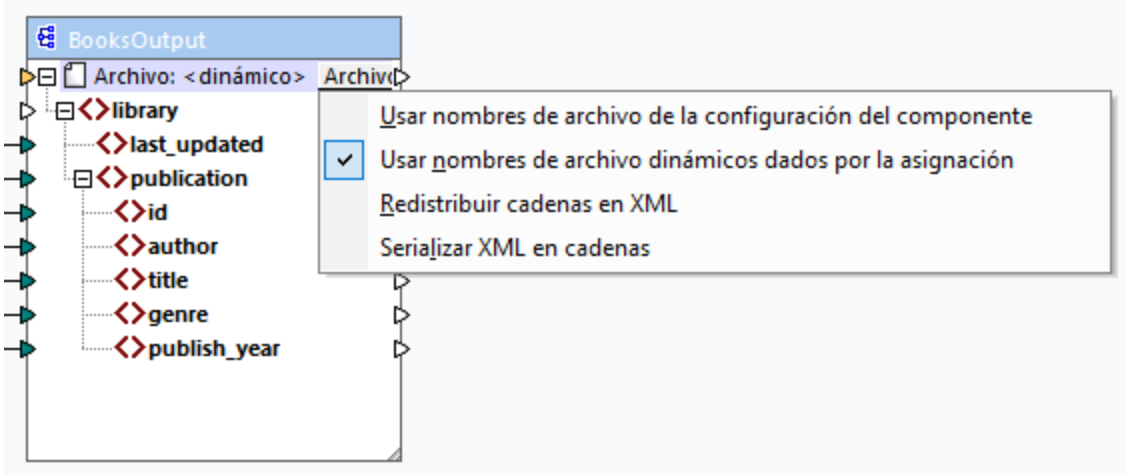

## 3.4.3 Configurar el componente de destino, parte 2

En la segunda parte de la configuración de salida, indicaremos a MapForce que genere archivos de salida 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 usaremos nombres de archivo dinámicos (*véase los puntos siguientes*). Para más información sobre nombres de archivo dinámicos consulte el apartado Procesar varios archivos de entrada y salida 784.

### Paso 1: Configurar nombres de archivo dinámicos

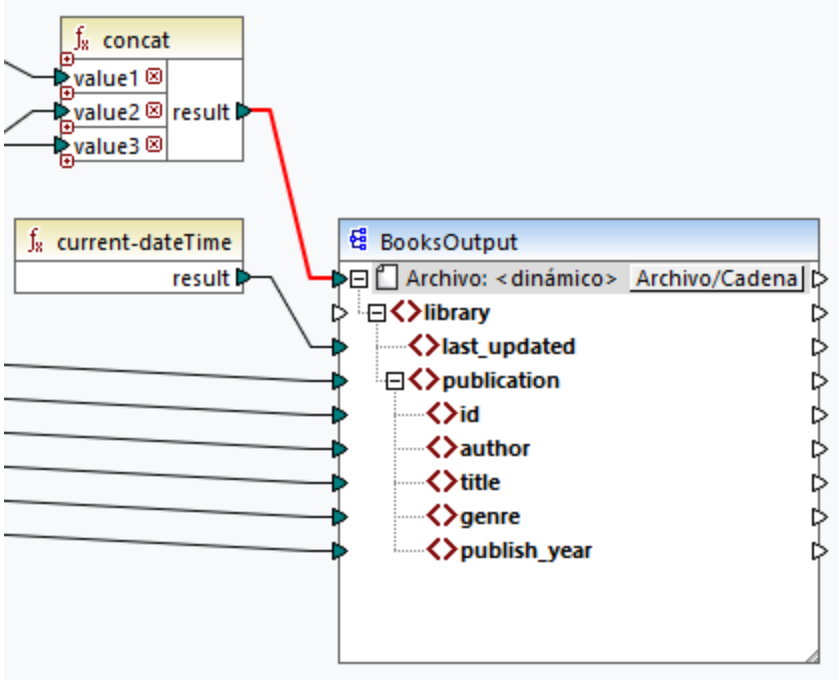
Para que MapForce genere dinámicamente archivos, haga clic en el botón Archivo o Archivo/Cadena junto con el nodo `Archivo` del componente de destino y seleccione **Usar nombres de archivo dinámicos dados por la asignación** del menú contextual (*imagen siguiente*).

 © 2019-2025 Altova GmbH



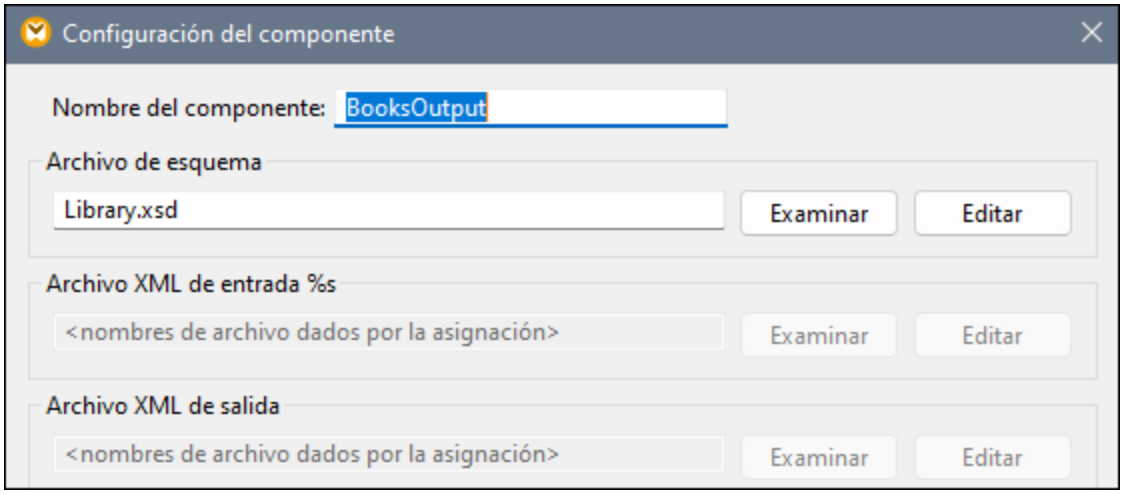

## Paso 2: Conectar la función concat con un nodo dinámico

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



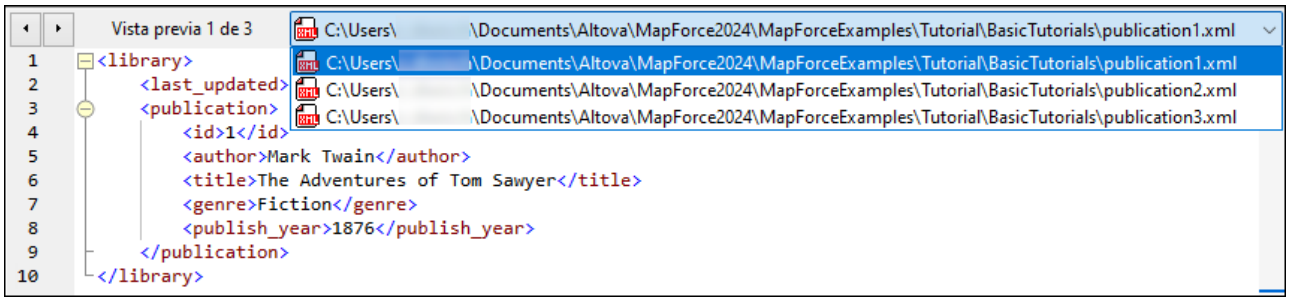

## Paso 3: Comprobar configuración del componente

En la configuración del componente, notará que las casillas *Archivo XML de entrada* y *Archivo XML de salida* están deshabilitadas y 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 posible definirlos en la configuración del componente.




## Paso 4: Generar archivos de salida

Ahora puede ejecutar la asignación y ver el resultado, así como los nombres de los archivos generados. 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 haciendo clic en un archivo en la lista desplegable adyacente (*imagen siguiente*).




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

 © 2019-2025 Altova GmbH

# 4 Componentes estructurales

En esta sección, encontrará información sobre distintos formatos de datos que puede usar como orígenes y destinos de datos:


- XML y XML Schema 138
- Bases de datos 174
- CSV y archivos de texto 344


MapForce también permite crear asignaciones de datos entre archivos binarios (datos BLOB). Para más información, consulte las funciones `lang | file` 662.

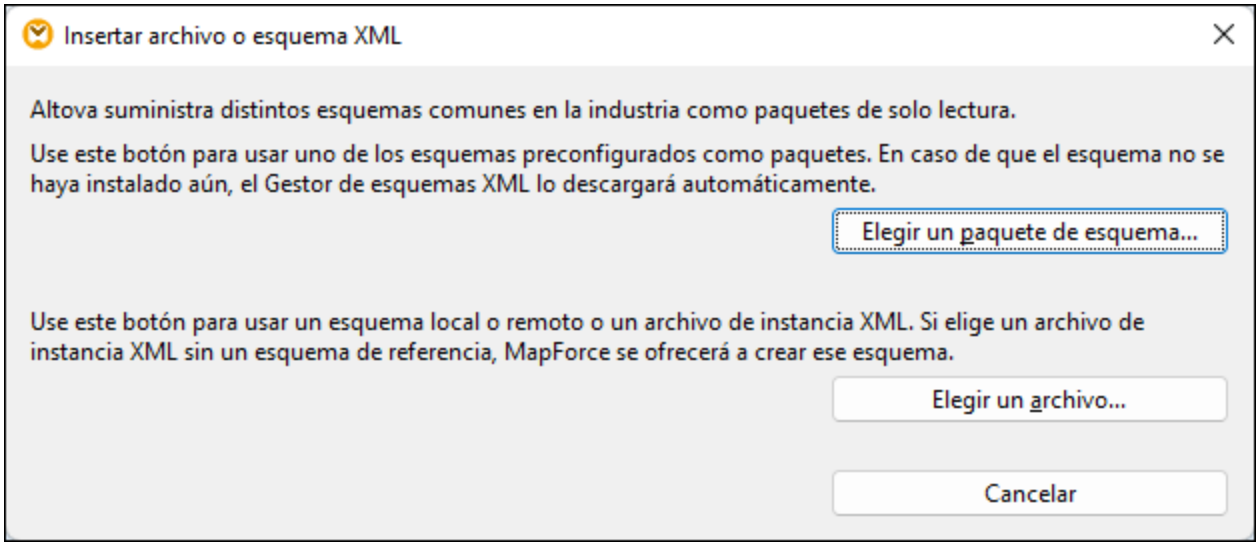
# 4.1 XML y esquemas XML

**Sitio web de Altova:** Asignación de datos XML

XML es un lenguaje de consultas para documentos de texto. XML Schema define la estructura y las restricciones de los documentos XML. En MapForce los archivos XML son componentes estructurales 34 que se pueden usar como orígenes y destinos de datos. Para más información sobre casos básicos de transformación de datos consulte los Tutoriales 103.

## 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 157 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

 © 2019-2025 Altova GmbH

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 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'
   ...
>
```

### Nota sobre los valores de enumeración

Para los nodos cuyos tipos de datos tienen facetas de enumeración, puede crear una asignación de valores que tendrá todos los valores de enumeración ya rellenados. Así le resultará más fácil procesar y asignar valores de enumeración. Para más información consulte el apartado Asignación de valores 449.

### En esta sección

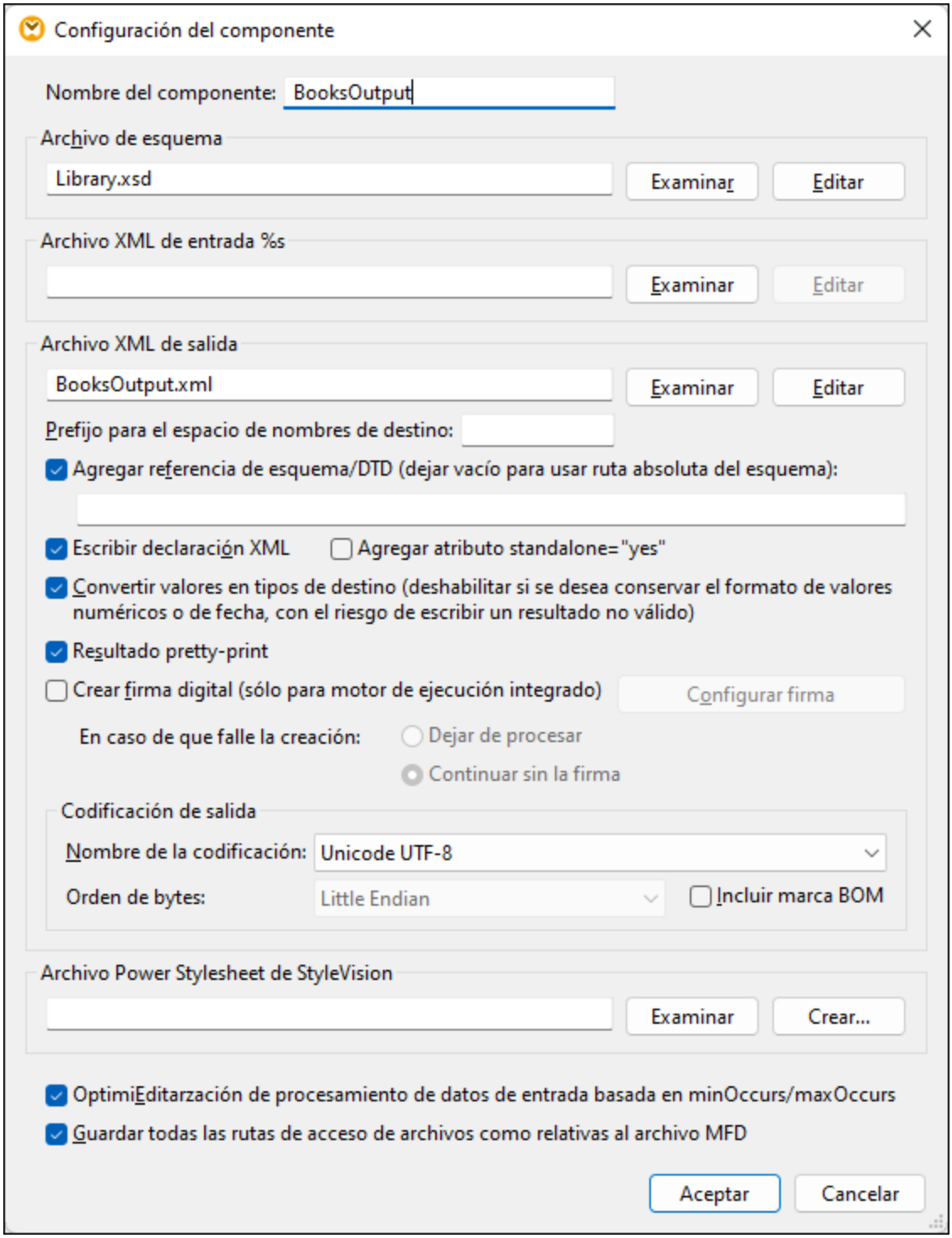
Estos son los apartados de esta sección:


- Configuración de componentes XML 139
- Tipos derivados 144
- Valores NULL 146
- Comentarios e instrucciones de procesamiento 149
- Secciones CDATA 150
- Comodines: xs:any / xs:anyAttribute 151
- Espacios de nombres personalizados 154
- Gestor de esquemas XML 157


## 4.1.1 Configuración de componentes XML

Una vez haya añadido un componente XML al área de la asignación puede configurarlo desde el cuadro de diálogo **Configuración del componente** (*imagen siguiente*). Puede abrir este cuadro de diálogo de una de las siguientes maneras:

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




A continuación puede ver las opciones que se pueden configurar en este cuadro de diálogo.

## Configuración general

- Nombre del componente


© 2019-2025 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 del esquema

Indica el nombre o la ruta de acceso del archivo de esquema XML que MapForce usa para validar y asignar los datos. Para cambiar el archivo del esquema haga clic en **Examinar** y seleccione un archivo nuevo. Para editar el archivo en Altova XMLSpy haga clic en **Editar**.

### Archivo XML de entrada

Indica el archivo de instancia XML del que MapForce debe leer los datos. Este campo es relevante para los componentes de origen y se rellena cuando se crea el componente y se le asigna un archivo de instancia XML. En un componente de origen el nombre del archivo de instancia también se usa para detectar el elemento raíz de la instancia XML y el esquema al que se hace referencia, así como para validar los datos con el esquema seleccionado. Para cambiar el archivo del esquema haga clic en **Examinar** y seleccione un archivo nuevo. Para editar el archivo en Altova XMLSpy haga clic en **Editar**.

### Archivo XML de salida

Indica el archivo de instancia XML en el que MapForce escribirá los datos. Este campo es significativo para un componente de destino. Para cambiar el archivo del esquema haga clic en **Examinar** y seleccione un archivo nuevo. Para editar el archivo en Altova XMLSpy haga clic en **Editar**.

### Prefijo para espacios de nombres de destino

Permite introducir un prefijo para el espacio de nombres de destino. Antes de asignar el prefijo debe asegurarse de que el espacio de nombres de destino está definido en el esquema de destino.

### Agregar referencia de esquema/DTD

Agrega la ruta del archivo de esquema XML al que se hace referencia al elemento raíz de los datos XML de salida. La ruta del esquema que se introduce 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 está usando una DTD.

*Ediciones MapForce Professional y Enterprise*: Si genera código en XQuery o C++ no se permite agregar también la referencia de la DTD.

Si introduce una ruta en este campo puede definir dónde se encuentra el archivo de esquema al que hace referencia el archivo de instancia XML. Esto permite poder validar el archivo de instancia de salida en la ubicación de destino de la asignación cuando esta se ejecute. En el campo Agregar referencia de esquema/DTD puede introducir una dirección **`http://`** o una ruta de acceso absoluta o relativa

Si desactiva esta opción puede desconectar la instancia XML de los esquemas XML o documentos DTD a los que se hace referencia. Esto puede ser útil, por ejemplo, si quiere enviar los datos XML de salida a alguien que no tiene acceso al esquema XML subyacente.

### Escribir una declaración XML

Esta opción está habilitada por defecto, lo que significa que la declaración XML se escribe en la salida. En la tabla siguiente puede ver la compatibilidad de esta funcionalidad con los lenguajes de destino de MapForce y sus motores de ejecución.

| **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*) | Sí | Sí |
| MapForce Server (*ediciones Professional y Enterprise*) | Sí | Sí |
| XQuery (*ediciones Professional y Enterprise*), XSLT | Sí | No |
| Generador de código (C++, C#, Java) (*ediciones Professional y Enterprise*) | Sí | Sí |

### Agregar independiente="yes"

Si selecciona esta opción se inserta la declaración `standalone="yes"` en la declaración XML del archivo XML de destino. Para saber más consulte la *declaración de documento independiente*.

Tenga en cuenta que:

- Cuando se selecciona la opción *independiente="yes"*, la generación de datos de salida es compatible con el lenguaje Built-In XSLT 1-3 y con la generación de código (C#, Java, C++ MSXML, C+ Xerces). El lenguaje de transformación Built-In [19] y el código generado están disponibles en las ediciones Professional y Enterprise. Para más información sobre la generación de código consulte Generador de código.
- No hay compatibilidad con XML incrustado en campos de bases de datos y solicitudes de servicios web (*ediciones Professional y Enterprise*).

### Convertir valores en tipos de destino

Permite (i) definir si los tipos del esquema XML de destino deben utilizarse durante la asignación o (ii) si todos los datos asignados al componente de destino deben considerarse valores de cadena. Por defecto, esta opción está habilitada. 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).

 © 2019-2025 Altova GmbH

- Resultado pretty-print

  Ajusta el formato del documento XML de salida para que tenga un aspecto estructurado. Cada nodo secundario se presenta alejado de su elemento primario por cuatro caracteres de espacio.

- Crear firma digital (*Enterprise Edition*)

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

## Codificación de salida

Permite especificar las siguientes opciones del archivo de instancia de salida:

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

La codificación de cualquier componente nuevo está definida por defecto en la opción *Codificación predeterminada para componentes nuevos*. Puede acceder a esta opción desde **Herramientas | Opciones** (sección *Generales*).

Si la asignación genera XSLT 1.0/2.0, marcar la casilla *Orden de bytes* no tiene ningún efecto, ya que estos lenguajes no son compatibles con el orden de bytes.

## Archivo hoja de estilos StyleVision Power

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 44.

## Otras opciones

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

  Esta opción permite un manejo especial de las secuencias de las que se sabe que contienen exactamente un elemento, como atributos necesarios o elementos secundarios con `minOccurs` y `maxOccurs="1"`. En este caso se extrae el primer elemento de la secuencia y se procesa directamente como un valor atómico (y no como una secuencia)

  Si los datos de entrada son **no válidos** para el esquema, la asignación genera un mensaje de error. Para permitir que se procesen datos de entrada **no válidos**, desmarque esta casilla.

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

  Cuando esta opción está habilitada, MapForce guarda las rutas de acceso al archivo que aparecen en el cuadro de diálogo **Configuración del componente** relativas a la ubicación del archivo de diseño de MapForce (**`.mfd`**). Consulte también Rutas de acceso relativas en un componente 44.

## 4.1.2 Tipos derivados

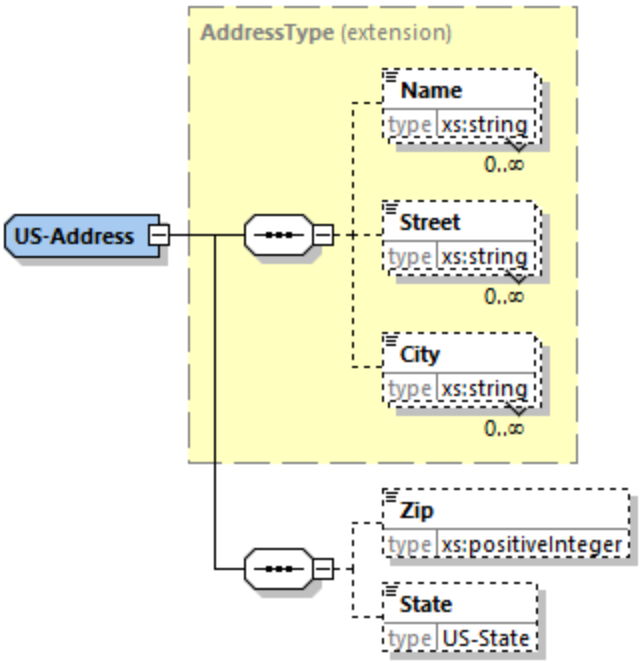
MapForce permite realizar asignaciones de datos entre tipos derivados de un tipo complejo. Los tipos derivados son tipos complejos de un esquema XML que usan el atributo `xsi:type` para identificar los tipos derivados especificados (p.e., `<Address xsi:type="UK-Address">`). Puede ver las definiciones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

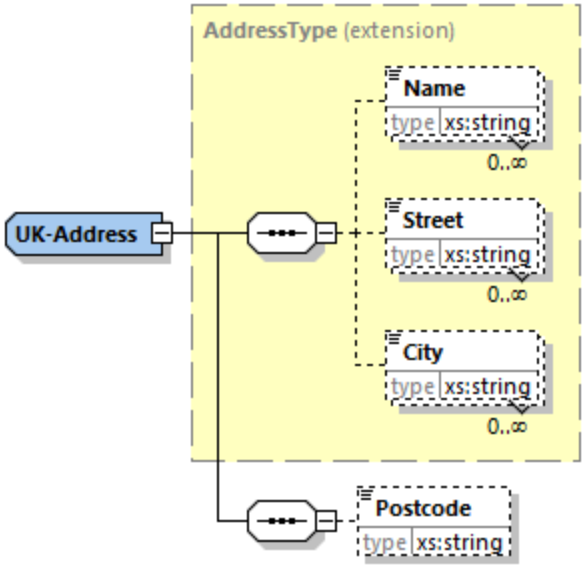
### Caso posible

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.

*Definición 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 sólo vamos a asignar al archivo de destino el elemento `UK-Address`.



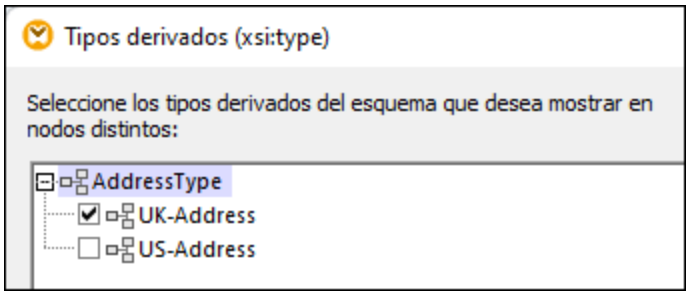

*Tipos derivados en una asignación*

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`**.

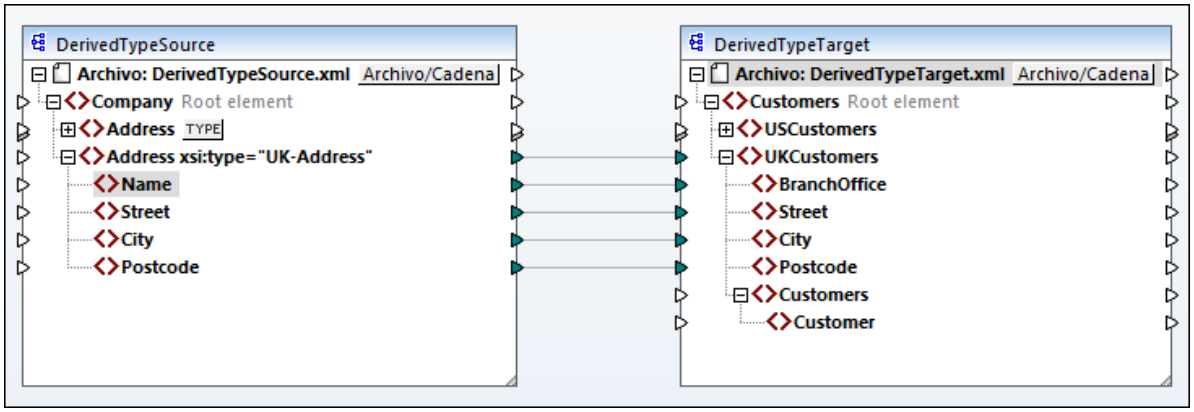
1. En el menú **Insertar** haga clic en el comando **Archivo o esquema XML** y abra **`DerivedTypeSource.xml`**. Este archivo XML se basa en **`DerivedTypeSource.xsd`**..

 © 2019-2025 Altova GmbH

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`.




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 NULL

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"`.

*Sólo 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 52.
- 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.

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

- La conexión está basada en destino 52.
- 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>`

 © 2019-2025 Altova GmbH

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>
  <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 52.
- 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 57.

```
<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`** 589: Ayuda a comprobar de forma explícita si un elemento de origen tiene el atributo `xsi:nil` definido como `true`.
- **`substitute-missing`** 623: Sustituye un valor NULL en el elemento de origen con algo específico.
- **`set-xsi-nil`** 591: 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`** 592: 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`** 597 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 sólo 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`** 592 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:

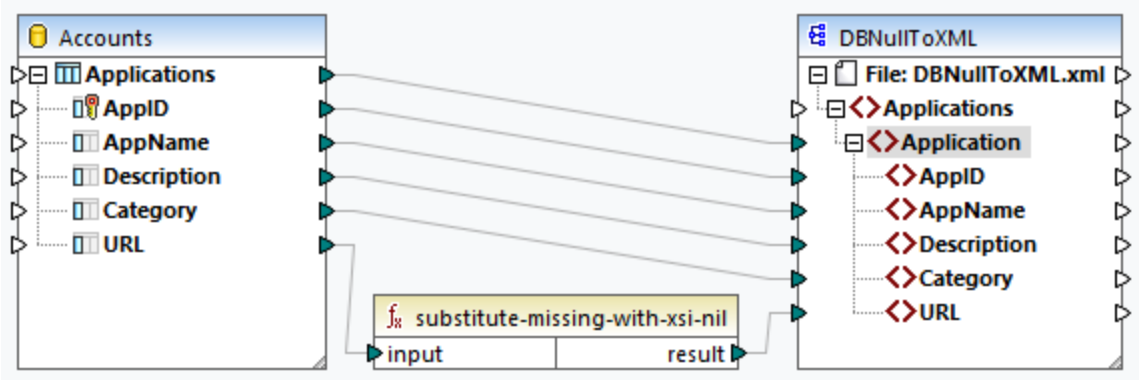
- Abra el panel **Consulta de BD**.
- Seleccione la base de datos `Accounts` para ver su estructura en el Explorador de BD.
- Haga clic con el botón derecho en la tabla `Application` y haga clic en **Mostrar en Editor SQL | SELECT**.
- Haga clic en el botón (**Ejecutar consulta**). Aparece la tabla `Application`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

Para saber más sobre consultas de BD consulte consulte Paneles de consulta de BD 294.

*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*).

 © 2019-2025 Altova GmbH




*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 sólo 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>
```

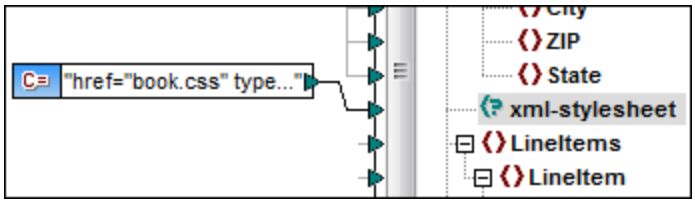
*Asignar elementos NULL a campos de BD NULL*
Si asigna un elemento XML NULL a una columna de BD MapForce escribe el valor NULL en la columna de BD correspondiente. También puede usar la función **`set-null`** 641 si quiere definir un campo de BD como NULL. Para saber más sobre las funciones relacionadas con bases de datos consulte la Biblioteca de bases de datos 640.

## 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 57.

*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:** Sólo 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 43.

*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.

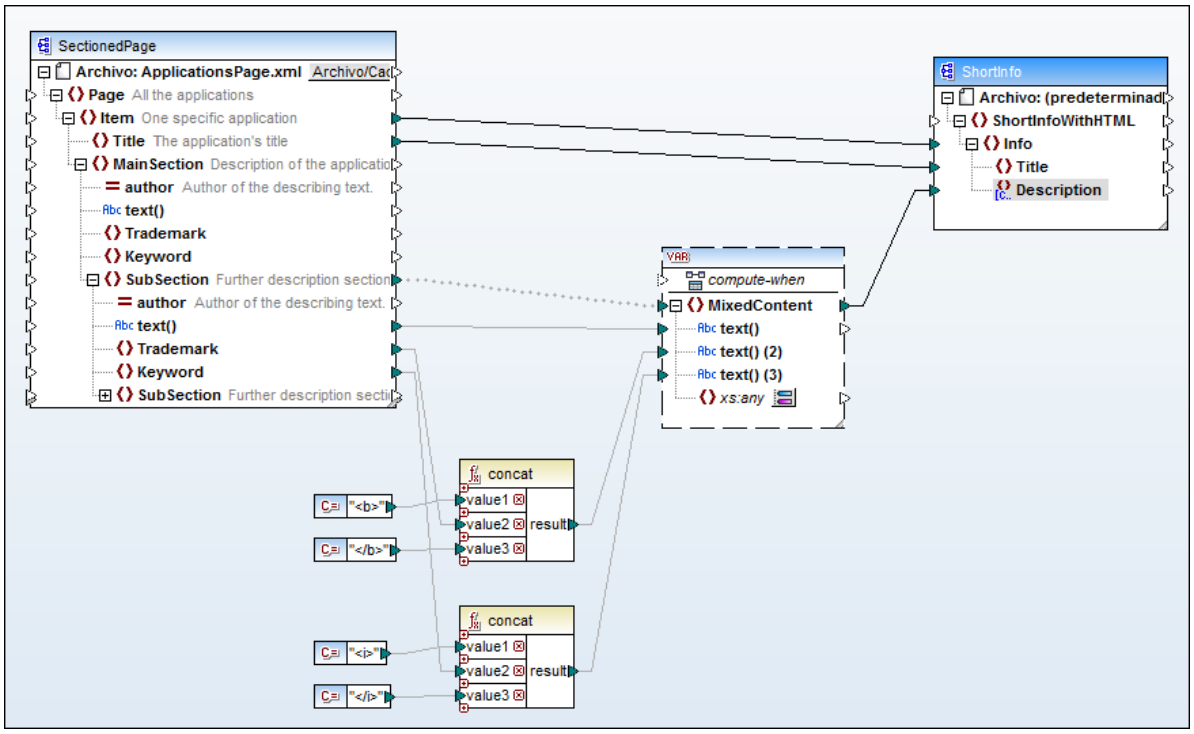
### Ejemplo

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 52 (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>`

 © 2019-2025 Altova GmbH

- 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 52 (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.




*Resultado*

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

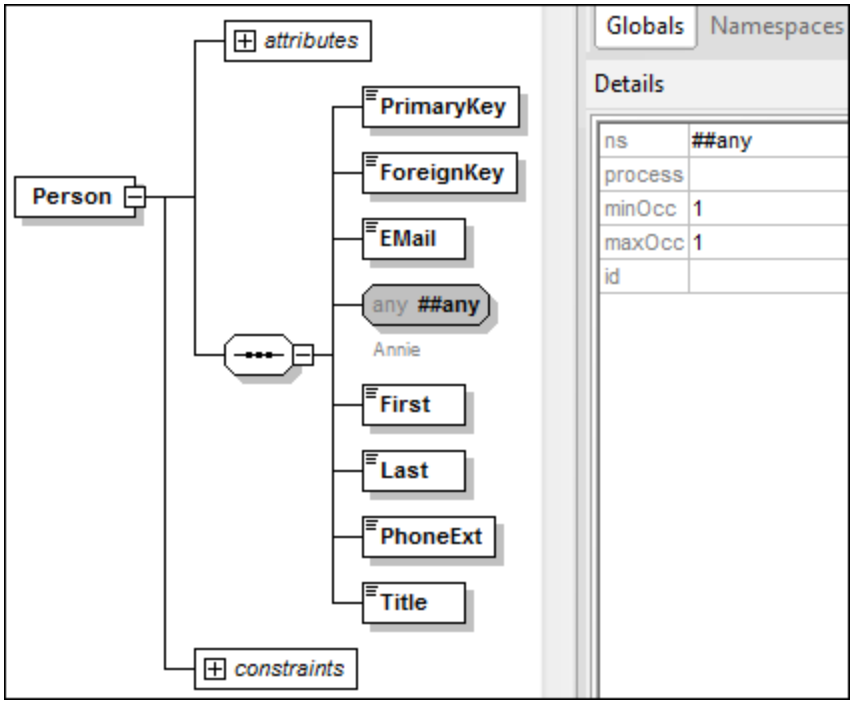
```
7   <Info>
8     <Title>MapForce</Title>
9     <Description><![CDATA[Altova <b>MapForce</b> 2014 Enterprise Edition is the premier <i>XML</i>
/ <i>database</i> / <i>flat file</i> / <i>EDI</i> data mapping tool that auto-generates mapping code in
<i>XSLT</i> 1.0/2.0, <i>XQuery</i> , <i>Java</i> , <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.

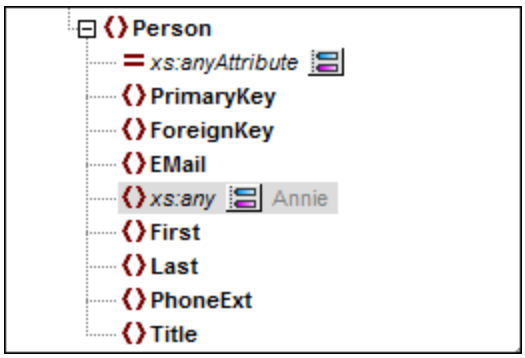
*Comodines en la definición del esquema*

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*

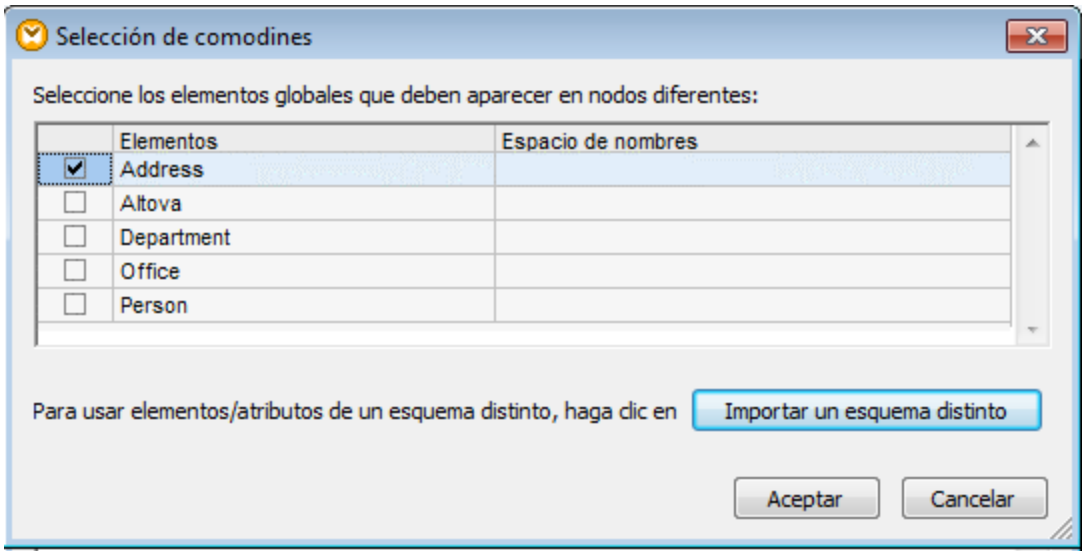
Cuando se define un comodín para un elemento y/o atributo, el nodo comodín aparece con el icono (**Cambiar selección**) junto a él (*imagen siguiente*).



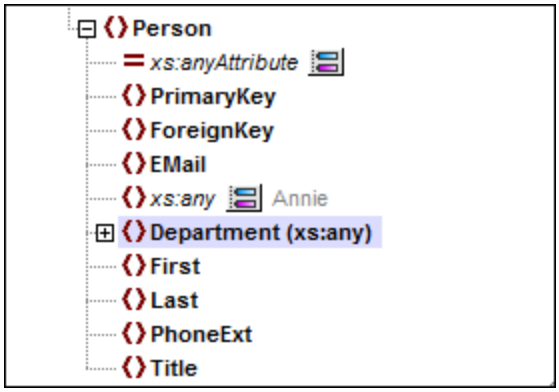

*Selección de comodines*

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.

 © 2019-2025 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 38) 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 763*).

## 4.1.7 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 32. Este es el comportamiento predeterminado de MapForce y el más apropiado en

 © 2019-2025 Altova GmbH

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 57.

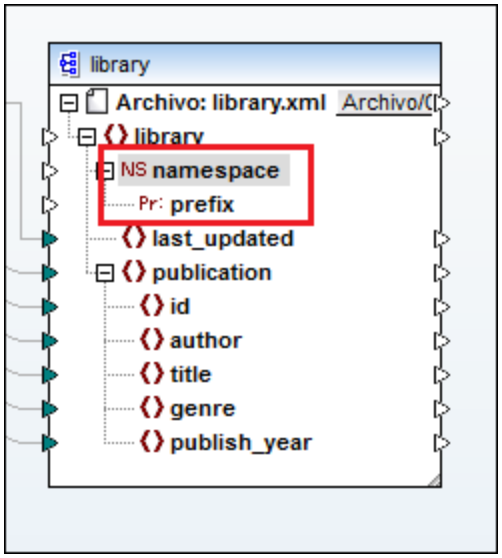
Para comprender mejor este funcionamiento siga las instrucciones de las subsecciones que siguen.

## Declarar un espacio de nombres manualmente

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

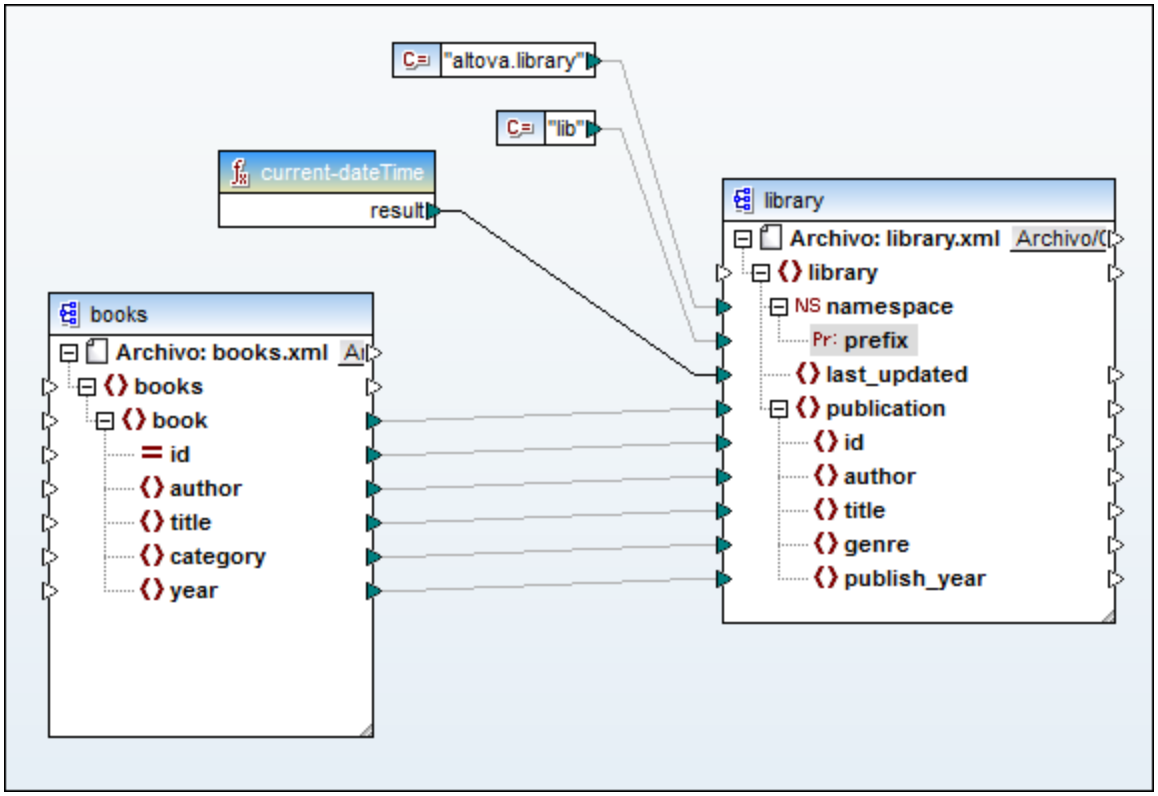
*Agregar un espacio de nombres*

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">
...
```

 © 2019-2025 Altova GmbH

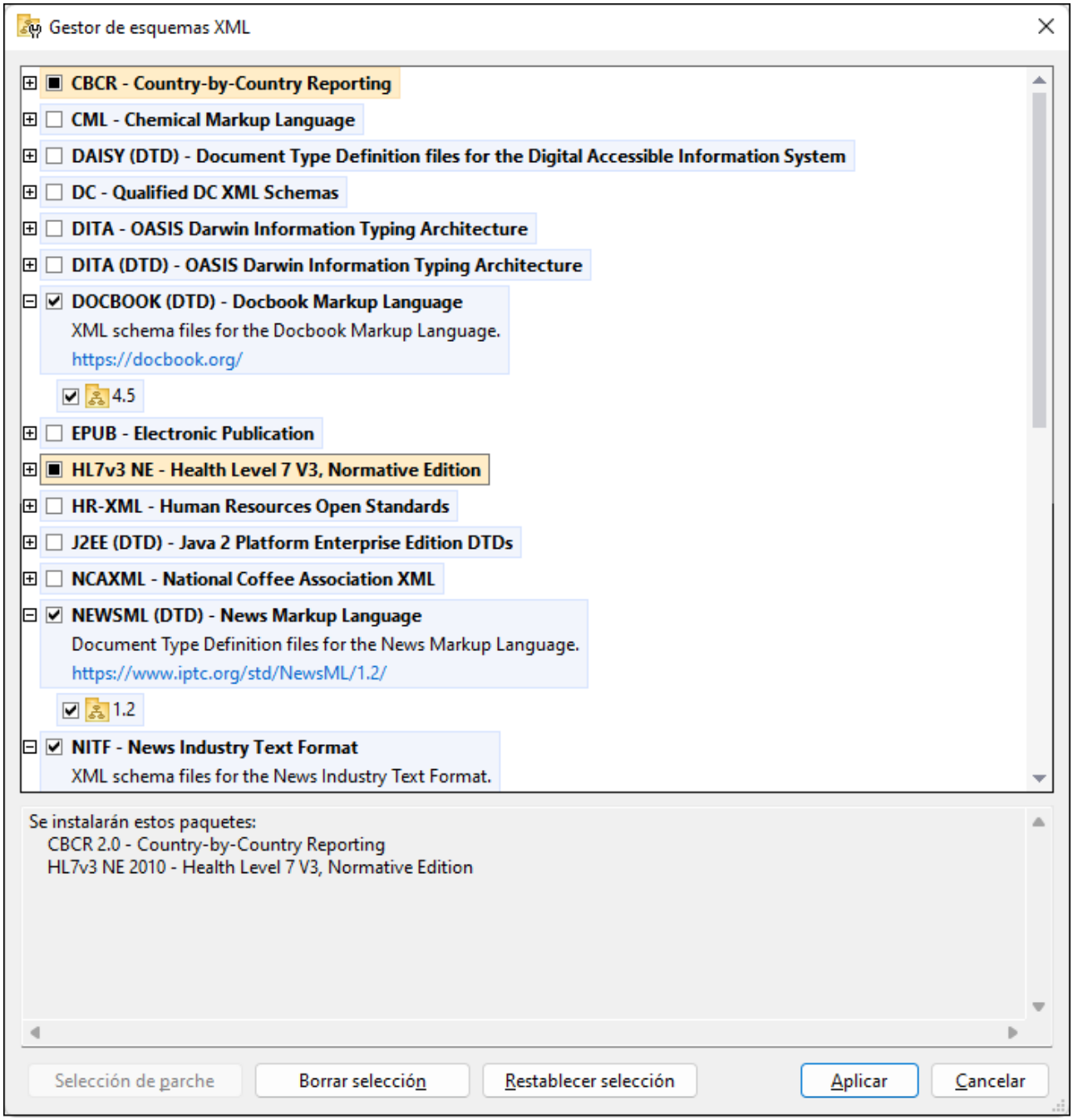
Si necesita crear prefijos para nombres de atributo (p. ej. `<number prod:id="prod557">557</number>`) puede habilitar el acceso dinámico a los atributos del nodo (*véase* *Asignar nombres de nodos* 763) o editar el esquema para que contenga un atributo `prod:id` para `<number>`.

*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 Schema, 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 |

 © 2019-2025 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 (*véase la tabla de más arriba*).

También se elimina automáticamente si desinstala todas las aplicaciones de Altova compatibles con XML 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.
- En los productos de Altova, si intenta validar con un esquema que no está instalado pero sí disponible con el Gestor de esquemas, este se instala automáticamente. Sin embargo, si el paquete de esquemas que quiere instalar contiene asignaciones de espacios de nombres, no puede instalarse automáticamente, sino que debe ejecutar Gestor de esquemas, seleccionar qué paquetes quiere instalar y ejecutar la instalación. Si después de instalar los paquetes la aplicación de Altova que está abierta no se reinicia automáticamente, debe reiniciarla manualmente.

## 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/es/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**[169], 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**[172] 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**[170] desde la línea de comandos y después ejecute el comando **initialize**[168].

## 4.1.8.1 Ejecutar el gestor de esquemas

### Interfaz gráfica del usuario

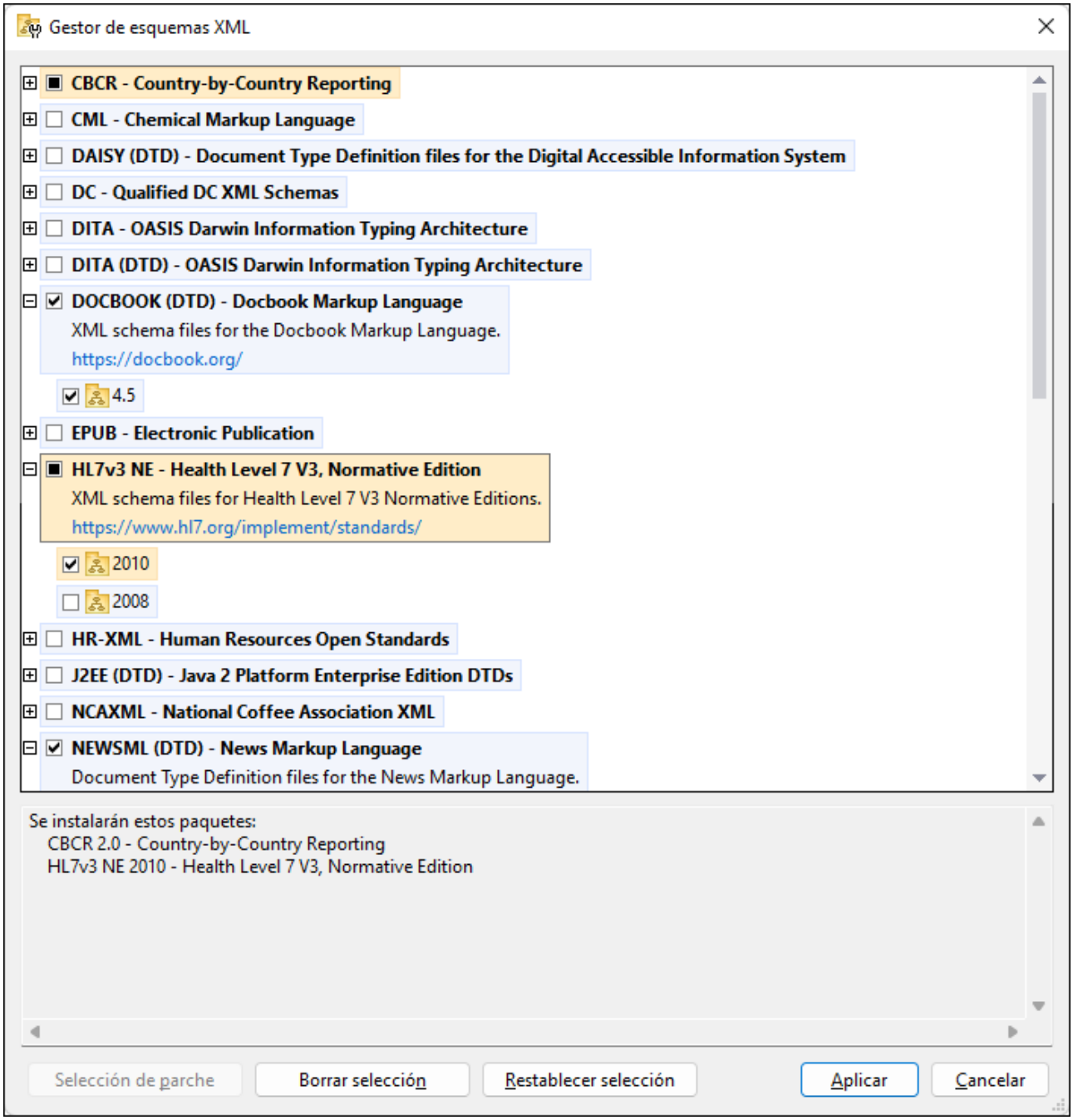
Hay varias formas de acceder a la interfaz gráfica del usuario de Gestor de esquemas:

- *Durante la instalación de MapForce:* al final del proceso de instalación, marque la casilla *Invocar el Gestor de esquemas XML de Altova* para acceder a la interfaz gráfica de usuario de Gestor de esquemas inmediatamente. Esto le permitirá instalar esquemas durante el proceso de instalación de su aplicación de Altova.
- *Después de instalar MapForce:* tras instalar la aplicación, puede acceder en todo momento a la interfaz gráfica del usuario de Gestor de esquemas, con el comando de menú **Herramientas | Gestor de esquemas XML**.


© 2019-2025 Altova GmbH

- A través del archivo **`.altova_xmlschemas`** descargado del sitio web de Altova: haga doble clic en el archivo descargado para ejecutar la interfaz gráfica del usuario de Gestor de esquemas, que se configurará para instalar los esquemas que haya seleccionado (en el sitio web).

Cuando se abra la interfaz gráfica del usuario de Gestor de esquemas (*imagen siguiente*), podrá ver los esquemas que ya están instalados. Si desea instalar algún esquema más, selecciónelo. Si desea desinstalar un esquema que está instalado, anule su selección. Cuando termine de seleccionar esquemas o de anular su selección, estará listo para aplicar sus cambios. Los esquemas que se vayan a instalar o desinstalar aparecerán resaltados y, en el panel Mensajes, un mensaje le avisará de los cambios que está a punto de hacer. El panel Mensajes se encuentra en la parte inferior de la ventana de Gestor de esquemas (*ver imagen*).




Al hacer clic en **Aplicar**, se muestra el progreso de la instalación. Si hay un error (por ejemplo, un error de conexión), se muestra un mensaje de error en el cuadro de diálogo. Si eso ocurre, haga clic en el botón **Avanzado** que aparece en el cuadro de diálogo, compruebe qué esquemas están seleccionados y vuelva a intentarlo con el botón **Aplicar**.

 © 2019-2025 Altova GmbH

### Interfaz de la línea de comandos

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

El archivo **`xmlschemamanager.exe`** se encuentra en esta carpeta:

- *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.

Luego, puede usar cualquiera de los comandos de la referencia de la interfaz de la línea de comandos 167.

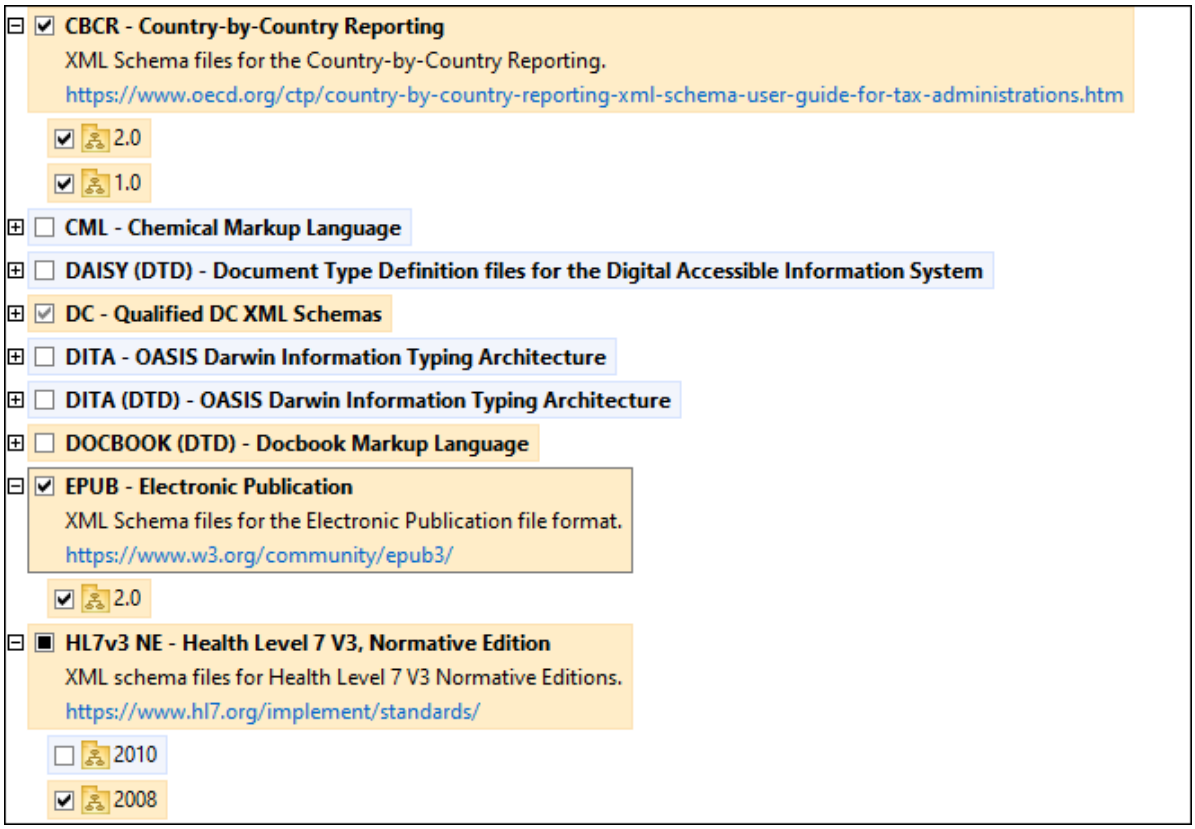
Si necesita ayuda con los comandos, ejecute:

- *En Windows:* `xmlschemamanager.exe --help`
- *En Linux o macOS (solo para las aplicaciones 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.




- *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`

 © 2019-2025 Altova GmbH

## 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*[163].) 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`[170]. Aparece una lista con los esquemas para las que hay parches disponibles.
1. Ejecute el comando `upgrade`[173] 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`[170]. Puede introducir tantos esquemas como quiera. Para más detalles consulte la descripción del comando `install`[169].

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

*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` 171:

```
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` 170. Puede introducir tantos esquemas como quiera. Para más detalles consulte la descripción del comando `uninstall` 171.

**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` 170.

 © 2019-2025 Altova GmbH

Una vez haya ejecutado este comando, asegúrese de que ejecuta también el comando `initialize` 168 para recrear el directorio caché. También puede ejecutar el comando `reset` 170 con la opción `-i`.

Recuerde que `reset -i` 170 restaura la instalación original del producto, por lo que es recomendable ejecutar el comando `update` 172 después de restaurar el gestor. Puede ejecutar el comando `reset` 170 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:

### 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` 170.)
- 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`**. |


© 2019-2025 Altova GmbH

| | |
|---|---|
| `--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` 170. 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* 157.

*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
```

### 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` 168 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` 172 después de una restauración. También puede ejecutar el comando `reset` con las opciones `-i` y `-u`.

#### Sintaxis

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

 © 2019-2025 Altova GmbH

*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` 170. 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* 157.

*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` 170 e `initialize` 168.

## 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
```

 © 2019-2025 Altova GmbH

### 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`**. |

## 4.2 Bases de datos

**Sitio web de Altova:** Asignación de datos de BD

MapForce le permite usar bases de datos como origen o destino de datos.

La tabla siguiente enumera todas las bases de datos compatibles. Si usa la versión de 64 bits , compruebe que tiene acceso a los controladores de BD de 64 bits de la BD a la que quiere conectarse.

| Base de datos | Notas |
|---|---|
| Firebird 2.x, 3.x, 4.x | |
| IBM DB2 8.x, 9.x, 10.x, 11.x, 12.x | |
| IBM Db2 for i 6.x, 7.4, 7.5 | Los archivos lógicos son compatibles y se muestran como vistas. |
| IBM Informix 11.70 y posterior | Informix es compatible con conexiones vía ADO, JDBC y ODBC. La implementación no admite tipos de datos de objetos grandes en ninguno de los lenguajes de generación de código. MapForce generará un mensaje de error (durante la generación de código) si se utiliza alguno de estos tipos de datos. |
| MariaDB 10 y posterior | MariaDB admite conexiones nativas. No se necesitan otros controladores. |
| Microsoft Access 2003 y posterior | Solo puede conectarse a una base de datos Access 2019 desde los productos de Altova (i) si está instalada la versión correspondiente de Microsoft Access Runtime y (ii) si la base de datos no utiliza el tipo de datos Large Number. |
| Microsoft Azure SQL Database | SQL Server 2016 codebase |
| Microsoft SQL Server 2005 y superior Microsoft SQL Server en Linux | |
| MySQL 5 y superior | MySQL 5.7 y sus versiones posteriores admiten conexiones nativas. No se necesitan otros controladores. |
| Oracle 9i y posterior | |
| PostgreSQL 8 y | Las conexiones PostgreSQL se admiten tanto como conexiones nativas como |

 © 2019-2025 Altova GmbH

| Base de datos | Notas |
|---|---|
| posterior | conexiones basadas en controladores a través de interfaces (controladores) como ODBC o JDBC. Las conexiones nativas no requieren controladores. |
| Progress OpenEdge 11.6 | |
| SQLite 3.x | Las conexiones SQLite se admiten como conexiones nativas y directas al archivo de base de datos SQLite. No se necesitan controladores separados. |
| Sybase ASE 15, 16 | |
| Teradata 16 | Son compatibles las conexiones vía ADO.NET, JDBC y ODBC.<br><br>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

Al generar código de programa 68 a partir de una asignación, compilar una asignación en archivos de ejecución de MapForce Server 839 o implementar una asignación en FlowForce Server 842, los detalles de la conexión de BD que se guardan con los archivos generados y están adaptados a los controladores compatibles para el entorno de destino elegido (*ver 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.

Cuando la asignación de datos se ejecute en un entorno distinto al de MapForce, tendrá que asegurarse de que los detalles de la conexión a la base de datos tienen sentido en el equipo que ejecuta la asignación (por ejemplo, puede que tenga que comprobar si el controlador de la base de datos está instalado, la ruta de la base de datos es correcta, el servidor de la base de datos es accesible, 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 | Tal cual | Transformada en JDBC | Tal cual | Transformada en JDBC |
| *ADO.NET* | Tal cual | Definida por el usuario | Transformada en JDBC | Tal cual | Transformada en JDBC |

| Tipo de conexión/Entorno de ejecución | C# | C++ | Java | MapForce Server en Windows | MapForce Server en Linux/Mac |
|---|---|---|---|---|---|
| *JDBC* | Definida por el usuario | Definida por el usuario | Tal cual | Tal cual | Tal cual |
| *ODBC* | puente ODBC | puente ODBC | Transformada en JDBC | Tal cual | Transformada en JDBC |
| *Native PostgreSQL* | No es compatible | No es compatible | No es compatible | Tal cual | Tal cual |
| *Native SQLite* | No es compatible | No es compatible | No es compatible | Tal cual | Tal cual |

Leyenda de la tabla:

- *Tal cual* significa que el tipo de conexión de la BD (p.ej., JDBC) es la que defina MapForce.
- *Transformada en JDBC* significa que la conexión de BD se transformará en una URL de conexión de BD similar a la JDBC.
- *Puente ADO* y *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 según corresponda (p.ej. `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 251.

*Prevención de posibles problemas con las conexiones JDBC en entorno Java*
Si la asignación se conecta a una base de datos por JDBC, compruebe que el controlador JDBC que utiliza la asignación está instalado en el sistema. Para ver la configuración JDBC actual de cualquier componente de BD en MapForce, haga doble clic en el encabezado del componente de BD. Esto abrirá el cuadro de diálogo **Configuración del componente**. Para más información consulte los apartados Configuración de componentes de base de datos 251 y Crear una conexión JDBC 197.

Si la asignación utiliza una conexión de base de datos no JDBC, esta conexión puede convertirse a JDBC durante la generación del código Java y así proporcionar compatibilidad en un entorno Java. Para más detalles consulte la tabla anterior.

Si ejecuta la aplicación Java que ha generado, puede que tenga que añadir el controlador JDBC como entrada classpath en la configuración actual. Si no lo hace, la ejecución de la aplicación podría resultar en un error similar a este: `java.lang.ClassNotFoundException: com.mysql.jdbc.Driver`.

En este subapartado explicamos algunos de los posibles métodos para solucionar problemas relacionados con JDBC.

***Método n°1: Añadir el controlador JDBC como dependencia en Eclipse***

 © 2019-2025 Altova GmbH

Si trabaja con el complemento Eclipse, deberá añadir un controlador JDBC como dependencia. Para ello, siga estos pasos:

1. Genere código Java en MapForce e importe el proyecto a Eclipse, tal y como se describe en el apartado Generar, compilar y ejecutar código 915.
2. Haga clic en la configuración deseada en Eclipse (p.ej. `MappingApplication`).
3. En la pestaña *Dependencias* haga clic en **Entradas classpath** y después en **Añadir JARs externas**.
4. Busque el archivo **`.jar`** en su controlador JDBC (p.ej. **`C:\jdbc\mysql\mysql-connector-java-5.1.16-bin.jar`**).
5. Haga clic en **Ejecutar** para ejecutar el programa con el controlador JDBC de BD añadido como dependencia.

***Método n°2: Añadir el controlador JDBC a la classpath de la tarea de prueba en build.xml***

Si aparece un mensaje de error relacionado al controlador JDBC al ejecutar el archivo de Ant **`build.xml`**, añada el controlador JDBC a la classpath de la tarea `test` en el archivo **`build.xml`**. El fragmento de código que se ve a continuación es un ejemplo de una tarea `test` de Ant que incluye una referencia al archivo **`.jar`** del controlador JDBC (*resaltado en amarillo más abajo*).

```
<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>
```

***Método n°3: Incluir el controlador JDBC en el manifiesto de la aplicación***

Si construye archivos JAR a partir de la aplicación Java generada, tendrá que añadir 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, siga estos pasos:

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 `value` sea el nombre del archivo **`.jar`**. Por ejemplo, para MySQL 5.1.16, el archivo `manifest` actualizado podría tener el siguiente aspecto (*observe la línea resaltada en amarillo*):

   ```
   <manifest file="C:\codegen\java\mysql_mapping/META-INF/MANIFEST.MF" mode="replace">
      <attribute name="Created-By" value="MapForce 2025"/>
      <attribute name="Main-Class" value="com.mapforce.MappingConsole"/>
      <attribute name="Class-Path" value="mysql-connector-java-5.1.16-bin.jar"/>
   </manifest>
   ```

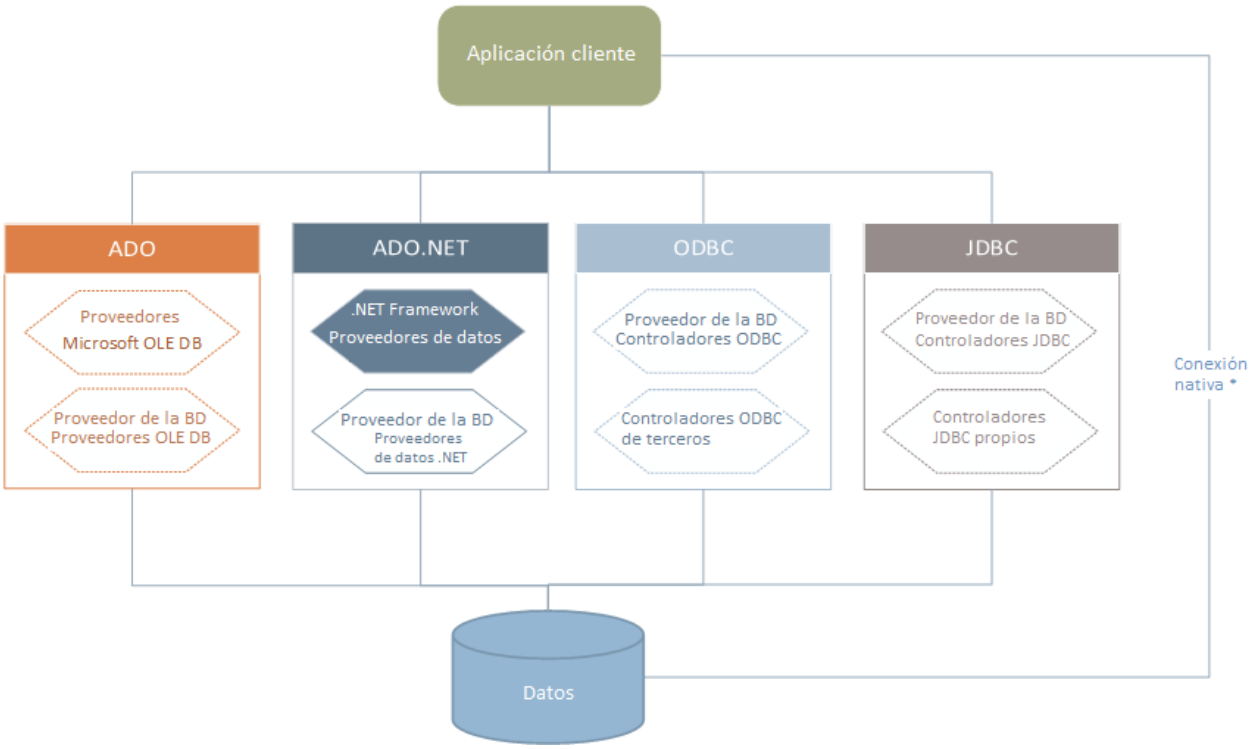
3. Copie el archivo JAR del controlador JDBC a la carpeta que contenga el archivo JAR de la aplicación generada.

## 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) se ejecute 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.

El diagrama que aparece más abajo presenta 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).




* Se admiten *conexiones nativas directas* 205 *para bases de datos SQLite, MySQL, MariaDB, PostgreSQL. Para conectarte a esas bases de datos, no necesita instalar ningún controlador adicional en tu sistema.*

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)

 © 2019-2025 Altova GmbH

- JDBC (Java Database Connectivity)
- ODBC (Open Database Connectivity)

**Nota:** Algunos proveedores de ADO.NET no son compatibles o tienen una compatibilidad limitada. Consulte el apartadoNotas sobre compatibilidad con ADO.NET 196.

### 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 de 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.

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 bases de datos publican habitualmente software cliente de bases de datos y controladores disponibles públicamente que proporcionan acceso multiplataforma a la base de datos respectiva a través de cualquier combinación de ADO, ADO.NET, ODBC o JDBC. Además, puede encontrar otros controladores de otras organizaciones para las tecnologías de acceso de datos mencionadas. En la mayoría de los casos, existe más de una forma de conectarse a la base de datos necesaria desde su sistema operativo y, en consecuencia, desde MapForce. 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 a BD

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 181.

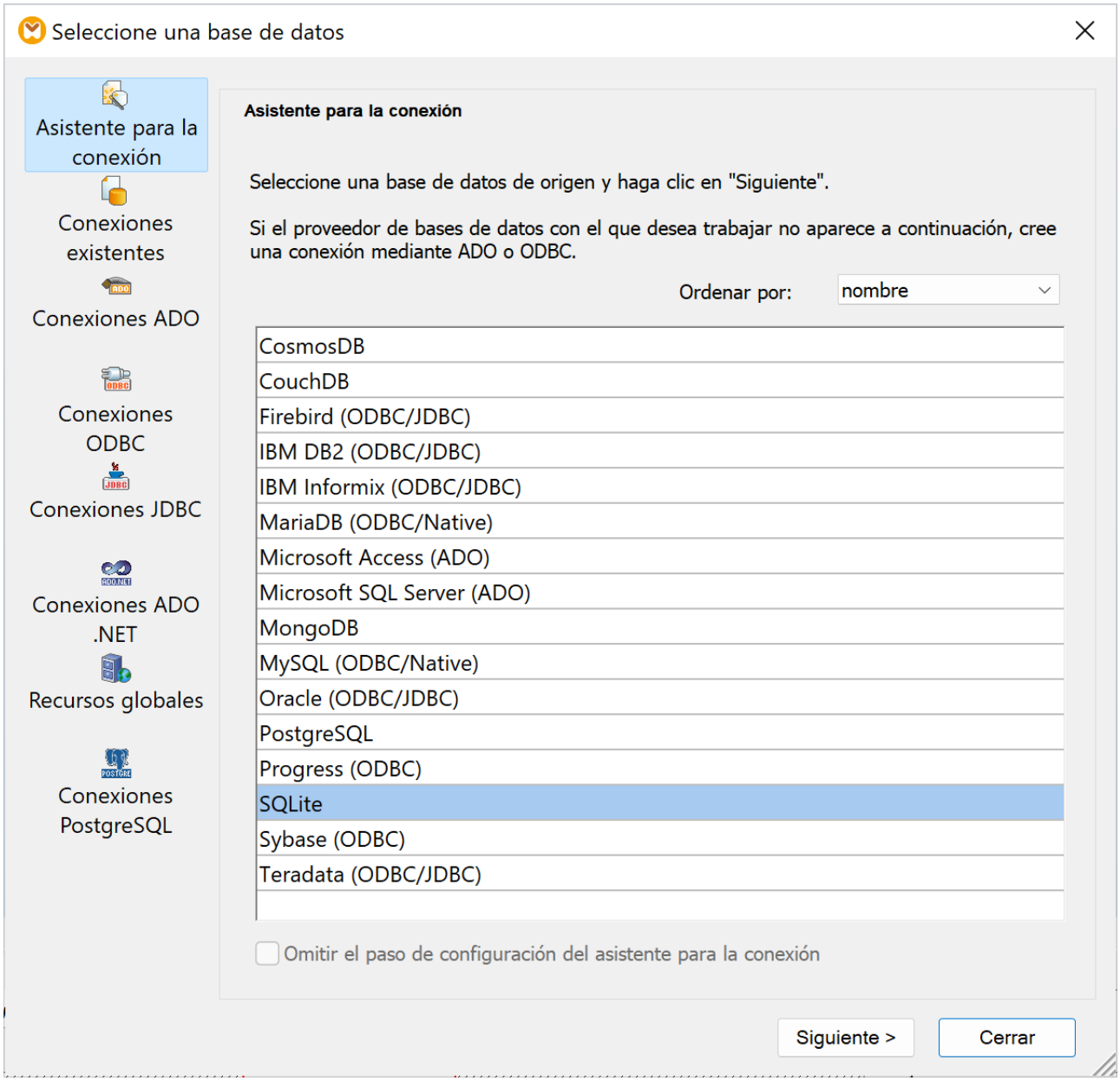
Para iniciar el asistente para la conexión de BD (*imagen siguiente*) siga estos pasos:

- 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 207.

También puede consultar estos apartados:


- Configurar una conexión ADO 185
- Configurar una conexión ADO.NET 191
- Configurar una conexión ODBC 201

- Configurar una conexión JDBC 197

## 4.2.1.2 Resumen de controladores de BD

En este apartado ofrecemos un resumen de los controladores de BD. Aunque puede que Windows venga con algunos controladores de BD, es posible que necesite descargar más controladores.

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 utilizar el software cliente de BD, le recomendamos que lea detenidamente las instrucciones de instalación y configuración del cliente de BD, que pueden variar según la versión de 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. Consulte el apartado Notas sobre compatibilidad con ADO.NET 196.
- 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 203).
- 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. Para más información, consulte Configurar una conexión ODBC 201.
- 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 configurada. Para más información, véase Configurar una conexión JDBC 197.
- 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.

### Tecnologías disponibles de acceso a los 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 a los datos. Tenga en cuenta que esta lista no es exhaustiva ni prescriptiva; puede utilizar otras alternativas nativas o de terceros además de los controladores que se muestran a continuación.

| Base de datos | Interfaz | Controladores |
|---|---|---|
| Firebird | ADO.NET | Proveedor de datos ADO.NET (https://www.firebirdsql.org/en/additional-downloads/) |
| | 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 para i | ADO | • Proveedor IBM DB2 para i5/OS IBMDA400 OLE DB<br>• Proveedor IBM DB2 para i5/OS IBMDARLA OLE DB<br>• Proveedor IBM DB2 para i5/OS IBMDASQL OLE DB |
| | ADO.NET | Proveedor de datos .NET Framework para IBM i |
| | JDBC | Controlador IBM Toolbox para Java JDBC |
| | ODBC | Controlador iSeries Access ODBC |
| IBM Informix | ADO | Proveedor IBM Informix OLE DB |
| | JDBC | Controlador IBM Informix JDBC |
| | ODBC | Controlador IBM Informix ODBC |
| Microsoft Access | ADO | • Proveedor Microsoft Jet OLE DB<br>• Proveedor Microsoft Access Database Engine OLE DB |
| | ADO.NET | Proveedor de datos .NET Framework para OLE DB |
| | ODBC | • Controlador Microsoft Access |


© 2019-2025 Altova GmbH

| Base de datos | Interfaz | Controladores |
|---|---|---|
| | | |
| MariaDB | ADO.NET | Si falta el conector .NET especial para MariaDB, use **Connector/NET** para MySQL (https://dev.mysql.com/downloads/connector/net/). |
| | JDBC | MariaDB Connector/JDBC (https://downloads.mariadb.org/) |
| | ODBC | MariaDB Connector/ODBC (https://downloads.mariadb.org/) |
| | Conexión nativa | Disponible. No se requiere ningún controlador. |
| Microsoft SQL Server | ADO | • Controlador Microsoft OLE DB para SQL Server (MSOLEDBSQL)<br>• Proveedor Microsoft OLE DB para SQL Server (SQLOLEDB)<br>• Cliente nativo de SQL Server (SQLNCLI) |
| | ADO.NET | • Proveedor de datos .NET Framework para SQL Server<br>• Proveedor de datos .NET Framework para OLE DB |
| | JDBC | • Controlador Microsoft JDBC para SQL Server ( https://msdn.microsoft.com/library/mt484311.aspx ) |
| | ODBC | • Controlador ODBC para Microsoft SQL Server ( https://docs.microsoft.com/en-us/SQL/connect/odbc/download-odbc-driver-for-sql-server ) |
| MySQL | ADO.NET | • Conector/NET (https://dev.mysql.com/downloads/connector/net/) |
| | JDBC | Conector/J ( https://dev.mysql.com/downloads/connector/j/ ) |
| | ODBC | Conector/ODBC ( https://dev.mysql.com/downloads/connector/odbc/ ) |
| | Conexión nativa | Disponible para MySQL 5.7 y superior. No se requiere ningún controlador. |
| Oracle | ADO | • Proveedor Oracle para OLE DB<br>• Proveedor Microsoft OLE DB para Oracle |
| | ADO.NET | Proveedor de datos Oracle para .NET (https://www.oracle.com/technetwork/topics/dotnet/index-085163.html) |

| Base de datos | Interfaz | Controladores |
|---|---|---|
| | JDBC | • Controlador JDBC Thin<br>• Controlador JDBC Oracle Call Interface (OCI)<br><br>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<br>• 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 se requiere ningún controlador. |
| 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 se requiere ningún controlador. |
| Sybase | ADO | Proveedor 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 Teradata JDBC (https://downloads.teradata.com/download/connectivity/jdbc-driver) |


© 2019-2025 Altova GmbH

| Base de datos | Interfaz | Controladores |
|---|---|---|
| | ODBC | Controlador Teradata ODBC 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 191 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* Microsoft OLE DB para SQL Server (SQLOLEDB)
- Cliente nativo de SQL Server (SQLNCLI)

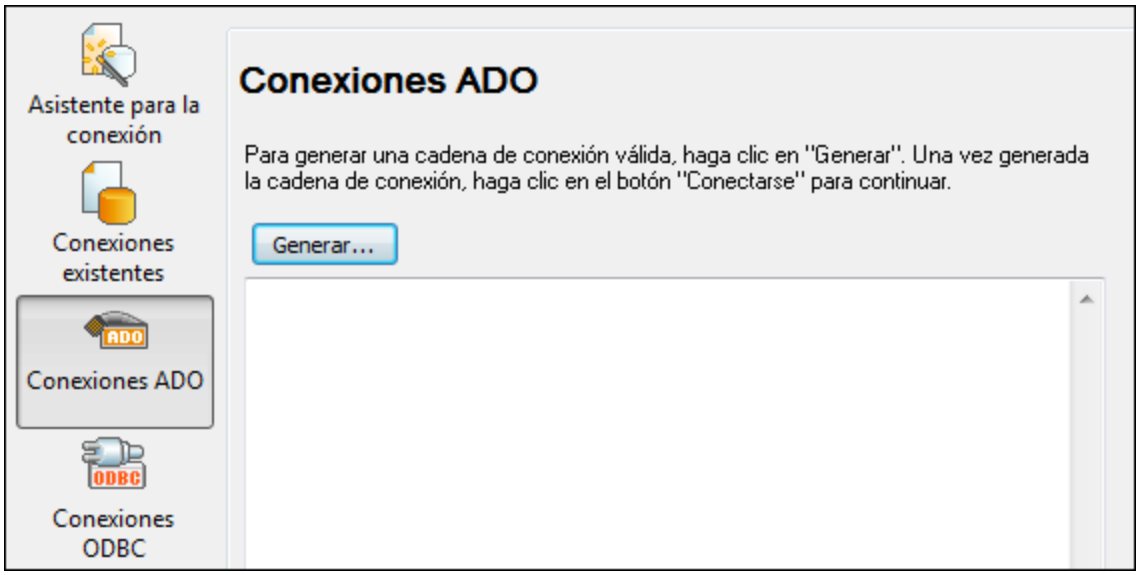
De entre estos tres proveedores recomendamos que use MSOLEDBSQL, que puede descargar desde https://docs.microsoft.com/en-us/sql/connect/oledb/download-oledb-driver-for-sql-server?view=sql-server-ver15. Tenga en cuenta que la versión que descargue debe coincidir con la de la plataforma de MapForce (32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

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

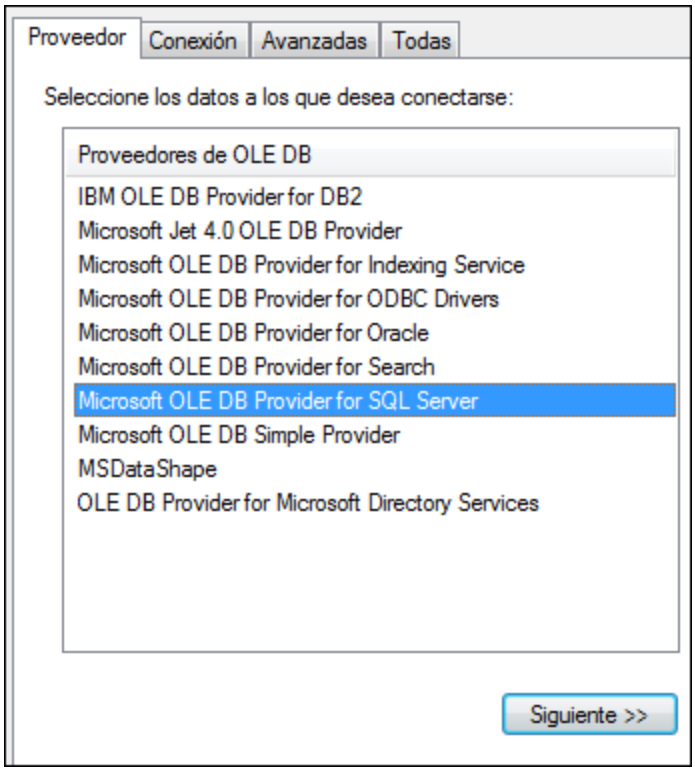
### Configurar una conexión ADO

Para configurar una conexión ADO:

1. Inicie el asistente para la conexión de base de datos 179.
2. Haga clic en **Conexiones ADO**.




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.

 © 2019-2025 Altova GmbH

| Para conectarse a esta BD... | Utilice este proveedor... |
|---|---|
| Microsoft Access | • **Proveedor de Microsoft Office Access Database Engine OLE DB** (recomendado)<br>• **Proveedor Microsoft Jet OLE DB**<br><br>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. |
| Servidor SQL | • **Controlador de Microsoft OLE DB para SQL Server (MSOLEDBSQL)**: este es el proveedor OLE DB recomendado. Para que este proveedor aparezca en la lista, debe descargarse de https://docs.microsoft.com/en-us/sql/connect/oledb/download-oledb-driver-for-sql-server?view=sql-server-ver15 e instalarse.<br>• **Microsoft OLE DB Provider for SQL Server (OLEDBSQL)**<br>• **Cliente nativo de SQL Server (SQLNCLI)** |
| Otra base de datos | Seleccione el proveedor correspondiente.<br><br>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 181). Otra opción es configurar una conexión ADO.NET, ODBC o JDBC.<br><br>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 201. |

5. Una vez seleccionado el proveedor de su elección, haga clic en **Siguiente** y siga los pasos del asistente hasta el final.

Las instrucciones de las siguientes pantallas dependen del proveedor elegido. Para SQL Server, deberá indicar o seleccionar el nombre de host del servidor de la BD, el método de autenticación y el nombre de la BD, así como el nombre de usuario y la contraseña de la BD. Para ver un ejemplo, consulte Conectarse a Microsoft SQL Server (ADO) 223. Para Microsoft Access, se le pedirá que busque o proporcione la ruta al archivo de la BD. Para ver un ejemplo consulte Conectarse a Microsoft Access (ADO) 221.

La lista completa de propiedades de inicialización (parámetros de conexión) está disponible en la pestaña **Todas** del cuadro de diálogo de conexión; estas propiedades varían en función del proveedor elegido y puede ser necesario establecerlas explícitamente para que la conexión sea posible. Consulte estos apartados del manual 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 188
- Configurar las propiedades de vínculo de datos de Microsoft Access 189

#### 4.2.1.3.1 Conectarse a una BD Microsoft Access existente

El procedimiento que se indica a continuación es adecuado si desea conectarse a una BD de Microsoft Access que no esté protegida por contraseña. Si la base de datos está protegida por contraseña, utilice el método indicado en Ejemplo de conexión para Microsoft Access (ADO)[221].

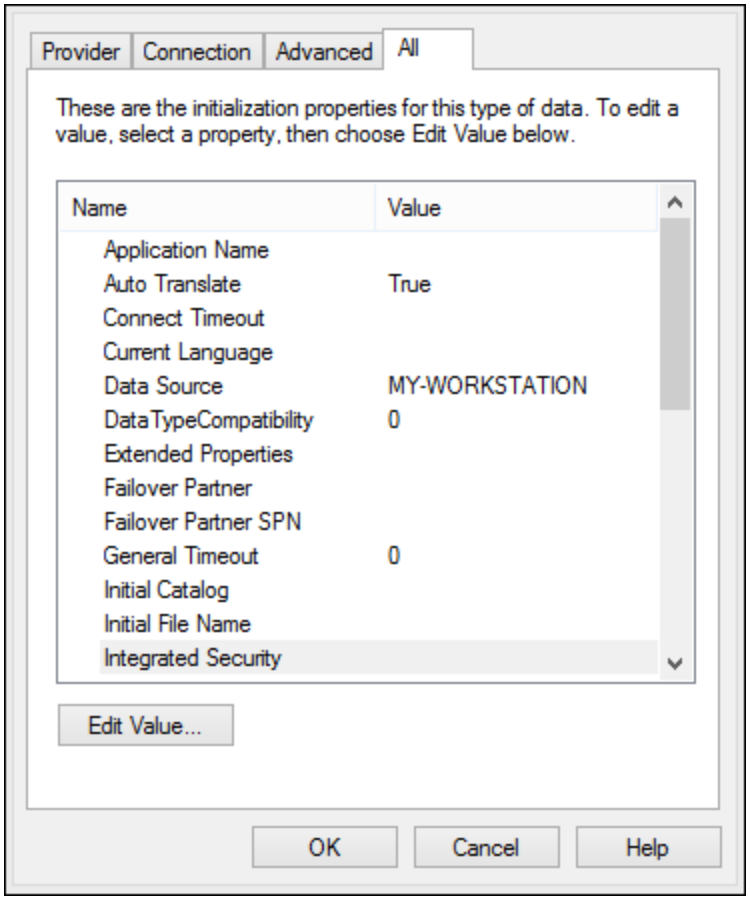
1. Ejecute el asistente de conexión a la base de datos (consulte Iniciar el asistente para la conexión de base de datos[179]).
2. Seleccione **Microsoft Access (ADO)** y, luego, haga clic en **Siguiente**.
3. Busque el archivo de BD o introduzca su ruta de acceso (relativa o absoluta).
4. Haga clic en **Conectarse**.

#### 4.2.1.3.2 Configurar las propiedades de vínculo de datos de SQL Server

Si se conecta a una BD de Microsoft SQL Server a través de ADO[185], compruebe las siguientes propiedades en la pestaña *Todas* del cuadro de diálogo Propiedades de vínculo de datos (*imagen siguiente*).

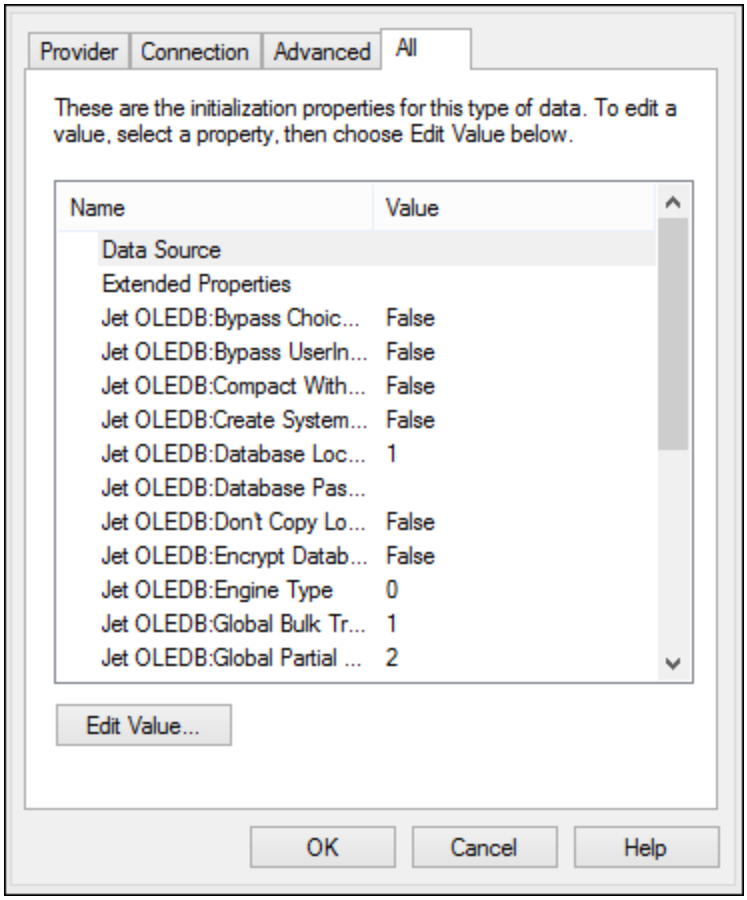
- *Seguridad integrada:* Si SQL Server Native Client es el proveedor de datos en la pestaña Proveedor, esta propiedad debe tener como valor un carácter de espacio.
- *Almacenar información de seguridad:* esta propiedad debe tener el valor *True*.


© 2019-2025 Altova GmbH



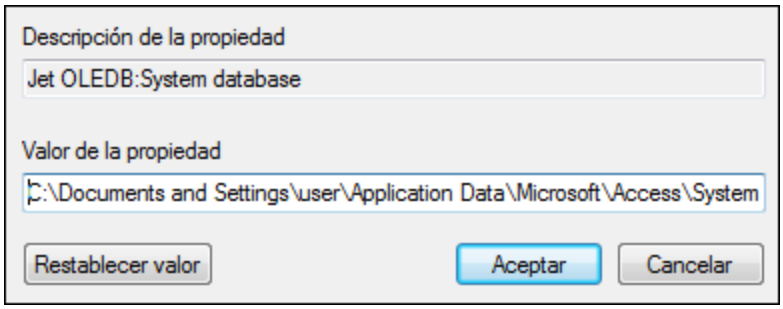

### 4.2.1.3.3 Configurar las propiedades de vínculo de datos de Microsoft Access

Si se conecta a una BD de Microsoft Access a través de ADO[185], en la pestaña *Todas* del cuadro de diálogo Propiedades de vínculo de datos (*imagen siguiente*), marque las propiedades que aparecen debajo de la imagen.



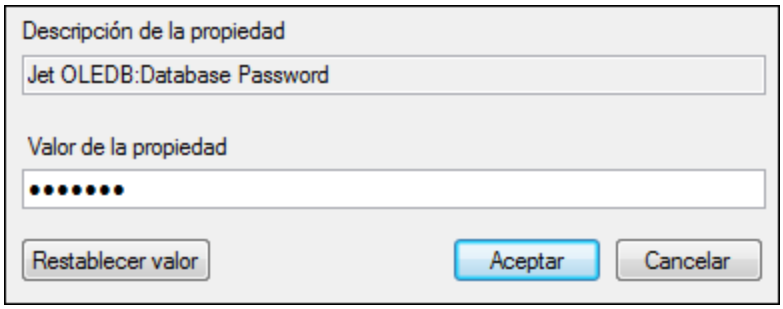

*Origen de datos*
Esta propiedad almacena la ruta de acceso del archivo de BD de Microsoft Access. Recomendamos utilizar el formato de ruta UNC (convención de nomenclatura universal), como por ejemplo: `\\anyserver\share$\filepath`

*Base de datos Jet OLEDB:System*
Esta propiedad almacena la ruta al archivo de información del grupo de trabajo. Es posible que tenga que configurar explícitamente el valor de esta propiedad para poder conectarse a una BD de Microsoft Access. Si no puede conectarse debido a un error relacionado con el archivo de información del grupo de trabajo, localice el archivo de información del grupo de trabajo (`System.MDW`) que corresponde a su perfil de usuario y configure el valor de la propiedad como la ruta del archivo `System.MDW`.




 © 2019-2025 Altova GmbH

*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. Conéctese a una base de datos a través de ADO.NET seleccionando un proveedor .NET e indicando 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 (por ejemplo, un servidor SQL o una base de datos Oracle), ejecutar comandos en el origen de datos y obtener datos de este. Así pues, cuando utiliza ADO.NET, MapForce interactúa con una base de datos a través de un proveedor de datos, uno que está optimizado para trabajar con el tipo específico de origen de datos para el que está diseñado.

Hay dos tipos de proveedores .NET:

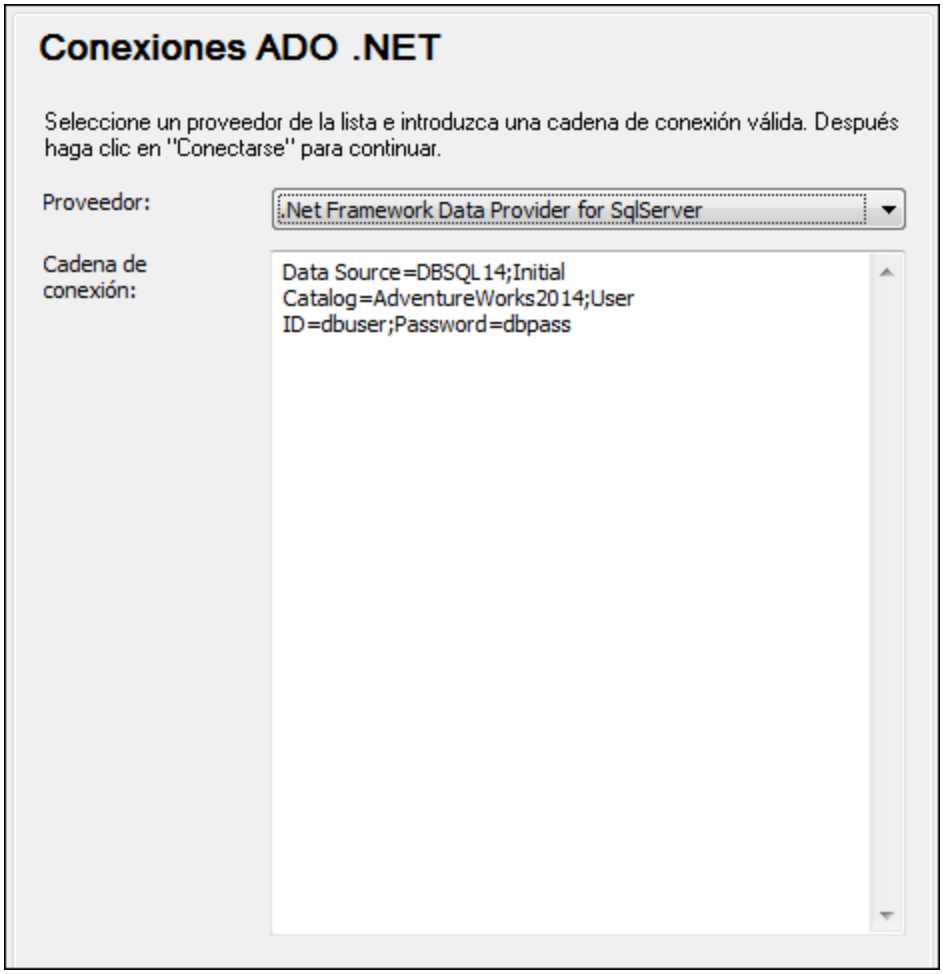
- El proveedor que viene por defecto con Microsoft .NET Framework.
- Proveedores que ofrecen los principales proveedores de bases de datos como extensión para .NET Framework. Este tipo de proveedores ADO.NET debe instalarse por separado y por lo general se pueden descargar del sitio web del proveedor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

**Nota:** ciertos proveedores ADO.NET no son compatibles o tienen una compatibilidad limitada. Consulte el apartado Notas sobre compatibilidad con ADO.NET 196.

Configure la conexión de la siguiente manera:

1. Inicie el asistente para la conexión de base de datos 179.
2. Haga clic en **Conexiones ADO.NET**.
3. Seleccione un proveedor de datos .NET de la lista. La lista de proveedores disponibles por defecto con .NET Framework aparece en la lista Proveedor. Los proveedores de datos .NET específicos de un proveedor solo estarán disponibles en la lista si ya están instalados en su sistema. Para que estén disponibles, los proveedores .NET específicos del proveedor deben instalarse en el caché global de ensamblados (GAC) ejecutando el archivo `.msi` o `.exe` suministrado por el proveedor de la base de datos.

4. Introduzca una cadena de conexión a la BD. Una cadena de conexión define la información de conexión a la base de datos utilizando pares de parámetros en formato clave/valor, separados por punto y coma. Por ejemplo, una cadena de conexión como **`Data Source=DBSQLSERV;Initial Catalog=ProductsDB;User ID=dbuser;Password=dbpass`** conecta con la base de datos SQL Server **`ProductsDB`** en el servidor **`DBSQLSERV`**, con el nombre de usuario **`dbuser`** y la contraseña **`dbpass`**. Puede crear una cadena de conexión escribiendo los pares clave/valor directamente en el cuadro de diálogo Cadena de conexión. Otra opción es crear la cadena con Visual Studio (véase Crear una cadena de conexión en Visual Studio 192). La sintaxis de la cadena de conexión depende del proveedor que se seleccione en la lista Proveedor. Para ver un ejemplo consulte el apartado Ejemplo: cadenas de conexión ADO.NET 195.




5. Haga clic en Conectarse.

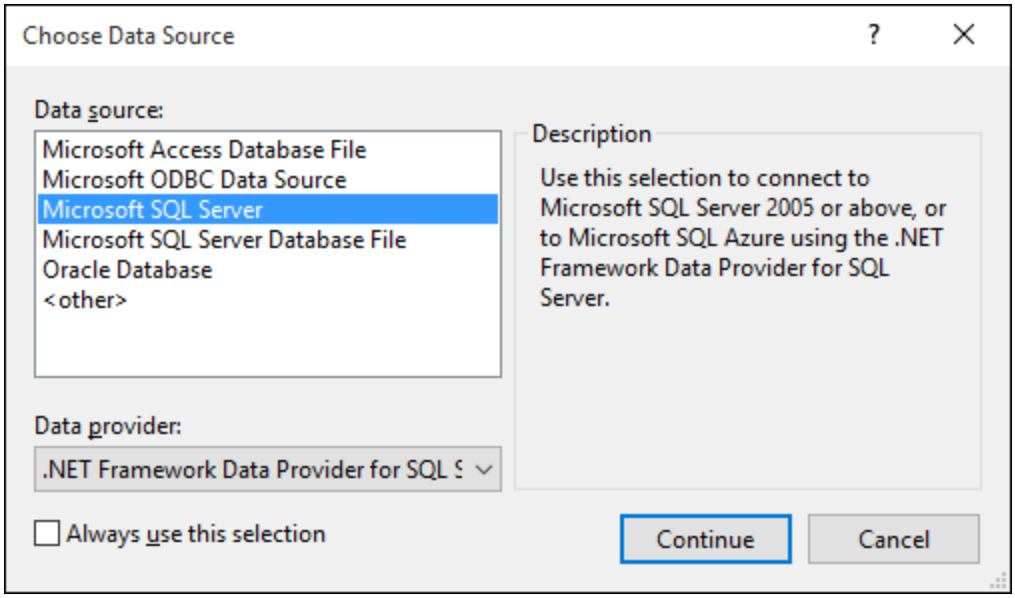
### 4.2.1.4.1 Crear una cadena de conexión en Visual Studio

Para conectarse a una fuente de datos mediante ADO.NET, se necesita una cadena de conexión a la base de datos válida. A continuación, explicamos cómo crear una cadena de conexión desde Visual Studio.

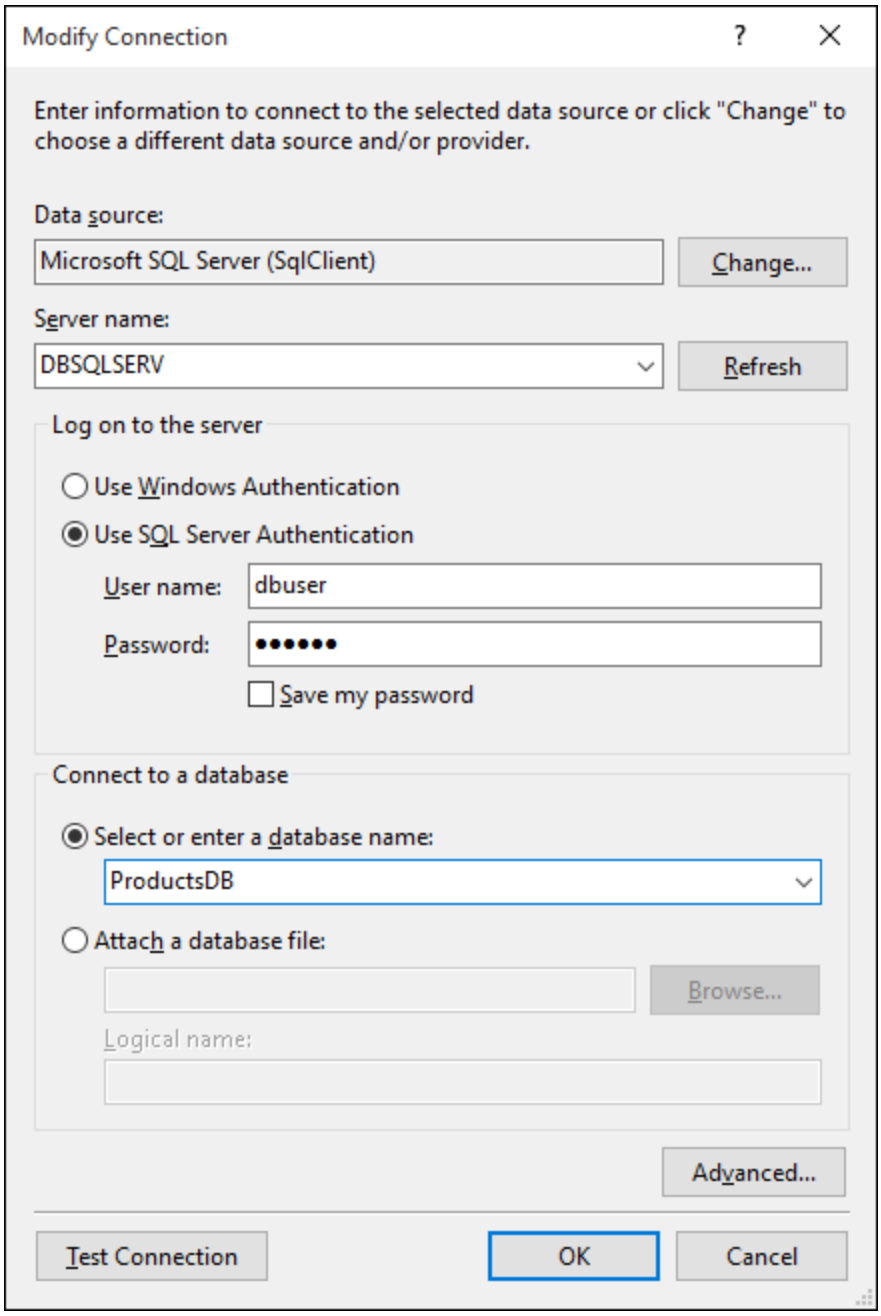
1. En el menú **Herramientas** haga clic en **Conectarse a una base de datos**.

 © 2019-2025 Altova GmbH

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**.




4. Introduzca el nombre de host del servidor, así como el nombre de usuario y la contraseña de la base de datos. En este ejemplo, nos estamos conectando a la base de datos `ProductsDB` en el servidor `DBSQLSERV`, utilizando la autenticación SQL Server.
5. 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 de menú **Vista | Explorador de servidores**. Para obtener la cadena de conexión de la base de datos, haga clic con el botón derecho en la conexión, en la ventana Explorador de servidores y seleccione **Propiedades**. Ahora aparece la cadena de conexión en la ventana Propiedades de

 © 2019-2025 Altova GmbH

Visual Studio. Recuerde que debe reemplazar los asteriscos con la contraseña antes de pegar la cadena en la casilla Cadena de conexión de MapForce.

#### 4.2.1.4.2 Ejemplo: cadenas de conexión ADO.NET

Para configurar una conexión ADO.NET, debe seleccionar un proveedor ADO.NET en el cuadro de diálogo de conexión a la BD e introducir una cadena de conexión (véase Configurar una conexión ADO.NET 191). A continuación se enumeran ejemplos de cadenas de conexión ADO.NET para varias bases de datos bajo el proveedor .NET al que corresponden.

*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=ServerAddress;User Id=usuario;Password=contraseña;
```

*Proveedor de datos .NET Framework para IBM i*
Este proveedor se instala con *IBM i Access Client Solutions - Windows Application Package*. Un ejemplo de cadena de conexión sería:

```
DataSource=ServerAddress;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 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

*Proveedor de datos .NET Framework 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 10.1.2 para .NET Framework 4.0*
Un ejemplo de cadena de conexión sería:

```
Database=PRODUCTS;UID=usuario;Password=contraseña;Server=localhost:50000;
```

**Nota:** Este proveedor suele instalarse con el paquete IBM DB2 Data Server Client. Si el proveedor no aparece en la lista de proveedores ADO.NET tras instalar el paquete IBM DB2 Data Server Client, consulte la siguiente 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) 232).

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=port)))
(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, |

 © 2019-2025 Altova GmbH

| Base de datos | Controlador | Notas |
|---|---|---|
| | | 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 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 funciones no disponibles a través de ODBC.

### Requisitos previos

- 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 determinará la ruta a la máquina virtual Java (JVM) desde las siguientes ubicaciones, en este orden: (i) la ruta de la JVM personalizada que puede haber configurado en las **Opciones** de la aplicación; ver Configuración de Java 1132; (ii) la ruta de la JVM que se encuentra en el registro de Windows; (iii) 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.
- Tener instalados los controladores JDBC del proveedor de la BD. Puede tratarse de controladores JDBC instalados como parte de la instalación de un cliente de base de datos o de bibliotecas JDBC compatibles (archivos `.jar`) que se descargan por separado. (ver Ejemplos de conexión a bases de datos 207).
- La variable de entorno `CLASSPATH` debe incluir la ruta al controlador JDBC (uno o varios archivos `.jar`) en su sistema operativo Windows. Algunos clientes de base de datos configuran esta variable automáticamente durante la instalación. Para más información consulte el apartado Configurar la variable CLASSPATH 199.

### Conectarse a un servidor SQL a través de JDBC con credenciales de Windows

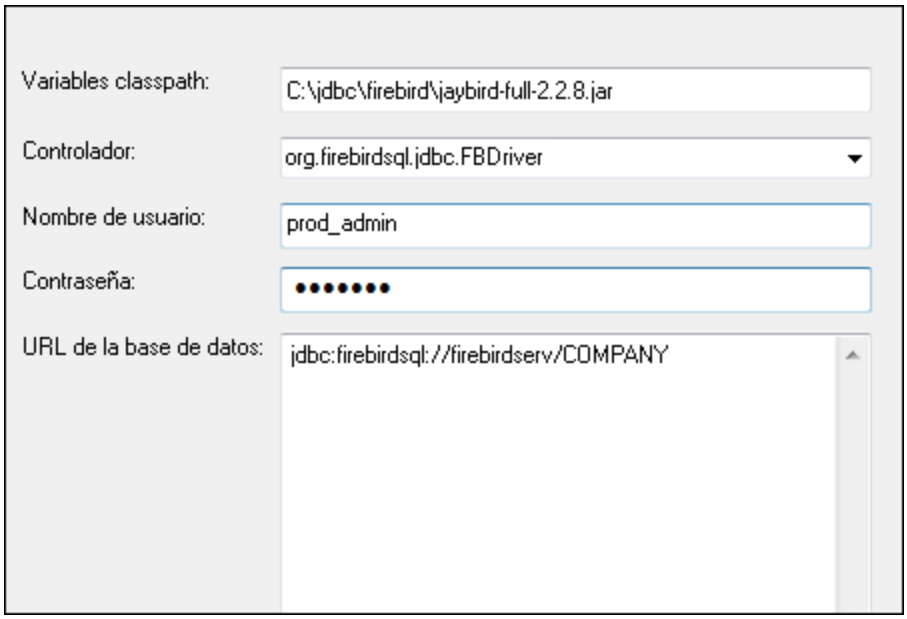
Si se conecta a un servidor SQL mediante JDBC con credenciales de Windows (seguridad integrada), tenga en cuenta lo siguiente:

- 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. Asegúrese de añadir a la variable `PATH` el que corresponde a su plataforma JDK. También debe asegurarse de que reinicia MapForce (o el programa que ejecute la asignación) 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 bases de datos (ver abajo)
  - desde la configuración del componente 251 de BD
  - si procede, editando la cadena de conexión a la BS en el código Java generado.

Para más información, consulte la documentación del *controlador JDBC de Microsoft para SQL Server*: https://docs.microsoft.com/es-es/sql/connect/jdbc/building-the-connection-url.

## Configurar una conexión JDBC

1. Inicie el asistente de conexión a bases de datos [179] y haga clic en Conexiones JDBC.
2. Si es necesario, introduzca una lista de rutas de archivos `.jar` separadas por punto y coma en el campo *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 [199]. Los controladores JDBC que se encuentran en las bibliotecas .jar de origen a las que se hace referencia a través del campo *Variables classpath* y de la variable de entorno CLASSPATH [199] del sistema se enumeran en la lista desplegable Controlador (ver paso siguiente).




3. En el campo *Controlador*, seleccione un controlador JDBC de la lista o introduzca un nombre de clase Java. La lista incluirá los controladores JDBC configurados a través del campo *Variables classpath* (ver más arriba) y la variable de entorno CLASSPATH [199].

> Las rutas de acceso del controlador JDBC definidas en la variable CLASSPATH, así como las rutas de acceso de los archivos .jar introducidas directamente en el cuadro de diálogo de conexión a la base de datos se envían a Java Virtual Machine (JVM). La JVM decide entonces qué controladores utilizar para establecer una conexión. Se recomienda hacer un seguimiento de las clases Java que se cargan en la JVM para evitar conflictos con el controlador JDBC y resultados inesperados al conectarse a la base de datos.

4. Introduzca el nombre de usuario y la contraseña de la base de datos en los campos correspondientes.
5. En el campo *URL de la base de datos*, introduzca la URL de conexión JDBC (cadena de conexión JDBC) en el formato específico de su tipo de base de datos. En la siguiente tabla puede ver la sintaxis de las cadenas de conexión JDBC para los tipos de base de datos más frecuentes.

 © 2019-2025 Altova GmbH

| Base de datos | URL de conexión JDBC |
|---|---|
| Firebird | `Jdbc:firebirdsql://<host>[:<puerto>]/<ruta o alias de la base de datos>` |
| IBM DB2 | `jdbc:db2://nombreHost:puerto/nombreBD` |
| IBM DB2 para i | `jdbc:as400://[host]` |
| IBM Informix | `jdbc:informix-sqli://nombreHost:puerto/nombreBD:INFORMIXSERVER=miservidor` |
| MariaDB | `jdbc:mariadb://nombreHost:puerto/nombreBaseDatos` |
| Microsoft SQL Server | `jdbc:sqlserver://nombreHost:puerto;nombreBaseDatos=nombre` |
| MySQL | `jdbc:mysql://nombreHost:puerto/nombreBaseDatos` |
| Oracle | `jdbc:oracle:thin:@nombreHost:puerto:SID`<br>`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` |

Tenga en cuenta que también son posibles variaciones sintácticas de los formatos enumerados anteriormente. Por ejemplo, la URL de la base de datos puede excluir el puerto o puede incluir el nombre de usuario y la contraseña de la base de datos. Consulte la documentación del proveedor de BD para obtener más información.

6. Haga clic en **Conectarse**.

#### 4.2.1.5.1 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 | Ejemplos 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;` |
| IBM Informix | `C:\Informix_JDBC_Driver\lib\ifxjdbc.jar;` |
| Microsoft SQL Server | `C:\Archivos de programa\Microsoft JDBC Driver 4.0 for SQL Server\sqljdbc_4.0\enu\sqljdbc.jar` |
| MariaDB | **<directorio de instalación>**`\mariadb-java-client-2.2.0.jar` |
| MySQL | **<directorio de instalación>**`\mysql-connector-java-`*version*`-bin.jar;` |
| Oracle | **ORACLE_HOME**`\jdbc\lib\ojdbc6.jar;` |
| Oracle (con XML DB) | **ORACLE_HOME**`\jdbc\lib\ojdbc6.jar;`**ORACLE_HOME**`\LIB\xmlparserv2.jar;`<br>**ORACLE_HOME**`\RDBMS\jlib\xdb.jar;` |
| PostgreSQL | **<directorio de instalación>**`\postgresql.jar` |
| Progress OpenEdge | `%DLC%\java\openedge.jar;%DLC%\java\pool.jar;`<br><br>Nota: Suponiendo que el SDK de Progress OpenEdge 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` |

Tenga en cuenta los siguientes aspectos:

- La opción `CLASSPATH` está disponible en la opción Variables de entorno de su sistema Windows. Incluya en la variable `CLASSPATH` la ruta donde se encuentra el controlador JDBC, separándola de otras rutas mediante un punto y coma.
- 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.


© 2019-2025 Altova GmbH

### 4.2.1.6 Conexiones ODBC

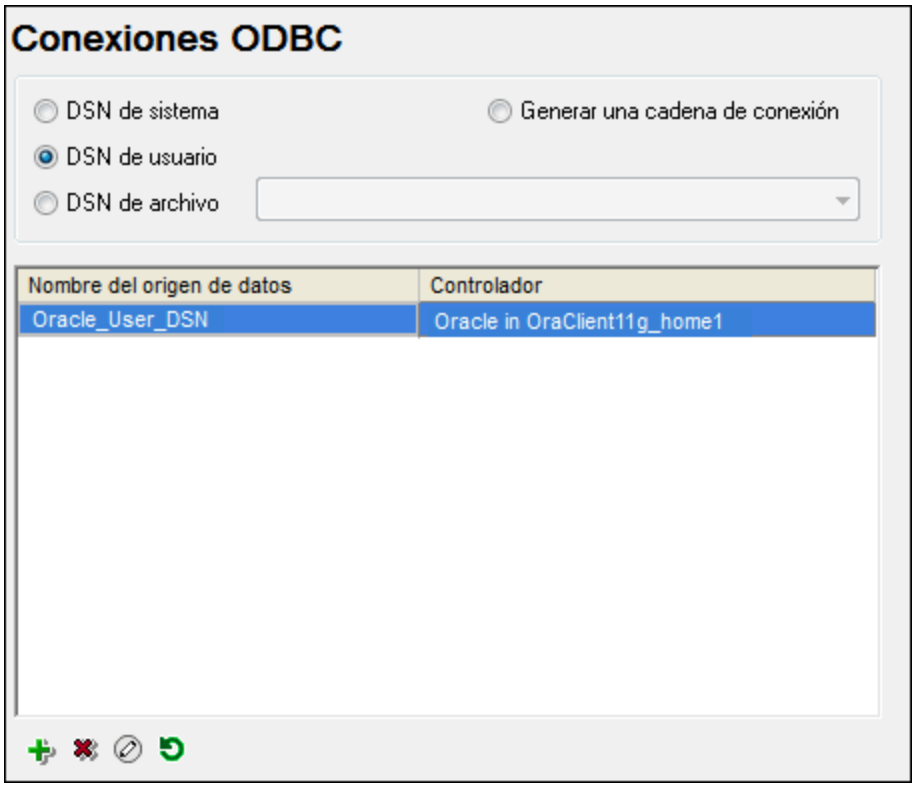
ODBC (Open Database Connectivity) es una tecnología de acceso a datos muy utilizada que le permite conectarse a una base de datos desde MapForce. Puede utilizarse como medio principal para conectarse a una base de datos o como alternativa a las conexiones nativas, OLE DB o JDBC.

Para conectarse a una base de datos por ODBC, primero es necesario contar con un DSN ODBC en el sistema operativo. Este paso no es necesario si ya se ha creado el DSN (p. ej. si lo ha creado otro usuario del sistema operativo). El DSN representa un medio uniforme para describir la conexión a la base de datos a cualquier aplicación cliente compatible con ODBC en el sistema operativo, incluyendo MapForce. Los DSN pueden ser de varios tipos:

- 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** (*imagen siguiente*).




Si no dispone de un DSN a la base de datos correspondiente, el asistente para la conexión a bases de datos de MapForce le ayudará a crearlo; no obstante, también puede crearlo directamente en su sistema operativo

Windows. 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 Ver los controladores disponibles 203).

## Conexión con un DSN nuevo

Siga estos pasos para conectarse con un nuevo DSN. Tenga en cuenta que para crear un DSN de sistema necesitará derechos de administrador en el sistema y debe ejecutar MapForce como administrador.

1. Inicie el asistente para la conexión de base de datos 179.
2. En el cuadro de diálogo de conexión a la base de datos, haga clic en **Conexiones ODBC**.
3. Seleccione un tipo de origen de datos (DSN de usuario, DSN sistema o DSN de archivo).
4. Ahora haga clic en **Agregar** .
5. Seleccione un controlador y haga clic en **DSN de usuario** o **DSN de sistema** (dependiendo del tipo de DSN que quiera crear). Si el controlador correspondiente a la BD no aparece en la lista, descárguelo e instálelo (consulte Resumen de controladores de BD 181).
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 indicar el nombre de host (o dirección IP) del servidor de la base de datos, así como el nombre de usuario y la contraseña de la base de datos. Puede haber otros parámetros de conexión opcionales; estos parámetros varían según los proveedores de bases de datos. Para obtener información detallada sobre los parámetros específicos de cada método de conexión, consulte la documentación del proveedor del controlador. Una vez creado, el DSN estará en la lista de nombres de orígenes de datos. Esto le permite reutilizar los detalles de conexión a la base de datos cada vez que quiera conectarse a ella. Tenga en cuenta que los DSN de usuario se añaden a la lista de DSN de usuario mientras que los DSN de sistema se añaden a la lista de DSN de sistema.

## Conexión con un DSN existente

Siga estos pasos para conectarse con un DSN existente.

1. Inicie el asistente para la conexión de base de datos 179.
2. Haga clic en **Conexiones ODBC**.
3. Elija el tipo de origen de datos existente (DSN de usuario, DSN sistema o DSN de archivo).
4. Haga clic en el registro DSN existente y después en **Conectarse**.

## Generar una cadena de conexión basada en un archivo .dsn existente

Siga estas instrucciones.

1. Inicie el asistente para la conexión de base de datos 179.
2. Haga clic en **Conexiones ODBC**.
3. Seleccione **Generar una cadena de conexión** y después haga clic en **Generar**.
4. Si quiere generar la cadena de conexión a partir de un DSN de archivo, haga clic en la pestaña *Origen de datos de archivo*. De lo contrario, haga clic en la pestaña *Origen de datos de la máquina*. (Los DSN de sistema y de usuario se conocen como orígenes de datos de la máquina).
5. Seleccione el archivo **.dsn** correspondiente y haga clic en **Aceptar**.

## Conectarse usando una cadena de conexión preparada

Siga estas instrucciones.

 © 2019-2025 Altova GmbH

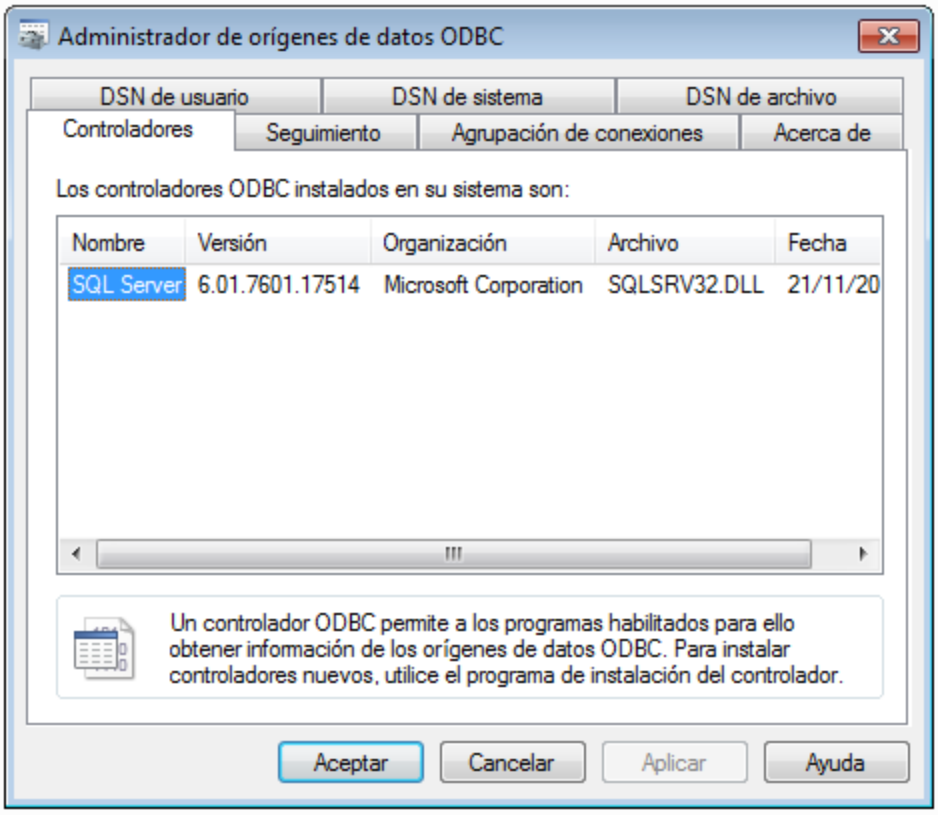
1. Inicie el asistente para la conexión de base de datos 179.
2. Haga clic en **Conexiones ODBC**.
3. Seleccione **Generar una cadena de conexión**.
4. Pegue la cadena de conexión en el cuadro de texto y haga clic en **Conectarse**.

### 4.2.1.6.1 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:

- La versión de 32 bits del archivo **`Odbcad32.exe`** se encuentra en el directorio **`C:\Windows\SysWoW64`** (suponiendo que **`C:`** sea la unidad de su sistema).
- La versión de 64 bits del archivo **`Odbcad32.exe`** se encuentra en el directorio **`C:\Windows\System32`**.

Todos los controladores de bases de datos de 32 bits instalados son visibles en la versión de 32 bits del Administrador de orígenes de datos ODBC (*imagen siguiente*), mientras que los controladores de 64 bits lo son en la versión de 64 bits. Por lo tanto, asegúrese de comprobar los controladores de la base de datos desde la versión correspondiente del Administrador de orígenes de datos ODBC.



Si el controlador de su base de datos de destino no está en la lista o si desea añadir un controlador alternativo, deberá descargarlo del proveedor de la base de datos (ver Resumen de controladores de base de datos 181). Una vez que el controlador ODBC esté disponible en su sistema, podrá crear conexiones ODBC con él (consulte Configurar una conexión ODBC 201).

## 4.2.1.7 Conexiones SQLite

SQLite es un tipo de base de datos autocontenida y basada en archivos,por lo que es una BD ideal cuando la portabilidad y la facilidad de configuración son importantes. 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 se admite el tiempo de espera de ejecución de instrucciones para bases de datos SQLite.
- Las tablas de búsqueda de texto completo no son compatibles.
- SQLite admite valores de diferentes tipos de datos en cada fila de una tabla. Todos los valores procesados deben ser compatibles con el tipo de columna declarado; por lo tanto, pueden recuperarse valores inesperados y producirse errores en tiempo de ejecución si su base de datos SQLite tiene valores de fila que no coinciden con el tipo de columna declarado.
- Si su 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 DatabaseSpyl o puede descargar la shell de comandos de SQLite del sitio web oficial y crear el archivo desde la línea de comandos (consulteEjemplo: asignar datos XML a SQLite). Para una referencia completa de la sintaxis de los comandos SQLite, consulte la documentación oficial 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 documentación de SQLite.

### 4.2.1.7.1 Conectarse a una BD SQLite existente

Siga estos pasos para conectarse a una base de datos SQLite existente:

1. Ejecute el asistente de conexión a la base de datos (consulte Iniciar el asistente para la conexión de base de datos 179).
2. Seleccione **SQLite** y luego 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. Haga clic en **Conectarse**.

 © 2019-2025 Altova GmbH

## 4.2.1.8 Conexiones nativas

Las conexiones nativas son conexiones directas a los protocolos y controladores de red propios de una base de datos. Por lo tanto, no es necesario instalar controladores adicionales. Las conexiones nativas ofrecen el método más eficaz de interactuar con la base de datos y total compatibilidad.

Si tiene pensado implementar archivos para ejecutarlos en servidores Linux o macOS, no será necesario instalar controladores en el servidor de destino.

Puede configurar conexiones nativas a las siguientes bases de datos:

- MariaDB
- MySQL
- SQLite
- PostgreSQL

### Configurar una conexión

Siga estos pasos para configurar una conexión nativa:

1. Inicie el asistente para la conexión de base de datos 179.
2. Seleccione la BD a la que desea conectarse.
3. En el cuadro de diálogo que aparece, introduzca los detalles relevantes de la conexión, como el host (por ejemplo, *localhost*), opcionalmente el puerto (normalmente 5432), el modo SSL en el caso de MySQL, el nombre de la base de datos, el nombre de usuario y la contraseña en las casillas correspondientes.
4. Haga clic en **Conectarse**.

### Conexiones SQLite

Para ver información más detallada sobre las conexiones SQLite, consulte el apartado Conexión SQLite 204.

### Notas para PostrgreSQL

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**).

## 4.2.1.9 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

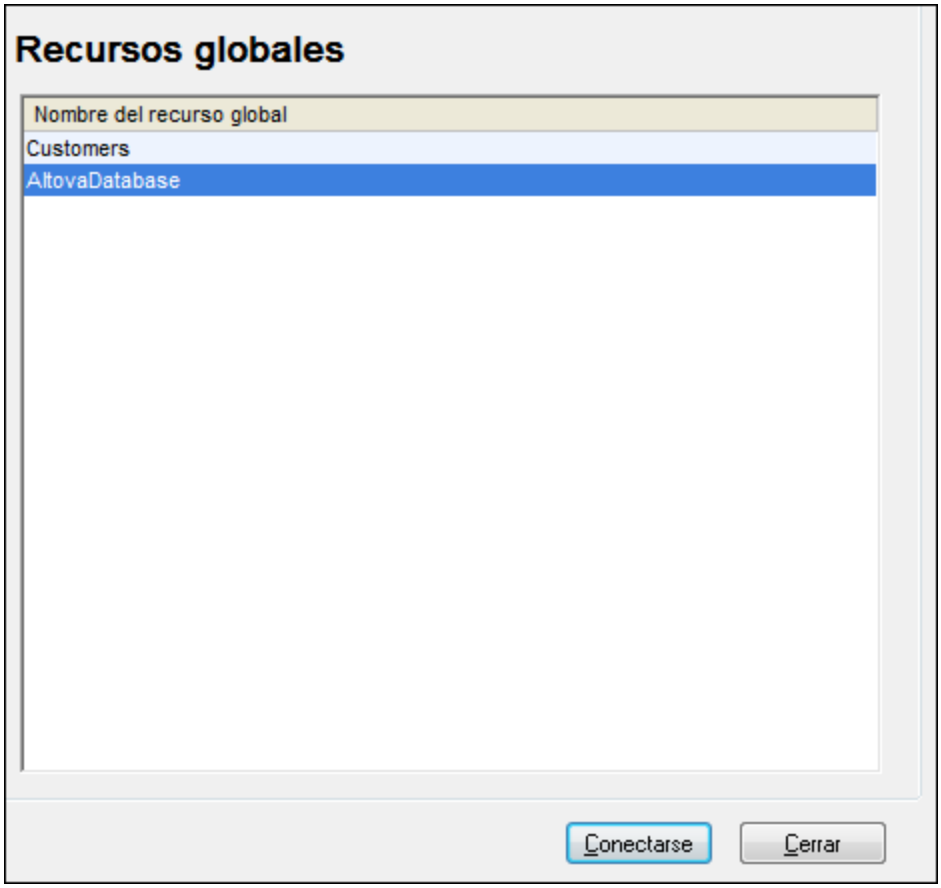
Para crear una BD como recurso global siga estos pasos:

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 179.
5. Use el asistente para agregar una conexión de base de datos como hemos explicado.

### 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*).




 © 2019-2025 Altova GmbH

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[195]. Para ver cómo establecer una conexión nativa a PostgreSQL y SQLite (consulte respectivamente Configurar una conexión PostgreSQL[205] y Configurar una conexión SQLite[204]).

Tenga en cuenta los siguientes aspectos:

- 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 seleccionado, el software cliente de BD (en su caso) y cualquier parámetro de conectividad adicional que haya configurado fuera de MapForce.

### 4.2.1.10.1 Firebird (JDBC)

Este ejemplo muestra cómo conectarse a una base de datos Firebird a través de JDBC.

#### Requisitos previos

- 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 determinará la ruta a la máquina virtual Java (JVM) desde las siguientes ubicaciones, en este orden: (i) la ruta de la JVM personalizada que puede haber configurado en las **Opciones** de la aplicación; ver Configuración de Java[1132]; (ii) la ruta de la JVM que se encuentra en el registro de Windows; (iii) 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.
- 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 5.0.6*.
- Disponer de los datos de conexión: host, puerto, ruta de acceso de la base de datos, nombre o alias, nombre de usuario y contraseña.

#### Conexión

1. Inicie el asistente para la conexión a bases de datos[179] y haga clic en **Conexiones JDBC** (*consulte Conexión JDBC*[197] *para ver una imagen del cuadro de diálogo*).
2. En el campo *Variables Classpath* introduzca la ruta de acceso del archivo `.jar` que proporciona al 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. Si ha añadido la ruta del archivo a la variable CLASSPATH[199] del sistema, puede dejar este campo vacío.
3. En el campo *Controlador*, seleccione el controlador adecuado. Los controladores pertinentes solo estarán disponibles si hay una ruta de archivo `.jar` válida en el campo *Variables Classpath* o en la

variable de entorno CLASSPATH.
4. Introduzca el nombre de usuario y la contraseña para la base de datos en los campos correspondientes.
5. Introduzca la cadena de conexión JDBC en el campo *URL de la base de datos* según el patrón de la tabla siguiente, sustituyendo los valores resaltados por los que corresponden a su servidor de base de datos.
6. Haga clic en **Conectarse**.

*Detalles de conexión del ejemplo con Firebird*

| Campo | Valor |
|---|---|
| Classpaths | `C:\jdbc\firebird\jaybird-full-5.0.6.jar` |
| Controlador | *org.firebirdsql.jdbc.FBDriver* |
| URL de la base de datos | `Jdbc:firebirdsql://<host>[:<port>]/<ruta<host>[:<port>]/<ruta de acceso o alias de la BD>` |

### 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 previos

- 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 utiliza el controlador ODBC de Firebird versión 2.0.3.154 descargado de la página web de Firebird.
- El cliente Firebird debe estar instalado en su sistema operativo. Tenga en cuenta que no existe un instalador independiente para el cliente Firebird 2.5.4; el cliente forma parte del paquete de instalación del servidor Firebird. Puede descargar el paquete de instalación del servidor Firebird de la página web de Firebird. Busque *el instalador ejecutable de Windows para Superclassic/Classic o Superserver*. Para instalar los archivos del cliente solamente elija la opción *Minimum client install - no server, no tools* del asistente para la instalación.
- Disponer de estos datos de conexión: nombre de host o dirección IP, ruta de acceso (o alias) de la BD en el servidor, nombre de usuario y contraseña.

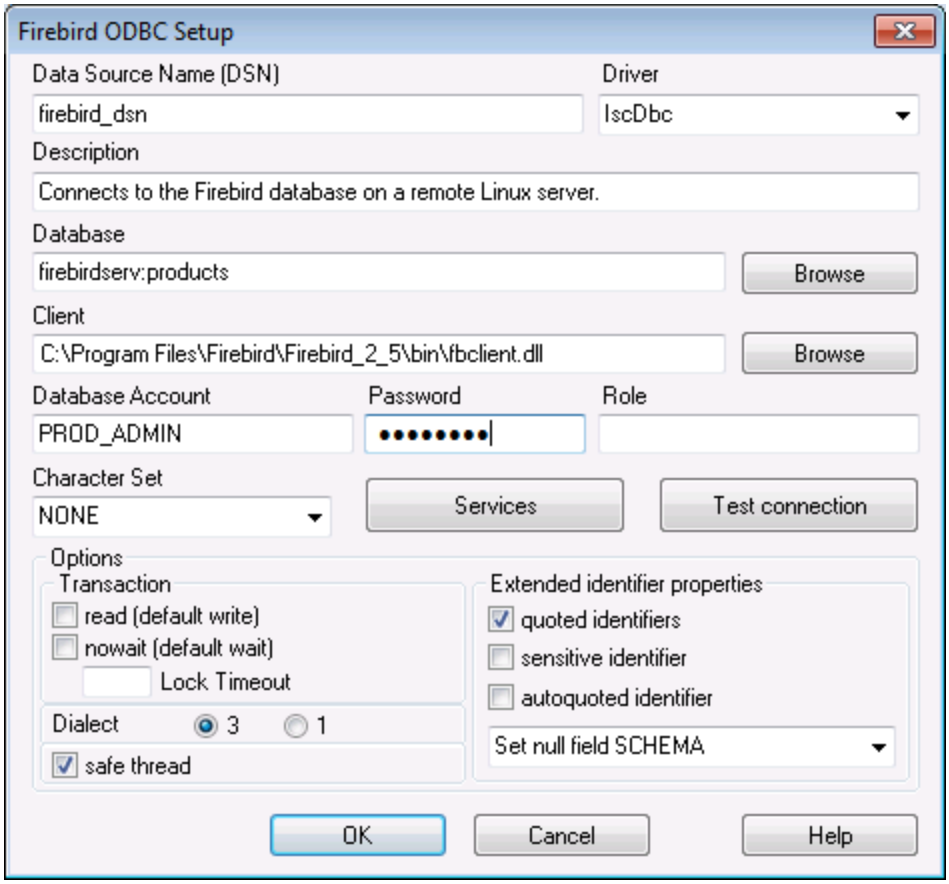
**Importante:** La plataforma del controlador ODBC de Firebird y del cliente (de 32 o 64 bits) debe coincidir con la plataforma de MapForce. Además, la versión del cliente Firebird debe corresponder a la versión del servidor Firebird al que se está conectando.

#### Conexión

1. Inicie el asistente de conexión a bases de datos 179 y haga clic en **Conexiones ODBC**.
2. Seleccione *DSN de usuario* (o *DSN de sistema* si tiene privilegios administrativos) y luego haga clic en **Crear DSN nuevo**.

 © 2019-2025 Altova GmbH

3. 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 está disponible en la lista, asegúrese de que está instalado en su sistema operativo (véase Ver los controladores ODBC disponibles 203).




4. 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`:<br><br>`firebirdserv:products`<br><br>Usamos el alias de la BD porque damos por hecho que, en el lado del servidor, el administrador de la base de datos ha configurado el alias *products* para que apunte al archivo de base de datos Firebird (.fdb) real en el servidor (consulte la documentación de Firebird para obtener más información).<br><br>También puede utilizar la dirección IP del servidor en lugar del nombre de host y una ruta en lugar de un alias; por lo tanto, cualquiera de las siguientes cadenas de conexión de ejemplo son válidas:<br><br>`firebirdserver:/var/Firebird/databases/butterflies.fdb`<br>`127.0.0.1:D:\Misc\Lenders.fdb`<br><br>Si la base de datos se encuentra en el equipo Windows local, haga clic en **Examinar** y seleccione directamente el archivo de base de datos Firebird (.fdb). |
| *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. |

5. Para terminar haga clic en **Aceptar**.

### 4.2.1.10.3 IBM DB2 (JDBC)

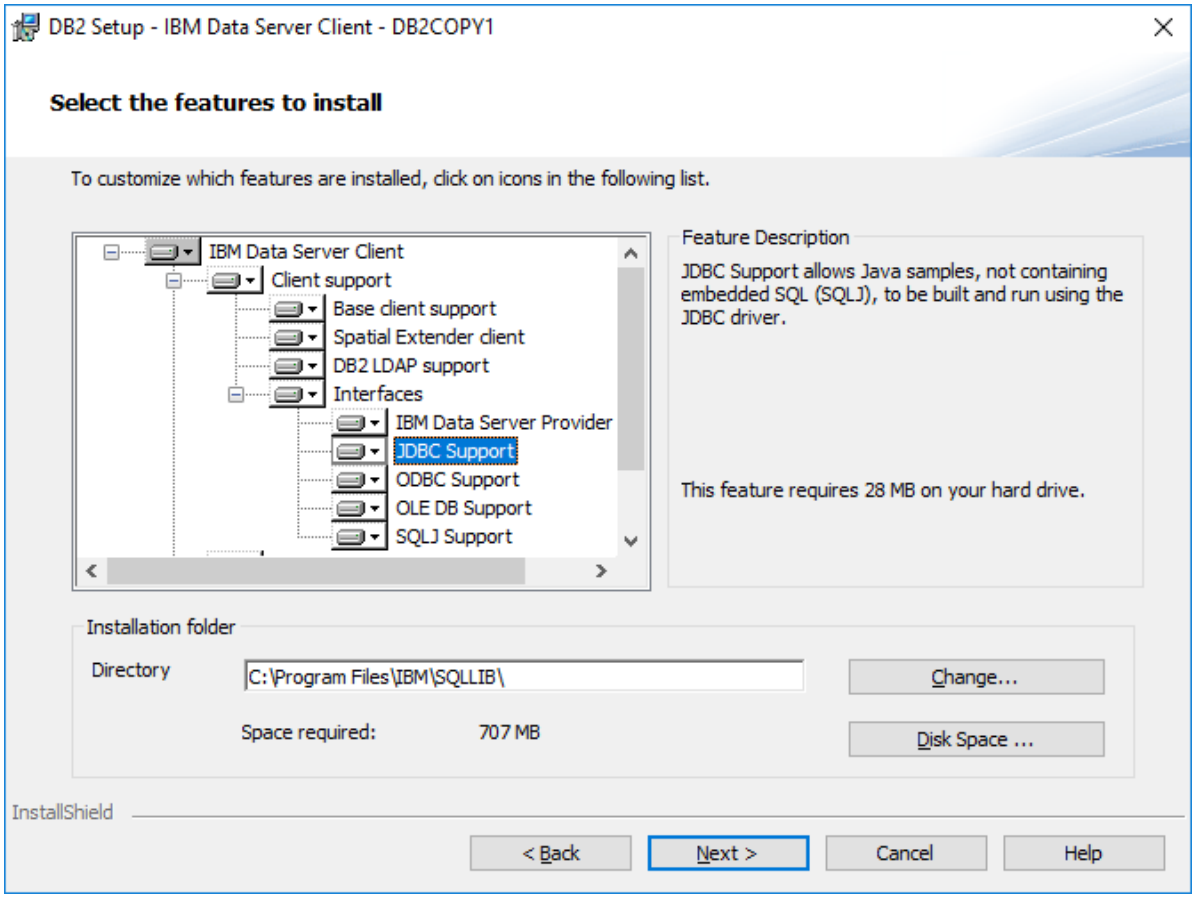
Este ejemplo explica cómo conectarse a una base de datos IBM DB2 mediante JDBC.

#### Requisitos previos

- 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 determinará la ruta a la máquina virtual Java (JVM) desde las siguientes ubicaciones, en este orden: (i) la ruta de la JVM personalizada que puede haber configurado en las **Opciones** de la aplicación; ver Configuración de Java 1132; (ii) la ruta de la JVM que se encuentra en el registro de Windows; (iii) 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.

 © 2019-2025 Altova GmbH

- 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. 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 *Typical* o seleccione esa opción en el asistente de instalación.




  Si no ha modificado la ruta predeterminada de la instalación, tras la instalación los archivos `.jar` necesarios están en el directorio `C:\Archivos de programa\IBM\SQLLIB\java`.
- Disponer de los datos de conexión: host, puerto, ruta de acceso de la base de datos, nombre o alias, nombre de usuario y contraseña.

## Conexión

1. Inicie el asistente para la conexión a bases de datos 179 y haga clic en **Conexiones JDBC** (*consulte Conexión JDBC* 197 *para ver una imagen del cuadro de diálogo*).
2. En el campo *Variables Classpath* introduzca la ruta de acceso del archivo `.jar` que proporciona al 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. Si ha añadido la ruta del archivo a la variable CLASSPATH 199 del sistema, puede dejar este campo vacío.
3. En el campo *Controlador*, seleccione el controlador adecuado. Los controladores pertinentes solo estarán disponibles si hay una ruta de archivo `.jar` válida en el campo *Variables Classpath* o en la variable de entorno `CLASSPATH`.
4. Introduzca el nombre de usuario y la contraseña para la base de datos en los campos correspondientes.
5. Introduzca la cadena de conexión JDBC en el campo *URL de la base de datos* según el patrón de la

tabla siguiente, sustituyendo los valores resaltados por los que corresponden a su servidor de base de datos.

6. Haga clic en **Conectarse**.

*Detalles de conexión del ejemplo IBM DB2*

| Campo | Valor |
|---|---|
| Classpaths | `C:\Archivos de programa\IBM\SQLLIB\java\db2jcc.jar` |
| Controlador | *com.ibm.db2.jcc.DB2Driver* |
| URL de la base de datos | `jdbc:db2://`**`nombreHost`**`:`**`puerto`**`/`**`nombreBD`** |

### 4.2.1.10.4 IBM DB2 (ODBC)

Este ejemplo explica cómo conectarse a una base de datos IBM DB2 mediante ODBC.

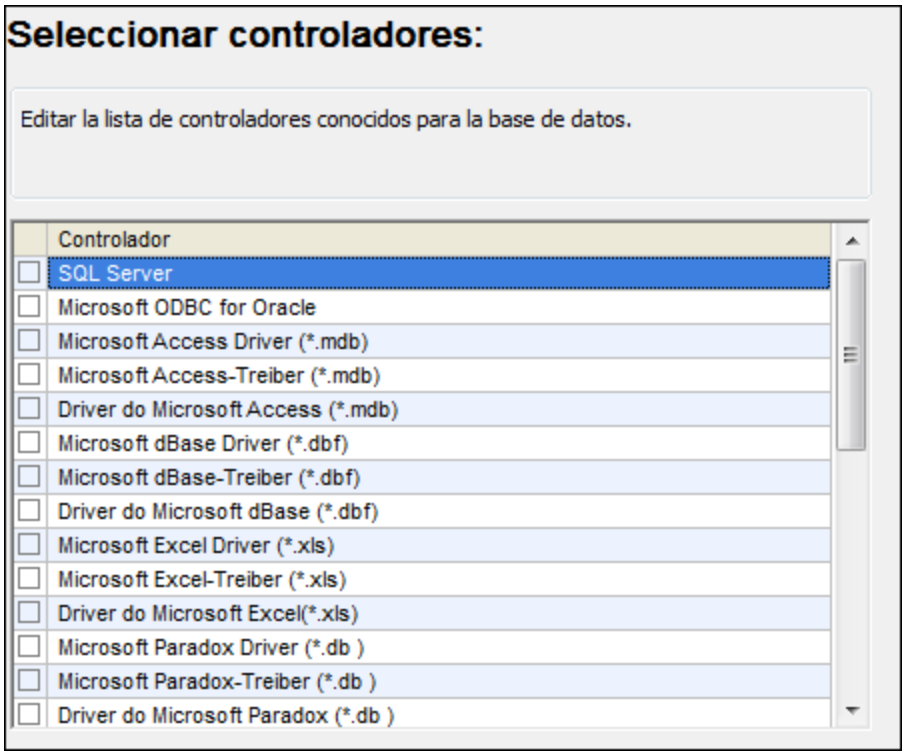
#### Requisitos previos

- El cliente IBM Data Server está instalado y configurado en el sistema (en este ejemplo usamos IBM Data Server Client 9.7). Para obtener instrucciones de instalación, consulte la documentación suministrada con el software IBM DB2. Tras instalar el cliente IBM Data Server, compruebe si los controladores ODBC están disponibles en su máquina (consulte Ver los controladores ODBC disponibles). 203
- Crear un alias de base de datos. Hay varias maneras de hacer esto:
  - Desde el asistente de configuración de IBM DB2
  - Desde el procesador de la línea de comandos de IBM DB2
  - Desde 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.

#### Conexión

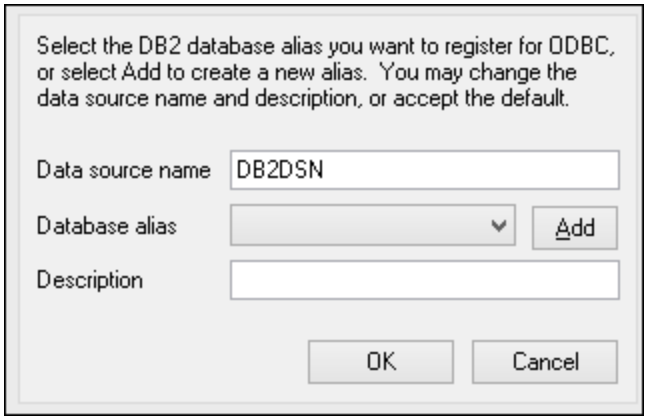
1. Inicie el asistente para la conexión de base de datos 179, seleccione **Asistente para la conexión** y después *IBM DB2 (ODBC/JDBC)*. Haga clic en **Siguiente**.
2. Seleccione *ODBC* y haga clic en **Siguiente**.
3. Si se le pide que edite la lista de controladores conocidos para la base de datos, seleccione los controladores de base de datos que corresponden a IBM DB2 (ver *Requisitos previos* más arriba) y haga clic en **Siguiente**.

 © 2019-2025 Altova GmbH



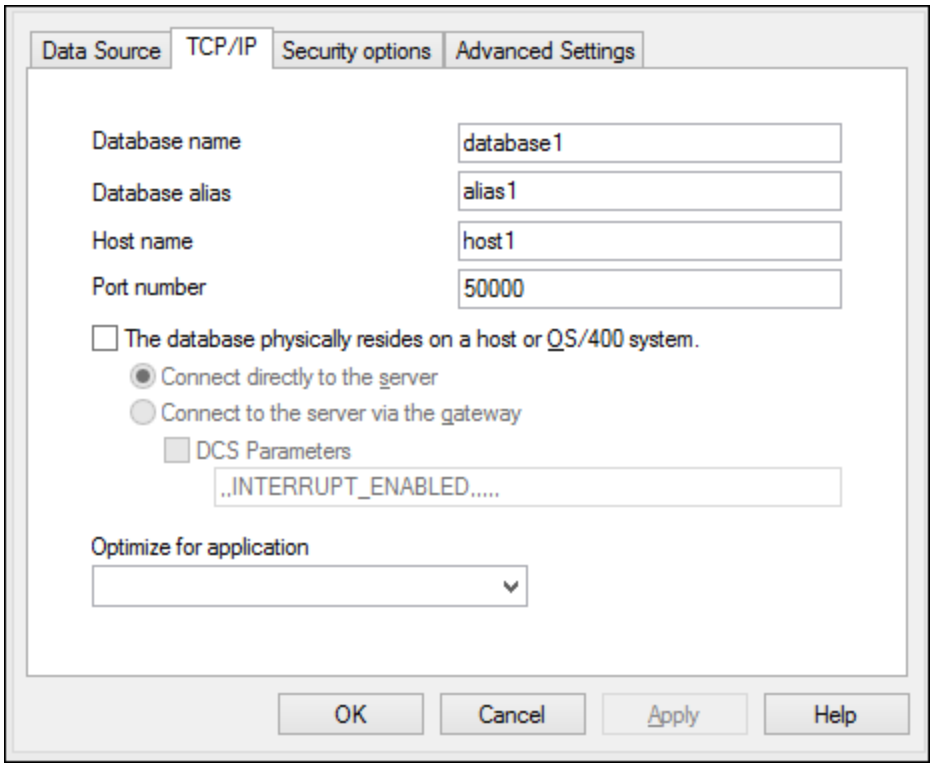

4. Seleccione el controlador que desea en la lista y haga clic en **Conectarse**. (Para editar la lista de controladores disponibles haga clic en **Editar controladores** y active/desactive los controladores que desea agregar o eliminar.)

5. Introduzca un nombre de origen de datos (en la imagen siguiente, DB2DSN) y, a continuación, haga clic en **Agregar**.




6. En la pestaña *Origen de datos* introduzca el nombre de usuario y la contraseña para 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 del host y el número de puerto y, a continuación, haga clic en **Aceptar**.

 © 2019-2025 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 para i mediante JDBC.

## Requisitos previos

- 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 determinará la ruta a la máquina virtual Java (JVM) desde las siguientes ubicaciones, en este orden: (i) la ruta de la JVM personalizada que puede haber configurado en las **Opciones** de la aplicación; ver Configuración de Java 1132; (ii) la ruta de la JVM que se encuentra en el registro de Windows; (iii) 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.
- 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 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**.
- Disponer de los datos de conexión: host, puerto, ruta de acceso de la base de datos, nombre o alias, nombre de usuario y contraseña.

## Conexión

1. Inicie el asistente para la conexión a bases de datos[179] y haga clic en **Conexiones JDBC** (*consulte Conexión JDBC*[197] *para ver una imagen del cuadro de diálogo*).
2. En el campo *Variables Classpath* introduzca la ruta de acceso del archivo `.jar` que proporciona al 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. Si ha añadido la ruta del archivo a la variable CLASSPATH[199] del sistema, puede dejar este campo vacío.
3. En el campo *Controlador*, seleccione el controlador adecuado. Los controladores pertinentes solo estarán disponibles si hay una ruta de archivo `.jar` válida en el campo *Variables Classpath* o en la variable de entorno `CLASSPATH`.
4. Introduzca el nombre de usuario y la contraseña para la base de datos en los campos correspondientes.
5. Introduzca la cadena de conexión JDBC en el campo *URL de la base de datos* según el patrón de la tabla siguiente, sustituyendo los valores resaltados por los que corresponden a su servidor de base de datos.
6. Haga clic en **Conectarse**.

*Detalles de conexión del ejemplo IBM DB2 para i*

| Campo | Valor |
|---|---|
| Classpaths | `C:\jdbc\jtopen_9_8\jt400.jar` |
| Controlador | *com.ibm.as400.access.AS400JDBCDriver* |
| URL de la base de datos | `Jdbc:as400://host[:<puerto>]/<ruta` **`host`** `[:` **`<puerto>`** `]/` **`<ruta o alias de la base de datos>`** |

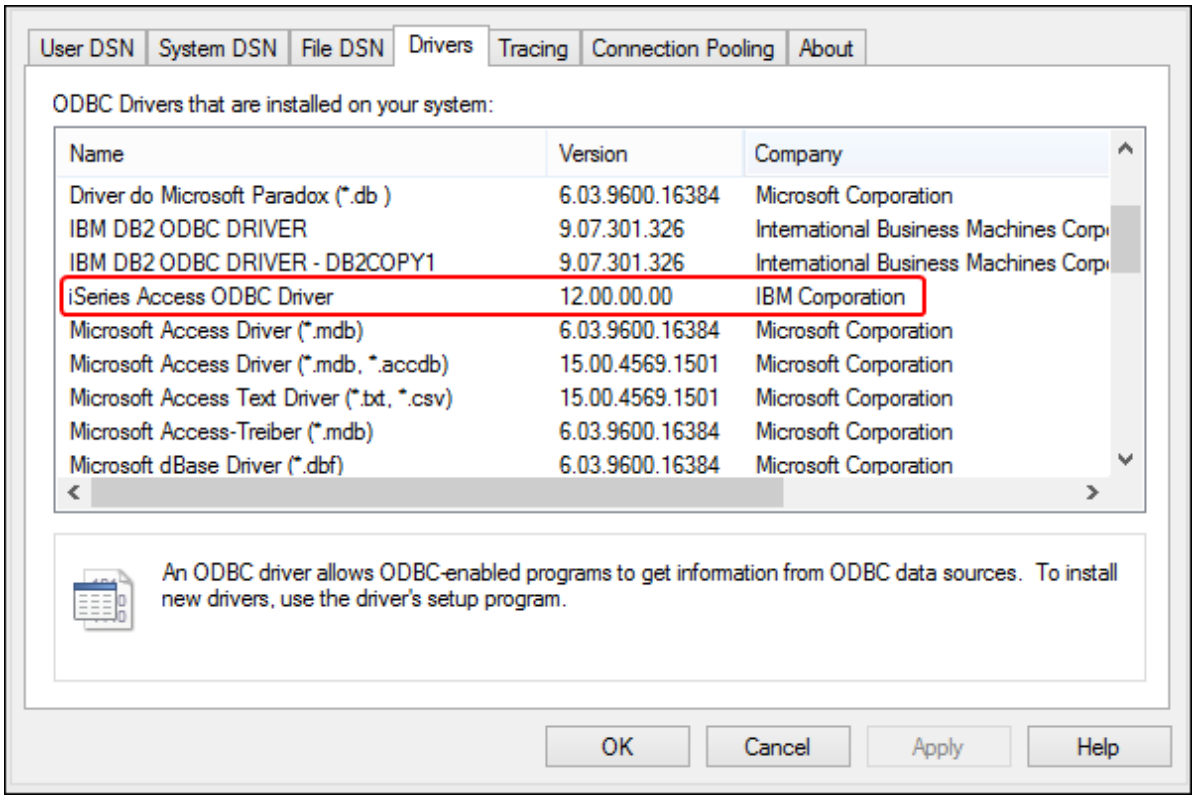
## 4.2.1.10.6 IBM DB2 para i (ODBC)

Este ejemplo explica cómo conectarse a una base de datos *IBM DB2 para i* mediante ODBC.

## Requisitos previos

- *IBM System i Access para Windows* debe estar instalado en su sistema operativo. Para obtener instrucciones de instalación, consulte la documentación suministrada con el software *IBM DB2 para i*. Tras la instalación, compruebe si el controlador ODBC está disponible en su equipo[203].

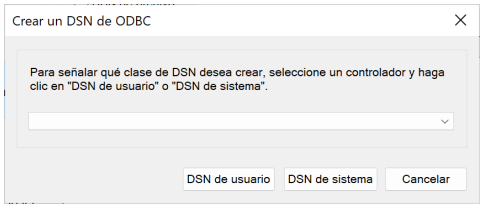

© 2019-2025 Altova GmbH




- Dispone de los siguientes datos de conexión a la BD: la dirección IP del servidor de la BD y el nombre de usuario y la contraseña de la BD.
- Ejecute System i Navigator y siga el asistente para crear una nueva conexión. Cuando se le pida que especifique un sistema, introduzca la dirección IP del servidor de la BD. Tras crear la conexión, verifíquela haciendo clic en ella y seleccionando **Archivo | Diagnóstico | Verificar conexión**.

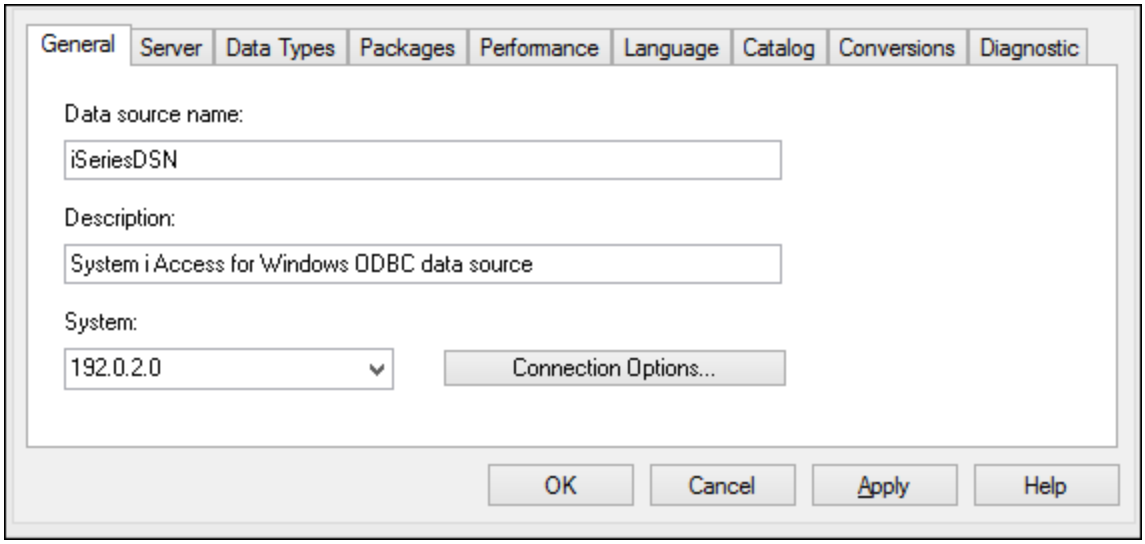
## Conexión

1. Inicie el asistente de conexión a bases de datos 179 y haga clic en **Conexiones ODBC**.
2. Haga clic en *DSN de usuario* y en **Crear un DSN nuevo** .
3. Seleccione el *controlador iSeries Access ODBC* en la lista y haga clic en **DSN de usuario** (o **DSN de sistema** si procede).

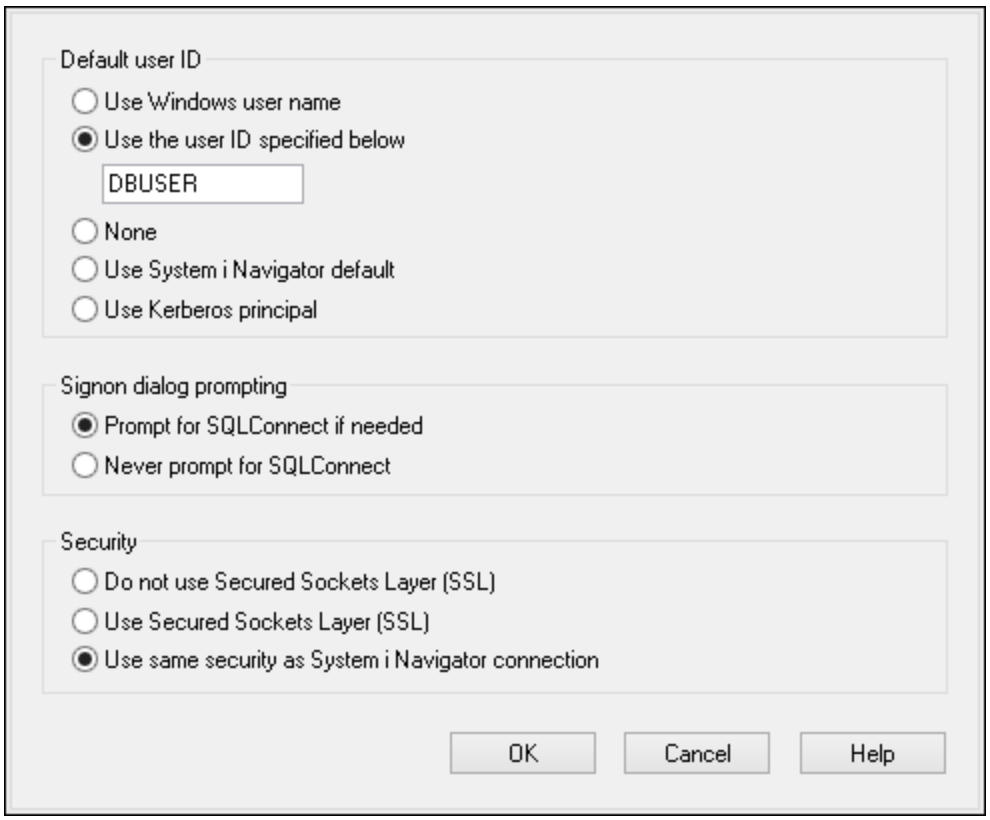



4. Introduzca el DSN y seleccione la conexión del cuadro combinado Sistema. En este ejemplo, el nombre del origen de datos es iSeriesDSN y el sistema es 192.0.2.0.Tenga en cuenta que cuando se añade un origen de datos ODBC para una base de datos IBM DB2 para i, se establece un indicador por defecto que habilita los tiempos de espera de las consultas. 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).




5. Haga clic en **Opciones de conexión**, seleccione *Utilizar el ID de usuario especificado a continuación* e introduzca el nombre del usuario de la base de datos (en este ejemplo, *DBUSER*).




6. Haga clic en **Aceptar**. El origen de datos nuevo aparece ahora en la lista de DSN.
7. Haga clic en **Conectarse**.
8. Introduzca el nombre de usuario y la contraseña de la BD cuando sea necesario y después haga clic en **Aceptar**.

 © 2019-2025 Altova GmbH

### 4.2.1.10.7 IBM Informix (JDBC)

Este ejemplo explica cómo conectarse a una base de datos IBM Informix mediante JDBC.

## Requisitos previos

- 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 determinará la ruta a la máquina virtual Java (JVM) desde las siguientes ubicaciones, en este orden: (i) la ruta de la JVM personalizada que puede haber configurado en las **Opciones** de la aplicación; ver Configuración de Java[1132]; (ii) la ruta de la JVM que se encuentra en el registro de Windows; (iii) 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.
- 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.
- Disponer de los datos de conexión: host, nombre del servidor Informix, base de datos, puerto, nombre de usuario y contraseña.

## Conexión

1. Inicie el asistente para la conexión a bases de datos[179] y haga clic en **Conexiones JDBC** (*consulte Conexión JDBC*[197] *para ver una imagen del cuadro de diálogo*).
2. En el campo *Variables Classpath* introduzca la ruta de acceso del archivo `.jar` que proporciona al 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. Si ha añadido la ruta del archivo a la variable CLASSPATH[199] del sistema, puede dejar este campo vacío.
3. En el campo *Controlador*, seleccione el controlador adecuado. Los controladores pertinentes solo estarán disponibles si hay una ruta de archivo `.jar` válida en el campo *Variables Classpath* o en la variable de entorno `CLASSPATH`.
4. Introduzca el nombre de usuario y la contraseña para la base de datos en los campos correspondientes.
5. Introduzca la cadena de conexión JDBC en el campo *URL de la base de datos* según el patrón de la tabla siguiente, sustituyendo los valores resaltados por los que corresponden a su servidor de base de datos.
6. Haga clic en **Conectarse**.

*Detalles de conexión del ejemplo IBM Informix*

| Campo | Valor |
|---|---|
| Classpaths | `C:\Informix_JDBC_Driver\lib\ifxjdbc.jar` |
| Controlador | *com.informix.jdbc.IfxDriver* |
| URL de la base de datos | `jdbc:informix-sqli://`**`nombreHost`**`:`**`puerto`**`/`**`nombreBD`**`:INFORMIXSERVER=`**`miservidor`**`;` |

### 4.2.1.10.8 MariaDB (ODBC)

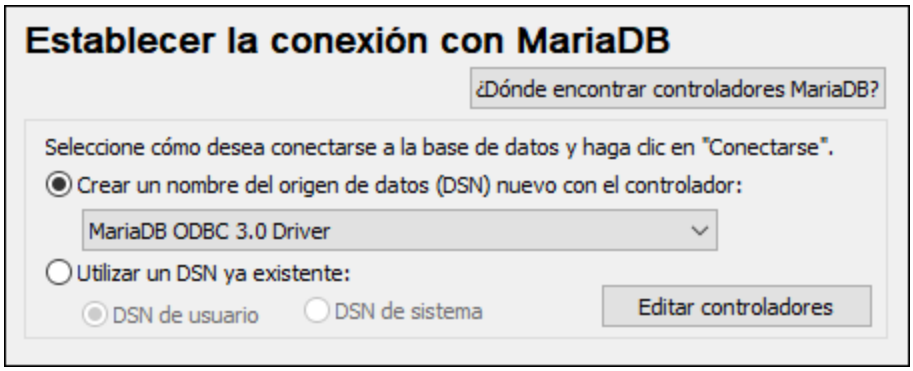
Este ejemplo explica cómo conectarse a un servidor de base de datos MariaDB mediante ODBC.

## Requisitos previos

- El conector ODBC de MariaDB debe estar instalado.
- Disponer de los datos de conexión: host, base de datos, puerto, nombre de usuario y contraseña.

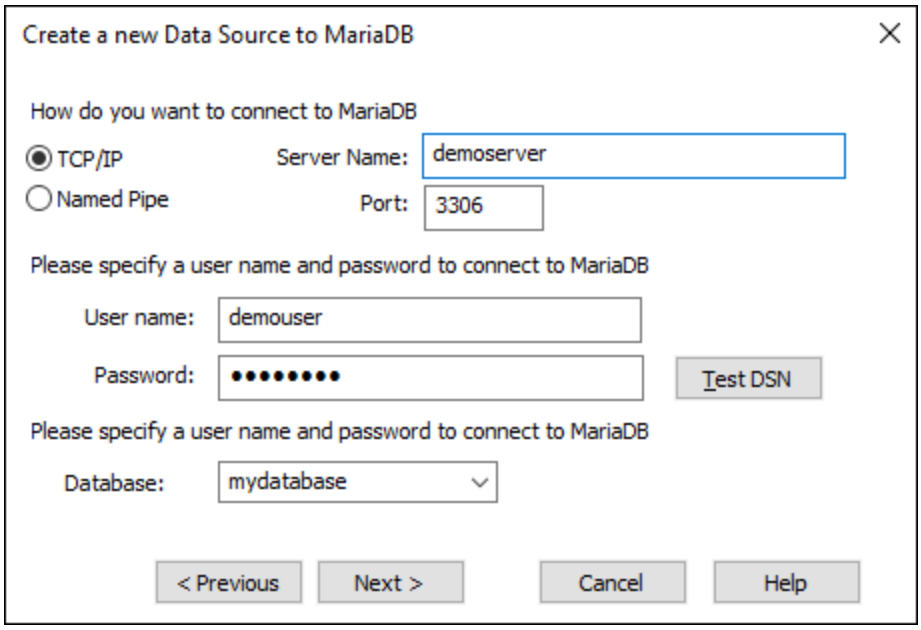
## Conexión

1. Inicie el asistente para la conexión de base de datos 179, seleccione **Asistente para la conexión** y después *MariaDB (ODBC/Native)*. Haga clic en **Siguiente**.
2. Seleccione *Crear un nombre del origen de datos (DSN) nuevo con el controlador* y elija el controlador *MariaDB ODBC 3.0 Driver*. (Si no hay ningún controlador disponible en la lista, haga clic en **Editar controladores** y seleccione cualquier controlador MariaDB disponible de la lista de controladores ODBC instalados en su sistema). Haga clic en **Conectarse** para iniciar el asistente para el nuevo origen de datos.




3. En la primera pantalla del asistente, introduzca un nombre para el origen de datos, opcionalmente, una descripción que le ayude a identificar este origen de datos ODBC en el futuro, y haga clic en **Aceptar**.
4. En la siguiente pantalla (imagen siguiente), rellene las credenciales de conexión a la base de datos (servidor TCP/IP, nombre de usuario y contraseña), seleccione una base de datos y haga clic en **Probar DSN**.


© 2019-2025 Altova GmbH




5. Si la conexión se establece correctamente, aparecerá un mensaje a tal efecto. Haga clic en **Siguiente** y siga los pasos del asistente hasta el final. Puede que sean necesarios más parámetros (p. ej. certificados SSL si se conecta a MariaDB a través de una conexión segura).

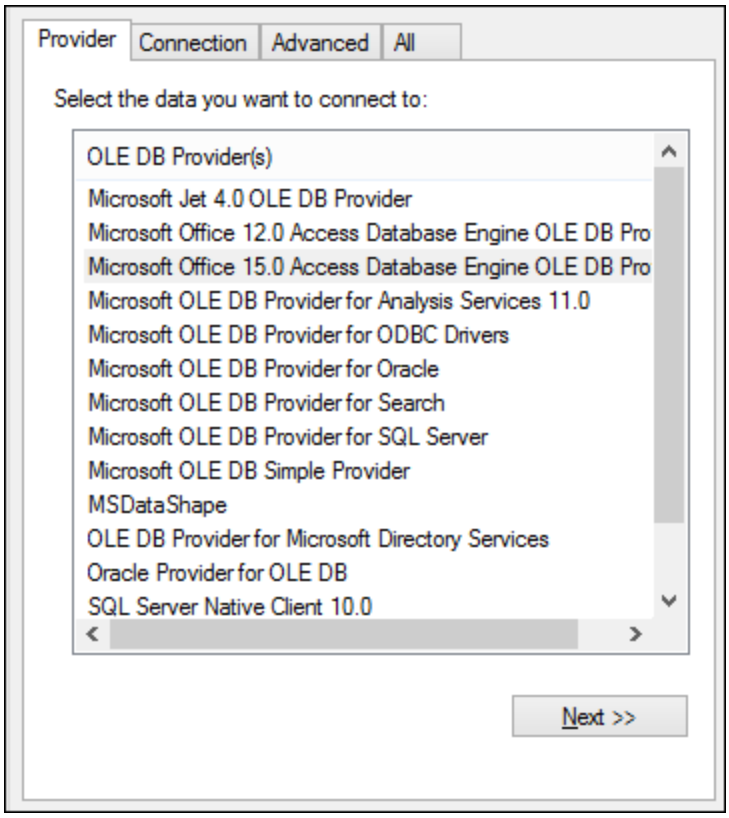
**Nota:** si el servidor de BD es remoto, el administrador del servidor debe configurarlo para que acepte conexiones remotas desde la dirección IP de su equipo.

### 4.2.1.10.9 Microsoft Access (ADO)

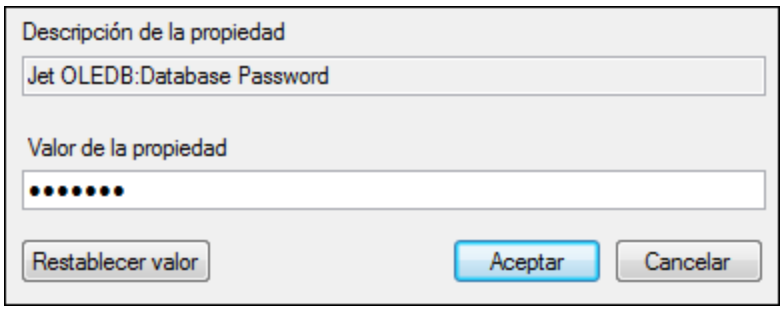
Lo más sencillo para conectarse a una base de datos Microsoft Access es seguir las instrucciones del asistente para la conexión de base de datos y buscar el archivo de base de datos, como se muestra ene l apartado Conectarse a una base de datos de Microsoft Access 188. También puede configurar explícitamente una conexión ADO, como muestra este ejemplo. Esta segunda opción se recomienda si la base de datos está protegida con contraseña. (También es posible conectarse a Microsoft Access a través de una conexión ODBC, pero tiene limitaciones, por lo que es mejor evitarlo).

#### Conexión

1. Inicie el asistente para la conexión de base de datos 179.
2. Haga clic en **Conexiones ADO** y, a continuación, en **Generar**.
3. En el cuadro de diálogo que aparece (*imagen siguiente*), vaya a la pestaña *Proveedor*, seleccione *Proveedor OLE DB de Microsoft Office 15.0 Access Database Engine* y, a continuación, haga clic en **Siguiente**.




4. Abra la pestaña *Conexión* del cuadro de diálogo y, en el campo *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.
5. En la pestaña *Todas* haga doble clic en la propiedad *Contraseña Jet OLEDB:Database* e introduzca la contraseña de la base de datos como valor de la propiedad (*imagen siguiente*).




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* como la ruta de acceso del archivo `System.MDW`.

 © 2019-2025 Altova GmbH

### 4.2.1.10.10 Microsoft Azure SQL (ODBC)

Para poder conectarse de manera correcta a una base de datos Azure SQL, debe tener instalada la última versión del Cliente nativo de SQL Server.

Para más información sobre cómo conectarse a una base de datos Azure SQL en la nube, consulte esta entrada del blog de Altova.

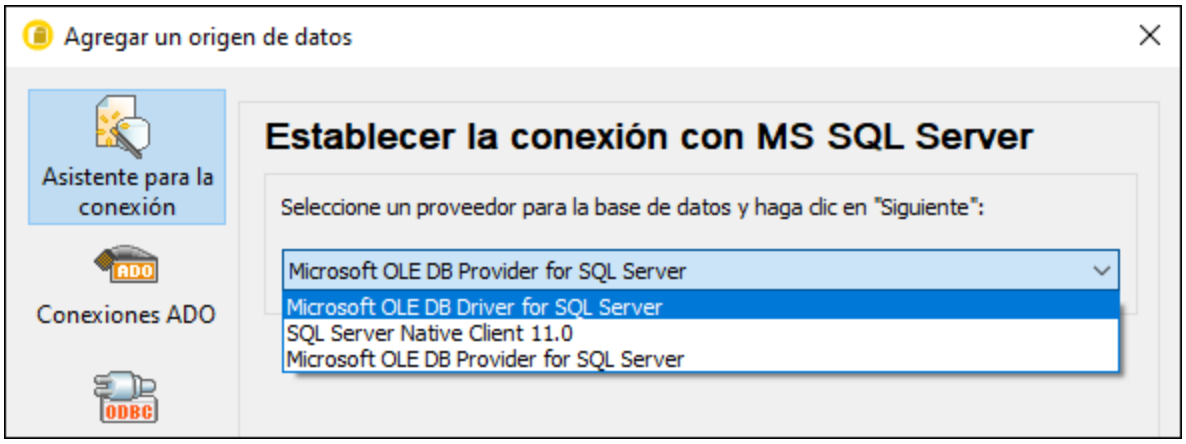
### 4.2.1.10.11 Microsoft SQL Server (ADO)

En este ejemplo explicamos cómo conectarse a una base de datos SQL Server mediante ADO. Estas instrucciones funcionan con el controlador OLE DB de Microsoft para SQL Server (MSOLEDBSQL) recomendado que coincide con su plataforma de MapForce (32 ó 64 bits). Para otros proveedores ADO, las instrucciones serían similares, pero puede que necesite configurar otras propiedades de conexión, como se describe en Configurar las propiedades de vínculo de datos de SQL Server 188.

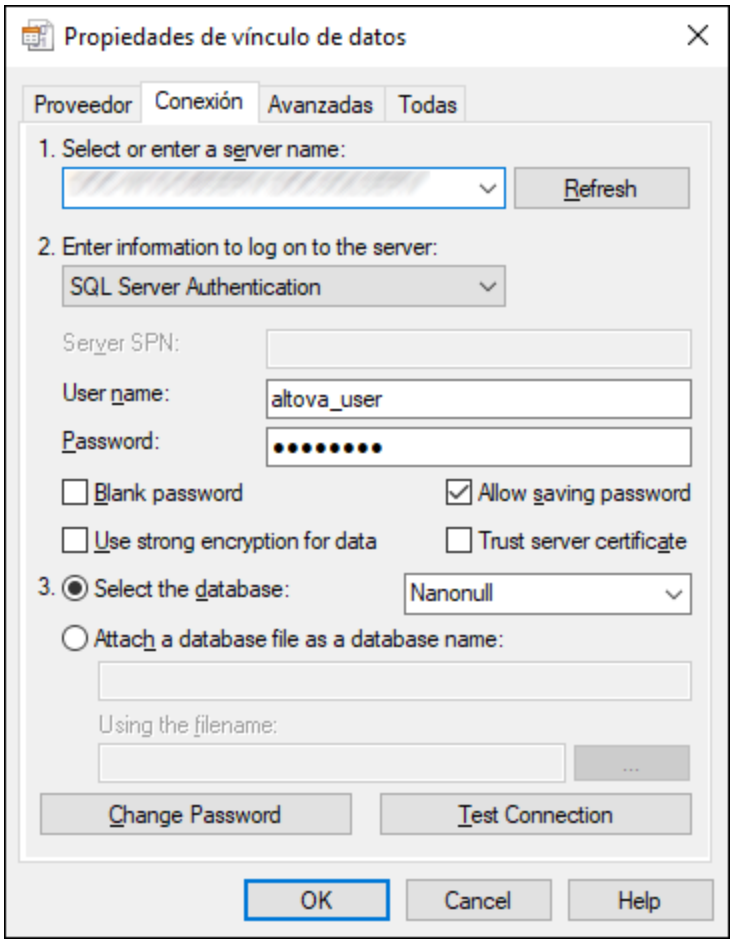
**Nota:** Es un problema conocido que el *proveedor Microsoft OLE DB para SQL Server (SQLOLEDB)* tiene dificultades para vincular parámetros de consultas complejas como las expresiones comunes de tabla e instrucciones SELECT anidadas.

#### Conexión

1. Inicie el asistente para la conexión de base de datos 179, seleccione **Asistente para la conexión** y después *Microsoft SQL Server (ADO)*. Haga clic en **Siguiente**.
2. Verá que aparece una lista de proveedores ADO disponibles (*imagen siguiente*). 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.




3. Haga clic en **Siguiente**. Aparecerá el cuadro de diálogo Propiedades del vínculo de datos (*imagen siguiente*).




4. En la pestaña *Conexión*, seleccione o introduzca el nombre del servidor de BD (por ejemplo, *SQLSERV01*). Si se va a conectar a una instancia de SQL Server, entonces el nombre del servidor tendrá el formato *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, desactive la casilla Contraseña en blanco e introduzca las credenciales de la 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*).
7. Puede probar la conexión haciendo clic en **Probar la conexión**.
8. Haga clic en **Aceptar** cuando haya terminado.

### 4.2.1.10.12 Microsoft SQL Server (ODBC)

En este ejemplo explicamos cómo conectarse a una base de datos SQL Server mediante ODBC.

#### Requisitos previos

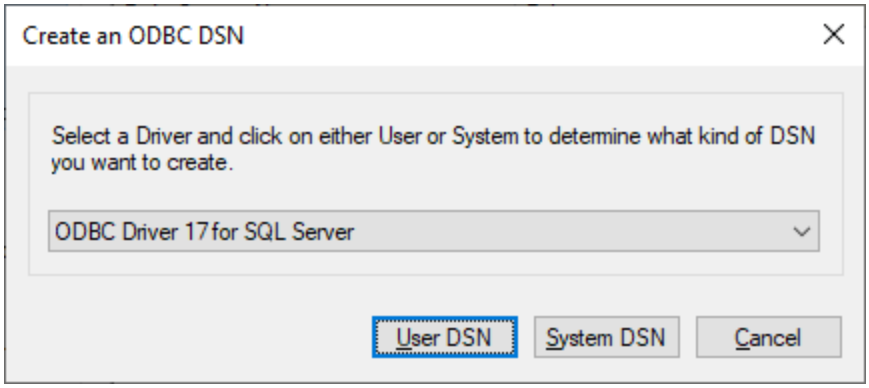
Descargar e instalar el controlador ODBC de Microsoft para SQL Server desde el sitio web de Microsoft. Este ejemplo utiliza Microsoft ODBC Driver 17 para SQL Server para conectarse a una base de datos SQL Server

 © 2019-2025 Altova GmbH

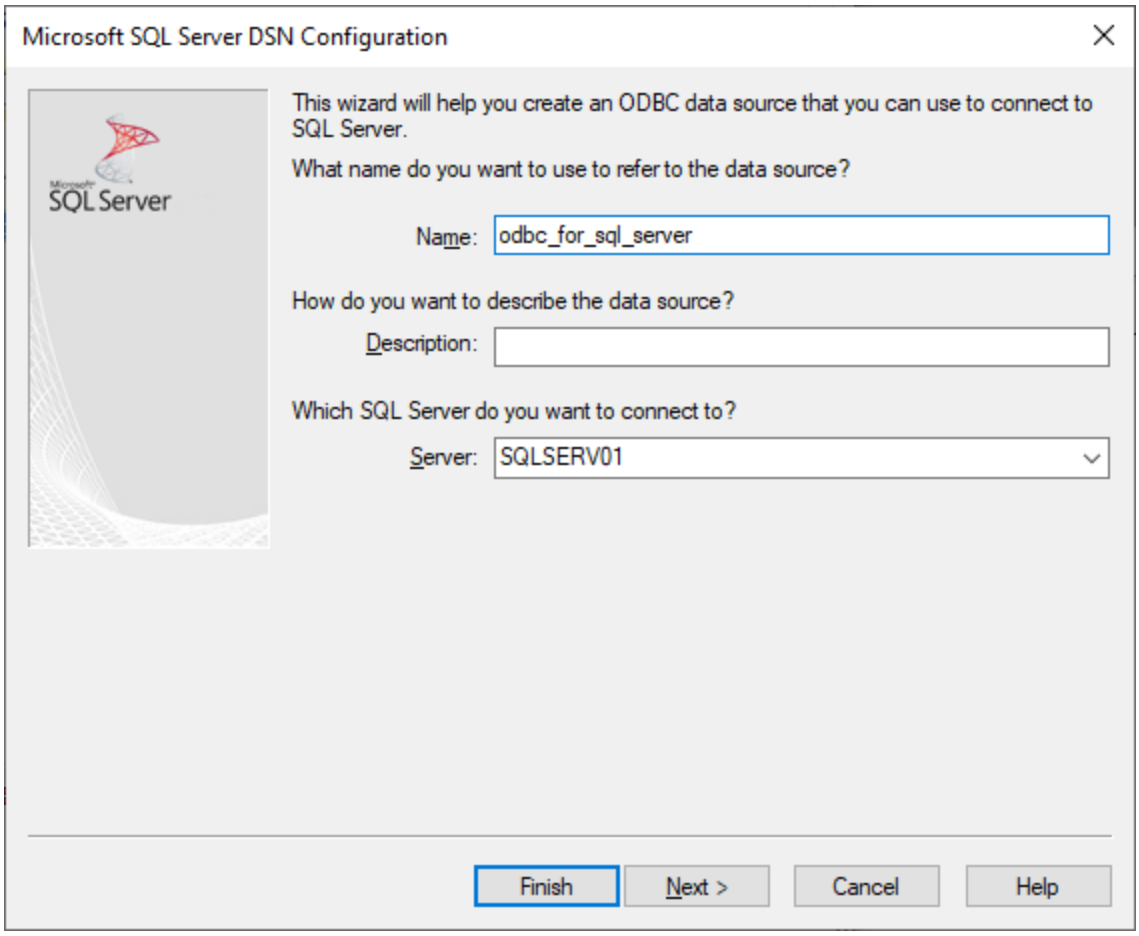
2016. Es posible que necesite una versión diferente del controlador ODBC, en función de su versión de SQL Server. Para obtener información sobre compatibilidad, consulte los requisitos del sistema del controlador.

## Conexión

1. Inicie el asistente de conexión a bases de datos 179 y haga clic en **Conexiones ODBC**.
2. Seleccione *DSN de usuario* (o *DSN de sistema* si tiene privilegios administrativos) y luego haga clic en **Crear DSN nuevo**.
3. Seleccione el controlador de la lista. Tenga en cuenta que este solo aparece en esta lista una vez se ha instalado.

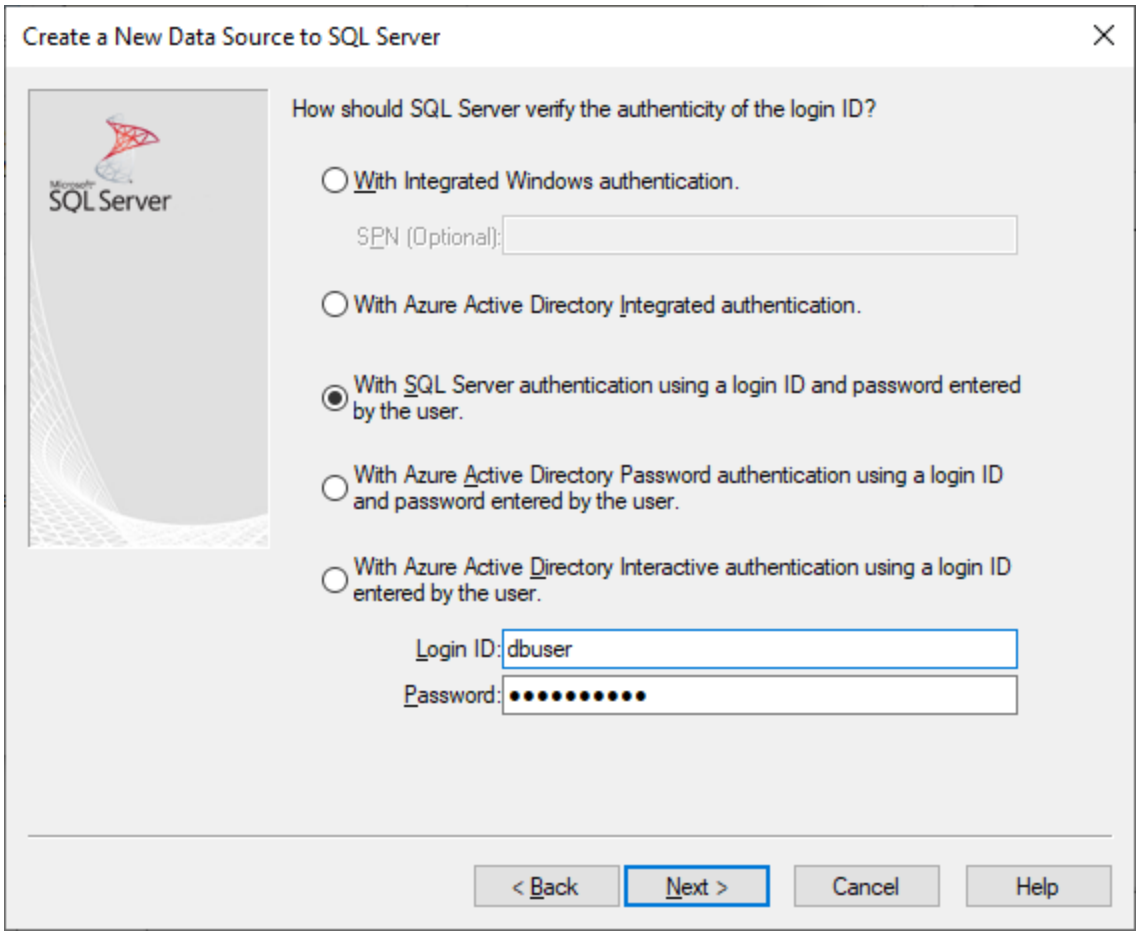



4. Haga clic en *DSN de usuario* (o en *DSN de sistema* si está creando un DSN de sistema). Para crear un DSN de sistema, necesita ejecutar MapForce como administrador. Por lo tanto, para crear un DSN de sistema, cancele el asistente, asegúrese de ejecutar MapForce como administrador y vuelva a realizar los pasos anteriores.
5. Introduzca un nombre para el origen de datos, introduzca, opcionalmente, una descripción para identificar esta conexión y seleccione de la lista el servidor SQL Server al que se va a conectar. Haga clic en **Siguiente** para pasar a la siguiente pantalla.

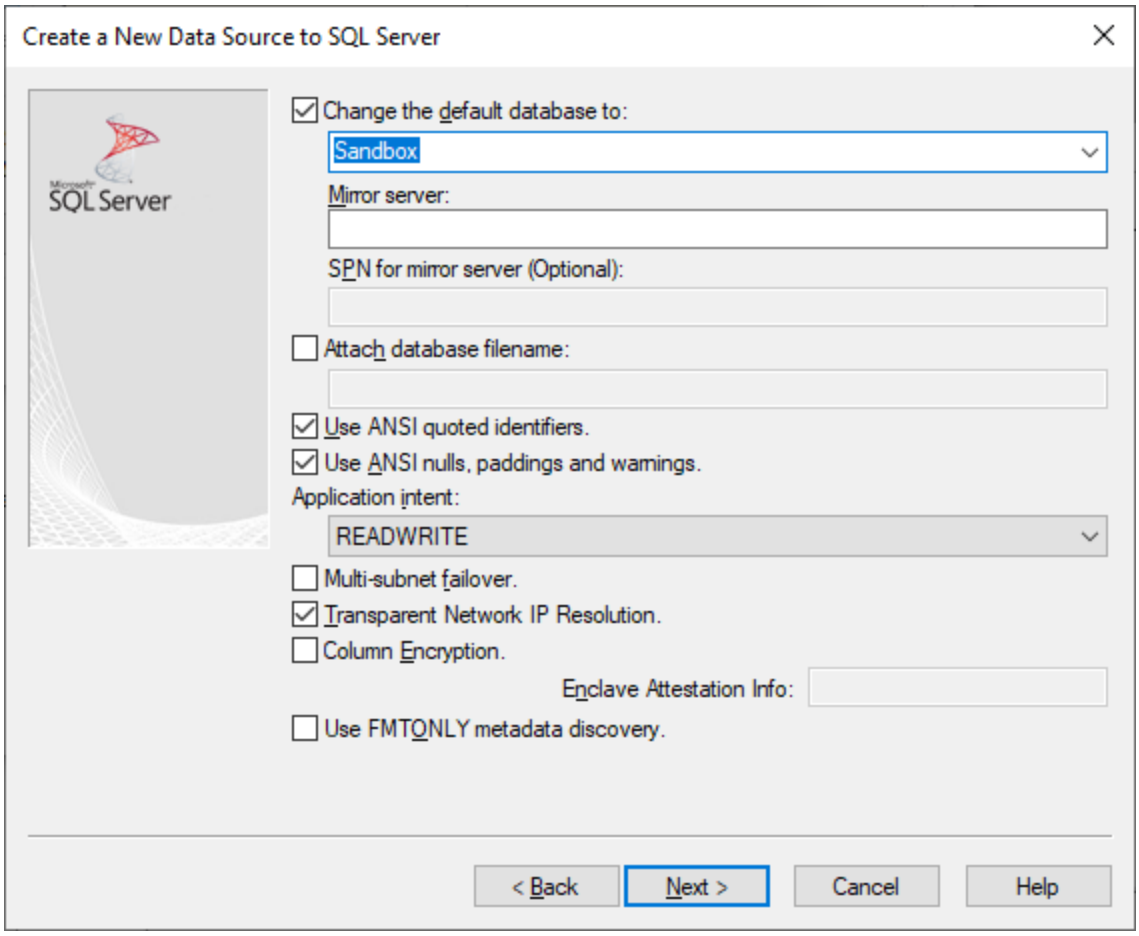



6. En la siguiente pantalla, seleccione el método de autenticación que desea utilizar. Si el servidor de base de datos se configuró para permitir conexiones de usuarios autenticados en el dominio de Windows, seleccione *Con autenticación Windows integrada*. De lo contrario, seleccione la opción que necesite. Este ejemplo utiliza la opción *Con autenticación SQL Server*, que no exige que se introduzca el nombre de usuario y la contraseña en las casillas correspondientes. Haga clic en **Siguiente** cuando haya terminado.

 © 2019-2025 Altova GmbH

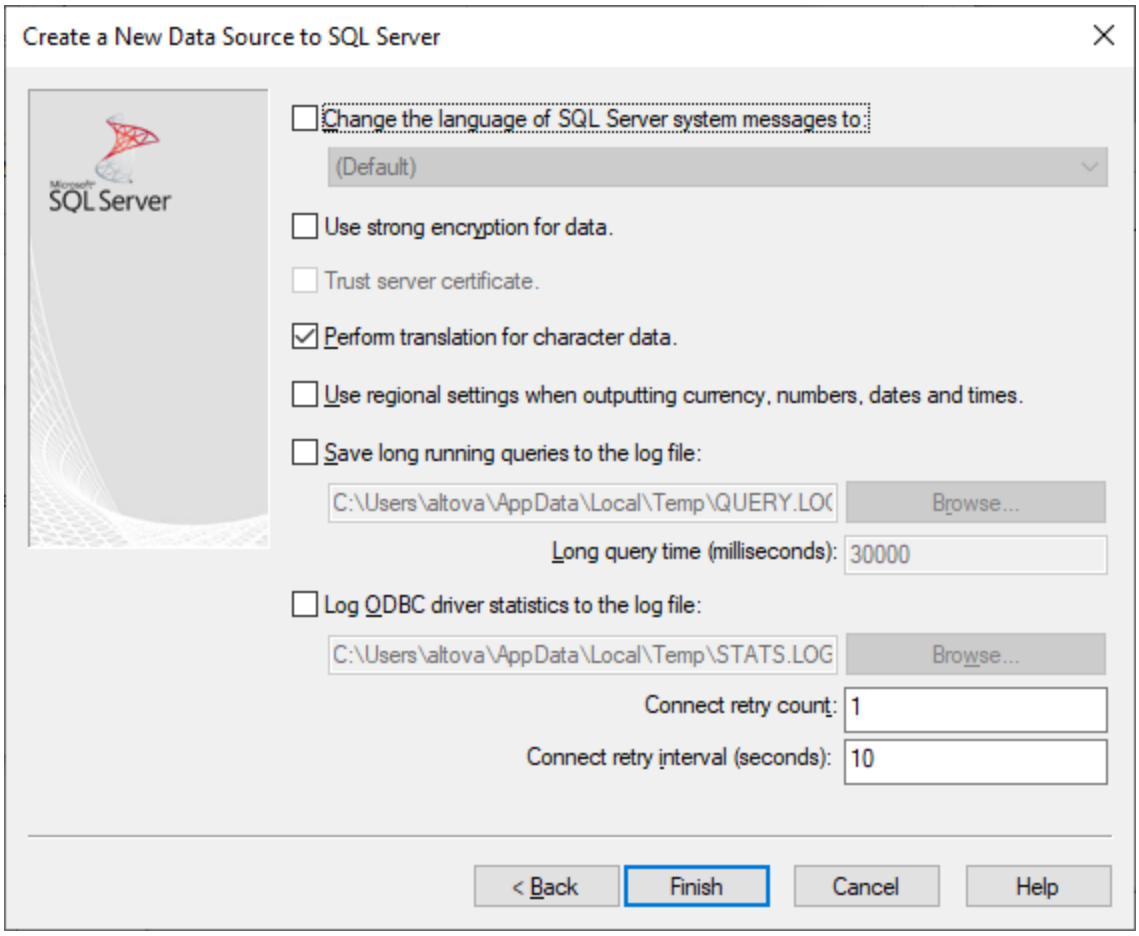



7. En la siguiente pantalla de nuestro ejemplo (*imagen siguiente*), hemos seleccionado *Cambiar la base de datos predeterminada* e introducido *Sandbox* (el nombre de la base de datos a la que conectarse). Haga clic en **Siguiente** cuando haya introducido los ajustes que desee.




8. Haga clic en **Siguiente** y, si quiere, configure el resto de los parámetros de esta conexión. Haga clic en **Finalizar** cuando haya terminado.


© 2019-2025 Altova GmbH




9. Aparece un cuadro de diálogo de conformación que contiene los detalles de la conexión. Haga clic en **Aceptar**.
10. Ahora el origen de datos aparece en la lista de orígenes de datos de *Usuario* o *Sistema*.

### 4.2.1.10.13 MySQL (ODBC)

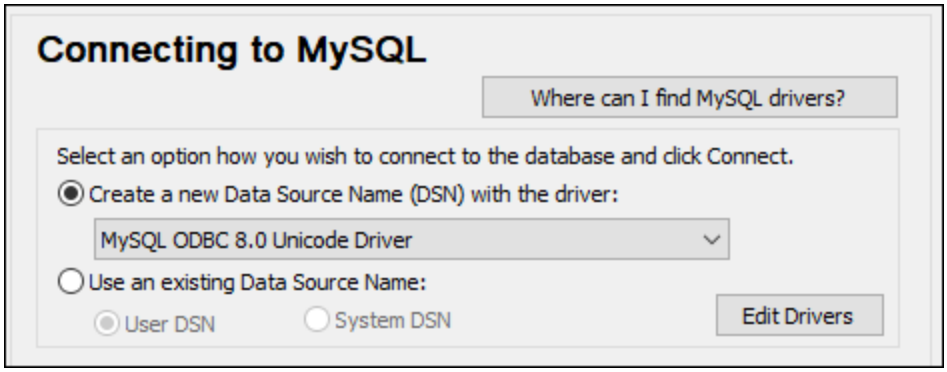
Este ejemplo ilustra cómo conectarse a un servidor de bases de datos MySQL desde un equipo Windows a través de MySQL ODBC, cuyo controlador debe descargarse e instalarse por separado. En este ejemplo, usamos el conector MySQL/ODBC 8.0.

#### Requisitos previos

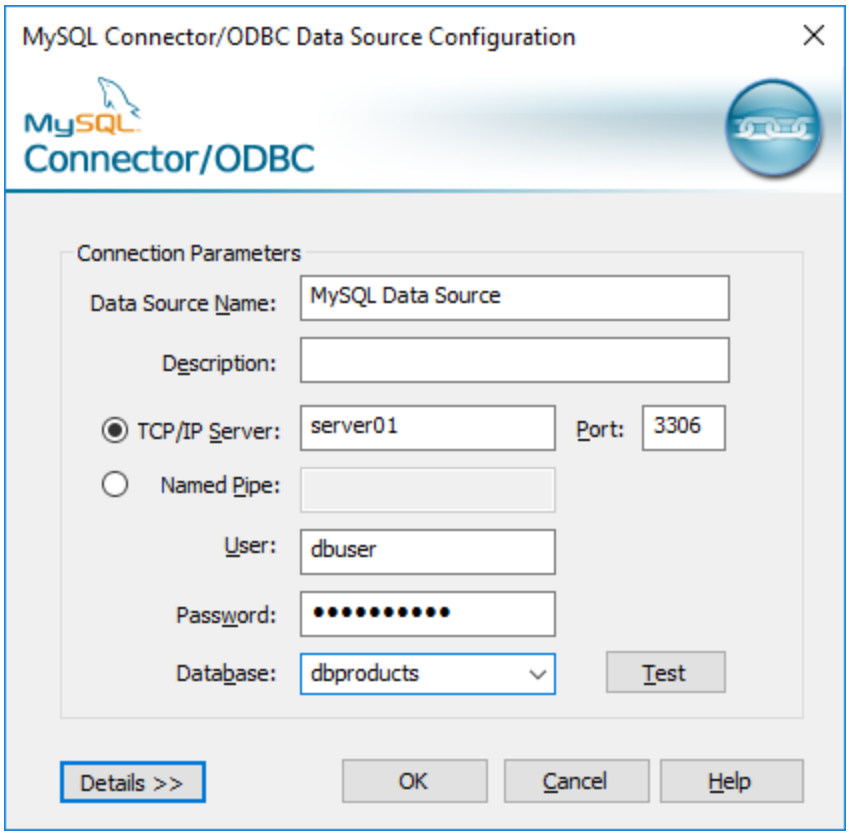
- El controlador ODBC de MySQL debe estar instalado en su sistema operativo. Consulte la documentación de MySQL para conocer la versión del controlador recomendada para la versión de su servidor de bases de datos.
- Asegúrese de que la plataforma de MapForce (32 o 64 bits) es la misma que la del controlador MySQL ODBC.
- Disponer de los datos de conexión: host, base de datos, puerto, nombre de usuario y contraseña.

## Conexión

1. Inicie el asistente para la conexión de base de datos [179], seleccione **Asistente para la conexión** y después *MySQL (ODBC/Native)*. Haga clic en **Siguiente**.
2. Seleccione la opción *Crear un DNS nuevo con el controlador* y seleccione un controlador MySQL. (Si no hay ningún controlador disponible en la lista, haga clic en **Editar controladores** y seleccione cualquier controlador MySQL disponible de la lista de controladores ODBC instalados en su sistema. Consulte también el apartado Ver los controladores ODBC disponibles [203].




3. Haga clic en **Conectarse** para abrir el cuadro de diálogo Configuración del origen de datos (*imagen siguiente*).




 © 2019-2025 Altova GmbH

4. 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. Si quiere, introduzca una descripción.
5. 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.

### 4.2.1.10.14 Oracle (JDBC)

Este ejemplo explica cómo conectarse a un servidor de base de datos Oracle 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 solo requiere el entorno Java y las bibliotecas .jar suministradas por el paquete Oracle Instant Client.

#### Requisitos previos

- 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 determinará la ruta a la máquina virtual Java (JVM) desde las siguientes ubicaciones, en este orden: (i) la ruta de la JVM personalizada que puede haber configurado en las **Opciones** de la aplicación; ver Configuración de Java[1132]; (ii) la ruta de la JVM que se encuentra en el registro de Windows; (iii) 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.
- El paquete Oracle Instant Client Package (Basic) está disponible en el sistema operativo. El paquete se puede descargar del sitio web oficial de Oracle. Este ejemplo utiliza el paquete Oracle Instant Client versión 12.1.0.2.0, para Windows de 32 bits y, en consecuencia, Oracle JDK de 32 bits.
- Disponer de los datos de conexión: host, puerto, nombre del servicio, nombre de usuario y contraseña.

#### Conexión

1. Inicie el asistente para la conexión a bases de datos[179] y haga clic en **Conexiones JDBC** (*consulte Conexión JDBC*[197] *para ver una imagen del cuadro de diálogo*).
2. En el campo *Variables Classpath* introduzca la ruta de acceso del archivo `.jar` que proporciona al 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. Si ha añadido la ruta del archivo a la variable CLASSPATH[199] del sistema, puede dejar este campo vacío.
3. En el campo *Controlador*, seleccione el controlador adecuado. Los controladores pertinentes solo estarán disponibles si hay una ruta de archivo `.jar` válida en el campo *Variables Classpath* o en la variable de entorno `CLASSPATH`.
4. Introduzca el nombre de usuario y la contraseña para la base de datos en los campos correspondientes.
5. Introduzca la cadena de conexión JDBC en el campo *URL de la base de datos* según el patrón de la tabla siguiente, sustituyendo los valores resaltados por los que corresponden a su servidor de base de datos.
6. Haga clic en **Conectarse**.

*Detalles de conexión del ejemplo con Oracle*

| Campo | Valor |
|---|---|
| Classpaths | `C:\jdbc\instantclient_12_1\ojdbc7.jar` |
| Controlador | *oracle.jdbc.OracleDriver* o *oracle.jdbcdriver.OracleDriver* |
| URL de la base de datos | `jdbc:oracle:thin:@//`**`host`**`:`**`puerto`**`:`**`servicio`** |

### 4.2.1.10.15 Oracle (ODBC)

Este ejemplo muestra cómo conectarse desde MapForce a un servidor de base de datos Oracle en un equipo de la red, a través de un cliente de base de datos Oracle instalado en el sistema operativo 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 Configurar una conexión ODBC 201.

#### Requisitos previos

- El cliente de BD Oracle (que incluye el controlador ODBC Oracle) está instalado y configurado en el sistema. Para más instrucciones, consulte la documentación 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)
      )
    )
  ```

  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:

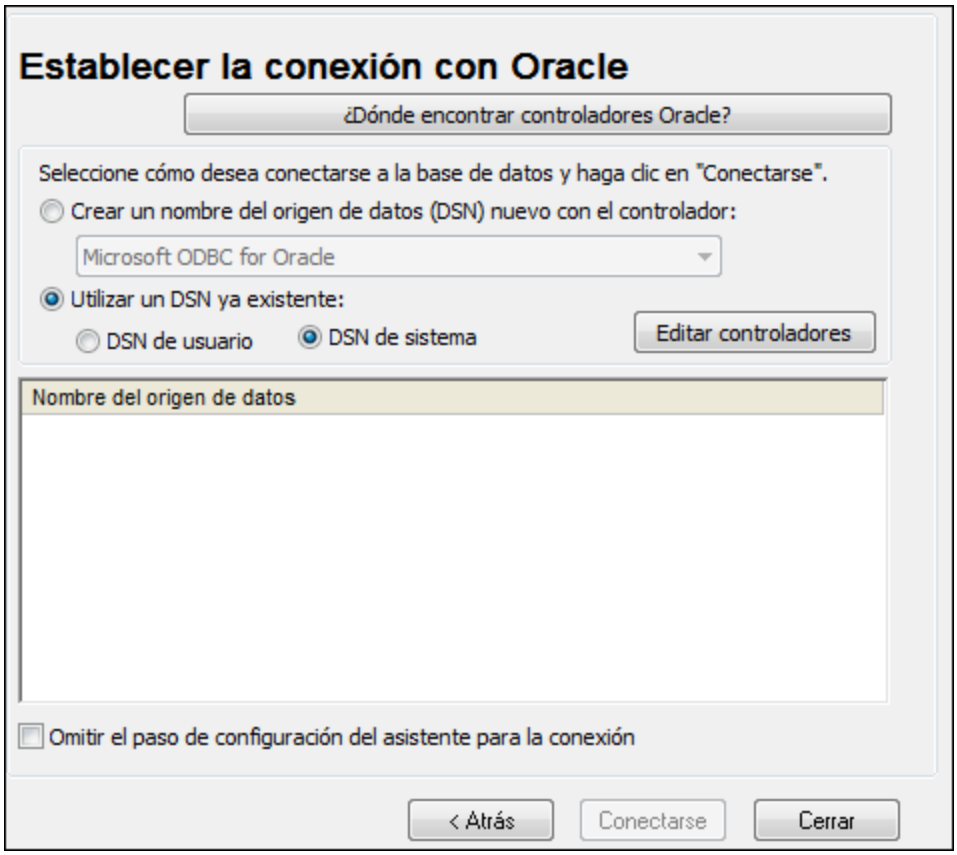
  ```
  C:\app\nombreUsuario\product\11.2.0\client_1\network\admin\tnsnames.ora
  ```

  Puede agregar nuevas entradas a **`tnsnames.ora`**, pegando los detalles de la conexión y guardando el archivo, o ejecutando el Asistente de configuración de Oracle Net, 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`**.

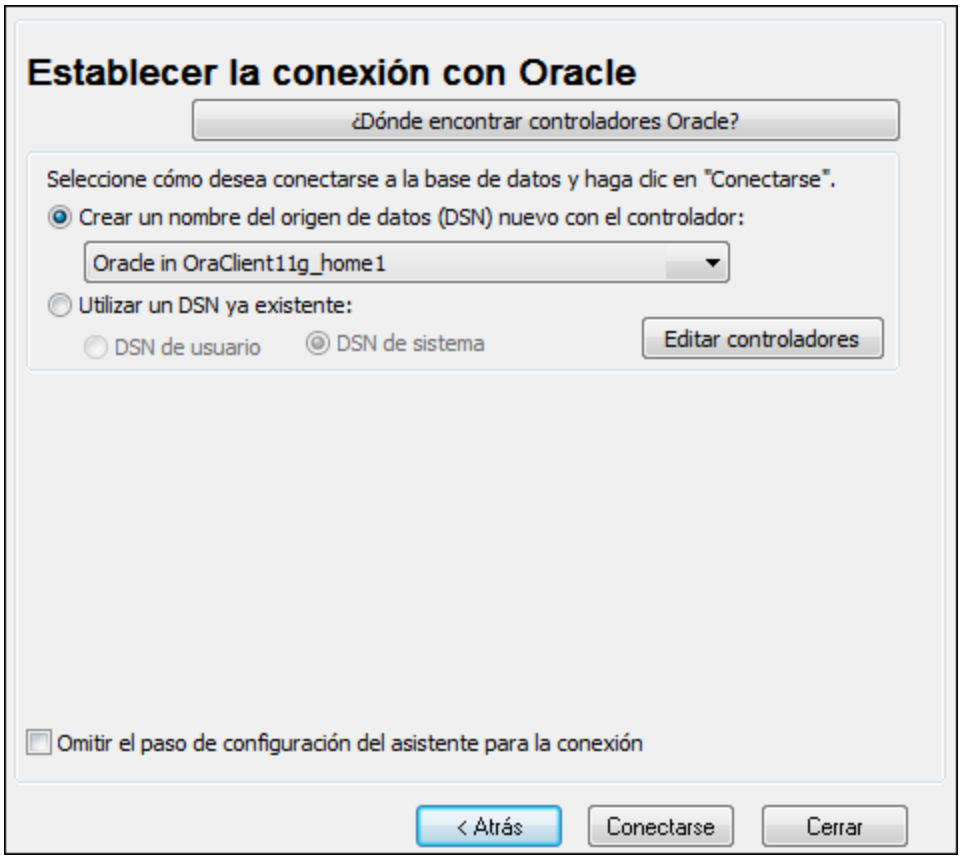
 © 2019-2025 Altova GmbH

## Conexión

1. Inicie el asistente para la conexión de base de datos 179, seleccione **Asistente para la conexión** y después *Oracle (ODBC/JDBC)*. Haga clic en **Siguiente**.
2. Seleccione*ODBC* y haga clic en **Siguiente**.
3. En el cuadro de diálogo que aparece (*imagen siguiente*), haga clic en **Editar controladores**.

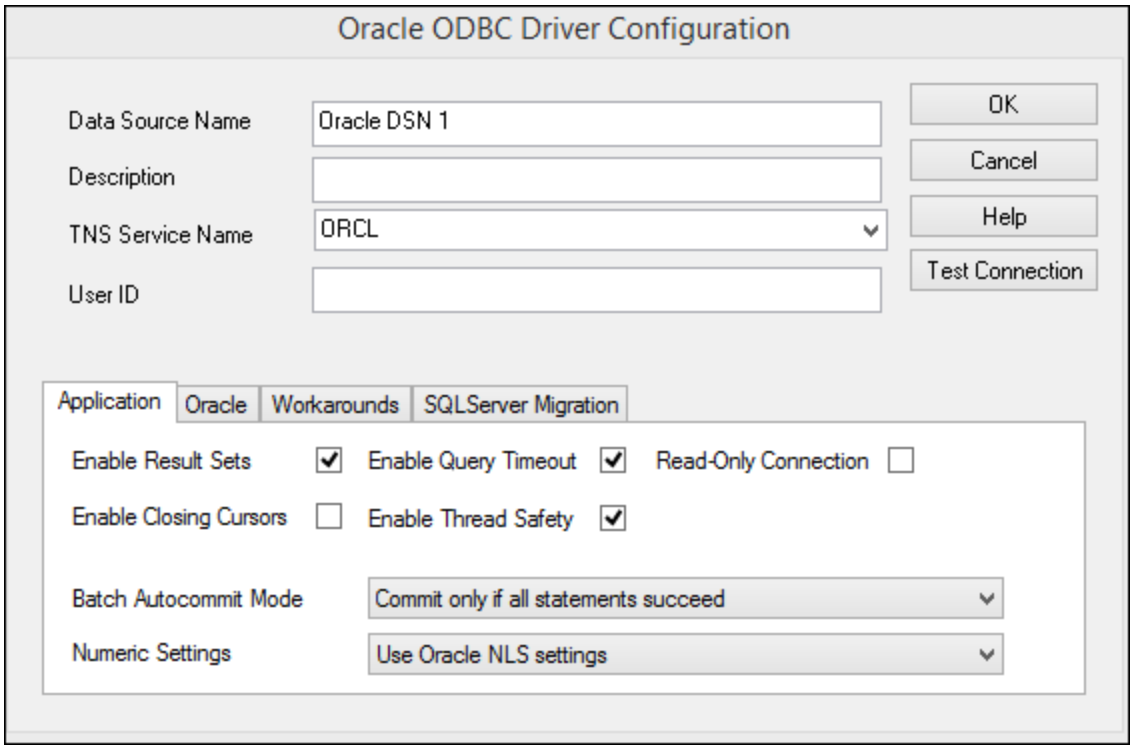



4. En la lista de controladores de Oracle disponibles en su sistema, seleccione el controlador de Oracle que desea utilizar y haga clic en **Atrás**.
5. Seleccione la opción *Crear un DSN nuevo con el controlador* y después seleccione el controlador de Oracle elegido en el paso nº 4. (Recomendamos no utilizar el controlador Microsoft ODBC para Oracle. Microsoft recomienda utilizar el controlador ODBC que ofrece Oracle.)




6. Haga clic en **Conectarse**.
7. En el cuadro de diálogo que aparece, rellene los campos obligatorios (*que se muestran rellenados en la imagen siguiente*). El nombre de la conexión debe introducirse en el campo *Nombre del servicio TNS* tal y como está definido en el archivo `tnsnames.ora` (ver *Requisitos previos* más arriba). *Nota:* Si desea que la lista desplegable del cuadro combinado se rellene automáticamente con los valores del archivo **`tnsnames.ora`**, puede que tenga que añadir la ruta a la carpeta admin como variable de entorno **`TNS_ADMIN`**.

 © 2019-2025 Altova GmbH




8. Haga clic en **Aceptar** cuando haya terminado.
9. En el cuadro de diálogo que aparece, introduzca el nombre de usuario y la contraseña de la base de datos y, a continuación, haga clic en **Aceptar**.

### 4.2.1.10.16 PostgreSQL (ODBC)

Este ejemplo muestra cómo conectarse a un servidor de bases de datos PostgreSQL desde un equipo Windows a través de un controlador ODBC PostgreSQL instalado en el equipo local. 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 181 ).

**Nota:** También se puede conectar a un servidor de base de datos PostgreSQL directamente (es decir, sin el controlador ODBC). Consulte el apartado Conexiones nativas 205 para más información.

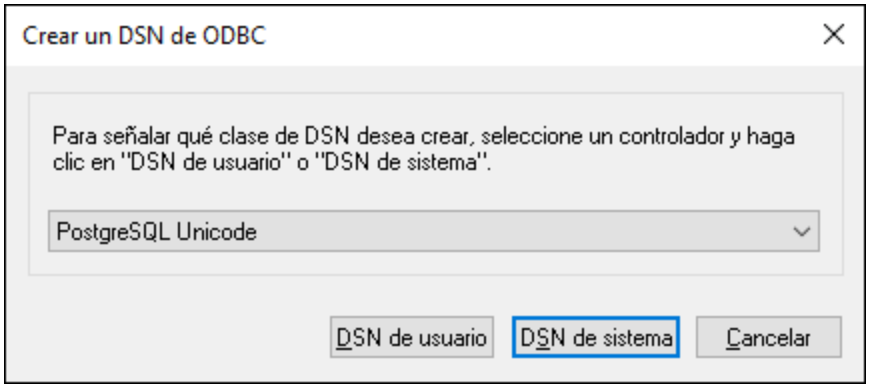
#### Requisitos previos

- El controlador *psqlODBC* debe estar instalado en su sistema.
- Disponer de los siguientes datos de conexión: servidor, puerto, base de datos, nombre de usuario y contraseña.

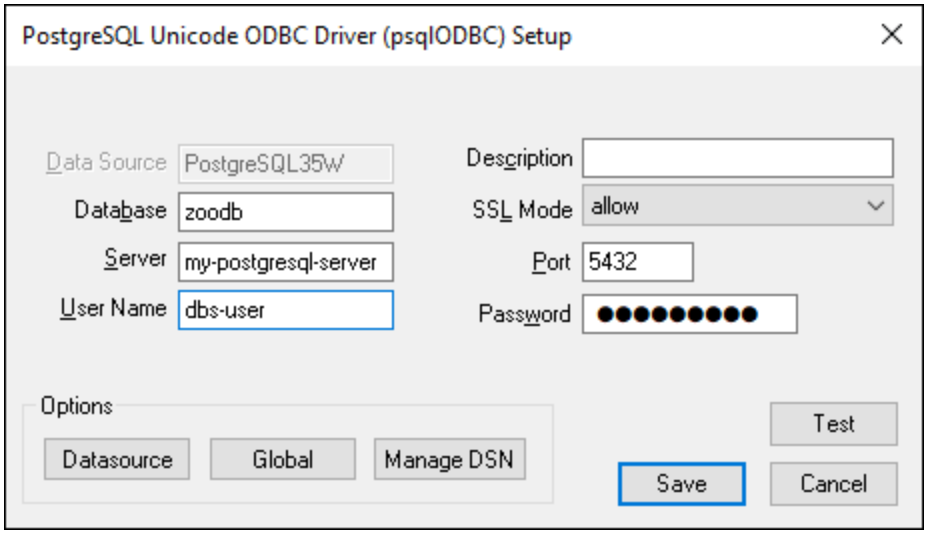
#### Conexión

1. Inicie el asistente de conexión a bases de datos 179 y haga clic en **Conexiones ODBC**.
2. Seleccione *DSN de usuario* (o *DSN de sistema* si tiene privilegios administrativos) y luego haga clic en **Crear DSN nuevo**.

3. Seleccione el controlador de la lista desplegable (*imagen siguiente*) y haga clic en **DSN de usuario**. Si no hay ningún controlador PostgreSQL en la lista, asegúrese de que el controlador ODBC PostgreSQL está instalado en su sistema operativo.




4. En el cuadro de diálogo que aparece (*imagen siguiente*), rellene la conexión a la base de datos (suministrada por el administrador de la base de datos).




5. Haga clic en **Guardar**.

La conexión estará 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**.

### 4.2.1.10.17 Progress OpenEdge (JDBC)

En este apartado encontrará instrucciones para conectarse a un servidor de base de datos Progress OpenEdge 11.6 mediante JDBC.

#### Requisitos previos

- 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 determinará la

 © 2019-2025 Altova GmbH

ruta a la máquina virtual Java (JVM) desde las siguientes ubicaciones, en este orden: (i) la ruta de la JVM personalizada que puede haber configurado en las **Opciones** de la aplicación; ver Configuración de Java[1132]; (ii) la ruta de la JVM que se encuentra en el registro de Windows; (iii) 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.
- 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, ruta de acceso de la base de datos, nombre o alias, nombre de usuario y contraseña.

## Conexión

1. Inicie el asistente para la conexión a bases de datos[179] y haga clic en **Conexiones JDBC** (*consulte Conexión JDBC*[197] *para ver una imagen del cuadro de diálogo*).
2. En el campo *Variables Classpath* introduzca la ruta de acceso del archivo `.jar` que proporciona al 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. Si ha añadido la ruta del archivo a la variable CLASSPATH[199] del sistema, puede dejar este campo vacío.
3. En el campo *Controlador*, seleccione el controlador adecuado. Los controladores pertinentes solo estarán disponibles si hay una ruta de archivo `.jar` válida en el campo *Variables Classpath* o en la variable de entorno `CLASSPATH`.
4. Introduzca el nombre de usuario y la contraseña para la base de datos en los campos correspondientes.
5. Introduzca la cadena de conexión JDBC en el campo *URL de la base de datos* según el patrón de la tabla siguiente, sustituyendo los valores resaltados por los que corresponden a su servidor de base de datos.
6. Haga clic en **Conectarse**.

*Detalles de conexión del ejemplo Progress OpenEdge*

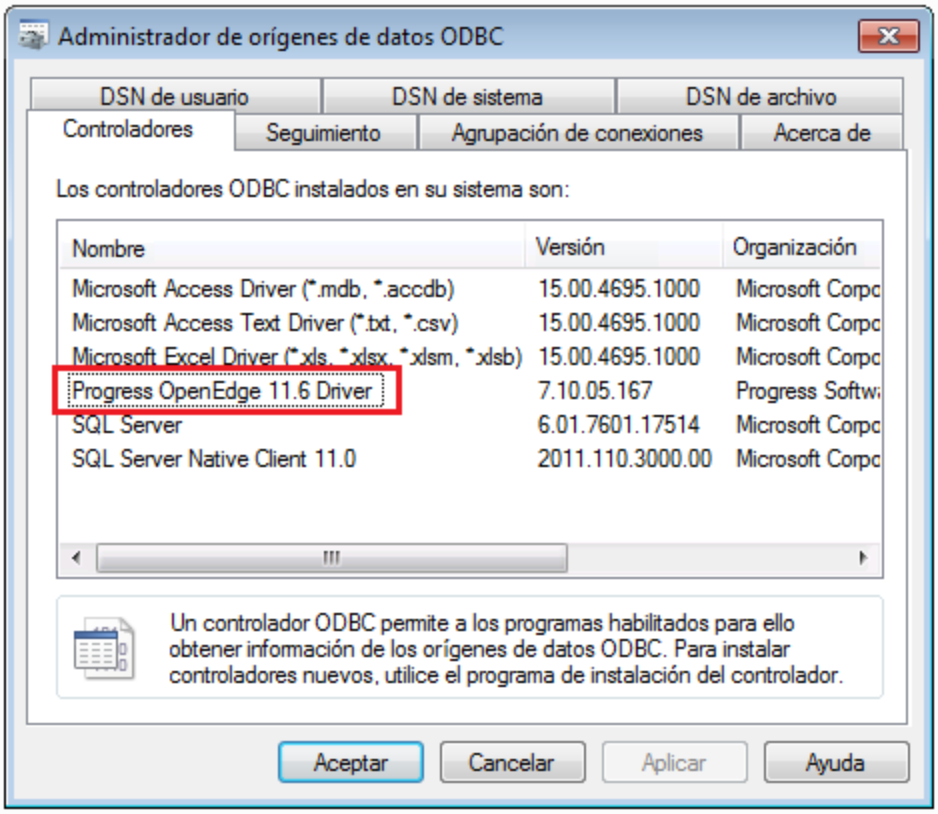
| Campo | Valor |
|---|---|
| Classpaths | `C:\Progress\OpenEdge\java\openedge.jar;C:\Progress\OpenEdge\java\pool.jar;` |
| Controlador | *com.ddtek.jdbc.openedge.OpenEdgeDriver* |
| URL de la base de datos | `jdbc:datadirect:openedge://`**`host`**`:`**`puerto`**`;databaseName=`**`nombre_bd`** |

## 4.2.1.10.18 Progress OpenEdge (ODBC)

Este ejemplo muestra cómo conectarse a un servidor de base de datos Progress OpenEdge a través del controlador ODBC Progress OpenEdge 11.6.

## Requisitos previos

- El controlador *ODBC Connector for Progress OpenEdge* debe estar instalado en su sistema (ver también Resumen de controladores de base de datos 181). Descargue la versión que corresponda a su versión de MapForce (32 bits o 64 bits). Tras la instalación, compruebe si el controlador ODBC está disponible en su equipo 203.



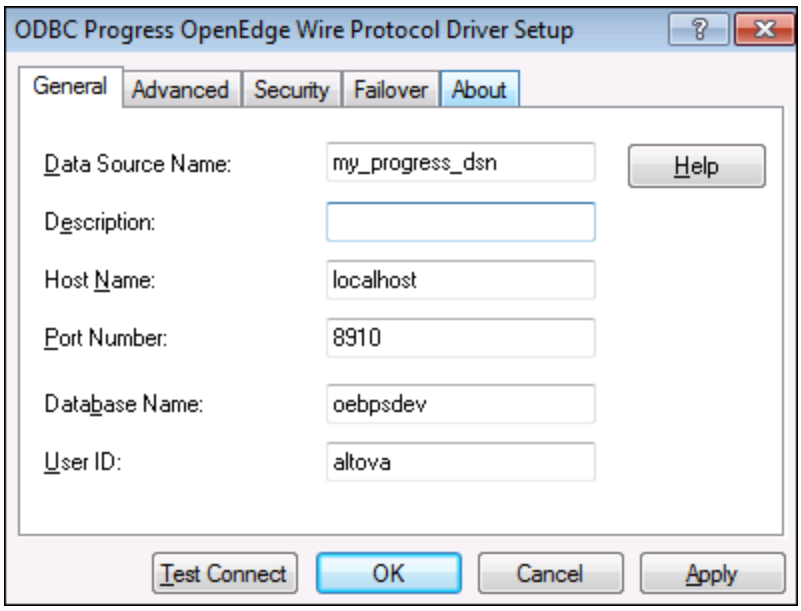

- Disponer de los siguientes datos de conexión a la base de datos: nombre del host, número de puerto, nombre de la base de datos, ID de usuario y contraseña.

## Conexión

1. Inicie el asistente de conexión a bases de datos 179 y haga clic en **Conexiones ODBC**.
2. Seleccione *DSN de usuario* (o *DSN de sistema* si tiene privilegios administrativos) y luego haga clic en **Crear DSN nuevo**.
3. Seleccione *Progress OpenEdge Driver* en la lista desplegable (*imagen siguiente*) y haga clic en **DSN de usuario** (o **DSN de sistema** si procede).


© 2019-2025 Altova GmbH

4. Rellene los datos de conexión a la base de datos (Base de datos, Servidor, Puerto, Nombre de usuario, Contraseña) y haga clic en **Aceptar**.




5. Para verificar la conexión antes de guardar los datos introducidos, haga clic en **Probar conexión**. Para guardar haga clic en **Aceptar**.
6. El origen de datos nuevo aparece ahora en la lista de orígenes de datos ODBC. Haga clic en **Conectarse** para iniciar la conexión.

### 4.2.1.10.19 Sybase (JDBC)

Este ejemplo explica cómo conectarse a un servidor de base de datos Sybase mediante JDBC.

#### Requisitos previos

- 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 determinará la ruta a la máquina virtual Java (JVM) desde las siguientes ubicaciones, en este orden: (i) la ruta de la JVM personalizada que puede haber configurado en las **Opciones** de la aplicación; ver Configuración

de Java[1132]; (ii) la ruta de la JVM que se encuentra en el registro de Windows; (iii) 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.
- 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*).
- Disponer de los datos de conexión: host, puerto, ruta de acceso de la base de datos, nombre o alias, nombre de usuario y contraseña.

## Conexión

1. Inicie el asistente para la conexión a bases de datos[179] y haga clic en **Conexiones JDBC** (*consulte Conexión JDBC*[197] *para ver una imagen del cuadro de diálogo*).
2. En el campo *Variables Classpath* introduzca la ruta de acceso del archivo `.jar` que proporciona al 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. Si ha añadido la ruta del archivo a la variable CLASSPATH[199] del sistema, puede dejar este campo vacío.
3. En el campo *Controlador*, seleccione el controlador adecuado. Los controladores pertinentes solo estarán disponibles si hay una ruta de archivo `.jar` válida en el campo *Variables Classpath* o en la variable de entorno `CLASSPATH`.
4. Introduzca el nombre de usuario y la contraseña para la base de datos en los campos correspondientes.
5. Introduzca la cadena de conexión JDBC en el campo *URL de la base de datos* según el patrón de la tabla siguiente, sustituyendo los valores resaltados por los que corresponden a su servidor de base de datos.
6. Haga clic en **Conectarse**.

*Detalles de conexión del ejemplo con Sybase*

| Campo | Valor |
|---|---|
| Classpaths | `C:\sybase\jConnect-7_0\classes\jconn4.jar` |
| Controlador | *com.sybase.jdbc4.jdbc.SybDriver* |
| URL de la base de datos | `jdbc:sybase:Tds:`**`nombreHost`**`:`**`puerto`**`/`**`nombreBaseDatos`** |

### 4.2.1.10.20 Teradata (JDBC)

Este ejemplo explica cómo conectarse a un servidor de base de datos Teradata mediante JDBC.

## Requisitos previos

- 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 determinará la ruta a la máquina virtual Java (JVM) desde las siguientes ubicaciones, en este orden: (i) la ruta de la JVM personalizada que puede haber configurado en las **Opciones** de la aplicación; ver Configuración de Java[1132]; (ii) la ruta de la JVM que se encuentra en el registro de Windows; (iii) la variable de entorno `JAVA_HOME`.

 © 2019-2025 Altova GmbH

- Asegúrese de que la plataforma de MapForce (32 o 64 bits) es del mismo tipo 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. En este ejemplo, se utiliza Teradata JDBC Driver 16.20.00.02. Para más información, consulte https://downloads.teradata.com/download/connectivity/jdbc-driver.
- Disponer de los datos de conexión: host, puerto, ruta de acceso de la base de datos, nombre o alias, nombre de usuario y contraseña.

## Conexión

1. Inicie el asistente para la conexión a bases de datos 179 y haga clic en **Conexiones JDBC** (*consulte Conexión JDBC* 197 *para ver una imagen del cuadro de diálogo*).
2. En el campo *Variables Classpath* introduzca la ruta de acceso del archivo `.jar` que proporciona al 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. Si ha añadido la ruta del archivo a la variable CLASSPATH 199 del sistema, puede dejar este campo vacío.
3. En el campo *Controlador*, seleccione el controlador adecuado. Los controladores pertinentes solo estarán disponibles si hay una ruta de archivo `.jar` válida en el campo *Variables Classpath* o en la variable de entorno `CLASSPATH`.
4. Introduzca el nombre de usuario y la contraseña para la base de datos en los campos correspondientes.
5. Introduzca la cadena de conexión JDBC en el campo *URL de la base de datos* según el patrón de la tabla siguiente, sustituyendo los valores resaltados por los que corresponden a su servidor de base de datos.
6. Haga clic en **Conectarse**.

*Detalles de conexión del ejemplo con Teradata*

| Campo | Valor |
|---|---|
| Classpaths | `C:\jdbc\teradata\` |
| Controlador | *com.teradata.jdbc.TeraDriver* |
| URL de la base de datos | `jdbc:teradata://`**`nombreServidorBaseDatos`** |

### 4.2.1.10.21 Teradata (ODBC)

Este ejemplo explica cómo conectarse a un servidor de base de datos Teradata mediante ODBC.

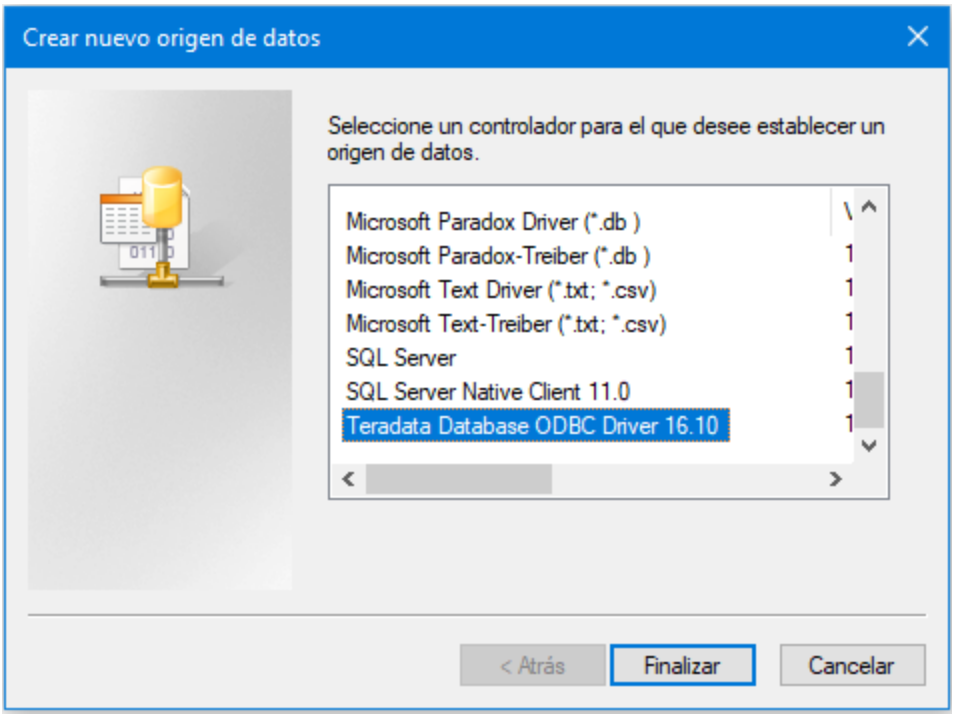
## Requisitos previos

- Tener instalado el controlador ODBC de Teradat (puede descargarse del sitio web de Teradata).
- Disponer de los datos de conexión: host, nombre de usuario y contraseña.

## Conexión

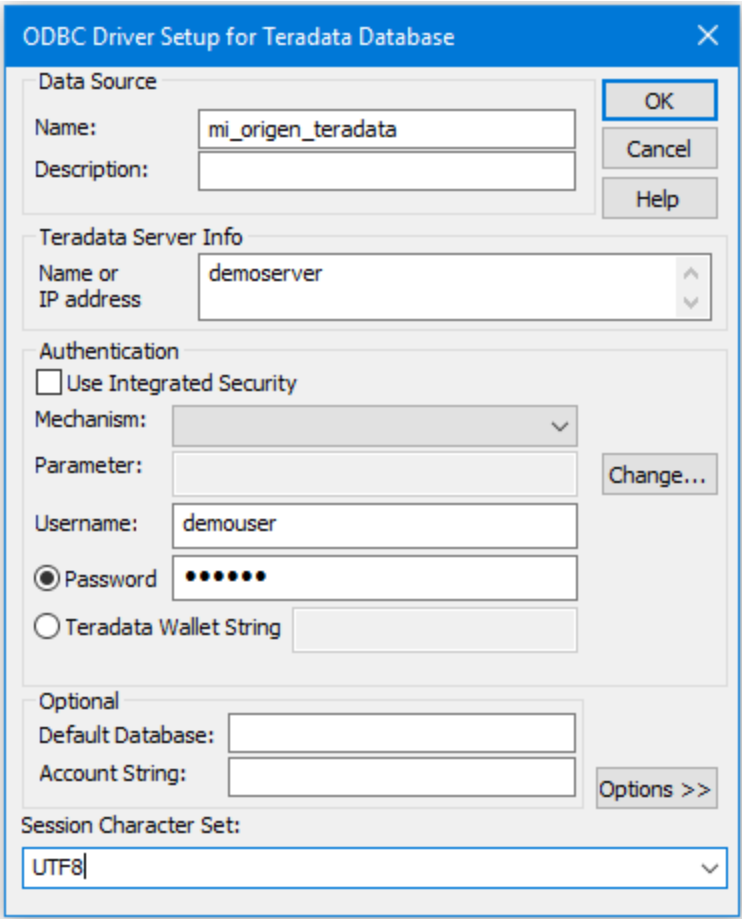
1. Pulse la tecla **Windows**, teclee `ODBC` y seleccione *Configurar orígenes de datos ODBC (32 bits)* en la lista de sugerencias. (Si procede, seleccione en su lugar la opción de 64 bits).

2. En el cuadro de diálogo Administrador de orígenes de datos ODBC que aparece, haga clic en la pestaña *DSN de sistema* y, a continuación, en **Agregar**.
3. En el cuadro de diálogo Crear origen de datos nuevo (imagen siguiente), seleccione *Controlador ODBC de base de datos Teradata* y haga clic en **Finalizar**.

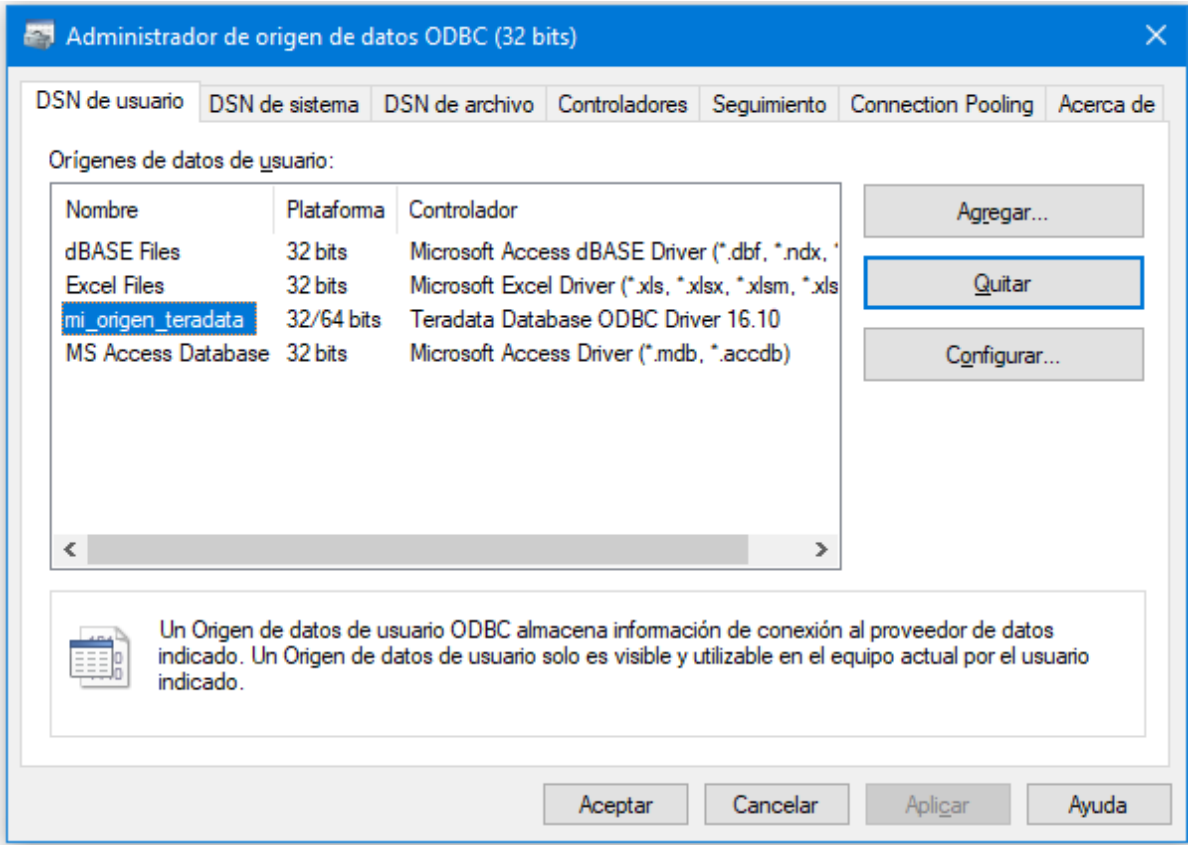



4. Introduzca la base de datos y los datos de autenticación, y haga clic en **Aceptar** cuando haya terminado.


© 2019-2025 Altova GmbH

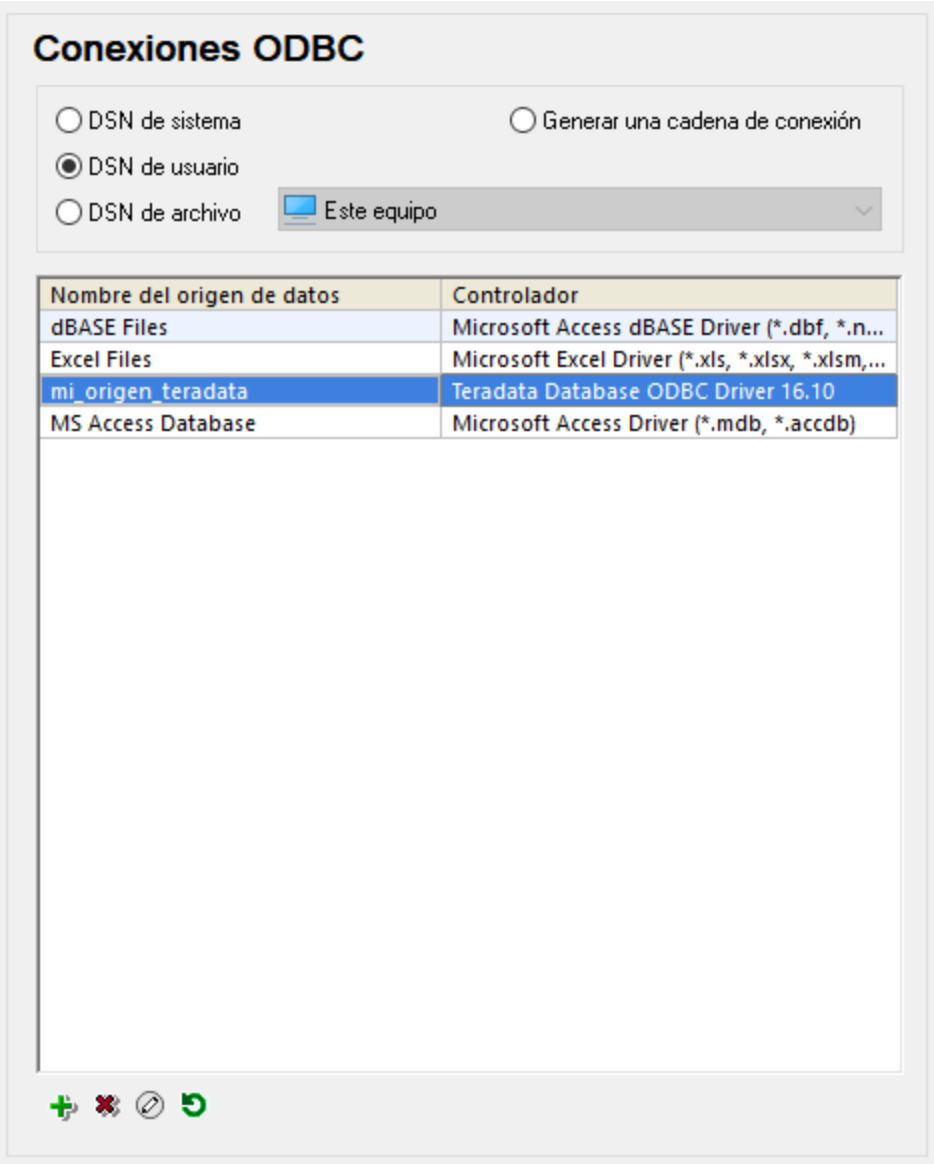



5. Haga clic en **Aceptar**. El origen de datos aparecerá en la lista Orígenes de datos.




6. Ejecute MapForce e inicie el asistente para la conexión de base de datos 179.
7. Haga clic en **Conexiones ODBC** y seleccione *DSN de sistema* (*imagen siguiente*).


© 2019-2025 Altova GmbH




8. El origen de datos que acaba de crear aparecerá en la lista. Haga clic en **Conectarse** para iniciar la conexión.

**Nota:** Si recibe este error: *El controlador devolvió un valor no válido (o no devolvió) SQL_DRIVER_ODBC_VER: 03.80,* asegúrese de que la ruta al cliente ODBC (por ejemplo, `C:\Archivos de programa\Teradata\Client\16.10\bin`) existe en la variable de entorno **`PATH`** de su sistema. Si falta esta ruta de acceso, entonces deberá añadirla a mano.

## 4.2.2 Procedimientos generales

En este apartado explicamos cómo agregar, editar y quitar objetos de BD en una asignación, así como administrar las relaciones de BD y configurar algunas de las opciones de la base de datos.

### Iconos de columna de la BD

Las tablas de BD están representadas por el icono . Las columnas de BD están representadas por el icono . Si existe una restricción en esa columna, el icono aparece modificado. Si la columna tiene asignada varias restricciones, se usa el icono de la restricción de mayor prioridad. La prioridad de las restricciones se describe en la tabla siguiente, empezando por la de mayor prioridad.

| | |
|---|---|
| | Esta columna se usa como clave primaria de la tabla. |
| | Esta columna tiene una restricción única. |
| | Esta columna tiene una clave foránea que hace referencia a la clave primaria de otra tabla. |
| | Esta columna contiene datos XML 304. |
| | Existe un valor predeterminado para esta columna. Si no se indica ningún valor para esta columna, se insertará el valor predeterminado en su lugar. |

### Agregar una base de datos a una asignación de datos

Para agregar una base de datos a una asignación debe seleccionar uno de estos lenguajes de transformación 19: Built-In, C++, C#, or Java. Las bases de datos SQLite solo se pueden usar con el lenguaje de transformación Built-in. Si quiere implementar y ejecutar la asignación en FlowForce Server, ejecutarla con MapForce Server o utilizar funciones como Usar transferencia por lotes 281 y procedimientos almacenados 315, debe seleccionar Built-In.

Una vez seleccionado el lenguaje de transformación deseado, puede añadir una base de datos a la asignación de una de las siguientes maneras:

- Seleccione **Base de datos** en el menú **Insertar**.
- Haga clic en el botón de la barra de herramientas.

Esto abre el asistente para la conexión a bases de datos 179, que le guiará en el proceso de conexión. Para más información sobre cómo conectarse a una base de datos consulte Conectarse a un origen de datos 178. Una vez se ha establecido correctamente la conexión con la BD debe seleccionar objetos de la BD que quiere agregar a su asignación (*ver los apartados a continuación*).

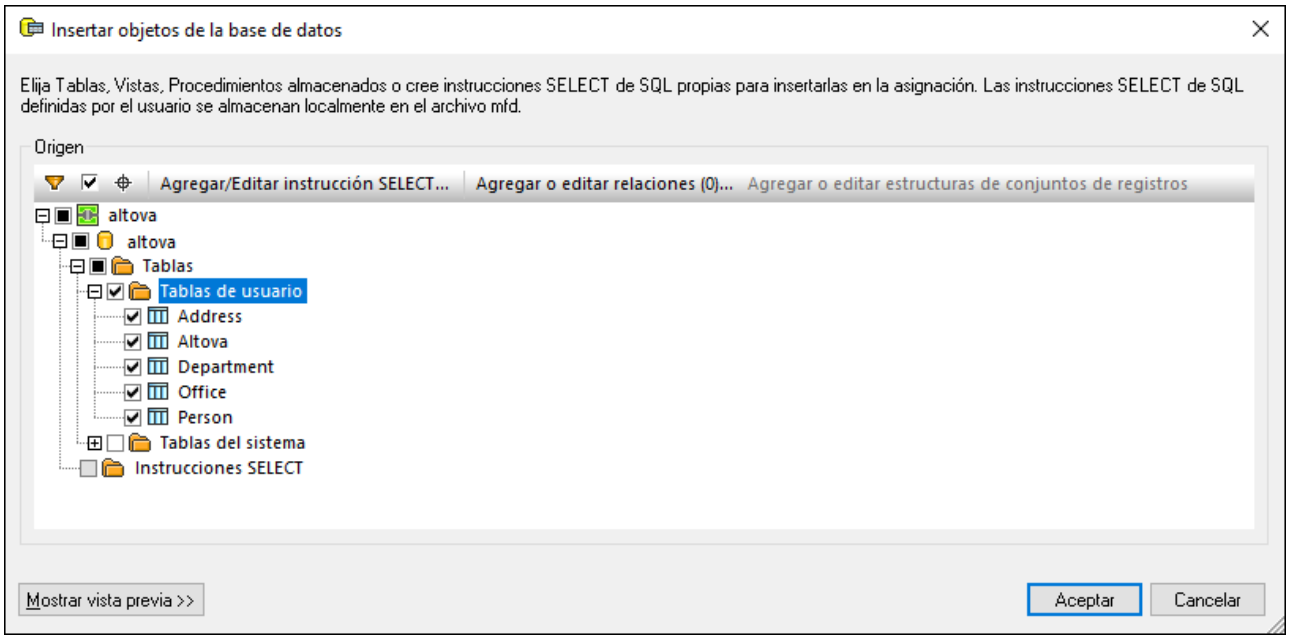
Las bases de datos también se pueden agregar a la asignación en forma de variable. Cuando se añade una estructura de base de datos como variable, aparece el mismo asistente de conexión.

### Agregar objetos de base de datos

Una vez se haya conectado al origen de datos debe seleccionar los objetos de la BD que quiere que aparezcan en la asignación. En el cuadro de diálogo **Insertar objetos de la base de datos** más abajo puede ver la estructura de la base de datos **`Altova.sqlite`**. Para incluir un objeto de BD en la asignación, marque la

 © 2019-2025 Altova GmbH

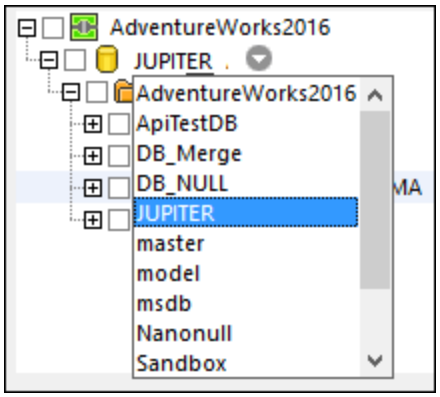
casilla que le corresponda y haga clic en **Aceptar**. En nuestro ejemplo hemos incluido todas las tablas de usuario.




*Estructura del cuadro de diálogo Insertar objetos de la base de datos*

El nodo de la estructura indica la conexión de BD. La estructura siguiente varía en función del tipo de BD. Por ejemplo, las base de datos Oracle y IBM DB2 tienen un nodo de esquema bajo el nodo de conexión, mientras que otros tipos de BD tienen un nodo de catálogo (BD). La fuente en negrita indica el catálogo (BD) o esquema predeterminados, según el caso.

Si su cuenta de usuario de la base de datos tiene acceso a varias base de datos o esquemas en el servidor, puede cambiar de unos a otros haciendo clic en el icono (*véase más abajo*).

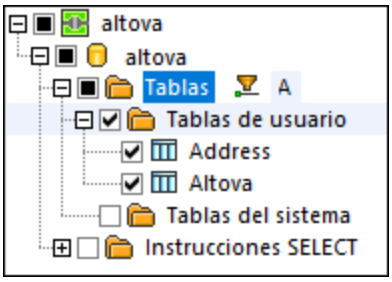



*Opciones disponibles en el cuadro de diálogo Insertar objetos de BD*

Las opciones disponibles en el cuadro de diálogo **Insertar objetos de BD** se describen a continuación.

- Filtro

El botón (**Filtro**) permite filtrar los objetos por el nombre. Una vez haya hecho clic en el botón **Filtro** el icono del filtro se habilita junto a los objetos que se pueden filtrar (en este ejemplo, `Tablas`). Haga clic en el icono de filtro para elegir una de estas opciones: *Sin filtro*, *Contiene*, *No contiene*, *Empieza con*, *Termina con*, *Es igual a*. En este ejemplo hemos decidido incluir solamente las tablas cuyo nombre empieza por `A` (*véase más abajo*).




- Mostrar únicamente los objetos seleccionados

  El botón (**Mostrar únicamente los objetos seleccionados**) muestra solamente los objetos cuya casilla está activa.

- Localizador de objetos

  El botón (**Localizador de objetos**) permite encontrar elementos concretos de la base de datos. Seleccione un objeto o tipo con concreto o teclee su nombre en el cuadro combinado que aparece en la parte inferior del cuadro de diálogo.

- Agregar/Editar instrucciones SELECT

  El botón **Agregar/Editar instrucción SELECT** sirve para agregar o editar instrucciones SELECT personalizadas para la base de datos actual. Los datos que devuelvan dichas instrucciones podrán utilizarse como componentes de origen en la asignación Para más información, consulte el apartado Instrucciones SELECT personalizadas 259.

- Agregar/Editar relaciones

  El botón **Agregar/Editar relaciones** sirve para definir relaciones de clave principal o foránea entre los campos de la BD, además de las relaciones que ya existen en la base de datos. Para más información consulte el apartado Relaciones locales 269.

- Agregar/Editar estructuras de conjuntos de registros

  El botón **Agregar/Editar estructuras de conjuntos de registros** se aplica a bases de datos compatibles con procedimientos almacenados 315. Solamente se habilita si se seleccionó un procedimiento almacenado en el cuadro de diálogo.

- Mostrar vista previa

  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

 © 2019-2025 Altova GmbH

consultar bases de datos fuera del proceso de asignación. Para más información, véase el apartado Consulta de bases de datos 294.

## Usar nombres de objeto relativos a un esquema predeterminado

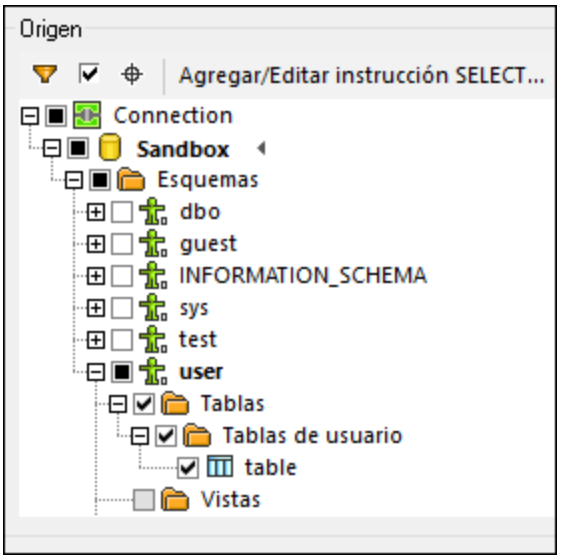
En MapForce, es importante asegurarse de que los nombres de los objetos de BD son relativos a un esquema si quiere cambiar a otra BD más adelante. También es útil si se cambia el nombre del esquema de BD en el servidor y necesita actualizar la asignación en consecuencia. Si el esquema nuevo tiene la misma estructura que el que se usó en tiempo de ejecución, puede cambiar a este sin tener que cambiar manualmente las conexiones de la asignación.

Tenga en cuenta lo siguiente:

- Los nombres de objetos relativos a un esquema predeterminado solamente se pueden usar en las bases de datos que son compatibles con los esquemas: IBM DB2, IBM Informix, IBM Db2 para i (iSeries), Oracle, PostgreSQL, Progress OpenEdge, SQL Server y Sybase.
- No se pueden usar nombres relativos si el componente de BD incluye relaciones locales 269 o instrucciones SQL SELECT como tablas virtuales 259.
- 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.

Para hacer que los objetos de BD sean relativos al esquema predeterminado, siga los siguientes pasos:

1. Abra el cuadro de diálogo **Insertar objetos de BD** o haga clic con el botón derecho en el encabezado del componente de BD y seleccione **Agregar/Editar/Quitar objetos de BD** en el menú contextual.
2. Seleccione uno o más objetos que *pertenezcan al esquema predeterminado* o al *catálogo y al esquema predeterminados (BD)*. 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.



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.
- Si tiene los privilegios necesarios, puede volver a configurar el servidor de BD para cambiar el esquema predeterminado del usuario de BD que ya existe.

En el siguiente ejemplo se ve cómo cambiar el esquema predeterminado de un/a usuario/a de BD. El ejemplo se basa en SQL Server y asume que el catálogo actual es `Sandbox` y que tanto el usuario como el esquema ya existe.

```
USE [Sandbox]
GO
ALTER USER [test_user] WITH DEFAULT_SCHEMA=[test_schema]
GO
```

*Cambiar a una BD o un esquema sin perder conexiones de asignación*
Cuando los nombres de objetos de BD son relativos a un esquema, puede cambiar a otra BD nueva u otro esquema nuevo *sin perder conexiones de asignación*. Estas son las opciones disponibles:

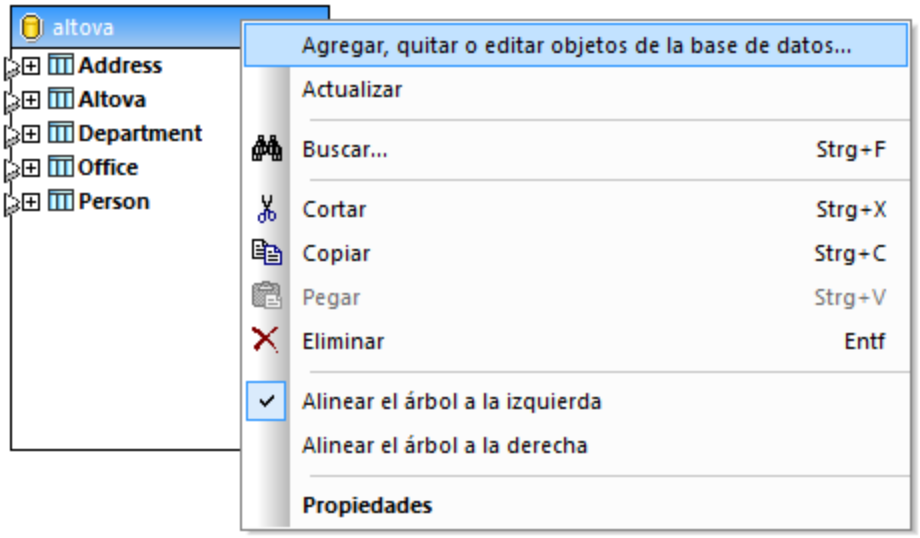
- Abra el cuadro de diálogo Configuración del componente 251 y haga clic en **Cambiar**. Siga las instrucciones del asistente para conectarse a una BD nueva *como usuario/a nuevo/a*. Si la BD nueva tiene la misma estructura, todas las conexiones de la asignación se actualizarán automáticamente. Es decir que estas conexiones ahora coincidirán con *el catálogo y el esquema predeterminados del usuario nuevo de BD*.
- Si necesita realizar este cambio de BD de forma regular, recomendamos que defina la conexión de BD como recurso global 856. Por ejemplo, el recurso global podría tener dos configuraciones: una configuración predeterminada para la BD de desarrollo y una configuración de producción.

Si hay objetos de la BD que aparecen en rojo después del cambio significa que no existen en el esquema de BD nuevo.

## Editar objetos de base de datos

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/Editar/Quitar objetos de la base de datos** del menú contextual (*véase más abajo*). Después, se abre el cuadro de diálogo **Agregar/Editar/Quitar objetos de BD** que permite definir las mismas opciones y características que en el cuadro de diálogo **Insertar objetos de BD**.

 © 2019-2025 Altova GmbH




### 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 el Editor SQL (véase el panel de consulta de BD 294), en el cuadro de texto *Instrucción SQL personalizada* del cuadro de diálogo Acciones de tabla de BD 276 y en el cuadro de diálogo Agregar instrucciones SELECT 259.

Para deshabilitar las sugerencias de finalización automática, siga los siguientes pasos:

1. Seleccione el elemento del menú **Herramientas | Opciones** o pulse **Ctrl+Alt+O**.
2. Abra el apartado *Base de datos | Editor SQL*.
3. Desactive la casilla *Abrir automáticamente* del aparado *Ayudantes de entrada*.

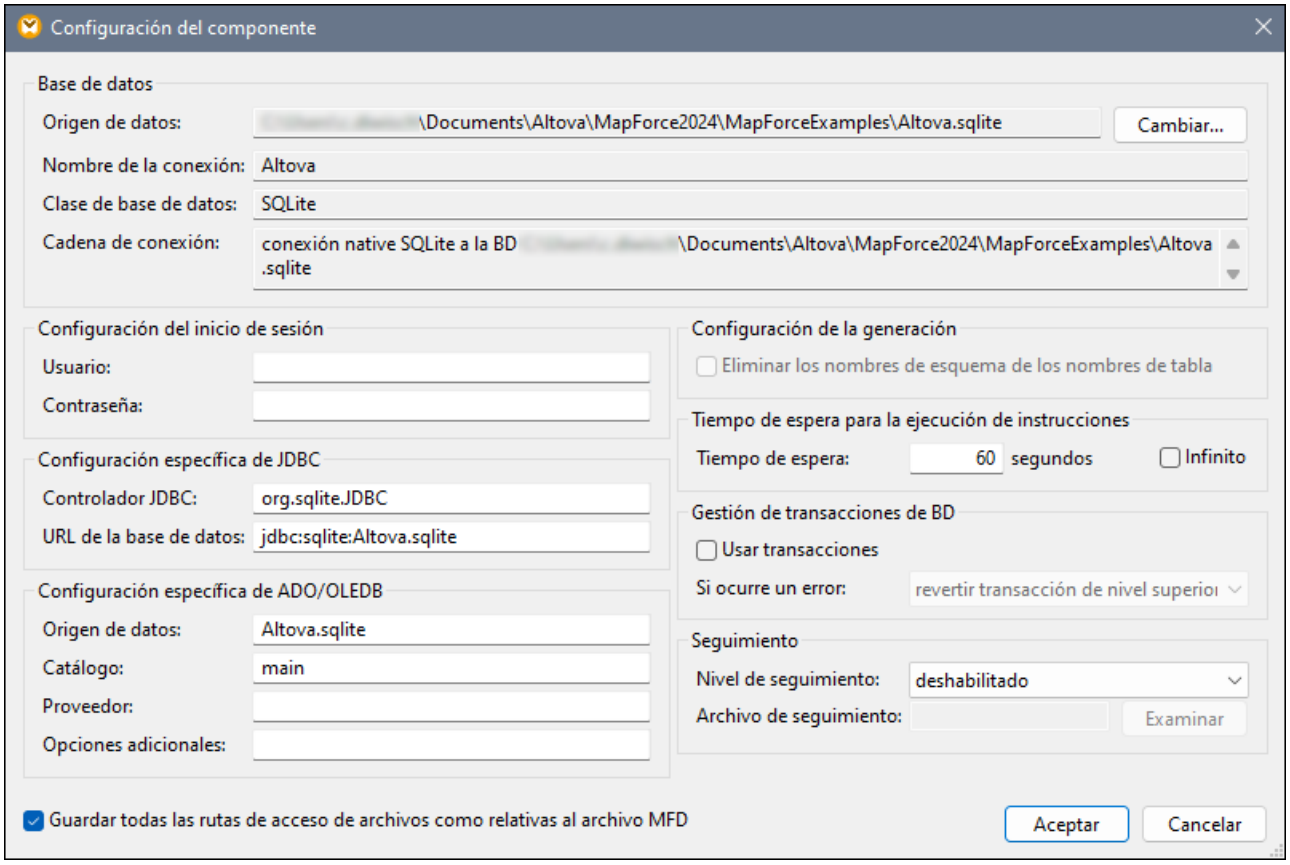
Para invocar las sugerencias de finalización automática a mano, pulse **Ctrl+Espacio**.

Para más información sobre la configuración de bases de datos, consulte Bases de datos 1133.

## 4.2.2.1 Configurar componentes de BD

Una vez haya añadido un componente de BD al área de la asignación puede configurar varios ajustes de BD desde el cuadro de diálogo **Configuración del componente** (*imagen siguiente*). Puede abrir el cuadro de diálogo **Configuración del componente** de una de las siguientes maneras:

- Haga doble clic en la barra de título del componente.
- Haga clic con el botón derecho en el componente y luego en **Propiedades**.
- Seleccione el componente en la asignación. Después haga clic en el menú **Componentes** y seleccione **Propiedades** del menú contextual.




A continuación se pueden ver las opciones disponibles.

## Base de datos

Este grupo muestra informaciones sobre conexiones de BD. Haga clic en **Modificar** para seleccionar otra base de datos o para redefinir los objetos de BD en el componente de BD existente. Se conservarán las conexiones a tablas del mismo nombre. También puede modificar las tablas 250 en el componente, haciendo clic con el botón derecho en un componente de BD y seleccionando **Agregar/Eliminar/Editar objectos de BD**.

- Origen de datos

  Indica la ruta de acceso absoluta o relativa del origen de datos actual.

- Nombre de la conexión

  Indica el nombre de la conexión. Este nombre se genera automáticamente a través de 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 856. En caso de que haya varios componentes de BD con la misma conexión de la asignación, el nombre de esta se mostrará así: `<connection1>`, `<connection2>`, etc.

- Tipo de BD


© 2019-2025 Altova GmbH

Indica el tipo de BD (p.ej. SQLite).

- Cadena de la conexión

  Indica la cadena de conexión a la BD 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.

## Configuraciones de acceso

Estas opciones se usan para los destinos de la generación de código y para el motor de ejecución integrado.

- Usuario/a

  Permite cambiar el nombre de usuario que se utiliza para conectarse a una 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 una base de datos. Es obligatorio si la base de datos exige una contraseña.

## Configuraciones específicas de JDBC

Las opciones específicas de JDBC son relevantes cuando 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. Para más información, consulte *Asignaciones de BD en distintos entornos de ejecución* 175.

- Controlador JDBC

  Muestra el controlador que está activo para el componente de base de datos. El controlador predefinido se introduce automáticamente al definir el componente de BD. Puede cambiar el controlador introducido aquí para adaptarlo a sus necesidades. 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*).

## Configuraciones específicas de ADO/OLEDB

Las opciones específicas de ADO/OLEDB son relevantes cuando la asignación contiene una conexión ADO y se ejecuta con el código C# o C++ generado, o con MapForce Server en Windows. Para más información consulte *Asignaciones de BD en distintos entornos de ejecución* 175. 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 del 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. La opción *Eliminar los nombres de esquema de los nombres de tabla* le permite eliminar los nombres de esquema del código generado y conservar solo los nombres de tabla. Recuerde que esta opción solamente funciona con las instrucciones SQL SELECT generadas con MapForce. No se modificarán las instrucciones SQL definidas por el usuario 259.

La opción *Configuración de la generación* existe para garantizar la compatibilidad con versiones anteriores, por lo que debe evitarla si ese no es su caso. Para hacer que los objetos de BD sean relativos al esquema predeterminado recomendamos que use el método que se describe en *Usar nombres de objetos relativos al esquema predeterminado* 249.

## 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.

- 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 una instrucción de BD. El tiempo de espera predeterminado es de 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

Al ejecutar asignaciones que tengan componentes de BD, es posible que se encuentren varios errores relacionados con la BD (p.ej., claves de índice duplicadas, valores NULL insertados en columnas no-NULL, etc.). Para que sea posible revertir datos de BD si ocurren errores, debe configurar la asignación para que use transacciones de base de datos. Puede habilitar las transacciones a nivel de la base de datos (*configuración actual*), a nivel de acciones de tabla 279, y a nivel de procedimientos almacenados 329. Para más información

 © 2019-2025 Altova GmbH

sobre algunos posibles casos de gestión de transacciones, consulte el apartado Reversión de transacciones: Casos 288.

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 de BD 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 . 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* (*véase más abajo*).

Usar transacciones

Habilita el procesamiento de transacciones para un componente de BD de 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. Si habilita las transacciones a nivel de la base de datos, se incluirán todos los cambios de la base de datos en una única transacción que se revertirá en caso de error relacionado con la BD.

Si ocurre un error

Si marca la casilla *Usar transacciones*, puede elegir qué desea hacer cuando ocurre un error relacionado con la BD:

- *Revertir transacción de nivel superior y detener la operación:* Se revierte la transacción que incluye todos los cambios de la BD y se detiene la ejecución de la asignación.
- *Revertir transacción de nivel superior y continuar:* Ocurre lo mismo, pero la asignación sigue ejecutándose después de la reversión de la transacción (p.ej., para procesar un segundo componente de destino).

A nivel de componente de BD, puede elegir si desea que el procesamiento continúe para otros componentes de destino. Por ejemplo: ha ocurrido un error en el componente de BD en una asignación XML-BD-JSON. En este caso, el archivo JSON todavía se puede procesar y revertir si se ha habilitado la opción *Revertir transacción de nivel superior y continuar*.

## Seguimiento

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 ha hecho en la base de datos durante la ejecución de la asignación. Los cambios que se han realizado en la BD se almacenan en un archivo de seguimiento. Si ocurren errores durante la ejecución, estos también se almacenan. El seguimiento solo es compatible con el lenguaje de transformación integrado.

Puede habilitar el seguimiento a diferentes niveles:

- *A nivel de componente de BD:* El seguimiento a este nivel puede ser útil para asignaciones que tienen varios componentes BD de destino de los que solo necesita habilitar el seguimiento para algunos de ellos. Al habilitar el seguimiento a nivel de componente de BD se habilita automáticamente también para todas las tablas y los procedimientos almacenados de este componente. Para que sea posible monitorizar las tablas relevantes y los procedimientos almacenados, estos deben estar conectados a nodos de origen.
- *A nivel de tabla* 280 *o de procedimiento almacenado* 329: Puede decidir si desea habilitar el seguimiento para una tabla específica o un procedimiento almacenado. A nivel de tabla, el seguimiento incluye eventos relacionados a las acciones de tablas (p.ej., *Insertar todo*). 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* 281*:* Por defecto todos los campos se monitorizan, pero puede excluir ciertos campos del informe de seguimiento o decidir incluir ciertos campos solo en caso de error.

Es importante recordar que los tres niveles mencionados son jerárquicos. Es decir que para configurar el seguimiento en un nivel inferior, primero debe habilitar el seguimiento en un nivel superior. Por ejemplo, si desea configurar el seguimiento a nivel de tabla, primero debe habilitar el seguimiento a nivel de componente de base de datos. Lo mismo ocurre si restringe el seguimiento. Por ejemplo, si lo limita a solamente errores a nivel de los componentes de BD, no puede usar el seguimiento completo a nivel de tabla y de procedimientos almacenados.

MapForce le permite configurar las siguientes opciones de seguimiento:

Nivel de seguimiento

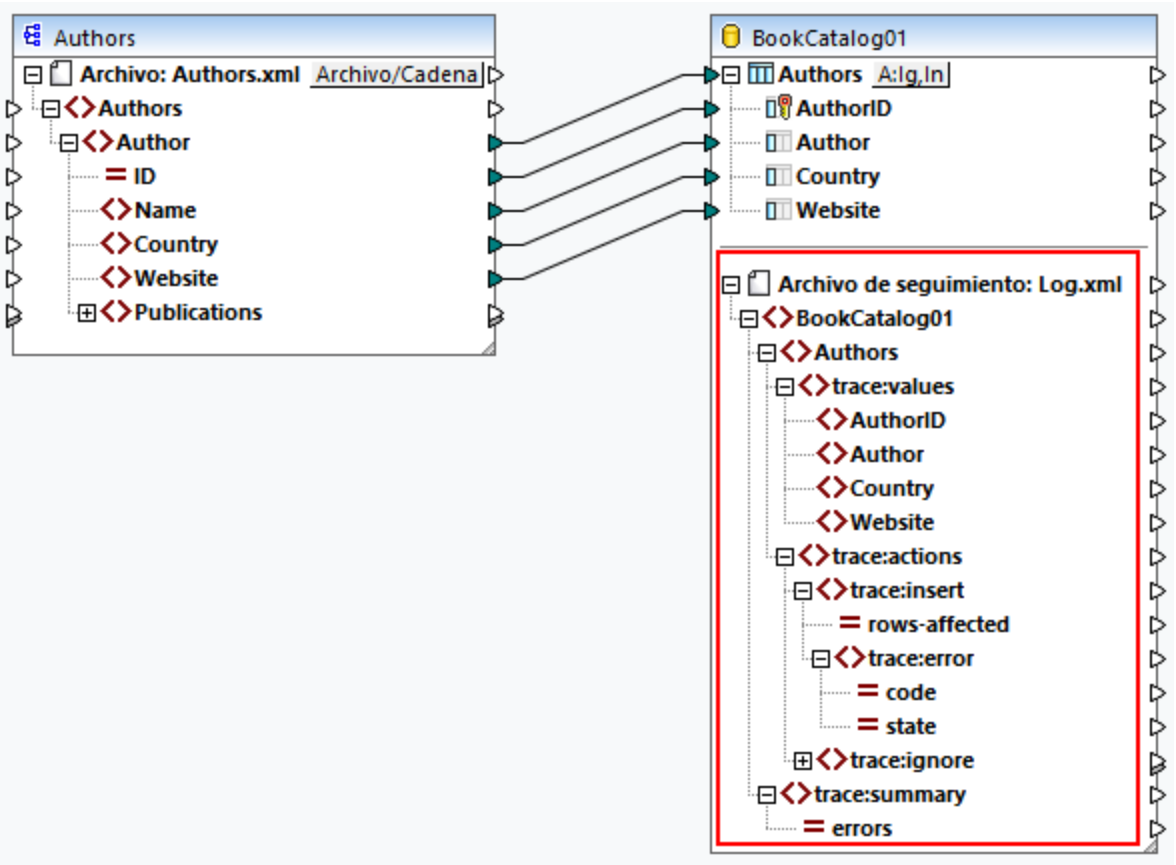
Si se activa el seguimiento, las acciones que ejecute la asignación en la BD se almacenan en un archivo. Puede escoger entre registrar todas las acciones, solo los errores o no registrar nada.

Archivo de seguimiento

Indica el archivo en el que se guarda la información del seguimiento una vez se ejecute la asignación. 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*. El archivo de seguimiento está en formato XML. 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 (p.ej., un archivo de texto, otra base de datos, etc.).

*Estructura de un 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 (*ver imagen siguiente*).

 © 2019-2025 Altova GmbH




Como se ve en la imagen anterior, el nodo superior de la estructura de seguimiento indica el nombre del archivo de seguimiento (`Log.xml`). 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 elemento superior tiene el mismo nombre que la base de datos. El elemento `BookCatalog01` tiene dos elementos secundarios: `Authors` y `trace:summary`. El elemento `Authors` refleja la estructura de la tabla que se añade al componente de BD. El elemento `trace:summary` incluye un atributo `errors`, que informa del número de errores encontrados.

El elemento `Authors` tiene dos elementos secundarios: `trace:values` y `trace:actions`. La estructura `trace:values` muestra todas las columnas de la tabla de BD. Por defecto, todas las columnas están incluidas en el seguimiento, pero puede configurar de qué columnas se debe hacer seguimiento a nivel de archivo de BD (*véase más arriba*). En el caso de los procedimientos almacenados esta estructura muestra los parámetros del procedimiento almacenado.

El elemento `trace:actions` incluye informaciones sobre todas las acciones definidas para esta tabla de BD particular. En nuestro ejemplo, se han configurado dos acciones para la tabla `Authors`: *Omitir si* e *Insertar restantes*. Cada acción monitorizada 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. Este elemento 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 diferentes bases de datos.

*Archivo de seguimiento en el panel Resultados*
Para obtener una vista previa del archivo de seguimiento, haga clic en la pestaña Resultados. Tenga en cuenta que el informe de seguimiento que se muestra en el panel Resultados sólo tiene fines informativos y no refleja los resultados reales de la ejecución. Para generar un informe de seguimiento real, debe ejecutar el script SQL 274 desde el panel Resultados. A continuación se puede ver un ejemplo de un archivo de seguimiento:

```
<BookCatalog01>
      <Authors>
            <trace:values>
                  <Author>Neil Gaiman</Author>
                  <Website>www.neilgaiman.com</Website>
            </trace:values>
            <trace:actions>
                  <trace:ignore/>
            </trace:actions>
      </Authors>
      <Authors>...</Authors>
      <Authors>...</Authors>
      <trace:summary errors="0"/>
</BookCatalog01>
```

Para más información sobre el seguimiento, consulte el *primer caso* en el apartado Reversión de transacciones: Casos 288.

## 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 Rutas de acceso relativas y absolutas 44.

## Usar conexión de BD compartida en tiempo de ejecución

Esta opción le permite elegir si quiere que varios componentes de BD que usan la misma fuente y función de datos compartan también la misma conexión de base de datos. Por defecto, esta opción está deshabilitada ya que podría influir en el proceso de la asignación, sobre todo cuando se comparte la misma conexión entre uno o varios componentes de origen y un componente de destino.

Compartir la misma conexión a la base de datos permite resolver varios problemas, por ejemplo los bloqueos de tablas/filas, el aislamiento de transacciones y el número de conexiones al servidor (*consulte los detalles a continuación*).

- Cuando intenta actualizar una fila de una tabla mientras la está leyendo, puede ocurrir que se encuentre con un error de bloqueo de tabla/fila (dependiendo del proveedor). Puede evitar este problema con una conexión de base de datos compartida.
- Con la conexión compartida activada, se podrán leer filas ya modificadas que estén envueltas en una transacción. En cambio, con conexiones separadas, sólo son visibles los cambios comprometidos.
- La conexión compartida también ayuda a reducir el número de inicios de sesión en la base de datos, lo que le permitirá reducir el tiempo total de procesamiento de las asignaciones que tengan muchos componentes de BD que utilicen la misma fuente de datos. Es útil saberlo ya que el procedimiento de registro o de inicio de sesión en la BD puede llevar mucho tiempo, especialmente cuando las

 © 2019-2025 Altova GmbH

instancias del servidor de la nube están conectadas a una red lenta o cuando el servidor de la BD está ocupado.

### 4.2.2.2 Instrucciones SELECT personalizadas

MapForce permite crear instrucciones SELECT personalizadas de SQL con o sin parámetros. Estas instrucciones se muestran como estructuras tipo tabla de las que se pueden asignar datos a otros componentes. Por ejemplo, puede crear una instrucción personalizada para combinar tablas, filtrar su datos de la BD y definir parámetros que pueden aceptar valores de otros componentes de la asignación.

Las instrucciones SQL SELECT sin parámetros son compatibles con los lenguajes C++, C#, Java y BUILT-IN. Las instrucciones SQL SELECT con parámetros de entrada sólo son compatibles con el lenguaje de transformación BUILT-IN.

*Crear/Editar/Quitar una instrucción SELECT*
Para agregar una instrucción SELECT a un componente de BD, siga las instrucciones siguientes:

1. Haga clic con el botón derecho en el encabezado del componente de BD y seleccione **Agregar/Editar/Quitar objetos de BD** en el menú contextual. También puede seleccionar el componente de BD y seleccionar el comando **Componente | Agregar/Editar/Quitar objetos de BD**.
2. En el cuadro de diálogo **Agregar/Editar/Quitar objetos de BD**, escoja una de las siguientes opciones:
   - Para introducir una instrucción SELECT personalizada, haga clic en el botón **Agregar/Editar instrucciones SELECT**.
   - Para generar una instrucción SELECT para una tabla particular, haga clic con el botón derecho en la tabla relevante y seleccione **Generar y agregar una instrucción SELECT** del menú contextual. Podrá editar la instrucción generada más adelante.

Para editar una instrucción SELECT que ya existe, escoja una de las siguientes opciones:

- Haga clic con el botón derecho en la instrucción SELECT en el componente y seleccione **Editar instrucciones SELECT**.
- Haga clic con el botón derecho en el componente de BD y seleccione **Agregar/Editar/Quitar objetos de BD** en el menú contextual. Después, haga doble clic en la instrucción SELECT en cuestión en el cuadro de diálogo **Agregar/Editar/Quitar objetos de BD**.
- En el cuadro de diálogo **Agregar/Editar/Quitar objetos de BD**, seleccione la instrucción SELECT en cuestión y haga clic en **Agregar/Editar instrucciones SELECT**.
- En el cuadro de diálogo **Agregar/Editar/Quitar objetos de BD**, haga clic con el botón derecho en la instrucción SELECT en cuestión y seleccione **Editar una instrucción SELECT**.

Para quitar una instrucción SELECT, siga los pasos siguientes:

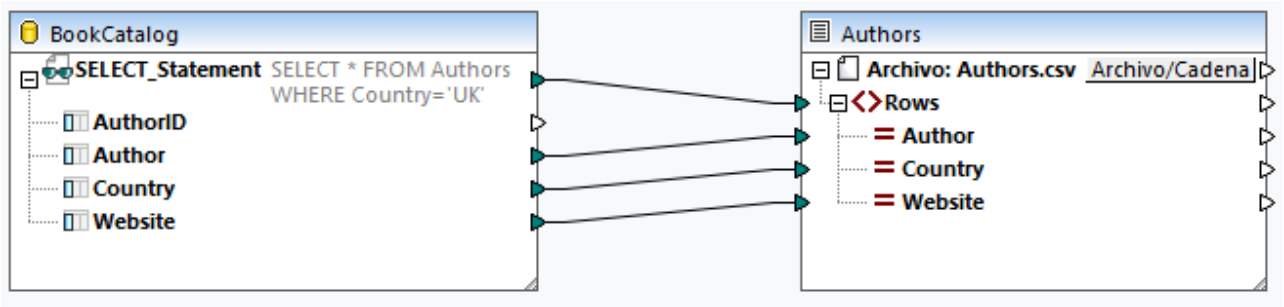
1. Haga clic con el botón derecho en el componente de BD y seleccione **Agregar/Editar/Quitar objetos de BD**.
2. Haga clic con el botón derecho en la instrucción SELECT que quiere eliminar y seleccione **Quitar instrucciones SELECT** en el menú contextual.

*Importante:*
Tenga en cuenta que:

- Todas las expresiones en la instrucción SELECT deben tener un nombre de correlación único (p.ej., `SELECT *, (Quantity*UnitPrice) AS Price`) para que se puedan usar en una asignación.
- Si se conecta a una base de datos Oracle o IBM DB2 con JDBC, la instrucción `SELECT` no puede tener un punto y coma al final.

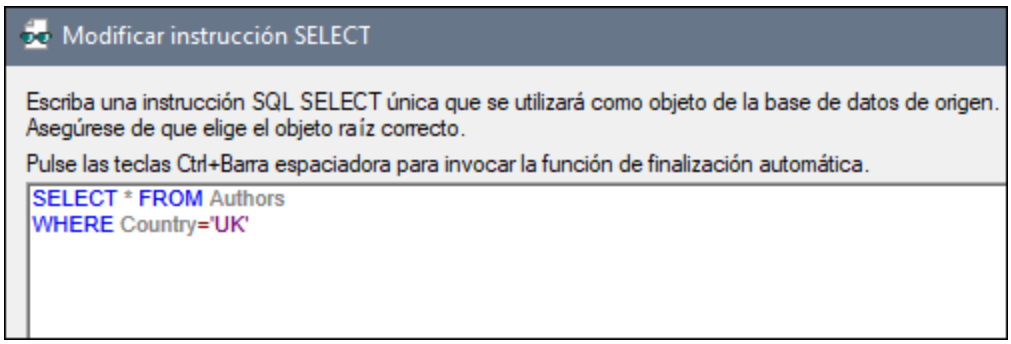
## Instrucciones SELECT de SQL sin parámetros

En el ejemplo que se puede ver a continuación, mostramos cómo trabajar con instrucciones SELECT personalizadas sin parámetros. En la asignación de datos que se puede ver a continuación, asignamos datos de BD a un archivo de texto. La base de datos **`BookCatalog.sqlite`** tiene una tabla principal llamada `Authors` y una tabla secundaria llamada `Books`. Sin embargo, sólo la instrucción SELECT con estructura jerárquica se mostrará en el componente. La estructura de la jerarquía depende de la consulta SQL que se define en el cuadro de diálogo **Introducir una instrucción SQL SELECT**. Las tablas `Authors` y `Books` no están en el componente porque no se asignan datos de estas mismas.



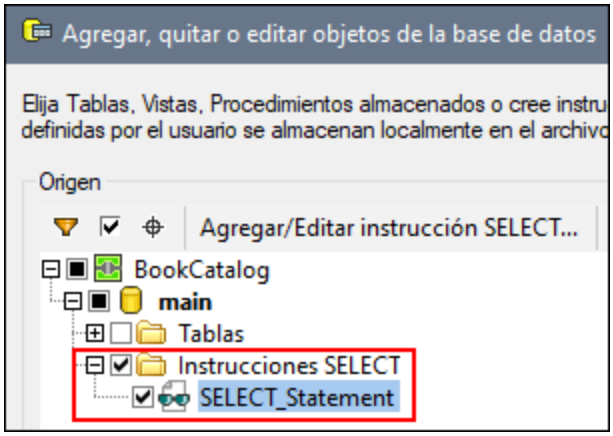

*Instrucción SELECT*

Para el componente de BD, hemos añadido la siguiente instrucción SELECT (*consulte las instrucciones en el apartado Crear/Editar/Quitar una instrucción SELECT*):




La instrucción SQL selecciona todas las tablas de la tabla `Authors` y filtra los datos de BD para incluir sólo autores y autoras del Reino Unido. Cuando se agregue esta instrucción al cuadro de diálogo **Introducir una instrucción SQL SELECT**, esta estará disponible en el cuadro de diálogo **Agregar/Editar/Quitar objetos de BD** (*ver imagen siguiente*). También se puede ver la instrucción en el componente de BD (*ver la asignación anterior*). La cantidad de líneas visibles de las instrucciones SELECT se puede configurar en el cuadro de diálogo Opciones 1128 (en la opción *Limitar vista de anotaciones*).

 © 2019-2025 Altova GmbH

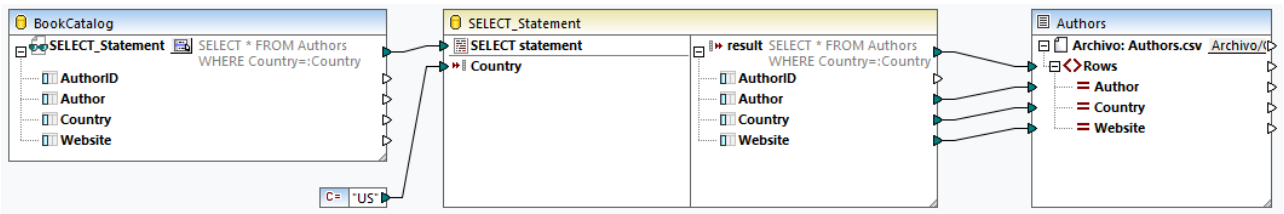



*Resultados*
Los resultados muestran una lista de valores separados por comas que incluyen autores y autoras del Reino Unido (*véase el extracto de código más abajo*).

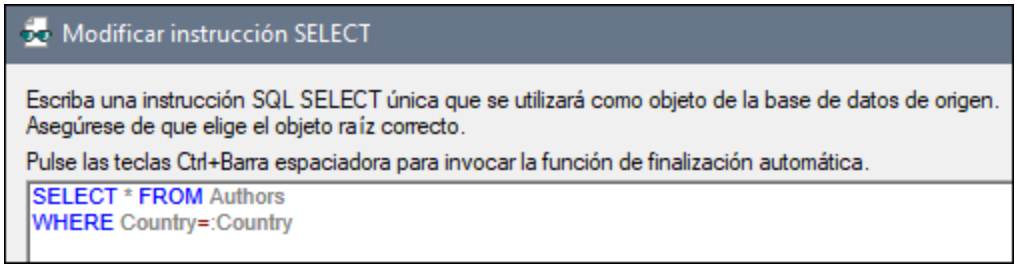
```
Autor/a, País, Página web
Bram Stoker,UK,www.bramstoker.org
Charles Dickens,UK,www.charlesdickensinfo.com
Emily Brontë,UK,n/a
James Herbert,UK,www.james-herbert.co.uk
Neil Gaiman,UK,www.neilgaiman.com
Terry Pratchett,UK,www.terrypratchettbooks.com
Agatha Christie,UK,www.agathachristie.com
Roald Dahl,UK,www.roalddahlfans.com
David Walliams,UK,www.worldofdavidwalliams.com
Kenneth Grahame,UK,n/a
Philip Pullman,UK,www.philip-pullman.com
J.K. Rowling,UK,www.jkrowling.com
Ann Cleeves,UK,www.anncleeves.com
```

## Instrucciones SELECT de SQL con parámetros

Nuestro segundo ejemplo ilustra una asignación en la que el componente de BD incluye una instrucción SELECT personalizada con un parámetro (*ver imagen siguiente*).




*Instrucción SELECT*
Para el componente `BookCatalog`, hemos añadido la siguiente instrucción SQL:




La instrucción usa el parámetro `Country`. Este parámetro aceptará valores de la constante (*bajo el componente BookCatalog*). Para poder asignar datos de la instrucción SELECT con el parámetro, haga clic en el botón junto al nodo `SELECT_Statement` en el componente de BD (*ver asignación anterior*) y seleccione **Insertar llamada con parámetros** del menú contextual. De esta manera se inserta un componente de llamada con parámetros (*el componente central de la asignación anterior*). El componente de llamada tiene dos partes: La parte izquierda acepta parámetros de entrada (en nuestro caso, `Country`), y la parte derecha replica la instrucción SELECT con la estructura jerárquica del componente de BD. Después, los datos filtrados se asignan al archivo de texto `Authors`.

*Resultados*
Ahora, los resultados muestran autores y autoras de EEUU (*ver el extracto de código a continuación*).

```
Autor/a, País, Página web
Stephen King,US,www.stephenking.com
Frank Herbert,US,n/a
Isaac Asimov,US,www.asimovonline.com
Blake Crouch,US,www.blakecrouch.com
Ray Bradbury,US,www.raybradbury.com
Joe Hill,US,www.joehillfiction.com
Josh Malerman,US,www.joshmalerman.com
George R. R. Martin,US,www.georgerrmartin.com
A. J. Finn,US,n/a
Dan Brown,US,www.danbrown.com
Dean Koontz,US,www.deankoontz.com
```

## Archivos de ejemplo

Para más información sobre asignaciones de datos que usan instrucciones SQL SELECT como entrada, consulte los siguientes ejemplos en la carpeta **`MapForceExamples`**:

- **`DB_EmployeeListByTitle.mfd`**
- **`DB_MostExpensiveArticle.mfd`**
- **`DB_ManagerList_AllOffices.mfd`**
- **`DB_ManagerList_SelectedDepartment.mfd`**
- **`DB_ManagerList_SelectedOffice.mfd`**

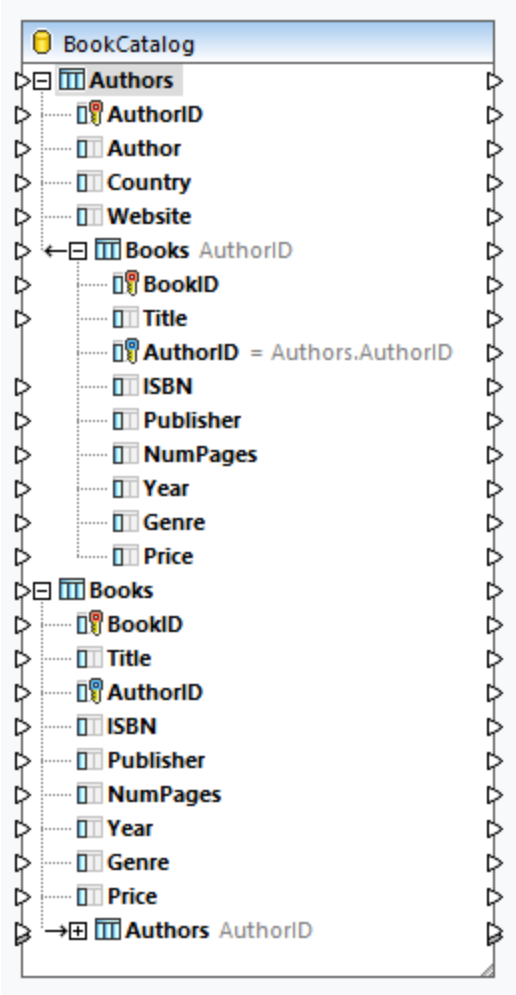
# 4.2.2.3 Relaciones de BD

Después de agregar una base de datos 246 como componente de origen a su asignación de datos, cada tabla aparece como la tabla raíz (*ver imagen siguiente*). Cuando hace clic en el icono plus de una tabla raíz, puede

 © 2019-2025 Altova GmbH

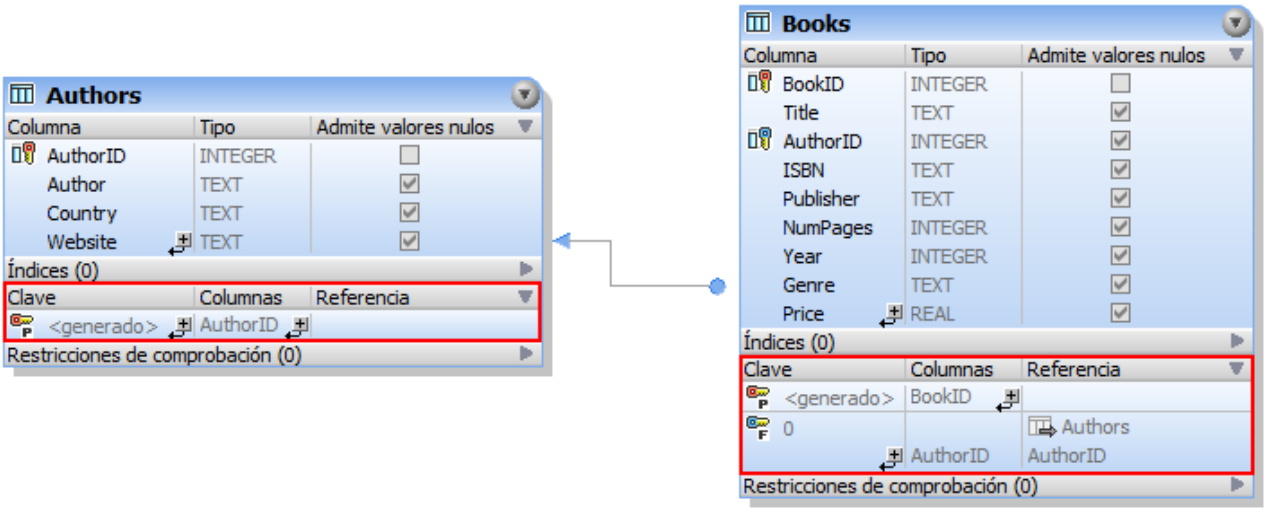
ver todas las tablas relacionadas bajo la tabla raíz. El componente de BD que se ve a continuación muestra dos tipos de flechas que tienen el siguiente significado:

- La flecha hacia la izquierda ( ← ) indica que la tabla `Books` es una tabla secundaria de la tabla `Authors`.
- La flecha hacia la derecha ( → ) muestra que la tabla `Authors` es la tabla principal de la tabla `Books`.




*Structure of BookCatalog.sqlite*

Puede usar varios tipos de asignaciones de datos, dependiendo de sus necesidades empresariales. A continuación presentamos algunos casos posibles. Todos los casos descritos a continuación presentan una base de datos jerárquica llamada **`BookCatalog.sqlite`**. La base de datos tiene dos tablas (`Authors` y `Books`) que tienen una relación de clave foránea. En la imagen siguiente se puede ver que la tabla `Books` tiene una clave foránea llama `AuthorID` que hace referencia a la clave primaria en la tabla `Authors`.




*Datos de muestra de BookCatalog.sqlite*

A continuación se pueden ver extractos de las tablas `Authors` y `Books`:

| | AuthorID | Author | Country | Website |
|---|---|---|---|---|
| 1 | 1 | Stephen King | US | www.stephenking.com |
| 2 | 2 | Ragnar Jonasson | Iceland | www.ragnarjonasson.com |
| 3 | 3 | Bram Stoker | UK | www.bramstoker.org |
| 4 | 4 | Charles Dickens | UK | www.charlesdickensinfo.com |
| 5 | 5 | Fyodor Dostoyevsky | Russia | n/a |
| 6 | 6 | Emily Brontë | UK | n/a |
| 7 | 7 | Frank Herbert | US | n/a |

*Tabla Authors*

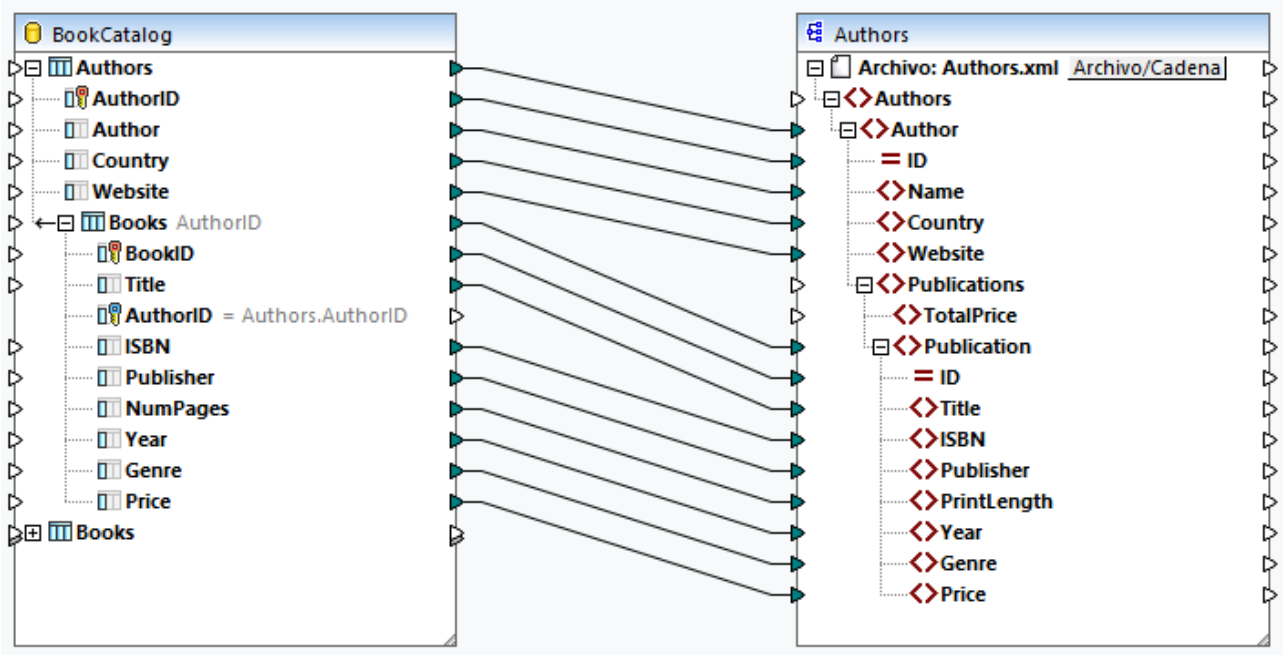
| | BookID | Title | AuthorID | ISBN | Publisher | NumPages | Year | Genre | Price |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | Misery | 1 | 1501143107 | Scribner | 368 | 2016 | Horror | 11.99 |
| 2 | 2 | Nightblind | 2 | 9781910633113 | Orenda Books | 231 | 2016 | Crime & Mystery | 9.99 |
| 3 | 3 | Blackout | 2 | 1910633461 | Orenda Books | 276 | 2016 | Crime & Mystery | 8.49 |
| 4 | 4 | Outsider | 1 | 1501180983 | Scribner | 576 | 2018 | Horror | 12.79 |
| 5 | 5 | Dracula | 3 | 9781435142817 | Barnes & Noble | 512 | 2013 | Classics | 13.69 |
| 6 | 6 | The Mystery of Edwin Drood | 4 | 9781400043286 | Everyman's Library | 336 | 2004 | Classics | 19.79 |
| 7 | 7 | Crime and Punishment | 5 | 0679420290 | Everyman's Library | 608 | 1993 | Classics | 14.99 |
| 8 | 8 | Wuthering Heights | 6 | 979-8469527794 | Independently published | 200 | 2021 | Classics | 22.99 |
| 9 | 9 | Dune (Dune Chronicles, Book 1) | 7 | 0441013597 | Penguin Publishing Group | 704 | 2005 | Sci-Fi | 14.99 |
| 10 | 10 | I, Robot | 8 | 978-0-00-749151-3 | Harper Collins | 256 | 2013 | Sci-Fi | 11.39 |
| 11 | 11 | Dark Matter | 9 | 9781447297581 | Pan Macmillan | 416 | 2017 | Sci-Fi | 9.99 |
| 12 | 12 | Nos4a2 | 11 | 978-1-4732-2641-8 | Orion | 704 | 2019 | Horror | 16.99 |

*Tabla Books*

 © 2019-2025 Altova GmbH

## Caso 1: Preservar la jerarquía

En nuestro primer caso, asignamos datos desde **`BookCatalog.sqlite`** a **`Authors.xsd`** (*ver imagen siguiente*). En esta asignación, `Authors` es la tabla raíz. El objetivo es preservar la relación jerárquica y obtener todos los autores con sus libros correspondientes en los resultados.



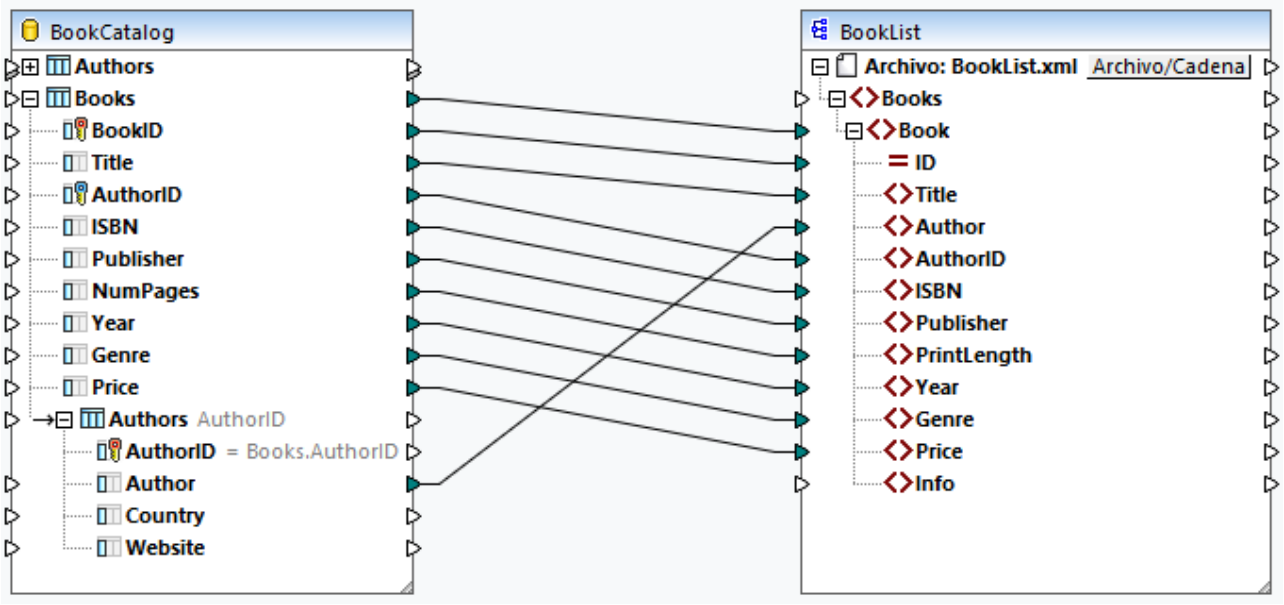

El siguiente extracto de código muestra una parte de los resultados:

```
<Authors>
      <Author ID="23">
            <Name>Fredrik Backman</Name>
            <Country>Sweden</Country>
            <Website>www.fredrikbackmanbooks.com</Website>
            <Publications>
                  <Publication ID="26">
                        <Title>Anxious People</Title>
                        <ISBN>978-1-4059-3025-3</ISBN>
                        <Publisher>Penguin Books Ltd</Publisher>
                        <PrintLength>416</PrintLength>
                        <Year>2021</Year>
                        <Genre>Humor</Genre>
                        <Price>9.99</Price>
                  </Publication>
                  <Publication ID="27">
                        <Title>A Man Called Ove</Title>
                        <ISBN>9781444775815</ISBN>
                        <Publisher>Sceptre</Publisher>
                        <PrintLength>320</PrintLength>
                        <Year>2015</Year>
                        <Genre>Humor</Genre>
                        <Price>11.46</Price>
```

```
                    </Publication>
                </Publications>
            </Author>
        </Authors>
```

## Caso 2: Intercambiar las tablas

En el segundo caso, el objetivo es obtener una lista de libros y sus detalles en el archivo de salida. Para conseguir el objetivo, usaremos `Books` como tabla raíz. Se mantendrán las relaciones de tabla. El diseño de la asignación tiene este aspecto:




El siguiente extracto de código muestra una parte de los resultados:

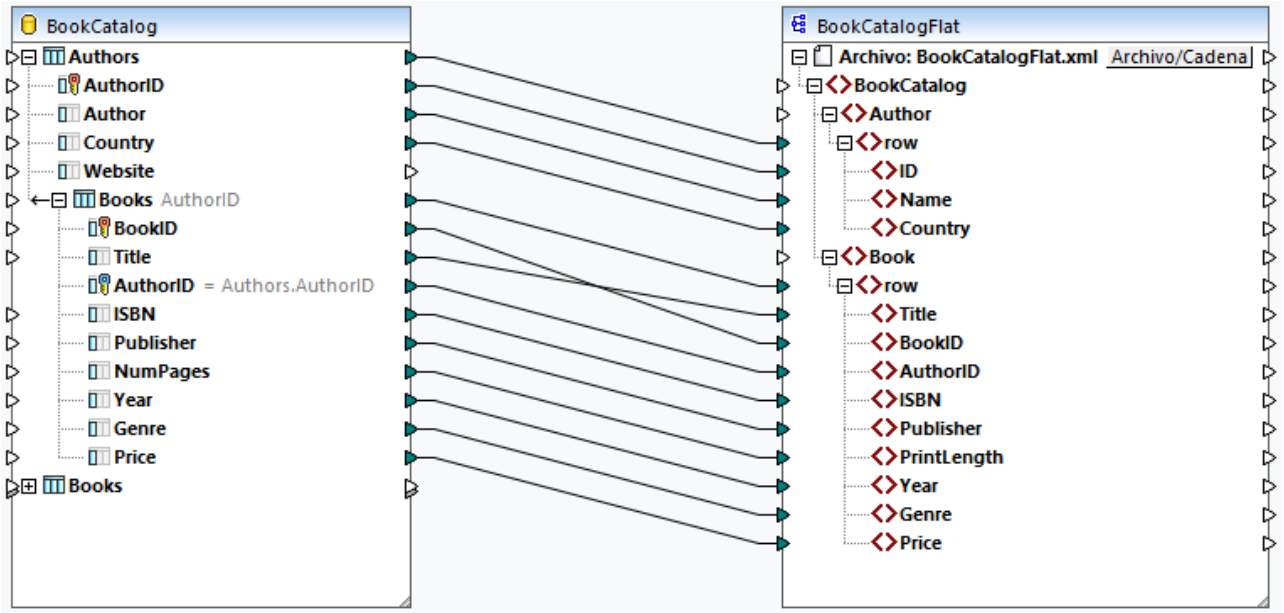
```
<Books>
    <Book ID="3">
        <Title>Blackout</Title>
        <Author>Ragnar Jonasson</Author>
        <AuthorID>2</AuthorID>
        <ISBN>1910633461</ISBN>
        <Publisher>Orenda Books</Publisher>
        <PrintLength>276</PrintLength>
        <Year>2016</Year>
        <Genre>Crime &amp; Mystery</Genre>
        <Price>8.49</Price>
    </Book>
    <Book ID="4">
        <Title>Outsider</Title>
        <Author>Stephen King</Author>
        <AuthorID>1</AuthorID>
        <ISBN>1501180983</ISBN>
        <Publisher>Scribner</Publisher>
        <PrintLength>576</PrintLength>
        <Year>2018</Year>
        <Genre>Horror</Genre>
```

 © 2019-2025 Altova GmbH

```
            <Price>12.79</Price>
        </Book>
    </Books>
```

## Caso 3: Asignar datos de BD desde diferentes tablas raíz

En el tercer caso, asignaremos datos desde cada tabla raíz del componente de BD a **`Authors.xsd`** (*ver imagen siguiente*). Se omitirán las tablas relacionadas.




Como resultado, cada libro, independientemente de su autor/a, aparecerá bajo cada autor/a (*ver el siguiente extracto de código*).

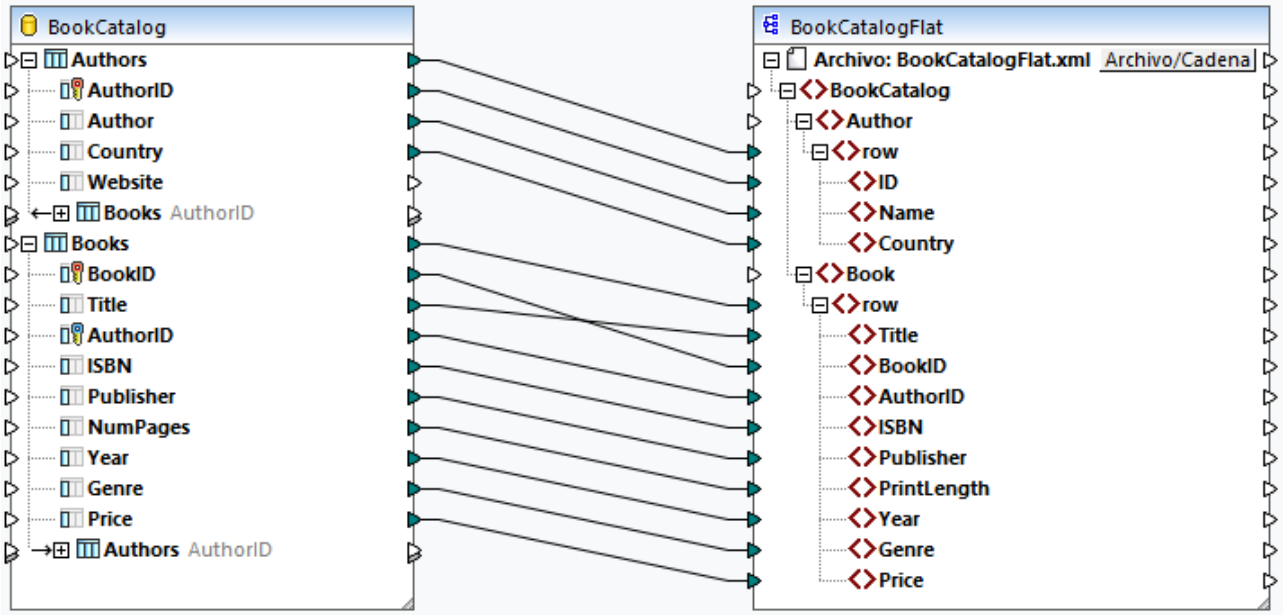
```
    <Author ID="19">
        <Name>Sebastian Fitzek</Name>
        <Country>Germany</Country>
        <Website>www.sebastianfitzek.com</Website>
        <Publications>
            <Publication ID="1">
                <Title>Misery</Title>
                <ISBN>1501143107</ISBN>
                <Publisher>Scribner</Publisher>
                <PrintLength>368</PrintLength>
                <Year>2016</Year>
                <Genre>Horror</Genre>
                <Price>11.99</Price>
            </Publication>
            <Publication ID="2">
                <Title>Nightblind</Title>
                <ISBN>9781910633113</ISBN>
                <Publisher>Orenda Books</Publisher>
                <PrintLength>231</PrintLength>
                <Year>2016</Year>
                <Genre>Crime &amp; Mystery</Genre>
                <Price>9.99</Price>
```

```
            </Publication>
            <Publication ID="3">...</Publication>
            <Publication ID="4">...</Publication>
            <Publication ID="5">...</Publication>
            <Publication ID="6">...</Publication>
            <Publication ID="7">...</Publication>
            <Publication ID="8">...</Publication>
      </Publications>
</Author>
```

## Caso 4: Asignar datos de BD a estructuras SQL/XML

En el cuarto caso, el objetivo es asignar datos de BD a una estructura de esquema plana/nivelada (estándar SQL/XML). El modelo de esquema plano se basa en la especificación ISO-ANSI SQL/XML INCITS/ISO/IEC 9075-14-2008. La especificación SQL/XML define cómo se asignan datos de BD a XML. Las relaciones se definen en esquemas por medio de restricciones de identidad y no hay referencias a los elementos. Por tanto, el esquema tiene una estructura plana o nivelada, similar a la vista en forma de árbol de la BD. Puede comprar la especificación en la tienda web de ANSI. Para más información, consulte www.iso.org.

La siguiente asignación muestra cómo se asignan datos de BD desde diferentes tablas raíz a una estructura plana o nivelada SQL/XML. Se omiten las tablas relacionadas. También es posible asignar datos de BD desde tablas relacionadas. Sin embargo, si algunos registros `Book` no son de ningun/a `Author`, estos registros `Book` no se asignarán al destino.




Como resultado, obtendremos una lista de filas `Author` y una lista separada de filas `Book` (*ver imagen siguiente*).

```
<Author>
      <row>
            <ID>1</ID>
            <Name>Stephen King</Name>
            <Country>US</Country>
      </row>
```

 © 2019-2025 Altova GmbH

```
        <row>
              <ID>2</ID>
              <Name>Ragnar Jonasson</Name>
              <Country>Iceland</Country>
        </row>
        <row>...</row>
        <row>...</row>
  </Author>
  <Book>
        <row>
              <Title>Misery</Title>
              <BookID>1</BookID>
              <AuthorID>1</AuthorID>
              <ISBN>1501143107</ISBN>
              <Publisher>Scribner</Publisher>
              <PrintLength>368</PrintLength>
              <Year>2016</Year>
              <Genre>Horror</Genre>
              <Price>11.99</Price>
        </row>
        <row>
              <Title>Nightblind</Title>
              <BookID>2</BookID>
              <AuthorID>2</AuthorID>
              <ISBN>9781910633113</ISBN>
              <Publisher>Orenda Books</Publisher>
              <PrintLength>231</PrintLength>
              <Year>2016</Year>
              <Genre>Crime &amp; Mystery</Genre>
              <Price>9.99</Price>
        </row>
        <row>...</row>
        <row>...</row>
  </Book>
```

Para más información sobre este caso, consulte también la siguiente asignación: **MapForceExamples\DB_Altova_SQLXML.mfd**.

## 4.2.2.4 Relaciones locales

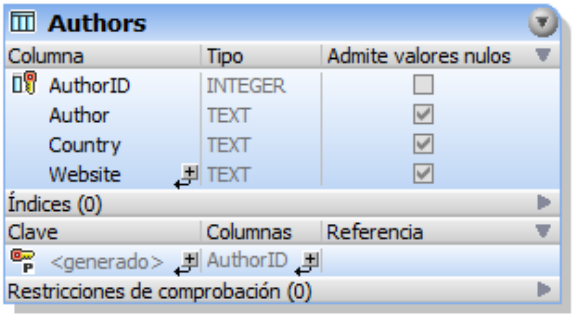
Cuando las tablas de BD no tienen relaciones entre sí, puede crear relaciones de clave primaria y foránea entre columnas de distintas tablas directamente en MapForce (relaciones locales). Como claves primarias y foráneas se puede usar cualquier columna de la base de datos. Además, se pueden crear relaciones nuevas además de las que ya existen en la base de datos. Las relaciones definidas a nivel local desde MapForce se guardan con la asignación.

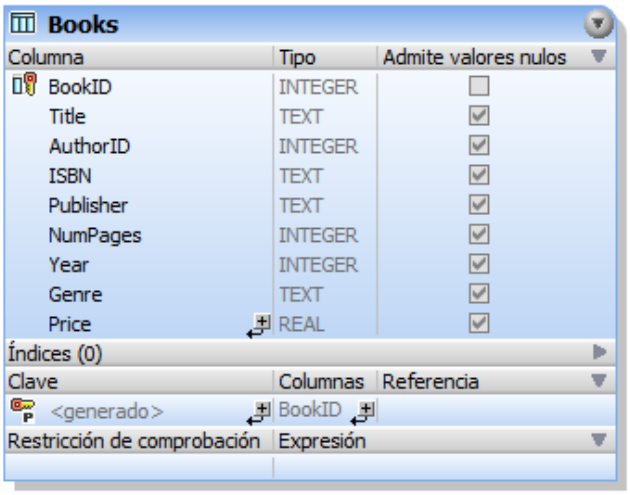
La siguiente tabla enumera todos los campos posibles entre los que puede definir relaciones locales. Es posible crear relaciones mixtas (p.ej., asignar los resultados de un procedimiento almacenado a una columna de BD). Los tipos de datos de los campos que forman parte de la relación deben ser iguales o compatibles.

| Clave principal/única | Clave foránea |
|---|---|
| • Columna de una tabla o vista de BD<br>• Parámetro de salida o valor de retorno de un procedimiento almacenado (consulte también Procedimientos almacenados 315)<br>• Columna de conjunto de registros devuelta por procedimiento almacenado. 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.<br>• Columna de una instrucción SELECT definida por el usuario (consulte también Instrucciones SQL SELECT como tablas virtuales 259). | • Columna de una tabla o vista de BD<br>• Parámetro de entrada de un procedimiento almacenado<br>• Parámetro de entrada de una instrucción SELECT definida por el usuario |

## Ejemplo

La tabla de datos **`BookCatalogNoRelation.sqlite`** tiene dos tablas: `Authors` y `Books` (*ver imagen siguiente*). En este momento todavía no existe ninguna relación de clave foránea entre las tablas.




*Componente de BD sin relaciones*

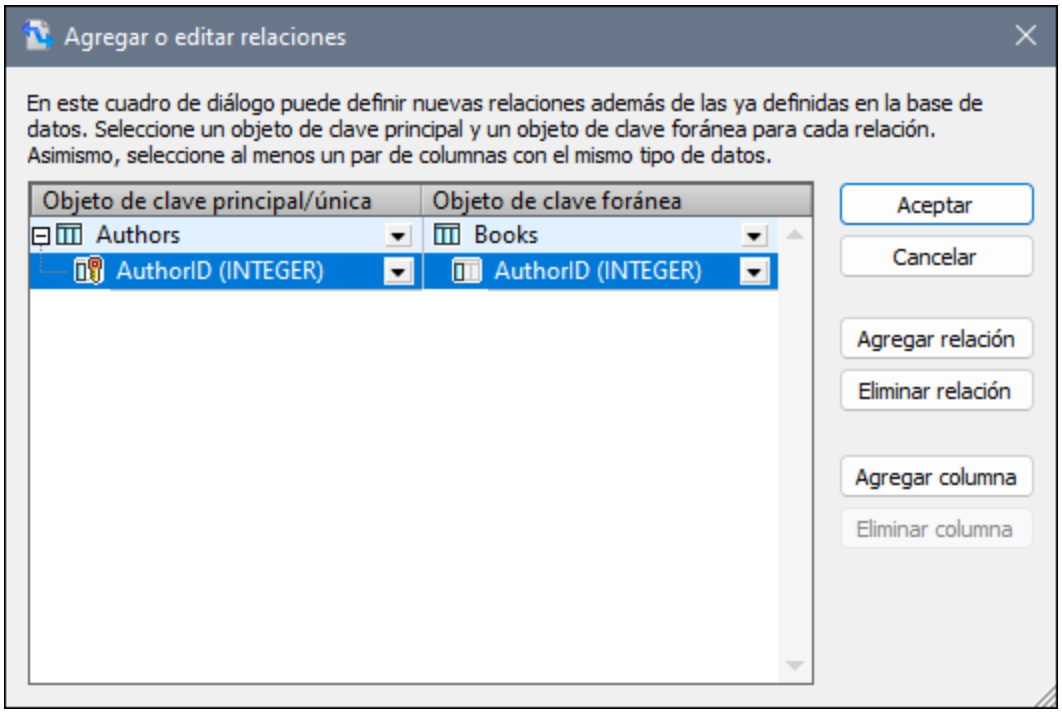
El componente de BD se muestra de la siguiente manera al insertarlo en la asignación:


© 2019-2025 Altova GmbH

*Definición de relaciones locales*

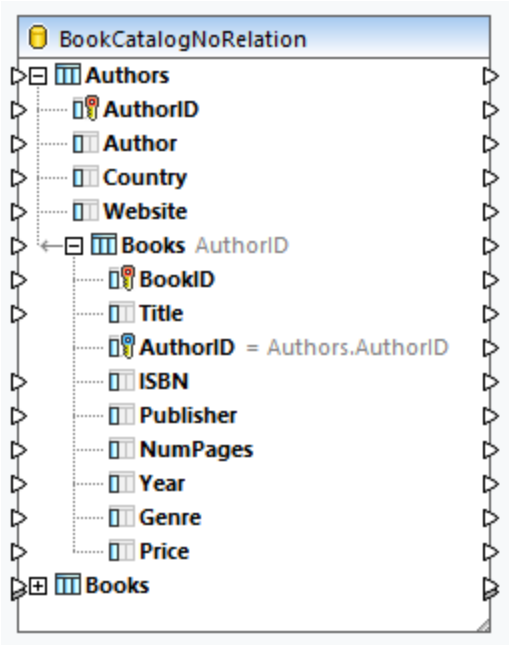
En este ejemplo el objetivo es hacer referencia a la tabla `Authors` en la tabla `Books`. Para ello siga estos pasos:

1. Haga clic con el botón derecho en el componente de BD y seleccione **Agregar/Editar/Quitar objetos de BD** en el menú contextual.
2. Haga clic en el botón **Agregar/Editar relaciones** en el cuadro de diálogo **Agregar/Editar/Quitar objetos de la base de datos**.
3. Haga clic en **Agregar relación** en el cuadro de diálogo **Agregar/Editar relaciones** (*imagen siguiente*).
4. Haga clic en *[seleccionar objeto]* en la columna *Objeto de clave primaria/única* y seleccione `Authors`. Ahora seleccione `AuthorID` en la lista desplegable *[seleccionar columna]*.
5. Haga clic en *[seleccionar objeto]* en la columna *Objeto de clave foránea* y seleccione `Books`. Ahora seleccione `AuthorID` en la lista desplegable *[seleccionar columna]*.
6. Haga clic en **Aceptar** para terminar de crear la definición de relación local.




*Componente de BD con relaciones*

Una vez haya terminado de definir las relaciones locales, el componente de BD se habilita en el área de asignación (*ver imagen siguiente*). El componente muestra dos posibles estructuras de BD. En cada una de ellas la tabla raíz es distinta. Por ejemplo, en la estructura ampliada de más abajo, `Authors` es la tabla raíz. En función de lo que necesite, puede asignar datos desde y a cualquiera de las estructuras del componente. También puede combinar tablas de distintas estructuras en el componente. Para más información sobre estos escenarios, consulte el apartado Relaciones de BD 262.




 © 2019-2025 Altova GmbH

## 4.2.2.5 Funciones relativas a BD

Cuando quiera trabajar con bases de datos, puede resultar necesario usar varias funciones para gestionar valores NULL, generar valores secuenciales y únicos, así como reemplazar caracteres especiales. Para más información siga leyendo.

### Gestionar valores NULL

A continuación, se pueden ver las funciones para gestionar valores NULL en MapForce:

- Para comprobar en tiempo de ejecución de la asignación si un campo de la base de datos es NULL, puede usar las funciones **is-null** 640 y **is-not-null** 640. Para ver si una tabla tiene campos NULL, puede consultarla en el Explorador de BD en MapForce (ver el Panel de consulta de BD 294).
- Para establecer un campo de base de datos como NULL, puede usar la función **set-null** 641.
- Para reemplazar valores de base de datos NULL con una cadena, puede usar la función **substitute-null** 641.

Para más información sobre cómo gestionar valores NULL en una base de datos, consulte *Null equal* 278. Véase también el apartado *Valores NULL en componentes de BD* 146.

### Generar valores secuenciales y únicos

Cuando se actualizan registros de BD, puede ser necesario crear valores secuenciales o únicos de forma instantánea para los campos de base de datos que no reciben datos de entrada de ningún componente de origen. En estos casos puede usar las siguientes funciones:

- La función **auto-number** 574 puede utilizarse para generar valores de clave principal.
- La función **create-guid** 667 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 BD 276 y puede ser muy práctica a la hora de generar claves principales.

### Reemplazar caracteres especiales

Cuando quiera actualizar datos de una base de datos, primero puede ser necesario quitar ciertos caracteres especiales (p.ej., los caracteres de retorno de carro o de salto de línea)- En estos casos puede usar los siguientes métodos:

1. Puede definir una función de nodo para uno o varios campos especificados que quiere procesar. La función de nodo recibirá el valor de un campo de BD como entrada, procesará este valor y luego lo devolverá en la asignación de datos. Para más información sobre este método, consulte Valores predeterminados y funciones 469.
2. También se pueden procesar valores de BD con la ayuda de funciones integradas de MapForce. Por ejemplo, para identificar caracteres especiales, incluyendo caracteres de control, puede usar la función **char-from-code** 623. Para reemplazar valores, puede usar la función **Reemplazar** 691.

### 4.2.2.6 Uso eficiente de los recursos de 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. Consulte también Funciones definidas por el usuario.[500] 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[393].
- 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 de combinación[408] (en modo SQL) o una instrucción SQL-SELECT[259]. 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 componentes de combinación no SQL[397]. 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[437] en vez de un filtro estándar, ya que en la gramática de la BD correspondiente el componente anterior está optimizado para que funcione específicamente con bases de datos.

Para obtener información sobre el contexto y las reglas y estrategias generales, consulte Reglas y estrategias básicas[80].

## 4.2.3 Acciones de tabla de BD

Cuando usa una base de datos como componente de destino, puede configurar varias acciones de tablas de BD. Por ejemplo, puede insertar todos los registros de un archivo de origen a su base de datos. También puede elegir cuándo quiere actualizar, eliminar u omitir registros. En este apartado se ofrece un resumen de todas las acciones disponibles y se muestran algunos de los escenarios posibles de usar acciones de tabla.

### Instrucciones SQL en los resultados

Cuando se asignan datos a una BD y se consulta la vista previa de resultados de una asignación en el panel Resultados, se obtiene un script SQL. El script muestra instrucciones pseudo SQL de carácter puramente informativo. No aplique el pseudo script SQL manualmente a la base de datos con herramientas SQL distintas a los motores de ejecución que se pueden ver a continuación: MapForce, MapForce Server[839] (independientemente o bajo la dirección de FlowForce Server[842]) o el entorno de ejecución del código generado para C++, C# o Java. El script en el panel Resultados puede contener formatos de valores no interpretables por editores SQL externos.

 © 2019-2025 Altova GmbH

Si quiere aplicar cambios a la base de datos desde MapForce directamente, abra el panel Resultados y haga clic en el comando **Ejecutar script SQL/NoSQL** de la barra de herramientas o del menú **Resultados**. Esta acción modificará la base de datos inmediatamente.

Cuando la asignación se ejecuta con MapForce Server (independientemente o bajo la dirección de FlowForce Server), los cambios se realizan inmediatamente en la base de datos. Lo mismo ocurre en el código generado: Los cambios de la base de datos se realizan cuando se compila y ejecuta el código (p.ej. con el comando **Ejecutar** de Visual Studio).

*Importante:*
Su instalación de MapForce incluye varios archivos de base de datos de ejemplo, que están disponibles en la carpeta **`MapForceExamples`**. No se recomienda modificar ninguna base de datos en esta carpeta porque puede que tras la modificación estas bases de datos sean inservibles. Lo más recomendable es crear una copia de seguridad de la carpeta **`MapForceExamples`** antes de actualizar ningún archivo de esta carpeta.

## Nota sobre MySQL/MariaDB ODBC

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;`).

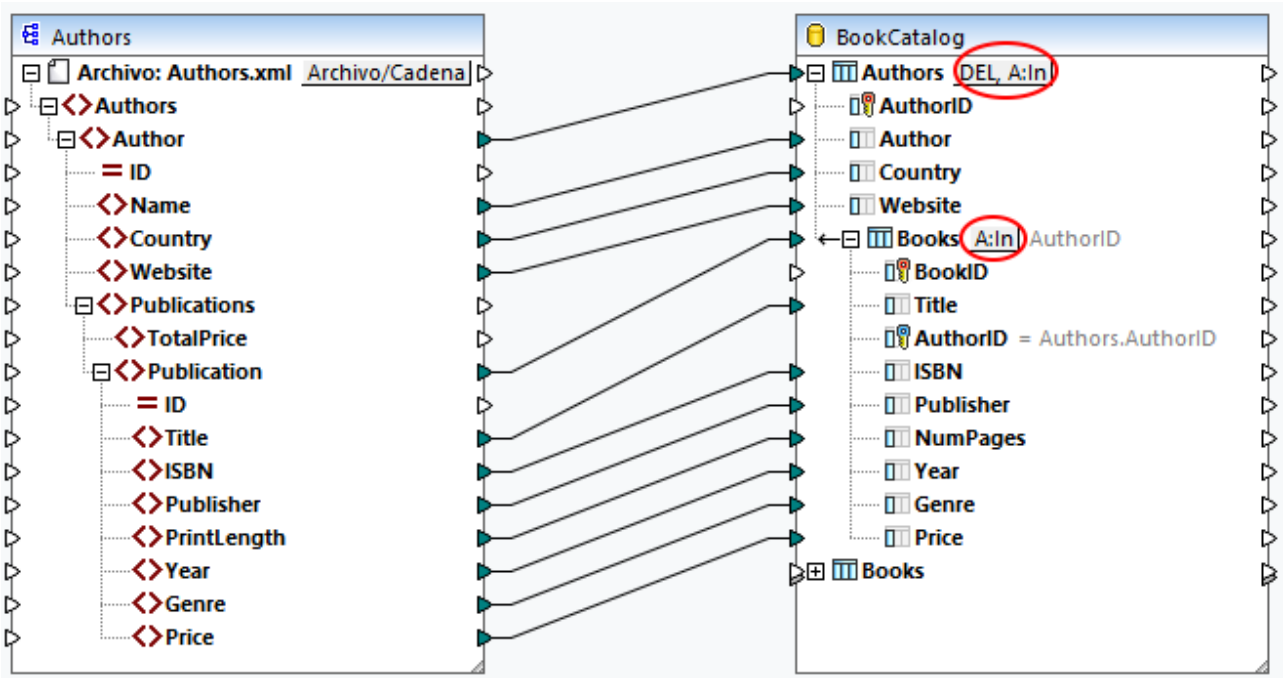
Para habilitar esta opción desde el conector MySQL ODBC, siga los siguientes pasos:

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** (*ver a continuación*).

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*.

### 4.2.3.1 Acciones de tabla de BD: Configuración

Cuando se asignan datos a una tabla de BD, junto a esta encontrará el botón **Acciones de BD**. En la imagen siguiente se ve una asignación de datos a dos tablas de BD, y cada tabla tiene su propio botón **Acciones de BD** (*rodeado de un círculo rojo en la imagen siguiente*).

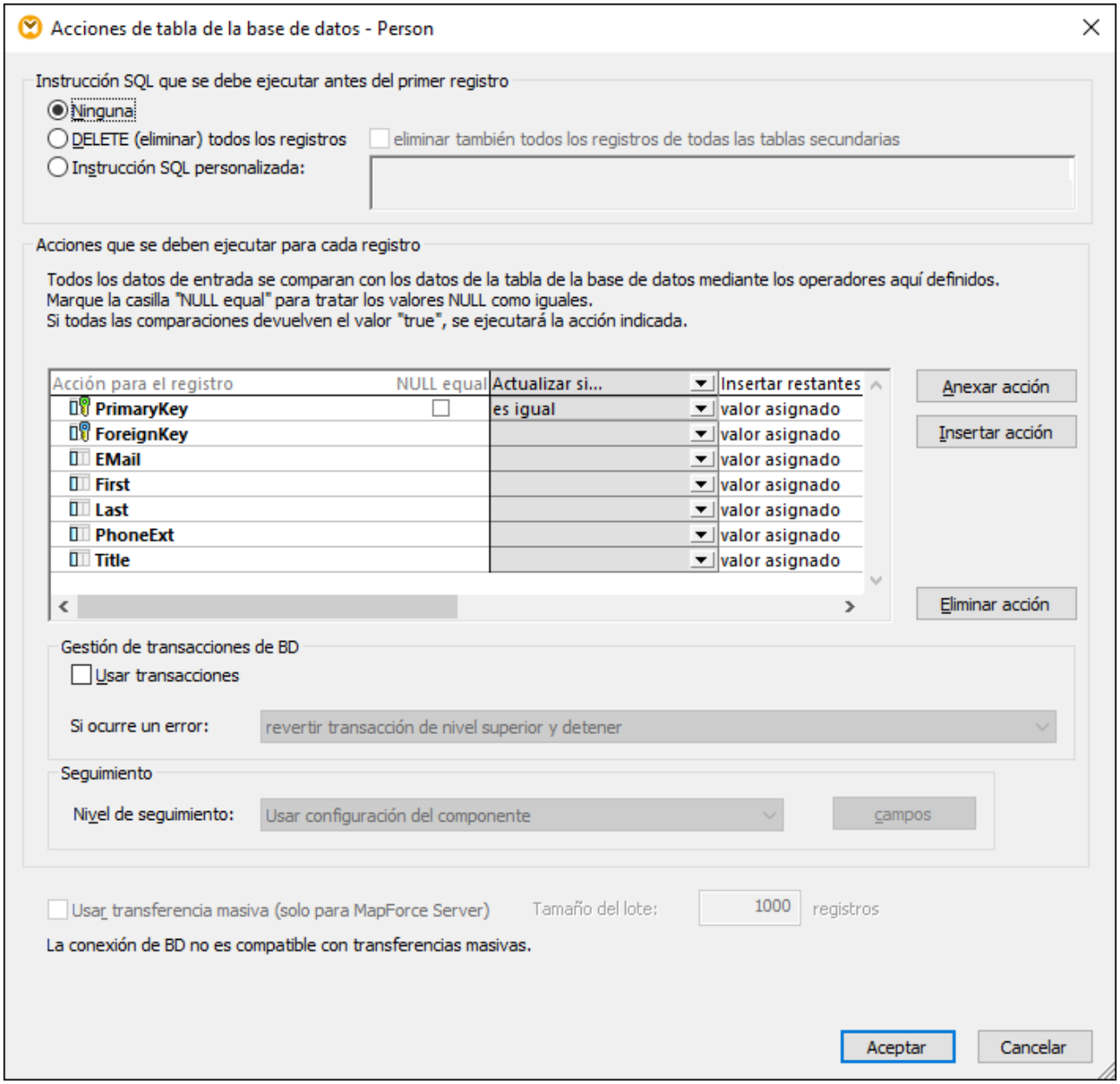



Al hacer clic en el botón **Acciones de BD**, se abre el cuadro de diálogo **Acciones de tabla de BD** (*ver a continuación*), en el que puede configurar varias acciones y opciones. El cuadro de diálogo **Acciones de tablas de BD** consiste en cinco partes:

1. Acciones que se deben ejecutar antes que las acciones definidas para cada registro
2. Acciones definidas para cada registro
3. Configuración de transacciones de BD
4. Configuración del seguimiento y del registro de errores
5. Configuración de la transferencia por lotes

Para ver más informaciones sobre cada parte del cuadro de diálogo, sigue leyendo.


© 2019-2025 Altova GmbH




## Ejecutar una instrucción SQL antes del primer registro

En este apartado puede definir instrucciones SQL que se ejecutarán *antes* que las acciones definidas en el apartado *Acciones que se deben ejecutar para cada registro*. Estas son las opciones disponibles:

- La opción *Ninguno/a* significa que no se realizará ninguna acción. Esta es la configuración predeterminada.
- La opción *DELETE (eliminar) todos los registros* eliminará todos los registros de la tabla secundaria. Además, también puede decidir eliminar todos los registros en todas las tablas secundarias (la casilla *eliminar también todos los registros en todas las tablas secundarias*).

- La configuración *Instrucción SQL personalizada* permite escribir una instrucción personalizada que afecte a toda la tabla. Por ejemplo, puede agregar una instrucción que dará información del seguimiento sobre una asignación. 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.

Para más información sobre astas acciones, consulte Acciones de tablas de BD: Escenarios 283.

## Acciones que se deben ejecutar para cada registro

El apartado *Acciones que se deben ejecutar para cada registro* permite definir acciones de BD que se ejecutarán para cada registro en su base de datos. Para gestionar las acciones de la tabla, haga clic en los botones **Anexar acción**, **Insertar acción** o **Eliminar acción**. Si es necesario, también se pueden definir acciones múltiples. Cualquier acción que se defina después de la primera acción *Insertar todo* o *Insertar restantes* no se ejecutará nunca porque las condiciones siguientes no se pueden cumplir. Si se ha agregado una acción de tabla después de la acción *Insertar todo* o *Insertar restantes*, aparece un cuadro de diálogo informando de que se eliminará la acción de tabla posterior.

Para cada acción se comparan todos los datos de entrada con los de la BD. Si no se encuentra ninguna diferencia, se ejecutará una acción específica. Las acciones de tabla definidas se procesan de izquierda a derecha. Por ejemplo, si ha definido primero una condición *Actualizar si* y luego una condición *Insertar restantes*, la acción *Actualizar si* se procesará primero. Si no se cumple la condición *Actualizar si*, pues se ejecutará la acción *Insertar restantes*. Si no se cumple ninguna condición, no se ejecutará ninguna acción.

*valor asignado/generado por la BD/max() + 1*
Al configurar las acciones *Insertar todo* o *Insertar restantes*, puede elegir cómo se deben generar los valores. Estas son las opciones disponibles: *valor asignado*, *generado por la BD* y *max() + 1*. La opción *valor asignado* permite asignar los datos de origen al campo de base de datos directamente. Se trata de la opción estándar para todos los campos de BD.

Al configurar el incremento automático para la(s) clave(s) primaria(s) en su base de datos, puede elegir si prefiere trabajar con un valor asignado o un valor generado por la BD. Si no se ha habilitado el incremento automático, puede elegir entre un valor asignado y la opción *max() + 1*. La opción `max() + 1` genera valores numéricos a partir de los valores que ya existen en la base de datos. Por ejemplo, si la tabla tiene tres registros con claves principales `1`, `2` y `3`, entonces la clave primaria del registro nuevo será `4`.

*NULL equal*
Si marca la casilla *NULL equal* junto a un registro (*ver imagen siguiente*), los valores NULL en el elemento de origen y de destino se considerarán iguales. No marcar esta casilla puede llevar a resultados incorrectos. Puede marcar la casilla *NULL equal* si se cumplen *todas* estas condiciones:

- El campo para el que desea activar la opción *NULL equal* acepta valores NULL. Esto significa que este campo se ha configurado de tal forma que se pueden introducir valores NULL.
- Se han configurado una o varias de las acciones siguientes: *Omitir si*, *Actualizar si* o *Eliminar si*.
- Una o varias acciones de la tabla (como *Omitir si*) tienen al menos una condición *es igual* o *es igual (omitir may/min)* (ver el *campo Author en la imagen siguiente*).

 © 2019-2025 Altova GmbH

| Acción para el registro | NULL equal | Omitir si... | Insertar restantes |
|---|---|---|---|
| AuthorID | | | generada por la BD |
| Author | ☐ | es igual | valor asignado |
| Country | | | valor asignado |
| Website | ☑ | es igual | valor asignado |

La imagen anterior muestra que se han configurado dos acciones para la tabla `Authors`: *Omitir si* e *Insertar restantes*. La condición *Omitir si* compara los valores `Author` y `Website` del origen con los de la BD. Si estos valores son iguales, estos registros se omitirán en la BD, y los registros de origen que no tengan equivalente en la BD simplemente se insertarán en ella.

La opción *NULL equal* se ha habilitado para la columna `Página web`. Por ejemplo, uno de los registros `Author` tiene un valor NULL en el campo `Website`, tanto en el registro de origen como de destino. Si la opción *NULL equal* está habilitada, se omitirá este registro `Author` y no se insertará en la base de datos. Sin embargo, si no se ha marcado la casilla *NULL equal*, el registro ya no cumplirá la condición *Omitir si* y se insertará en la BD.

*Tablas secundarias*
Si en su base de datos hay registros vinculados por una relación de clave foránea, podrá ver los nombres de tablas secundarias en el cuadro de diálogo **Acciones de tablas de BD** (*Books en el cuadro de diálogo Acciones de tablas de BD anterior*). Aparecerán las siguientes opciones:

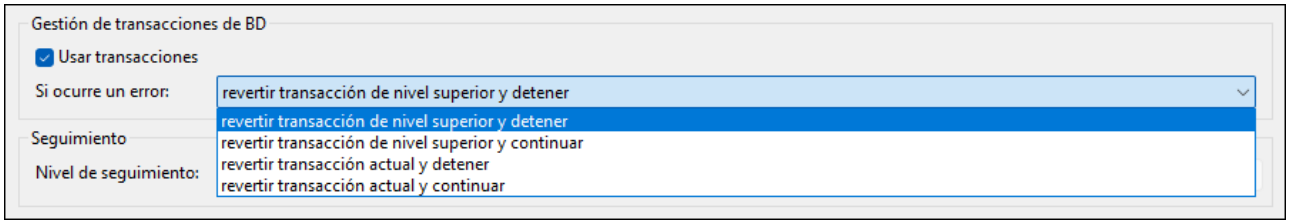
- *Eliminar datos en tablas secundarias:* Esta opción puede resultar muy útil cuando configura, por ejemplo, la acción *Actualizar si* para una tabla principal y la acción *Insertar todo* para su tabla secundaria. Si habilita la condición *Actualizar si*, se actualizarán sólo los registros que existen tanto en el origen como en la base de datos. Gracias a la condición *Actualizar si*, puede evitar registros duplicados en la tabla principal.

  La opción *Eliminar datos en tablas secundarias* borrará sólo aquellos registros `Book` cuyos registros principales (`Authors`) cumplan la condición *Actualizar si* (es decir que existen en el origen y en la BD). Puede evitar registros duplicados o huérfanos en la tabla secundaria eliminando registros secundarios. La acción *Insertar todo* definida para la tabla secundaria insertará sólo aquellos registros secundarios de origen cuyos registros principales cumplan la condición *Actualizar si*.

- *Omitir datos secundarios de entrada:* Esta opción es útil cuando quiere actualizar solamente una tabla principal sin cambiar sus registros secundarios.

## Gestión de transacciones de BD

El apartado *Gestión de transacciones de BD* le permite revertir una operación a una secuencia de operaciones (transacción) cuando ocurren errores relacionados a la BD (p.ej., si se insertan valores NULL en columnas no-NULL). Para habilitar las transacciones a nivel de acciones de tabla marque la casilla *Usar transacciones* en el cuadro de diálogo **Acciones de tabla de BD** (*ver imagen siguiente*).




Existen varias opciones de gestión de transacciones:

- *Revertir transacción de nivel superior y detener la operación:* Puesto que la gestión de transacciones se puede configurar a distintos niveles de la jerarquía, la *transacción de nivel superior* puede hacer referencia a (i) la transacción a nivel de componente de BD 251, si es que se han habilitado a ese nivel, o (ii) la transacción definida la tabla de nivel más alto en la jerarquía. Los cambios que se han realizado en la BD se revertirá en cada nivel de la jerarquía hasta el nivel superior, y luego se detendrá la ejecución.
- *Revertir transacción de nivel superior y continuar:* Ocurre lo mismo, pero la asignación sigue ejecutándose después de la reversión de la transacción (p.ej., para procesar otro componente de destino si existe).
- *Revertir transacción actual y detener la operación:* Si ocurre un error relacionado con la BD, se revertirán únicamente los cambios contenidos en la transacción actual, y se detiene el procesamiento de los siguientes registros. Los cambios realizados previamente fuera de la transacción actual se conservan.
- *Revertir transacción actual y continuar la operación:* Ocurre lo mismo, pero la asignación sigue ejecutándose después de la reversión de la transacción.

Para más información sobre varios casos de gestión de transacciones, consulte el apartado Reversión de transacciones: Casos 288.

## Seguimiento

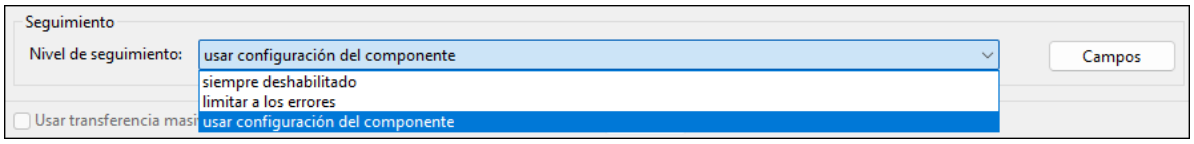
Cuando una asignación escribe datos en una BD se puede habilitar el seguimiento de BD y el registro de errores. El seguimiento es útil para monitorizar todos los cambios que la asignación ha hecho en la base de datos durante la ejecución de la asignación. Los cambios que se han realizado en la BD se almacenan en un archivo de seguimiento. Si ocurren errores durante la ejecución, estos también se almacenan.

Para más información sobre la estructura de un informe de seguimiento, consulte el apartado Archivo de seguimiento 255. Para ver un ejemplo en el que se ha habilitado el seguimiento, consulte el *primer caso* en el apartado Reversión de transacciones: Casos 292.

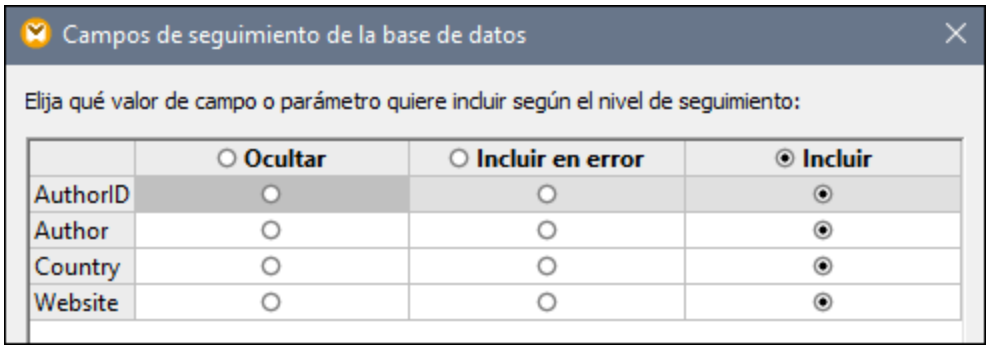
Para habilitar el seguimiento a nivel de tabla, siga estos pasos:

1. Asegúrese de que el nivel de seguimiento a nivel de componentes de DB 255 se ha configurado en *Siempre* o *Error*.
2. Haga clic en el botón **Acción de tabla** junto a la tabla para la que quiere habilitar el seguimiento.
3. Seleccione uno de los siguientes niveles de seguimiento en la sección *Seguimiento* de las **acciones de tabla de BD** (*imagen siguiente*): (i) la opción *Siempre deshabilitado* significa que no se hará ningún seguimiento para esa tabla; (ii) la opción *Limitar a los errores* restringe el seguimiento a los errores solamente; (iii) la opción *Usar configuración del componente* hereda la configuración que se definió a nivel del componente.

 © 2019-2025 Altova GmbH




*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, todos los campos (columnas de BD) se incluyen automáticamente en el informe de seguimiento (*ver imagen siguiente*). Si quiere que el informe de seguimiento incluya únicamente información de seguimiento sobre campos de BD específicos, haga clic en el botón **Campos** del cuadro de diálogo **Acciones de tabla de BD**. Puede decidir ocultar campos, incluirlos siempre o sólo en caso de error.




## Usar transferencia por lotes

La opción *Transferencia por lotes* permite ejecutar varias instrucciones INSERT en una consulta. El uso de esta opción acelera considerablemente el proceso de inserción, ya que sólo es necesario ejecutar una instrucción en lugar de muchas. La opción *Transferencia por lotes* se configura en MapForce, pero la transferencia por lotes de los datos realmente ocurre cuando se ejecuta una asignación de datos en MapForce Server.

*Requisitos*
La opción de la transferencia por lotes solamente es compatible si se cumplen estas dos condiciones:

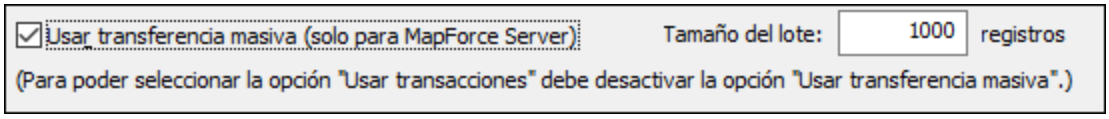
- El lenguaje de la asignación de transformación es el motor de ejecución integrado (BUILT-IN).
- La asignación de datos se ejecuta en MapForce Server 839 (independientemente o bajo control de FlowForce Server 842).
- La licencia de MapForce Server no está limitada a la ejecución por subprocesos simples en un equipo con varios núcleos. Es decir que la opción *Limit to single thread execution (limitar a la ejecución por subprocesos simples)* en la pestaña Server Management de Altova LicenseServer se debe deshabilitar.
- La acción *Insertar todo* se configura para la tabla de BD en cuestión.
- La tabla a la que quiere transferir datos con la inserción masiva, no puede tener ni tablas o vistas relacionadas ni procedimientos almacenados que hagan referencia a ella en la asignación.
- El controlador de BD es compatible con la operación inserción masiva en condiciones WHERE.

La compatibilidad de la operación Inserción masiva depende del tipo y del controlador de BD que se usa (*ver siguiente tabla*).

| Tipo de BD | ADO | ODBC | JDBC | ADO.NET | Nativo |
|---|---|---|---|---|---|
| *Acceso* | 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 *(se requiere MySQL Versión 5 o superior)* | Sí | No | n/a |
| *Oracle* | No | Sí | Sí | Sí | n/a |
| *PostgreSQL* | n/a | Sí | Sí | n/a | Sí |
| *Progreso* | n/a | Sí | Sí | n/a | n/a |
| *Servidor SQL* | 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 |

**Nota:** Para habilitar la compatibilidad de la inserción masiva con MySQL y MariaDB a través de JDBC, use la opción de conexión `rewriteBatchedStatements=true`.

*BD de muestra*
Para probar la operación de la inserción masiva, puede usar el servidor SQL y la base de datos AdventureWorks. Cuando carga esta base de datos en su asignación y abre el cuadro de diálogo **Acciones de tablas de BD**, podrá especificar el tamaño de lote (en nuestro ejemplo es de 1000 registros). El tamaño de lote define el numero de registros que se podrán insertar a la vez. Es importante tener en cuenta que las opciones *Transferencia por lotes* y *Transacciones* se excluyen mutuamente: Al habilitar una de estas opciones, la otra se desactiva automáticamente.




*Siguiente paso*
Ahora que hemos habilitado la opción Inserción masiva, podremos ejecutar la asignación en MapForce Server[839] (independientemente o bajo control de FlowForce Server[842]).

 © 2019-2025 Altova GmbH

## 4.2.3.2 Acciones de tabla de BD: Escenarios

**Sitio web de Altova:** Vídeos de demostración de MapForce

En esta sección se ilustrarán algunos casos posibles de usar acciones de tablas de BD[276]. Todos los casos usan una base de datos jerárquica llamada **`BookCatalog.sqlite`**. Esta base de datos tiene tres tablas: `Authors` (principal), `Books` (secundaria), `TrackingInfo` (no está conectada a otras tablas). Las tablas `Authors` y `Books` están vinculadas por una relación de clave foránea. Es importante tener en cuenta que el incremento automático está habilitado para las claves principales en las tablas `Authors` y `Books`. La estructura de la BD es la que aparece a continuación:

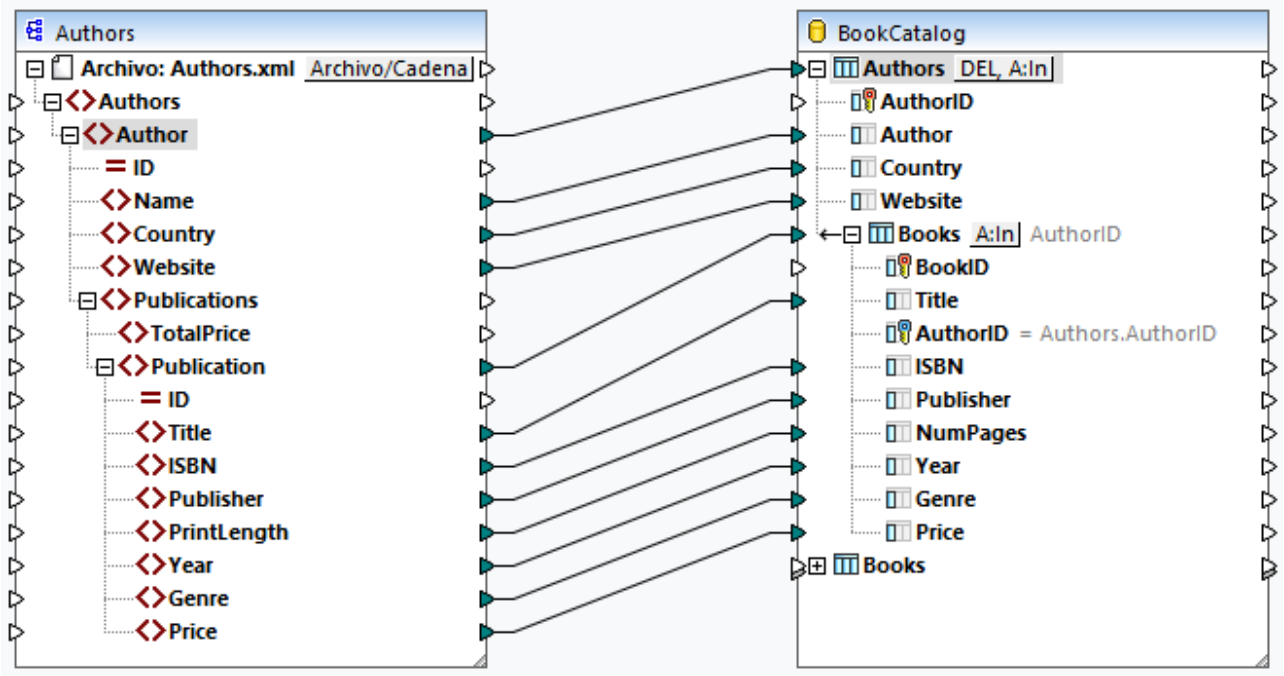
```
CREATE TABLE
       "main"."Authors" (
  "AuthorID" INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT,
       "Author" TEXT,
       "Country" TEXT,
       "Website" TEXT
);

CREATE TABLE
       "main"."Books" (
       "BookID" INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT,
       "Title" TEXT,
       "AuthorID" INTEGER,
       "ISBN" TEXT,
       "Publisher" TEXT,
       "NumPages" INTEGER,
       "Year" INTEGER,
       "Genre" TEXT,
       "Price" DECIMAL,
       FOREIGN KEY ("AuthorID") REFERENCES "Authors" ("AuthorID") ON DELETE CASCADE
ON UPDATE NO ACTION
);

CREATE TABLE
       "main"."TrackingInfo" (
       "MappedOn" DATETIME
);
```

### Caso 1: Eliminar todos los datos de BD e insertar todos los datos de origen

En el primer caso, queremos eliminar *todos* los datos de la BD **`BookCatalog.sqlite`** y rellenar la base de datos con *todos* los datos del archivo de origen. La asignación tiene este aspecto:




Aunque la base de datos tiene tres tablas, sólo se incluyen las tablas `Authors` y `Books` en el componente de BD. La ausencia de la tabla `TrackingInfo` en el componente se debe a que ningún elemento se asigna a esta tabla. Puesto que se ha configurado el incremento automático para las tablas `Authors` y `Books`, no tenemos que conectar nada a las columnas `AuthorID` y `BookID`: La base de datos generará estos IDs automáticamente.

*Acciones de tablas de BD para Authors*

Las acciones de tablas para la tabla `Authors` (*ver imagen siguiente*) se han configurado de la siguiente manera:

- En el apartado *Ejecutar una instrucción SQL antes del primer registro*, hemos configurado la acción DELETE que eliminará todos los registros de la BD, incluyendo todos los registros secundarios.
- En la sección *Acciones que se deben ejecutar para cada registro*, hemos configurado la acción *Insertar Todo*.
- La base de datos generará automáticamente los IDs de Authors (la opción *generado por la BD* en la acción *Insertar todo*).
- Los demás valores se asignarán desde el archivo de origen.

| Acción para el registro NULL equal | Insertar todo |
|---|---|
| AuthorID | generada por la BD |
| Author | valor asignado |
| Country | valor asignado |
| Website | valor asignado |

*Acciones de tablas de BD para Books*

Para la tabla `Books`, hemos configurado la misma acción *Insertar todo* (*ver imagen siguiente*). La base de datos generará automáticamente los IDs de Book. La columna `AuthorID` hace referencia a la clave primaria en

la tabla `Authors`. Los valores para esta columna se indicarán automáticamente. Todos los demás valores se asignarán desde el archivo de origen.

| Acción para el registro NULL equal | Insertar todo |
|---|---|
| BookID | generada por la BD |
| Title | valor asignado |
| AuthorID | clave foránea |
| ISBN | valor asignado |
| Publisher | valor asignado |
| NumPages | valor asignado |
| Year | valor asignado |
| Genre | valor asignado |
| Price | valor asignado |

*Resultados*
El siguiente extracto de código muestra una parte de los resultados:

```
DELETE FROM "Books"

DELETE FROM "Authors"

INSERT INTO "Authors" ("Author", "Country", "Website") VALUES ('Stephen King', 'US',
'www.stephenking.com')

SELECT "AuthorID" FROM "Authors" WHERE "AuthorID" = last_insert_rowid()
-- >>> %AuthorID1%

INSERT INTO "Books" ("AuthorID", "Title", "ISBN", "Publisher", "NumPages", "Year",
"Genre", "Price") VALUES ('%AuthorID1%', 'Misery', '1501143107', 'Scribner', 368,
2016, 'Horror', 11.99)

INSERT INTO "Books" ("AuthorID", "Title", "ISBN", "Publisher", "NumPages", "Year",
"Genre", "Price") VALUES ('%AuthorID1%', 'Outsider', '1501180983', 'Scribner', 576,
2018, 'Horror', 12.79)
```

Tenga en cuenta que las instrucciones SQL de la salida sólo tienen fines informativos. Para ejecutar las instrucciones SQL, abra el panel Resultados y ejecute el comando **Ejecutar script SQL/NoSQL** de la barra de herramientas. Para más información consulte *Instrucciones SQL en los resultados* 274

## Caso 2: Actualizar autores/autoras y libros, insertar restantes, insertar información del seguimiento

En situaciones reales, la base de datos cambia constantemente, y varias personas podrían trabajar en la misma base de datos. Por tanto, no sería deseable eliminar todos los registros de la BD. En el segundo caso, los objetivos son estos:

- Actualizar autores/as y sus libros, a saber los que están tanto en el archivo de origen como en la base de datos
- Insertar registros de origen que no existen en la base de datos
- Dar información del seguimiento sobre la fecha y la hora del seguimiento

Nuestra asignación de datos tiene los mismos componentes y conexiones que en el primer caso. Sin embargo, las acciones de tablas de BD se han configurado de otra manera (*ver a continuación*).

*Acciones de tablas de BD para Authors*
La imagen siguiente muestra las acciones definidas en la tabla `Authors`. En la condición *Actualizar si*, hemos configurado el valor *igual a* en la columna `Author`. Es decir que si hay autores/as que tengan el mismo nombre tanto en el archivo de origen como en la base de datos, se actualizarán solamente los registros de estos/as autores/as en la base de datos. Autores/as que existen solamente en el archivo de origen simplemente se insertarán en la base de datos. Autores/as que existen solamente en la base de datos no cambiarán.

| Acción para el registro | NULL equal | Actualizar si... | Insertar restantes |
|---|---|---|---|
| AuthorID | | | generada por la BD |
| Author | ☐ | es igual | valor asignado |
| Country | | | valor asignado |
| Website | | | valor asignado |

*Acciones de tablas de BD para Books*
Para la tabla `Books`, disponemos de la misma combinación de acciones (*ver a continuación*). En este caso, el valor "igual a" se configura en la columna `Title`. Es decir que si hay los mismos libros tanto en el archivo de origen como en la base de datos, se actualizarán estos registros de libros en la base de datos. Los libros que existen solamente en el de origen simplemente se insertarán en la base de datos. Los libros que existen solamente en la base de datos no cambiarán.

| Acción para el registro | NULL equal | Actualizar si... | Insertar restantes |
|---|---|---|---|
| BookID | | | generada por la BD |
| Title | ☐ | es igual | valor asignado |
| AuthorID | | clave foránea | clave foránea |
| ISBN | | | valor asignado |
| Publisher | | | valor asignado |
| NumPages | | | valor asignado |
| Year | | | valor asignado |
| Genre | | | valor asignado |
| Price | | | valor asignado |

*No registros duplicados*
La mayor ventaja de la condición *Actualizar si* es que se evitan registros duplicados en la base de datos.

*Resultados*
El siguiente extracto de código muestra una parte de los resultados:

```
UPDATE "Authors" SET "Country" = 'US', "Website" = 'www.stephenking.com' WHERE
("Authors"."Author" = 'Stephen King')

SELECT "AuthorID" FROM "Authors" WHERE ("Author" = 'Stephen King')
-- >>> %AuthorID1%

UPDATE "Books" SET "ISBN" = '1501143107', "Publisher" = 'Scribner', "NumPages" =
368, "Year" = 2016, "Genre" = 'Horror', "Price" = 11.99 WHERE ("Books"."AuthorID" =
'%AuthorID1%') AND ("Books"."Title" = 'Misery')
```

 © 2019-2025 Altova GmbH

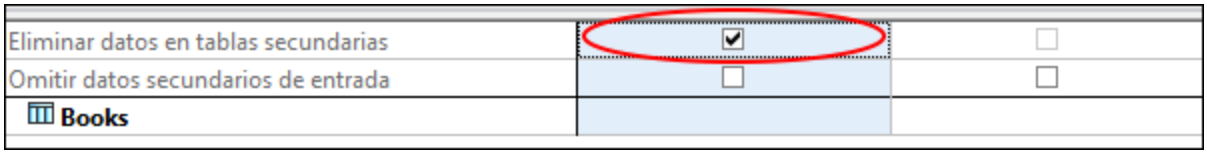
```
UPDATE "Books" SET "ISBN" = '1501180983', "Publisher" = 'Scribner', "NumPages" =
576, "Year" = 2018, "Genre" = 'Horror', "Price" = 12.79 WHERE ("Books"."AuthorID" =
'%AuthorID1%') AND ("Books"."Title" = 'Outsider')
```

*Una solución alternativa*
Para evitar registros duplicados en la tabla `Books`, también puede decidir eliminar tablas secundarias 279 (*ver a continuación*) y configurar la condición *Insertar todo* en la tabla `Books`. Las acciones para la tabla `Authors` no cambiarán. Con esta configuración se eliminarán todos los registros `Book` en la base de datos si estos también existen en el archivo de origen. En caso de que se haya configurado la condición *Insertar todo* en la tabla `Books`, se insertarán los registros `Book` de origen que cumplan estos requisitos:

- Los registros `Book` cuyos autores/as existen tanto en el archivo de origen como en la BD
- Los registros `Book` de nuevos autores/as que existen sólo en el archivo de origen

Si un registro `Book` de origen no tiene ningún registro principal, este también se asignará a la BD, mientras su registro principal se creará en la tabla `Authors`. El nuevo registro `Author` recibirá un nuevo id. (una clave primaria), y los demás campos recibirán valores NULL. Esto es posible *sólo* si todos los campos en la tabla `Authors`, menos el de la clave primaria, aceptan valores NULL. Si los campos no aceptan valores NULL, recibirá un mensaje de error informándole que la restricción NOT NULL ha fallado.

Es importante destacar que los resultados del escenario principal y de soluciones alternativas pueden ser diferentes. Por ejemplo, si un/a autor/a en la BD tiene cinco registros de libros, y el/la mismo/a autor/a sólo tiene tres en el archivo de origen, se eliminarán los cinco registros de la BD y se reemplazarán con los tres del archivo de origen.




*Insertar información del seguimiento*
Cuando hay varias personas que trabajan en la misma base de datos, puede ser beneficioso saber en qué momento se ha ejecutado la última asignación de datos. Puede haber diferentes posibilidades: p.ej., puede usar diferentes funciones `datetime` 643 o insertar una instrucción SQL personalizada en el cuadro de diálogo Acciones de tablas de BD 277. En nuestro ejemplo, agregaremos la siguiente instrucción SQL al apartado *Acciones que se deben ejecutar para cada registro* (tabla `Authors`):

```
INSERT INTO TrackingInfo VALUES (DATETIME())
```

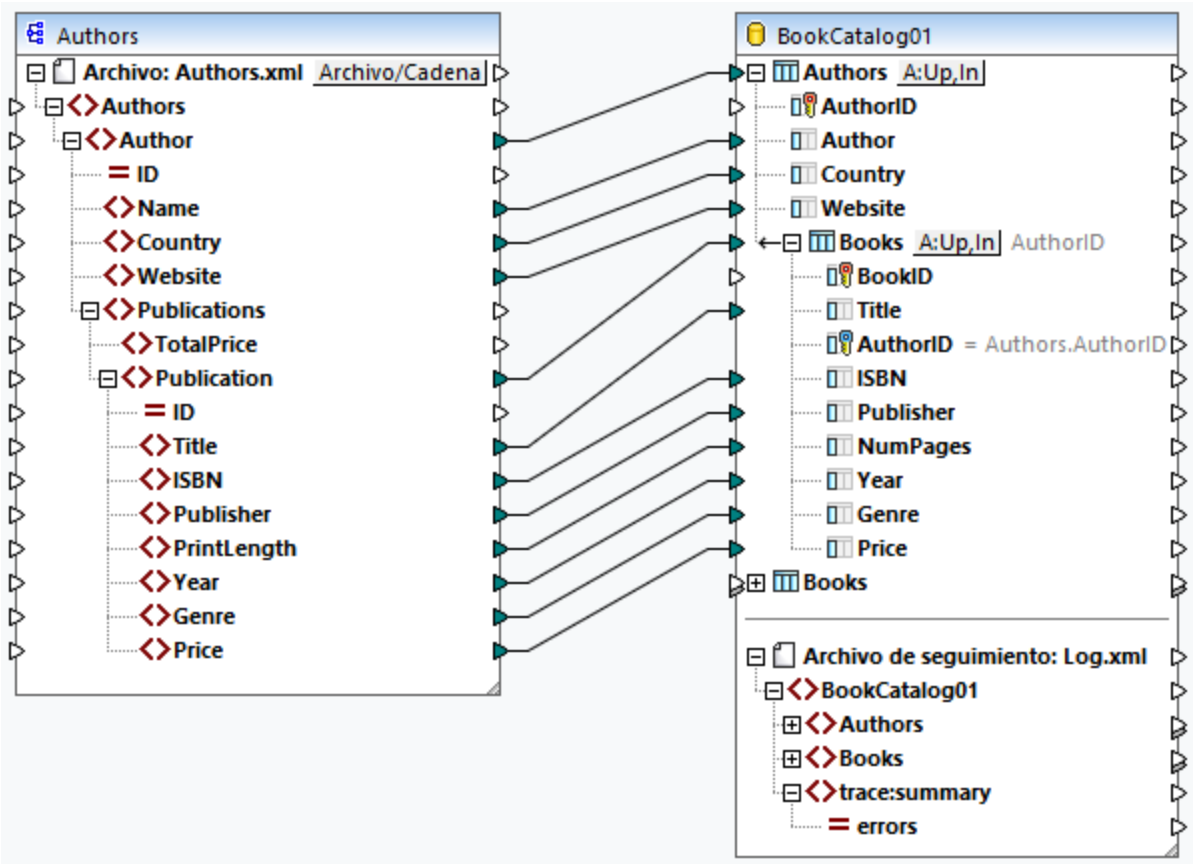
Al ejecutar el script SQL 274, esta instrucción SQL se ejecutará primero, es decir antes de cualquier instrucción para registros de BD.

*Otros casos posibles*
También puede configurar la acción *Eliminar si* en vez de la condición *Actualizar si*. En este caso se eliminarán los registros `Author` y `Book` que existen tanto en el archivo de origen como en la BD, y se insertarán nuevos registros en la BD (mediante la acción *Insertar restantes*). También puede decidir omitir estos mismos registros (con la condición *Omitir si*) e insertar nuevos registros en la BD (mediante la acción *Insertar restantes*).

## 4.2.3.3 Reversión de transacciones: Casos

Al asignar datos a bases de datos, es posible que encuentre errores relacionados con la BD (p.ej., puede que la cuenta de la base de datos no tenga los privilegios suficientes para ejecutar una acción específica de BD). Para evitar que estos errores cancelen la ejecución de una asignación de datos, puede configurar la reversión de transacciones que le permitirán revertir cualquier cambio. Puede habilitar las transacciones a nivel de la base de datos 254, a nivel de acciones de tabla 279, y a nivel de procedimientos almacenados 329.

En este apartado, se describen algunos escenarios que pueden ocurrir al revertir transacciones. Todos los casos en este apartado presentan un archivo de fuente XML llamado **`Authors.xml`** y una BD de destino llamada **`BookCatalog.sqlite`** (*ver la siguiente asignación*). Las tablas `Authors` y `Books` están vinculadas por una relación de clave foránea. La tabla `Author` es la tabla principal de la tabla `Books`.




En todos los casos, se han configurado las mismas acciones de tabla de BD 276:

- La combinación de las acciones de tablas *Actualizar si* e *Insertar restantes* en la tabla `Authors`. La condición *Actualizar si* se configura en la columna `Author`.
- La combinación de las acciones de tablas *Actualizar si* e *Insertar restantes* en la tabla `Books`. La condición *Actualizar si* se configura en la columna `Title`.

Además, se ha habilitado el seguimiento 280 a nivel de los componentes (configurado *Siempre*), en la tabla `Authors` (*Usar configuración de componentes*) y en la tabla `Books` (*Usar configuración de componentes*). Las configuraciones de la reversión de transacciones dependerán del caso.

 © 2019-2025 Altova GmbH

*Resumen general de casos*
A continuación enumeramos los casos descritos en este apartado:

- *Caso 1:* Si hay un registro `Author` erróneo, este registro principal y sus registros secundarios relacionados se revertirán. Si hay un registro `Book` defectuoso, solamente este registro secundario se revertirá.
- *Caso 2:* Si hay un registro `Author` erróneo, este registro principal y sus registros secundarios relacionados se revertirán.
- *Caso 3:* Si hay un registro `Book` defectuoso, solamente este registro secundario se revertirá.
- *Caso 4:* Si un registro `Book` no se puede insertar, se revertirá su registro principal (similar al *caso 2*).
- *Caso 5:* Si se produce un error relacionado con la BD, todos los cambios realizados en la BD se revertirán.

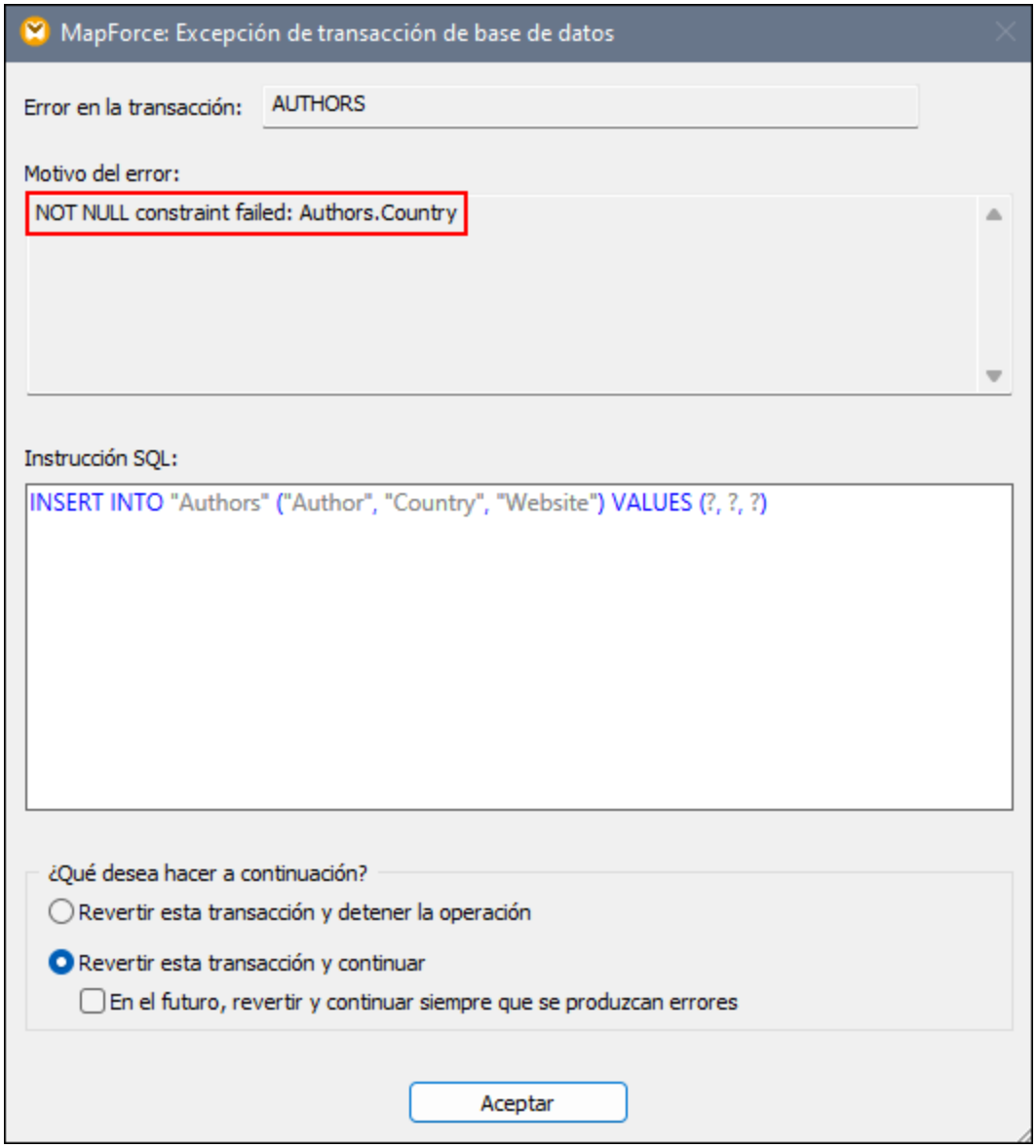
## Caso 1: Revertir Author actual y Book actual

En el primer caso, habilitamos la gestión de transacciones a nivel de acciones de tabla: La opción *Revertir transacción actual y continuar* está configurada en ambas tablas de la BD. La combinación de estas configuraciones significa que (i) si hay un registro `Author` erróneo, ni este registro ni sus registros secundarios `Book` se insertarán en la base de datos, y después se iniciará el procesamiento del siguiente registro; (ii) si hay un registro `Book` erróneo, pero su registro principal `Author` es válido, solamente el registro `Book` erróneo se revertirá, e iniciará el procesamiento del siguiente registro.

*Authors.xml*
El archivo **`Authors.xml`** contiene informaciones sobre autores/autoras y sus libros. En uno de los registros `Author` falta la información sobre el país del autor o de la autora. El archivo de origen también contiene un registro `Book` que carece de información sobre el editor.

*BookCatalog.sqlite*
Hemos configurado una restricción `NOT NULL` en la columna `Country` de la tabla `Authors` y en la columna `Publisher` de la tabla `Books`. Esto significa que si se asignan valores NULL a estas columnas, se genera un error.

*Resultados*
Después de ejecutar el script SQL 274 en el panel Resultados, aparece el siguiente cuadro de diálogo:

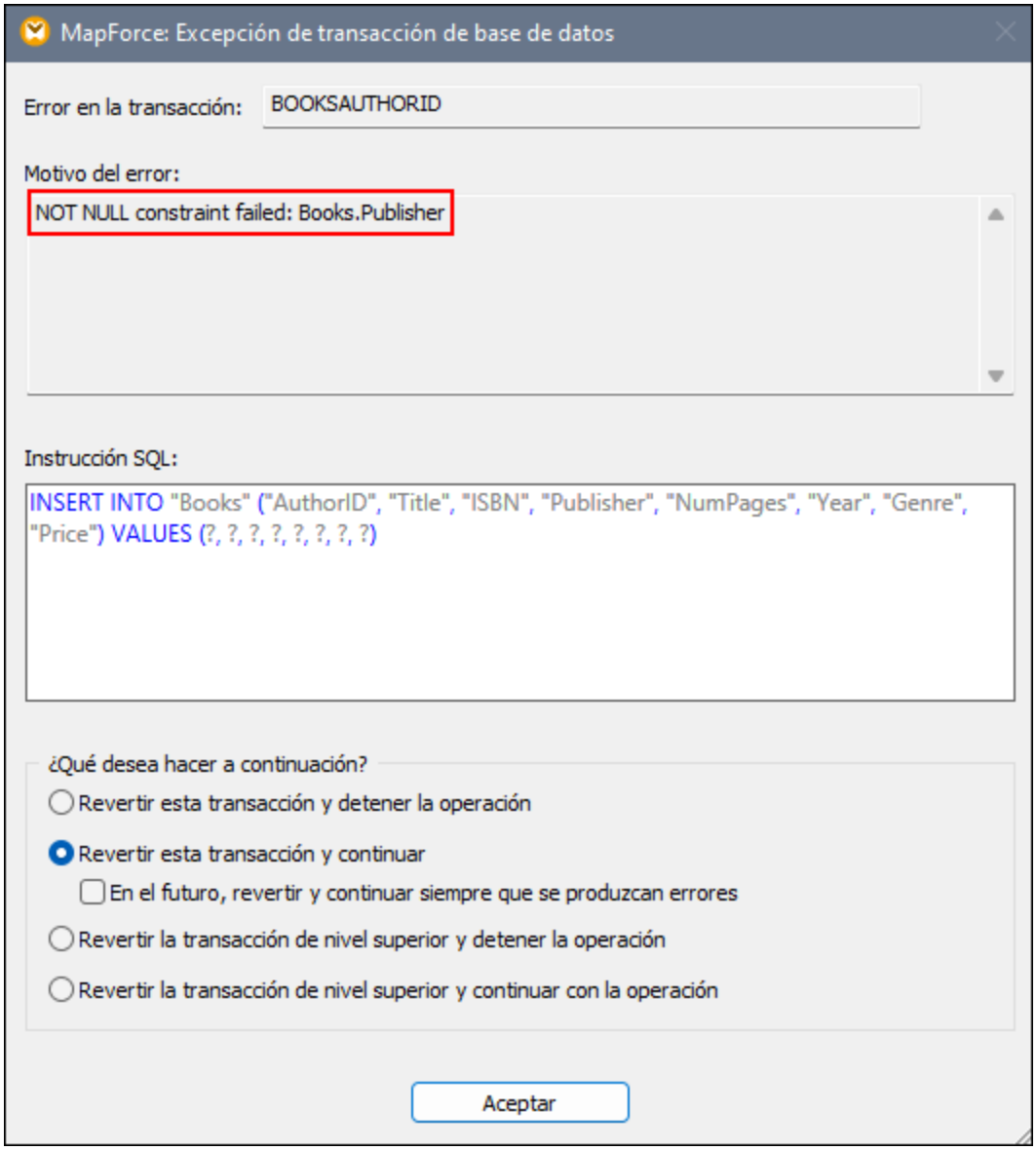



El cuadro de diálogo **Excepción de transacción de BD** nos informa sobre un error relacionado con la BD y el motivo de este error. En este ejemplo, no se ha podido insertar un valor NULL en la columna `Country` de la tabla `Authors` (*ver rectángulo anterior*). Este cuadro de diálogo también permite elegir la siguiente acción. Por defecto se selecciona la acción configurada en el cuadro de diálogo Acciones de tablas de BD 276. También se puede aplicar la misma configuración a todos los errores posteriores o revertir la transacción errónea y detener la operación.

Después de confirmar la selección, recibiremos un mensaje sobre otro error que ha ocurrido en la tabla `Books` (*ver imagen siguiente*). Este error se ha producido porque la asignación ha intentado insertar la información ausente de `Publisher` de uno de nuestros registros `Book` en el campo `Publisher` de la tabla `Books`. Dado que el error se ha producido en la tabla secundaria, es posible revertir la transacción actual (*Revertir esta transacción y detener/continuar*) o todos los cambios hasta el nivel superior (*Revertir transacción de nivel superior y detener/continuar*). En este contexto, la transacción de nivel superior es el registro principal (`Author`) del registro secundario defectuoso (`Book`).

 © 2019-2025 Altova GmbH

Si hay varios niveles en la jerarquía, se realizará una reversión para cada nivel hasta que se alcance el nivel superior en el que se haya habilitado la gestión de transacciones. En función de sus necesidades, la asignación se detendrá o pasará a ejecutar el siguiente componente. La opción *Revertir transacción de nivel superior y continuar/detener* aparece solamente si hay transacciones anidadas. Para más información sobre cómo revertir transacciones de nivel superior, véase el *caso 4* más abajo.




En este ejemplo, dejamos la opción de reversión de transacciones mostrada en la imagen anterior tal y como está. Una vez finalizado el procesamiento de todos los registros, podemos ver las instrucciones SQL generadas en el panel Resultados, el informe de seguimiento 255 en formato XML y los datos que están ahora disponibles en la BD.

*Instrucciones SQL generadas*

A continuación se puede ver un extracto del script SQL generado. La operación de inserción errónea se ha revertido al `SAVEPOINT` (punto de recuperación). El comando `SAVEPOINT` representa un punto de recuperación al

cual la transacción se retrocederá. El comando SAVEPOINT permite deshacer los cambios realizados después del SAVEPOINT y restaurar la transacción al estado del SAVEPOINT.

```
SAVEPOINT "BOOKSAUTHORID"
-- >>> OK. One or more rows.

UPDATE "Books" SET "ISBN" = '0099590328', "Publisher" = NULL, "NumPages" = 464, "Year" = 2016, "Genre" =
'Crime & Mystery', "Price" = 8.6 WHERE ("Books"."AuthorID" = 41) AND ("Books"."Title" = 'Cockroaches')
-- >>> OK. 0 row(s).

INSERT INTO "Books" ("AuthorID", "Title", "ISBN", "Publisher", "NumPages", "Year", "Genre", "Price")
VALUES (41, 'Cockroaches', '0099590328', NULL, 464, 2016, 'Crime & Mystery', 8.6)

>>> FAILED!
-->>> NOT NULL constraint failed: Books.Publisher


ROLLBACK TRANSACTION TO SAVEPOINT "BOOKSAUTHORID"
-- >>> OK. One or more rows.
```

*Informe de seguimiento*

A continuación se ilustra un extracto del informe de seguimiento. El informe de seguimiento muestra los campos junto con sus valores monitorizados (p.ej., `<Price>8.6</Price>`), da información sobre las acciones realizadas (p.ej., `insertar`) y el error ocurrido (ver el elemento `trace:error`).

```
<Books>
      <trace:values>
            <Title>Cockroaches</Title>
            <AuthorID>41</AuthorID>
            <ISBN>0099590328</ISBN>
            <Publisher xsi:nil="true"/>
            <NumPages>464</NumPages>
            <Year>2016</Year>
            <Genre>Crime &amp; Mystery</Genre>
            <Price>8.6</Price>
      </trace:values>
      <trace:actions>
            <trace:update rows-affected="0"/>
            <trace:insert>
                  <trace:error code="19" state="1299">La restricción NOT NULL ha
                  fallado: Books.Publisher</trace:error>
            </trace:insert>
      </trace:actions>
</Books>
```

*Datos actualizados en la BD*

En el panel de consulta de BD 294, puede consultar qué datos están disponibles en la BD actualmente. El extracto de la tabla `Books` que aparece a continuación muestra que se han insertado correctamente varios registros nuevos. Como la inserción de uno de los libros de Jo Nesbo ha dado error (*ver informe de seguimiento anterior*), sólo se han asignado dos de sus tres libros (`The Bat` y `The Redbreast`).

 © 2019-2025 Altova GmbH

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 52 | 52 | Night | 35 | 1473678161 | Mulholland Books UK | 384 | 2020 | Crime & Mystery | 13.11 |
| 53 | 53 | Don Quixote | 36 | 0142437239 | Penguin Classics | 1072 | 2003 | Classics | 13.99 |
| 54 | 54 | The Count of Monte Cristo | 37 | 0140449264 | Penguin Classics | 1276 | 2003 | Classics | 13.99 |
| 55 | 55 | The Odyssey | 38 | 0141192445 | Penguin Classics | 416 | 2010 | Classics | 18.49 |
| 56 | 56 | The Divine Comedy | 39 | 1435162064 | Barnes & Noble | 492 | 2001 | Classics | 25 |
| 57 | 57 | Miss Peregrine's Home for Peculiar Children | 40 | 9781594746031 | Random House UK Ltd | 384 | 2013 | Fantasy | 2.42 |
| 58 | 58 | The Bat | 41 | 9780099581871 | Random House UK Ltd | 432 | 2013 | Crime & Mystery | 8.93 |
| 59 | 59 | The Redbreast | 41 | 0099546779 | Vintage | 624 | 2009 | Crime & Mystery | 9.03 |

## Otros casos posibles

A continuación se describen otros casos posibles.

*Caso 2: Revertir Author actual y todos sus Books*

En este caso, la opción *Revertir transacción actual y continuar* se establece sólo en la tabla `Authors`. Con esta configuración, cada registro `Author` y los registros `Book` correspondientes se consideran una operación atómica. Si hay un registro `Author` defectuoso o si al menos uno de los libros del autor es erróneo, este registro principal y sus registros secundarios relacionados se revertirán. Después, el procesamiento seguirá con el siguiente registro `Author`.

*Caso 3: Revertir solamente el Book actual*

En este caso, la opción *Revertir transacción actual y continuar* se establece sólo en la tabla `Books`. Si hay un registro `Book` erróneo, solamente este registro `Book` se revertirá, pues se iniciará el procesamiento del siguiente registro. Si configura la opción *Revertir transacción de nivel superior y continuar* en la tabla `Books` y ninguna otra opción de gestión de transacciones en el nivel principal, también se revertirá sólo el registro `Book` defectuoso. En este caso no hay diferencia entre revertir la transacción superior y la actual, porque sólo hay un único nivel de transacción y no hay anidamiento de transacciones.

*Caso 4: Revertir Author si no se puede insertar Book*

En este caso, la transacción de la tabla `Authors` se ha configurado en *Revertir transacción actual y continuar*, mientras para la tabla `Books` se ha seleccionado *Revertir transacción de nivel superior y continuar*. Con esta configuración, si falla la inserción de un registro secundario, la transacción se revertirá en cada nivel de la jerarquía, hasta el nivel superior en el que se haya habilitado la gestión de transacciones. Por ejemplo, queremos insertar un nuevo registro `Author` (`Jo Nesbo`) y sus tres registros secundarios. En uno de los registros `Book` (`The Cockroaches`), falta la información sobre el editor. Dado que la tabla `Books` tiene una restricción NOT NULL en la columna `Publisher`, la asignación de un valor null a este campo provocará un error. En este caso, cuando se produce un error, el registro `Book` defectuoso (`The Cockroaches`) se revertirá junto con los otros dos registros `Book` y el registro principal. Después, el procesamiento continuará con el siguiente registro `Author`.

La combinación de las opciones de gestión de transacciones descritas en este caso es especialmente relevante cuando se trata de estructuras anidadas más complejas (p.ej., una tabla principal con dos o más tablas secundarias).

*Caso 5: Revertir todas las transacciones de BD y continuar*

En este caso, la opción *Revertir transacción de nivel superior y continuar* se establece a nivel de componente de BD 251. Esta opción puede ser útil si se trata de una asignación encadenada (p.ej., XML-BD-JSON). Si se produce un error relacionado con la BD, todos los cambios realizados en el componente de BD se revertirán y el procesamiento continuará con el componente JSON.

### 4.2.3.4 Instrucciones MERGE

Para determinadas bases de datos MapForce genera instrucciones de combinación MERGE (*ver imagen siguiente*) que se ejecutarán en la base de datos cuando se ejecute la asignación. La ventaja de las instrucciones MERGE es que reducen el número de llamadas al servidor de la base de datos porque estas instrucciones compaginan instrucciones INSERT y UDPATE en una sola instrucción. En el caso de las instrucciones MERGE la comprobación de coherencia la lleva a cabo la propia base de datos. Las instrucciones MERGE son compatibles si:

- La base de datos es una de las siguientes: SQL Server 2008 y superior, Oracle, DB2, Firebird;
- la base de datos de destino tiene la combinación de las acciones de tablas 278 *Insertar si* e *Insertar restantes*.

Si su base de datos admite instrucciones MERGE, el script SQL que genera MapForce sólo incluirá instrucciones UPDATE. No se puede obtener una vista previa para ninguna instrucción INSERT, ya que estas sólo se ejecutan si la condición *Actualizar si* no se cumple.

```
SET QUOTED_IDENTIFIER ON

UPDATE [dbo].[Users] SET [FirstName] = (CAST('Despine' AS nvarchar(50))), [LastName] = (CAST('Buttler' AS
nvarchar(50))) WHERE ([dbo].[Users].[UserID] = 1)

SELECT [UserID] FROM [dbo].[Users] WHERE ([UserID] = 1)
-->>> %UserID1%

DELETE FROM [dbo].[Addresses] WHERE EXISTS(SELECT * FROM [dbo].[Users] WHERE [dbo].[Users].[UserID] = [dbo].
[Addresses].[UserID] AND ([dbo].[Users].[UserID] = '%UserID1%'))

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] );
```

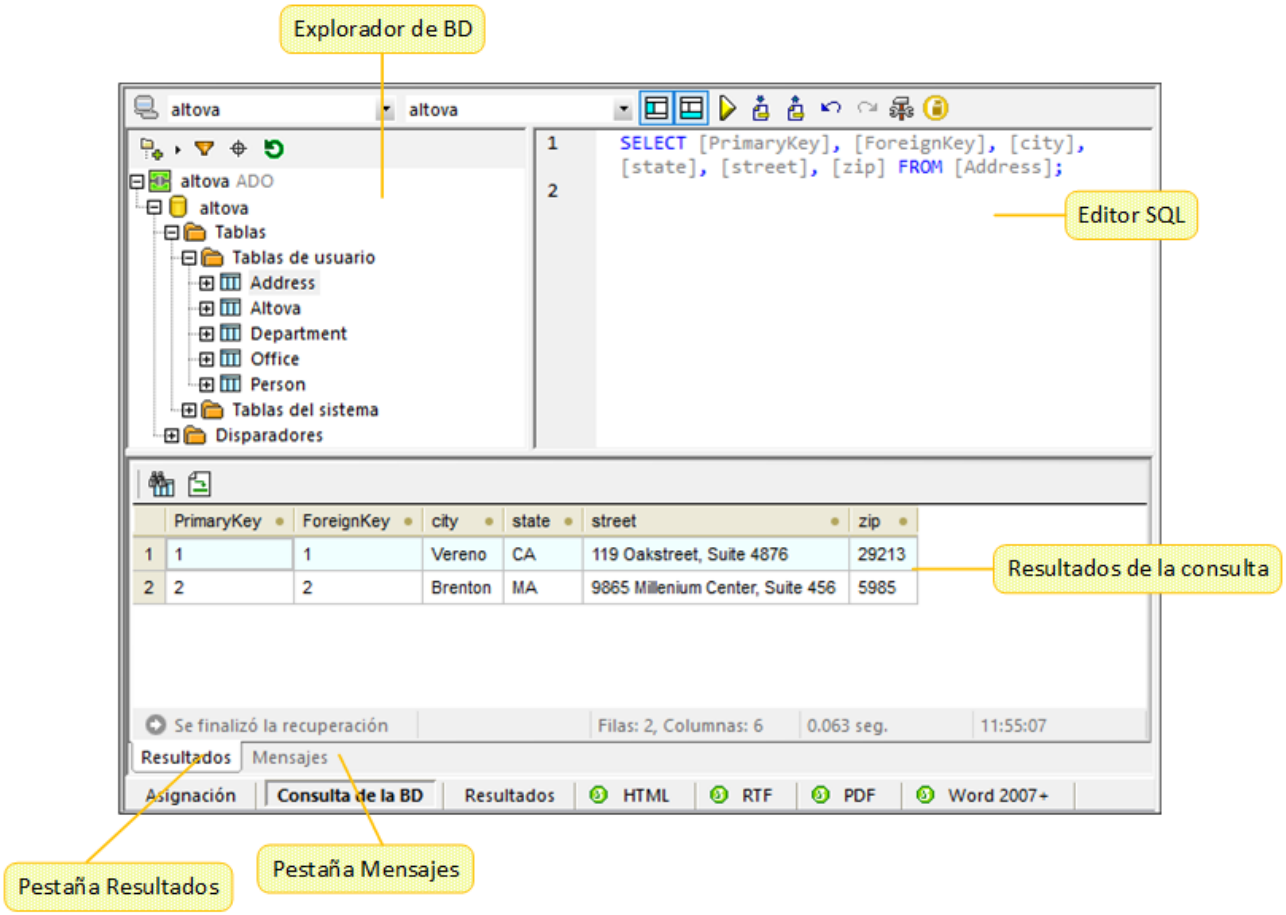
Si está actualizando varias tablas entre las que existen relaciones primaria-secundaria, entonces sólo se crean instrucciones MERGE para la tabla de menor nivel a la que se asignan datos. Por ejemplo, si la base de datos tiene una tabla principal llamada `Authors` y una tabla secundaria llamada `Books`, la instrucción MERGE sólo se generará para la tabla `Books`. En cambio, para la tabla `Authors` se generarán instrucciones UPDATE.

Con las instrucciones MERGE la opción Transferencia por lotes 281 sólo es compatible con conexiones de BD ODBC y JDBC.

## 4.2.4 Panel Consulta de BD

MapForce dispone de un panel de consulta de BD (*ver a continuación*) que permite consultar y modificar la base de datos independientemente del proceso de asignación. Estas consultas directas no se guardan con la asignación. Existe un panel de consulta de BD independiente para cada asignación activa en ese momento. En cada panel de consulta de BD puede conectarse a diferentes bases de datos. Recuerde que las

conexiones creadas en el panel de consulta de BD no forman parte de la asignación y no se conservan cuando se cierra la aplicación, a no ser que las defina como recursos globales 856.




El panel de consulta de BD se divide en varias vistas:

- *Explorador de la BD:* 296 muestra información sobre la conexión y las tablas de la BD;
- *Editor SQL:* 298 en él se escriben las consultas SQL;
- *Pestaña Resultados:* 301 muestra los resultados de las consultas en forma de tabla;
- *Pestaña Mensajes:* 303 muestra mensajes de advertencia y de error.

Para configurar las opciones de la consulta de BD, use la sección *Base de datos* en el cuadro de diálogo **Opciones**. Para más información, véase el apartado Configuración de Consulta de BD 1133.

*Conectarse a una base de datos*

Antes de ejecutar consultas en una base de datos 298, debe conectarse a ella. 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 BD y empezar a ejecutar consultas en su BD.

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 (ver Ejemplos 207). También puede seleccionar una conexión de BD existente de Recursos globales 856.

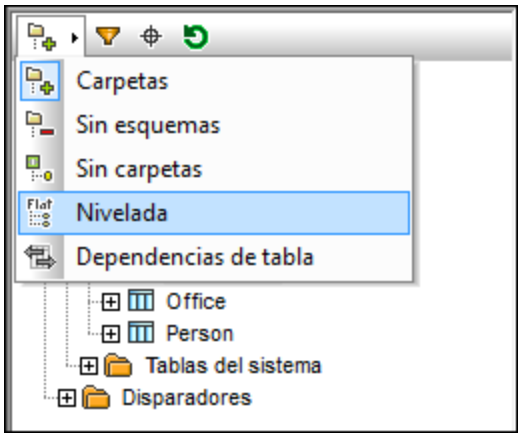
## 4.2.4.1 Explorador de BD

En este apartado explicaremos cómo puede personalizar la vista del árbol de BD, así como filtrar y buscar objetos de la base de datos. También describiremos las opciones del menú contextual que están disponibles para diferentes objetos de BD.

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.

### Diseños de árbol de BD

El explorador de la BD le permite personalizar la vista del árbol de BD. En la parte superior del panel Explorador de la BD puede elegir una disposición para todos estos objetos. (*ver imagen siguiente*).




Para seleccionar una disposición, haga clic en este botón (**Disposición en carpetas**) y elija la opción requerida de la lista. Observe que el aspecto del botón cambia dependiendo de la opción elegida. Estas son las opciones disponibles:

- La *Disposición en carpetas* organiza los objetos de la BD en carpetas en función del tipo de objeto (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.
- La *Disposición sin carpetas* muestra todos los objetos de la BD en una jerarquía pero sin usar carpetas.
- La *Disposición nivelada* muestra los objetos de la BD divididos por tipos (p.ej., 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 (p.ej., tablas con claves foráneas o tablas a las que se hace referencia).

Además de la navegación de la disposición, puede usar el Explorador de la BD para:

- Crear instrucciones SQL 299
- Filtrar y buscar objetos de la base de datos (*ver los apartados siguientes*).
- Dividir tablas en tablas del *sistema* y del *usuario* (*ver a continuación*).
- Actualizar el objeto raíz del origen de datos activo (el botón ).

 © 2019-2025 Altova GmbH

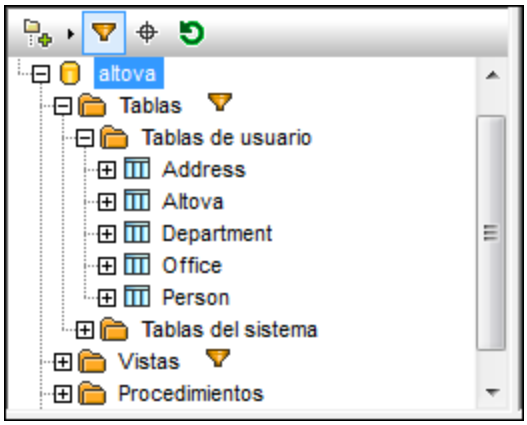
Para dividir tablas en tablas del *usuario* y tablas del *sistema*, en el Explorador de la BD haga clic con el botón derecho en la carpeta `Tablas` y seleccione **Ordenar por tablas de usuario y del sistema** en el menú contextual. Esta función sólo funciona si elige las siguientes disposiciones: *Disposición en carpetas*, *Disposición sin esquemas* o *Disposición nivelada*.

## Filtrar objetos de base de datos

Puede filtrar los objetos de base de datos (esquemas, tablas, vistas, etc.) por nombre o parte del nombre. Los objetos se filtran cuando introduce los caracteres. Por defecto el filtrado no distingue entre mayúsculas y minúsculas. La función de filtrado no funciona con la *disposición sin carpetas*.

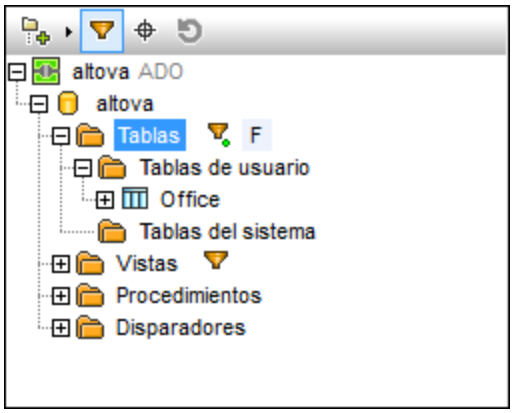
Para filtrar objetos de base de datos, siga estos pasos:

1. Haga clic en el botón en la parte superior del explorador de la BD. Ahora aparece un icono en forma de embudo junto a cada una de las carpetas de la jerarquía (*ver imagen siguiente*).




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., *G*). Los resultados aparecen a medida que se teclea. (*ver imagen siguiente*).



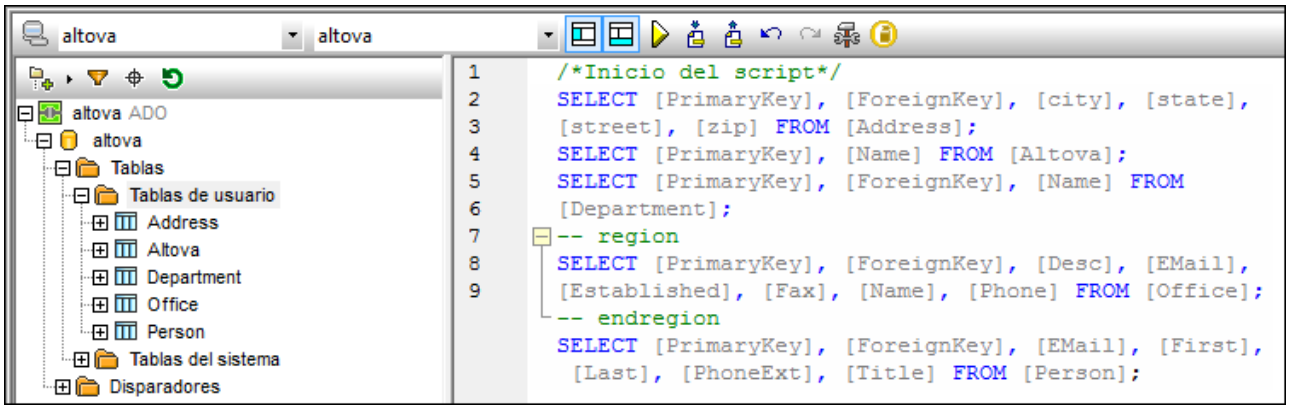

### Buscar objetos de base de datos

Para buscar por su nombre un elemento concreto de la base de datos puede usar funciones de filtrado (*ver más arriba*) o el **localizador de objetos**. Para buscar por su nombre un elemento concreto de la base de datos, siga estos pasos:

1. Haga clic en el el botón en la parte superior del explorador de la BD.
2. Introduzca el texto de búsqueda en la barra de texto.
3. Pulse **Entrar** o haga clic en un objeto de la lista de resultados para seleccionarlo.

## 4.2.4.2 Editor SQL

Tras conectarse a la base de datos podrá usar el editor SQL para escribir y ejecutar instrucciones SQL. El Editor SQL muestra todas las instrucciones SQL que posiblemente haya generado automáticamente, cargado desde scripts SQL existentes o escrito manualmente. El editor SQL incluye características como la finalización automática 251, las regiones y los comentarios en bloque o en línea.




### Botones de la barra de herramientas del Editor SQL

Estos son los botones de la barra de herramientas del Editor SQL:

| | |
|---|---|
| | El botón **Mostrar u ocultar Explorador** hace visible/invisible el panel Explorador. |
| | El botón **Mostrar u ocultar Resultados** hace visible/invisible el panel Resultados. |
| | El botón **Ejecutar consulta** ejecuta las instrucciones SQL que estén seleccionadas en el editor. Si existen varias instrucciones y no se selecciona ninguna, sólo se mostrará el resultado de la última instrucción en la pestaña Resultados. En caso de que haya múltiples resultados, MapForce sugerirá abrir estos resultados en Altova DatabaseSpy. |
| | El botón **Deshacer** permite deshacer un número ilimitado de operaciones de edición en el Editor SQL. |
| | El botón **Rehacer** permite rehacer comandos que se han deshecho previamente. |
| | El comando **Importar archivo SQL** abre un script SQL externo que se podrá ejecutar a continuación. |
| | El comando **Exportar archivo SQL** guarda un script SQL en la ubicación que desee. |
| | El comando **Abrir script SQL en DatabaseSpy** abre el script SQL actual en Altova DatabaseSpy (si lo tiene instalado). |
| | El botón **Opciones** abre el cuadro de diálogo Opciones 1133, donde se pueden definir opciones de configuración generales sobre la consulta de bases de datos y opciones de configuración del editor SQL. |

## Crear instrucciones SQL

Puede consultar su base de datos usando los siguientes métodos:

- Puede importar un script SQL o copiar instrucciones SQL y pegarlas en el Editor SQL (a través del comando **Importar archivo SQL** en la barra de herramientas).
- Puede escribir instrucciones SQL en el Editor SQL manualmente.
- También puede hacer clic con el botón derecho en un objeto del Explorador de la BD 296 y generar una consulta (normalmente, SELECT).

Para generar instrucciones SELECT SQL desde el Explorador de la BD, tiene las siguientes opciones:

- Hacer clic y mantener un objeto de la BD (p.ej., una tabla) o una carpeta en el Explorador de la BD y arrastrar ese objeto o esa carpeta en el Editor SQL.
- Hacer clic con el botón derecho en un objeto en el Explorador de la BD y seleccionar **Mostrar en Editor SQL | SELECT**.

## Ejecutar instrucciones SQL

Las instrucciones SQL en el Editor SQL se pueden ejecutar con efecto inmediato. Para ejecutar una instrucción SQL, haga clic en el botón . 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, en la pestaña Resultados solamente se mostrarán los resultados de la instrucción seleccionada o el resultado de la última instrucción (si no se ha seleccionado ninguna instrucción). En caso de que haya múltiples resultados, MapForce sugerirá abrir estos resultados en Altova DatabaseSpy.

## Importar/Exportar scripts SQL

Puede importar y exportar scripts SQL. Para importar un archivo SQL externo, haga clic en el botón de la barra de herramientas y seleccione el archivo SQL que desea abrir. Para exportar los contenidos del Editor SQL en un archivo SQL, haga clic en el botón de la barra de herramientas e introduzca el nombre del script SQL.

## Convertir el bloque (o la línea) SQL en comentarios/quitar comentarios

En el editor SQL de MapForce puede convertir instrucciones, parte de instrucciones o grupos de instrucciones en comentarios. Estas instrucciones se omiten cuando se ejecuta el script SQL. Puede insertar un comentario de bloque o de línea. El comentario de línea indica que la línea actual o la parte restante de la misma se ha convertido en un comentario. El comentario de bloque puede extenderse a varias lineas. El comentario de bloque también se puede usar para convertir palabras individuales en comentarios.

Para insertar un comentario de bloque o de línea, siga los siguientes pasos:

1. Seleccione una instrucción o parte de la instrucción.
2. Haga clic con el botón derecho en la instrucción seleccionada y elija la opción **Convertir el bloque (o la línea) en comentario/quitar comentario** en el menú contextual.

En la imagen siguiente puede ver un comentario de bloque (*texto verde*).

```
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 eliminar un comentario de bloque o de línea, siga los siguientes pasos:

1. Seleccione la parte de la instrucción que es un comentario.
2. Haga clic con el botón derecho en la parte seleccionada y elija la opción **Convertir el bloque (o la línea) en comentario/quitar comentario** en el menú contextual.

También puede eliminar manualmente los caracteres del comentario.

 © 2019-2025 Altova GmbH

### Agregar marcadores

Los marcadores se usan para marcar un elemento de interés en scripts. 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. El icono del marcador 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 quitar todos los marcadores, haga clic con el botón derecho en cualquier parte del Editor SQL y seleccione **Quitar todos los marcadores** en el menú contextual.

Para pasar al marcador siguiente/anterior, haga clic con el botón derecho en la línea correspondiente y seleccione **Ir al marcador siguiente/anterior**.

### Insertar regiones SQL

Las regiones son apartados del texto que se marcan y se declaran como unidad para estructurar sus scripts SQL. Las regiones pueden contraerse y expandirse para mostrar u ocultar partes de los scripts SQL. También es posible anidar regiones dentro de otras regiones. Al insertar una región, se abren un icono mostrar/ocultar y un comentario `--region` en la parte superior del texto seleccionado. Puede cambiar el nombre de una región añadiendo texto descriptivo al comentario `--region`. La palabra *region* no se debe eliminar en ningún caso.

Para agregar una región, siga los siguientes pasos:

1. Seleccione la instrucción que desea transformar en una región.
2. Haga clic con el botón derecho en las instrucciones seleccionadas y seleccione **Insertar región** en el menú contextual. A continuación se puede ver un ejemplo de una región.

```
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
```

Para quitar una región, elimine los comentarios `--region` y `--endregion`.

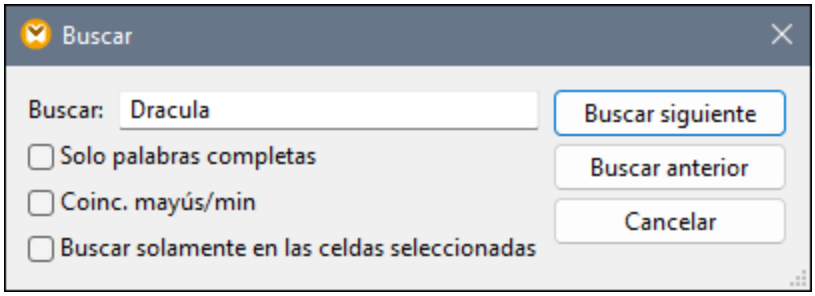
## 4.2.4.3 Pestaña Resultados

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 (*ver imagen siguiente*).

| | BookID | Title | AuthorID | ISBN | Publisher | NumPages | Year | Genre | Price |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | Misery | 1 | 1501143107 | Scribner | 368 | 2016 | Horror | 11.99 |
| 2 | 2 | Nightblind | 2 | 9781910633113 | Orenda Books | 231 | 2016 | Crime & Mystery | 9.99 |
| 3 | 3 | Blackout | 2 | 1910633461 | Orenda Books | 276 | 2016 | Crime & Mystery | 8.49 |
| 4 | 4 | Outsider | 1 | 1501180983 | Scribner | 576 | 2018 | Horror | 12.79 |
| 5 | 5 | Dracula | 3 | 9781435142817 | Barnes & Noble | 512 | 2013 | Classics | 13.69 |
| 6 | 6 | The Mystery of Edwin Drood | 4 | 9781400043286 | Everyman's Library | 336 | 2004 | Classics | 19.79 |
| 7 | 7 | Crime and Punishment | 5 | 0679420290 | Everyman's Library | 608 | 1993 | Classics | 14.99 |
| 8 | 8 | Wuthering Heights | 6 | 979-8469527794 | Independently published | 200 | 2021 | Classics | 22.99 |
| 9 | 9 | Dune (Dune Chronicles, Book 1) | 7 | 0441013597 | Penguin Publishing Group | 704 | 2005 | Sci-Fi | 14.99 |

*Botones de la barra de herramientas para navegar*

En la parte superior de la pestaña Resultados, hay dos botones de la barra de herramienta que sirven para navegar por los resultados de la consulta. El botón (**Buscar**) de la barra de herramientas le permite ejecutar una búsqueda en los resultados recuperados. Al hacer clic en este botón, se abre el cuadro de diálogo **Buscar** (*ver imagen siguiente*). Este cuadro de diálogo le permite configurar varios parámetros de búsqueda (p.ej., buscar palabras completas). Use los botones **Buscar siguiente** y **Buscar anterior** para cambiar entre las instancias del término de búsqueda.




El comando de la barra de herramientas (**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.

*Seleccionar datos*

Para seleccionar datos en los resultados de la consulta, puede usar los métodos siguientes:

- Puede hacer clic en el encabezado de columna para seleccionar una columna entera.
- Puede hacer clic en un número de fila para seleccionar una fila entera.
- Puede hacer clic y mantener la tecla **Mayús** para seleccionar un rango de celdas.
- Puede hacer clic con el botón derecho en una celda en cuestión y seleccionar varias opciones del menú contextual: **Selección | [Rango| Columna | Todo]**.
- También puede seleccionar todas las celdas haciendo clic dentro de la tabla y después en **Ctrl + A**.

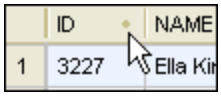
*Copiar datos*

Para copiar las celdas seleccionadas, haga clic en **Ctrl + C**. También puede hacer clic con el botón derecho en las celdas seleccionadas y pulsar **Copiar celdas seleccionadas** o **Copiar celas seleccionadas con encabezado** del menú contextual.

 © 2019-2025 Altova GmbH

*Ordenar datos*
Para ordenar datos, puede elegir entre las siguientes opciones:

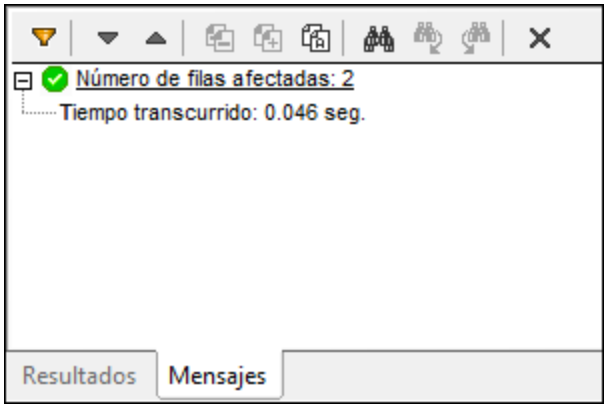
- Hacer clic con el botón derecho en cualquier parte de la columna en cuestión y pulsar **Ordenación | [Ascendente | Descendente]** en el menú contextual.
- Hacer clic en el icono de punto (*ver imagen siguiente*) en el encabezado de la columna hasta que haya seleccionado la opción relevante: la flecha hacia arriba ordena los datos en orden ascendente, la flecha hacia abajo ordena los datos en orden descendente.




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.

## 4.2.4.4 Pestaña Mensajes

La pestaña Mensajes del panel de consulta de BD muestra información sobre la última instrucción (o instrucciones) SQL que se ejecutó y puede mostrar errores y advertencias.




*Botones de la barra de herramientas*
Puede usar diferentes filtros para personalizar la vista de la pestaña Mensajes. Los botones en la parte superior de la pestaña Mensajes (*imagen anterior*) sirven para navegar por los mensajes, copiar texto en el portapapeles y ocultar ciertos partes del mensaje. También podrá ver estas opciones en el menú contextual haciendo clic con el botón derecho dentro de la pestaña Mensajes.

| | |
|---|---|
| | El botón **Filtro** abre un menú que permite seleccionar los tipos de mensajes que quiere ver en la pestaña Mensajes. Estas son las posibles opciones: *Seleccionar todo*, *Deseleccionar todo*, *Resumen*, *Operación correcta*, *Advertencia*, *Error*, *Inserción automática* y *Progreso*.<br><br>La opción *Inserción automática* se refiere a aquellos mensajes provocados al insertar instrucciones SQL o construcciones SQL automáticamente en el editor SQL. La opción *Progreso* informa sobre el resultado de la conexión con la base de datos y el resultado de analizar datos SQL y cargar la estructura de datos. |

| | |
|---|---|
| | El botón **Siguiente** pasa al siguiente mensaje y lo resalta. |
| | El botón **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** |
| | El comando **Buscar** abre el cuadro de diálogo **Buscar** que permite configurar criterios de búsqueda. |
| | El comando **Buscar anterior** pasa a la instancia anterior de la cadena especificada en el cuadro de diálogo **Buscar**. |
| | El comando **Buscar siguiente** pasa a la instancia siguiente de la cadena especificada en el cuadro de diálogo **Buscar**. |
| | El botón **Borrar** elimina todos los mensajes de la pestaña Mensajes del Editor SQL. |

## 4.2.5 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.

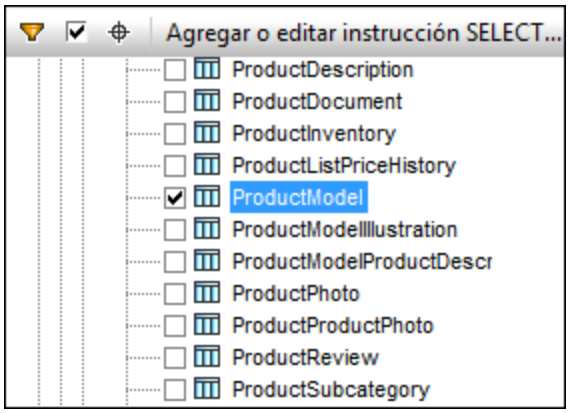
 © 2019-2025 Altova GmbH

## 4.2.5.1 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.

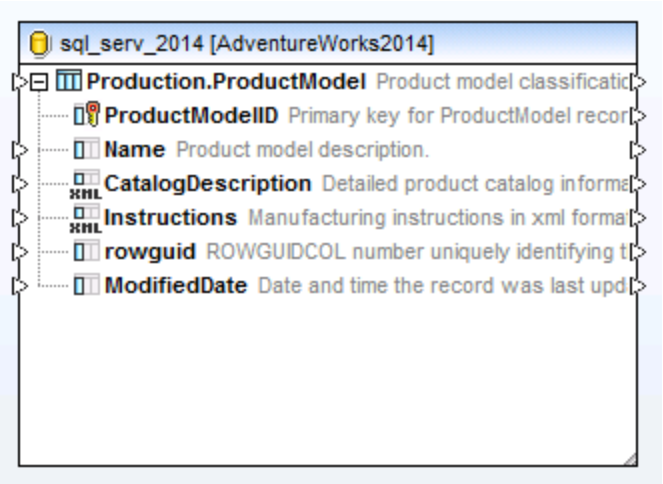
**Para agregar la base de datos Adventure Works 2014 como componente de asignación:**

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)[185] y Conectarse a Microsoft SQL Server (ODBC)[201]. **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*[188]).
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**.

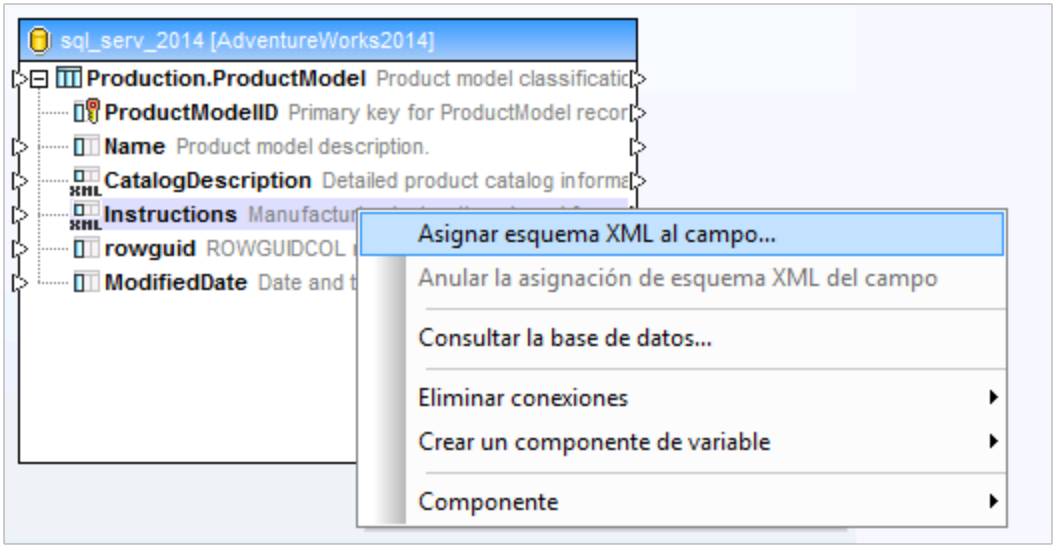



3. Para terminar haga clic en **Aceptar**.

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**:

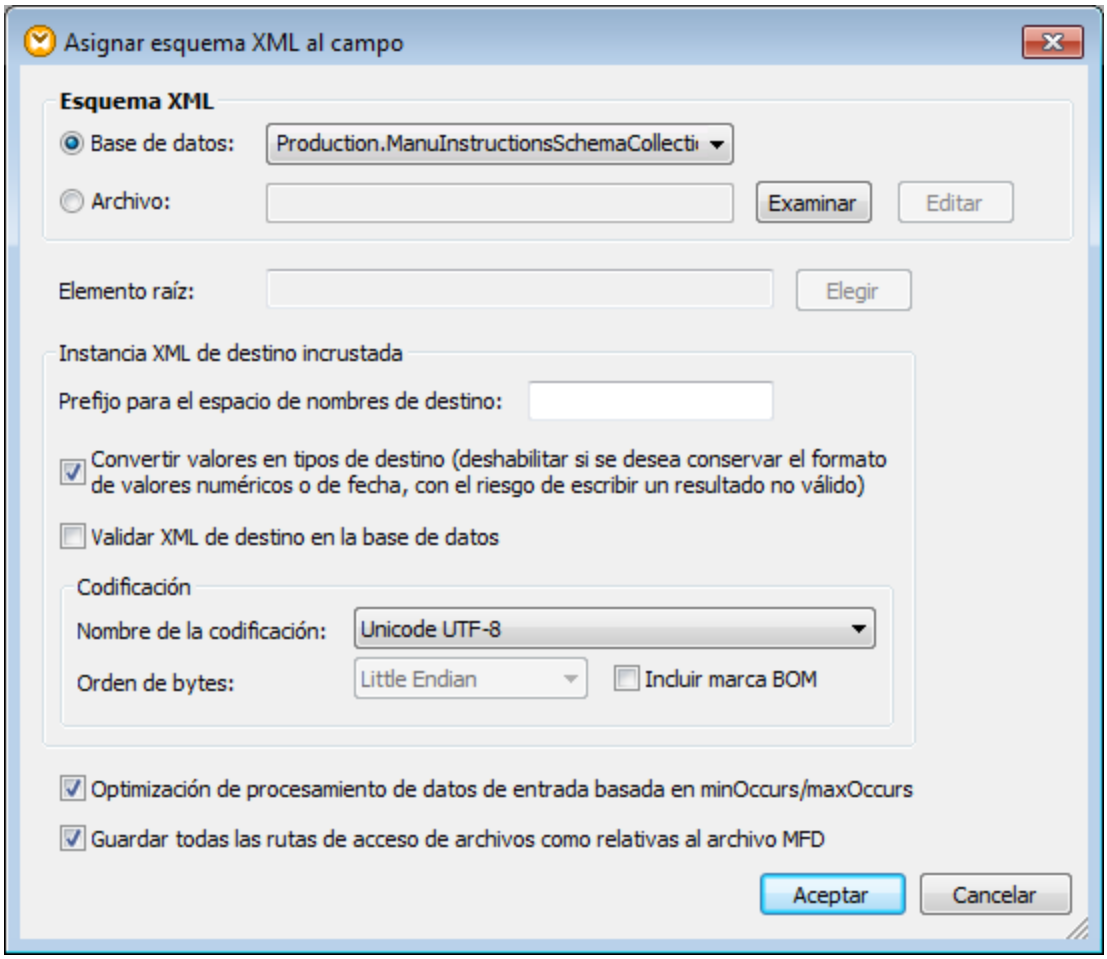



Para que la estructura de los campos XML aparezcan en la asignación es necesario el esquema XML del contenido del campo. Haga clic con el botón derecho en el campo *Instructions* y seleccione **Asignar esquema XML al campo** en el menú contextual.

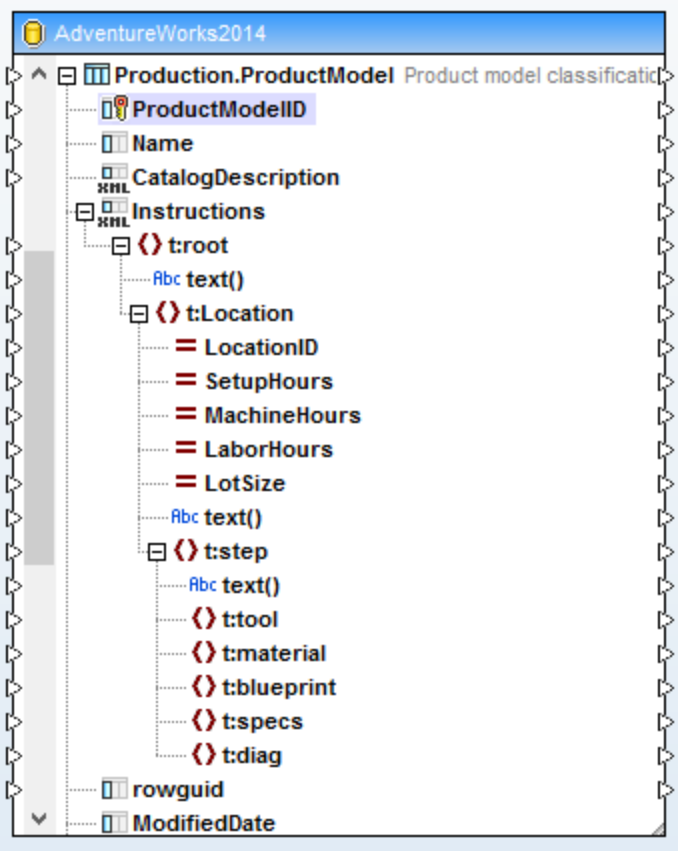



En este ejemplo concreto asignaremos un esquema al campo **Instructions** desde la BD directamente. Para ello seleccionamos el elemento **Production.ManuInstructionsSchemaCollection** en el cuadro combinado de la opción *Base de datos:* y hacemos clic en **Aceptar**.

 © 2019-2025 Altova GmbH




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.




## 4.2.5.2 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\MapForce2025\MapForceExamples\Tutorial\`:

| | |
|---|---|
| El archivo de diseño de asignación | • **XmlToSqliteField.mfd** |
| Los archivos XML de origen | • **bookentry1.xml**<br>• **bookentry2.xml**<br>• **bookentry3.xml** |
| El esquema XML utilizado para la validación | • **books.xsd** |
| La base de datos SQLite de destino | • **Library.sqlite** |

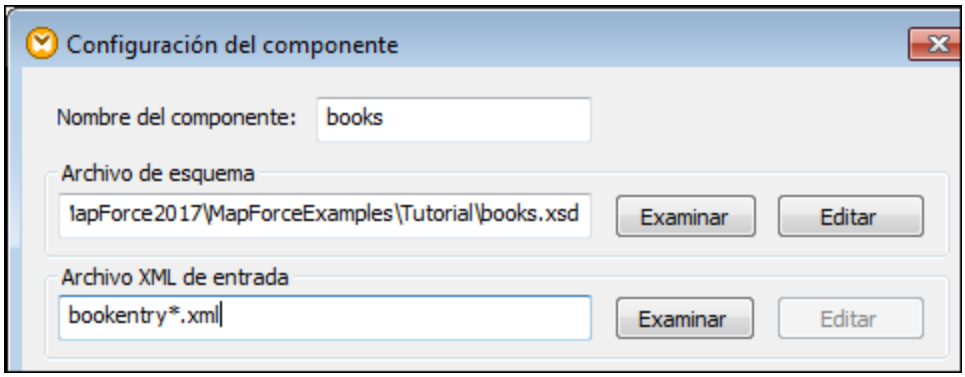
 © 2019-2025 Altova GmbH

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.

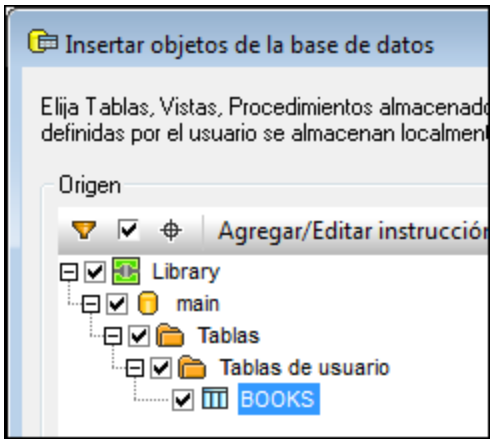
## Paso nº1: agregar el componente XML

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\MapForce2025\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 784.

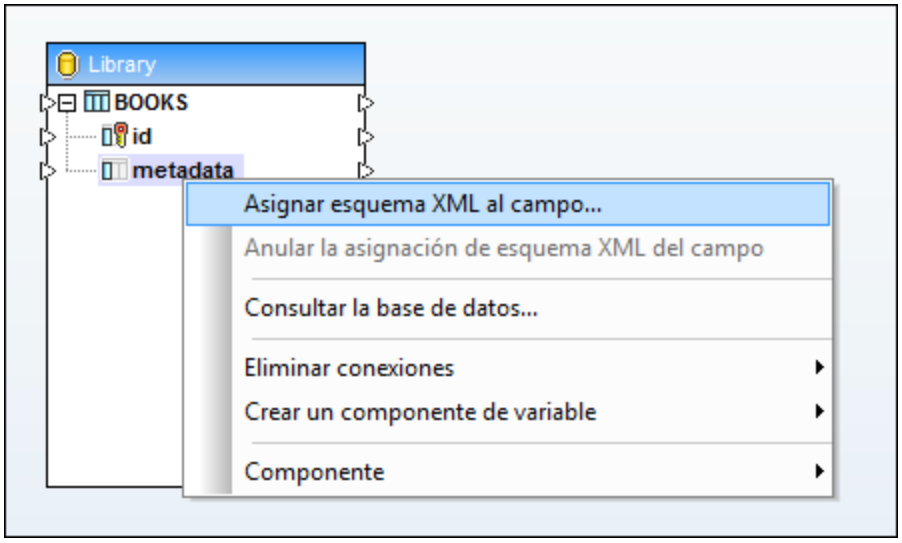



## Paso nº2: agregar el componente SQLite

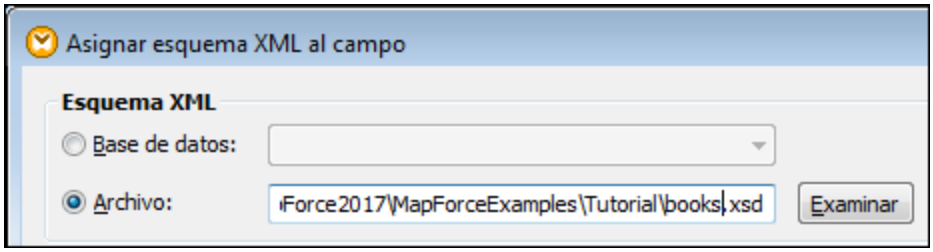
En el menú **Insertar** haga clic en **Base de datos** y siga los pasos del Asistente para conectarse al archivo de base de datos **Library.sqlite** del directorio `<Documentos>\Altova\MapForce2025\MapForceExamples\Tutorial\` directory (*véase Conectarse a una base de datos SQLite* 204). Llegada la hora de seleccionar objetos de la BD, seleccione la tabla `BOOKS`.



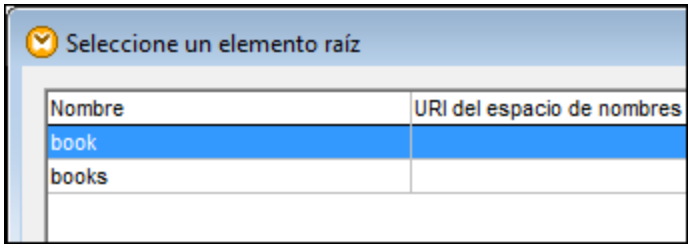
El campo de BD donde se escribirá el contenido XML se llama `metadata`. Para asignar un esquema XML a este campo haga clic con el botón derecho y seleccione **Asignar esquema XML al campo** en el menú contextual.




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\MapForce2025\MapForceExamples\Tutorial\`:



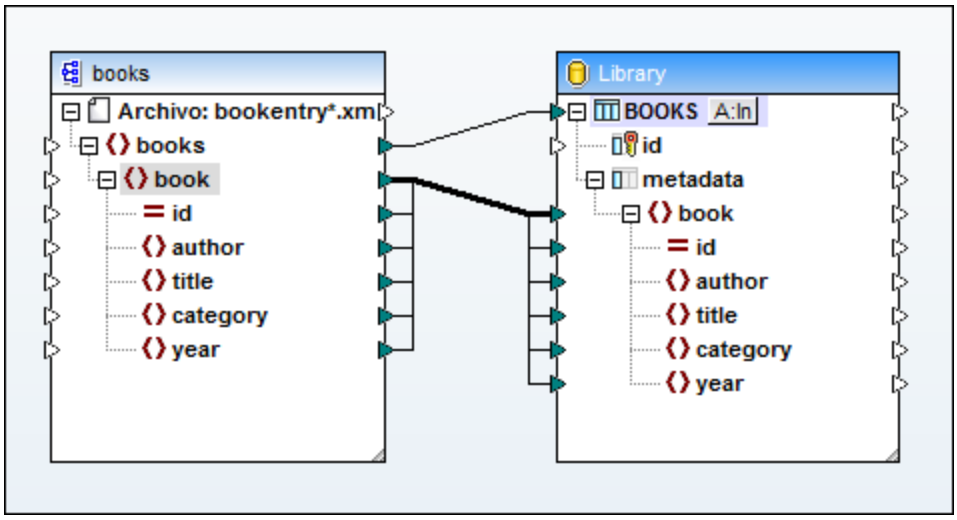

El esquema **books.xsd** tiene dos elementos con declaración global: `book` y `books`. En este ejemplo estableceremos `book` como elemento raíz de los datos XML escritos en el campo de la BD. Haga clic en **Elegir** y seleccione `book` como elemento raíz:




## Paso nº3: crear las conexiones de asignación y configurar la acción INSERT

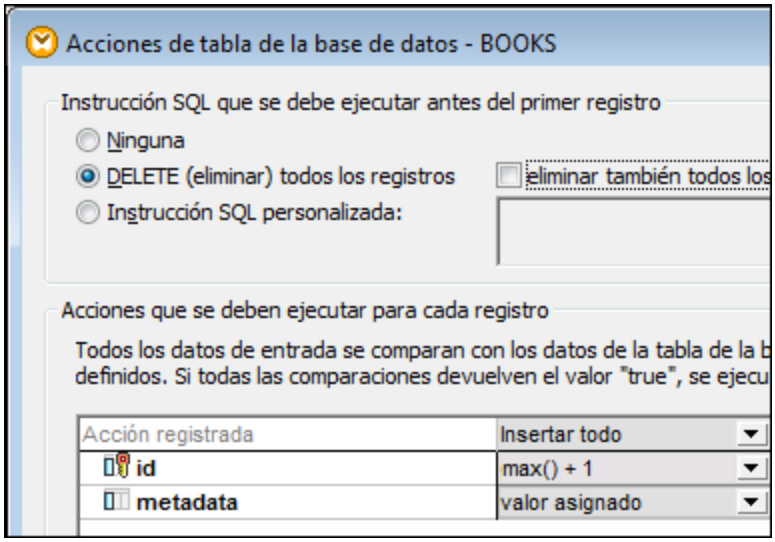
Cree estas conexiones de asignación:


© 2019-2025 Altova GmbH




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 57.

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.

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).

### 4.2.5.3 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>');
```

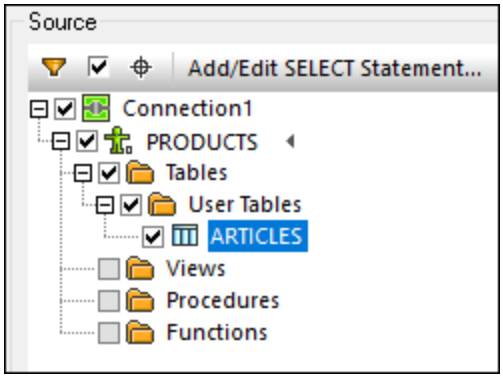
 © 2019-2025 Altova GmbH

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`.

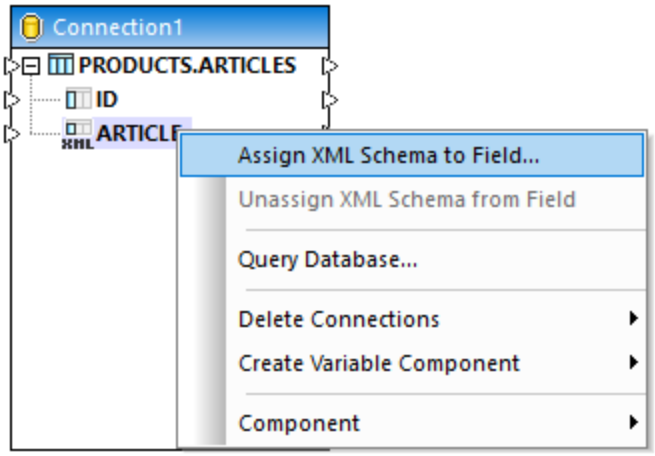
## Paso nº1: agregar la base de datos

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

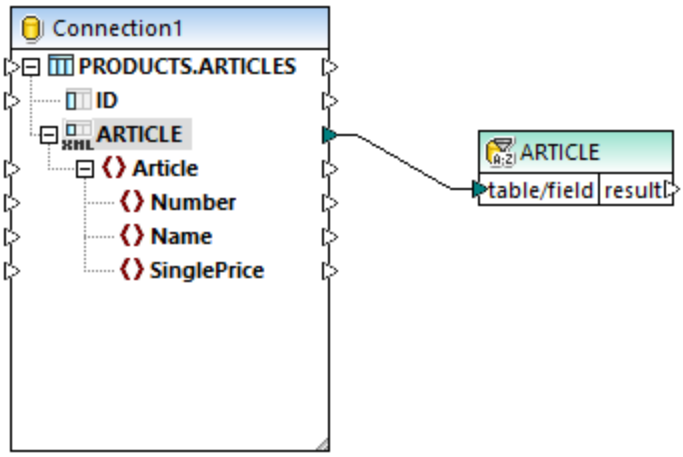
1. Haga clic con el botón derecho en el elemento `ARTICLE` del componente y seleccione **Asignar esquema XML al campo** en el menú contextual.




2. Seleccione *Archivo* y navegue hasta este esquema: **<Documentos>\Altova\MapForce2025\MapForceExamples\Tutorial\DB2xsd.xsd**.

## Paso nº3: agregar el componente WHERE/ORDER de SQL

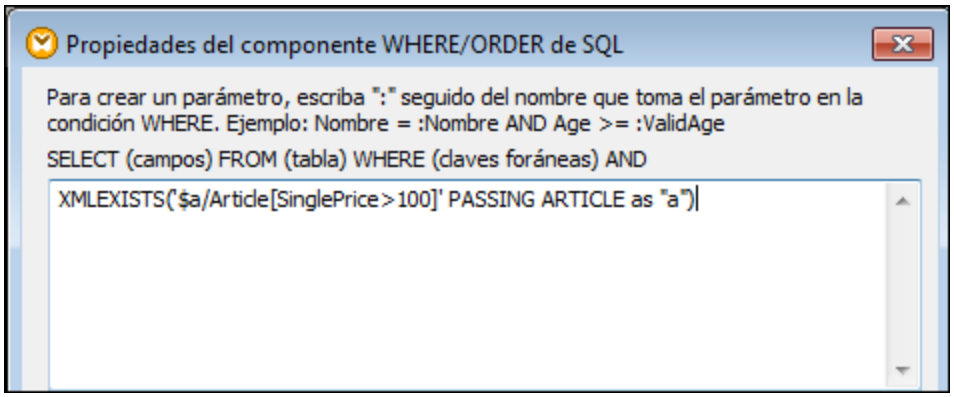
1. En el menú **Insertar** seleccione el comando **WHERE/ORDER de SQL**.
2. Conecte la columna de tipo XML `ARTICLE` a la entrada del componente WHERE/ORDER de SQL.




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")
```

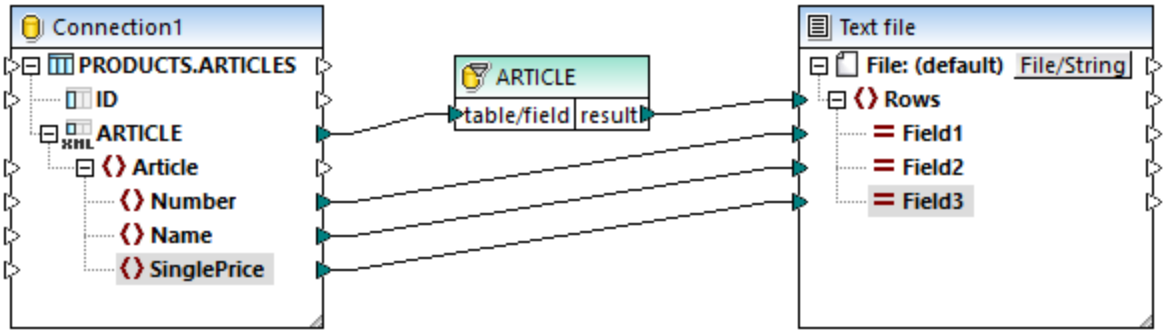
 © 2019-2025 Altova GmbH




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 sólo aparecen los productos con un precio superior a 100.

```
1    2,Socks,230
2    4,Jacket,5750
```

## 4.2.6 Procedimientos almacenados

Los procedimientos almacenados son programas que se alojan y ejecutan en un servidor de bases de datos. Los procedimientos almacenados pueden ser llamados por aplicaciones cliente y suelen estar escritos 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 origen de datos 321.
- Llamar a un procedimiento almacenado 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 entrada y salida 325.
- Llamar a un procedimiento almacenado 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, en este enfoque puede ejecutar el procedimiento almacenado dentro de una transacción de base de datos que puede revertirse en caso de error. Para ver un ejemplo, consulte Procedimientos almacenados en componentes de destino 329.

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 Procedimientos almacenados y relaciones locales 333.

**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. Puede descargarse de https://github.com/Microsoft/sql-server-samples/releases/tag/adventureworks.

## Notas sobre compatibilidad

- Los procedimientos almacenados solo pueden utilizarse en el motor de ejecución BUILT-IN. No se admite 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.

 © 2019-2025 Altova GmbH

- 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 329). 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).

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<br>• Los valores devueltos de los procedimientos almacenados de DB2 no son compatibles porque no pueden leerse a través de las API de base de datos utilizadas en MapForce.<br>• Las funciones con valores de fila (RETURNS ROW) no son compatibles.<br>• 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 de ADO que provoca que falte una fila del conjunto de resultados. |
| 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 y funciones escalares |
| MySQL | • Compatibles con MapForce: procedimientos almacenados y funciones escalares<br>• 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 Esto incluye los procedimientos almacenados y las |

| Base de datos | Notas sobre compatibilidad |
|---|---|
| | funciones independientes, así como los definidos dentro de un paquete Oracle.<br>• Se recomienda usar un controlador Oracle nativo en lugar del proveedor **Microsoft OLE DB Provider for Oracle**.<br>• 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 y funciones con valores de tabla<br>• En PostgreSQL, cualquier parámetro de salida definido en una función describe 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<br>• Para obtener información sobre los proveedores ADO disponibles, consulte Conexión ADO 185.<br>• 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.<br>• 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<br>• Las funciones escalares, las funciones agregadas y las funciones de tabla no son compatibles<br>• Problema conocido: el controlador ODBC de Teradata se niega a rellenar los valores de los parámetros de salida después de una llamada a un procedimiento. |


© 2019-2025 Altova GmbH

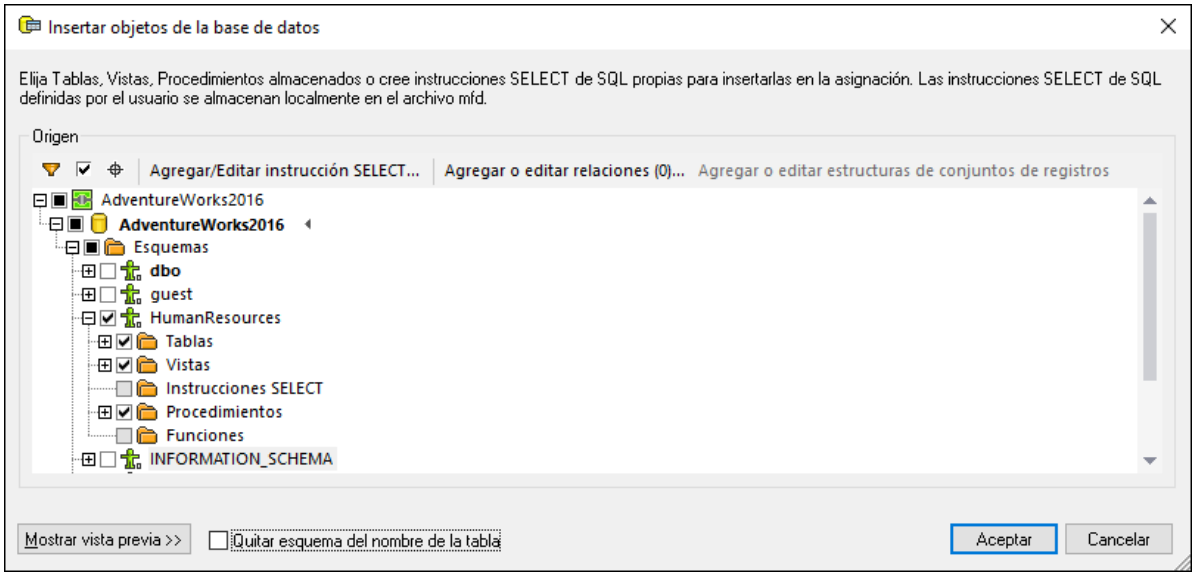
## 4.2.6.1 Agregar procedimientos almacenados a la asignación

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.

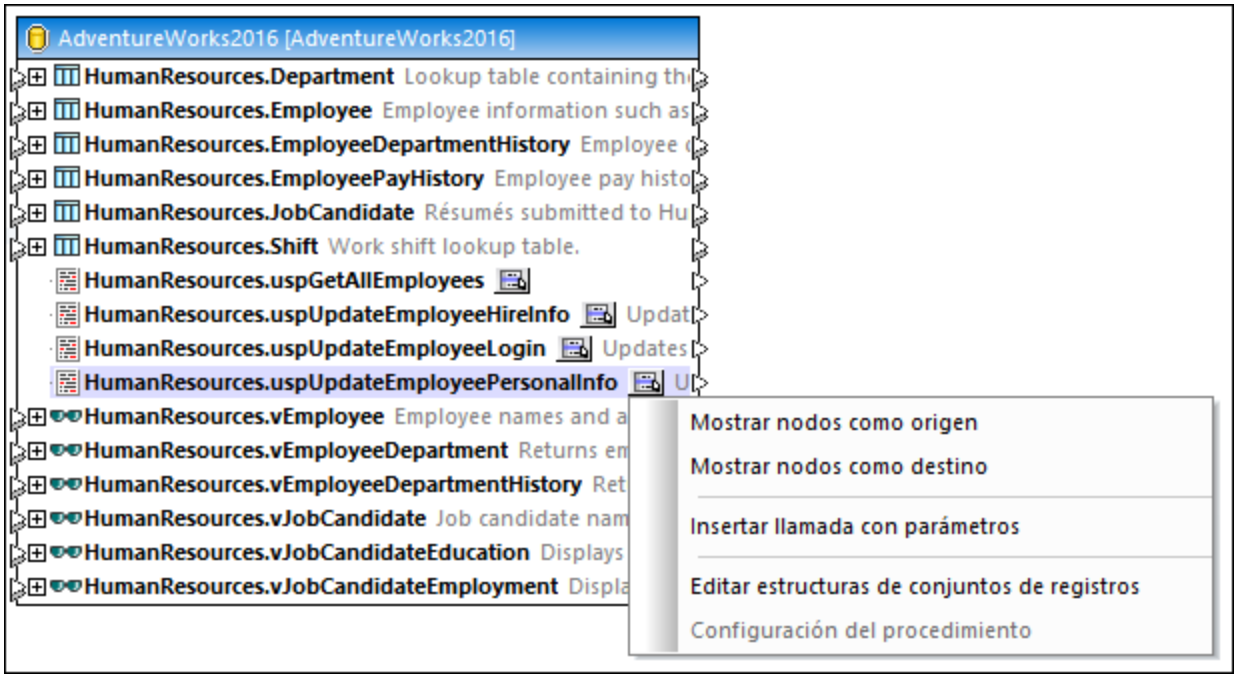



3. Marque las casillas de los objetos de BD que necesita que aparezcan en la asignación y haga clic en **Aceptar**. En este ejemplo hemos seleccionado todas las tablas, vistas y procedimientos almacenados disponibles en el esquema "HumanResources".

**Notas**

- 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 321. |
| **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 329. |

 © 2019-2025 Altova GmbH

| | |
|---|---|
| **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 325. |
| **Editar estructuras de conjuntos de registros** | Sólo 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** | Sólo 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. |

## 4.2.6.2 Procedimientos almacenados como fuente de datos

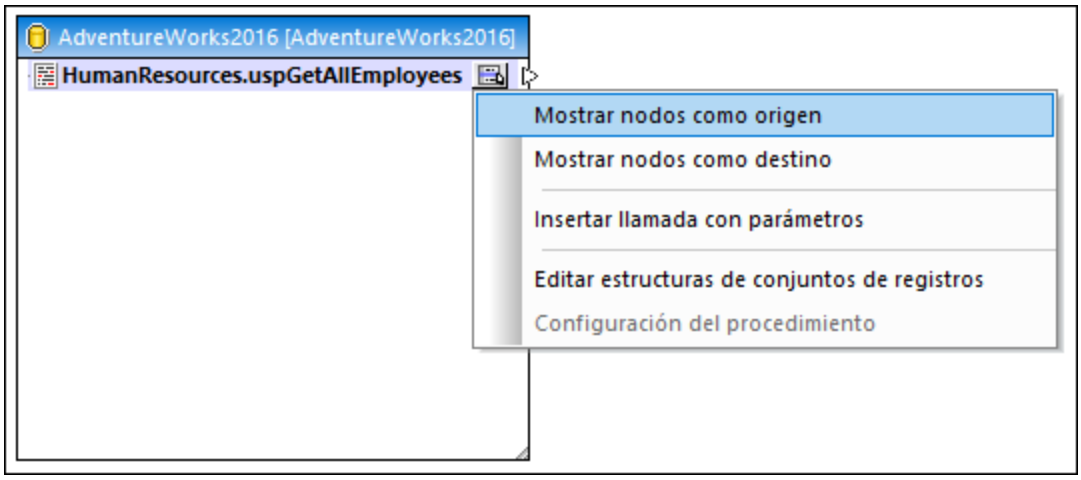
En este ejemplo explicamos cómo llamar a un procedimiento almacenado que no toma parámetros de entrada y únicamente obtiene datos de la base de datos. En este caso el procedimiento almacenado actúa como componente de origen de la asignación y los datos que obtiene se pueden asignar a otro componente 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 325.

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* 294). 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
```

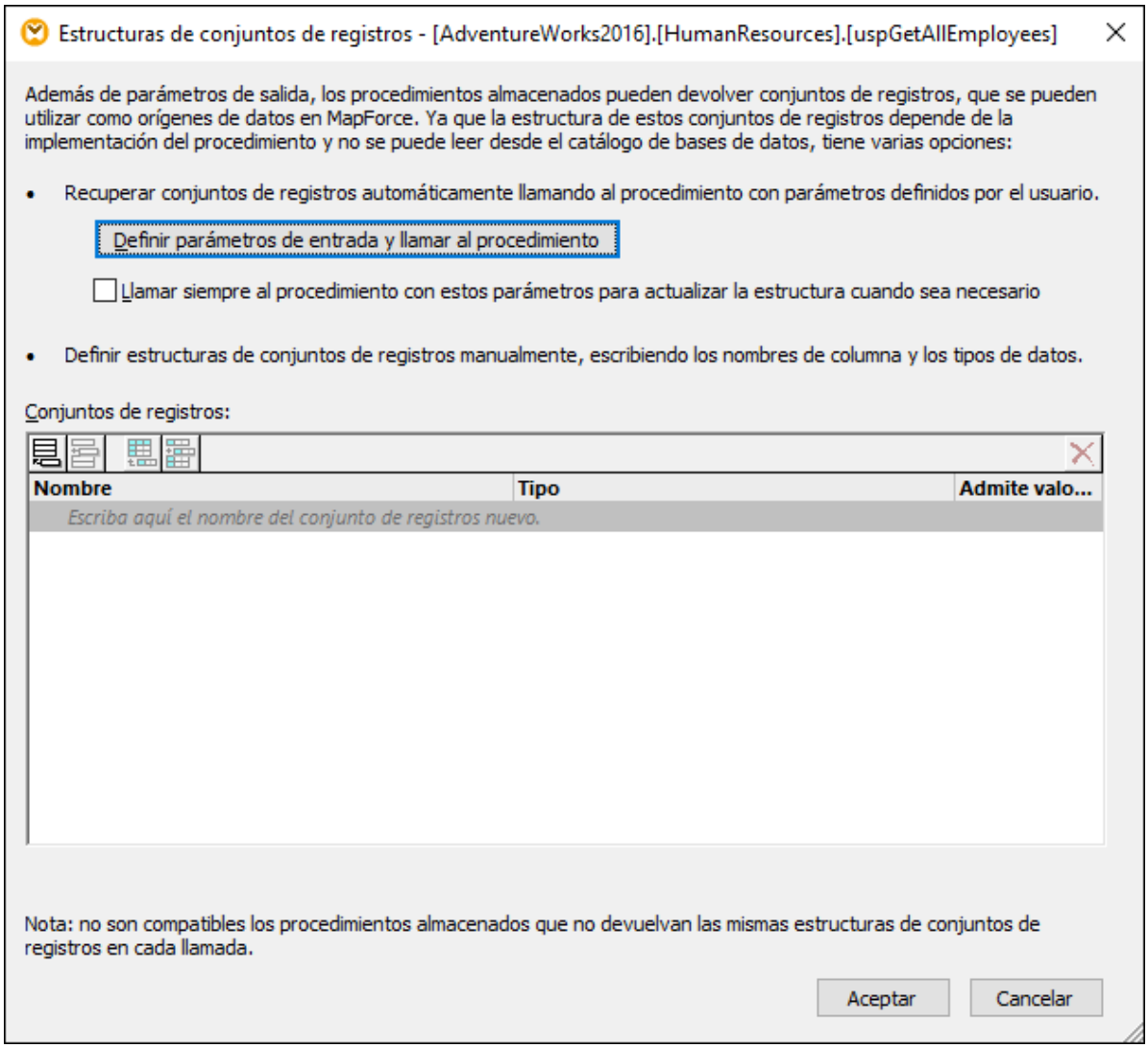
El procedimiento almacenado anterior devuelve información sobre los empleados de la vista **vEmployeeDepartment**. Los siguientes pasos muestran cómo crear una asignación que consume datos devueltos por este procedimiento.

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 319. 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**.



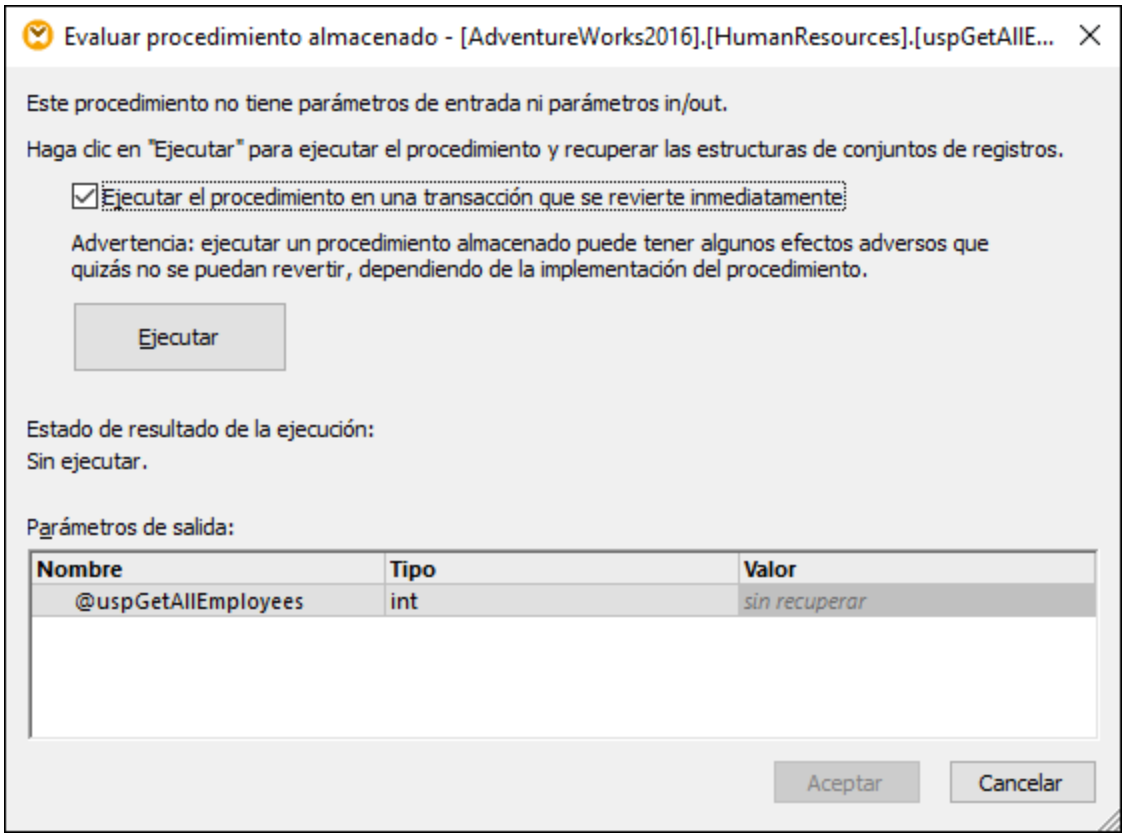

3. Vuelva a hacer clic en el botón para mostrar el menú contextual ( ) y seleccione **Editar estructuras de conjuntos de registros**. Aparecerá el cuadro de diálogo "Estructuras de conjuntos de registros".


© 2019-2025 Altova GmbH

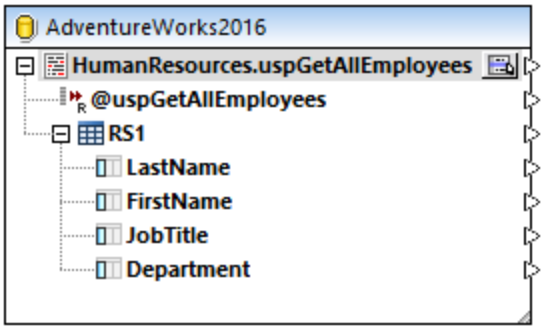



Llamar a un procedimiento almacenado en el momento de diseñar la asignación puede tener efectos secundarios (dependiendo de la implementación del procedimiento). Si no quiere ejecutar el procedimiento almacenado en el momento de diseñar la asignación, entonces no haga clic en **Ejecutar**, como se describe 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".

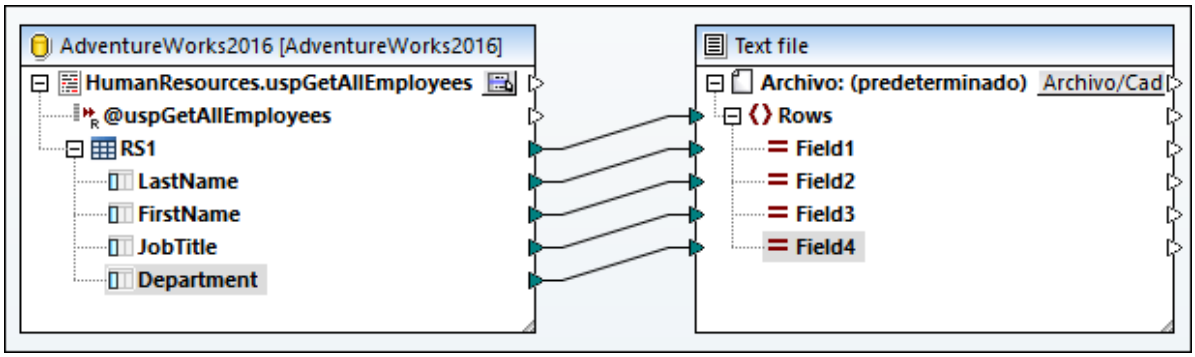
4. Haga clic en **Definir parámetros de entrada y llamar al procedimiento** y después haga clic en **Aceptar**. Aparecerá el cuadro de diálogo "Evaluar procedimiento almacenado".



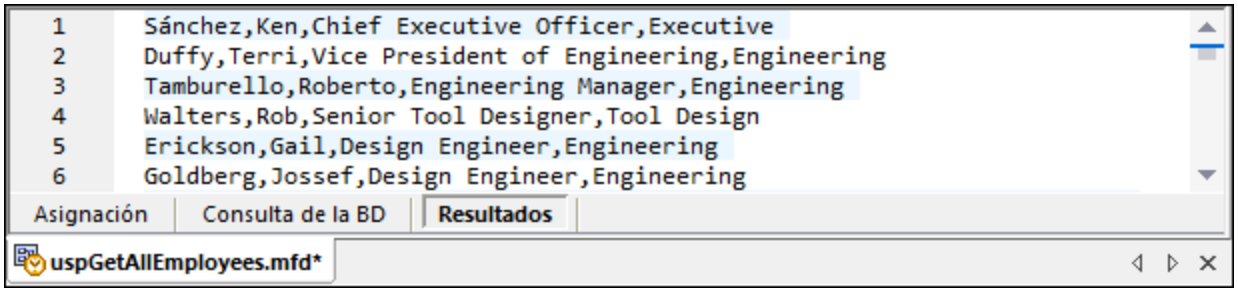

5. Haga clic en **Ejecutar** y después en **Aceptar**. La estructura del conjunto de datos ("RS1") ahora es visible tanto en cuadro de diálogo "Estructuras de conjuntos de registros" como en la asignación.




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 344.




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:




## 4.2.6.3 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* 294). 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

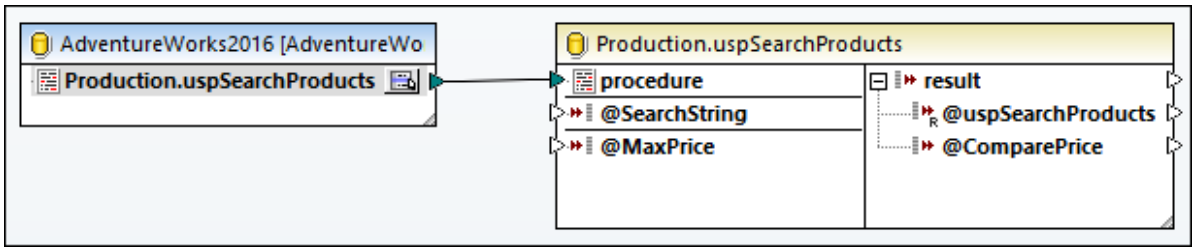
   SET @ComparePrice = @MaxPrice
```

```
    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`).

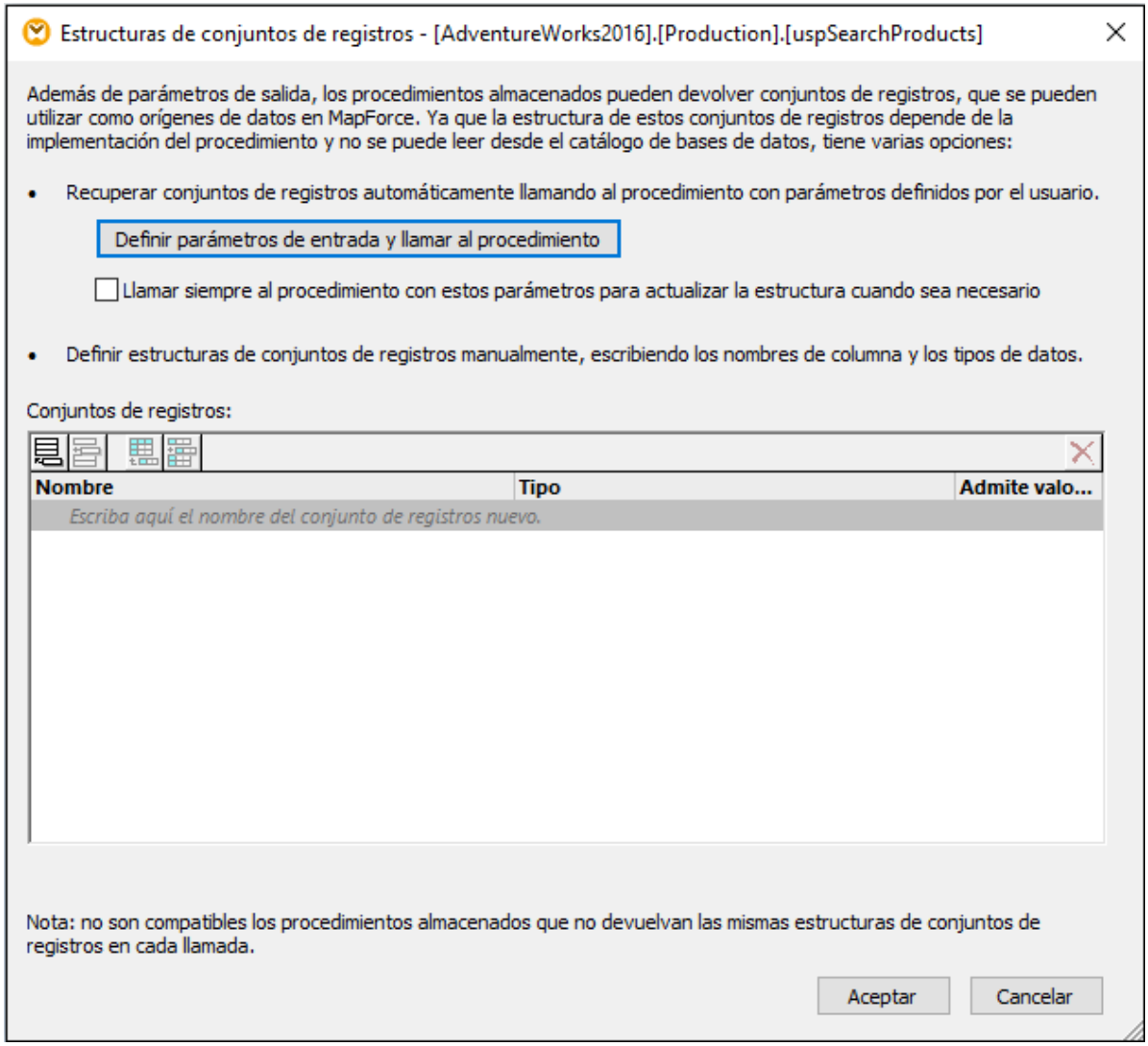
Para crear una asignación que consuma datos devueltos por este procedimiento siga los pasos que explicamos a continuación:

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 319. 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.



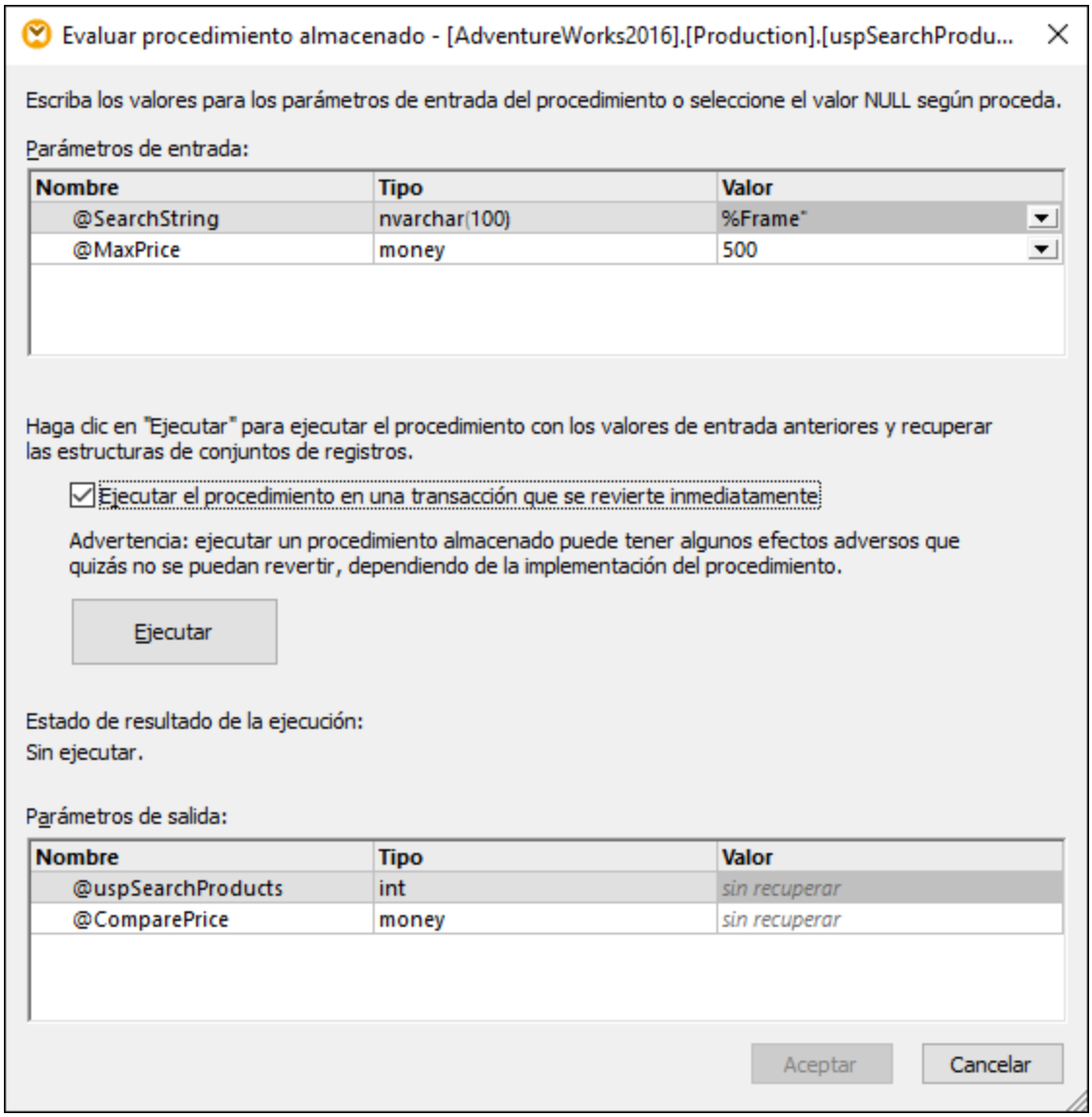

3. Vuelva a hacer clic en el botón para mostrar el menú contextual () y seleccione **Editar estructuras de conjuntos de registros**. Esto es necesario para suministrar a MapForce la información sobre la estructura de los registros devuelta por el procedimiento. Aparece el cuadro de diálogo "Estructuras de conjuntos de registros".

 © 2019-2025 Altova GmbH

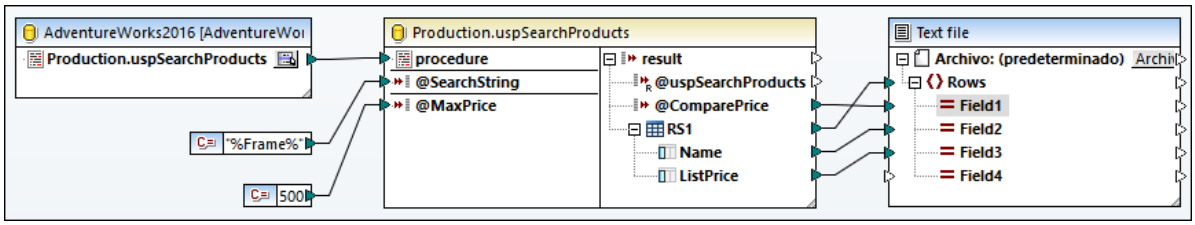



Llamar a un procedimiento almacenado en el momento de diseñar la asignación puede tener efectos secundarios (dependiendo de la implementación del procedimiento). Si no quiere ejecutar el procedimiento almacenado en el momento de diseñar la asignación, entonces no haga clic en **Ejecutar**, como se describe 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".

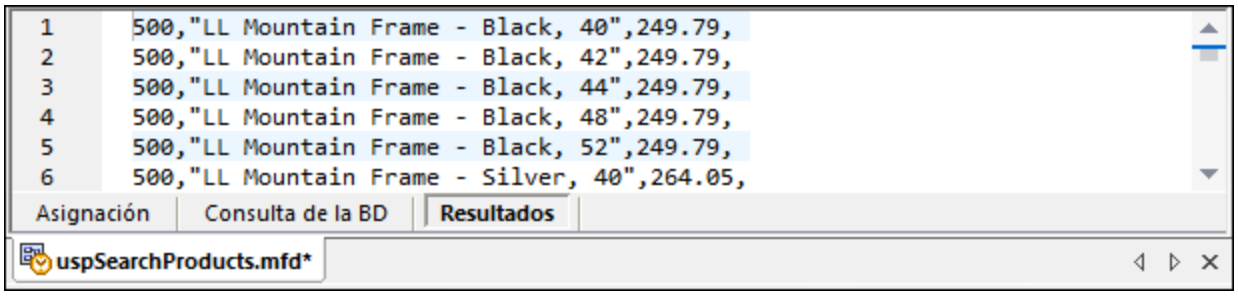
4. Haga clic en **Definir parámetros de entrada y llamar al procedimiento**. Aparecerá el cuadro de diálogo "Evaluar procedimiento almacenado".




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* 344). 344. 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 462.




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:




## 4.2.6.4 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 325.

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* 294). En ambos casos, asegúrese de que su cuenta de usuario de la base de datos tiene permisos para crear procedimientos almacenados.

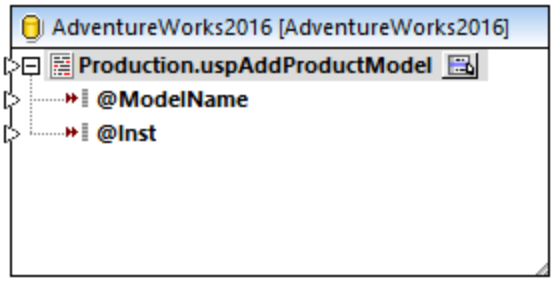
```
CREATE PROCEDURE Production.uspAddProductModel
   @ModelName nvarchar(50)
   ,@Inst xml
AS
BEGIN
INSERT INTO [Production].[ProductModel]
           ([Name]
           ,[Instructions]
           ,[rowguid]
           ,[ModifiedDate])
     VALUES
           (@ModelName
```

```
            ,@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.

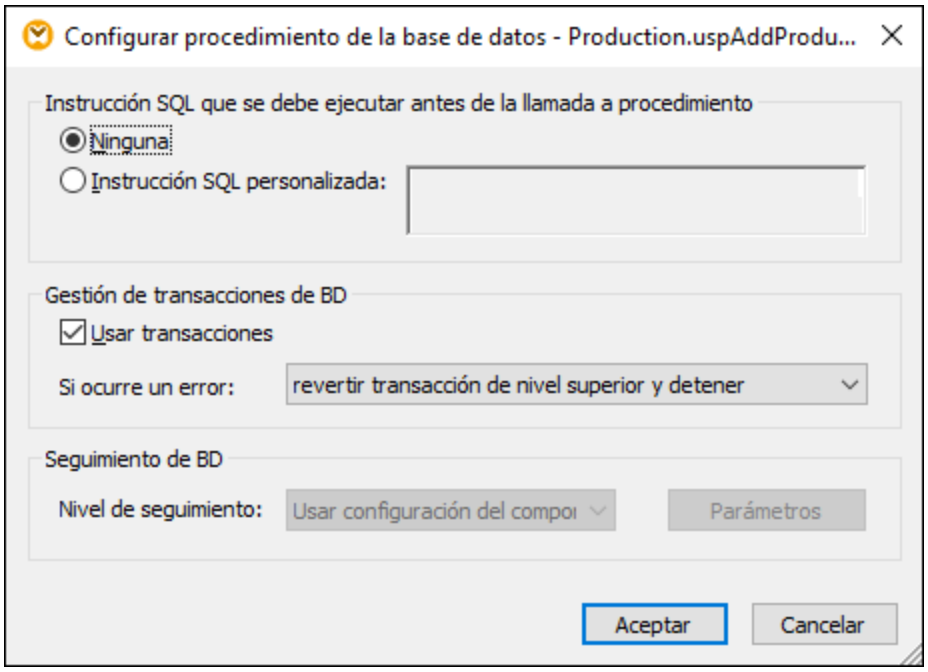
Los siguientes pasos explican cómo crear una asignación que consuma fatos devueltos por este procedimiento.

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 319. 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 **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.



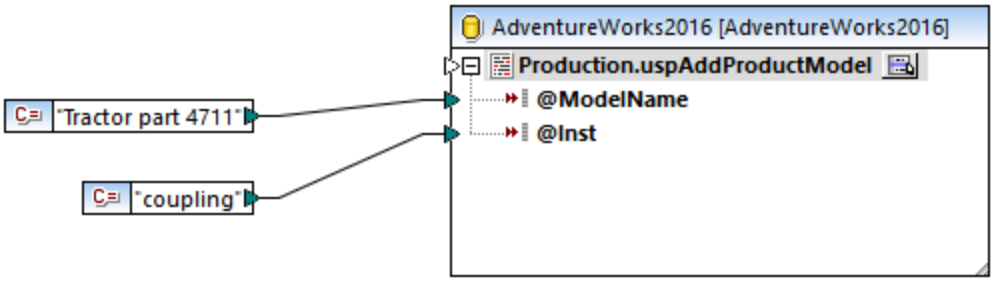

3. Vuelva a hacer clic en el botón **Mostrar el menú contextual** y seleccione **Configuración del procedimiento**. Este paso opcional permite ejecutar el procedimiento almacenado dentro de una transacción que se puede revertir en caso de error. También puede añadir una instrucción SQL personalizada que se ejecute antes de llamar al procedimiento.
4. Marque la casilla **Usar transacciones**.

 © 2019-2025 Altova GmbH




**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.

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 462.




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.

```
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 sólo 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 43). 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").

## Seguimiento a nivel de procedimiento almacenado

Para habilitar el seguimiento a nivel de procedimiento almacenado:

1. Asegúrese de que el nivel de seguimiento a nivel de los componentes de DB es **Siempre** o **Error** (véase más arriba).
2. 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.

 © 2019-2025 Altova GmbH

## 4.2.6.5 Procedimientos almacenados y relaciones locales

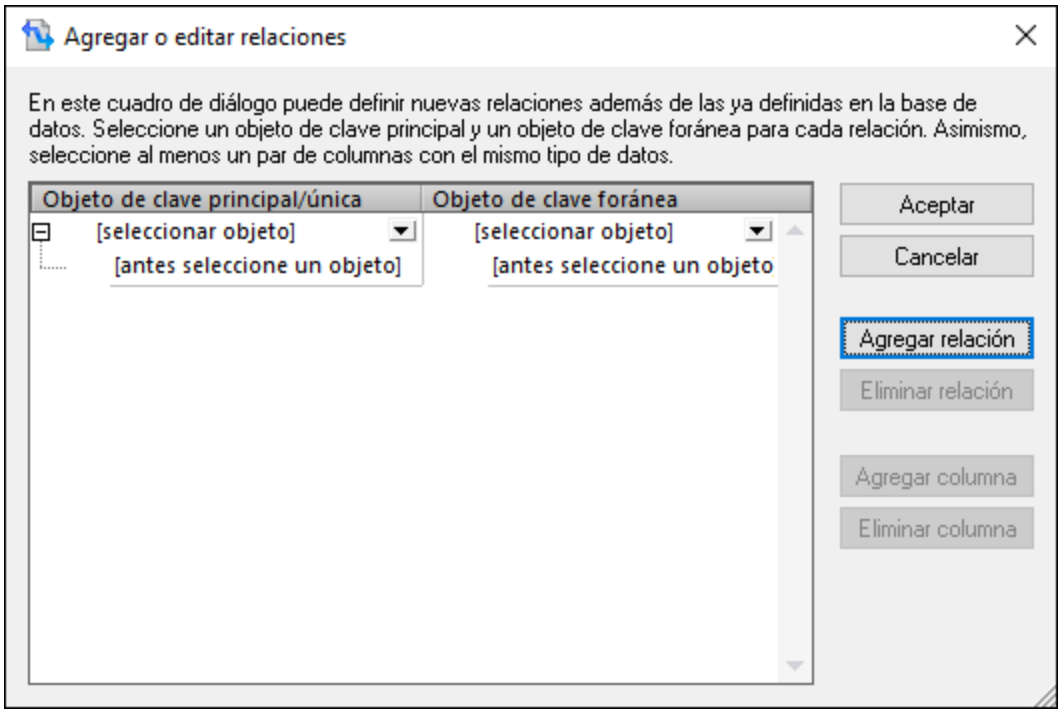
Las relaciones locales son relaciones lógicas entre campos de BD que puede crear en MapForce, lo que le evita tener que cambiar la BD subyacente (*consulte también* *Definir relaciones locales* 269). Puede definir relaciones locales no sólo para campos de BD sino también para procedimientos almacenados, 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.

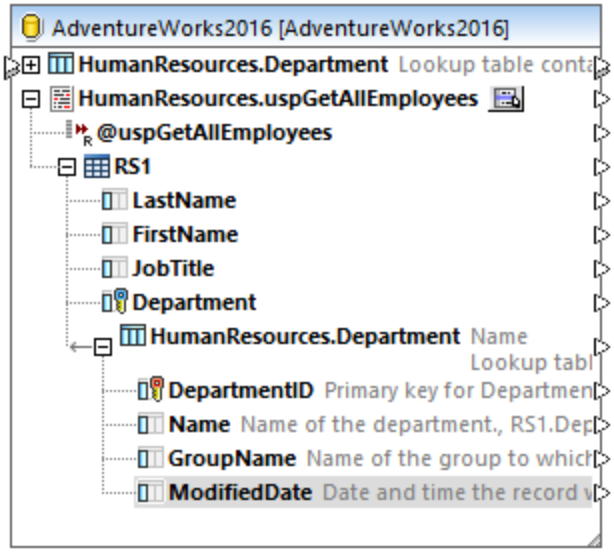
En componentes de destino las relaciones locales permiten definir un orden jerárquico con el que se puede llamar a varios procedimientos relacionados. Por ejemplo, puede llamar primero a un procedimiento almacenado que cree un valor de ID y a otro que inserte información relacionada en una tabla. En 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* 338).

**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 335.




La siguiente tabla enumera todos los campos posibles entre los que puede definir relaciones locales. Es posible crear relaciones mixtas (p.ej., asignar los resultados de un procedimiento almacenado a una columna de BD). Los tipos de datos de los campos que forman parte de la relación deben ser iguales o compatibles.

| Clave principal/única | Clave foránea |
|---|---|
| • Columna de una tabla o vista de BD<br>• Parámetro de salida o valor de retorno de un procedimiento almacenado (consulte también Procedimientos almacenados 315)<br>• Columna de conjunto de registros devuelta por procedimiento almacenado. 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.<br>• Columna de una instrucción SELECT definida por el usuario (consulte también Instrucciones SQL SELECT como tablas virtuales 259). | • Columna de una tabla o vista de BD<br>• Parámetro de entrada de un procedimiento almacenado<br>• Parámetro de entrada de una instrucción SELECT definida por el usuario |

 © 2019-2025 Altova GmbH

## 4.2.6.6 Relaciones locales en componentes de origen

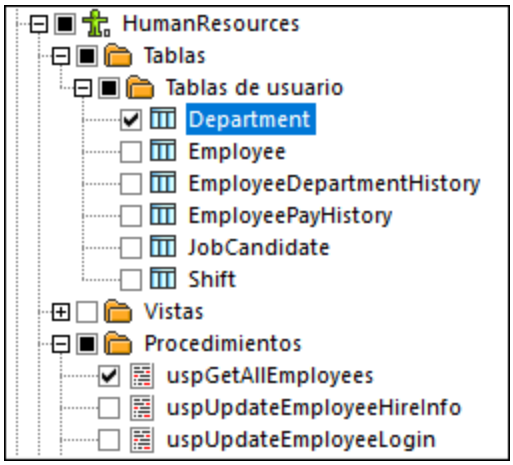
En este ejemplo explicamos cómo combinar datos devueltos por un procedimiento almacenado con datos de una tabla de la misma base de datos, con ayuda de las relaciones locales.

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* 294). En ambos casos, asegúrese de que su cuenta de usuario de la base de datos tiene permisos para crear procedimientos almacenados.

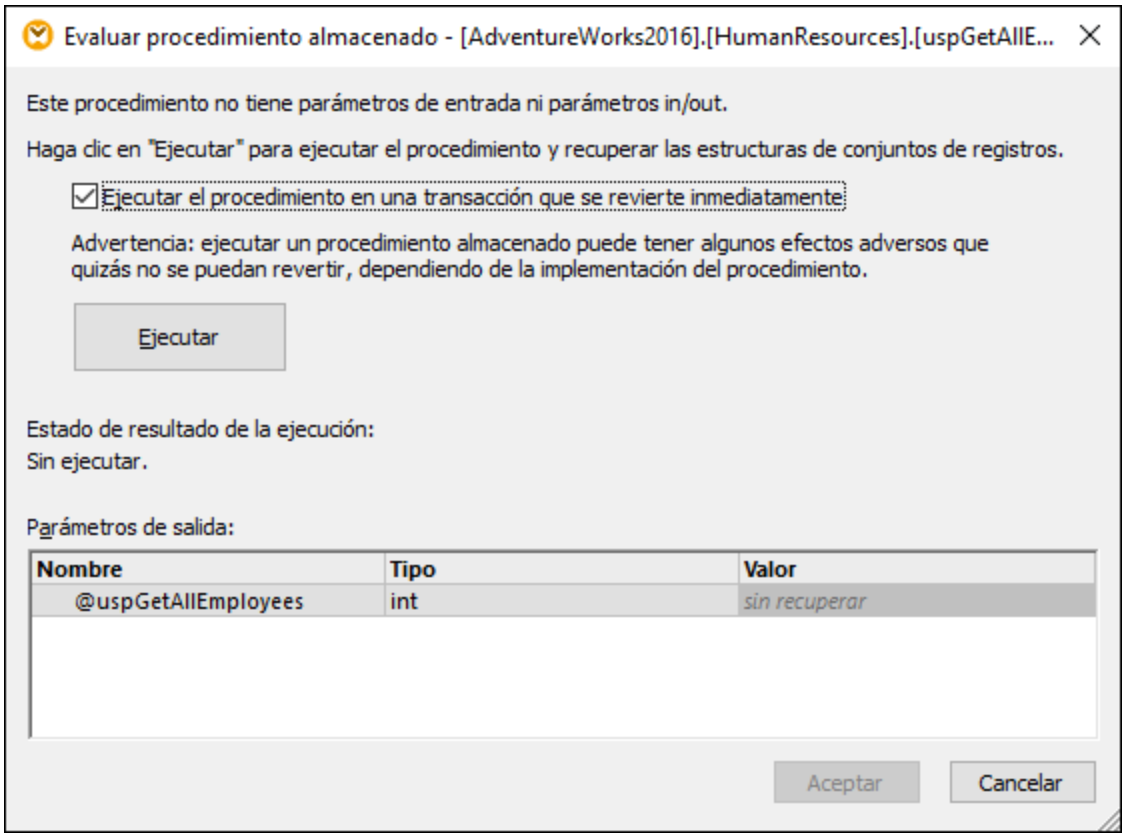
```
CREATE PROCEDURE HumanResources.uspGetAllEmployees
AS
    SELECT LastName, FirstName, JobTitle, Department
    FROM HumanResources.vEmployeeDepartment
```

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.

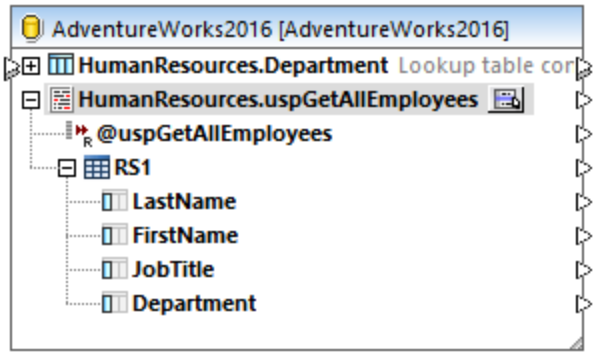
1. Conéctese a la base de datos "AdventureWorks" desde MapForce, como se describe en el apartado Agregar procedimientos almacenados a la asignación 319. Asegúrese de que su cuenta de usuario de la base de datos tiene permisos para crear procedimientos almacenados.
2. Cuando la aplicación solicite que 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".




- 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.

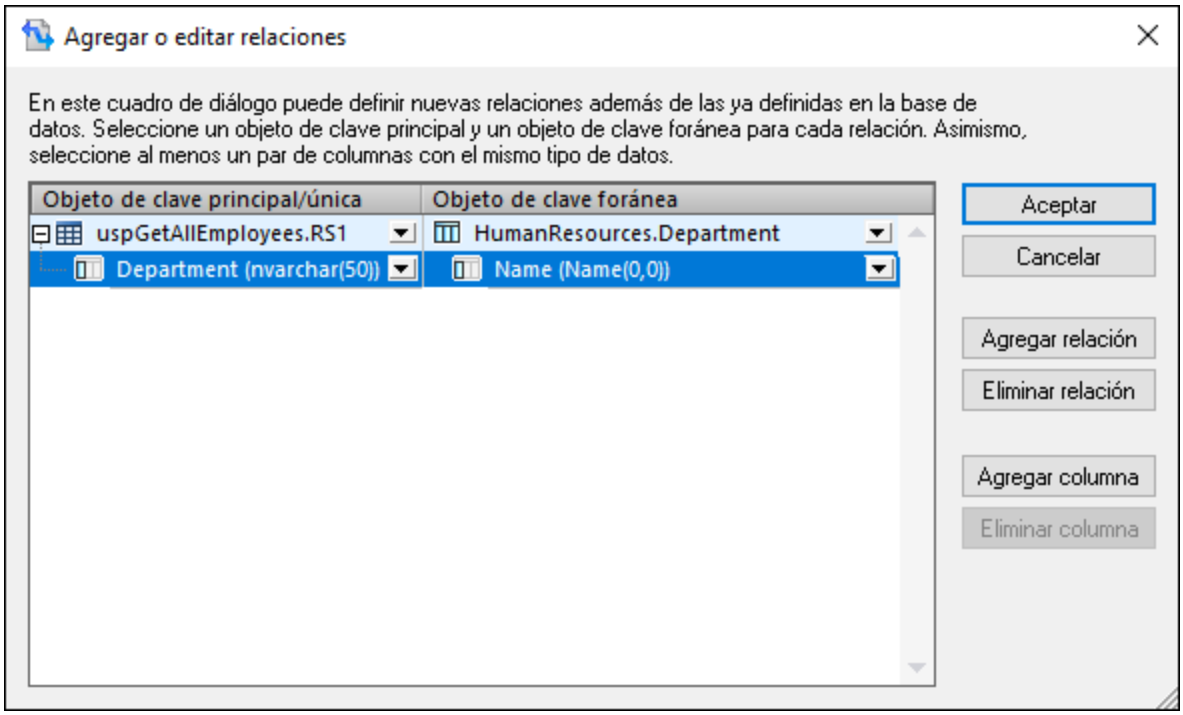



## Definir relaciones locales

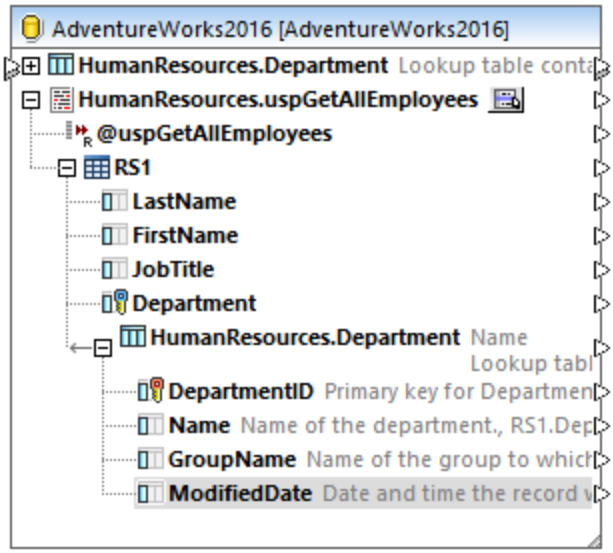
Ahora vamos a definir una relación local entre la columna **Department** del conjunto de registros devuelto y la columna **Name** de la tabla **Department**.

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.

 © 2019-2025 Altova GmbH




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**.

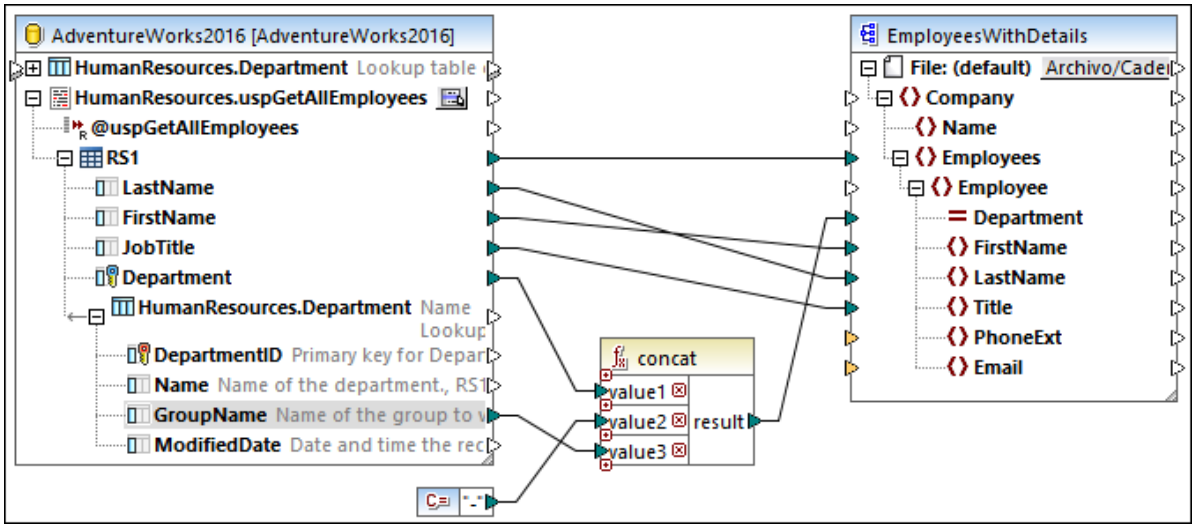



## Completar la asignación

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:

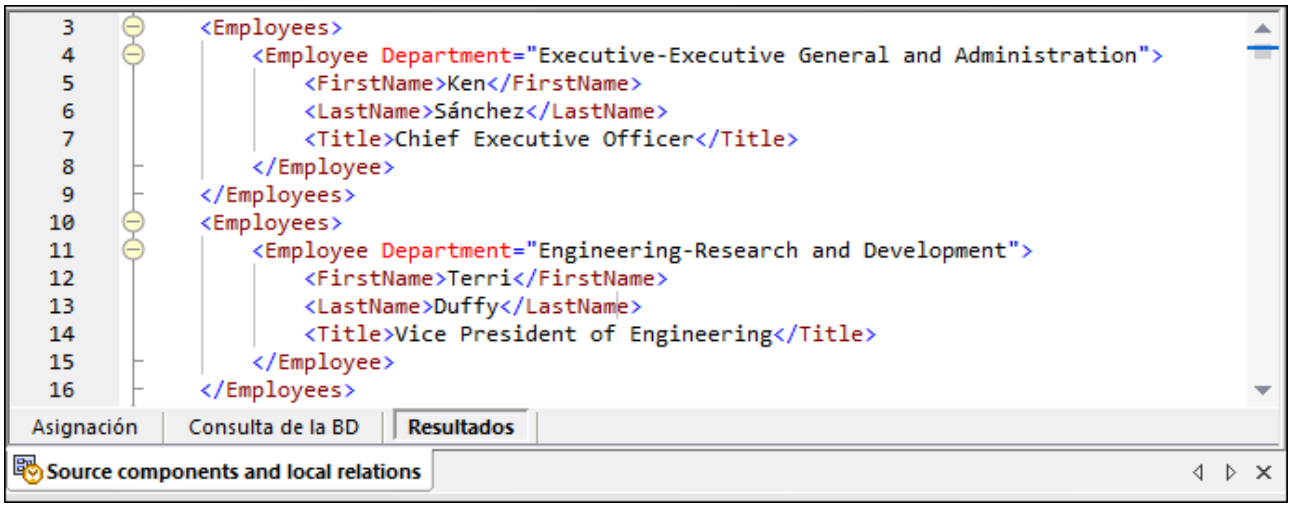
1. En el menú Insertar, haga clic en **Archivo o esquema XML** y seleccione el siguiente archivo: **<Documentos>\Altova\MapForce2025\MapForceExamples\EmployeesWithDetails.xsd**.

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** 626 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

## 4.2.6.7 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

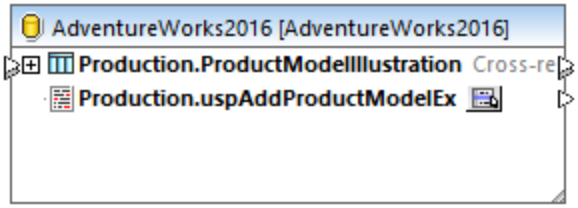
ello, ejecute el siguiente script en la BD. Puede hacerlo desde una ventana de consultas de la aplicación de **Microsoft SQL Server Management Studio** o directamente desde la pestaña *Consulta de la BD* de MapForce (*consulte* *Examinar y consultar bases de datos* 294). En ambos casos, asegúrese de 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
```

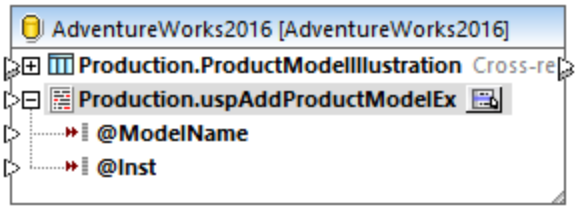
El procedimiento almacenado anterior toma dos parámetros (`INSERT`) como entrada y ejecuta una operación `INSERT` en la tabla **ProductModel**. Entonces devuelve el ID generado `@ProductModelID` como parámetro de salida. El requisito es insertar el `@ProductModelID` devuelto por el procedimiento almacenado en la tabla **ProductModelIllustration**.

Para crear una asignación que cumpla con el requisito anterior siga los pasos que describimos a continuación.

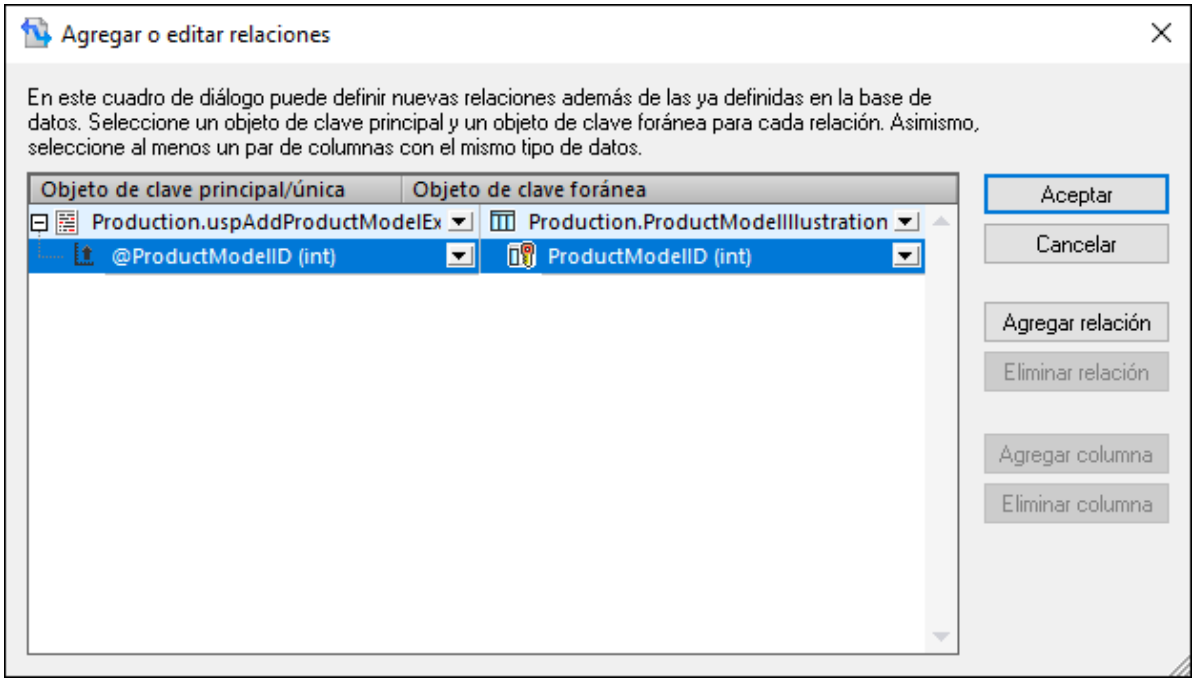
1. Conéctese a la base de datos "AdventureWorks" desde MapForce, como se describe en el apartado Agregar procedimientos almacenados a la asignación 319. Asegúrese de que su cuenta de usuario de la base de datos tiene permisos para crear procedimientos almacenados.
2. Cuando la aplicación solicite que 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.

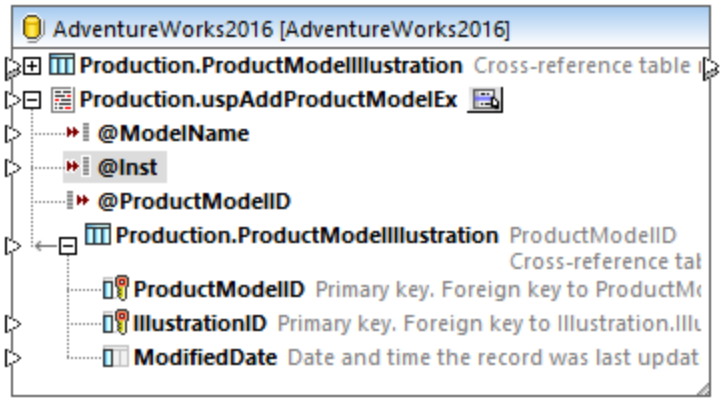



4. Otra opción, si quiere ejecutar el procedimiento almacenado dentro de una transacción, es volver a hacer clic en el botón para mostrar el menú contextual (), seleccionar **Configuración del procedimiento** y luego marcar la casilla **Usar transacciones**. Al definir la transacción para el procedimiento 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.

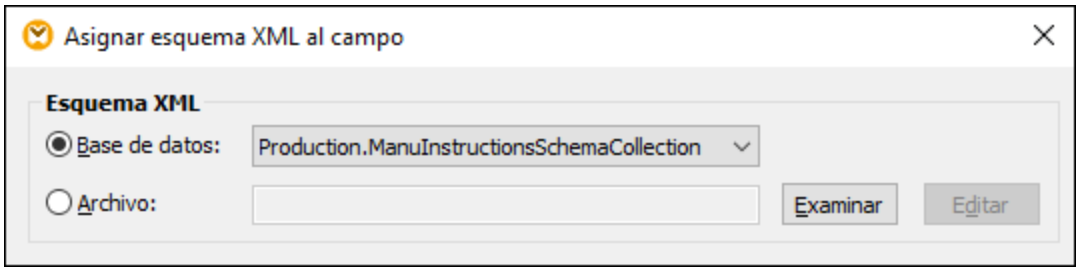



7. Haga clic en **Aceptar** para cerrar el cuadro de diálogo. Observe que la tabla **ProductModelIllustration** ahora aparece como secundaria del procedimiento almacenado. El parámetro de salida del procedimiento almacenado (`@ProductModelID`) se muestra como un indicador que se usará en la relación local, pero no tiene conectores de entrada ni de salida.

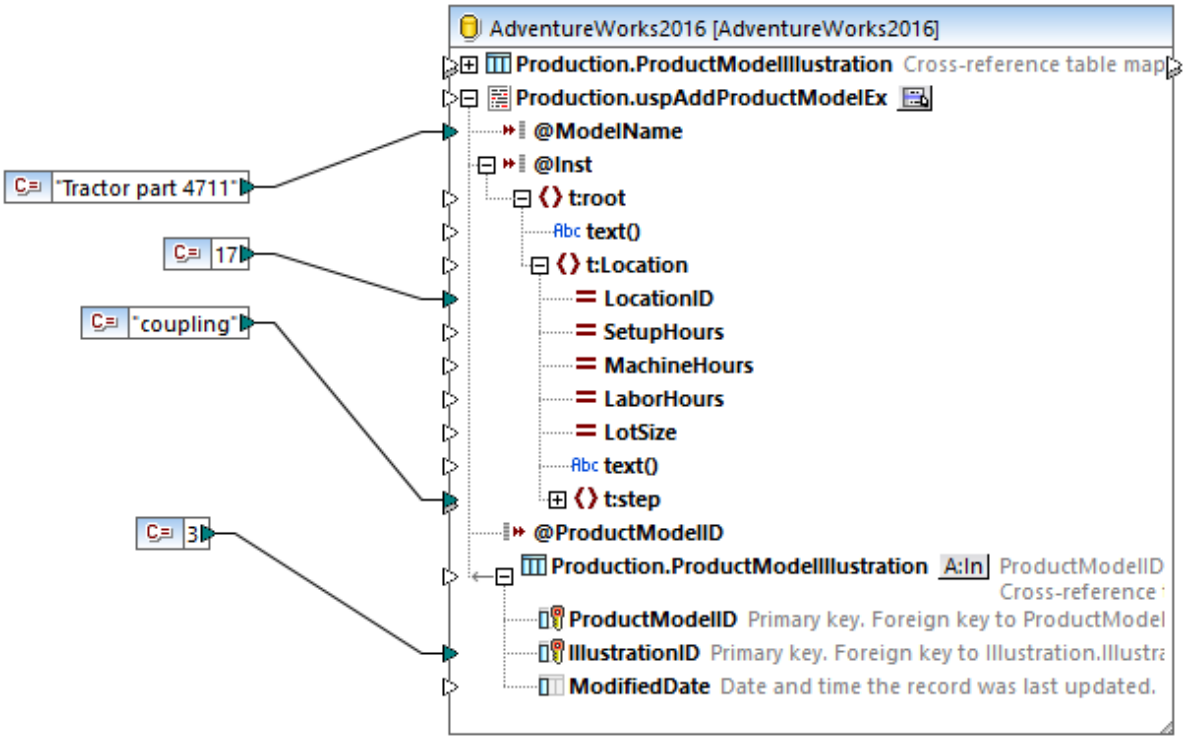
 © 2019-2025 Altova GmbH




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 304.




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 462.




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 sólo pistas (como comentarios). Los comandos SQL de verdad se envían a la API de BD subyacente.

 © 2019-2025 Altova GmbH

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** .

# 4.3 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). En MapForce, los archivos CSV y FLF son componentes estructurales que se pueden usar como orígenes y destinos de datos.

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:

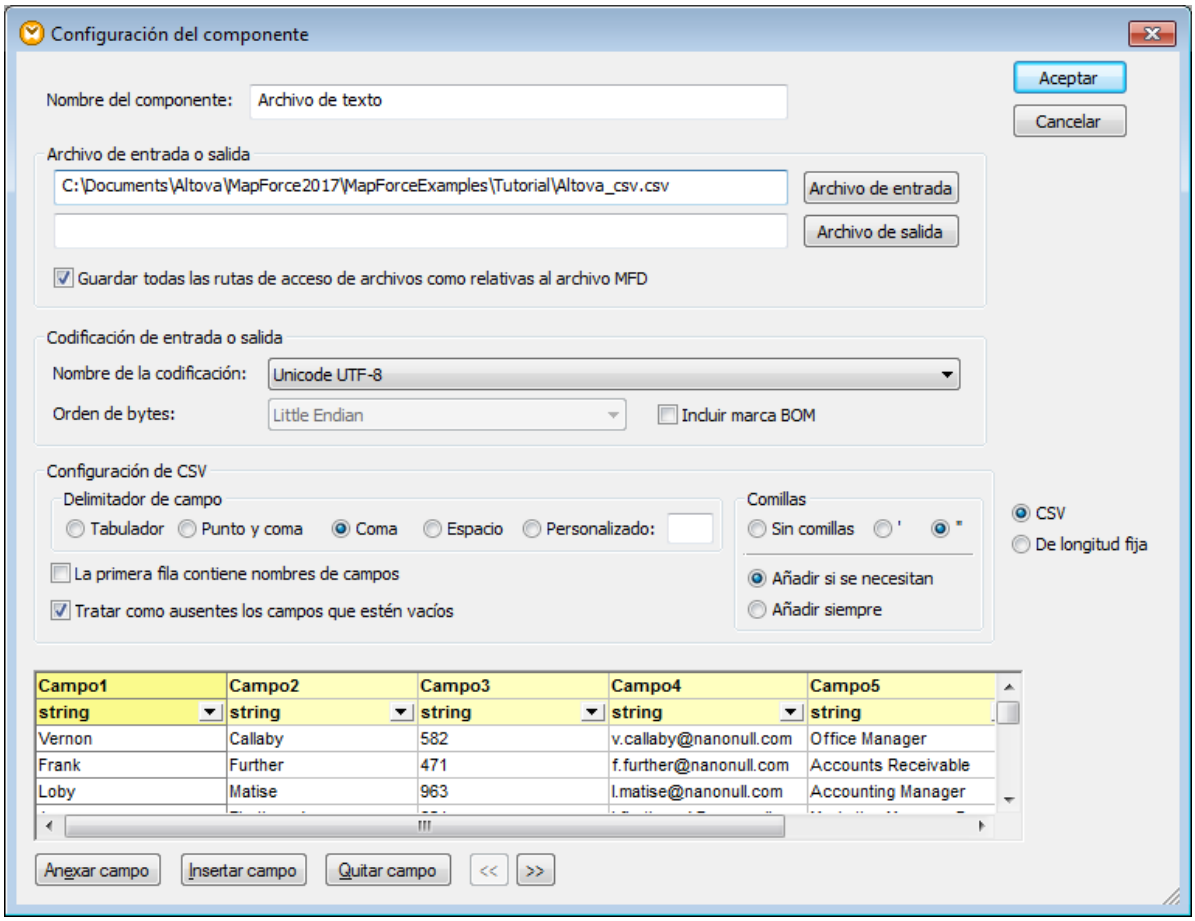
- Abriendo el panel *Resultados*: esto genera una vista previa con el motor de ejecución integrado. También puede guardar el resultado de la asignación con el comando de menú **Resultados | Guardar el archivo de salida...** o con el botón **Guardar resultado generado** de la barra de herramientas del 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

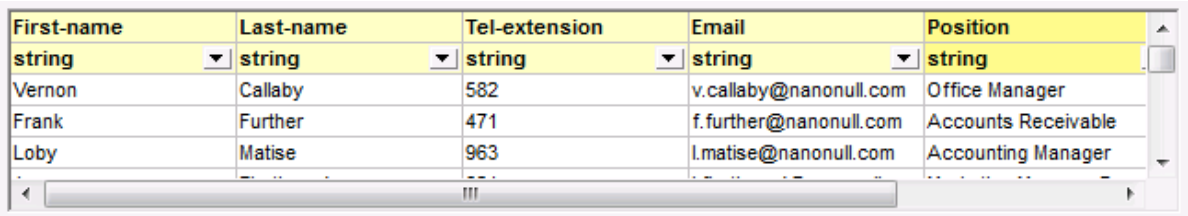
El objetivo de este ejemplo es crear una asignación que lea datos de un archivo CSV simple y los escriba en un archivo XML de destino. Los archivos utilizados en este ejemplo se encuentran en la carpeta `<Documentos>\Altova\MapForce2025\MapForceExamples\Tutorial\`.

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 sólo aparecen las primeras 20 filas del archivo de texto.


© 2019-2025 Altova GmbH

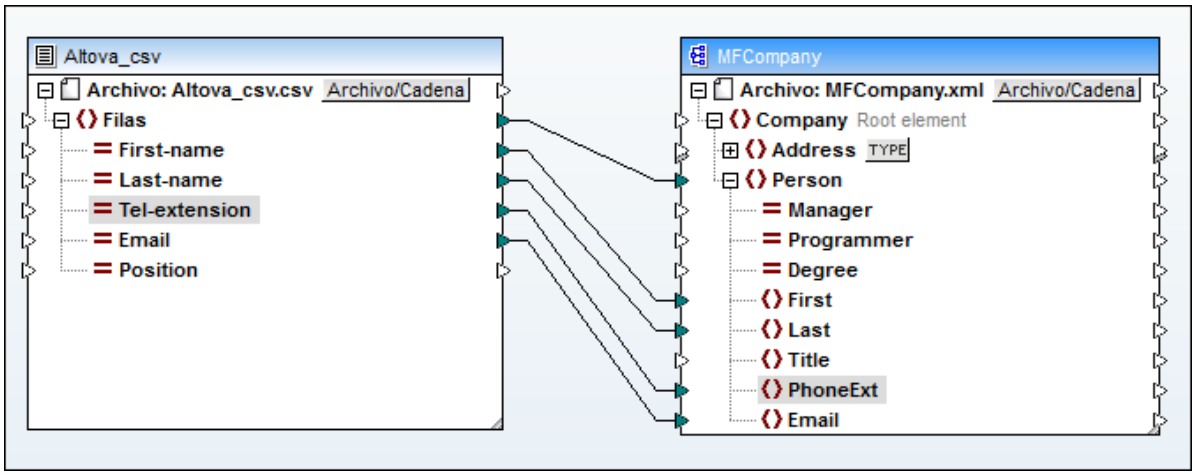



4. Haga clic dentro del encabezado Campo1 y cambie el texto por First-name. Haga lo mismo con el resto de campos: Campo 2 => Last-name, Campo 3 =>Tel-extension, Campo 4 => Email, Campo 5 => Position. **Nota:** pulse la tecla **Tabulación** para recorrer los campos.




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).

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.

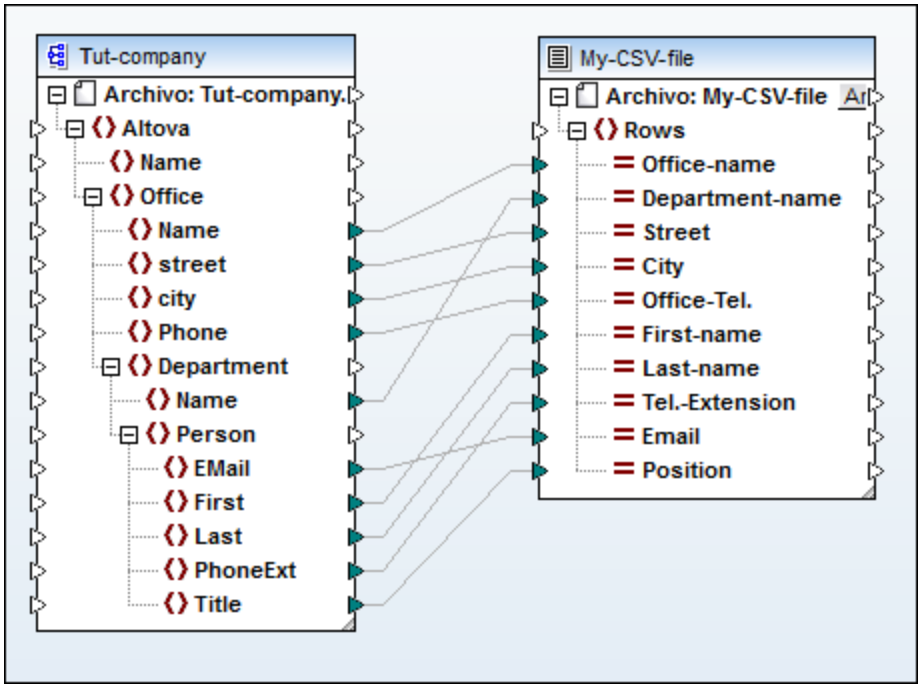
```
1   <?xml version="1.0" encoding="UTF-8"?>
2   <Company xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:
3     <Person Manager="true">
4       <First>Vernon</First>
5       <Last>Callaby</Last>
6       <PhoneExt>582</PhoneExt>
7       <Email>v.callaby@nanonull.com</Email>
8     </Person>
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 la asignación que acompaña este ejemplo está en la ruta de acceso `<Documentos>\Altova\MapForce2025\MapForceExamples\Tutorial\Tut-xml2csv.mfd`.

 © 2019-2025 Altova GmbH

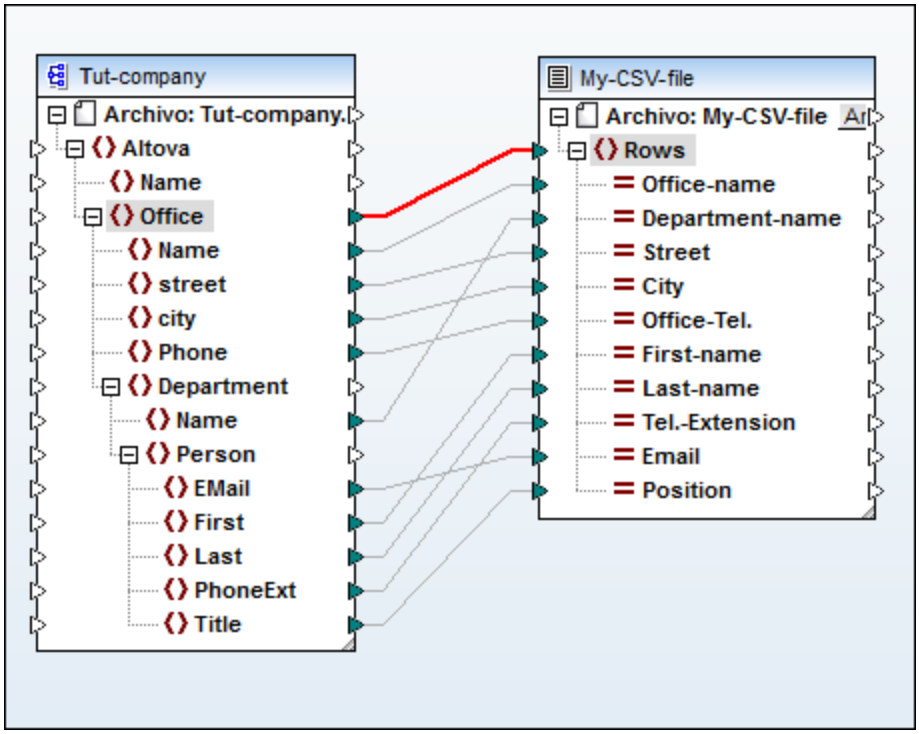



*Tut-xml2csv.mfd*

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 necesita 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.

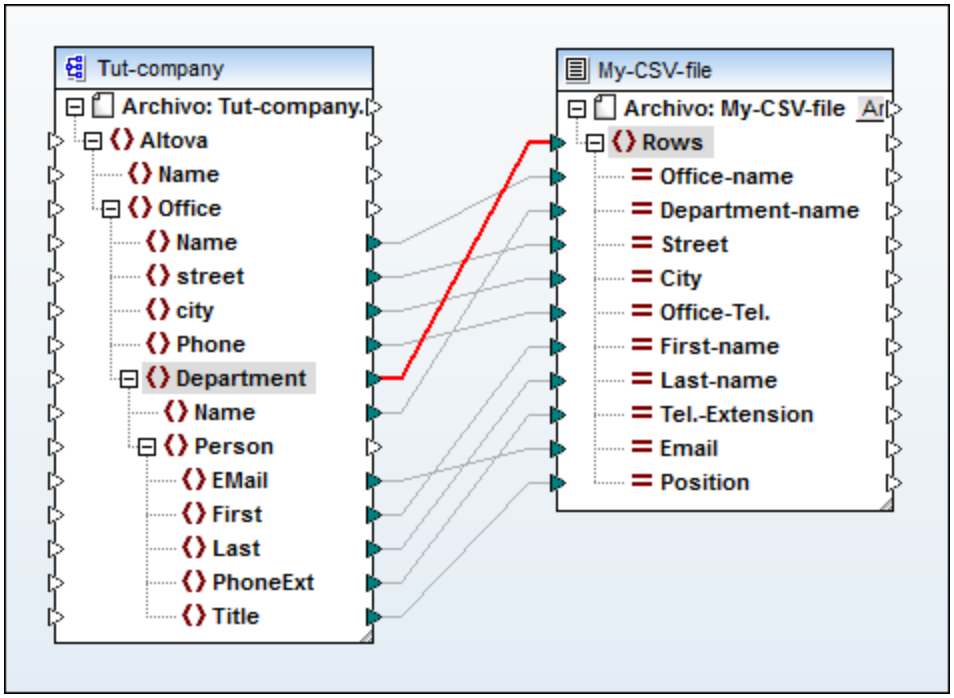



El elemento `Rows` del componente CSV hace de iterador para la secuencia de elementos a los que está conectado. Por tanto, si conecta el elemento `Office`, el resultado crea una fila por cada oficina que se encontrara en el archivo XML de origen.

```
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.

 © 2019-2025 Altova GmbH




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 FLF

Este ejemplo está disponible en la siguiente ruta de acceso: **<Documentos>\Altova\MapForce2025\MapForceExamples\Tutorial\Tut-headerDetail.mfd**. El ejemplo utiliza un archivo CSV (Pedidos.csv) que tiene el siguiente formato:

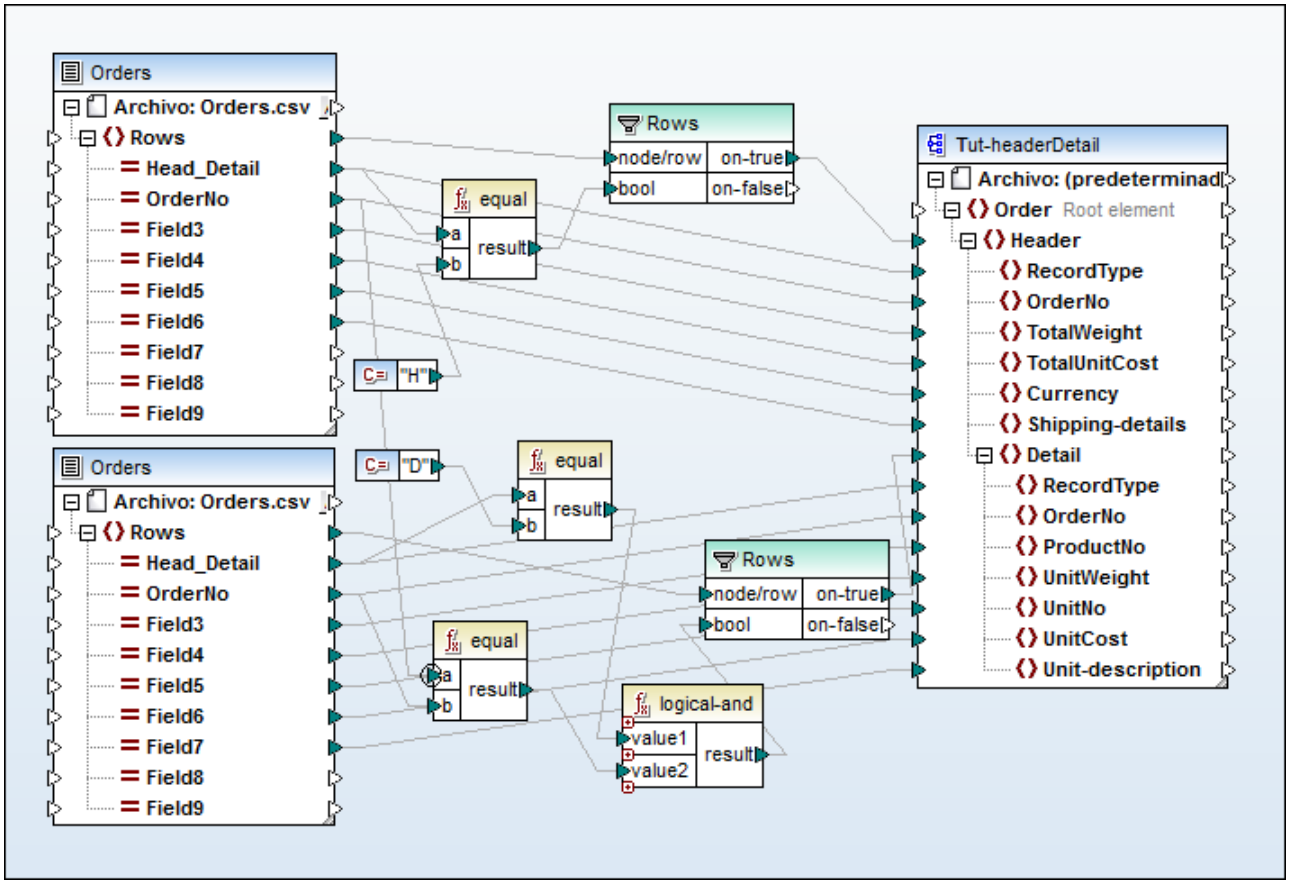
- Campo 1: H define un registro de encabezado y D un registro de detalles.
- Campo 2: una clave común para los registros de encabezado y de detalles.
- Cada registro de encabezado o de detalles está en una línea distinta.

A continuación, puede ver el contenido del archivo Orders.csv.

```
H,111,332.1,22537.7,,Container ship,,,
D,111,A-1579-227,10,3,400,Microtome,,
D,111,B-152-427,7,6,1200,Miscellaneous,,
H,222,978.4,7563.1,,Air freight,,,
D,222,ZZ-AW56-1,10,5,10000,Gas Chromatograph,,
```

El objetivo de esta asignación de datos es:

- Asignar el archivo CSV plano a un archivo XML jerárquico.
- Filtrar los registros de encabezado (señalados con una H).
- Asociar los correspondientes registros de detalles (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 de detalles 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 de detalles 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.

 © 2019-2025 Altova GmbH

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. Esto se consigue de esta manera:

- Comparando el campo **OrderNo** del registro de encabezados con el mismo campo de los registros de detalles, utilizando la función **equal** (el contexto de prioridad 89 se establece en el parámetro **a** para mejorar el rendimiento).
- Utilizando la función **Logical-and** para suministrar sólo aquellos registros Detail que contengan el mismo campo OrderNo que el registro Header.

El elemento **Rows** suministra estos registros filtrados a los elementos Header y Detail del archivo de esquema, a través del parámetro on-true del componente de filtro.

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>
```

Veamos ahora otro ejemplo, que utiliza un archivo CSV ligeramente diferente y está disponible en la carpeta **<Documentos>\Altova\MapForce2025\MapForceExamples\Tutorial** como **Head-detail-inline.mfd**. La diferencia es que:

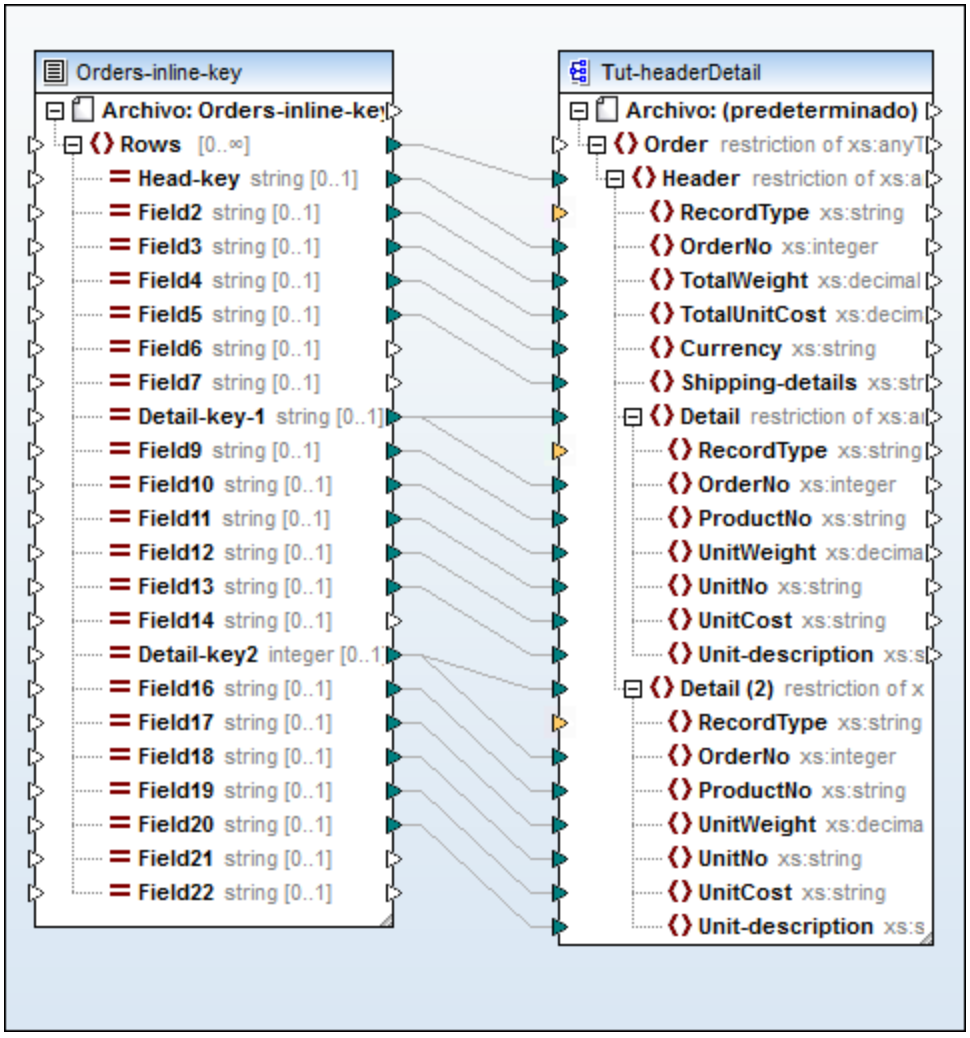
- No hay ningún designador de registro (H, o D) disponible
- Sigue existiendo un campo clave común, el primer campo del archivo CSV, tanto para los registros de encabezado como para los de detalles (Head-key, Detail-key...). El campo se asigna a OrderNo en el esquema destino.
- Los campos Header y todos Detail respectivos están todos 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,,
```

La asignación de datos se ha diseñado de la siguiente manera:

- Los campos clave se asignan a los respectivos elementos OrderNo en el esquema de destino.
- El elemento Detail del archivo de destino del esquema se ha duplicado y aparece como **Detail (2)**. Esto le permite asignar el segundo conjunto de registros de detalles al elemento correcto.
- El resultado de la asignación es el mismo archivo XML que se generó en el primer ejemplo de este apartado.


© 2019-2025 Altova GmbH

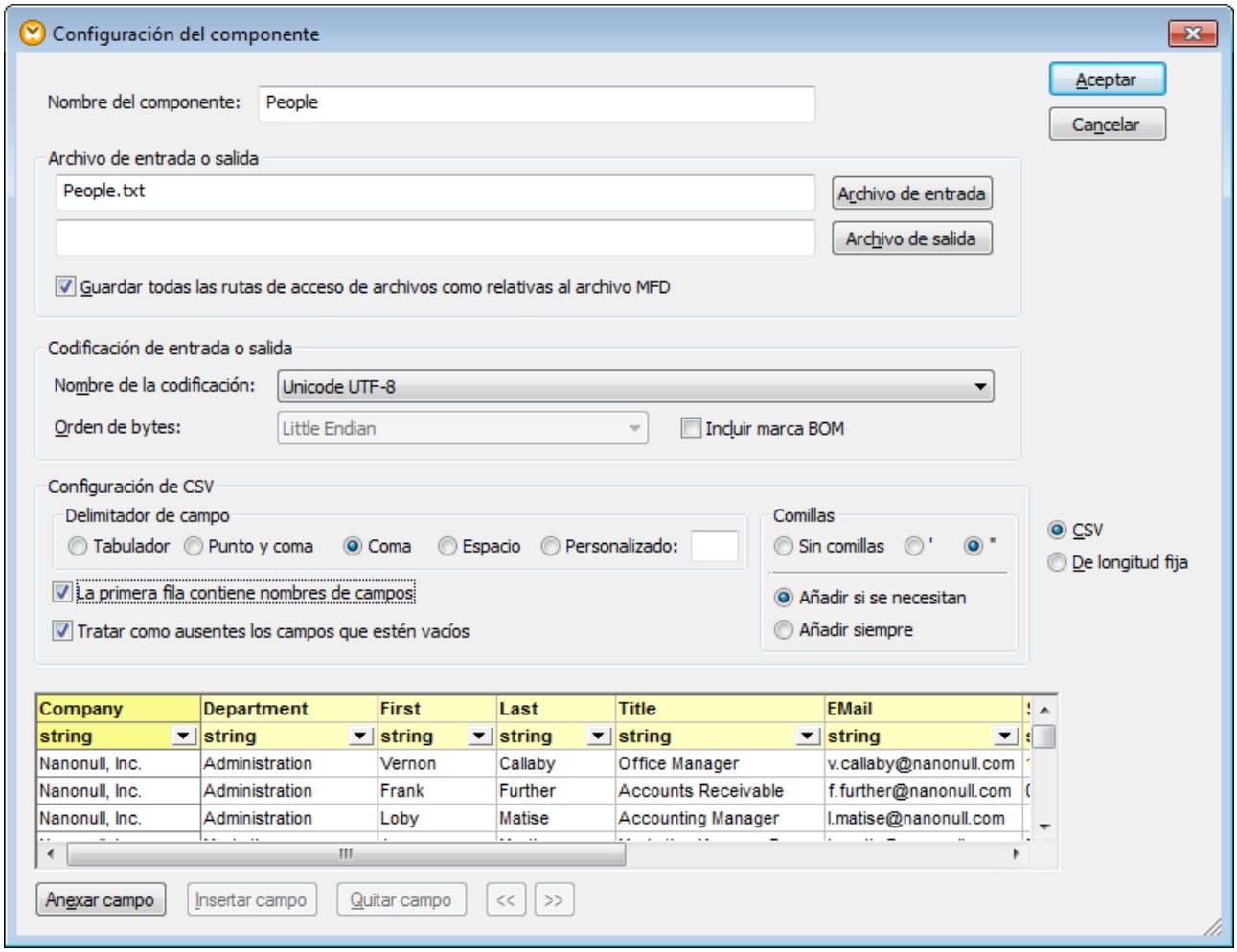



*Head-detail-inline.mfd*

## 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.




*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.<br><br>Si cambia el nombre del componente, tenga en cuenta que:<br><br>• Si quiere implementar la asignación en FlowForce Server, el nombre del componente debe ser único.<br>• 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. |

 © 2019-2025 Altova GmbH

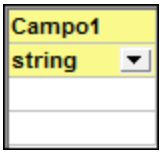
| | |
|---|---|
| *Archivo de entrada* | Especifica el archivo donde MapForce debe leer los datos. Este campo es relevante para componentes de origen y se rellena cuando se crea el componente y se le asigna un archivo de texto. El campo puede dejarse vacío si se usa un componente de archivo de texto como destino de la asignación.<br><br>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.<br><br>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.<br><br>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* 44). |
| *Codificación de entrada / salida* | Permite configurar estos aspectos del archivo de instancia de salida:<br><br>• Nombre de la codificación<br>• Orden de bytes<br>• El carácter de la marca de orden de byte (marca BOM).<br><br>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.<br><br>Por ejemplo, el registro CSV `"General outgassing pollutants,,,,"` está compuesto por cuatro campos y los tres últimos están vacíos. |

| | |
|---|---|
| | 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`):<br><br>33 <Person><br>34 <First>General outgassing pollutants</First><br>35 <Last/><br>36 <Title/><br>37 <Email/><br>38 </Person><br><br>Si se activa esta opción, los campos vacíos no se crearán en el destino de la asignación:<br><br>38 <Person><br>39 <First>General outgassing pollutants</First><br>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.<br><br>Comillas<br>Sin comillas ' "<br>Añadir si se necesitan<br>Añadir siempre<br><br>Además, los archivos de salida cuentan con dos opciones de configuración más:<br><br>**Añadir si se necesitan** — Añade el carácter de comillas seleccionado en campos cuyo texto contiene el delimitador de campo o saltos de línea.<br><br>**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.<br><br>Si lo necesita, puede crear la estructura del archivo (o cambiarla) con estos botones: |


© 2019-2025 Altova GmbH

| | |
|---|---|
| **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 Asignar archivos FLF 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 **`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:

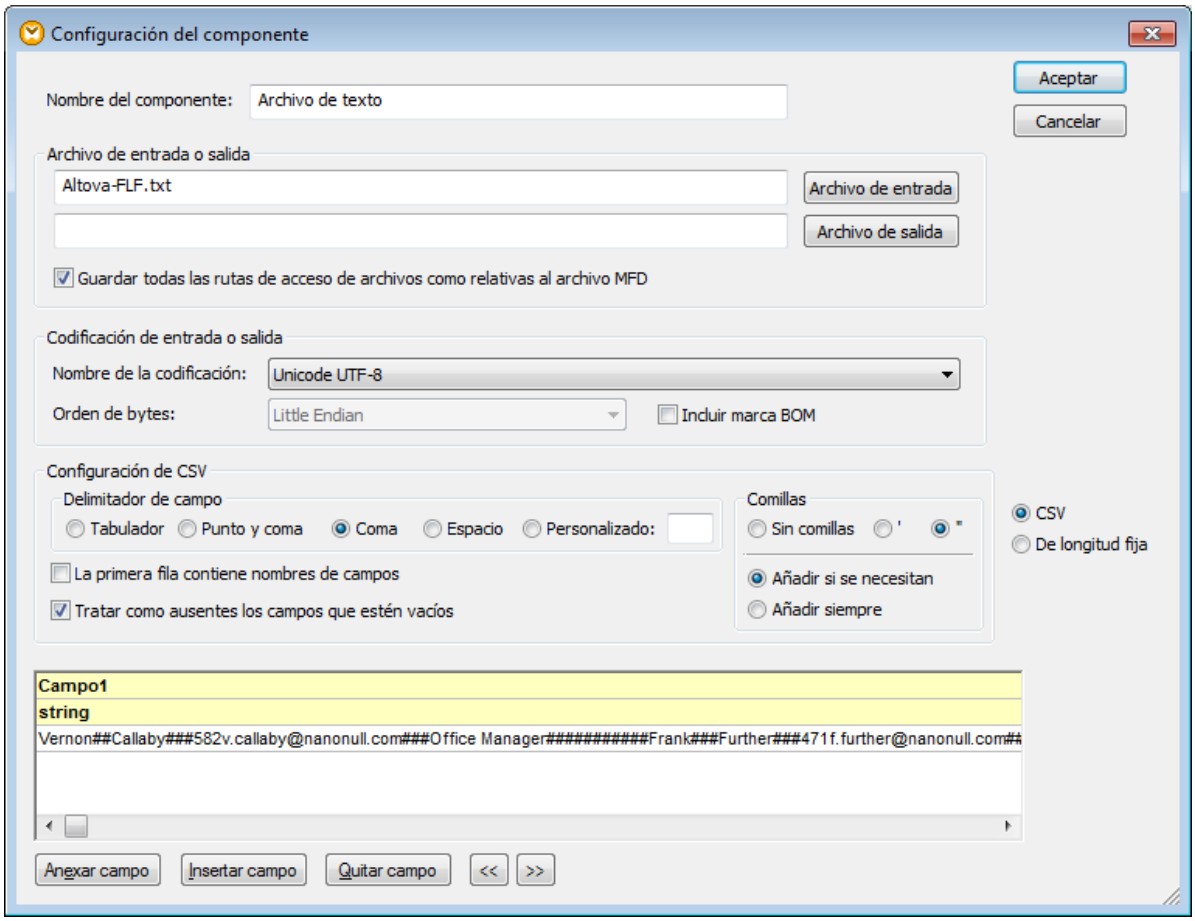
| Posición y nombre del campo | Tamaño (en caracteres) |
|---|---|
| Campo 1 (Nombre) | 8 |

| Posición y nombre del campo | Tamaño (en caracteres) |
|---|---|
| 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 **`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

 © 2019-2025 Altova GmbH

campo y pulse **Entrar**. Ahora pueden verse más datos en la primera columna, que tiene un ancho de 8 caracteres.

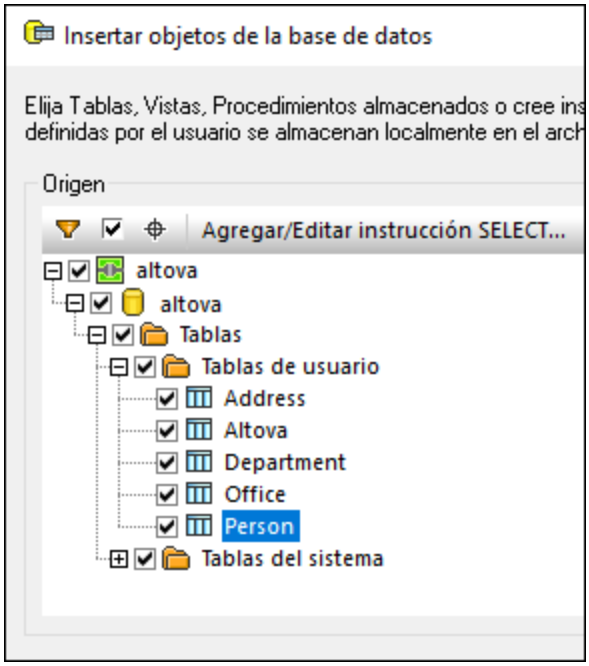
6. Haga clic en **Anexar campo** para añadir un campo nuevo y establezca su longitud en 10 caracteres.
7. Use el mismo método para crear tres campos más con una longitud de 3, 25 y 25 caracteres respectivamente. Ahora cambie el nombre de los 5 campos por: First, Last, Tel.-Ext, Email, Title. Esta sería 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.

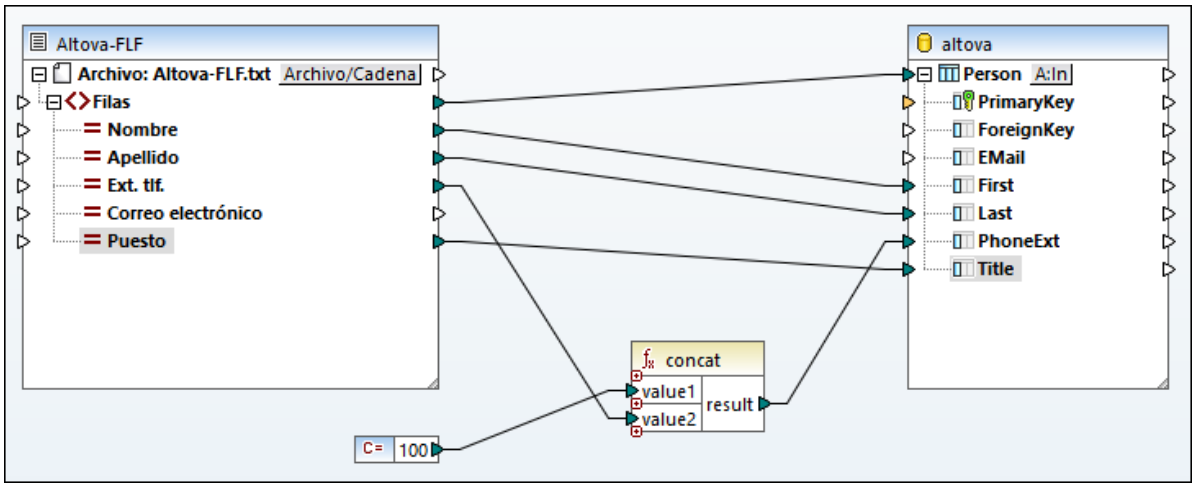
## Paso nº2: insertar el componente de base de datos

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.sqlite`** y después haga clic en **Conectarse**.
3. Seleccione la tabla **`Person`** y haga clic en **Aceptar**.



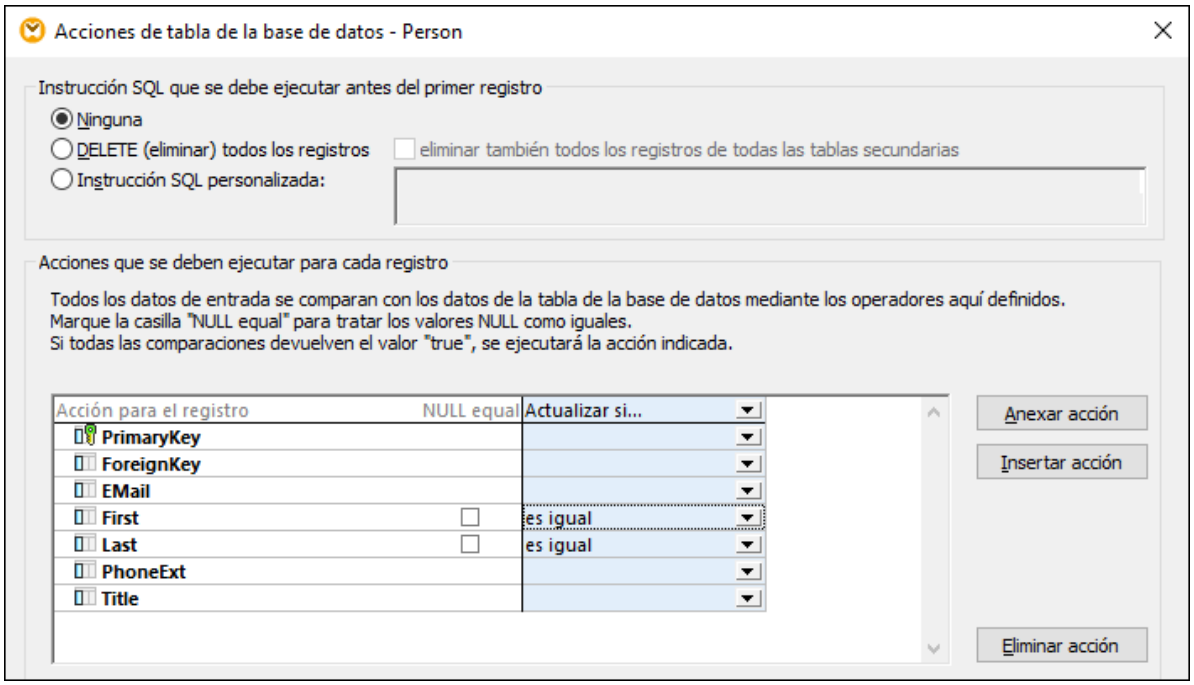
## Paso nº3: diseñar la asignación

1. Arrastre la función `core | concat` 626 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 **Acciones 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 sólo 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**.

 © 2019-2025 Altova GmbH

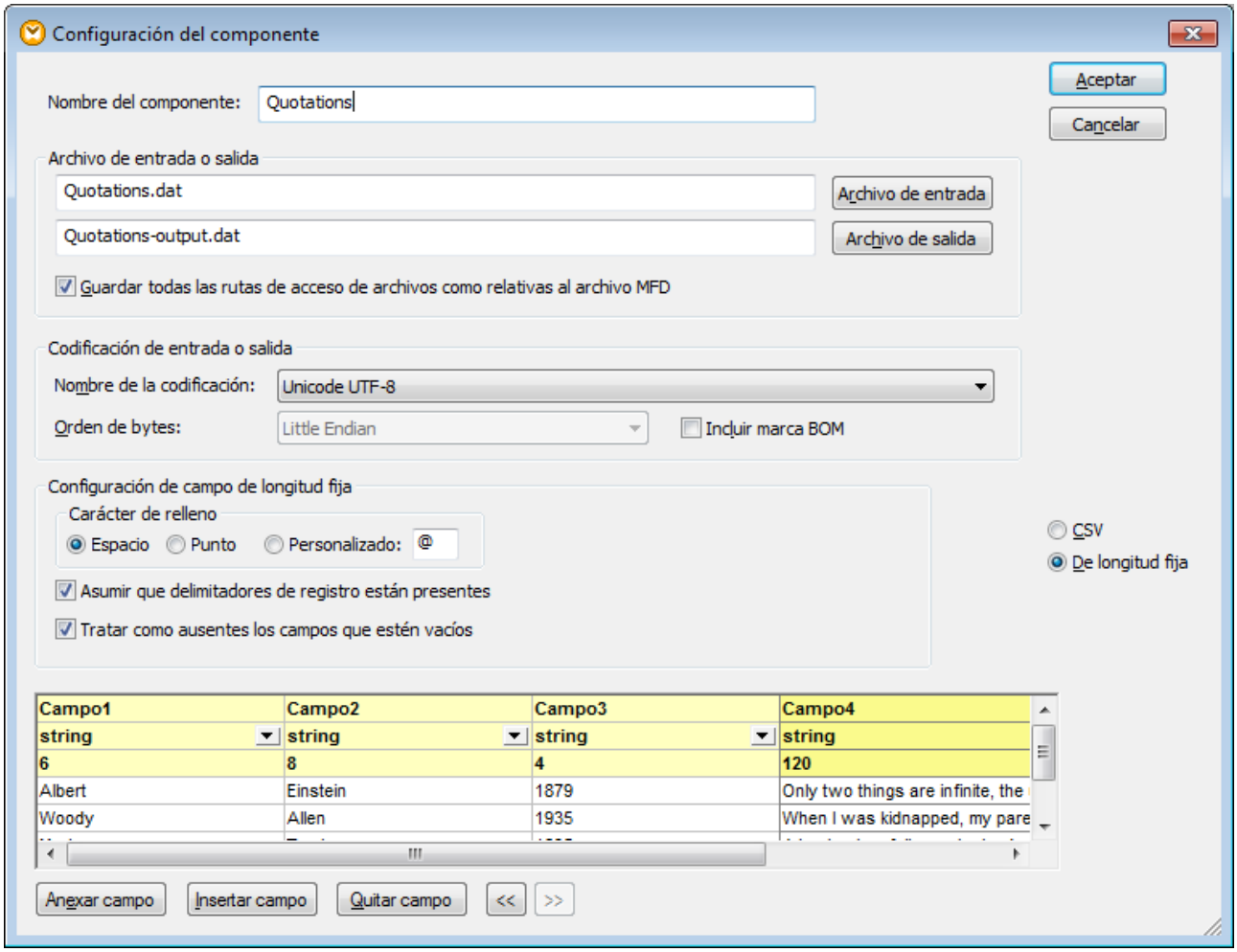



6. Para generar la instrucción SQL (y obtener la vista previa en MapForce) haga clic en el panel *Resultados*. Para ejecutar las instrucciones SQL en la base de datos haga clic en el botón **Ejecutar script SQL** .

## 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.




*Cuadro de diálogo "Configuración del componente" (en modo de longitud fija)*

Estas son las opciones de configuración de los componentes FLF:

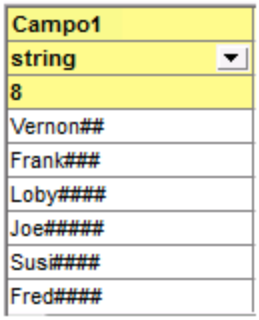
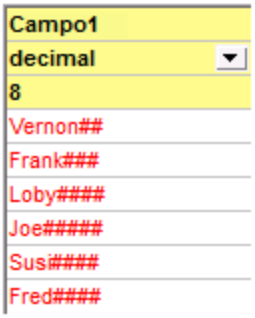
| | |
|---|---|
| *Nombre del componente* | El nombre del componente se genera automáticamente al crearlo, pero se puede volver a cambiar cuando quiera. El nombre del componente puede contener espacios y puntos, pero no barra, barra inversa, dos puntos, comillas dobles, ni espacios antes ni después del nombre.<br><br>Si cambia el nombre del componente, tenga en cuenta que:<br><br>• Si quiere implementar la asignación en FlowForce Server, el nombre del componente debe ser único.<br>• 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. |

 © 2019-2025 Altova GmbH

| | |
|---|---|
| *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.<br><br>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.<br><br>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.<br><br>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* 44). |
| *Codificación de entrada / salida* | Permite configurar estos aspectos del archivo de instancia de salida:<br><br>• Nombre de la codificación<br>• Orden de bytes<br>• El carácter de la marca de orden de byte (marca BOM).<br><br>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.<br><br>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.<br><br>Consulte el párrafo Más información sobre la opción 365 *Asumir que delimitadores de registro están presentes* 365 que aparece más adelante. |

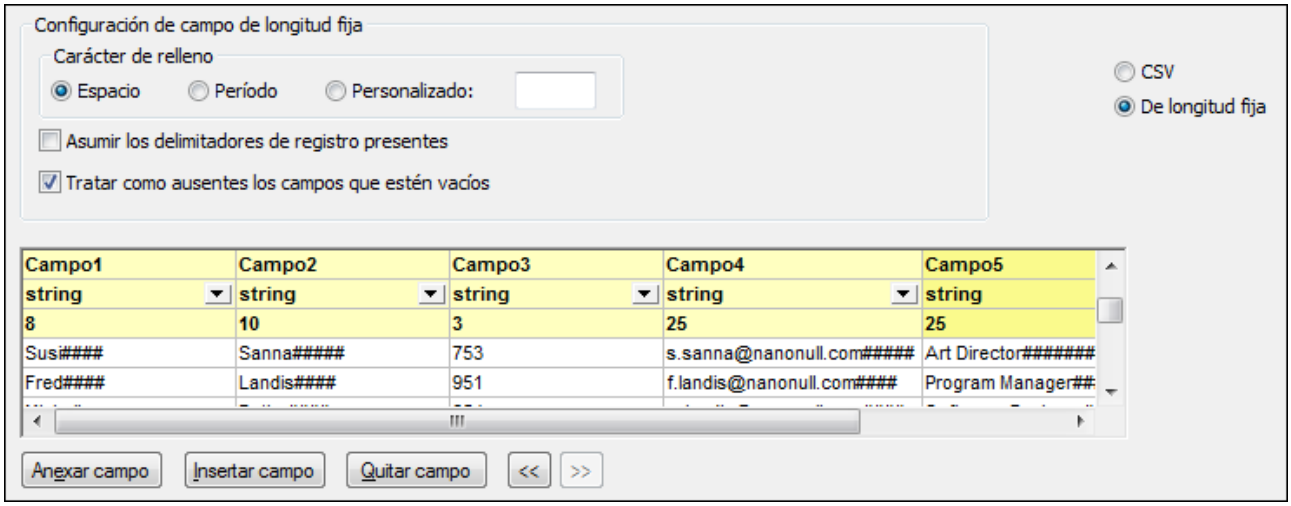
| | |
|---|---|
| *Tratar como ausentes los campos que estén vacíos* | Si marca esta casilla, los campos vacíos del archivo de origen no producirán un elemento correspondiente vacío (elemento o atributo) en el archivo de destino.<br><br>Imaginemos que el destino de la asignación es un archivo XML. Si 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`):<br><br>33 <Person><br>34 <First>General outgassing pollutants</First><br>35 <Last/><br>36 <Title/><br>37 <Email/><br>38 </Person><br><br>Si se activa esta opción, los campos vacíos no se crearán en el destino de la asignación:<br><br>38 <Person><br>39 <First>General outgassing pollutants</First><br>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.<br><br>Si lo necesita, puede crear la estructura del archivo (o cambiarla) con estos botones:<br><br>**Anexar campo** — Crea un campo nuevo después del último regis<br>**Insertar campo** — Crea un campo nuevo inmediatamente antes d registro que está seleccionado.<br>**Quitar campo** — Elimina el campo que está seleccionado.<br>**<<** — Desplaza el campo que está seleccionado hac izquierda.<br>**>>** — Desplaza el campo que está seleccionado hac derecha.<br><br>Para cambiar el nombre de un campo haga clic en el encabezado (por ejemplo, **`Campo1`**) y teclee el nuevo valor. |


© 2019-2025 Altova GmbH

| | |
|---|---|
| |  <br><br>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.<br>  <br><br>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\MapForce2025\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 357.




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.

 © 2019-2025 Altova GmbH

```
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.*

# 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:

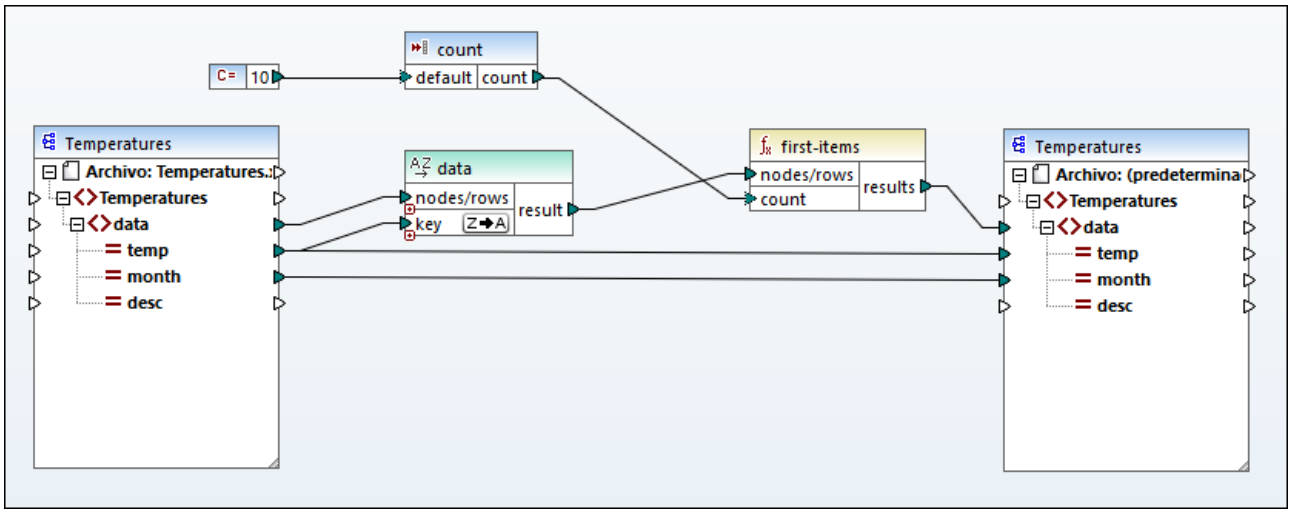

- Entrada simple 369
- Salida simple 380
- Variables
- Combinar componentes 397
- Ordenar componentes 426
- Filtros y condiciones 432
- Asignación de valores 445
- Excepciones 457
- Funciones 461



© 2019-2025 Altova GmbH

# 5.1 Entrada simple

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 599 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 784).

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* 371). Si fuera necesario, también podrá crear valores predeterminados para los parámetros de entrada de la asignación (*véase* *Crear un valor de entrada predeterminado* 373). 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* 485). 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 sólo 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.

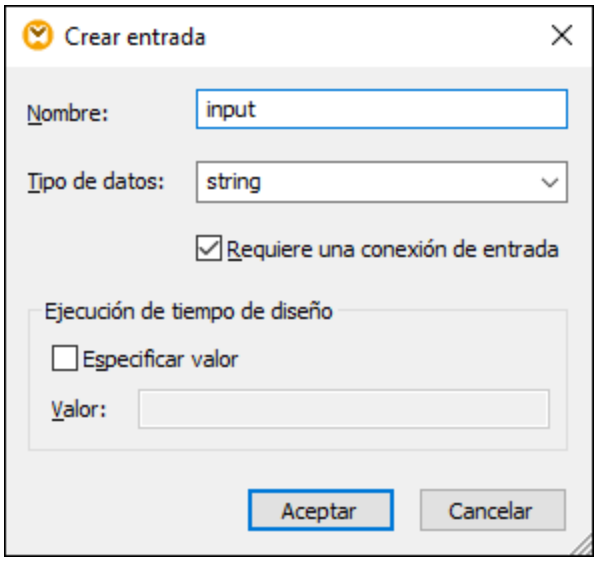
## 5.1.1 Agregar componentes de entrada simples

**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**.

 © 2019-2025 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 371.)

**Nota:** El nombre del parámetro sólo 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**.

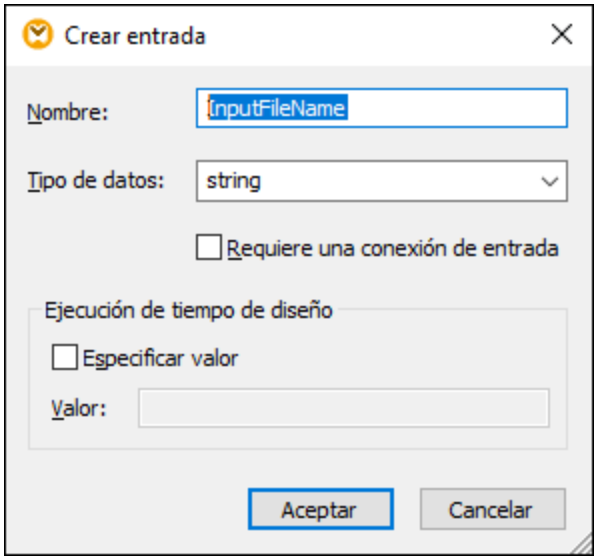
Las demás opciones de configuración se describen en el apartado Configurar componentes de entrada simples 371.

## 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:

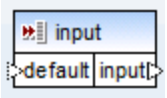
| | |
|---|---|
| *Nombre* | Introduzca un nombre descriptivo para el parámetro de entrada que corresponde a este componente. En tiempo de ejecución el valor que se introdujo en este campo pasa a ser el nombre del parámetro que se pasa a la asignación. Por tanto, no está permitido el uso de espacios 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).<br><br>Desactive esta casilla si desea especificar un valor predeterminado para el parámetro de entrada (*véase* *Crear un valor de entrada predeterminado* 373). |
| *Especificar valor* | Esta casilla sólo 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 sólo 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.<br><br>**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 |

 © 2019-2025 Altova GmbH

| | |
|---|---|
| | la implementación en FlowForce Server. |

## 5.1.3 Crear un valor de entrada predeterminado

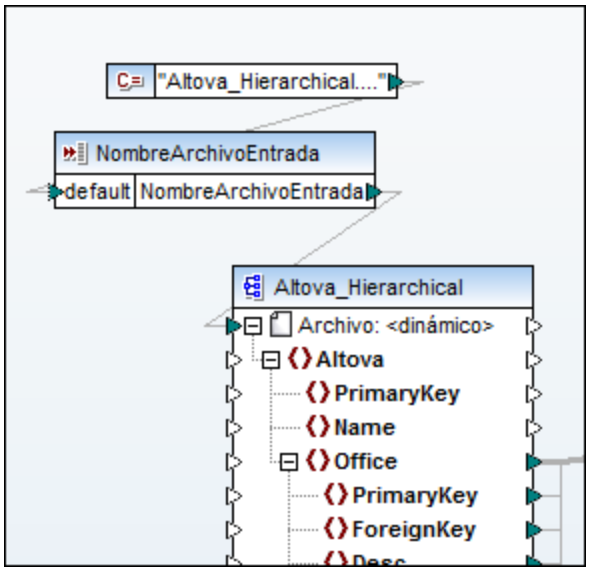
Tras añadir el componente de entrada al área de asignación, debemos fijarnos en el elemento **default** situado en el lado izquierdo del componente.



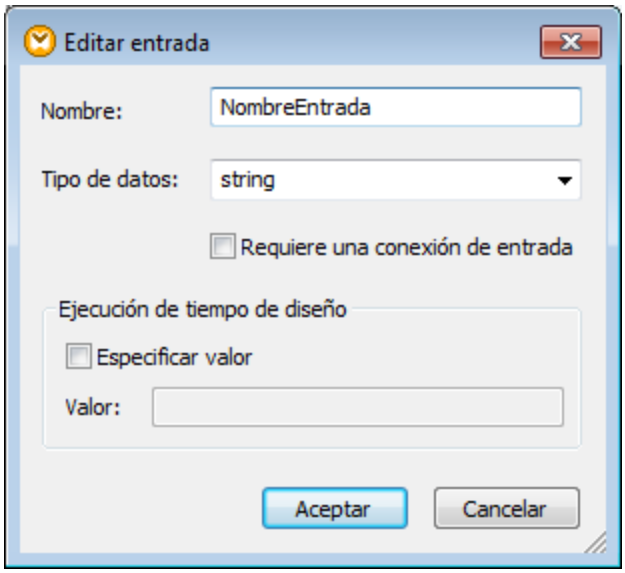

*Componente de entrada simples*

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.

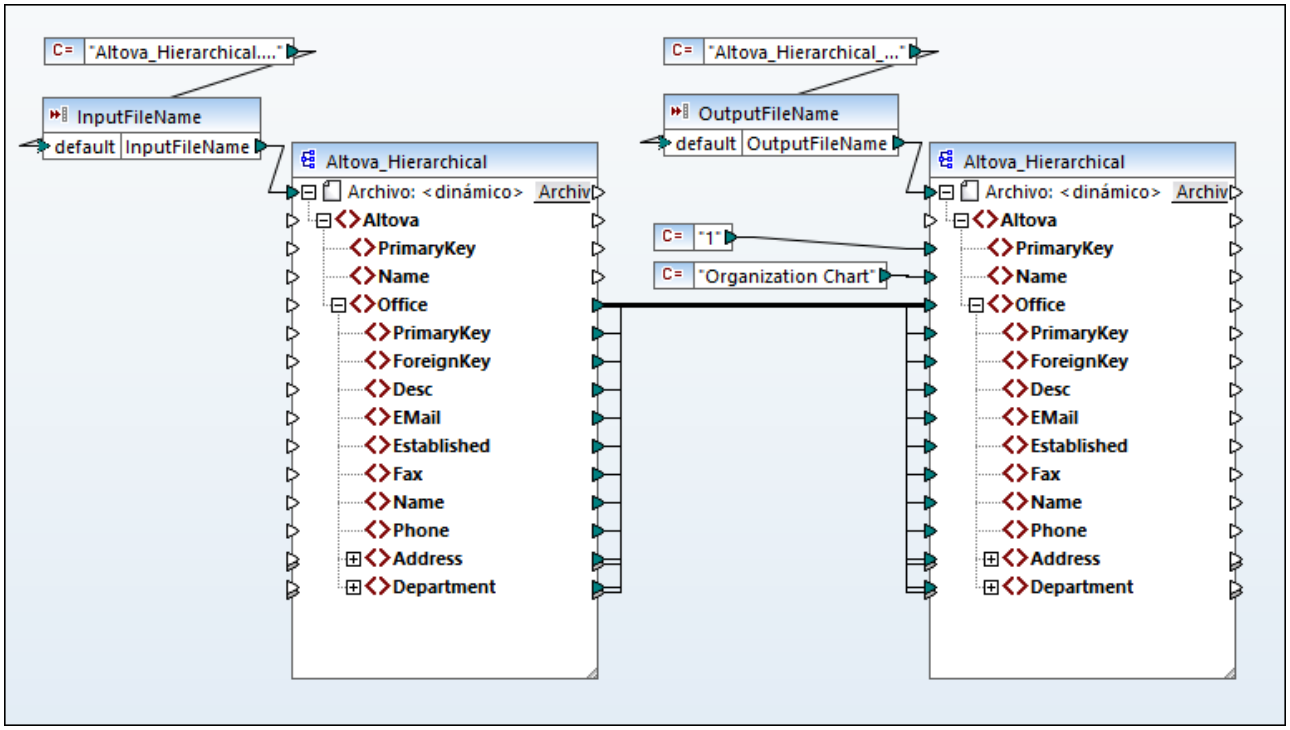
### 5.1.4 Ejemplo: usar nombres de archivo como parámetros de asignación

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\MapForce2025\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**.

 © 2019-2025 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 371. 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 2.0 376, con ayuda de RaptorXML Server
- Built-in (archivo de ejecución de MapForce Server) 376, con ayuda de MapForce Server
- Java 377
- C# 378
- C++ 378

## 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* 831). 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`.

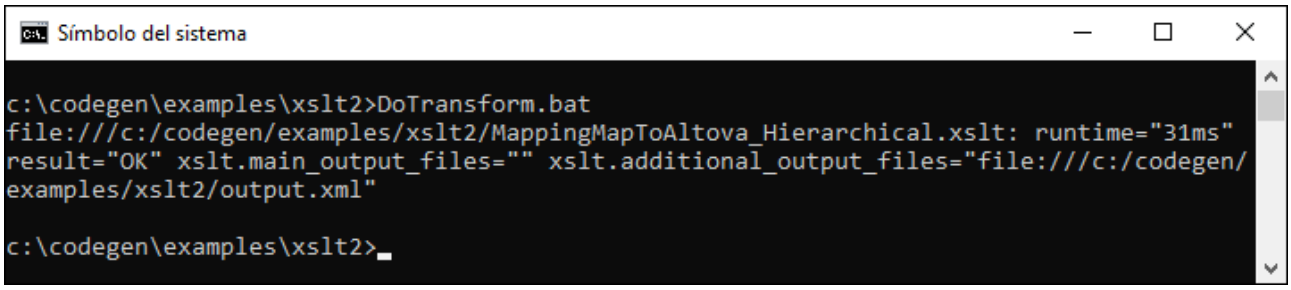
2. Copie el archivo `Altova_Hierarchical.xml` del directorio `<Documentos>\Altova\MapForce2025\MapForceExamples\` al directorio donde generó el código XSLT 2.0 (en este ejemplo usamos el directorio de salida `c:\codegen\examples\xslt2\`). Este archivo hará las veces de parámetro personalizado.

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* 839).

 © 2019-2025 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\MapForce2025\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\MapForceServer2025\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/es/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).
3. Copie el archivo `Altova_Hierarchical.xml` del directorio `<Documentos>\Altova\MapForce2025\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\MapForce2025\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\MapForce2025\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\MapForce2025\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\MapForce2025\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:


© 2019-2025 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 Salida simple

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* 382). 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* 485). 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 sólo 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* 839), 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**: |

 © 2019-2025 Altova GmbH

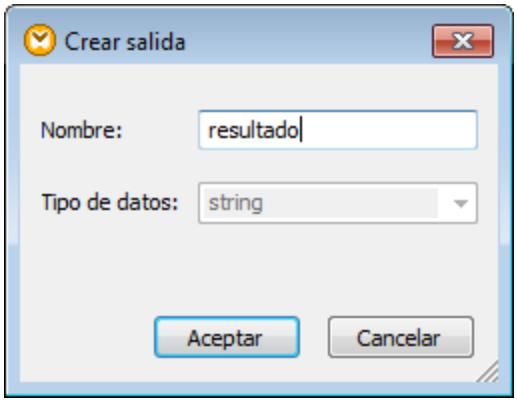
| | |
|---|---|
| | `MapForceServer.exe run MiAsignación.mfx >resultado.txt 2>log.txt` |
| XSLT 1.0, XSLT 2.0, XSLT 3.0 | En los archivos XSLT que se generan, el componente de salida simple definido en la asignación se convierte en el resultado de la transformación XSLT.<br><br>Si usa un servidor RaptorXML Server, puede ordenar al servidor que escriba el resultado de la asignación en el archivo que se pasa como valor del parámetro `--output`.<br><br>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).<br><br>`RaptorXML xslt --xslt-version=2 --input="MappingMapToResult1.xslt" --output="Resultado.txt" %* "MappingMapToResult1.xslt"`<br><br>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.<br><br>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 461.

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:

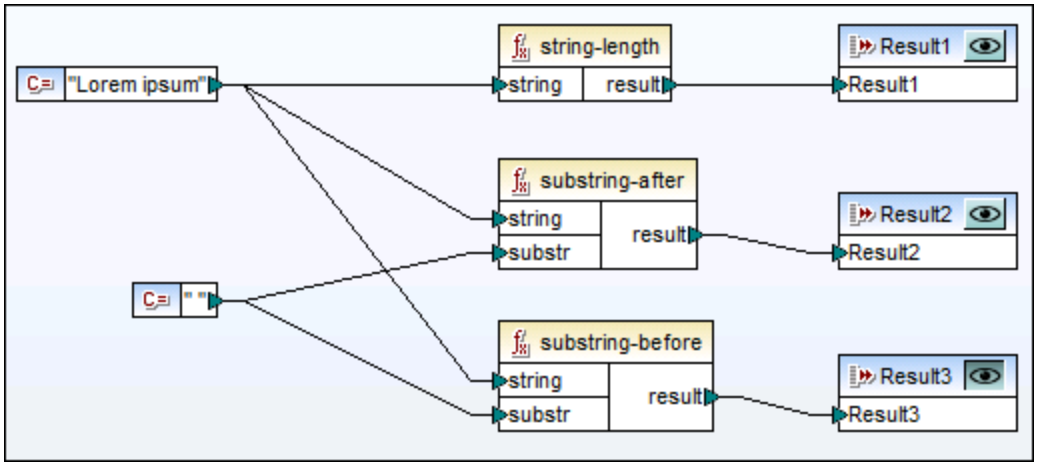
| | |
|---|---|
| `string-length` 629 | Devuelve el número de caracteres de la cadena que se dio como argumento. Por ejemplo, si pasamos a esta función el valor "Lorem ipsum", se obtiene el resultado "11" porque es el número de caracteres que toma el texto "Lorem ipsum". |
| `substring-after` 630 | Devuelve 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` 631 | 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.

 © 2019-2025 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

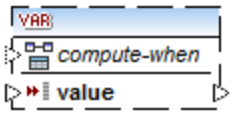
Una variable es un tipo especial de componente que sirve para almacenar el resultado intermedio de una asignación de datos para procesarlo más tarde. Las variables pueden ser de tipo simple (por ejemplo, string, integer, boolean, etc.) y de tipo complejo (una estructura jerárquica). Puede ver ejemplos de ambos tipos en los apartados siguientes.

Una de las cosas más importantes de las variables es que son secuencias y que pueden utilizarse para crear secuencias. El término *secuencia* significa una lista de cero o más elementos. Esto hace posible que una variable procese varios elementos durante el tiempo de vida de la asignación. Para más información, consulte también Reglas y estrategias de asignación 80. Sin embargo, también es posible asignar un valor a una variable una vez y mantener este valor igual para el resto de la asignación. Para más detalles, consulte Cambiar el contexto y ámbito de las variables 390.

Para más detalles sobre cómo agregar una variable a una asignación, consulte Agregar variables 386.

## 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 una estructura jerárquica. Las estructuras en las que puede basarse una variable compleja se resumen en la siguiente lista.

*MapForce Basic Edition:*

- Estructura de XML Schema

*MapForce Professional Edition:*

- Estructura de XML Schema
- Estructura de BD

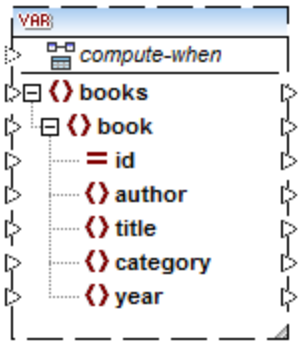
*MapForce Enterprise Edition:*

- Estructura de XML Schema
- Estructura de BD
- Estructura EDI
- Estructura FlexText
- Estructura de JSON Schema

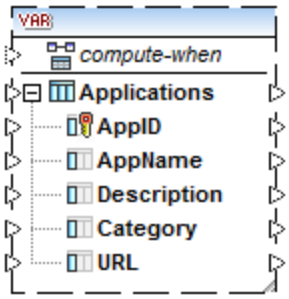
*Ejemplo 1: variable basada en un esquema XML*

Puede crear una variable compleja suministrando un esquema XML que defina la estructura de la variable (*imagen siguiente*). Si el esquema define algún elemento de forma global, puede elegir cuál debe convertirse en el nodo raíz de la estructura variable. Tenga en cuenta que una variable no tiene asociado un archivo XML de instancia. Los datos de la variable se calculan en tiempo de ejecución de la asignación.

 © 2019-2025 Altova GmbH




*Ejemplo 2: variable basada en una base de datos (MapForce Professional y Enterprise Editions)*
Si elige una estructura de base de datos para su variable (*imagen siguiente*), puede elegir una tabla específica de la base de datos como elemento raíz de la estructura de la variable. MapForce le permite crear variables basadas en BD con una estructura de tablas relacionadas.



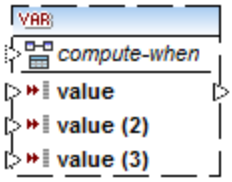

La estructura de tablas relacionadas representa una estructura en memoria que no tiene conexión con la BD en tiempo de ejecución. La variable solo describe la estructura, pero no hace nada con los datos. Esto supone lo siguiente:

- No hay validación basada en el tipo de datos
- No hay posibilidad de utilizar el filtro SQL WHERE
- No hay acciones de tabla en parámetros ni variables
- No hay control automático de las claves foráneas
- No hay comprobaciones de clave primaria/foránea

*Compute-when*
En los dos ejemplos anteriores, cada variable tiene un elemento llamado `compute-when`. La conexión de este elemento es opcional: esto le permite controlar cómo debe calcularse el valor de la variable en la asignación. Para más detalles, consulte Cambiar el contexto y ámbito de las variables 390.

## Variables con entradas duplicadas

Cuando sea necesario, los elementos de una estructura variable pueden duplicarse para aceptar datos de más de una conexión de origen. Esto es similar a la duplicación de entradas 43 en componentes estándar. En la imagen siguiente puede ver un ejemplo de variable simple con entradas duplicadas.




### Asignaciones en cadena frente a variables

Las variables pueden equipararse a los componentes intermedios de una asignación en cadena 119. 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 encadenadas | 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. Para más detalles, consulte Cambiar el contexto y ámbito de las variables 390. |
| 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:** las variables no son compatibles si el lenguaje de transformación de la asignación está configurado como XSLT 1.0.

## 5.3.1 Agregar variables

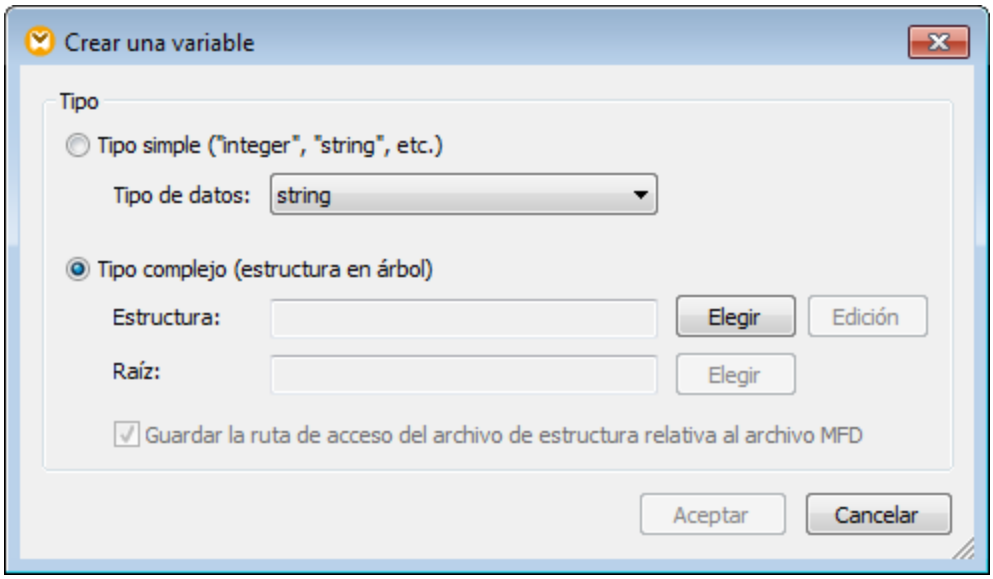
En este apartado explicamos cómo agregar variables a una asignación. La primera opción es agregar variables con comandos de menú o de la barra de herramientas. La segunda opción permite agregar variables desde el menú contextual.

### 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. Para ello siga estos pasos:

 © 2019-2025 Altova GmbH

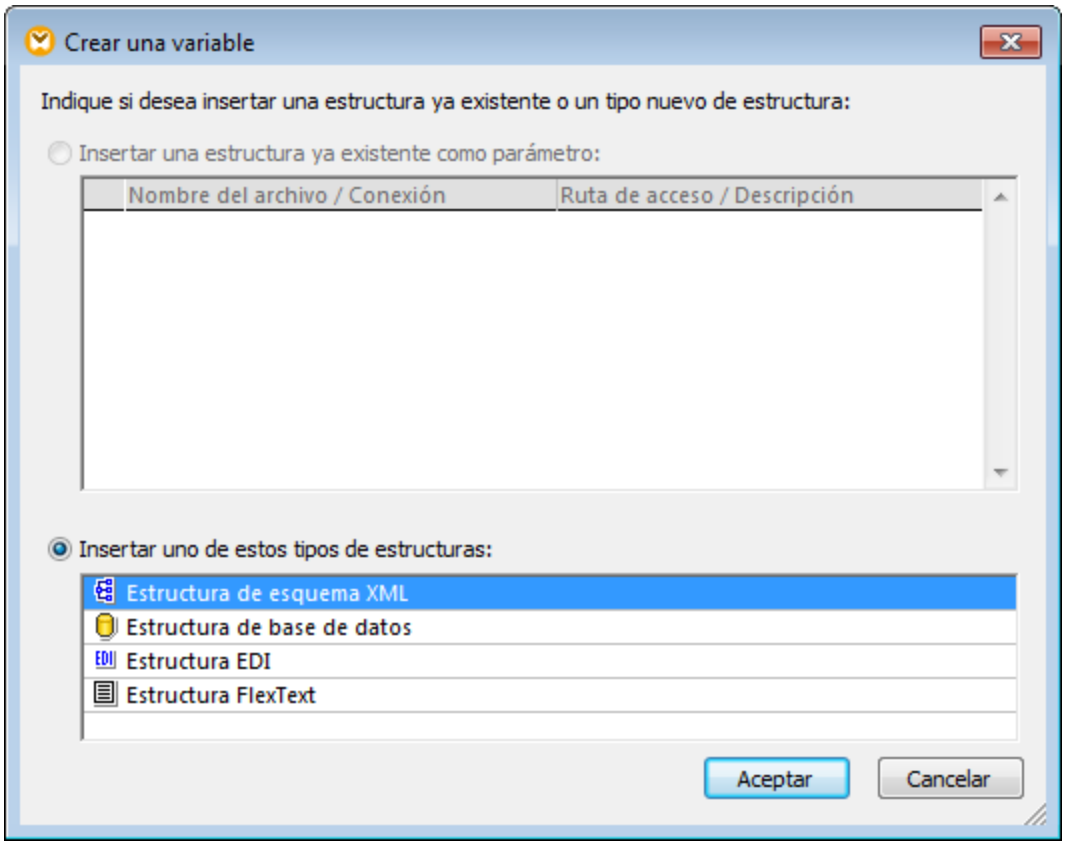
1. En el menú **Insertar** haga clic en el comando **Variable**. También puede hacer clic en el botón (**Variable**) de la barra de herramientas.




2. Seleccione el tipo de variable que quiere insertar (de tipo simple o complejo).

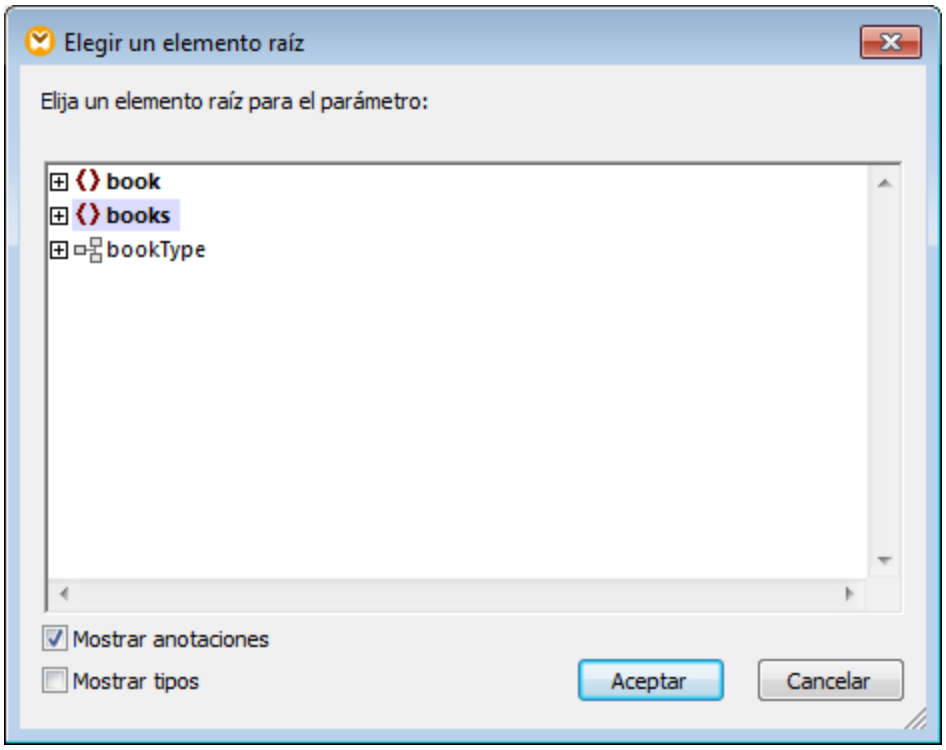
Si selecciona **Tipo complejo** debe seguir también estos pasos:

3. Haga clic en **Elegir** para seleccionar el origen del que se debe tomar la estructura de la variable. 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.




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*).


© 2019-2025 Altova GmbH

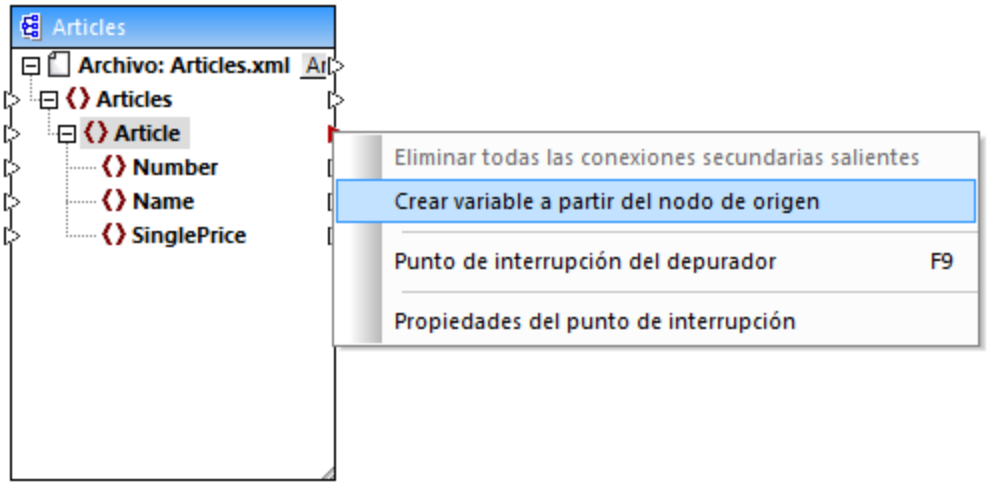



## Opción 2: desde el menú contextual

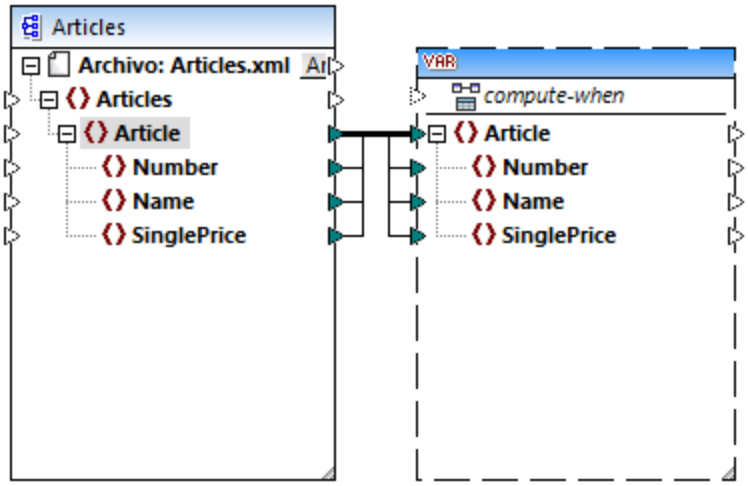
La segunda opción permite crear variables desde el menú contextual. A continuación se enumeran todas las opciones posibles.

*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 57 (*imagen siguiente*).

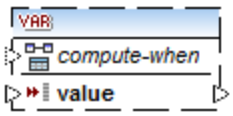



*Variable a partir de un nodo de destino*
Para crear una variable a partir de un nodo de destino, haga clic con el botón derecho en el conector de entrada de un componente de destino y seleccione **Crear variable a partir del nodo de destino**. Se crea una variable compleja con el mismo esquema del componente 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 Contexto y ámbito de las variables

Todas las variables tienen un elemento de entrada `compute-when` (*imagen siguiente*) que permite controlar el alcance de la variable. Esto significa que puede controlar cuándo y con qué frecuencia se computa 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.




Los siguientes términos son relevantes para la discusión del alcance y el contexto de las variables: *subestructura* y *valor de la variable*. Una subestructura es un conjunto formado por un elemento/nodo de un

 © 2019-2025 Altova GmbH

componente de destino y todos sus descendientes: por ejemplo, un elemento `<Persona>` con sus elementos secundarios `<Nombre>` y `<Apellido>`.

El valor de la variable son los datos disponibles en el lado de salida 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 de entrada de la variable y de con qué elementos principales del componente de origen/destino esté conectado.

## 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 489 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.

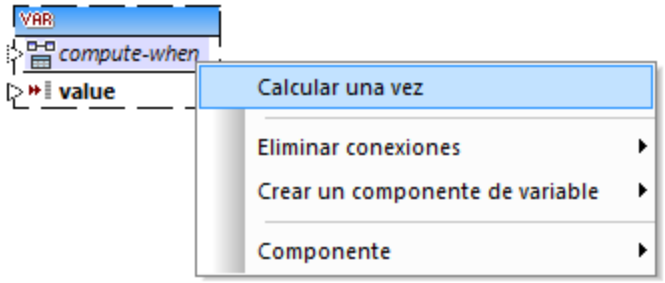
## Compute-when está conectado

Si se conecta un conector de salida de un componente de origen con el elemento `compute-when`, entonces 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 actúa como si fuera un elemento secundario del elemento conectado a `compute-when`. 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. Esto está relacionado con la regla general que rige las conexiones en MapForce: por cada elemento de origen, se crea uno de destino. En este caso, `compute-when` indica a MapForce que calcule el valor de la variable para cada elemento de origen. Para más información, consulte Reglas y estrategias de asignación 80.

## Compute-once

Si es necesario, puede optar por calcular el valor de la variable *una vez antes de cada uno de los componentes de destino*, haciendo que la variable sea esencialmente una constante global para el resto de la asignación. Para ello, haga clic con el botón derecho en el elemento `compute-when` y seleccione **Calcular una vez** en el menú contextual:

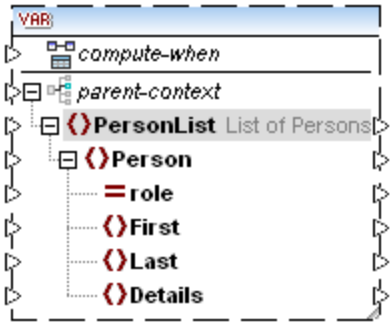



Cuando se cambia el ámbito de una variable por `compute-when=once`, el conector de entrada se elimina del elemento `compute-when` (porque dicha variable solamente se evaluará una vez). En funciones definidas por el usuario, la variable `compute-when=once` se evalúa cada vez que se llama a la función y antes de que el resultado de la función se evalúe.

## Parent-context

**parent-context** es un argumento opcional de algunas funciones de agregado de la biblioteca de funciones **core** de MapForce (como, por ejemplo, de las funciones **`min`**, **`max`**, **`avg`** y **`count`**). En un componente de origen que tiene varias secuencias jerárquicas, el contexto primario determina en qué conjunto de nodos debe operar la función.

Agregar un elemento `parent-context` puede ser necesario, por ejemplo, si su asignación utiliza varios filtros y necesita un nodo primario adicional sobre el que iterar. Para más detalles, consulte Ejemplo: cambiar el contexto primario 85. Para agregar un parent-context a una variable, haga clic con el botón derecho en el nodo raíz (en este ejemplo, `PersonList`) y seleccione **Agregar contexto primario** en el menú contextual. Como resultado se añade un nodo nuevo llamado `parent-context` a la jerarquía de la variable.



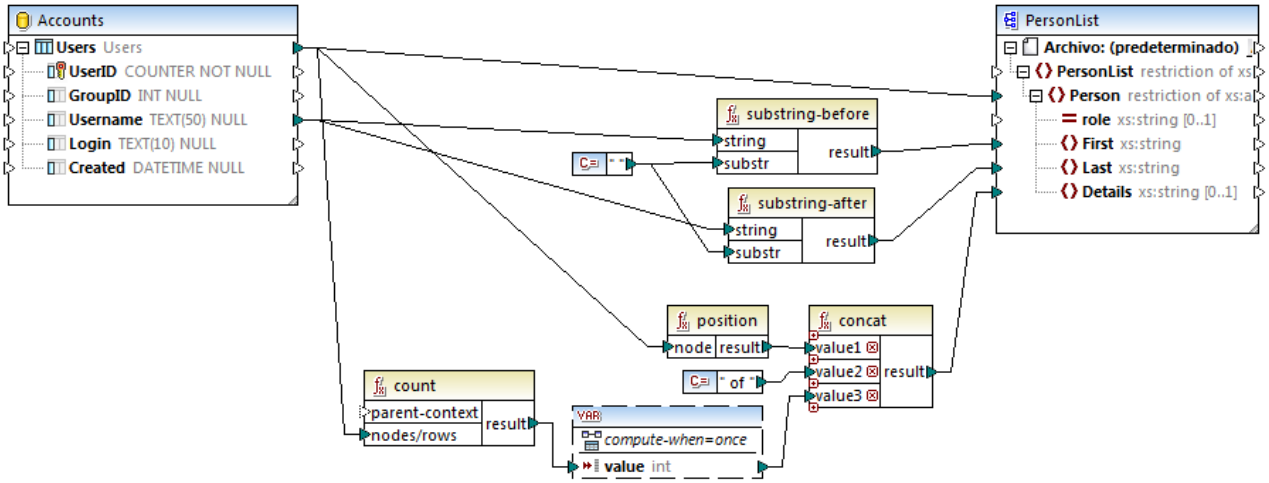

El contexto primario agrega un nodo primario virtual a la jerarquía dentro del componente. Esto le permite iterar sobre un nodo adicional en el mismo componente de origen o en otro diferente.

 © 2019-2025 Altova GmbH

## 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\MapForce2025\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.
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.

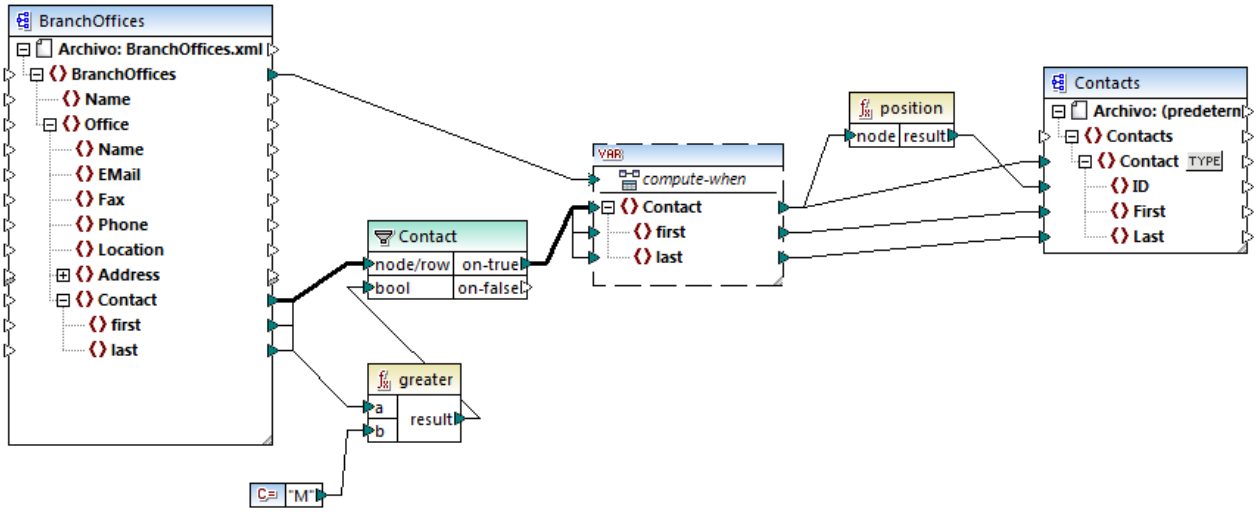
El valor del campo "Username" se pasa a las funciones **`substring-before`** 631 y **`substring-after`** 630 , que se encargan de extraer el texto situado antes y después del carácter de espaciado (" ") respectivamente. Así se consigue el segundo objetivo del diseño.

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* 390).

## 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\MapForce2025\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* 432)
- Una variable compleja (*véase* *Agregar variables* 386)
- Las funciones **`greater`** 579 y **`position`** 616 (*véase* *Agregar una función* 462)
- 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`** 579 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

 © 2019-2025 Altova GmbH

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.

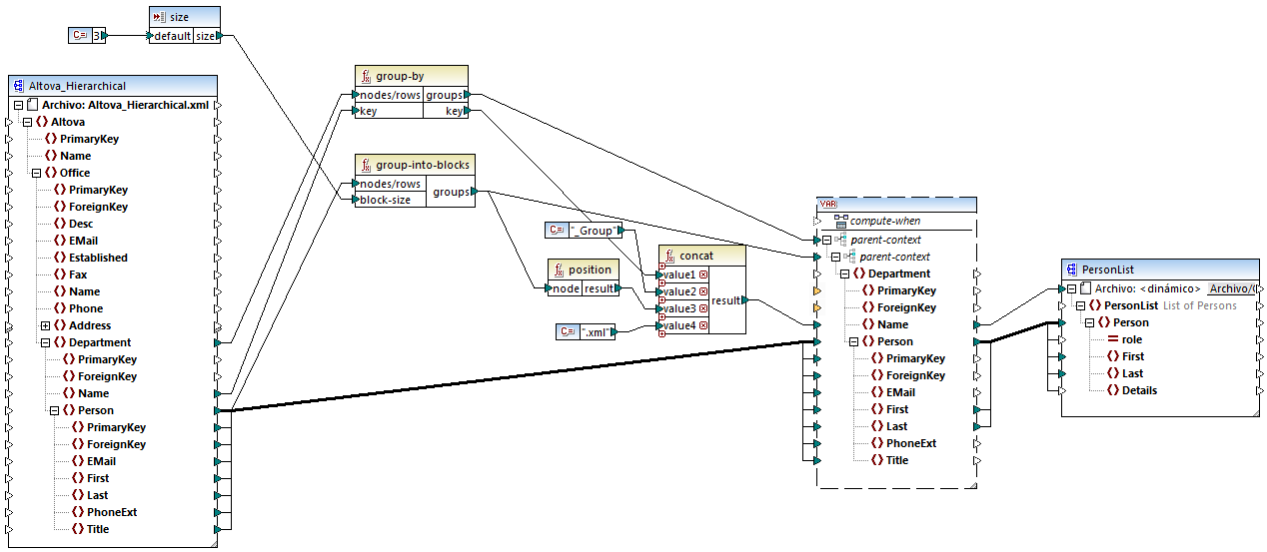
La función **`position`** 616 devuelve (por cada iteración de la variable) el número de la secuencia actual. Sólo ocho contactos cumplen la condición. Por tanto, si consultamos la vista previa del resultado de la asignación, veremos que se escribieron los identificadores 1 - 8 en el elemento `ID` del 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.

## 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\MapForce2025\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* 390).

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`** 603 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`** 608 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* 369).

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`** 626, 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 784.

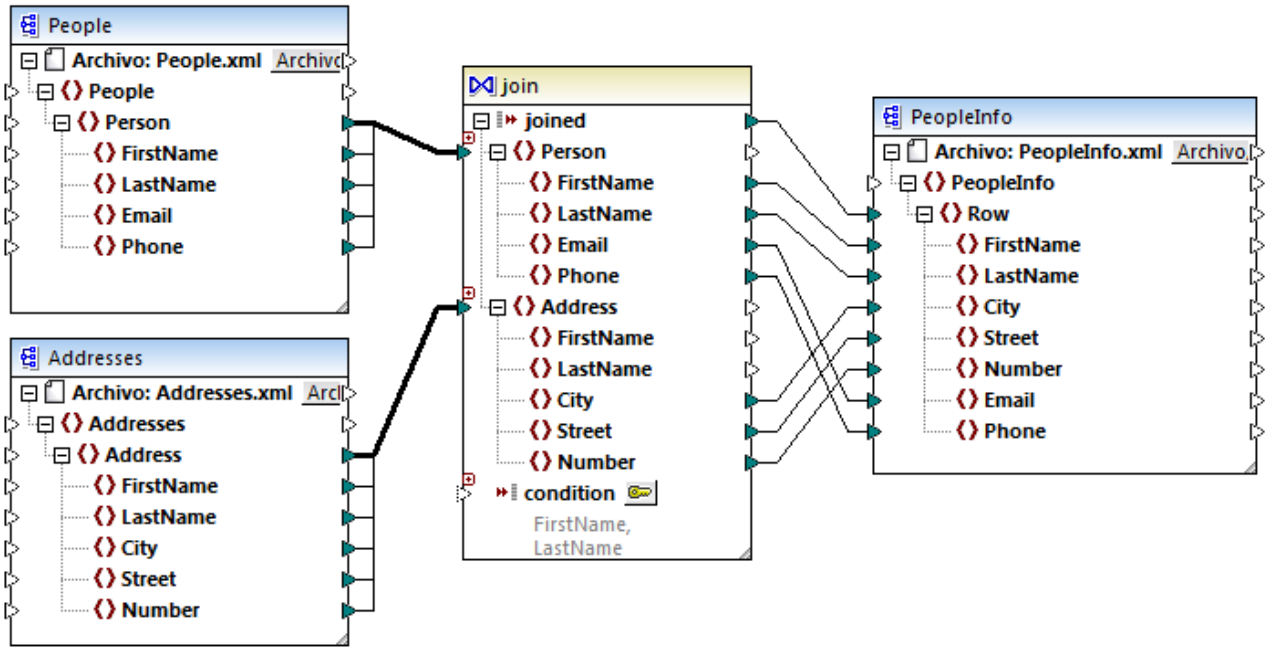
 © 2019-2025 Altova GmbH

## 5.4 Componentes de combinación

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 va acompañado de una asignación de muestra y se explica con más detalle en Ejemplo: combinar estructuras XML 403.



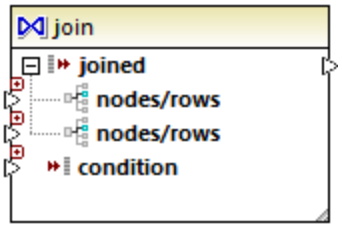

*JoinPeopleInfo.mfd*

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, MapForce crea automáticamente estas conexiones cuando el contexto es relevante (para más información, consulte Conexiones de copia total). 57

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 408 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* 402).




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* 399).

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* 66).
- 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* 410).
- 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.

 © 2019-2025 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 y Filtros y condiciones 432). 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 408).

### 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. Para más información consulte Ejemplo: cambiar el contexto primario 85.

**parent-context** es un argumento opcional de algunas funciones de agregado de la biblioteca de funciones **core** de MapForce (como, por ejemplo, de las funciones **`min`**, **`max`**, **`avg`** y **`count`**). En un componente de origen que tiene varias secuencias jerárquicas, el contexto primario determina en qué conjunto de nodos debe operar la función.

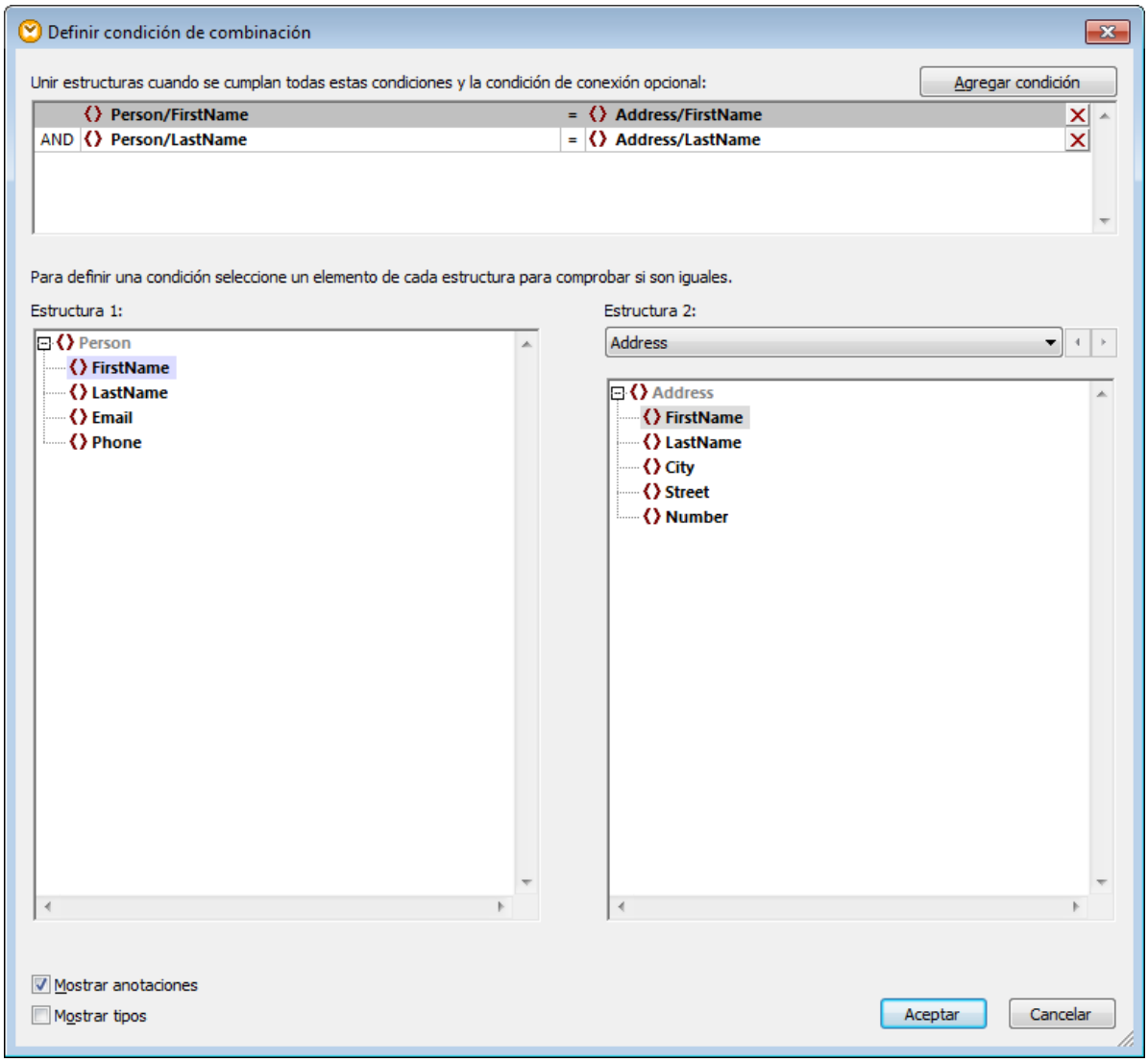
## 5.4.1 Agregar condiciones de combinación

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 413 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\MapForce2025\MapForceExamples\Tutorial**. Esta asignación se describe más detalladamente en el apartado Ejemplo: combinar estructuras XML 403.
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.

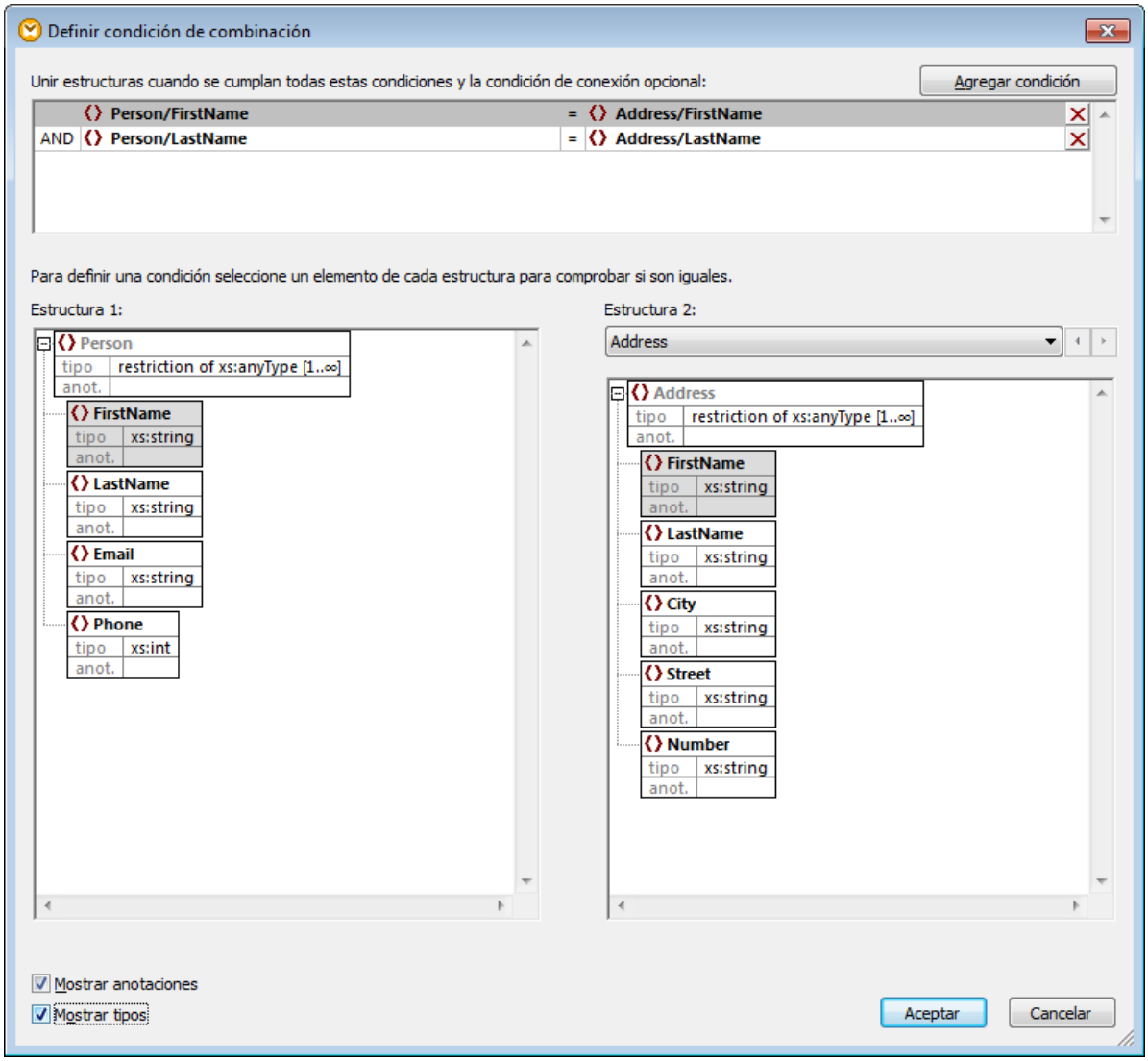
**Notas:**

- 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

 © 2019-2025 Altova GmbH

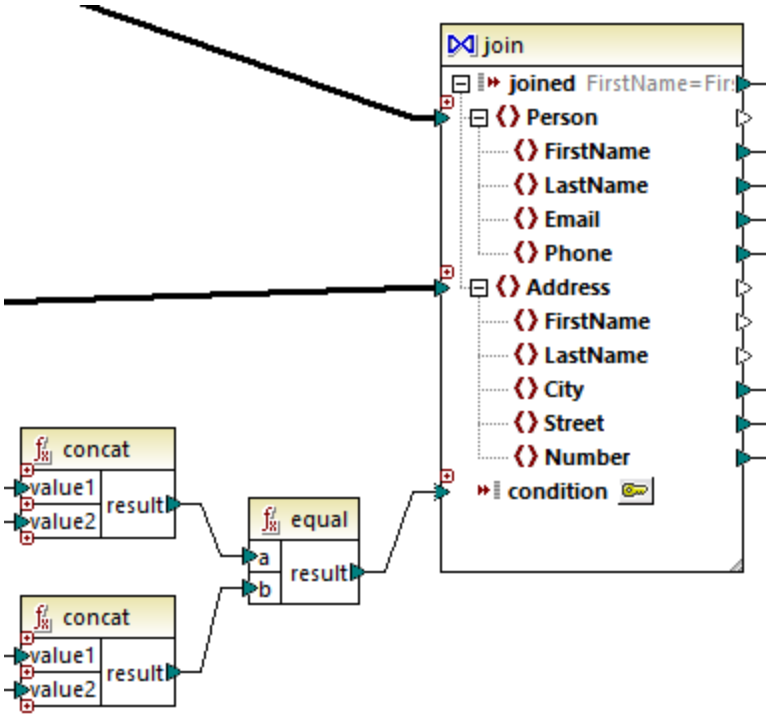
estructuras que aparecen antes de ella en el componente de combinación. Así podrá definir condiciones de combinación entre cualquier estructura. Para ver un ejemplo consulte Ejemplo: crear un informe CSV a partir de varias tablas 421.

- 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.



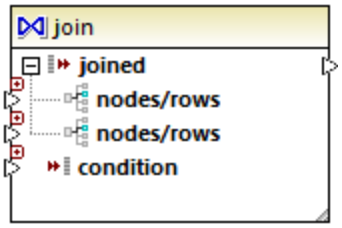

**Nota:** si no se define ninguna condición desde las propiedades del componente (método 1), debe conectarse el nodo `condition` del componente de combinación (método 2).

### 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* 410).

## 5.4.2 Combinar tres o más estructuras

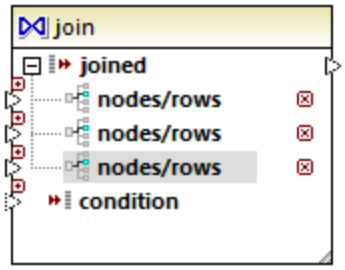
Si se añade un componente de combinación a la asignación con el comando de menú Insertar **| Combinación**, el componente acepta dos estructuras por defecto (es decir, el componente sólo contiene dos entradas `nodes/rows`).




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**

 © 2019-2025 Altova GmbH

**entrada** ⊠. Tenga en cuenta que un componente de combinación necesita dos estructuras como mínimo (y, por tanto, el botón ⊠ sólo 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* 399). 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 421.

## 5.4.3 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\MapForce2025\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.

 © 2019-2025 Altova GmbH

## Paso nº1: agregar los archivos XML de origen en la asignación

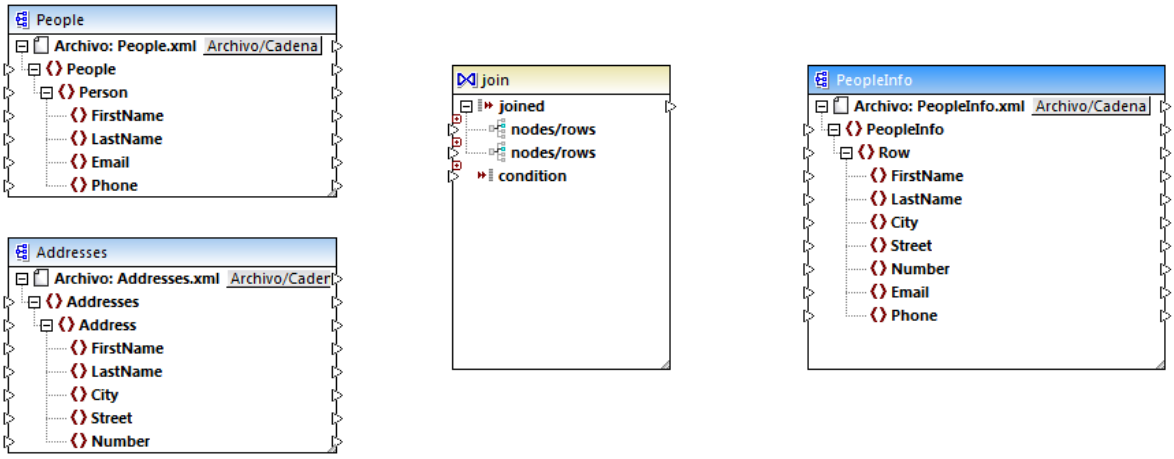
1. En el menú **Insertar** haga clic en el comando **Archivo o esquema XML** y navegue hasta este archivo de origen: **<Documentos>\Altova\MapForce2025\MapForceExamples\Tutorial\People.xml**.
2. Repita el paso anterior e inserte este archivo **<Documentos>\Altova\MapForce2025\MapForceExamples\Tutorial\Addresses.xml**.

## 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\MapForce2025\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.

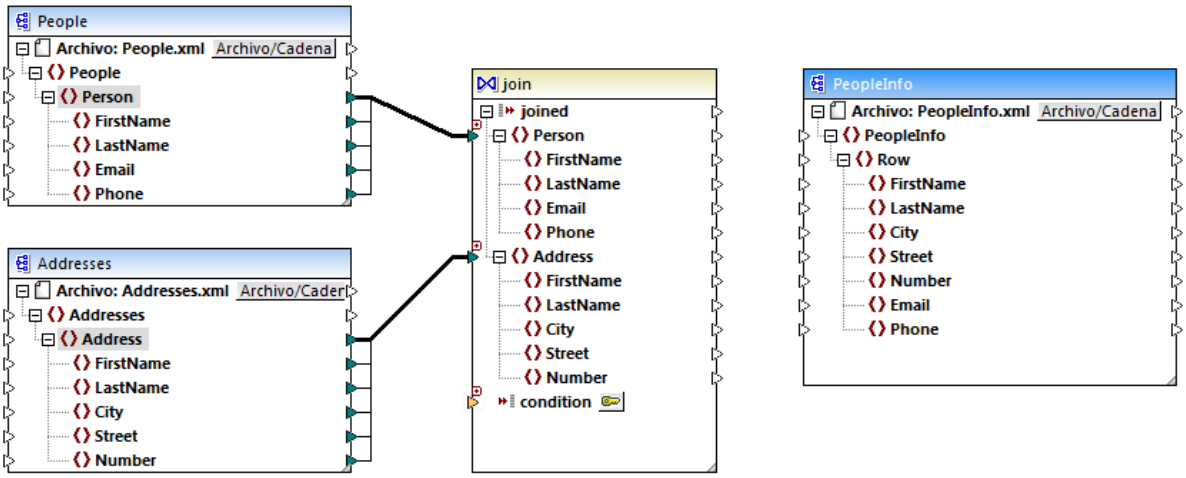
## Paso nº3: agregar el componente de combinación

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):



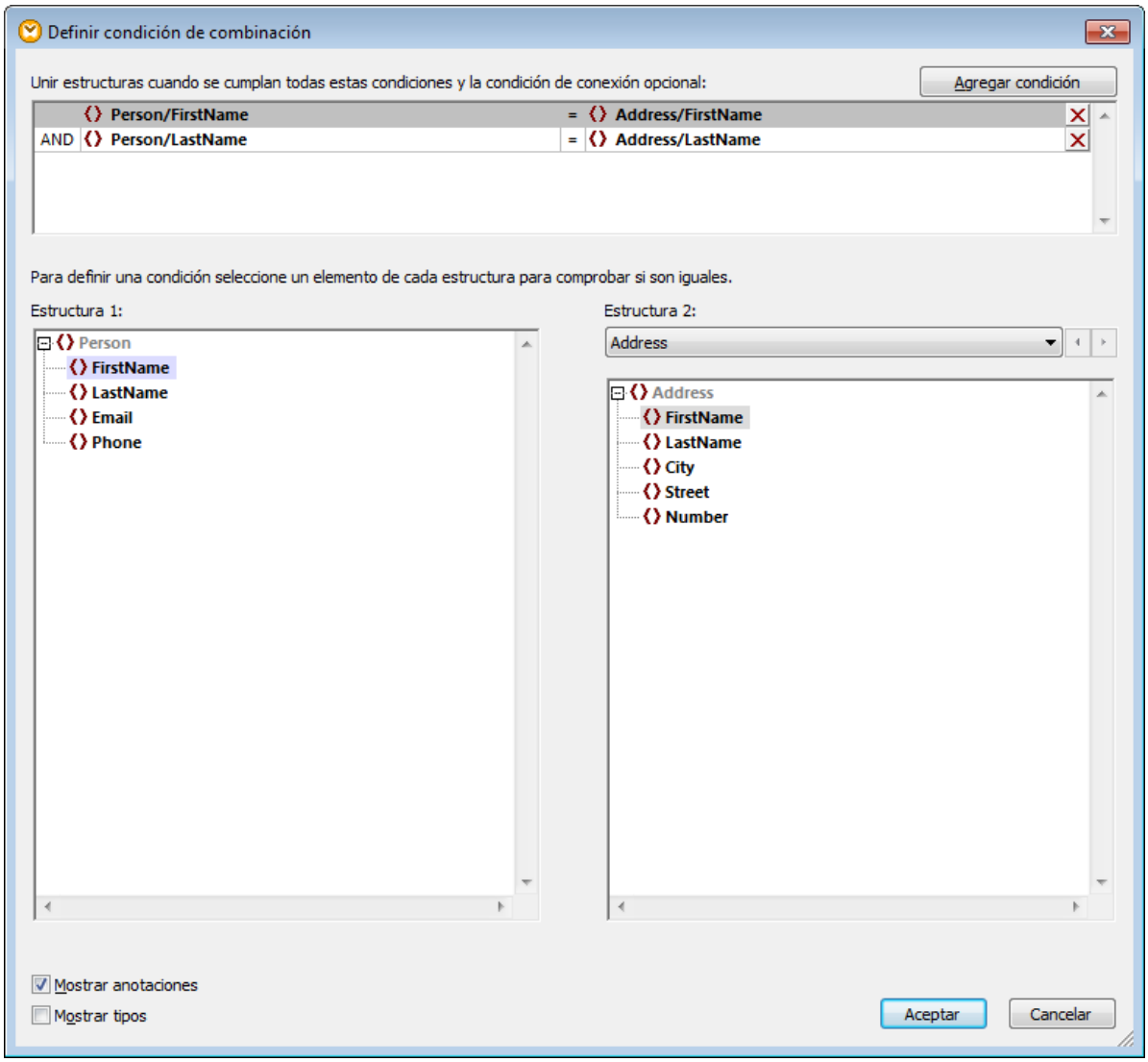

Fíjese en la estructura del componente de combinación. Tiene dos entradas `nodes/rows` para que podamos conectarlo a las dos estructuras que deseamos comparar (las estructuras `Person` y `Address` en este caso).

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 sólo 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.


© 2019-2025 Altova GmbH




En algunas asignaciones puede ser suficiente una condición formada por una comparación. Sin embargo, en este ejemplo es importante crear dos comparaciones:

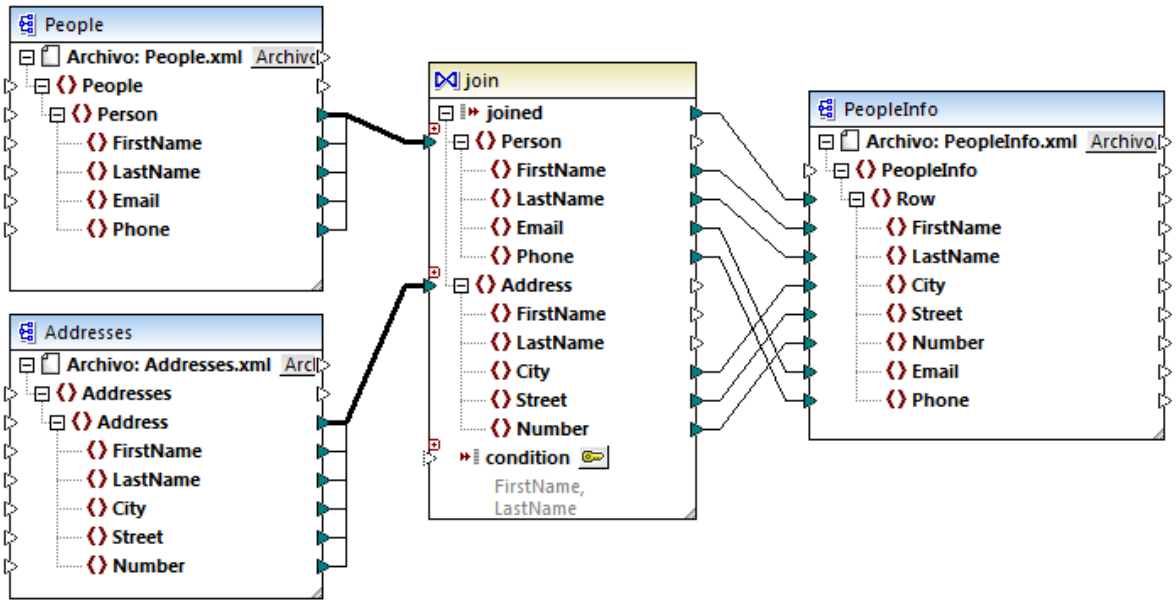
1) `FirstName` en Estructura 1 = `FirstName` en Estructura 2
2) `LastName` en Estructura 1 = `LastName` en Estructura 2.

Cuando se definen varias condiciones, todas ellas deben cumplirse. De lo contrario no tiene lugar la combinación. Por tanto, en este ejemplo, la combinación sólo tendrá lugar si las dos comparaciones dan como resultado true. Si, por el contrario, sólo 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.

## 5.4.4 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 sólo 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 403. 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.

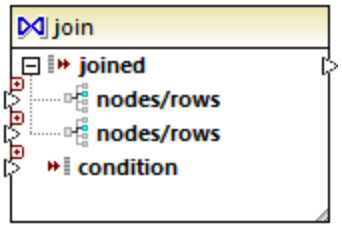
 © 2019-2025 Altova GmbH

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 410.

**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* 259). 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* 402).




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* 399).

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* 66).
- 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* 410).
- 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.

## 5.4.4.1 Combinaciones en modo SQL

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 aparece el botón **Modo SQL** SQL. Cuando el modo SQL está activado, la operación de combinación la realiza la base de datos de la que la asignación lee los datos. En otras palabras, MapForce enviará internamente a la base de datos una consulta con la sintaxis SQL adecuada para seleccionar y combinar los datos de todas las tablas que participan en la combinación. Es importante destacar que no necesita escribir ningún SQL. La consulta necesaria se produce en función de cómo haya diseñado visualmente el componente de combinación en la asignación, como verá en los ejemplos siguientes.

Para poder trabajar en modo SQL, deben cumplirse las siguientes condiciones:

1. Los dos objetos (tablas o vistas) que se van a combinar deben proceder de la misma base de datos.
2. Los dos objetos que se van a combinar deben proceder del mismo componente MapForce. (Tenga en cuenta que puede agregar/eliminar rápidamente objetos de base de datos en un componente del siguiente modo: clic con el botón derecho en el componente de base de datos y seleccione **Agregar/Eliminar/Editar objetos de 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 (ver Agregar condiciones de combinación 399).

**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 413 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 utiliza en un componente de destino, ninguna de las tablas combinadas puede estar conectada directa o indirectamente a ningún nodo primario de destino. Para más información sobre cómo se ejecuta una asignación de datos, consulte Reglas y estrategias de asignación de datos 80.

Puede ver o controlar el modo SQL con el botón **SQL** ( SQL ) situado en la esquina superior derecha del componente de combinación, como se indica a continuación:

SQL El modo SQL está deshabilitado (MapForce o, en su caso, MapForce Server se encargará de ejecutar la combinación).

SQL El modo SQL está habilitado (la base de datos se encargará de ejecutar la combinación).

 © 2019-2025 Altova GmbH

Si falta el botón SQL, significa que el modo SQL no tiene sentido o no es compatible con los datos que se están combinando.

En algunos casos, el modo SQL debe deshabilitarse explícitamente ( SQL ), por ejemplo:

- Cuando su asignación de datos requiera condiciones de combinación fuera de las propiedades del componente de combinación (es decir, condiciones definidas en la asignación y conectadas al elemento `condición` del combinación).
- Cuando desee unir tablas de diferentes bases de datos. En este caso, deberá usar una combinación estándar (no SQL).

## Cambiar el modo de combinación

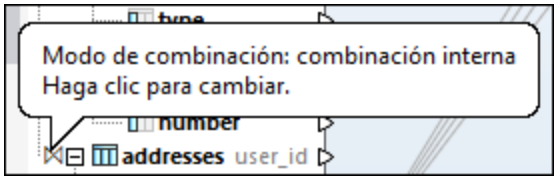
Cuando el componente de combinación está en **Modo SQL** SQL, hay varias maneras de combinar varias tablas o vistas de BD en una:

- COMBINACIÓN INTERNA: el componente de combinación solo devuelve los registros que cumplan la condición en ambos conjuntos de entrada.
- COMBINACIÓN EXTERIOR IZQUIERDA: el componente de combinación incluye todos los registros de la tabla situada más a la izquierda (en MapForce, esta es la tabla situada más arriba de un componente de combinación), así como los registros de la tabla que se combina posteriormente y que cumplen 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:

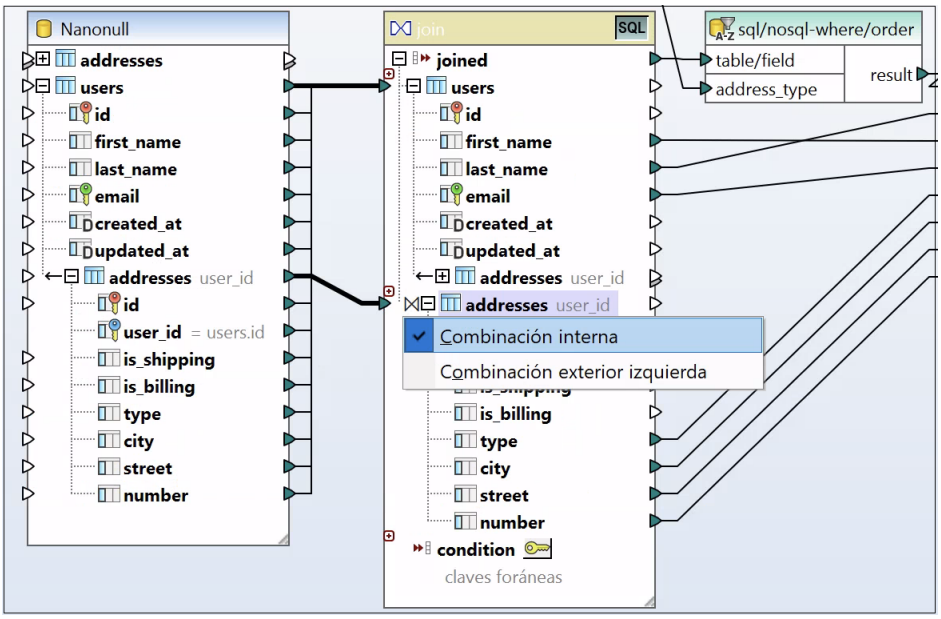
- Combinación interna ⋈
- Combinación izquierda ⋊

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 **Combinación interna** ⋈ o **Combinación izquierda** ⋊ situado delante de cada tabla o vista combinada y seleccione **Combinación interna** o **Combinación externa izquierda** en el menú contextual.
- Haga clic con el botón derecho en la segunda (o tercera, cuarta, etc.) tabla o vista combinada del componente de combinación y seleccione **Tipo de combinación | Combinación interna** o **Tipo de combinación | Combinación exterior izquierda** en el menú contextual.




Nota:

- Si cambia el modo de combinación a COMBINACIÓN EXTERIOR IZQUIERDA, entonces la tabla o vista superior representa la parte izquierda de la combinación.
- Cambiar el modo de combinación afecta a los datos que devuelve el componente de combinación de la misma forma que INNER JOIN o LEFT JOIN afectan a los resultados de una consulta SQL en una BD.

## Nombres de alias

A menudo ocurre que las tablas o vistas de bases de datos combinadas contienen nombres de campo idénticos en ambas estructuras combinadas. 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).

El campo del alias o los nombres de las tablas 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 del 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 413.

**Nota:** No se permite usar componentes WHERE/ORDER de SQL entre una tabla de base de datos y el componente de combinación. Solo pueden agregarse después (pero no antes) de un componente de

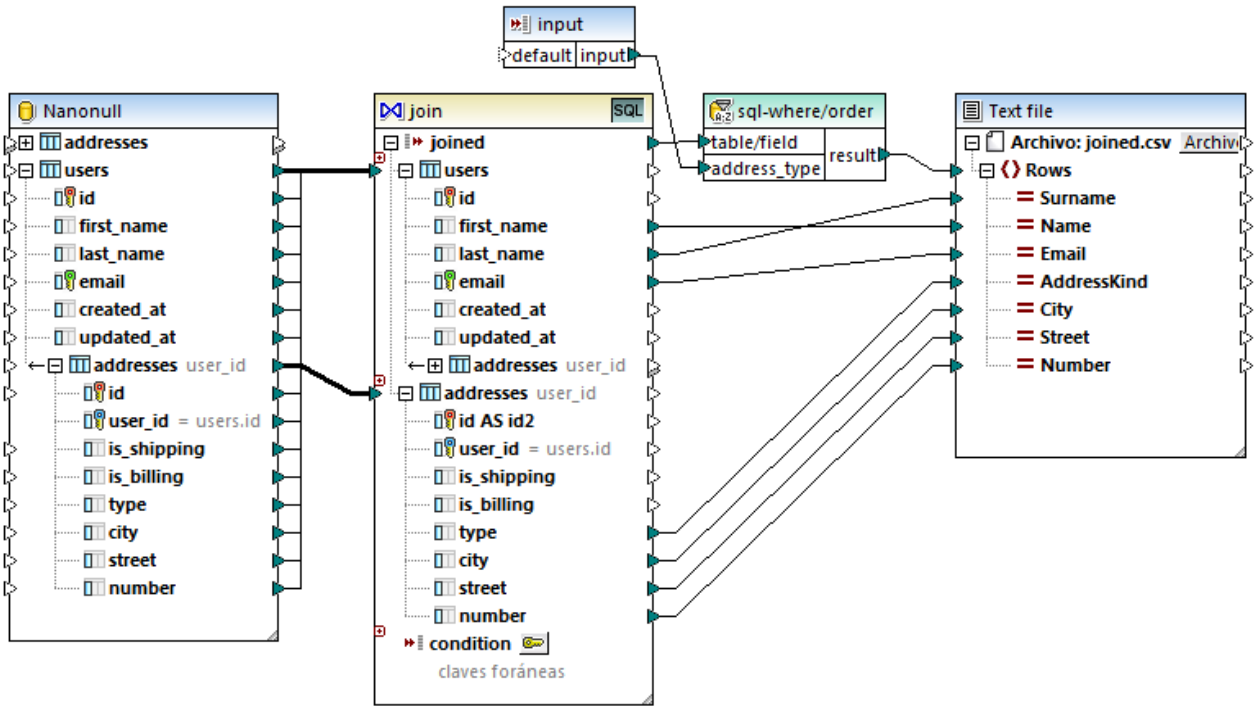
 © 2019-2025 Altova GmbH

combinación. Para más información consulte el apartado Filtrar y ordenar datos de BD (SQL WHERE/ORDER) 437.

## 5.4.4.2 Ejemplo: combinar tablas en modo SQL

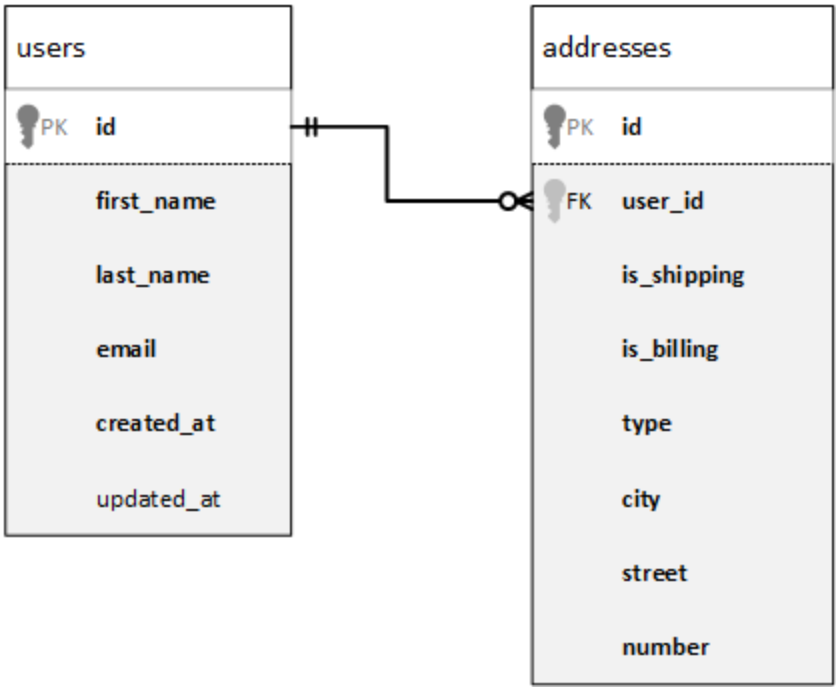
Este ejemplo demuestra cómo combinar datos de dos tablas de base de datos con ayuda de un componente de combinación de MapForce. La operación de combinación se lleva a cabo en modo SQL (véase Combinaciones en modo SQL 410). Recuerde que también se pueden combinar tres o más tablas de una forma muy parecida (véase Ejemplo: crear un informe CSV a partir de varias tablas 421).

Este ejemplo viene acompañado de un diseño de asignación de datos que se encuentra en esta carpeta: **<Documentos>\Altova\MapForce2025\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 los nombres y las direcciones de correos de personas y la segunda (`addresses`) almacena las direcciones 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.


© 2019-2025 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.

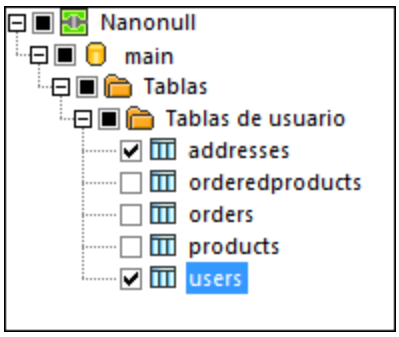
El objetivo de la asignación es recuperar todos los datos (nombre, apellido, correo electrónico, ciudad, calle y número) de todos los usuarios que tengan como mínimo una dirección en la tabla addresses. También debería ser posible recuperar solamente cierto tipo de direcciones (p.ej. sólo las de tipo "home" o sólo 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.

El requisito de la asignación se conseguirá con ayuda de un componente de combinación, tal y como se ilustra en los siguientes pasos.

**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* 259). 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.

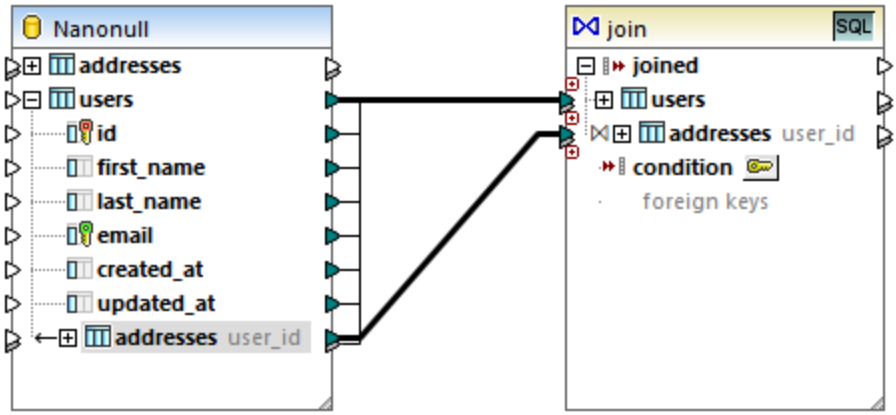
## Paso n°1: Agregar la base de datos de origen

1. En el menú **Insertar** haga clic en el comando **Base de datos**. (También puede hacer clic 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** que está en la carpeta **<Documentos>\Altova\MapForce2025\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

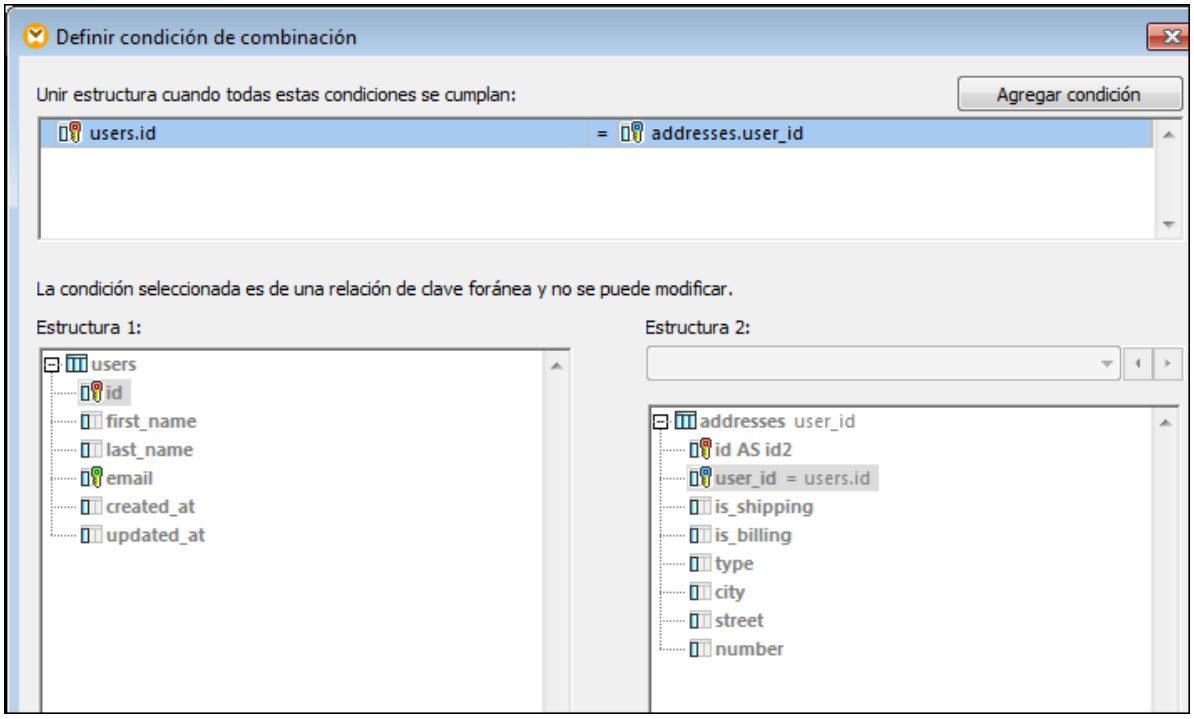
1. En el menú **Insertar**, haga clic en **Combinar**. (También puede hacer clic en el botón **Combinar** de la barra de herramientas.)
2. Dibuje una conexión entre la tabla `users` y la primera entrada del componente de combinación.
3. Expanda la tabla `users` y dibuje una conexión entre la tabla `addresses` (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 399). 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.

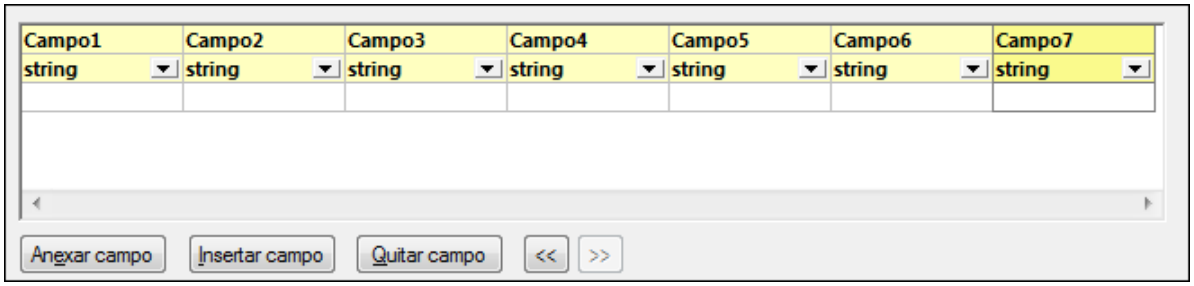
 © 2019-2025 Altova GmbH

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`).

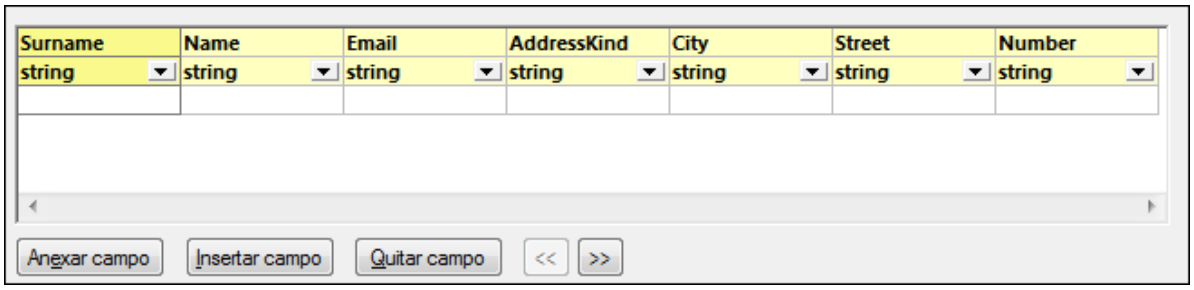



## Paso n°3: Agregar el componente CSV de destino

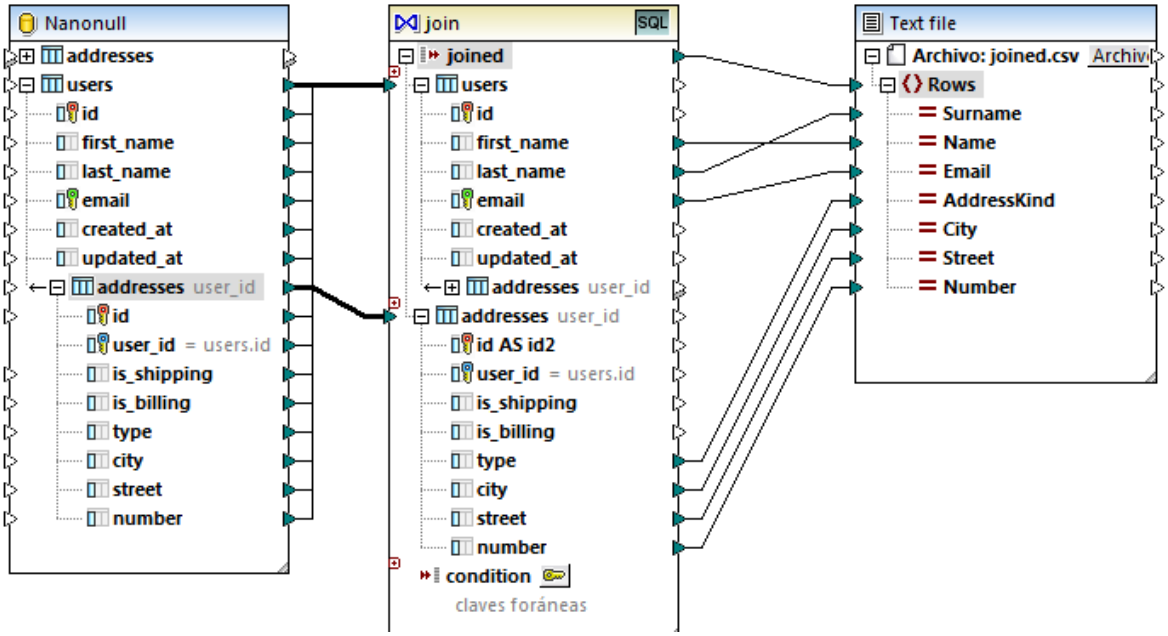
1. En el menú **Insertar** seleccione el comando **Archivo de texto**. (También puede hacer clic en el botón **Insertar archivo de texto** de la barra de herramientas.)
2. Cuando la aplicación le pida que elija un modo de procesamiento, seleccione la opción **Utilizar procesamiento básico para archivos CSV....** .
3. Haga clic en **Anexar campo** varias veces para crear siete campos CSV. Deje las demás opciones de configuración 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 (para que su asignación sea más fácil de leer).



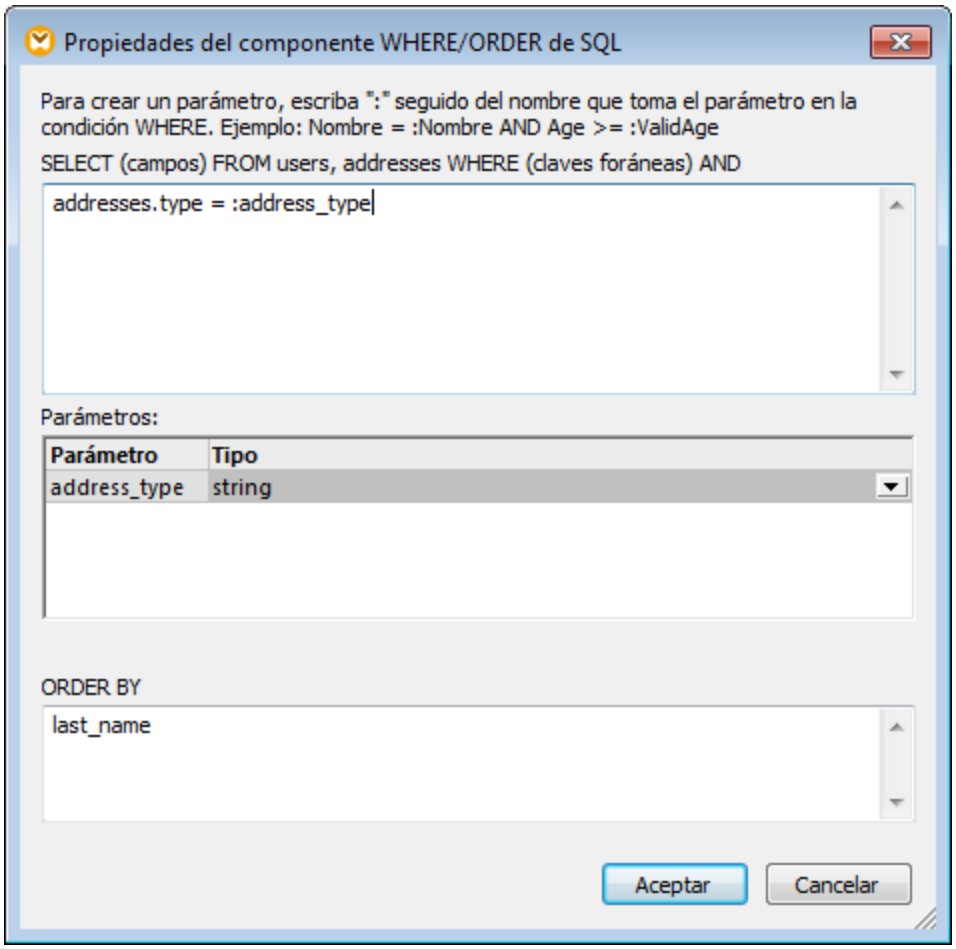

5. Dibuje las conexiones que pueden verse en la siguiente imagen entre el componente de combinación y el componente CSV. La conexión que une el elemento `joined` del componente de combinación y el elemento `Rows` del componente CSV de destino da la orden de crear tantos registros (filas) en el destino como registros existan que cumplan la condición de combinación.



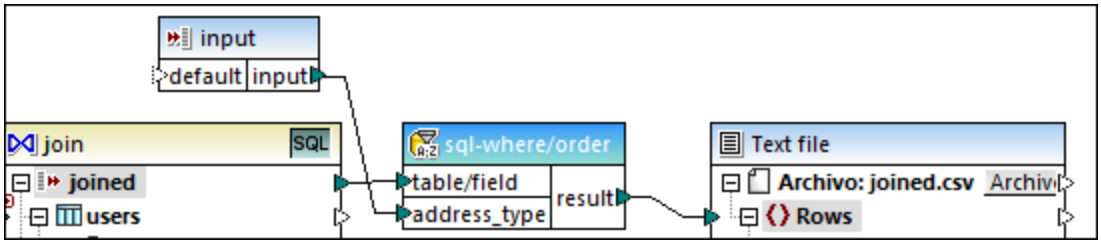

## Paso nº4: Agregar la condición WHERE/ORDER de SQL y el parámetro de entrada

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.

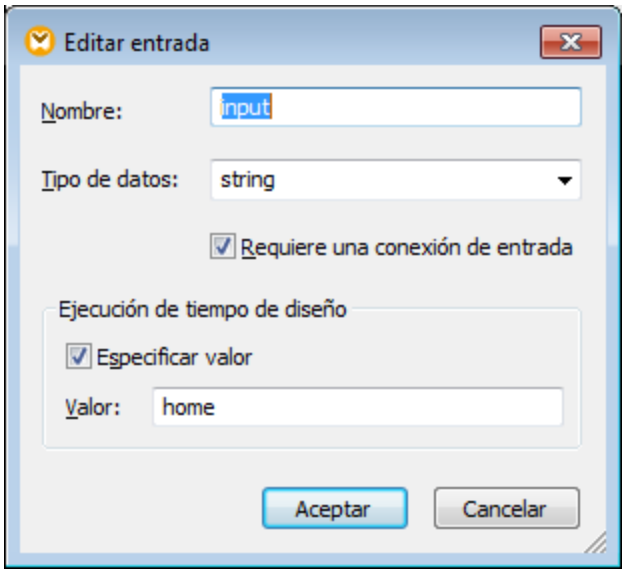

© 2019-2025 Altova GmbH




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 sólo 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. Para más información consulte el apartado Gestionar relaciones de BD 262. Como en este ejemplo se han utilizado las relaciones de tablas ya existentes, no ha sido necesario definir manualmente ninguna condición de combinación. Para ver un ejemplo que describa cómo definir condiciones de combinación manualmente véase el Ejemplo: crear un informe CSV a partir de varias tablas 421.

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 tipo de combinación (INNER JOIN) se indica mediante el icono **Combinación interna** ⋈ situado delante de la tabla **addresses** en el componente de combinación. También puede cambiar el modo de combinación LEFT OUTER JOIN ⋊ (véase Cambiar el modo de combinación 411). No obstante, tenga en cuenta que al cambiar el modo de combinación no cambia el resultado de este ejemplo.

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 el apartado Filtrar y ordenar datos de BD (SQL WHERE/ORDER) 437.

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 el apartado Pasar parámetros a la asignación 369.

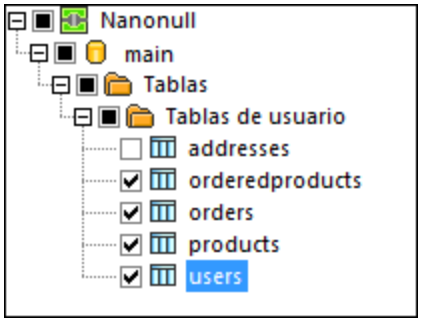
 © 2019-2025 Altova GmbH

### 5.4.4.3 Ejemplo: crear un informe CSV a partir de varias tablas

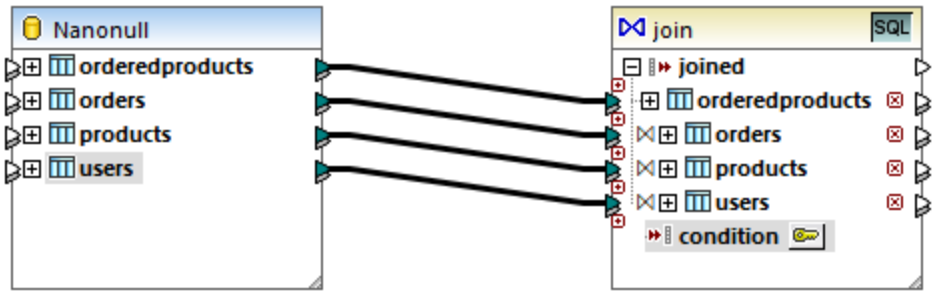
Este ejemplo demuestra cómo combinar varias tablas de base de datos para extraer datos y guardarlos en un solo informe en formato CSV. La base de datos utilizada en este ejemplo se llama **Nanonull.sqlite** y está guardada en este directorio: **<Documentos>\Altova\MapForce2025\MapForceExamples\Tutorial**. 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.

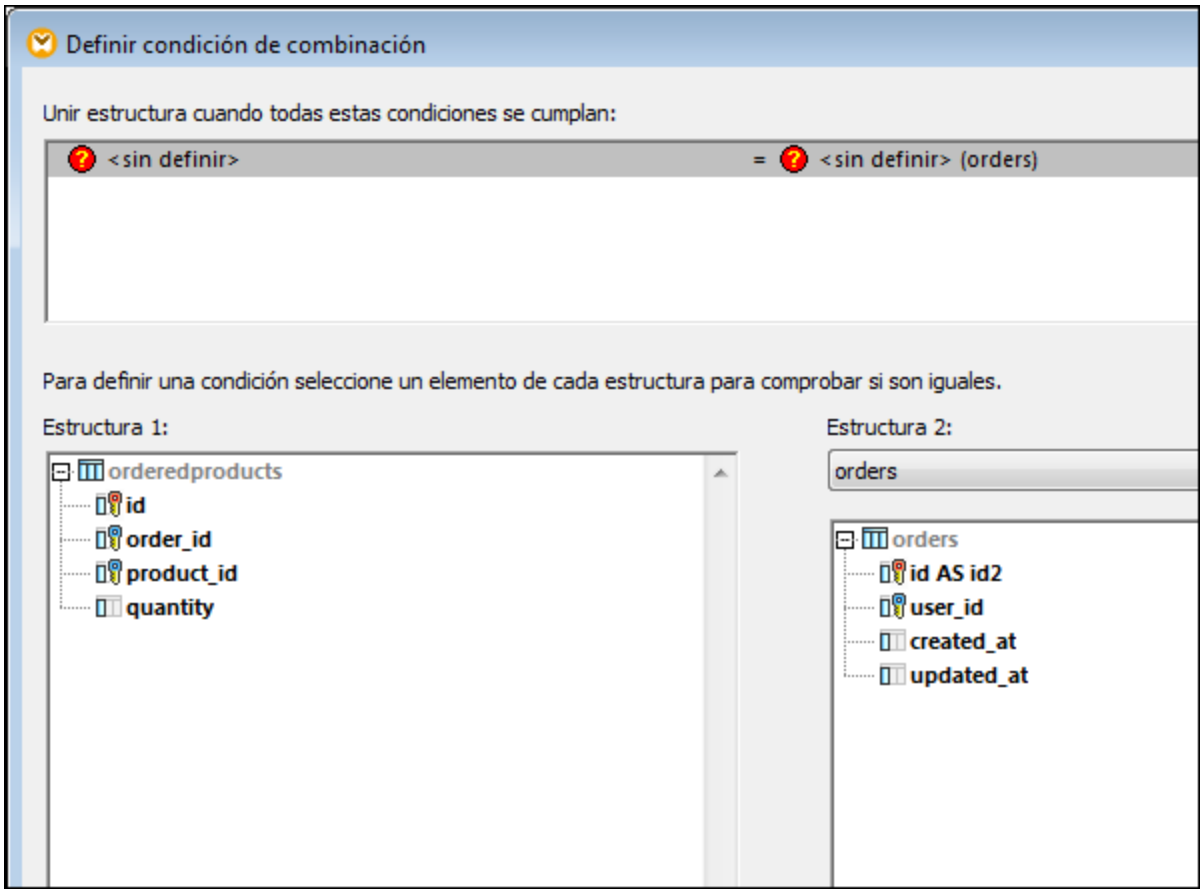


**Nota: otro método alternativo sería conectar la tabla** `orderedproducts` al componente de combinación, después la tabla `orders` (la que tiene anidada y no la que está al mismo nivel) y así sucesivamente hasta que todas las tablas combinadas estén anidadas bajo la misma tabla raíz (*véase* *Gestionar relaciones de BD* 262). 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 413. También puede consultar la asignación de datos **<Documentos>\Altova\MapForce2025\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:

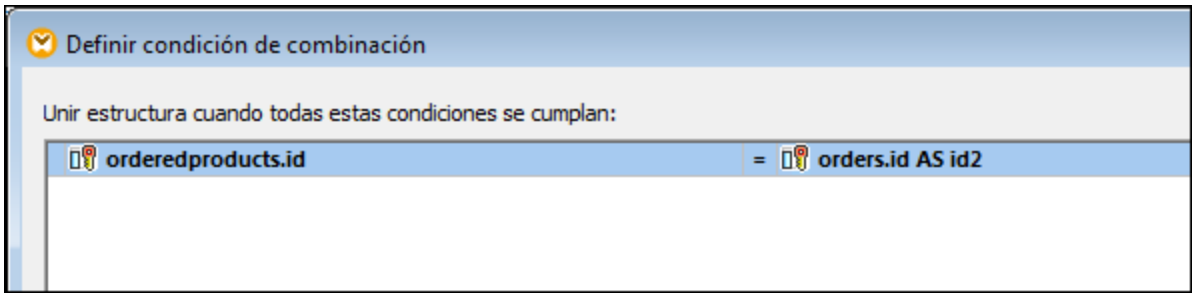
1. `orderedproducts`
2. `orders`
3. `products`
4. `users`

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*.

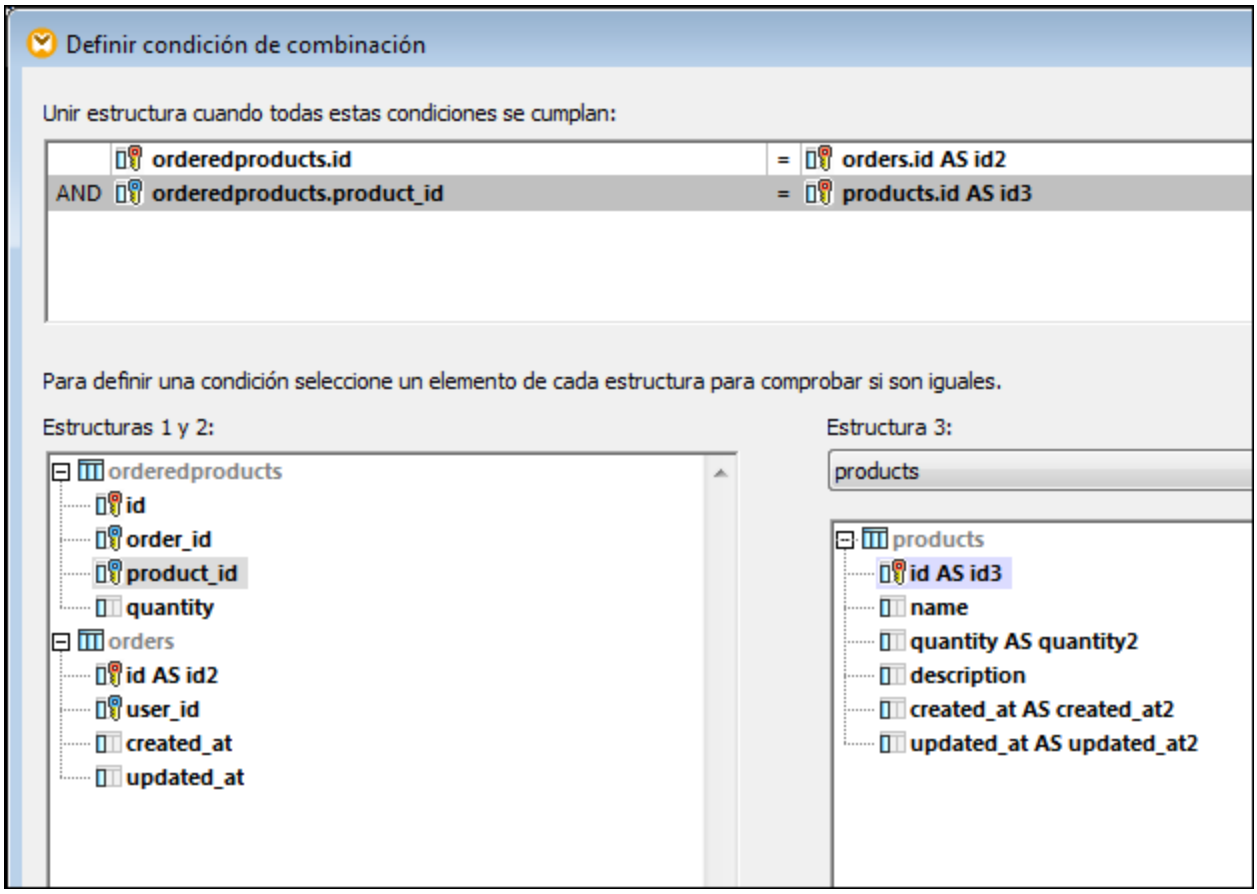



 © 2019-2025 Altova GmbH

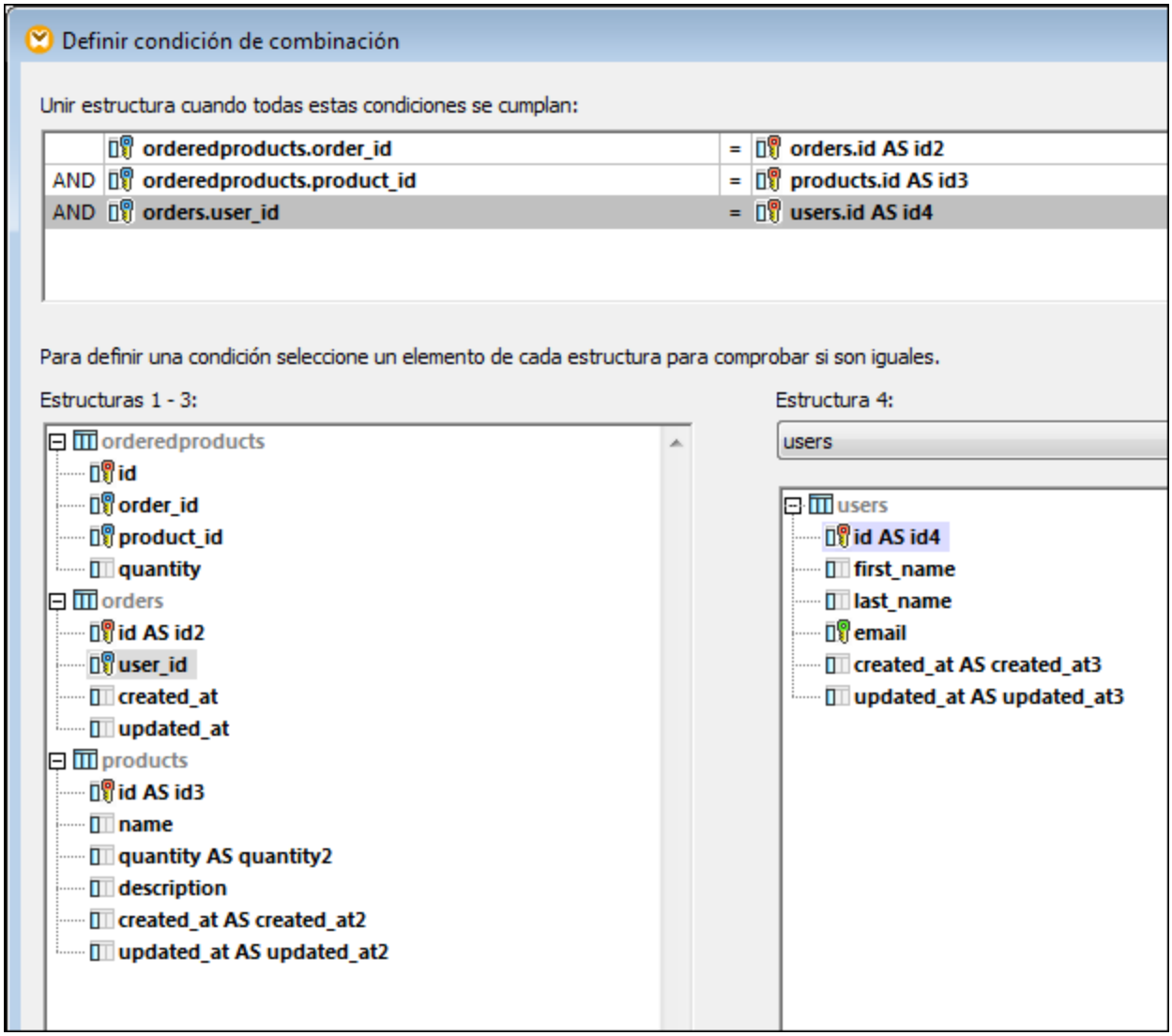
Para definir la primera condición de combinación haga clic en el elemento `order_id` del panel izquierdo y en el elemento `id` del panel derecho. Esto empareja los campos `orderedproducts.order_id` y `orders.id`:




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`).

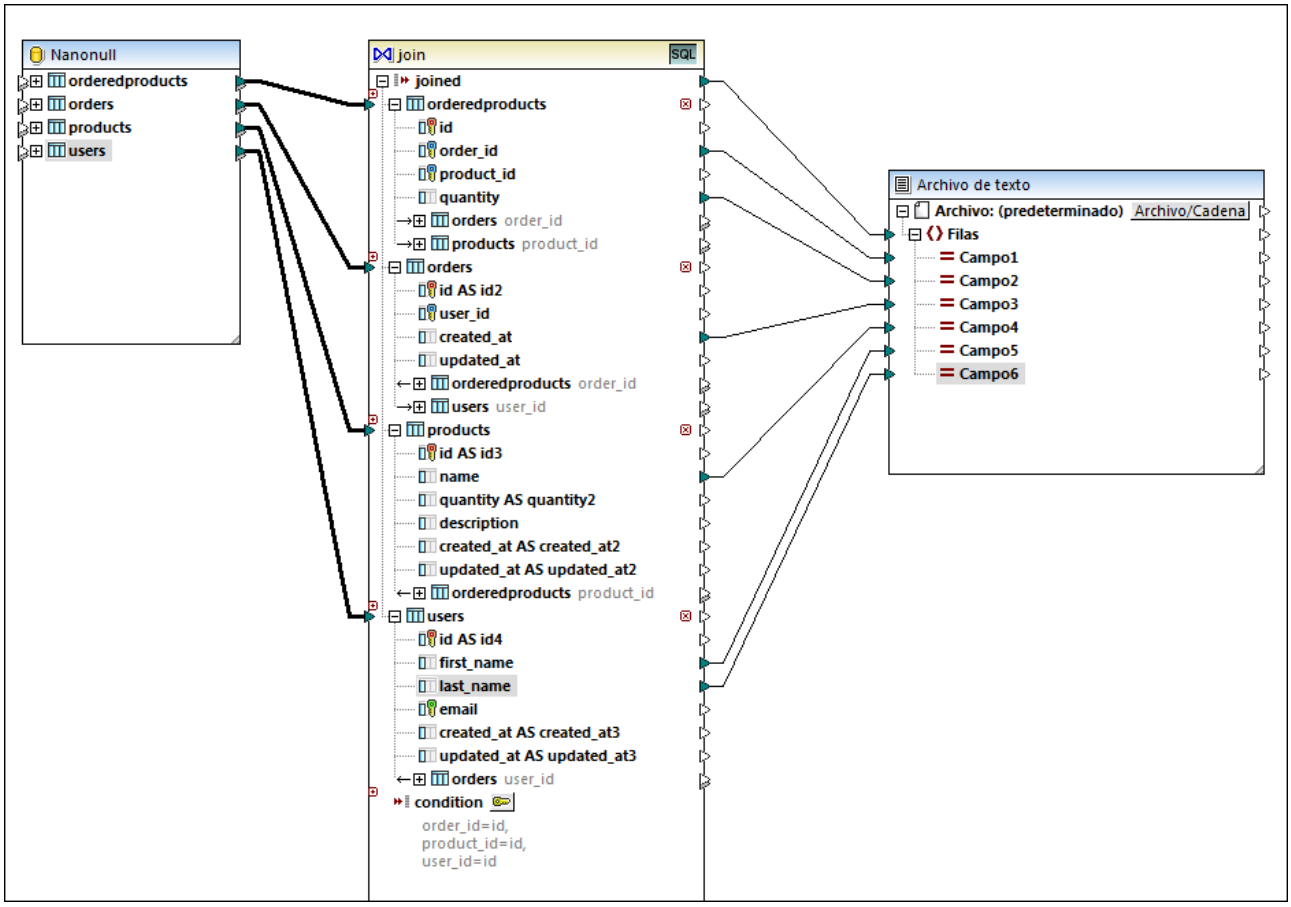



Para combinar la cuarta tabla (`users`) basta con seleccionar la tabla `users` en la lista desplegable y podremos emparejar los campos `orders.user_id` con los campos `users.id`.




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* 344) y conecte los elementos del componente de combinación con el componente CSV de destino tal y como muestra la imagen siguiente.

 © 2019-2025 Altova GmbH




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`)

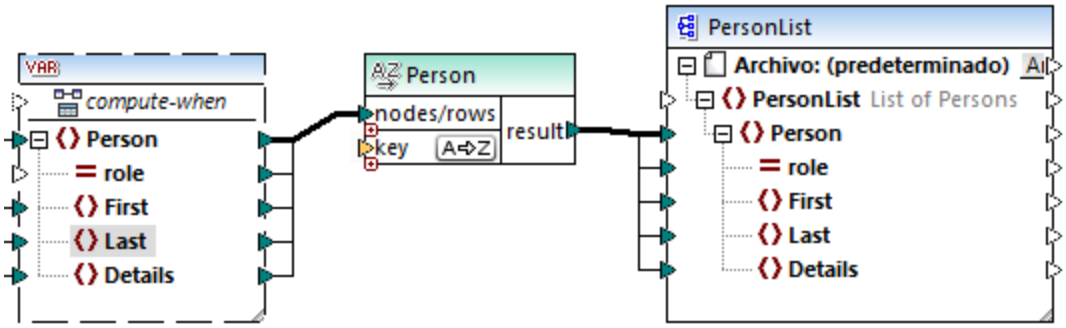
Todas las tablas de este ejemplo están combinadas con el modo de combinación interna. Para más información sobre cómo cambiar al modo de combinación externa izquierda consulte Cambiar el modo de combinación 411.

# 5.5 Componentes de ordenación

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* 437) 437).

### 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).



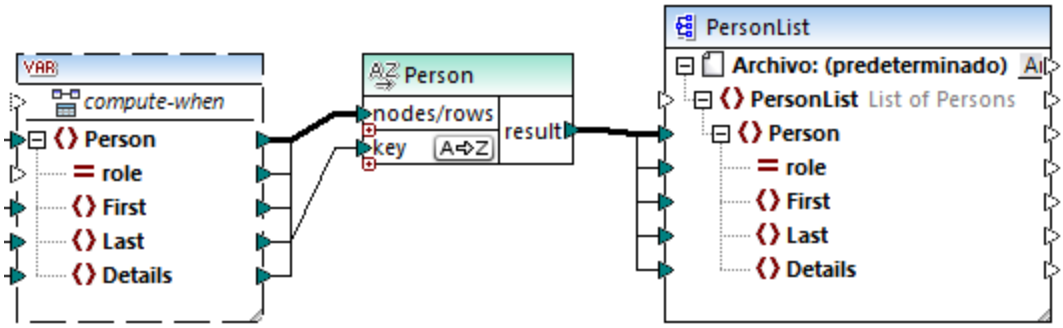

- En el menú **Insertar** seleccione el comando **Componente de ordenación: nodos/filas** (o haga clic en el botón **Insertar componente de ordenación** de la barra de herramientas). Esto inserta el componente de ordenación pero sin conectarlo.

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`.

 © 2019-2025 Altova GmbH

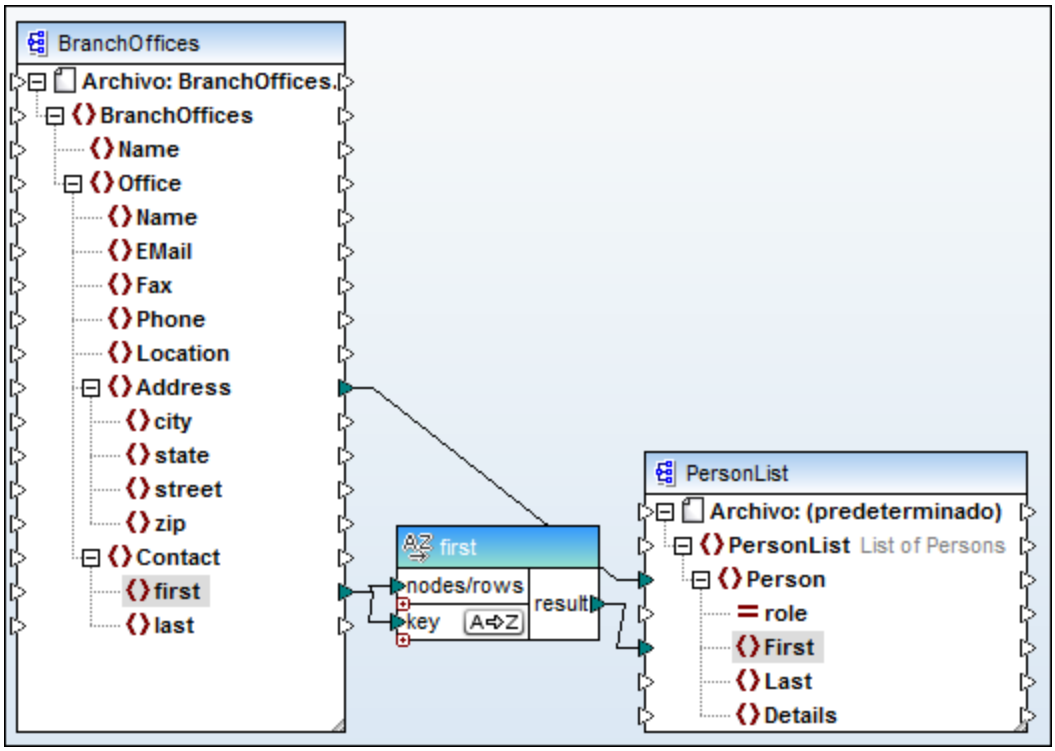



### Para cambiar el orden:

- 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.

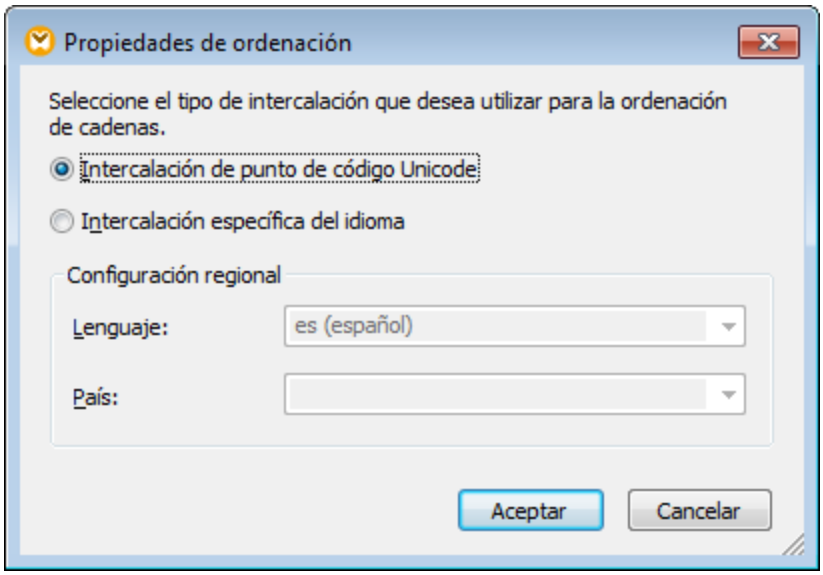
### Para ordenar datos de entrada compuestos por elementos de tipo simple:

- 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`.



**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*).

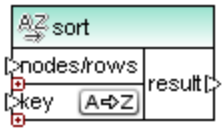



**Intercalación de punto de código:** esta opción predeterminada compara/ordena las cadenas basándose en valores de punto de código. Los valores de punto de código son enteros que se asignaron a caracteres abstractos del conjunto de caracteres universal adoptado por el consorcio Unicode. Esta opción 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.

## 5.5.1 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*

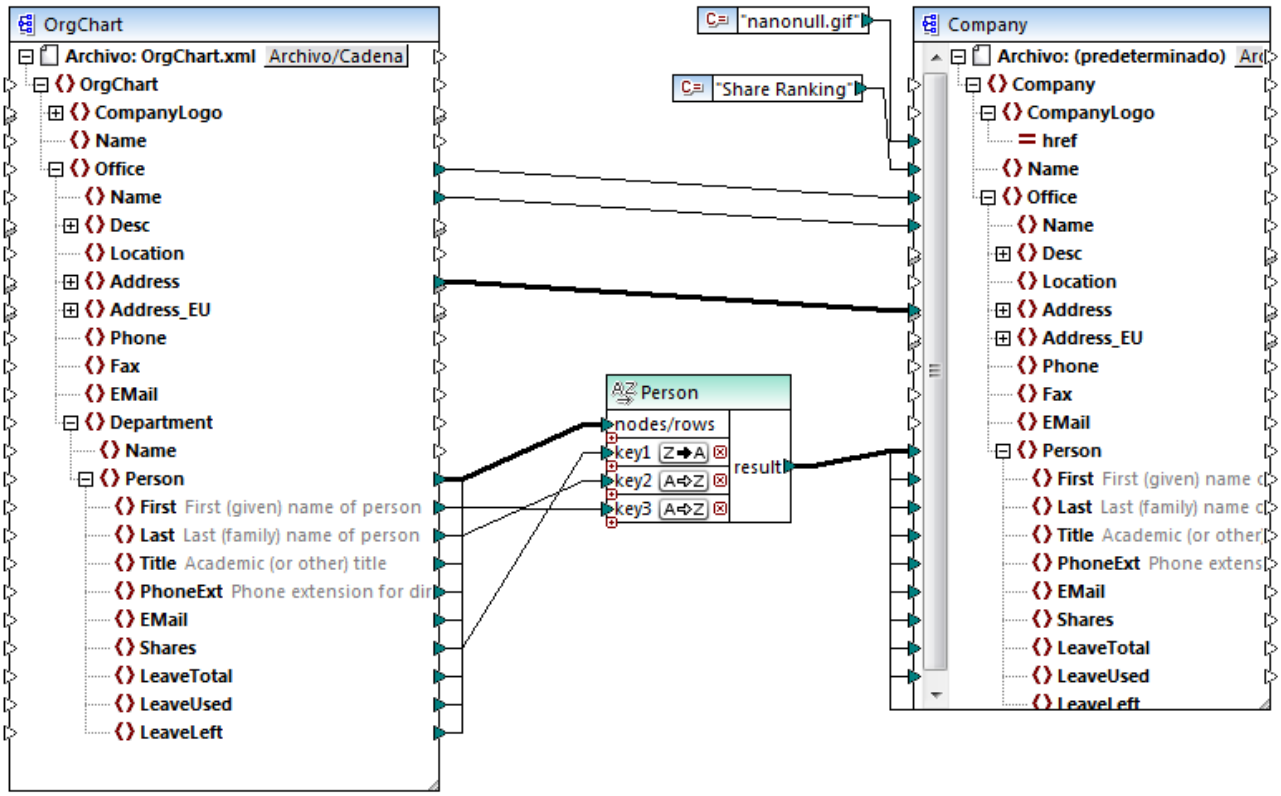
Si desea usar varios criterios de ordenación, deberá realizar varios ajustes en el componente:

- 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.

 © 2019-2025 Altova GmbH

- Arrastre y coloque una conexión encima del icono para agregar un criterio y al mismo tiempo conectarlo.

El diseño de asignación **<Documentos>\Altova\MapForce2025\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:

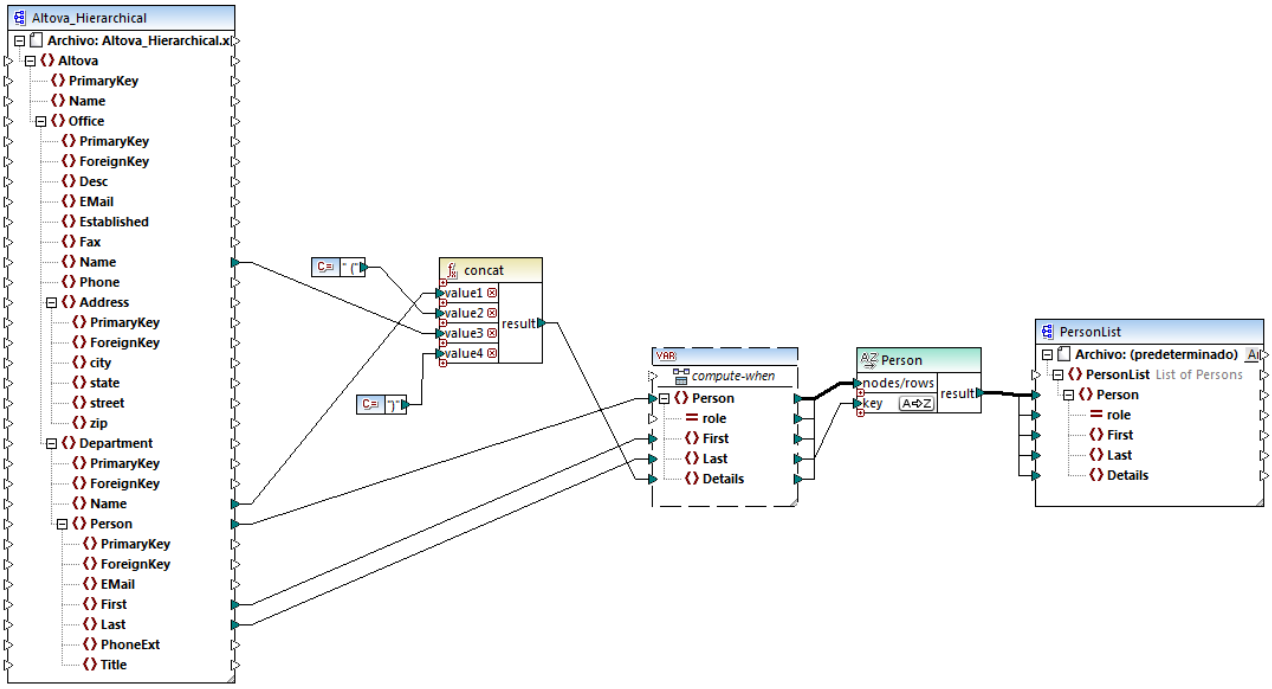
1. `Shares` (número de acciones en propiedad de cada persona)
2. `Last` (apellido)
3. `First` (nombre)

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).

## 5.5.2 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 estos se orden con ayuda de variables intermedias. Este ejemplo viene acompañado de un diseño de asignación de datos ubicado en esta ruta de acceso:
**<Documentos>\Altova\MapForce2025\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. Las oficinas se dividen en departamentos y, 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`**. En esta asignación, esta función devuelve una cadena con el formato `Office(Departament)`. Toma como entrada el nombre de la oficina, el nombre del departamento y dos constantes que suministran los corchetes inicial y final. Consulte también Agregar una función a una asignación 462.

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 (consulte cómo se ejecuta una asignación de datos en Reglas y estrategias de asignación de datos 80). 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 y Conexiones de copia total 57 para obtener más información.
3. Un componente de ordenación que ordena los datos. Observe que el criterio de ordenación del `componente de ordenación` está conectado al elemento `Last` de la variable, lo cual permite ordenar todos los registros por apellido.

# 5.6 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:

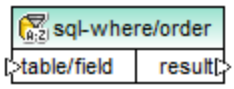
En la estructura de filtro anterior, la condición que está conectada a bool determina si la entrada **node/row** que está conectado se pasa a la salida **on-true** o a la salida **on-false**. Es decir, si la condición se cumple (es true), entonces **node/row** se redirige a la salida **on-true**. Por 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* 457).

Para ver un ejemplo con instrucciones paso a paso consulte el apartado Ejemplo: filtrar nodos 434.

## 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 437.

 © 2019-2025 Altova GmbH

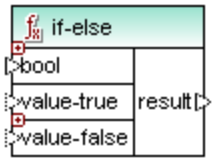
## Devolver un valor de forma condicional

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>
```

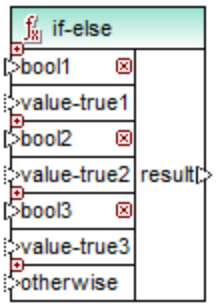
Una **condición If-Else** permite devolver por cada elemento de la lista el valor "alto" si la temperatura supera los 20 grados centígrados y el valor "bajo" si la temperatura es inferior a los 5 grados centígrados.

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.



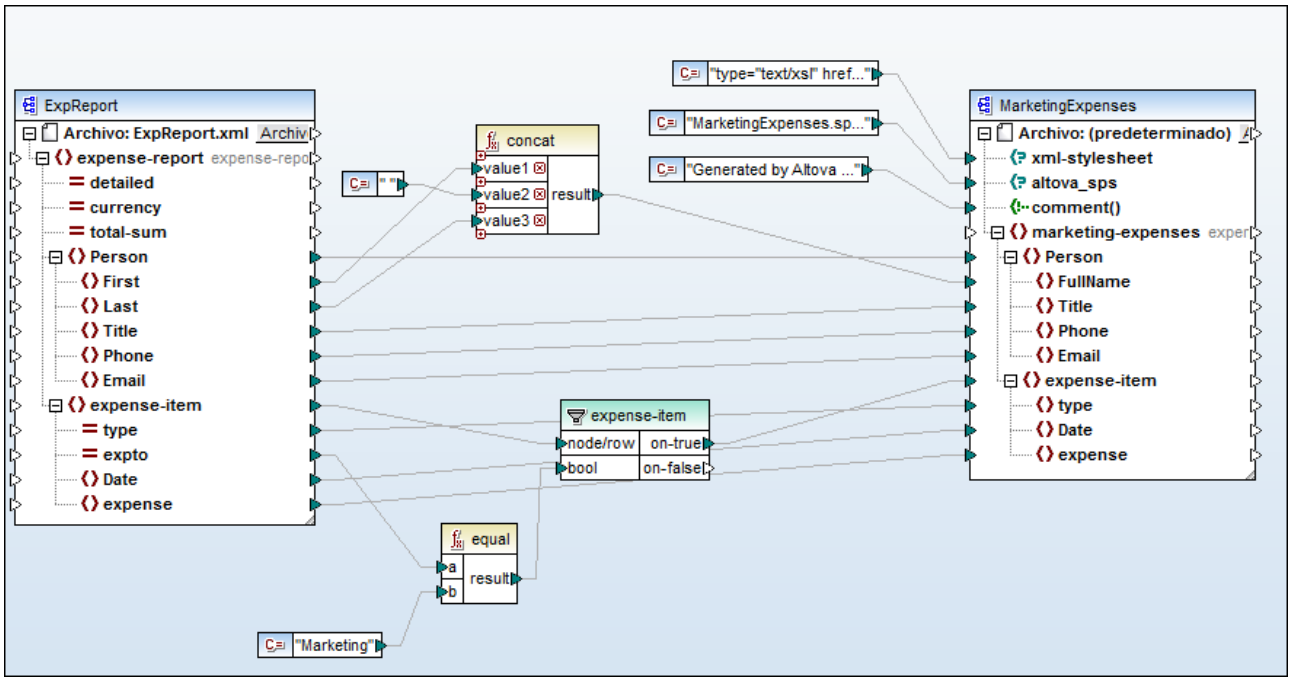
Las **condiciones If-Else** extendidas, se evalúan de arriba a abajo (se comprueba en primer lugar las condiciones situadas más arriba). Si desea devolver un valor aunque no se cumpla ninguna condición (es decir, aunque ninguna condición sea true), conecte el valor a **`otherwise`**.

Para ver un ejemplo con instrucciones paso a paso consulte el apartado Ejemplo: devolver un valor de forma condicional 436.

## 5.6.1 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 437).

La asignación de este ejemplo es **<Documentos>\Altova\MapForce2025\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**.)

 © 2019-2025 Altova GmbH

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* 461). 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>
```

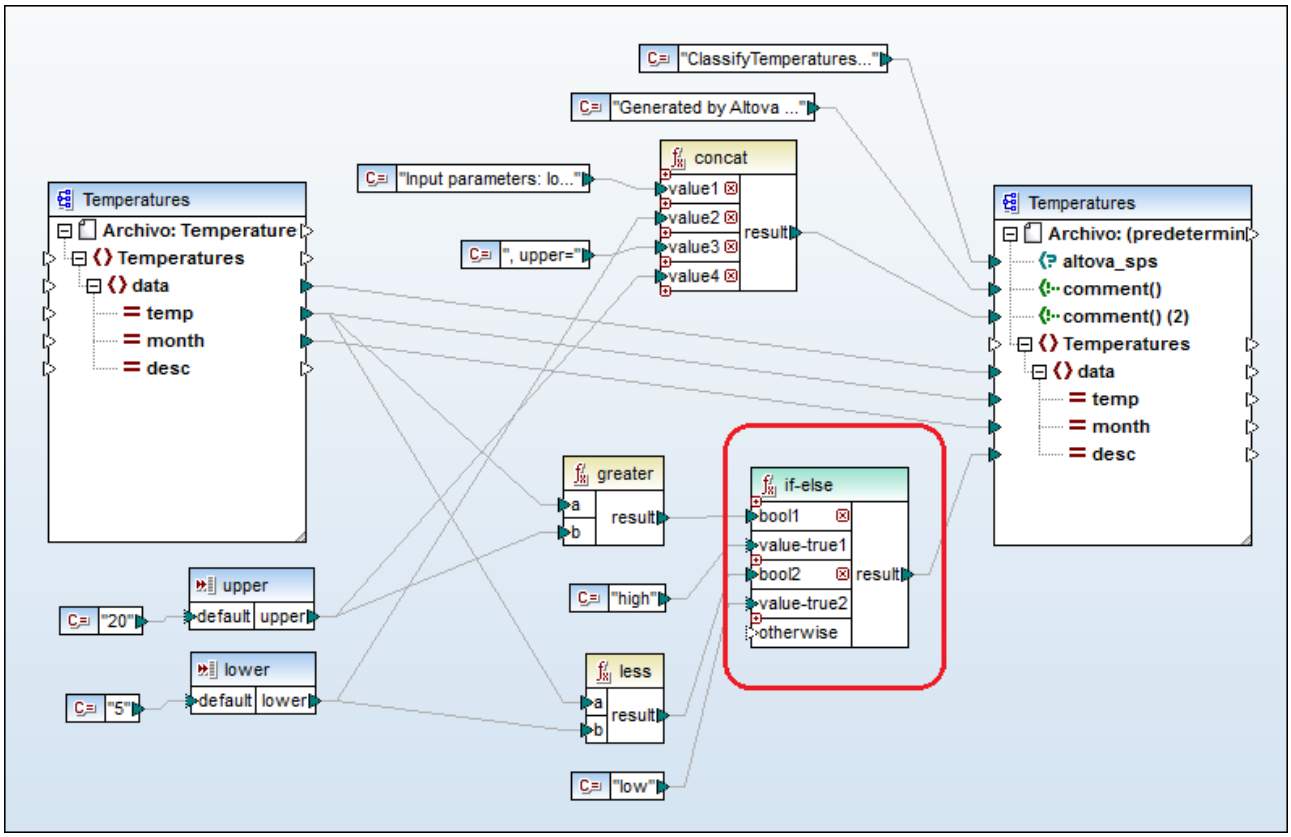
```
      <expense>299.45</expense>
   </expense-item>
...
```

*XML de salida después de ejecutarse la asignación*

## 5.6.2 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* 434).

La asignación de este ejemplo es **<Documentos>\Altova\MapForce2025\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

 © 2019-2025 Altova GmbH

"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* 462). 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 369 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*

## 5.6.3 Filtrar y ordenar datos de BD

Cuando necesite filtrar y ordenar datos de base de datos, utilice un componente **SQL/NoSQL-WHERE/ORDER**. Esto le permite introducir manualmente una cláusula SQL WHERE que filtre los datos. También tiene la opción de especificar una cláusula ORDER BY si desea ordenar el conjunto de registros según cierto campo de la base de datos, en orden ascendente o descendente.

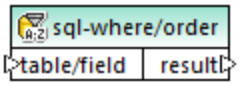
El componente **SQL/NoSQL-WHERE/ORDER** debe estar conectado a una tabla o campo de un componente de base de datos. También es posible conectar un componente **SQL/NoSQL-WHERE/ORDER** con un

componente de **combinación** si necesita filtrar el conjunto o los registros combinados. Para más información, consulte Combinar datos de bases de datos 408.

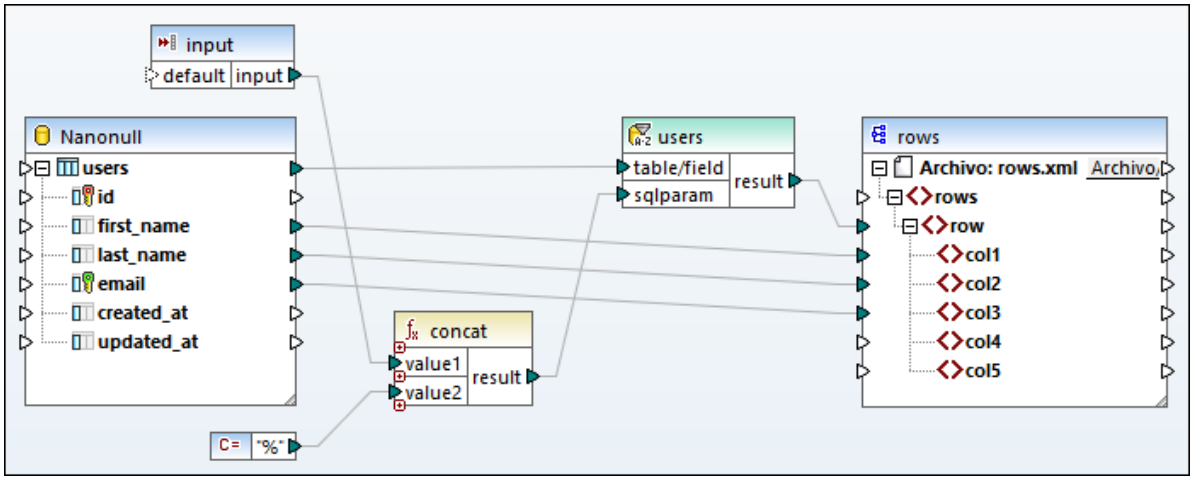
## Agregar un componente SQL/NoSQL-WHERE/ORDER

Para añadir un componente **SQL/NoSQL-WHERE/ORDER** a la asignación:

1. Vaya al menú **Insertar** y haga clic en **SQL/NoSQL-WHERE/ORDEN**. Este componente presenta esta estructura predeterminada:

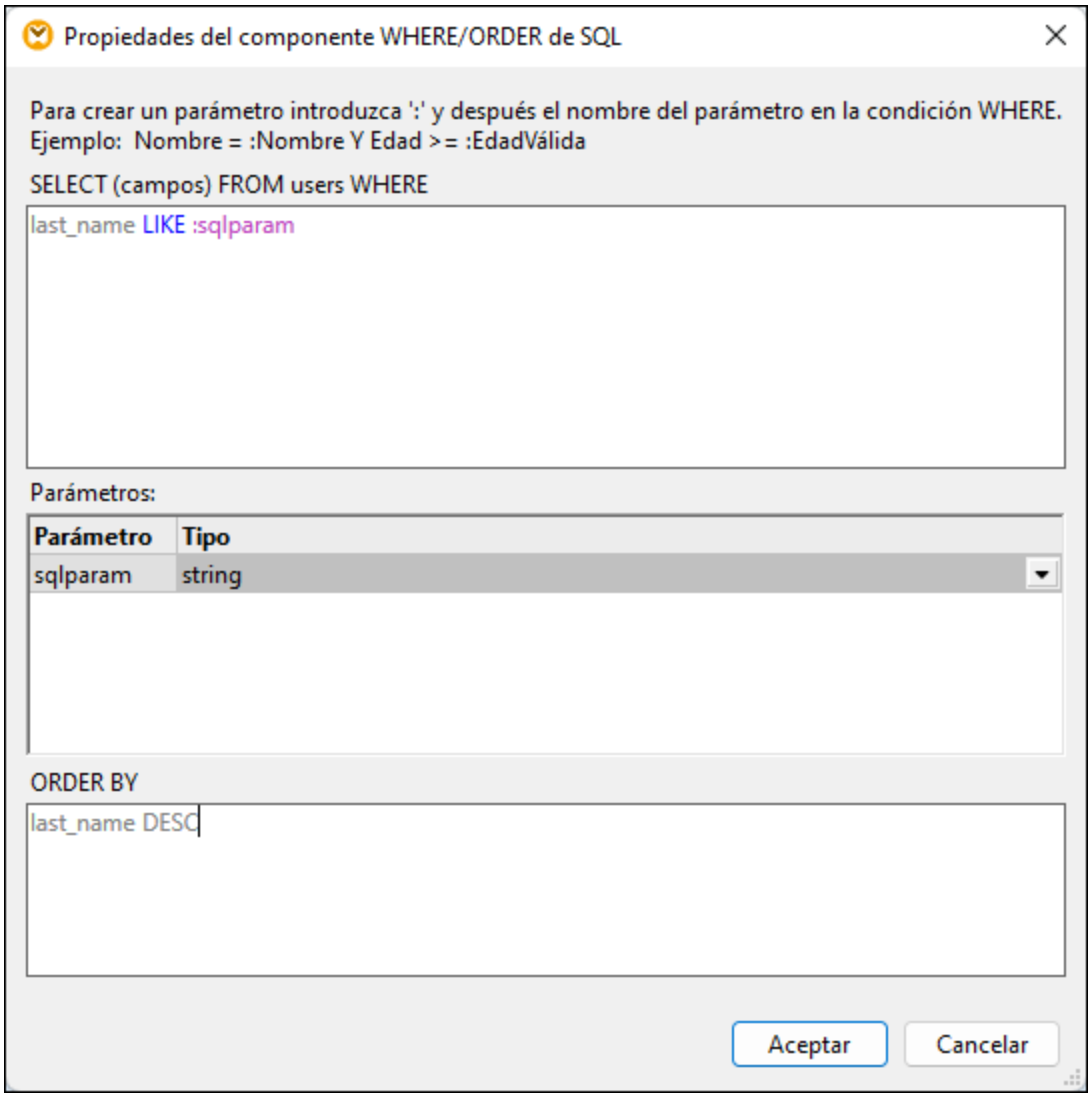



2. Conecte una tabla o campo de la base de datos de origen al elemento `tabla/campo` del componente **SQL/NoSQL-WHERE/ORDER**. Encontrará la asignación de muestra **`FilterDatabaseRecords.mfd`** (*imagen siguiente*) en la siguiente carpeta: **`<Documentos>\Altova\MapForce2025\MapForceExamples\Tutorial\`**. En esta asignación, el componente **SQL/NoSQL-WHERE/ORDER** toma los datos de la tabla de origen `users`, filtra todos sus registros y selecciona solos aquellos en los que el apellido comienza por la letra M (*ver subapartado siguiente*).




3. Haga doble clic en el título del componente **SQL/NoSQL-WHERE/ORDER**. Alternativamente, haga clic con el botón derecho y seleccione **Propiedades** en el menú contextual. Esto abre el cuadro de diálogo **Propiedades WHERE/ORDER de SQL/NoSQL.**

 © 2019-2025 Altova GmbH



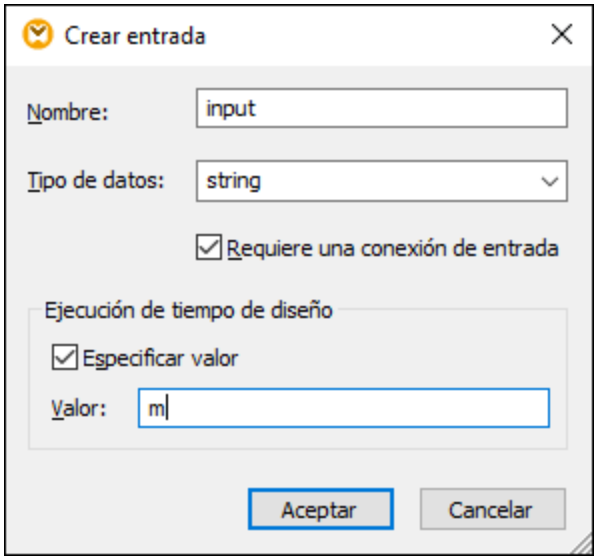

4. Escriba la cláusula WHERE de SQL en el cuadro de texto superior. En nuestro ejemplo, la cláusula WHERE de SQL es la siguiente: **`last_name LIKE :sqlparam`**. Opcionalmente, escriba una cláusula ORDER BY. La imagen anterior ilustra las cláusulas WHERE y ORDER BY definidas en la asignación **`FilterDatabaseRecords.mfd`** (esta configuración se explica con más detalle a continuación). Para más ejemplos, consulte Crear cláusulas WHERE y ORDER BY 441.

## Parámetros en componentes SQL/NoSQL-WHERE/ORDER

El componente **SQL/NoSQL-WHERE/ORDER** utilizado en la asignación **`FilterDatabaseRecords.mfd`** tiene la siguiente cláusula WHERE: **`last_name LIKE :sqlparam`**, donde `last_name` se refiere al nombre del campo de la base de datos en la tabla conectada; `LIKE` es un operador SQL; `:sqlparam` crea un parámetro llamado `sqlparam` en la asignación.

Los parámetros del componente **SQL/NoSQL-WHERE/ORDER** son opcionales. La misma cláusula WHERE se puede escribir sin parámetros de la siguiente manera: Sin parámetros, la cláusula WHERE anterior podría haberse escrito así: `Last LIKE "M%"`. Esto recuperaría todas las personas cuyo apellido empiece por la letra M. Para que esta consulta sea aún más flexible, hemos agregado un parámetro en lugar de «`M%`». Esto permite

suministrar cualquier otra letra de la asignación: por ejemplo, D, y recuperar así las personas cuyo apellido empiece por D con solo cambiar una constante o un parámetro de entrada de la asignación. En la asignación anterior, la letra de entrada procede de un componente de entrada llamado `input`. Si hace doble clic en la barra de título de este componente y abre sus propiedades, observará que se da `m` como valor de ejecución en tiempo de diseño (*imagen siguiente*).




En la asignación, el carácter comodín SQL % se suministra mediante una constante. A continuación, este carácter comodín se concatena con el valor del parámetro con ayuda de la función **`concat`**. La ventaja es que no tiene que escribir comodines SQL en la línea de comandos si esta asignación se ejecuta en otro entorno (por ejemplo, MapForce Server).

## Aspecto de los componentes SQL/NoSQL-WHERE/ORDER

Los componentes **SQL/NoSQL WHERE/ORDER** cambian de aspecto en función de la configuración que tengan definida. De este modo, podrá ver rápidamente y directamente desde la asignación lo que hace el componente **SQL/NoSQL WHERE/ORDER** (*ver tabla siguiente*).

| | |
|---|---|
| users<br>table/field result | Se ha definido una cláusula WHERE. |
| users<br>table/field<br>sqlparam result | Se ha definido una cláusula WHERE con un parámetro. El nombre del parámetro es visible bajo el elemento `table/field`. |
| users<br>table/field<br>sqlparam result | Se ha definido una cláusula WHERE con un parámetro. Además, se ha definido una cláusula ORDER BY. La ordenación se indica mediante el icono de ordenación A-Z. |

Si coloca el cursor sobre el título del componente **SQL/NoSQL WHERE/ORDER**, aparecerá información rápida que muestra las cláusulas que se han definido.


© 2019-2025 Altova GmbH

### Filtrado de datos en Azure CosmosDB (Enterprise Edition)

Si el nombre de un contenedor en su CosmosDB tiene caracteres especiales (por ejemplo, **+**, **-**, **@**, etc.) o es una palabra clave, este nombre no podrá utilizarse en las consultas y obtendrá un error de sintaxis de la base de datos. Para evitar posibles problemas, MapForce utiliza la siguiente sintaxis para las consultas en CosmosBD:

```
SELECT * FROM ROOT AS c WHERE c.<campo> ORDER BY c.<campo>
```

La palabra clave `ROOT` de la cláusula `FROM` hace referencia al contenedor actual que está consultando. Dado que la palabra clave `ROOT` no puede utilizarse para apuntar a campos en las cláusulas `WHERE` y `ORDER BY`, el contenedor recibe el nombre de alias **c** que se asigna al contenedor mediante la palabra clave `AS`. Asignar el nombre del alias al contenedor permite filtrar y ordenar su contenido.

Tenga en cuenta que en el cuadro de diálogo **WHERE/ORDER**, todas las referencias de campo deben llevar como prefijo el nombre de alias **c** del contenedor (*ver ejemplo siguiente*).

```
SELECT * FROM ROOT AS c WHERE c.age > 20 ORDER BY c.name
```

## 5.6.3.1 Crear cláusulas WHERE y ORDER BY

Tras añadir un componente **WHERE/ORDER de SQL/NoSQL** a la asignación deberá añadir una condición (cláusula) 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/NoSQL** (*véase el apartado anterior*).

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 un parámetro que se pueda 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. 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 |

| Operador | Descripción |
|---|---|
| <= | 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 acaben en «r» de un campo llamado lastname, debe usar esta expresión:

```
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")
```

Véase también el apartado Ejemplo: extraer datos de columnas tipo XML de IBM DB2 312.

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/NoSQL**. 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 n°1

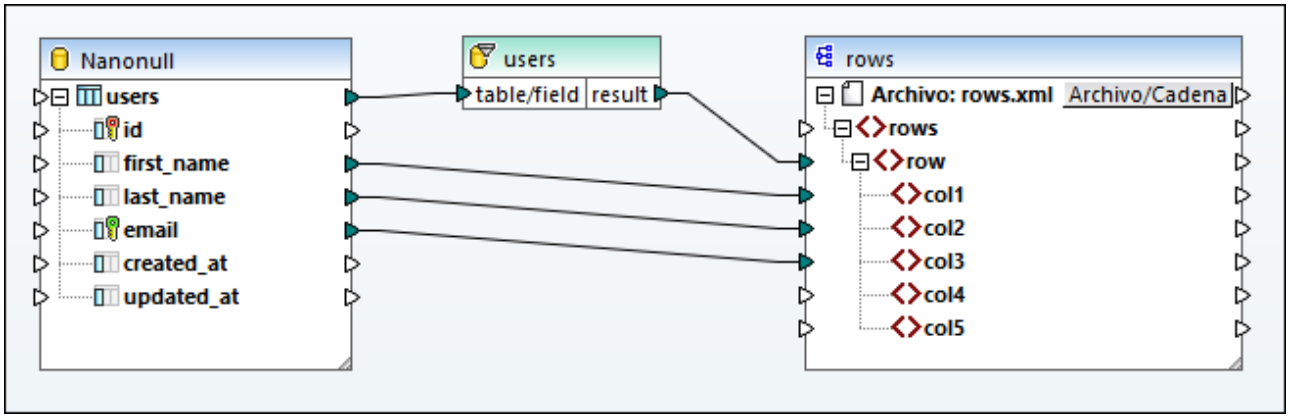
La condición WHERE que aparece a continuación está vinculada a la tabla users del componente de BD **Nanonull.sqlite**. Recupera los registros en los que last_name es mayor que la letra "M". En otras palabras, todos los nombres a partir del usuario "Marzolla".

```
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).

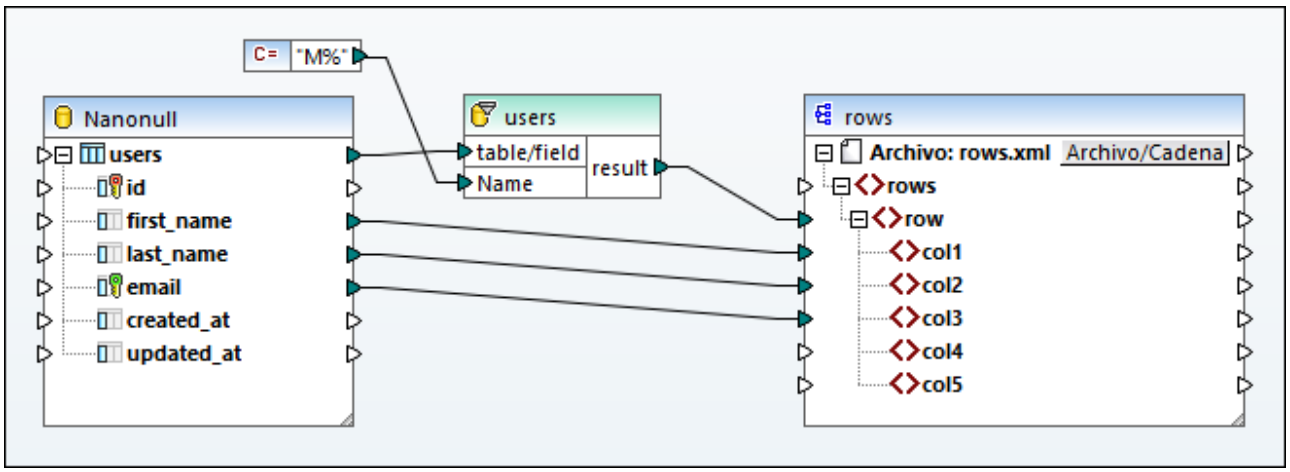
 © 2019-2025 Altova GmbH




## Ejemplo n°2

La condición WHERE que aparece a continuación crea un parámetro `param` que aparece en el componente **WHERE/ORDER de SQL/NoSQL** de la asignación.

```
last_name LIKE :param
```

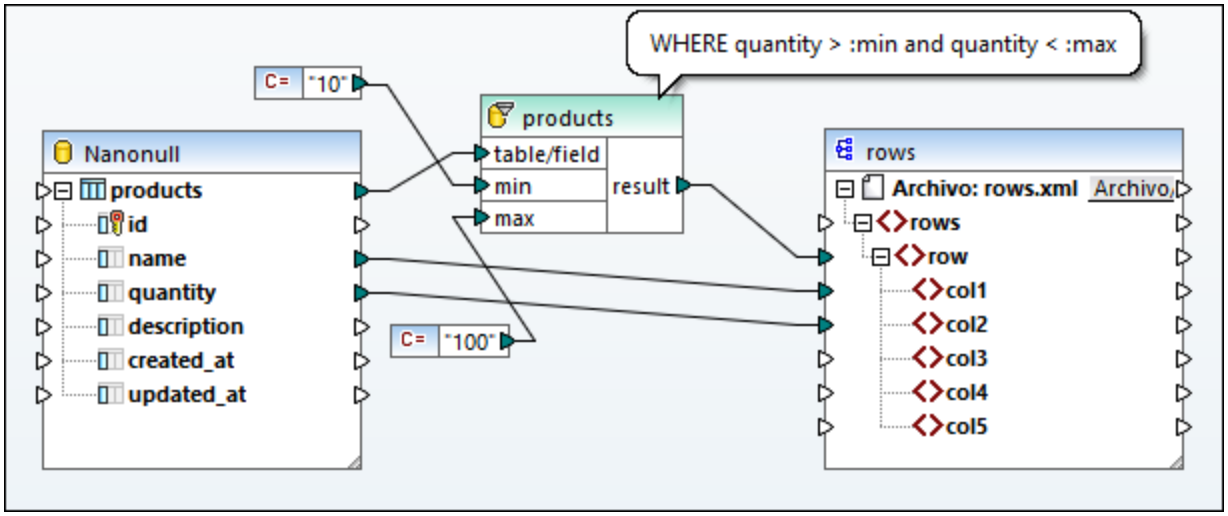



El componente de constante `%M` aporta el valor del parámetro `param`. El comodín % denota cualquier número de caracteres. Esto hace que la asignación busque un parámetro en la columna `last_name` (todos los apellidos que empiecen en "M").

## Ejemplo nº3

La condición WHERE que aparece a continuación crea los dos parámetros `min` y `max` a los que se comparan los valores actuales de `quantity`. Los valores máximo (max) y mínimo (min) se indican por medio de constantes.

```
quantity > :min and quantity < :max
```



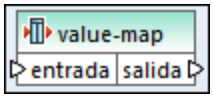

La condición WHERE de este ejemplo también se podría escribir con el operador BETWEEN:

```
quantity BETWEEN :min and :max
```


© 2019-2025 Altova GmbH

# 5.7 Asignación de valores

Con los componentes value-map (*imagen siguiente*) 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 una **salida**.




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 432.

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 Excel.

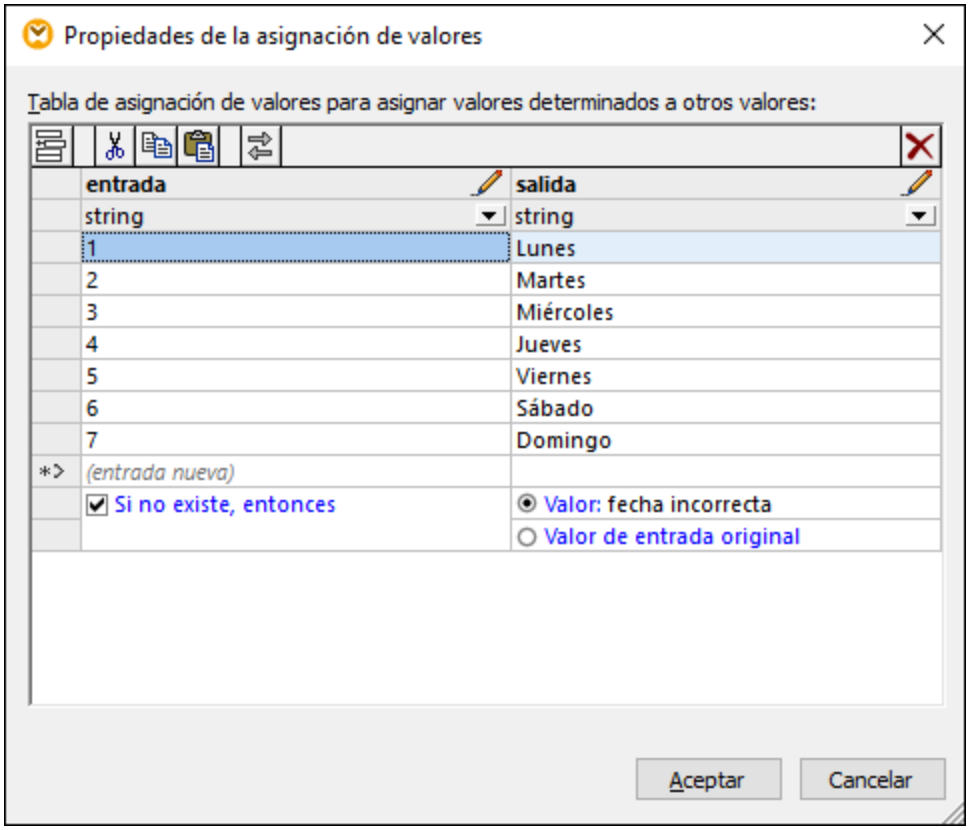
Si la tabla de consulta es enorme ralentizará la ejecución de la asignación. En este caso es mejor utilizar un componente de BD con SQL-Where 437. En estos casos son útiles las bases de datos SQLite, dada su portabilidad. Del lado servidor puede mejorar el rendimiento de las tablas de consulta ejecutando sus asignaciones con MapForce Server o MapForce Server Advanced Edition.

## Crear una asignación 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**.
- 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`623.

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 Ejemplo: reemplazar puestos de trabajo 453.

 © 2019-2025 Altova GmbH

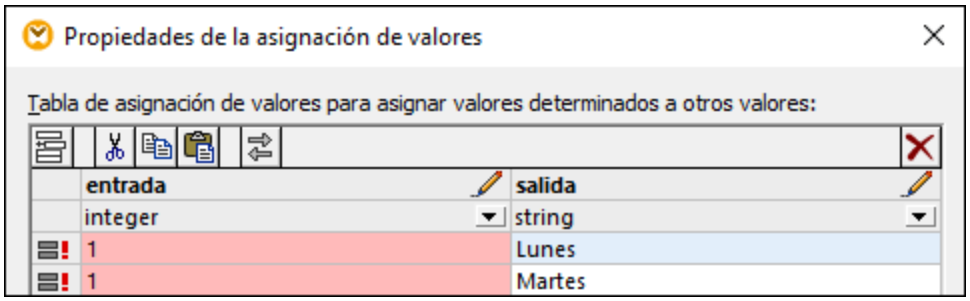
## Rellenar una asignación de valores

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 panel de consulta de BD 294. 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 450.

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:




Para usar una fuente de datos externa para importar datos en el componente de asignación de valores, siga estos pasos:

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** sólo 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 omite. 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.

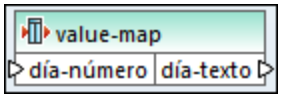
Si quiere copiar o cortar filas para después pegarlas en una posición distinta, primero seleccione la fila en cuestión y después haga clic en el botón **Copiar** (o **Cortar**). También puede copiar o cortar filas que no sean consecutivas. Para ello, mantenga pulsada la tecla **Ctrl** mientras va haciendo 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**.

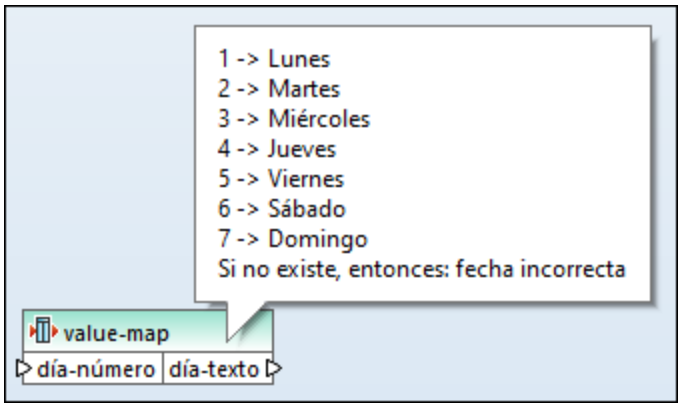
## Cambiar el nombre de parámetros de asignación de valores

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 sólo debe pasar el cursor por encima de la barra del título del componente:





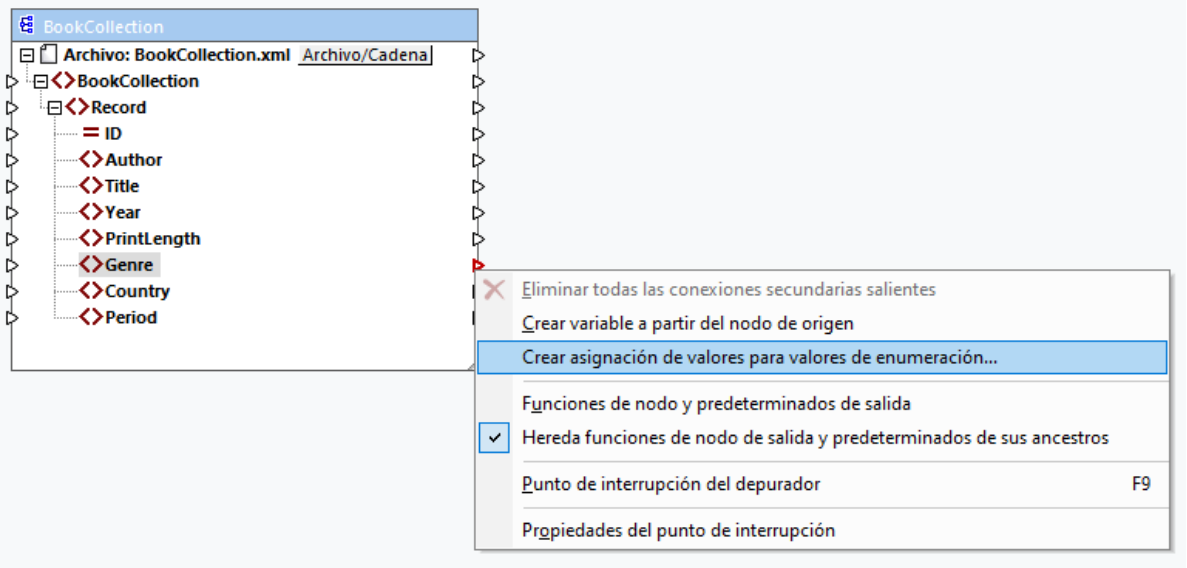
© 2019-2025 Altova GmbH

## Crear una asignación de valores a partir de un valor de enumeración

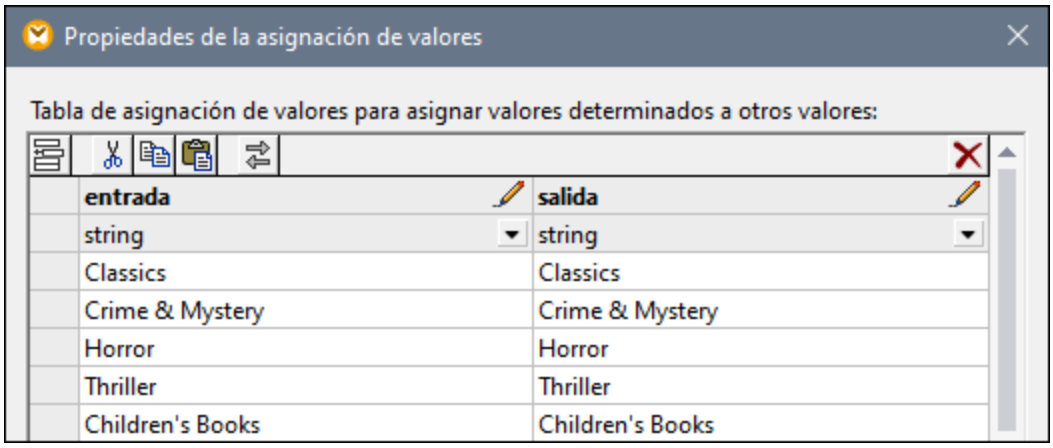
MapForce le permite crear una asignación de valores a partir de nodos con valores de enumeración. Actualmente, esta función es compatible con los componentes XML cuyos nodos tienen facetas de enumeración (*todas las ediciones*) y con los componentes EDI cuyos nodos tienen listas de códigos EDI (*edición Enterprise*). Puede crear este tipo de asignación de valores a partir del conector de entrada o de salida de un nodo, según lo que necesite.

Para crear una asignación de valores a partir de un valor de enumeración, siga estos pasos:

1. Haga clic con el botón derecho en el conector de entrada o salida del nodo (según sus objetivos) para los valores de enumeración desde los que desea crear una asignación de valores. En nuestro ejemplo (*imagen siguiente*) hemos seleccionado el conector de salida del nodo `Genre`.




2. Seleccione **Crear asignación de valores para valores de enumeración** en el menú contextual.
3. Ahora aparecerá el cuadro de diálogo **Propiedades de la asignación de valores**. Tanto la parte de `entrada` como la de `salida` de la asignación de valores se prellenan con los mismos valores de enumeración. Ahora puede revisar y editar los valores según sea necesario. La siguiente imagen muestra la lista de valores `Genre` que tiene inicialmente el archivo XML de origen (`entrada`) y la lista de valores modificados que queremos asignar (`salida`).

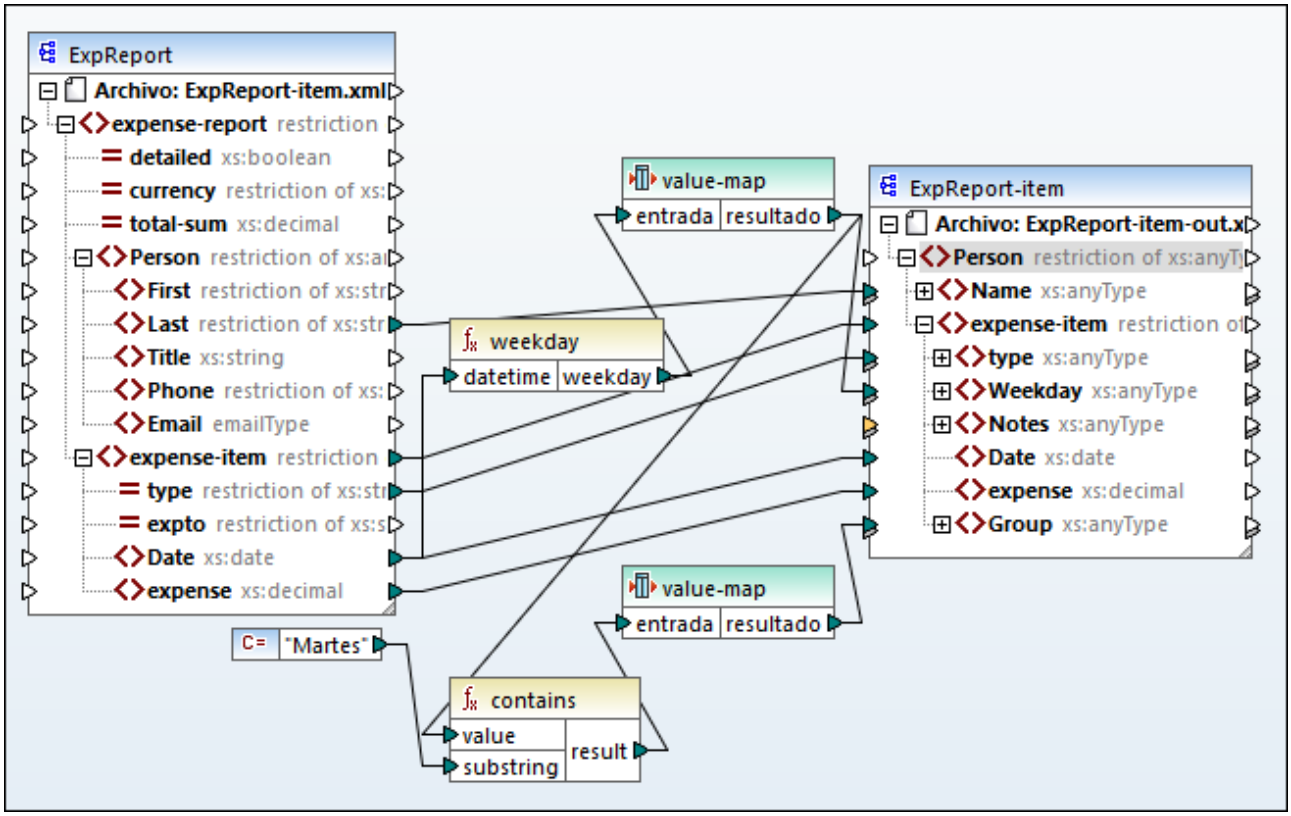



4. Cuando termine de revisar los valores de enumeración, haga clic en **Aceptar**. Con esta acción se añadirá un componente value-map al área de asignación. El componente value-map se conectará automáticamente al nodo con cuyos valores de enumeración se creó la asignación de valores.
5. Conecte el otro parámetro de la asignación de valores con el nodo correspondiente y continúe diseñando su asignación como desee.

## 5.7.1 Ejemplo: reemplazar días de la semana

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\MapForce2025\MapForceExamples\Tutorial\Expense-valmap.mfd**.

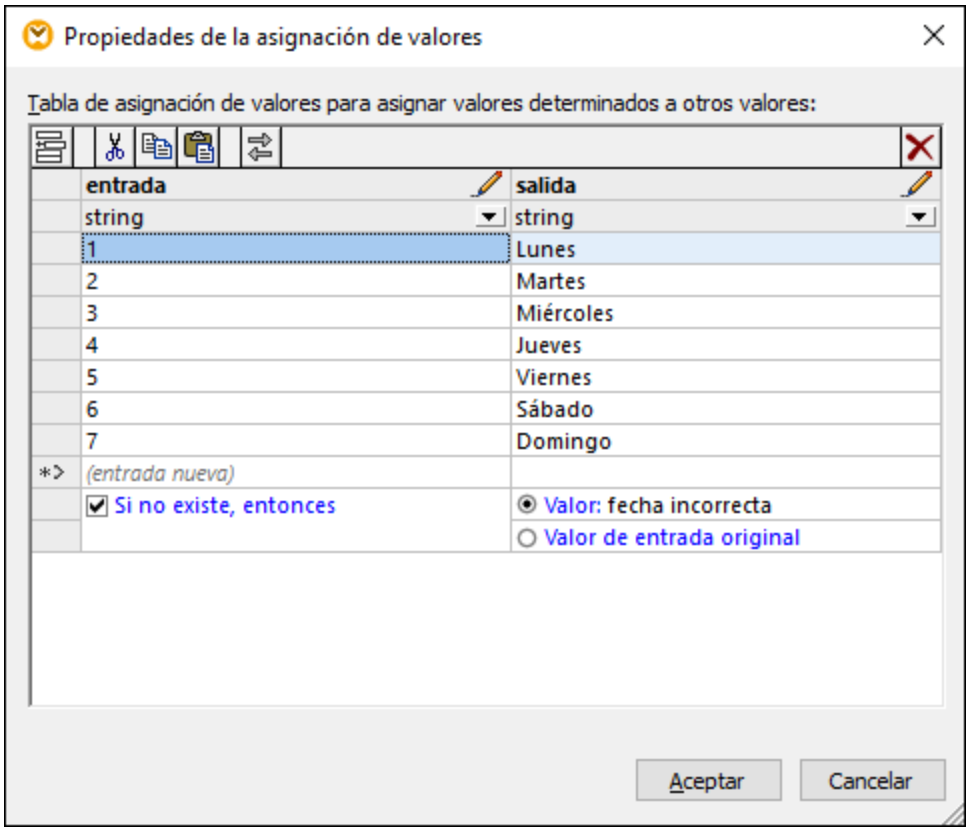
 © 2019-2025 Altova GmbH



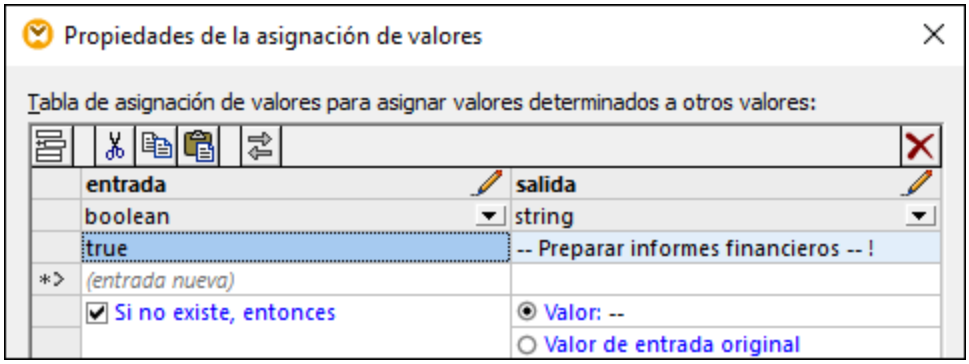

*Expense-valmap.mfd*

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:




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.

 © 2019-2025 Altova GmbH

## 5.7.2 Ejemplo: reemplazar puestos de trabajo

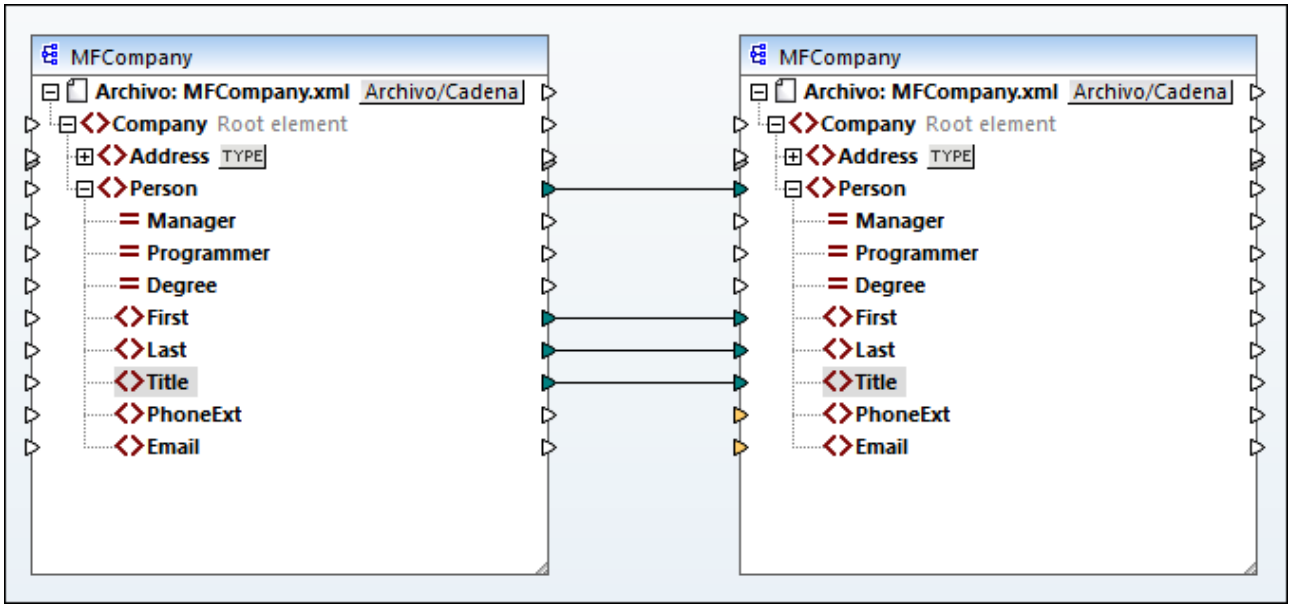
En este ejemplo aprenderá cómo reemplazar valores de elementos específicos en un archivo XML con ayuda de componentes de asignación de valores (es decir, usando una tabla de consulta predefinida).

Puede encontrar el archivo XML necesario para este ejemplo en **<Documentos>\Altova\MapForce2025\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>
```

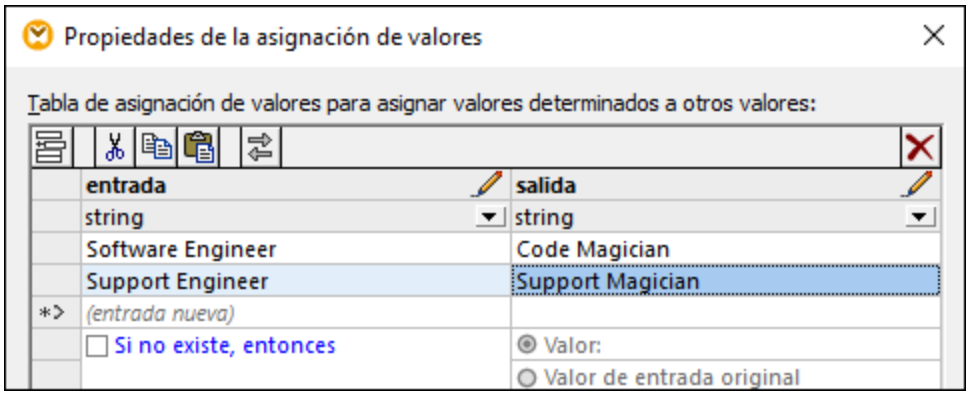
Imagine que necesita reemplazar algunos de esos puestos de trabajo en el archivo XML que hemos mencionado más arriba. En concreto, debe reemplazar "Software Engineer" (Ingeniero de software) por "Code Magician" (Mago del código). También el título "Support Engineer" (Ingeniero de soporte) se debe sustituir 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:



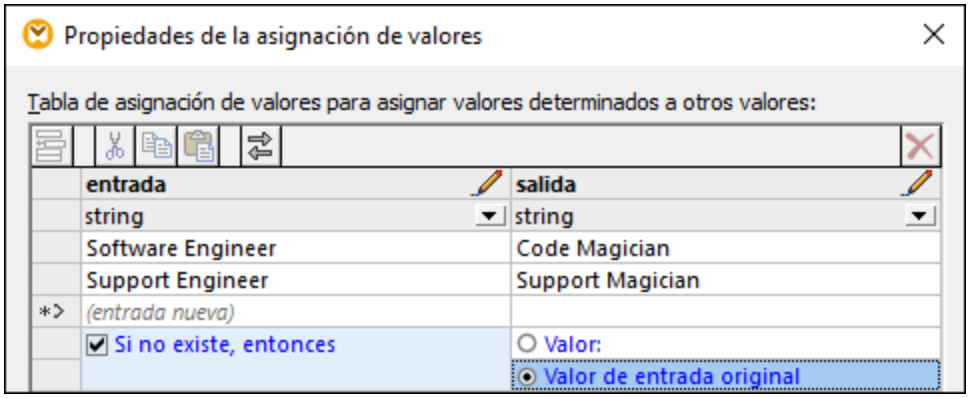

Según la configuración anterior, cada ocurrencia de "Software Engineer" se reemplaza con "Code Magician" y cada ocurrencia de "Support Engineer" se reemplaza con "Support Magician". Tenga en cuenta que no se ha definido la condición **Si no existe, entonces**, por lo que la asignación de valores devuelve 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>
```

 © 2019-2025 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>
```

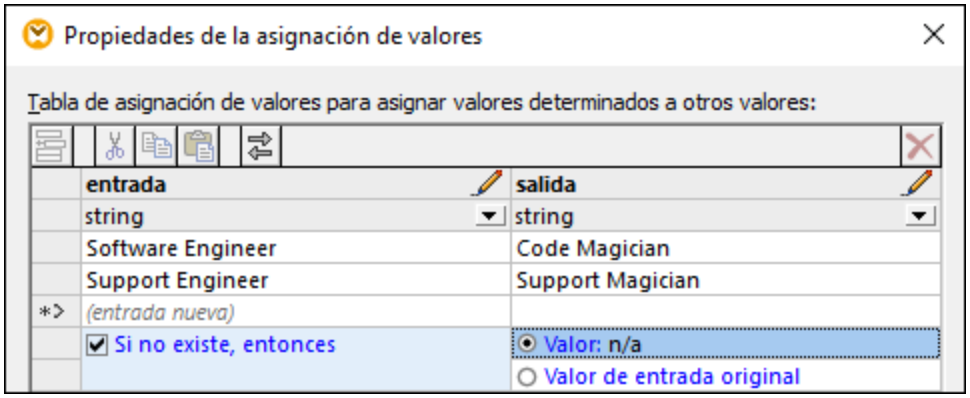
Como acabamos de explicar, si hay nodos vacíos en el resultado generado faltan algunas entradas; por tanto, en el fragmento XML anterior solamente se reemplazó el puesto de Michelle Butler, ya que este estaba en la tabla de consulta. La configuración que ha creado hasta ahora no se ajusta al objetivo 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.

Sólo 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.

 © 2019-2025 Altova GmbH

# 5.8 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 432). Por ejemplo, puede diseñar la asignación para que genere una excepción si el valor de algún elemento de la asignación es superior a un valor predefinido.

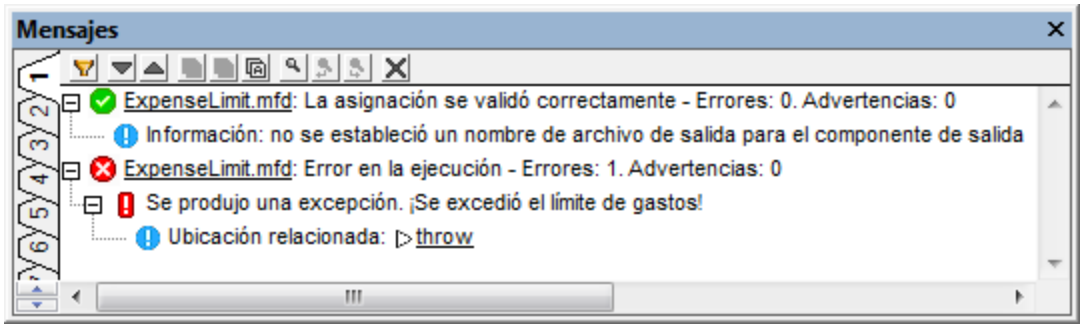
### Para agregar una excepción a la asignación:

1. En el menú **Insertar**, haga clic en **Excepción**.
2. Haga clic en el botón **Insertar excepción** ( ) de la barra de herramientas
3. Conecte la entrada **throw** de la excepción con la salida **on-true** o con la salida **on-false** de un filtro.
4. 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, aparecerá el error «Error de ejecución» en la ventana Mensajes y se abrirá la pestaña correspondiente de XSLT2 o XQuery. La línea del error aparece resaltada 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.

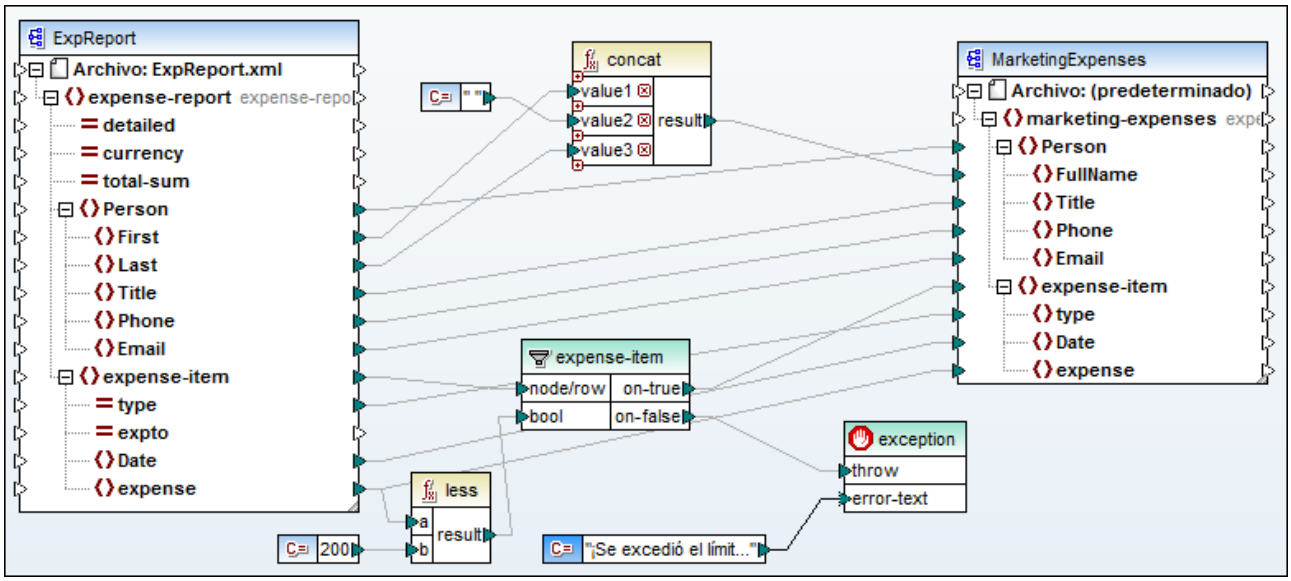
```
c:\codegen\mfx\ExpenseLimit>"C:\Program Files (x86)\Altova\MapForceServer2016\bi
n\MapForceServer.exe" run ExpenseLimit.mfx
Exception was thrown! Expense limit exceeded!
```

- 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.

```
C:\codegen\cs\ExpenseLimit\Mapping\bin\Debug>Mapping.exe
Mapping Application
USER EXCEPTION: Expense limit exceeded!
```

## 5.8.1 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\MapForce2025\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.

## 5.8.2 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.

 © 2019-2025 Altova GmbH

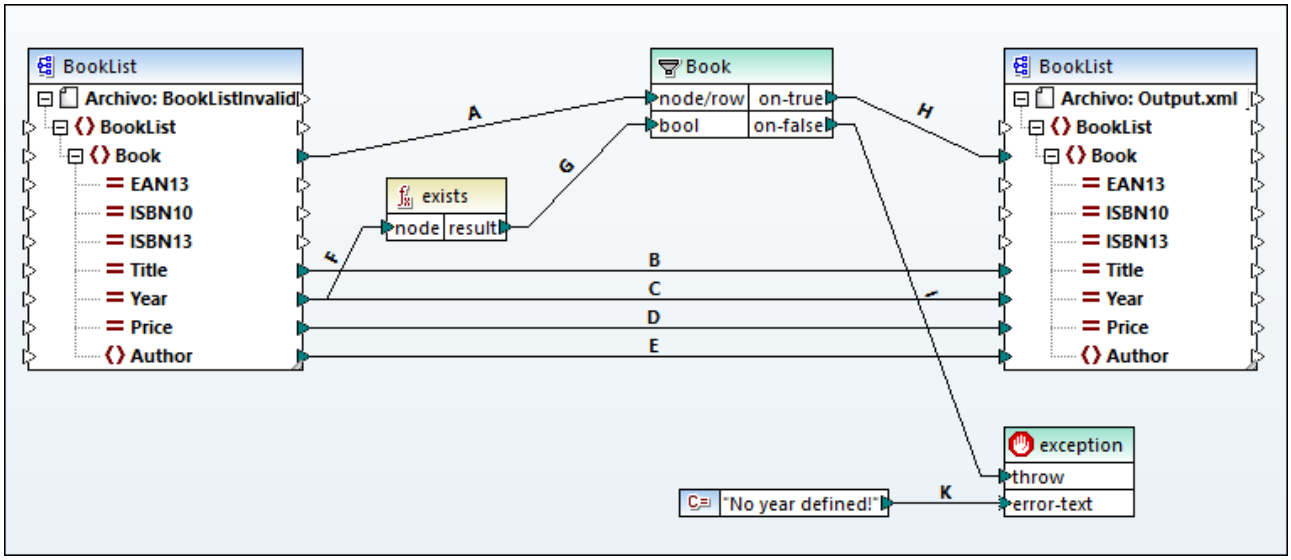
### Para añadir el esquema de origen a la asignación:

1. En el menú **Insertar** haga clic en **Archivo o esquema XML...** y navegue hasta el archivo **<Documentos>\Altova\MapForce2025\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* 432)
- 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\MapForce2025\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.


© 2019-2025 Altova GmbH

# 5.9 Funciones

En MapForce, puede utilizar las siguientes categorías de funciones para transformar los datos:

- **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 544.

- **Funciones de nodo y valores predeterminados (*Professional y Enterprise Edition*):** 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 469.

- **Funciones definidas por el usuario:** son funciones de MapForce que puede crear usted mismo usando varios tipos de componentes nativos y las funciones integradas que ya están disponibles en MapForce (véase Funciones definidas por el usuario 485).

- **Funciones personales:** 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 507, Importar funciones XQuery 1.0 personales 514 e Importar bibliotecas Java y .NET 518 personales.

- **Llamadas a servicios web (*Enterprise Edition*):** MapForce le permite hacer una llamada a un servicio web previamente definido (un servicio basado en WSDL o una API HTTP genérica).

**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 526.

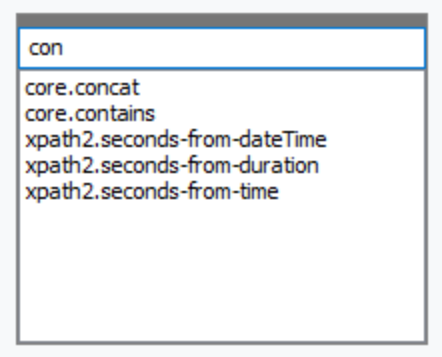
## 5.9.1 Fundamentos de las funciones

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 19.

## 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 544. Hay varios métodos para agregar una función a una asignación:

- Pulse y mantenga pulsada 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 (*ver 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.

## 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

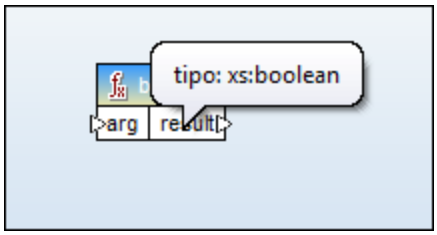
 © 2019-2025 Altova GmbH

flecha y deshabilite la opción *Incluir descripción en la búsqueda*. Para cancelar la búsqueda pulse la tecla **Esc** o haga clic en el icono ✖.

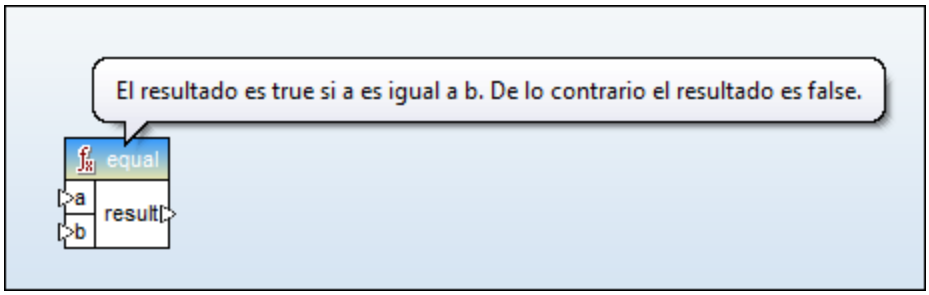
Para buscar todas las instancias de una función dentro de la asignación activa basta con hacer clic con el botón derecho en la función en la ventana **Bibliotecas** y elegir el comando **Buscar todas las llamadas** en el menú contextual. El resultado de la búsqueda aparece en la ventana **Mensajes**.

## 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.

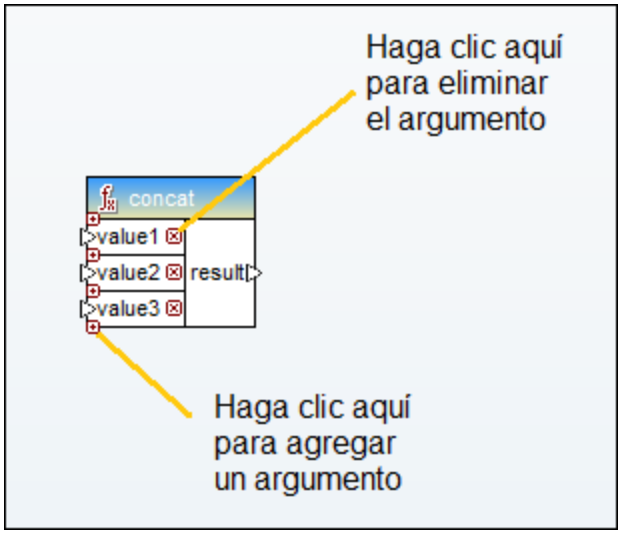



Para ver la descripción de una función mueva el ratón sobre el encabezado de la función (*imagen siguiente*). Asegúrese de que el botón (**Ver información rápida**) está habilitado.




## 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** 626. 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 (*véase más abajo*). Si mueve una conexión al símbolo ⊞ se añade otro parámetro que se conecta a este.




## 5.9.2 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 487.
- 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 507.
- 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 514.
- Archivos Java .class y bibliotecas .NET .dll que se pueden importar en MapForce como se describe en Importar bibliotecas Java y .NET personales 518.

**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 526.

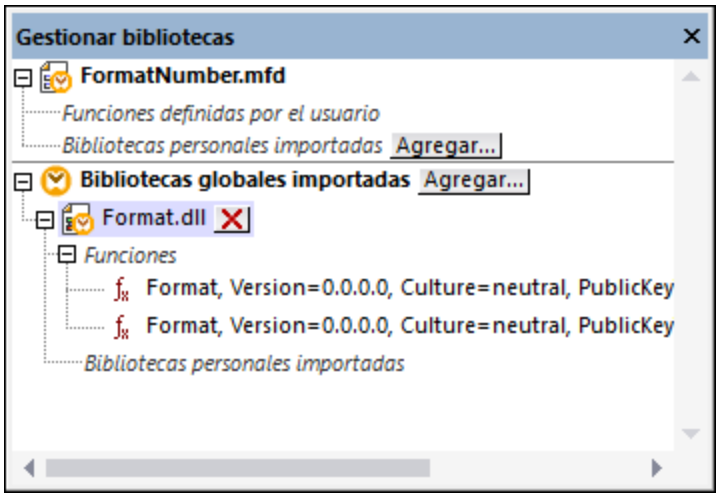
### Ventana Gestionar bibliotecas

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:

 © 2019-2025 Altova GmbH

- 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 sólo 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 |
| **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 sólo 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 |

## 5.9.2.1 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.

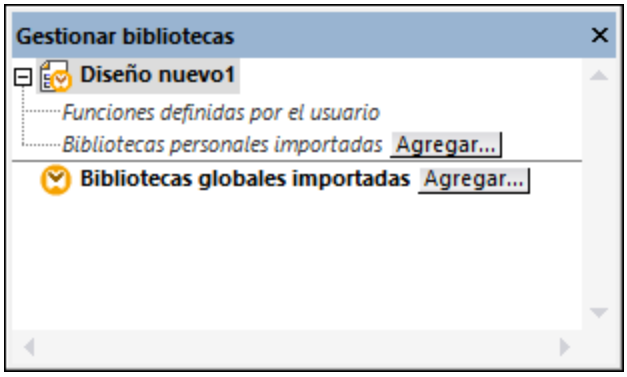
Las importaciones a nivel local se hacen a nivel del archivo de asignación. Por ejemplo, imagine que está trabajando en la asignación **A.mfd** y decide importar todas las funciones definidas por el usuario de la asignación **B.mfd**. En este caso la asignación **B.mfd** se considera como una biblioteca importada localmente

 © 2019-2025 Altova GmbH

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 23. 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, sólo 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.

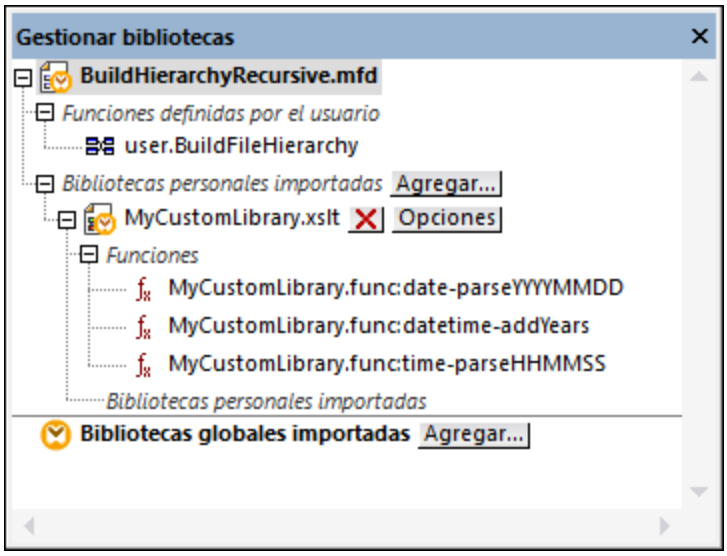
## 5.9.2.2 Rutas relativas de acceso a bibliotecas

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 466.

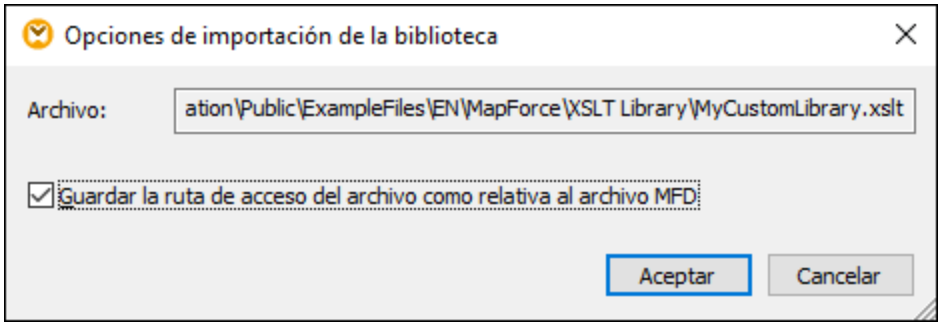
Sólo se pueden configurar como relativas las rutas de acceso a bibliotecas que se hayan importado de forma *local* en una asignación. Si se importa una asignación *globalmente*, es decir, a nivel del programa, su ruta será siempre absoluta.

**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.




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* 44).

 © 2019-2025 Altova GmbH

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 46.

## 5.9.3 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 474. 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* 479.

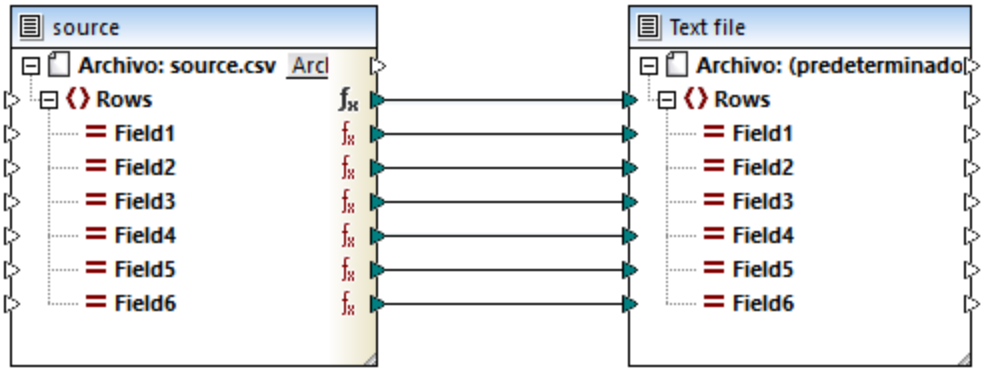
Por ejemplo, puede indicar a MapForce que:

- 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.

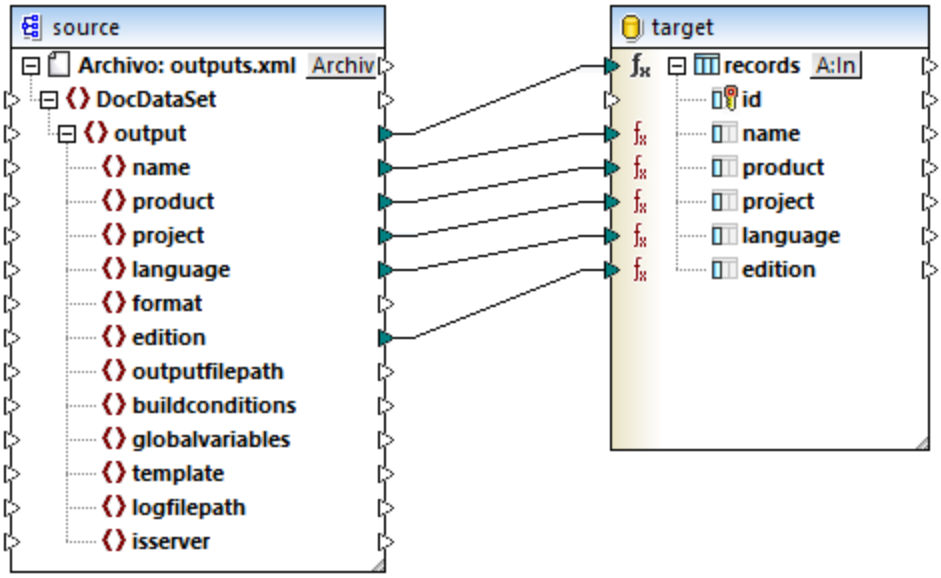
## Lado de salida vs. de entrada.

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:

- 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*).

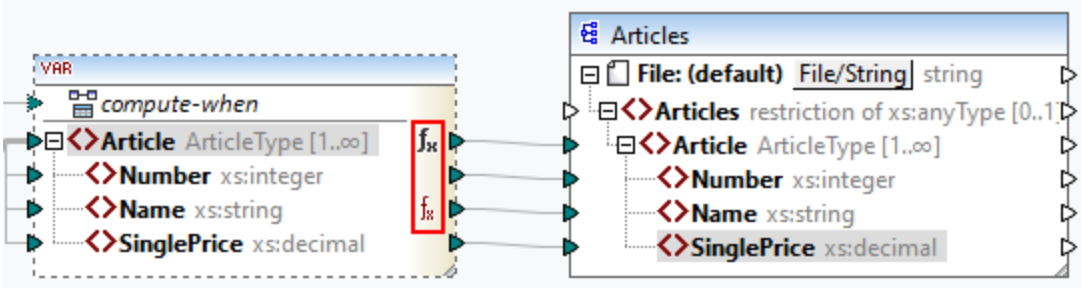



- Inmediatamente *antes* de que se escriban datos en el componente de destino (y después de que haya finalizado todo el procesamiento intermedio), lo cual quiere decir que la función o el valor predeterminado se definen en el lado de entrada del componente de destino (*véase el ejemplo siguiente*).

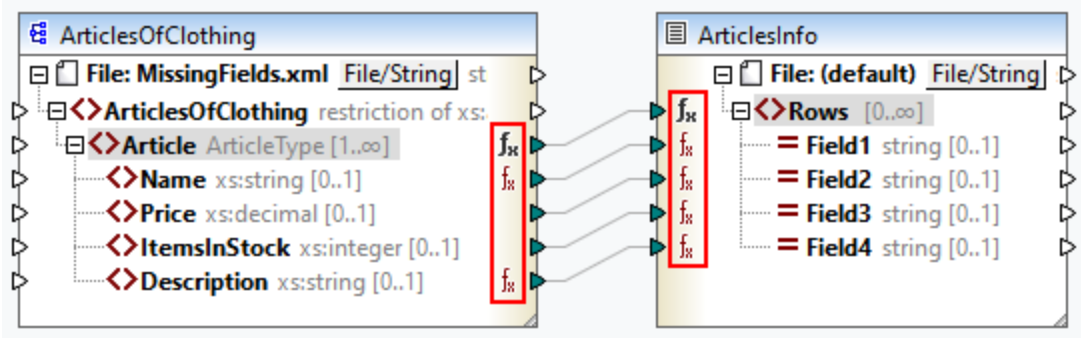



 © 2019-2025 Altova GmbH

- 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 471
- Casos de uso 474
- Suministrar metadatos del nodo a funciones de nodo 483


## 5.9.3.1 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:

- 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 470.

- 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 478.

- 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 52.

- 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 483.

- 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.

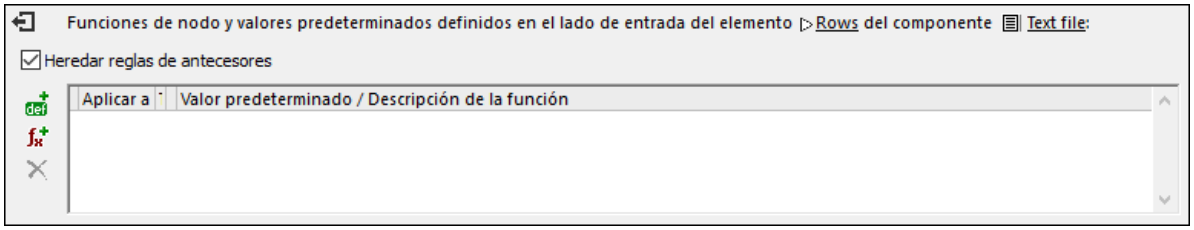
## Crear una regla

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

 © 2019-2025 Altova GmbH

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 474.

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.

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

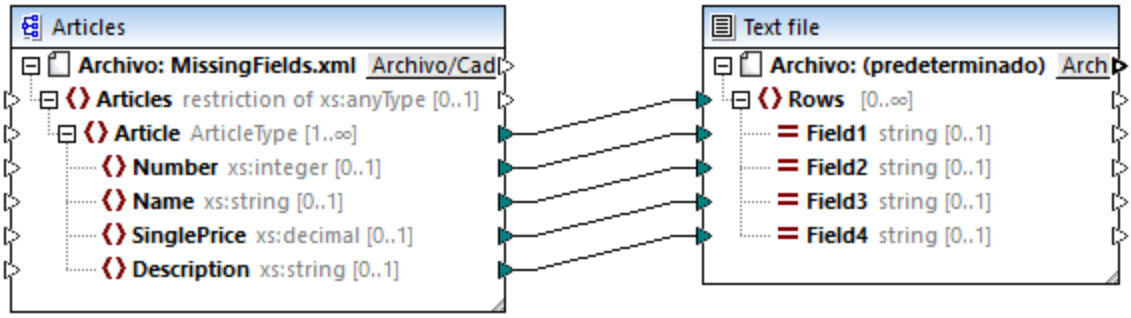
Para que comprenda mejor la diferencia entre reglas definidas y aplicadas, MapForce ofrece las siguientes ayudas visuales en la asignación (*tabla siguiente*).

| 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. |

## 5.9.3.2 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.

 © 2019-2025 Altova GmbH




Algunos de los nodos del archivo de origen están vacíos. En este punto la asignación tiene este aspecto:

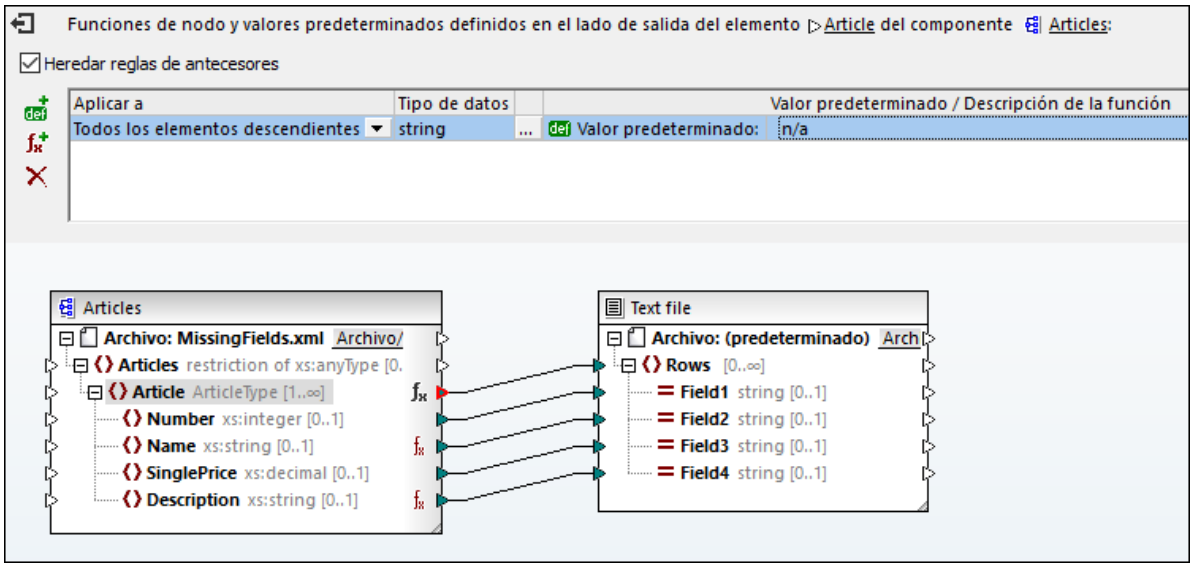
| | | | |
|---|---|---|---|
| T-Shirt | 25,3 | 20 | Available in black, blue, and red |
| Shirt | 70,3 | 60 | |
| Pants | | | Limited stock |
| Jacket | 57,5 | 40 | |

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*).




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*).

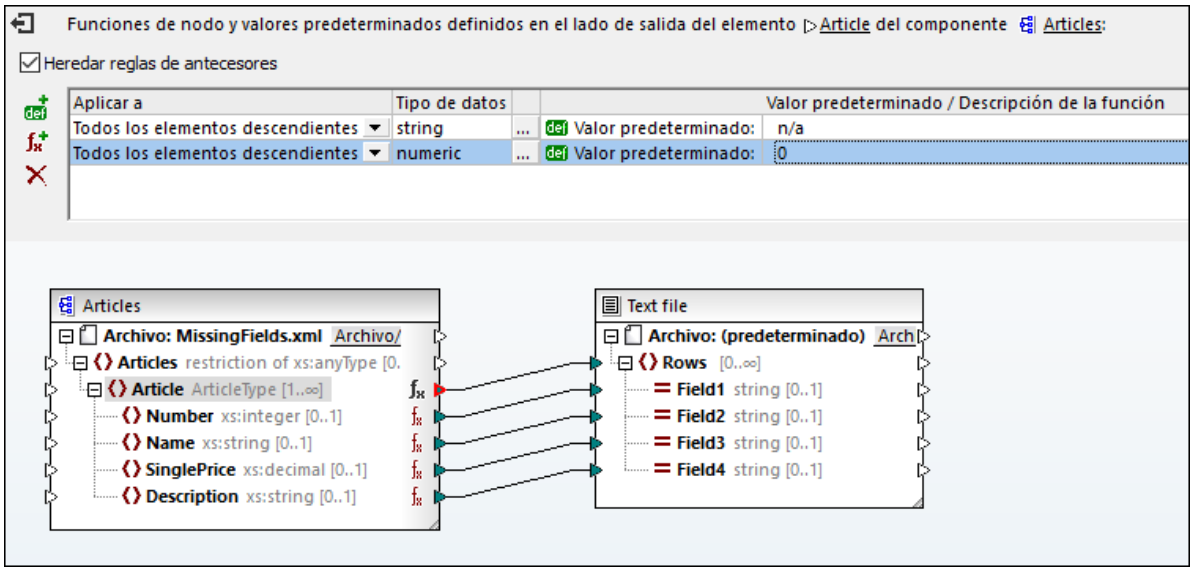
| | | | |
|---|---|---|---|
| T-Shirt | 25,3 | 20 | Available in black, blue, and red |
| Shirt | 70,3 | 60 | n/a |
| Pants | | | Limited stock |
| Jacket | 57,5 | 40 | n/a |

## 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*).

 © 2019-2025 Altova GmbH



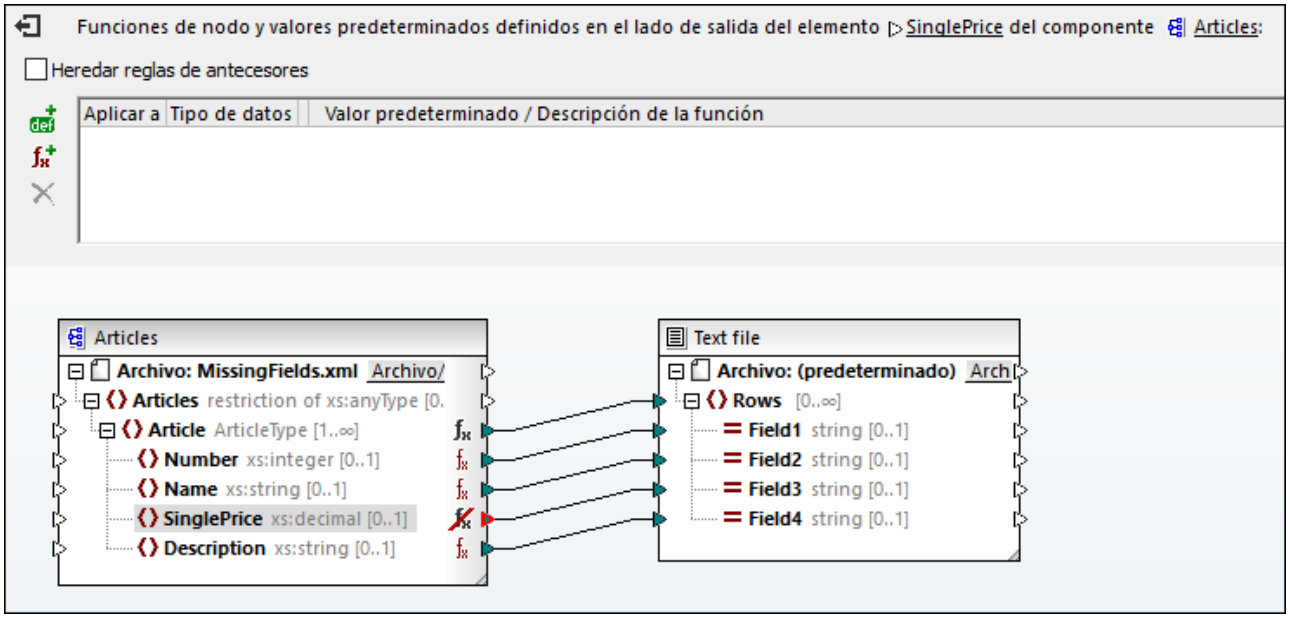

El resultado ahora tiene este aspecto (*observe las partes resaltadas en amarillo*):

| | | | |
|---|---|---|---|
| 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*).



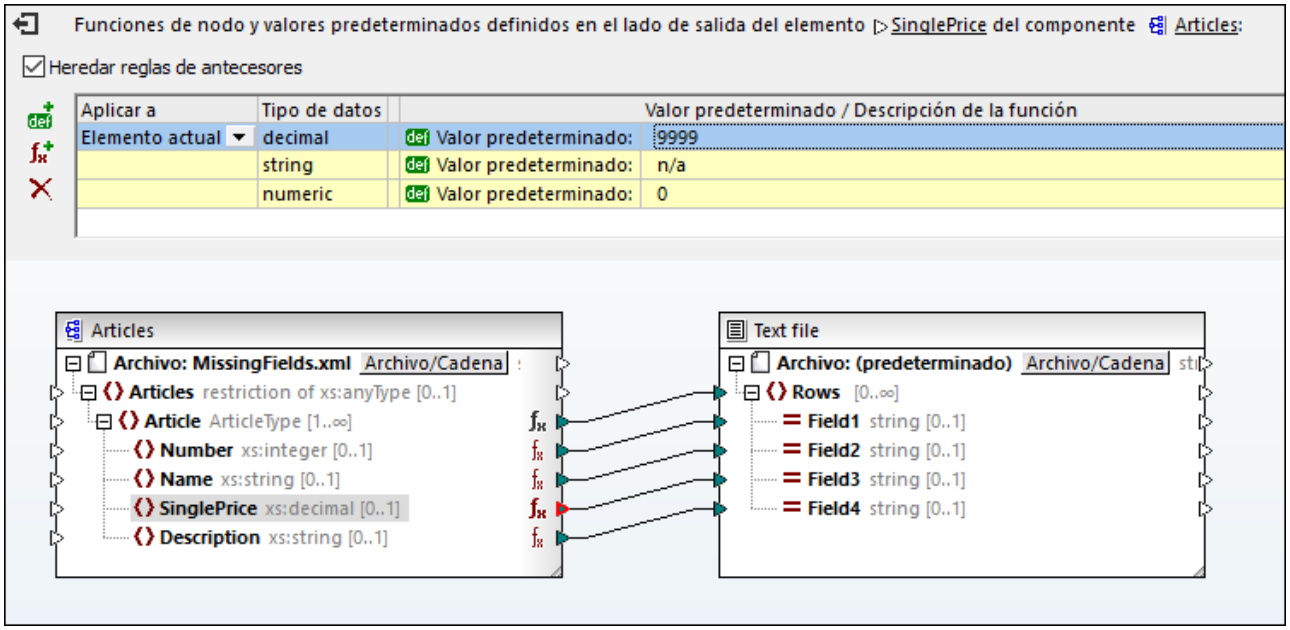

En el resultado siguiente puede ver que el valor vacío del elemento `Price`, para el que hemos bloqueado la regla, se ha asignado a `Field2` (*observe las partes resaltadas*).

| | | | |
|---|---|---|---|
| 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.

 © 2019-2025 Altova GmbH




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*):

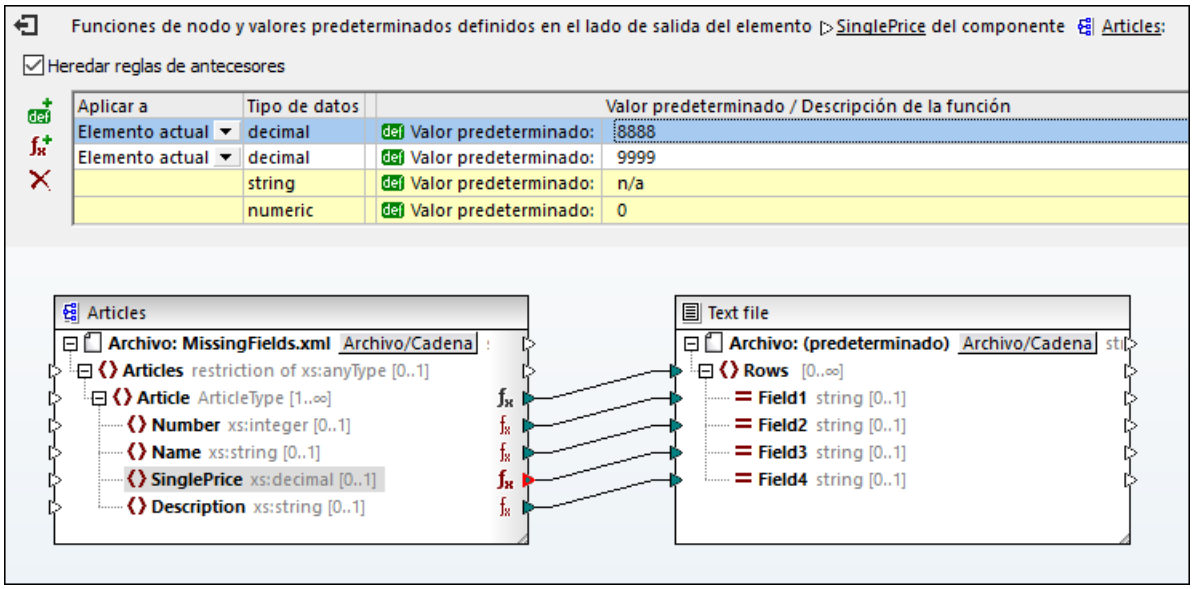
| | | | |
|---|---|---|---|
| T-Shirt | 25,3 | 20 | Available in black, blue, and red |
| Shirt | 70,3 | 60 | n/a |
| Pants | 100 | 0 | Limited stock |
| Jacket | 57,5 | 40 | n/a |

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.

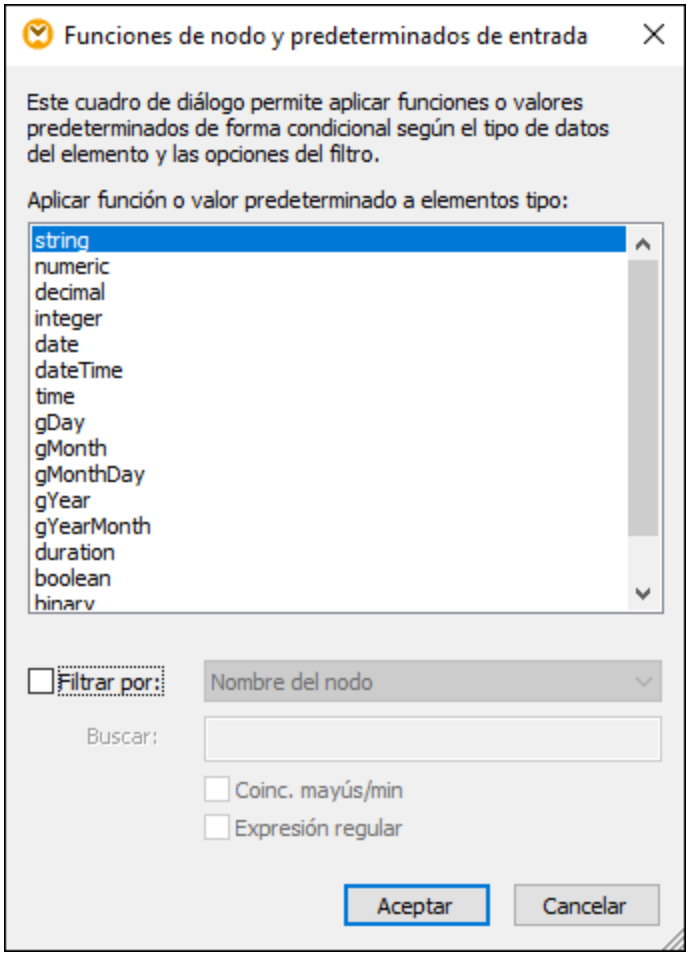



En este punto todavía no hemos filtrado los nodos y el resultado tiene el siguiente aspecto (*observe las partes resaltadas*):

| | | | |
|---|---|---|---|
| T-Shirt (XXS, L-XXL)<br>Shirt (XXS, L-XXL)<br>Pants (XXS, L-XXL)<br>Jacket (XXS, L-XXL) | 25,3<br>70,3<br><br>57,5 | 20<br>60<br><br>40 | Available in black, blue, and red (XXS, L-XXL)<br><br>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.

 © 2019-2025 Altova GmbH




Ahora el resultado tiene este aspecto:

| | | | |
|---|---|---|---|
| T-Shirt (XXS, L-XXL)<br>Shirt (XXS, L-XXL)<br>Pants (XXS, L-XXL)<br>Jacket (XXS, L-XXL) | 25,3<br>70,3<br><br>57,5 | 20<br>60<br><br>40 | Available in black, blue, and red<br><br>Limited stock |

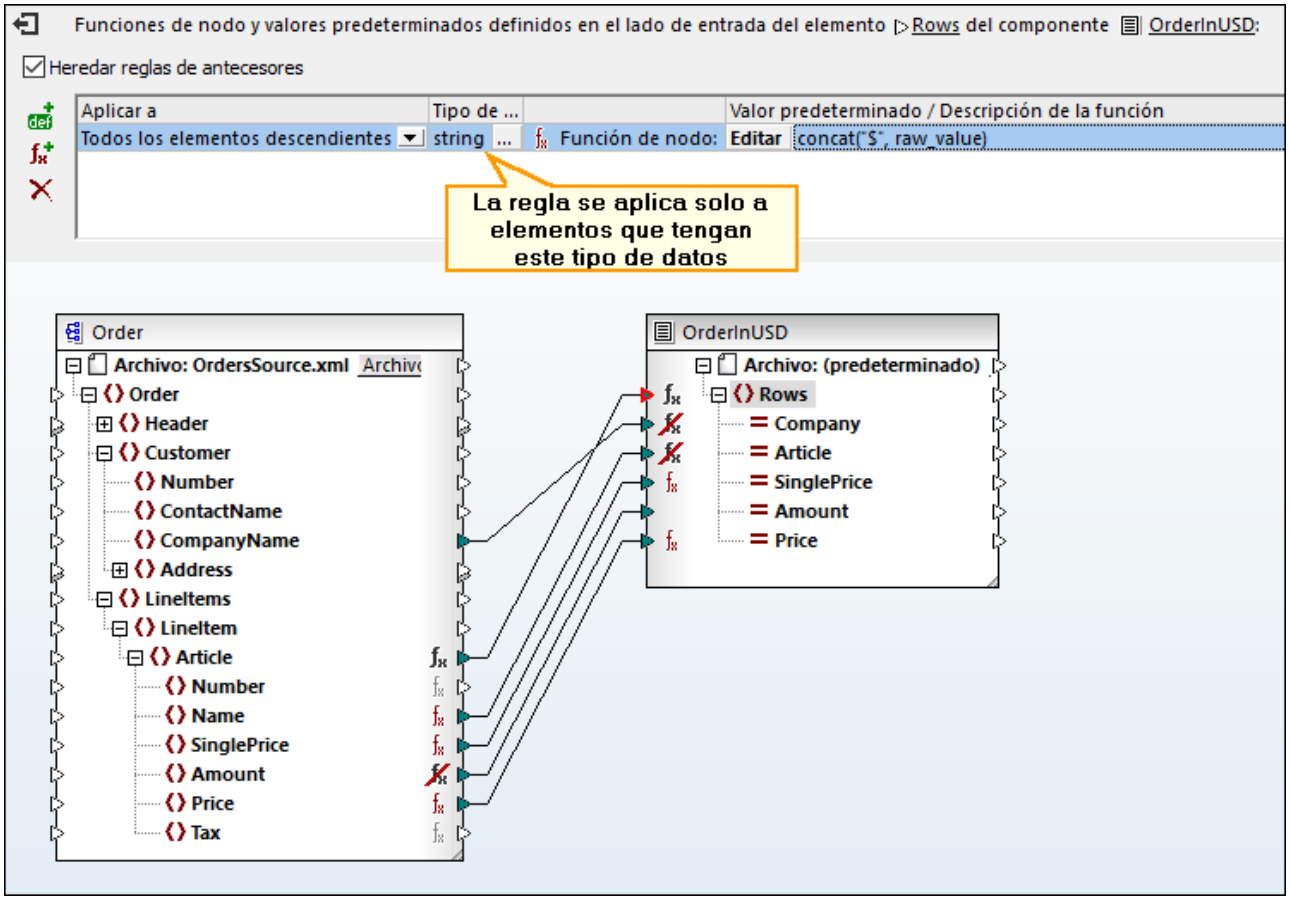
*Filtrar por tipo de nodo*
Esta opción permite acotar el tipo de datos que seleccionó en el cuadro de diálogo **Funciones de nodo y valores predeterminados**. Por ejemplo, si ha elegido aplicar la función de nodo a nodos numéricos, puede indicar un subtipo (p. ej.: *Filtrar por/Nombre del nodo:* `xs:decimal`).

*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 [540]. 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 sólo 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 |

 © 2019-2025 Altova GmbH

**Nota:** los valores predeterminados no afectan a los nodos sin conexión de los componentes JSON.

### 5.9.3.3 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. |

**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 472 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**.

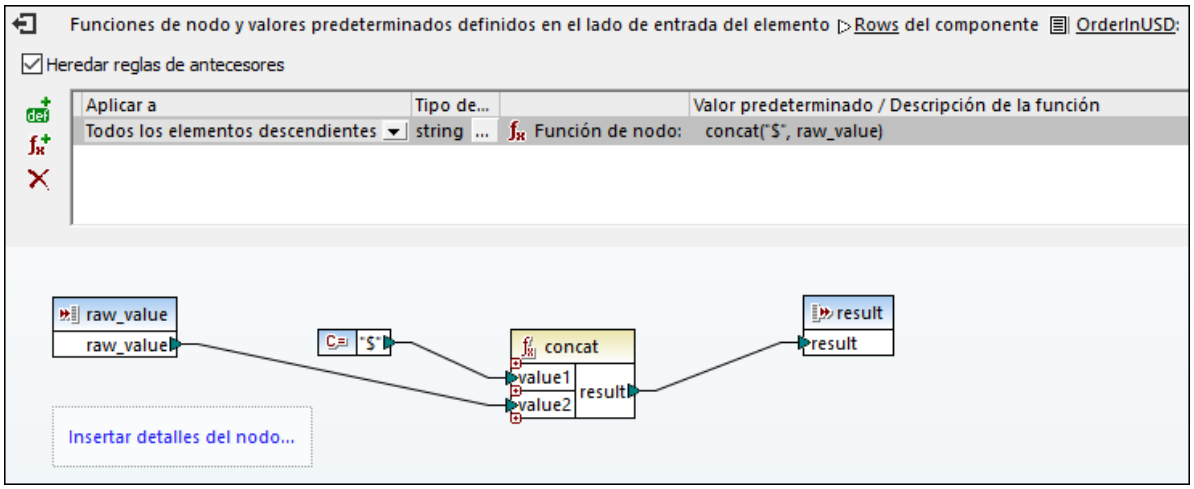
*Metadatos no compatibles con nodos*
Si un parámetro de metadatos no es compatible con un nodo, puede indicar a MapForce que devuelva una secuencia vacía o que no aplique la función de nodo (puede encontrar ambas opciones en el cuadro de diálogo **Insertar componente de entrada con nodo específico**). 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` 623 y `exists` 597) 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\MapForce2025\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 472 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 461.

 © 2019-2025 Altova GmbH




**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
```

## 5.9.4 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:

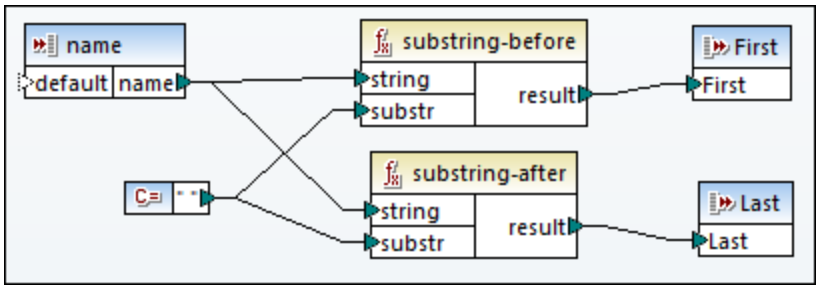
- Se pueden reutilizar dentro de la misma asignación o en asignaciones distintas.

- 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 503.
- 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) 489. Las funciones definidas por el usuario recursivas 497 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 487


© 2019-2025 Altova GmbH

- Parámetros en funciones definidas por el usuario 492
- Ejemplo: búsqueda recursiva 497
- Implementación de la búsqueda 503


## 5.9.4.1 Funciones básicas definidas por el usuario

En esta sección explicamos cómo crear, importar, editar, copiar, pegar y eliminar funciones definidas por el usuario.

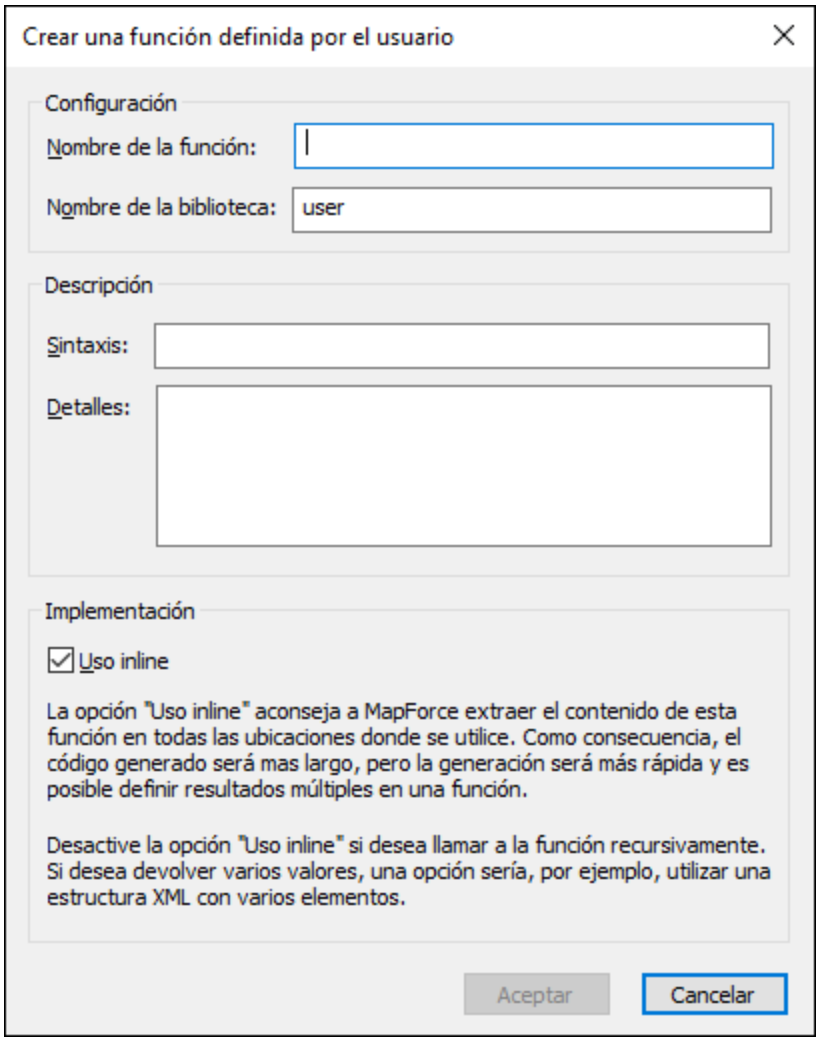
### Crear una función definida por el usuario

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 492.

*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*).




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, 09), 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 23). 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

 © 2019-2025 Altova GmbH

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 89 en un parámetro. | Las funciones estándar permiten establecer un contexto de prioridad en un parámetro. |
| Las funciones definidas por el usuario inline afectan y se ven afectadas por el contexto de la asignación | Las funciones definidas por el usuario normales tienen su propio contexto local. |

| Funciones inline (borde discontinuo) | Funciones regulares (borde sólido) |
|---|---|
| de datos que llama a la función definida por el usuario. | |

**Nota:** cambiar el tipo de una función definida por el usuario de inline a estándar y viceversa puede afectar el contexto de la asignación 82, lo que a su vez puede afectar al resultado de la asignación.

## Importar y llamar a funciones definidas por el usuario

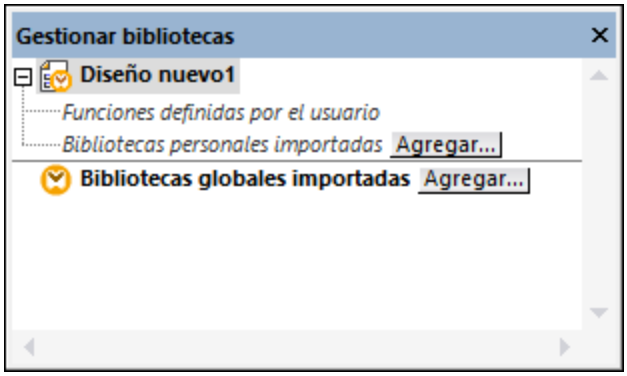
Una vez haya creado una función definida por el usuario puede llamarla desde la asignación en que la creó o desde cualquier otra.

*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 23. 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**.

 © 2019-2025 Altova GmbH

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 464.

*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.

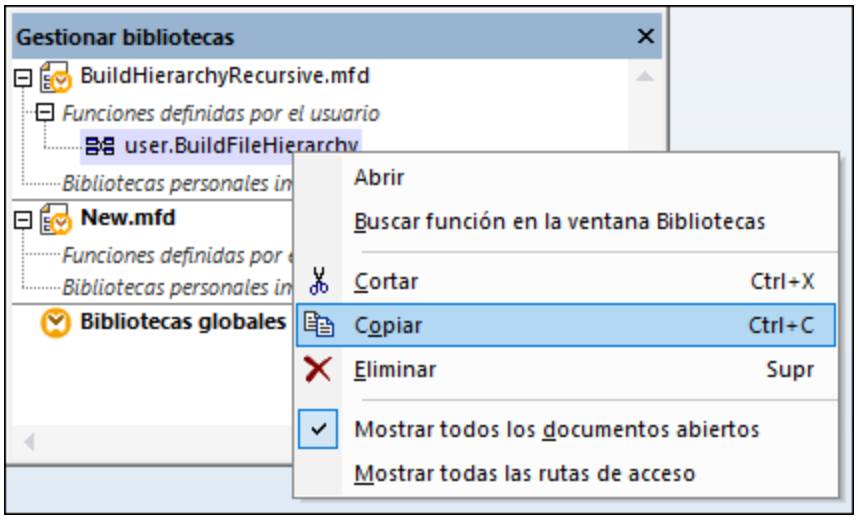
Para volver a la asignación principal haga clic en el botón , que está en la esquina superior izquierda de la ventana de la asignación.

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 464.
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 44.
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**.




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:

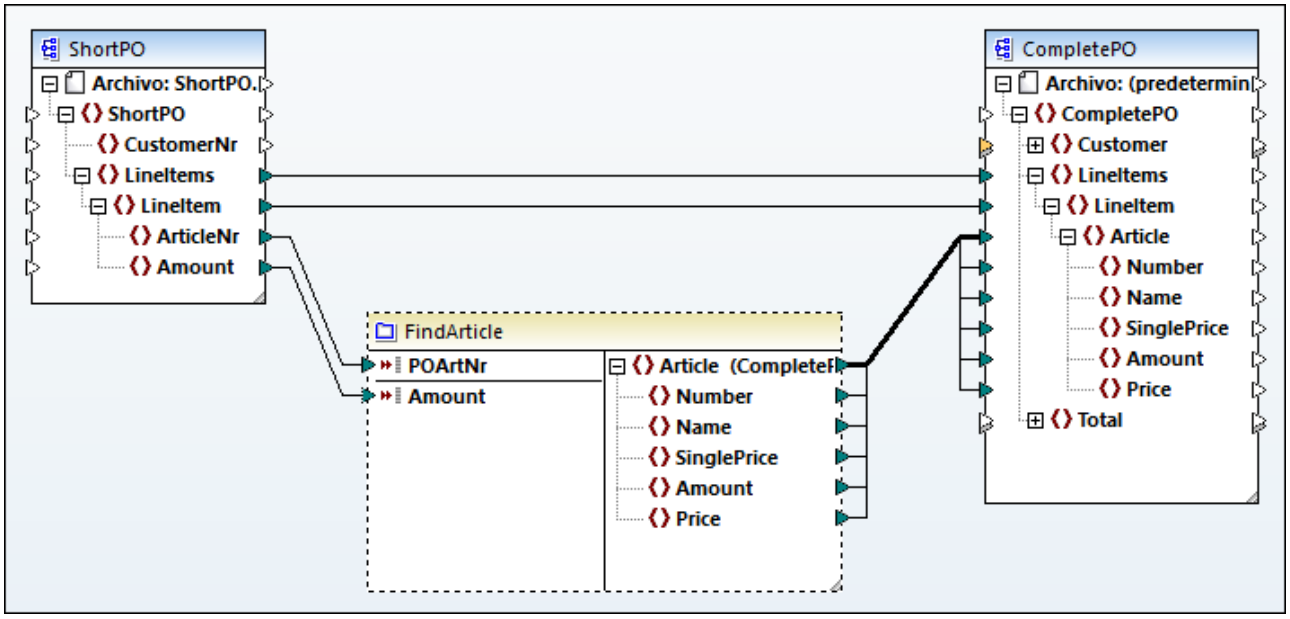
1. En la asignación haga doble clic en la barra del título de la función en cuestión.
2. Haga clic en el botón ERASE, en la parte superior derecha de la ventana Asignación.
3. Si la función se usa en la asignación abierta actualmente, MapForce le preguntará si quiere reemplazar todas las instancias con 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.

## 5.9.4.2 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 493. 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.

 © 2019-2025 Altova GmbH




*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

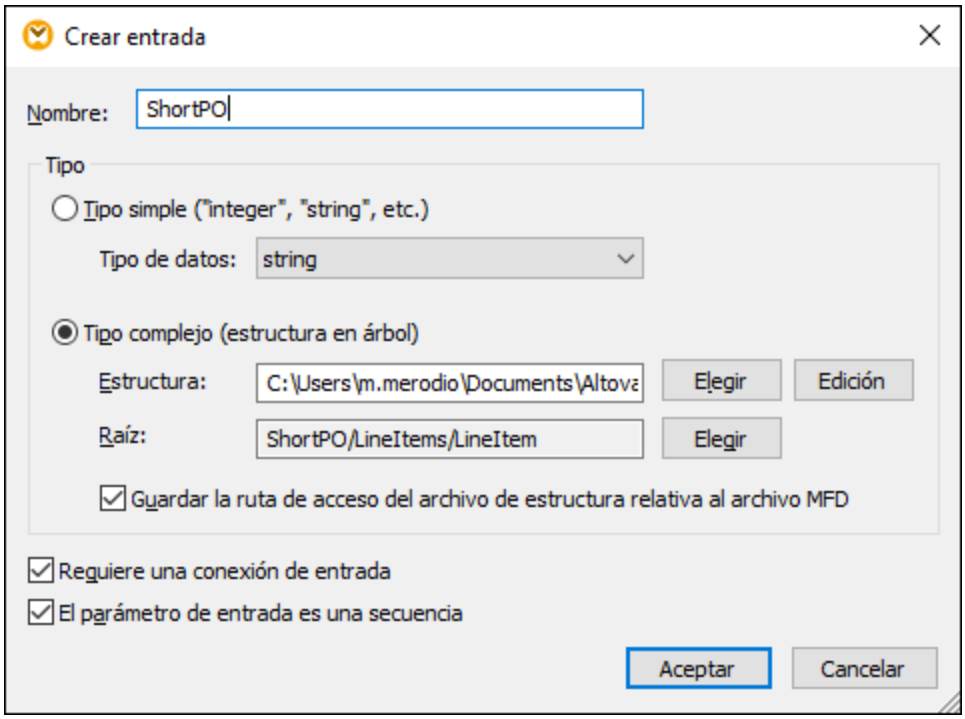
*MapForce Enterprise Edition*
- Estructura de XML Schema
- Estructura de BD
- Estructura EDI
- Estructura FlexText
- Estructura de JSON Schema

*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 487 o abra una que ya exista 490.
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.

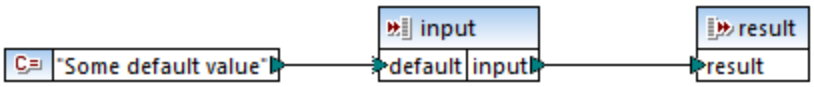
Si la casilla *Guardar ruta de acceso de la estructura como relativas al archivo MFD* está marcada, la ruta absoluta de la estructura pasará a ser relativa a la asignación actual cuando la guarde. Para más información

 © 2019-2025 Altova GmbH

consulte el apartado relativa y ruta absoluta 44. Las casillas *Requiere una conexión de entrada* y *El parámetro de entrada es una secuencia* se explican en los siguientes apartados.

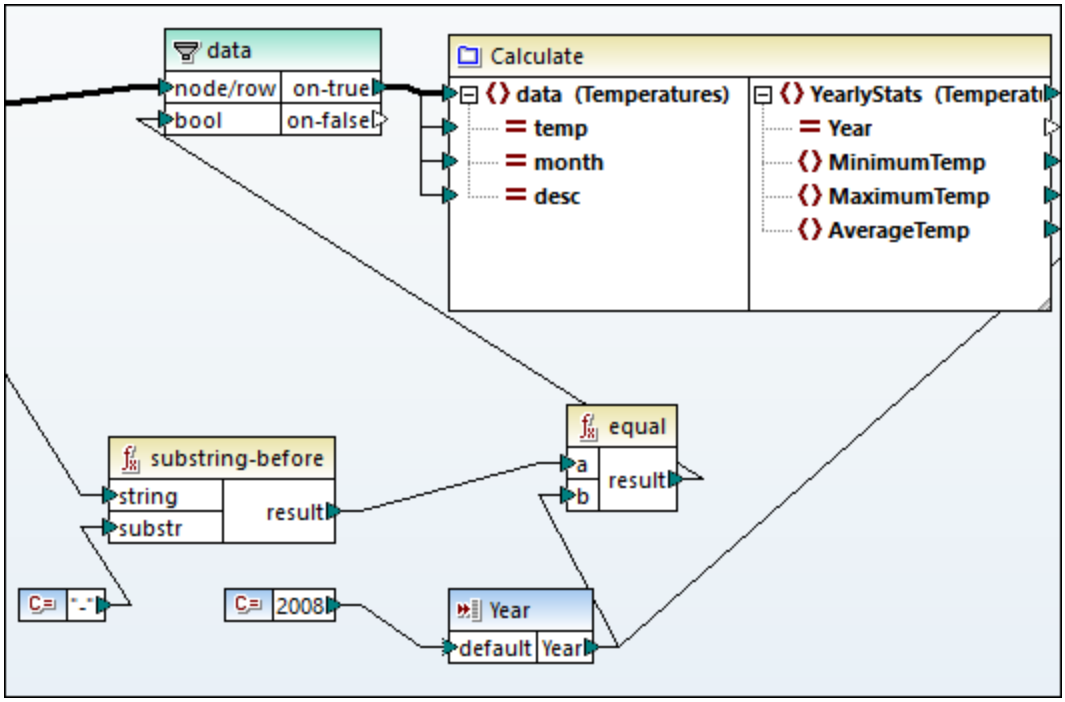
*Requiere una conexión de entrada*
Para hacer que un parámetro sea obligatorio en una función definida por el usuario, marque la casilla *Requiere una conexión de entrada* (*véase el cuadro de diálogo de más arriba*). Si desmarca esta casilla, el parámetro se convierte en opcional y aparece con un borde 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 sólo 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.

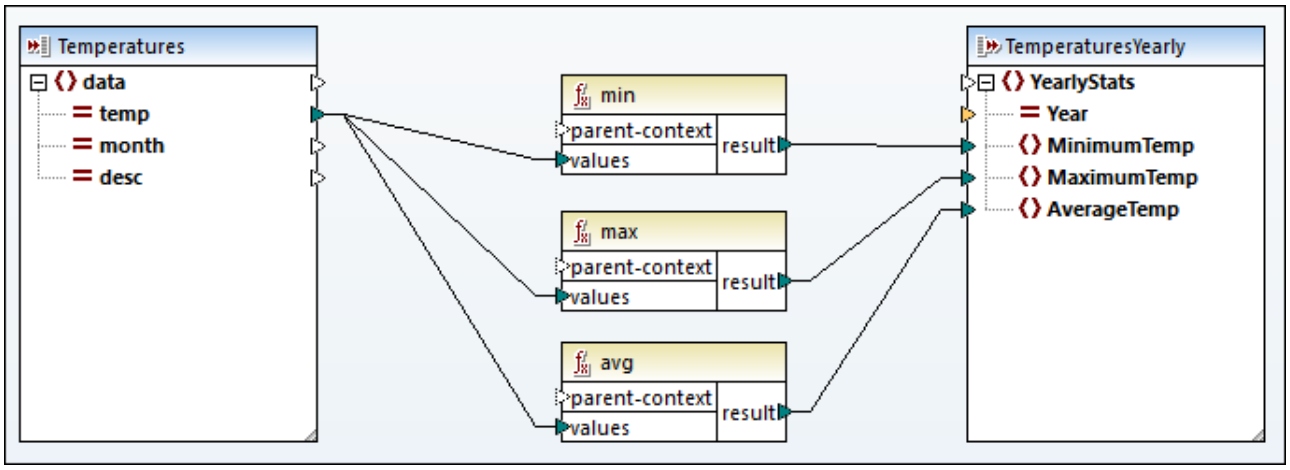



*El componente de entrada es una secuencia*
Puede decidir si un parámetro de una función se debe tratar como un valor único (opción predeterminada) o como una secuencia. Una secuencia es un rango de cero o más valores. Una secuencia puede ser útil si la función definida por el usuario espera que los 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 sólo se habilita para funciones definidas por el usuario estándar 489.

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.




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.

 © 2019-2025 Altova GmbH

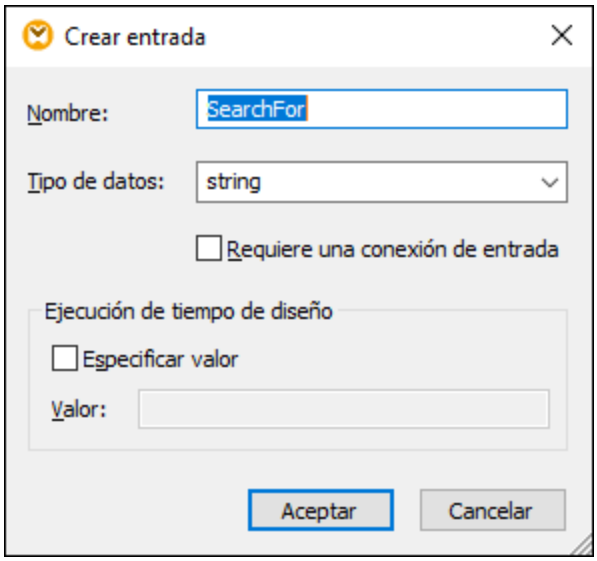
- Si conecta una secuencia vacía a un parámetro que no sea una secuencia, no se llama a la función. Esto puede ocurrir si la estructura de origen tiene elementos opcionales o si una de las condiciones de un filtro no devuelve coincidencias. Para evitarlo, use la función `substitute-missing` 623 antes 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.

### 5.9.4.3 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.

## Archivo de origen

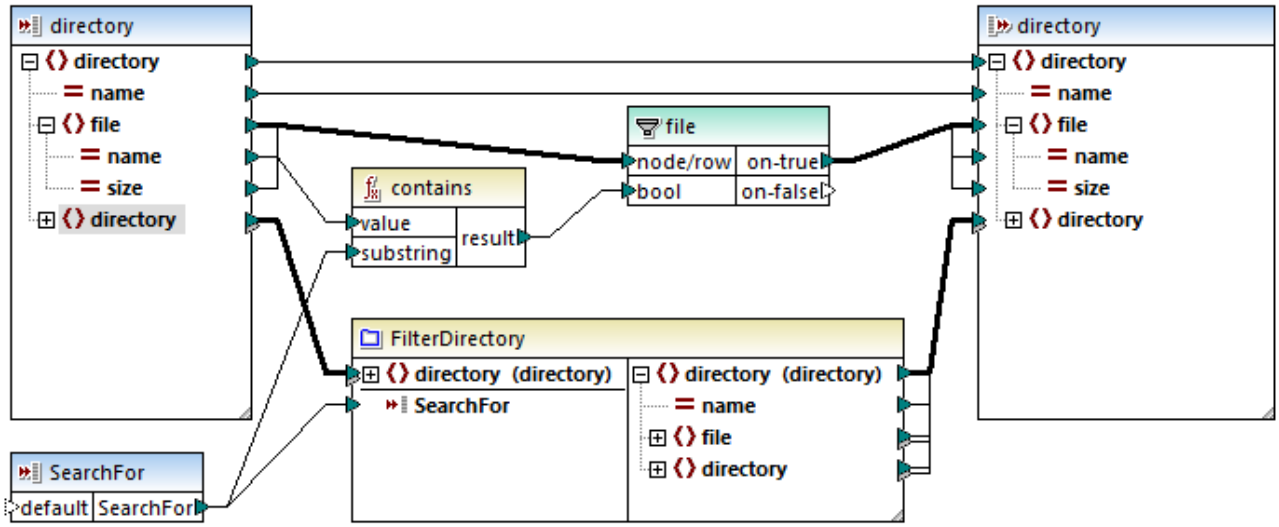
Más abajo puede ver un extracto del archivo XML de origen (**`Directory.xml`**) que contiene información sobre los archivos y directorios. Observe que los archivos de esta lista tienen extensiones distintas (e.g., **`.xml`**, **`.dtd`**, **`.sps`**). Según el esquema (**`Directory.xsd`**), el elemento `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[489].


© 2019-2025 Altova GmbH




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>
```

### 5.9.4.4 Contexto de las funciones defindas por el usuario

Las funciones definidas por el usuario son funciones personalizadas incrustadas en la asignación, donde se definen los componentes 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.

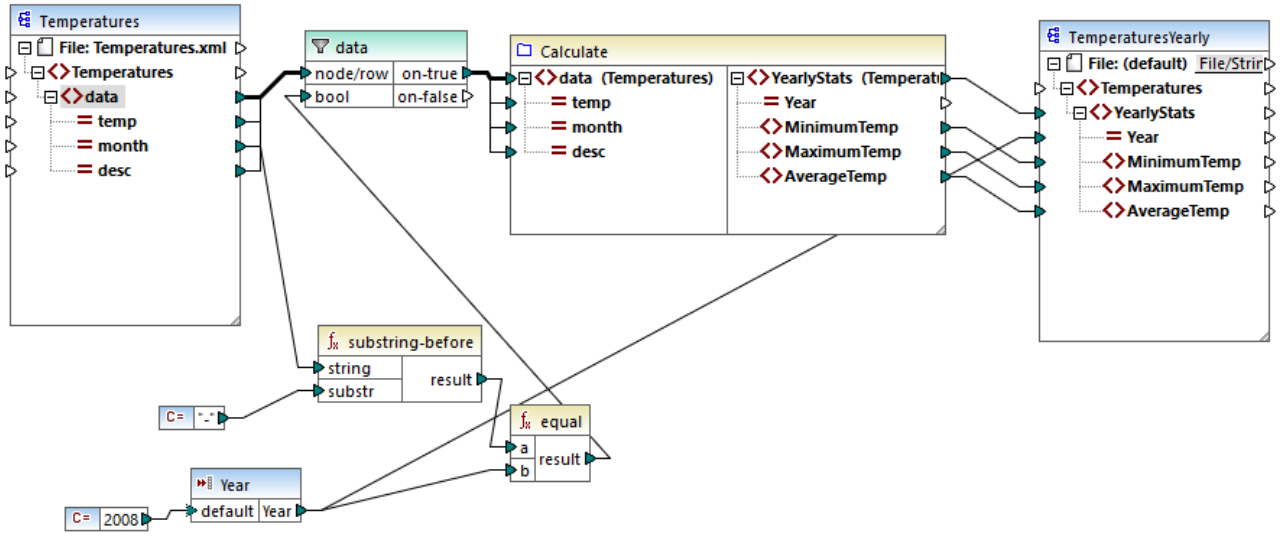
Si no quiere que ocurra eso, puede configurar la función definida por el usuario para que solo se la llame una vez, incluso aunque el componente de entrada sea una secuencia de valores. Tiene sentido usar esta opción en el caso de aquellas funciones definidas por el usuario que operan en un conjunto 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) 485. 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\MapForce2025\MapForceExamples\InputIsSequence.mfd**.

 © 2019-2025 Altova GmbH

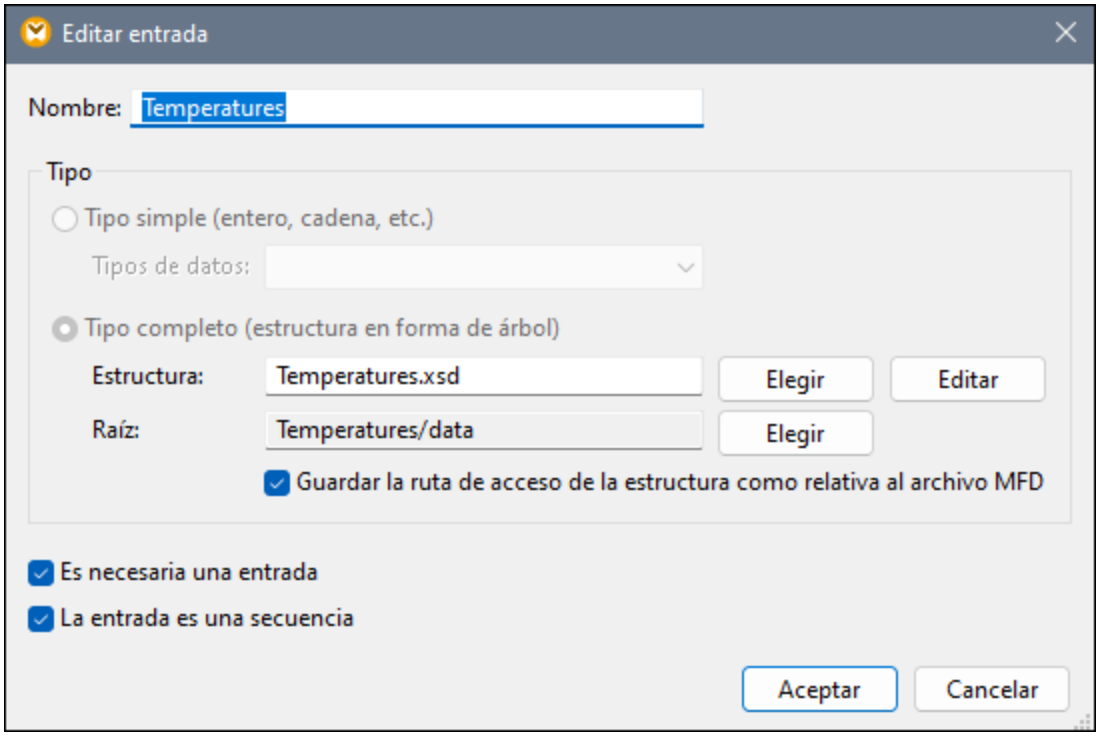



La asignación anterior ilustra un caso típico de una función definida por el usuario que opera en un conjunto de valores y, por tanto, necesita todos los valores de entrada en una sola llamada. Más concretamente, la función definida por el usuario **Calculate** devuelve las temperaturas mínima, máxima 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>
```

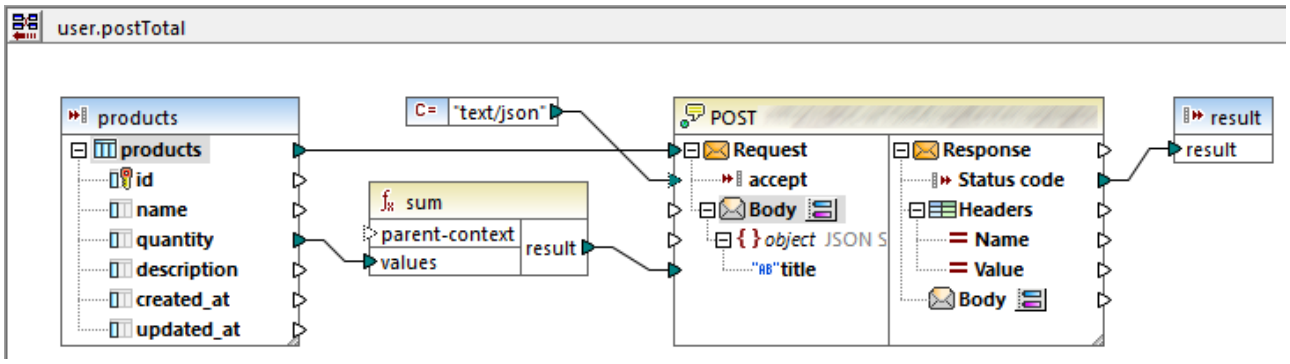
Como es habitual, la ejecución de la asignación empieza por el primer elemento del componente de destino (**YearlyStats** en este ejemplo). Para rellenar este nodo, la asignación intenta obtener datos de entrada de la función definida por el usuario, lo que a su vez desencadena el filtro. El papel del 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. Aquí puede ver un ejemplo:

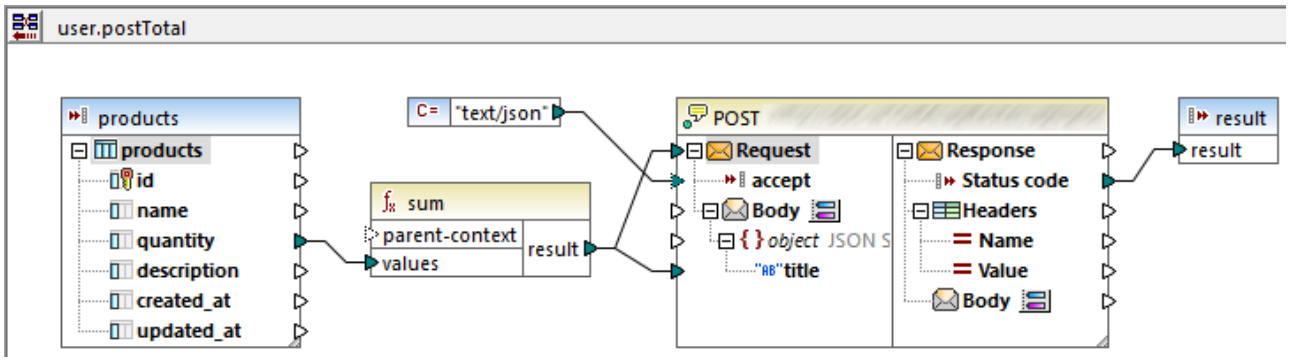



La función definida por el usuario de la imagen anterior recibe una secuencia de valores de la asignación externa. Más concretamente, los datos suministrados al parámetro de origen provienen de una BD. Se ha marcado la casilla **El parámetro de entrada es una secuencia** para el parámetro de origen, por lo que se

 © 2019-2025 Altova GmbH

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.

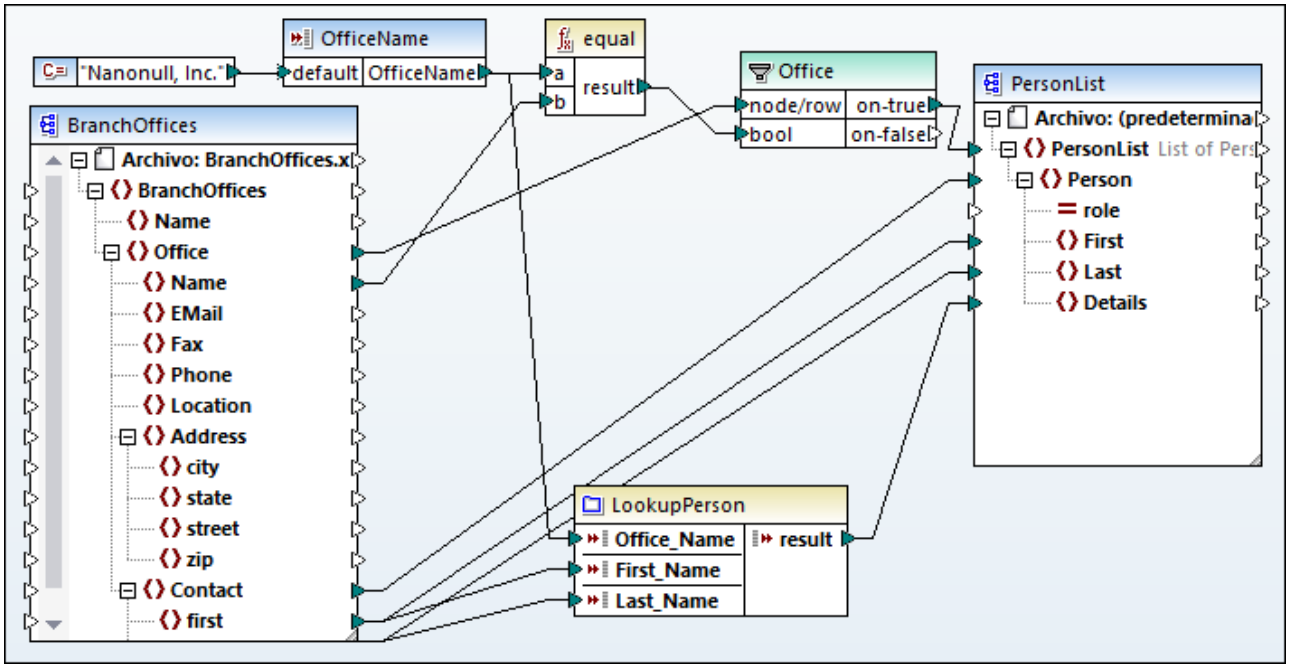
Para solucionar este problema, conecte el componente de entrada **Request** del servicio web al resultado de la función **sum**. La función produce un solo valor, por lo que también se llama al servicio web una sola vez:




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 el Ejemplo: cambiar el contexto primario 85.

## 5.9.4.5 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.

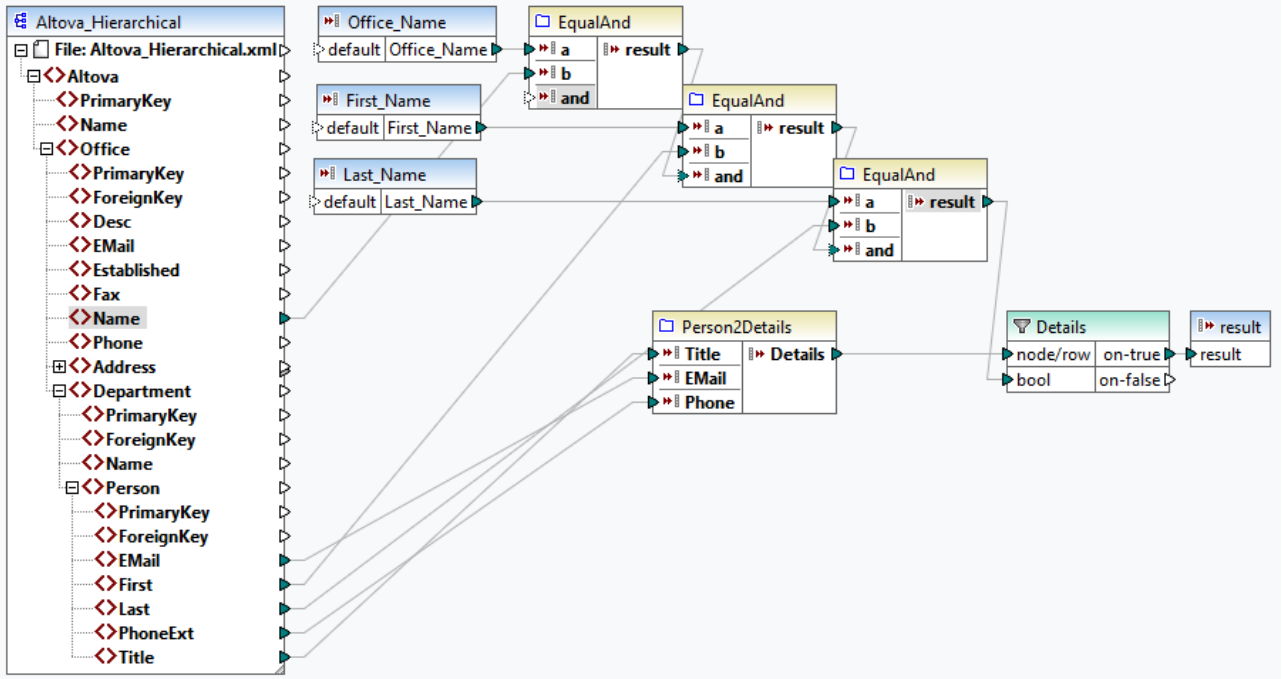
Para ello hemos diseñado la asignación de la siguiente forma:

- 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`.

## LookupPerson: estructura interna

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.

 © 2019-2025 Altova GmbH

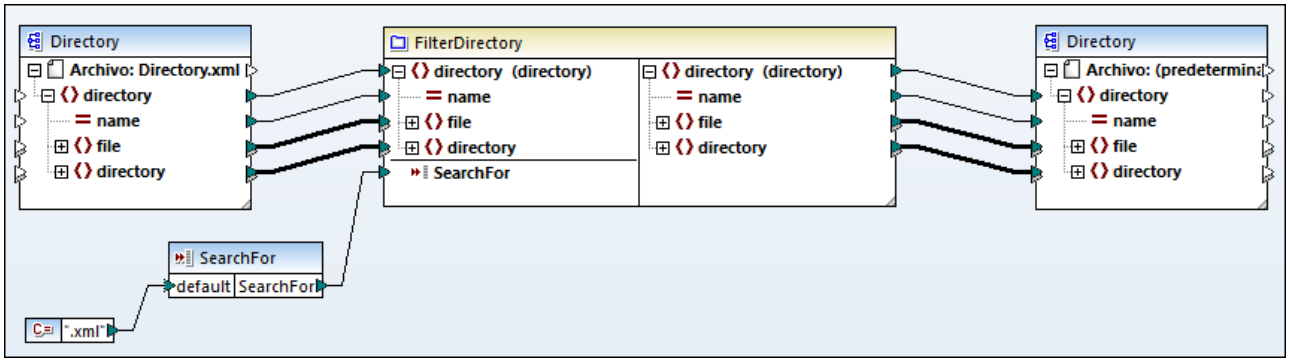



Así se configuró la unción definida por el usuario:

- 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*

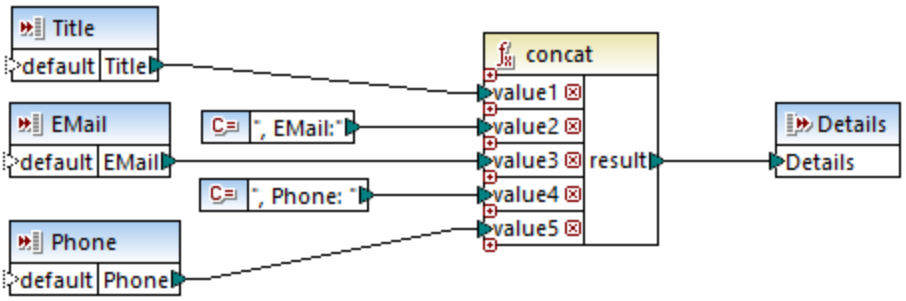
La función `EqualAnd` (*véase más abajo*) es una función definida por el usuario aparte que está definida dentro de la función definida por el usuario `LookupPerson`. Para ver la estructura interna de `EqualAnd` haga doble clic en su encabezado.




`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>
```

 © 2019-2025 Altova GmbH

## 5.9.5 Funciones personales

En esta sección explicamos cómo importar funciones personales XSLT[507], XQuery[514], Java y .NET[518]. También aprenderá a hacer referencia a bibliotecas C#, C++ y Java manualmente[526].

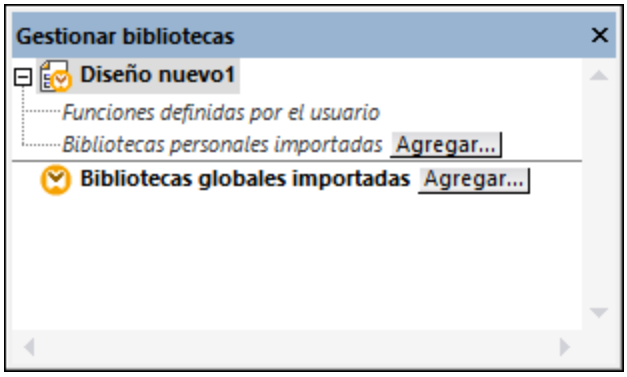
### 5.9.5.1 Importar funciones XSLT 1.0/2.0 personales

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[23]. 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[19] seleccionado es XSLT (*véase también* Gestionar bibliotecas de funciones[464]).

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 68.

## 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 508
Ejemplo: sumar valores de nodos 511
Implementación del motor XSLT 1.0 1153
Implementación del motor XSLT 2.0 1154

### 5.9.5.1.1 Ejemplo: agregar funciones XSLT personales

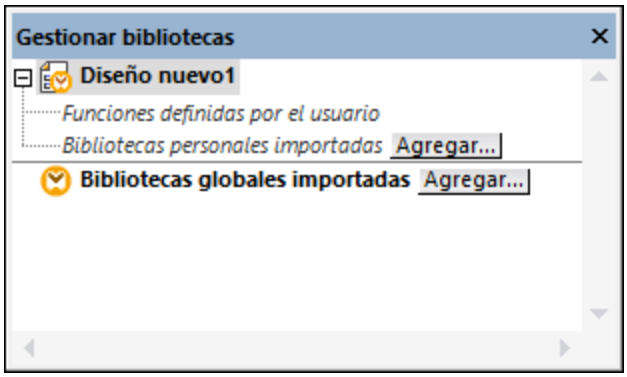
Este ejemplo explica cómo importar funciones XSLT 1.0 personales en MapForce. Los archivos necesarios para seguir este ejemplo están en la carpeta **`C:\Usuarios\<usuario>\Documentos\Altova\MapForce2025\MapForceExamples`**.

- **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.

 © 2019-2025 Altova GmbH

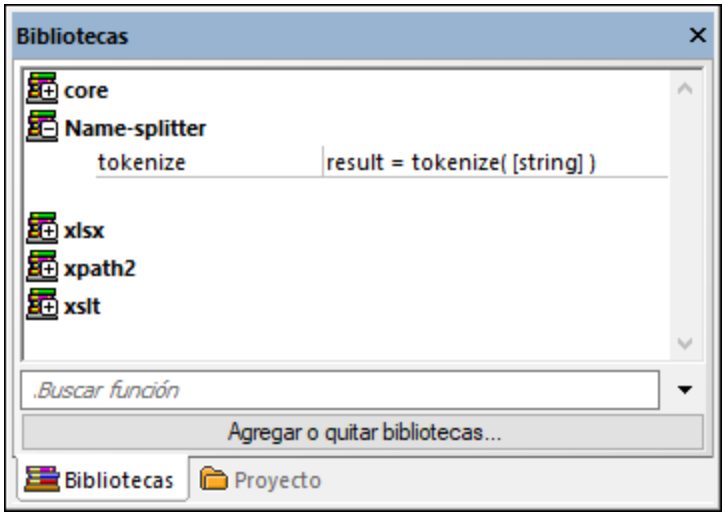
### Para agregar una función XSLT personal:

1. Haga clic en el botón **Agregar o quitar bibliotecas**, en la parte inferior de la ventana Bibliotecas 23. Se abre la ventana **Gestionar bibliotecas**.



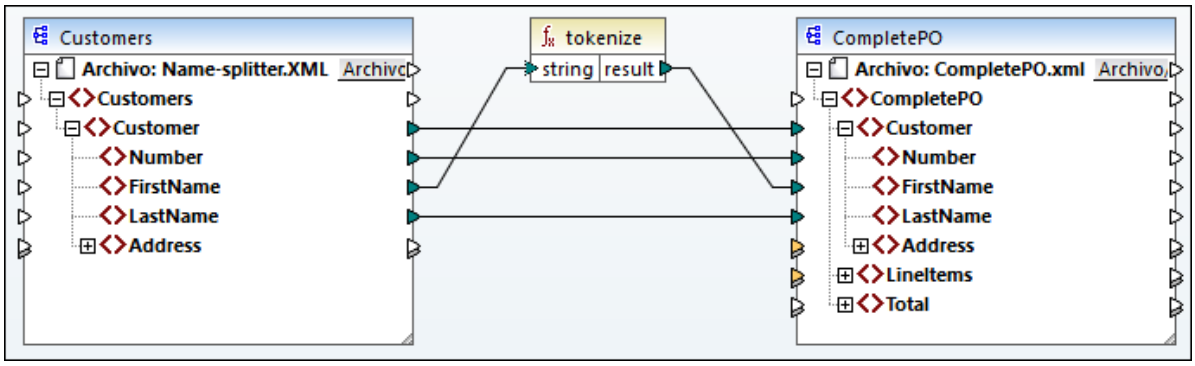

2. Para importar funciones como biblioteca local (dentro del archivo de asignación actual solamente), haga clic en **Agregar** bajo el nombre de la asignación activa. Para importar funciones como biblioteca global (a nivel de programa), haga clic en **Agregar**, junto a **Bibliotecas globales importadas**. 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.

 © 2019-2025 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** ✕.

### 5.9.5.1.2 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\MapForce2025\MapForceExamples\Tutorial**:

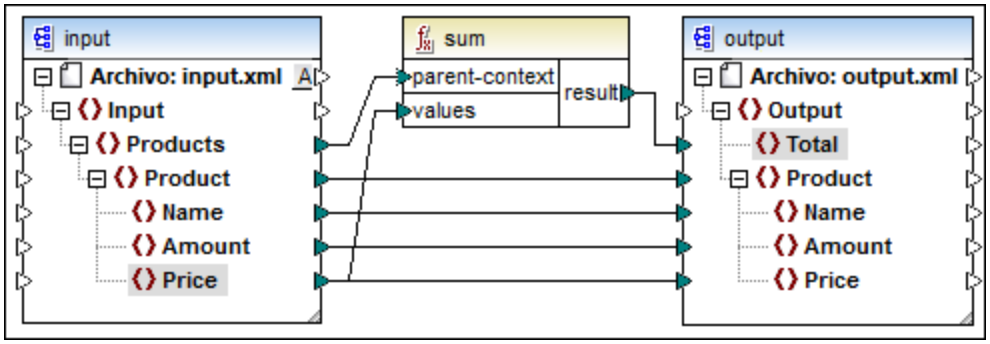
- **Summing-nodes.mfd:** el archivo de asignación de datos
- **input.xml:** el archivo XML de origen
- **input.xsd:** el esquema XML de origen
- **output.xsd:** el esquema XML de destino
- **Summing-nodes.xslt:** una hoja de estilos XSLT personal que contiene una plantilla con nombre que sumará los nodos.

Hay dos maneras de conseguir nuestro objetivo:

- con ayuda de la función de agregado **sum** 553 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) 547.

## 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>

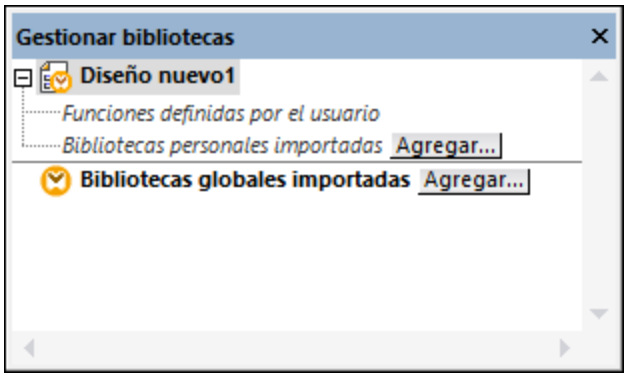
   <xsl:template name="Total">
   <xsl:param name="string"/>
      <xsl:value-of select="sum($string/Product/Price)"/>
   </xsl:template>
</xsl:stylesheet>
```

**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:

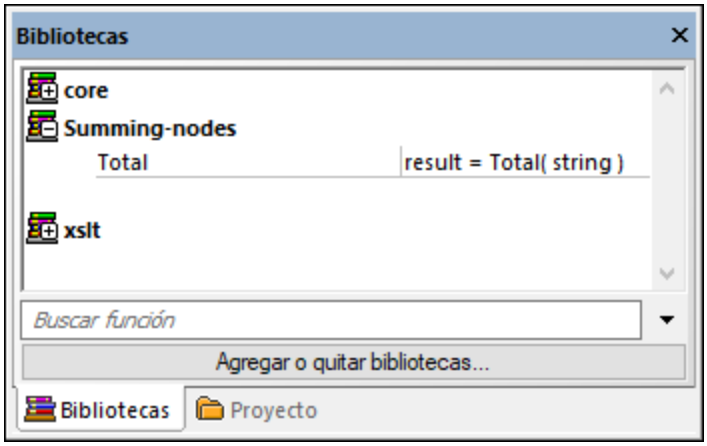
1. Haga clic en el botón **Agregar o quitar bibliotecas**, en la parte inferior de la ventana Bibliotecas [23]. Se abre la ventana **Gestionar bibliotecas**.

 © 2019-2025 Altova GmbH

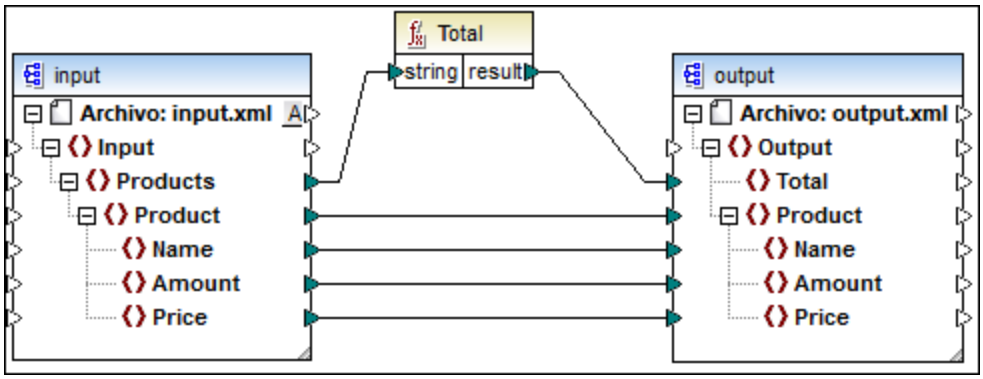



2. Para importar funciones como biblioteca local (dentro del archivo de asignación actual solamente), haga clic en **Agregar** bajo el nombre de la asignación activa. Para importar funciones como biblioteca global (a nivel de programa), haga clic en **Agregar**, junto a **Bibliotecas globales importadas**. 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\MapForce2025\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>
```

## 5.9.5.2 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 sólo 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* 19).

 © 2019-2025 Altova GmbH

## Temas relacionados:

Implementación del motor XQuery 1155

### 5.9.5.2.1 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\MapForce2025\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*

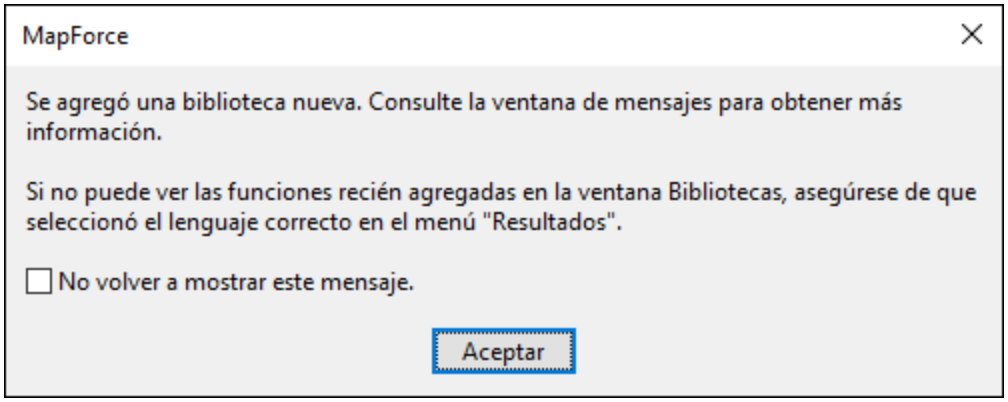
Después de importar el archivo del módulo XQuery en MapForce ya puede llamar a la función `demo:calculatetax` desde una asignación. Observe que estamos calculando el importe con la ayuda de una función XQuery para este ejemplo, pero se puede obtener el mismo resultado usando las funciones integradas de MapForce.

Existe una asignación que llama a la función `demo:calculatetax` en la siguiente ruta: **<Documentos>\Altova\MapForce2025\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.

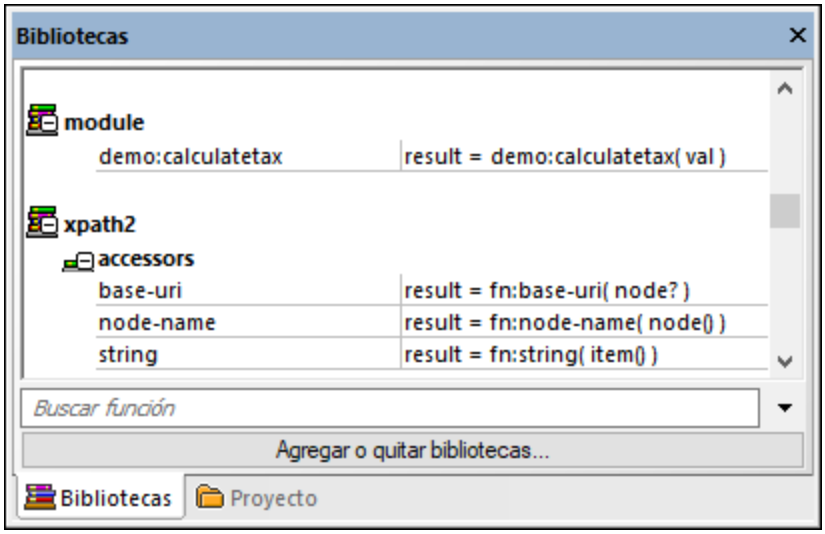
#### Importar el módulo XQuery en MapForce:

1. Vaya hasta el final de la ventana Bibliotecas y haga clic en **Agregar o quitar bibliotecas**. Se abre el cuadro de diálogo "Opciones".

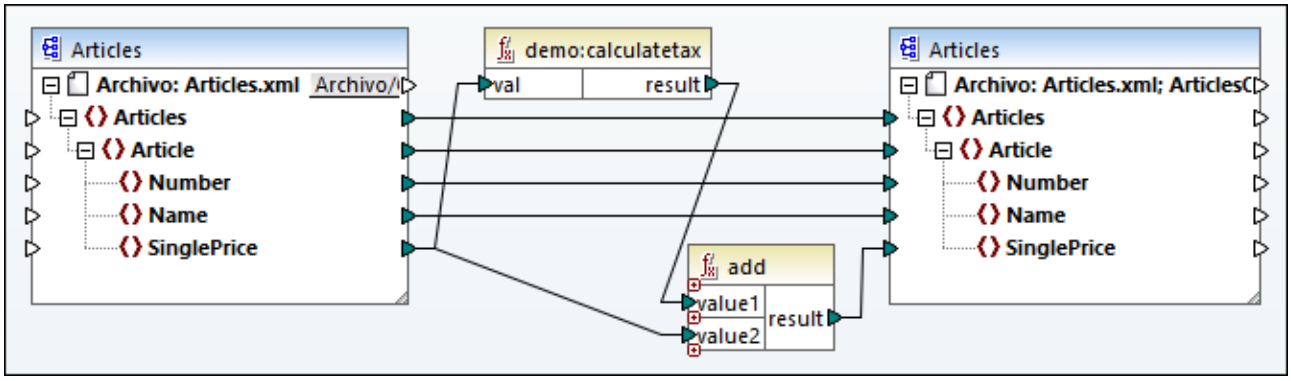
2. Haga clic en **Add** (*Agregar*) y navegue hasta el archivo **<Documentos>\Altova\MapForce2025\MapForceExamples\Tutorial\module.xq**.
3. Haga clic en **Aceptar** para cerrar el cuadro de diálogo. Aparece un cuadro de mensaje que confirma que se ha agregado una biblioteca nueva.




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* 462).




*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>
```

## 5.9.5.3 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 522 e Ejemplo: importar ensamblado .NET DLL personal 524.

**Tenga en cuenta que:**

 © 2019-2025 Altova GmbH

- 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* 520).

- 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* 519).

- 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* 526).

- 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 |


© 2019-2025 Altova GmbH

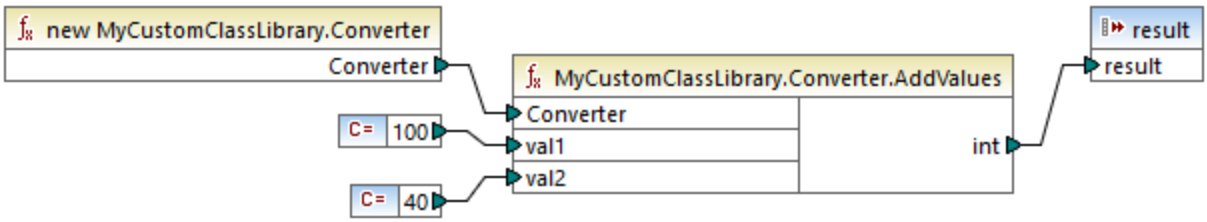
| Tipos XML Schema | Tipos .NET | Tipos C# |
|---|---|---|
| xs:unsignedInt | System.UInt32 | uint |
| xs:unsignedLong | System.UInt64 | ulong |
| xs:boolean | System.Boolean | bool |
| xs:float | System.Single | float |
| xs:double | System.Double | double |
| xs:decimal | System.Decimal | decimal |

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

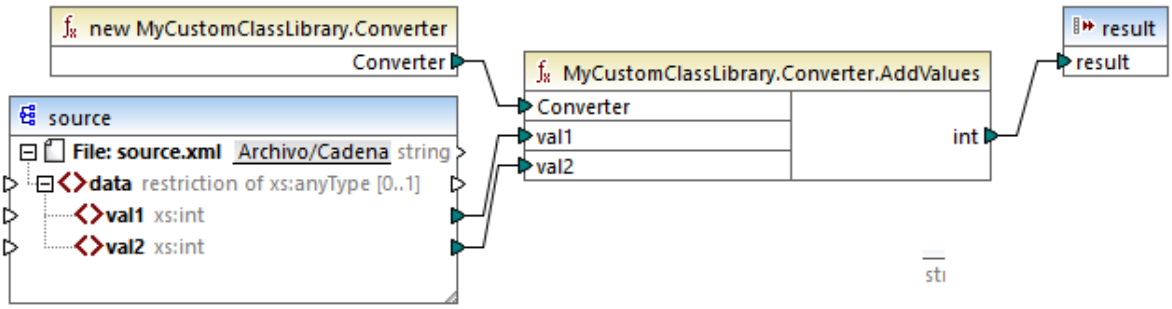
Si una función de una biblioteca personal espera tipos enteros y usted conecta constantes de tipo "Número" a los argumentos de la función, puede ocurrir un error de incompatibilidad parecido a este:

`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.

### 5.9.5.3.1 Ejemplo: importar clase Java personales

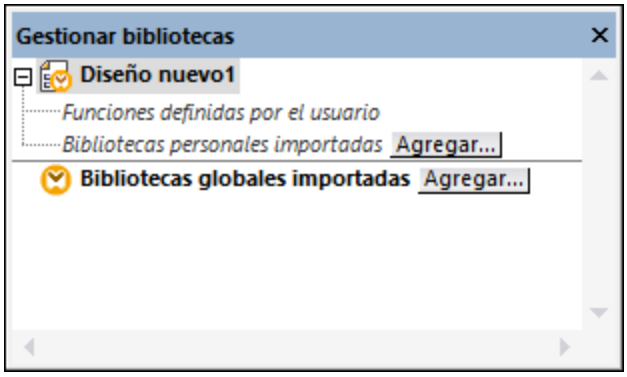
Este ejemplo explica cómo importar un archivo de clases Java personal en MapForce.

**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 23. Se abre la ventana **Gestionar bibliotecas**.

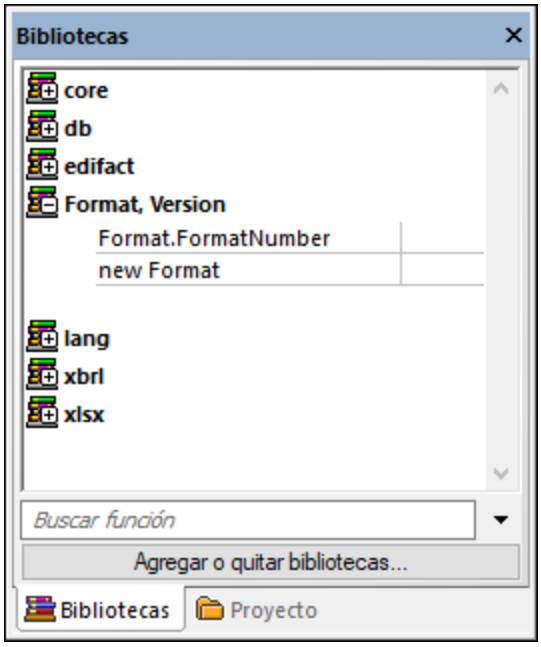



2. Para importar funciones como biblioteca local (dentro del archivo de asignación actual solamente), haga clic en **Agregar** bajo el nombre de la asignación activa. Para importar funciones como biblioteca global (a nivel de programa), haga clic en **Agregar**, junto a **Bibliotecas globales importadas**. 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\MapForce2025\MapForceExamples\Java\Format\Format.class`**. Aparece un

 © 2019-2025 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 19 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* 462).

*Resultado de la asignación*
Para obtener una vista previa del resultado de la asignación en MapForce:

1. Abra el archivo **`<Documentos>\Altova\MapForce2025\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*).

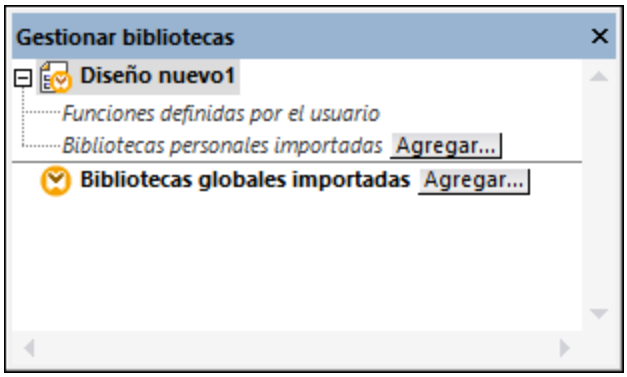
### 5.9.5.3.2 Ejemplo: importar ensamblado .NET DLL personal

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\MapForce2025\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:

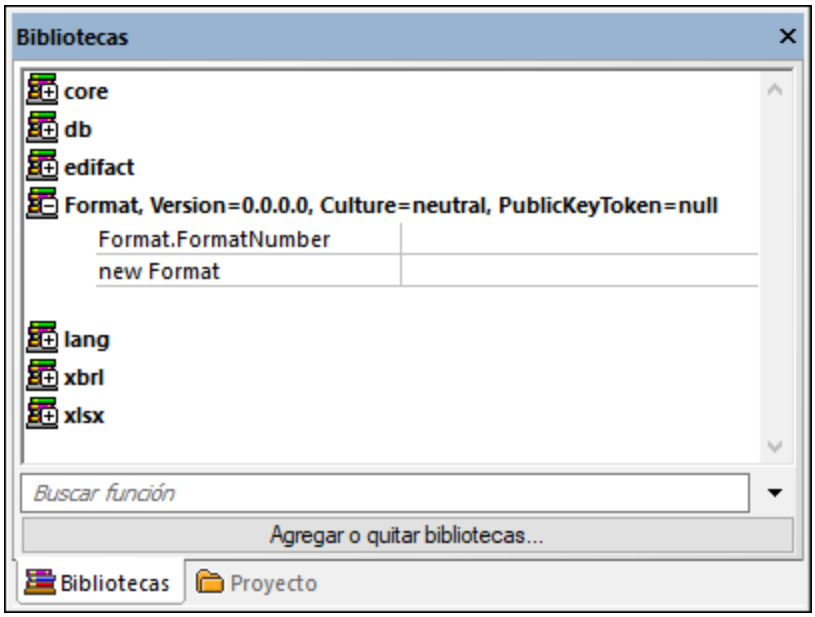
1. Haga clic en el botón **Agregar o quitar bibliotecas**, en la parte inferior de la ventana Bibliotecas 23. Se abre la ventana **Gestionar bibliotecas**.




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.

 © 2019-2025 Altova GmbH




Si no ve la biblioteca que acaba de importar, asegúrese de que el lenguaje de transformación [19] 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* [462]).

*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#*).

## 5.9.5.4 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 532.
- 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 487.
- 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* 532).

Para aprender a crear y configurar un archivo `.mff` personal consulte el apartado Configurar el archivo .mff 526. Para ver algunos ejemplos consulte estos apartados:


- Ejemplo: hacer referencia a C# en .mff 535
- Ejemplo: hacer referencia a C++ en .mff 536
- Ejemplo: hacer referencia a Java en .mff 539


### 5.9.5.4.1 Configurar un 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 Bibliotecas. Un archivo `.mff` hace de intermediario entre las bibliotecas personales del usuario y MapForce y

 © 2019-2025 Altova GmbH

debe configurarse para especificar las interfaces con las funciones personales y dónde se puede encontrar la implementación en el código generado.

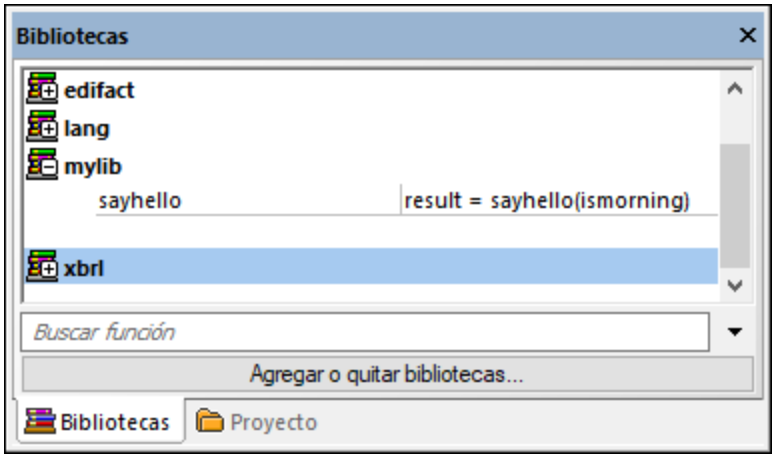
Tenga en cuenta que:

- 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

 © 2019-2025 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:

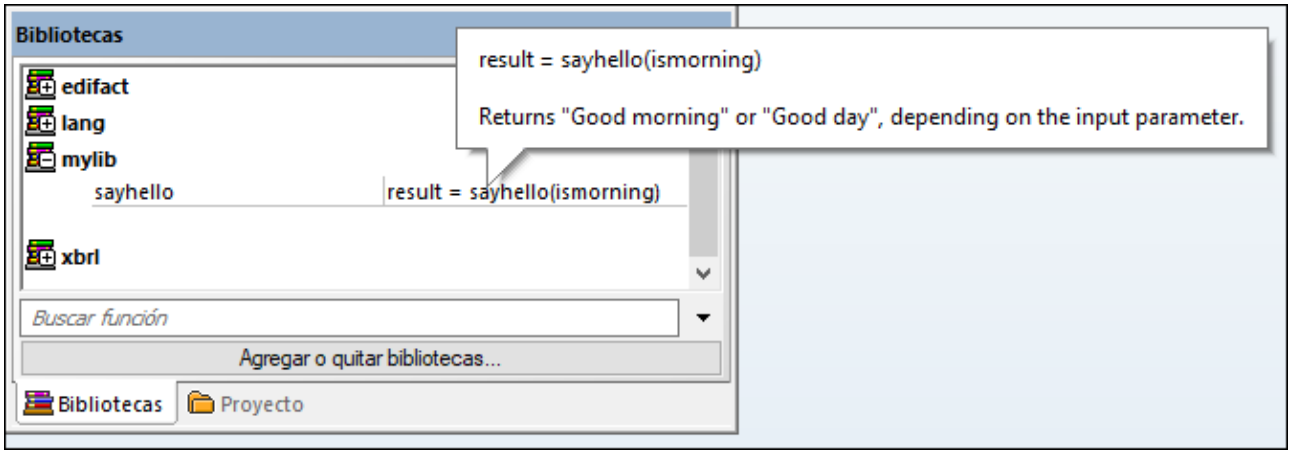
En el fragmento de código, se definió `<datapoint>` como parámetro de entrada o salida de una función (o conector de entrada o salida). El argumento `type` de `<datapoint>` especifica el tipo de datos del parámetro (o el tipo de datos del valor devuelto).

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 532.

 © 2019-2025 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:

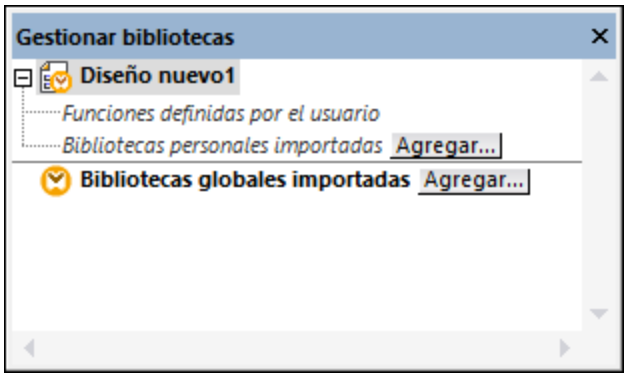
```
<!-- ... -->
<component name="sayhello">
<!-- ... -->
   <implementations>
      <implementation language="cs">
         <function name="HelloFunction"/>
      </implementation>
      <implementation language="java">
         <function name="Hello"/>
      </implementation>
      <implementation 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++.

### 5.9.5.4.2 Importar bibliotecas .mff

Tras crear el archivo `.mff` personal (*véase* *Configurar el archivo .mff* 526), 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 23. 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\MapForce2025\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 sólo agregar o eliminar el correspondiente archivo de biblioteca (`*.mff`).

### 5.9.5.4.3 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.

| Tipo XML Schema | Tipo Java | Tipo C# | Tipo C++ |
|---|---|---|---|
| `anyAtomicType` | `String` | `string` | `string_type` |

 © 2019-2025 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 1005 | Altova.Types.DateTime 988 | altova::DateTime 973 |
| dateTime | com.altova.types.DateTime 1005 | Altova.Types.DateTime 988 | altova::DateTime 973 |
| dayTimeDuration | com.altova.types.Duration 1010 | Altova.Types.Duration 994 | altova::Duration 976 |
| decimal | java.math.BigDecimal | decimal | double |
| double | double | double | double |
| duration | com.altova.types.Duration 1010 | Altova.Types.Duration 994 | altova::Duration 976 |
| ENTITIES | String | string | string_type |
| ENTITY | String | string | string_type |
| float | double | double | double |
| gDay | com.altova.types.DateTime 1005 | Altova.Types.DateTime 988 | altova::DateTime 973 |
| gMonth | com.altova.types.DateTime 1005 | Altova.Types.DateTime 988 | altova::DateTime 973 |
| gMonthDay | com.altova.types.DateTime 1005 | Altova.Types.DateTime 988 | altova::DateTime 973 |
| gYear | com.altova.types.DateTime 1005 | Altova.Types.DateTime 988 | altova::DateTime 973 |
| gYearMonth | com.altova.types.DateTime 1005 | Altova.Types.DateTime 988 | altova::DateTime 973 |
| 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 1005 | Altova.Types.DateTime 988 | altova::DateTime 973 |
| 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 1010 | Altova.Types.Duration 994 | altova::Duration 976 |


© 2019-2025 Altova GmbH

### 5.9.5.4.4 Referencias a bibliotecas C# en .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* 524).

## 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 532.

*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 526.

```
<?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 532.

### 5.9.5.4.5 Referencias a bibliotecas C++ en .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`**.

 © 2019-2025 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 526. Para más información sobre los tipos de datos consulte Correspondencias entre tipos de datos 532.

```
<?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 532.

*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++. Si recibe un error de compilador en esta línea:

```
#import "msado15.dll" rename("EOF", "EndOfFile")
```

 © 2019-2025 Altova GmbH

modifique las propiedades del proyecto para incluir una referencia al archivo **msado15.dll** en **C:\Program Files\Common Files\System\ADO**.

### 5.9.5.4.6 Referencias a bibliotecas Java en .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 522).

#### 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 526.

```
<?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 532.

## 5.9.6 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`** 689 y **`tokenize-regexp`** 637
- 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 479.

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.

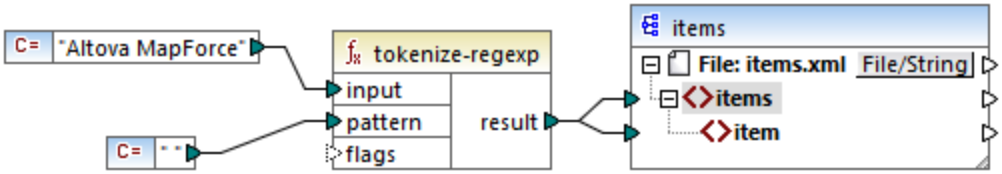
 © 2019-2025 Altova GmbH

## 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 632. 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.

## 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. |


© 2019-2025 Altova GmbH

| | |
|---|---|
| ***** | 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

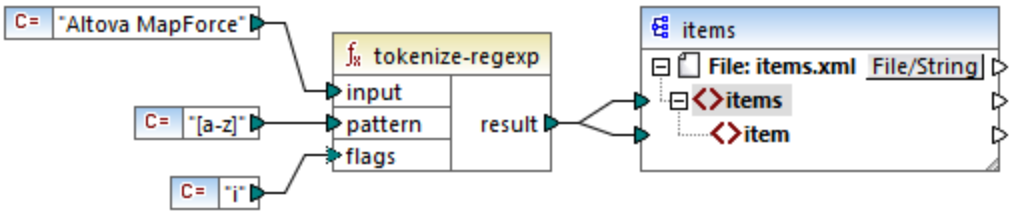
Los paréntesis **(** y **)** se usan para agrupar partes de la expresión regular. Se pueden usar para aplicar cuantificadores a una subexpresión (en vez de a un solo carácter) o con alternancia (véase a continuación).

## 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".<br><br>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.<br><br>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.<br><br>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.<br><br>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.<br><br>  |

| | |
|---|---|
| **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**.<br><br>**Nota:** los caracteres de espacio en blanco sin expresiones de clases de caracteres no se eliminan (por ejemplo, `[#x20]`). |

## 5.9.7 Referencia de la biblioteca de funciones

Esta sección de la referencia describe las funciones integradas de MapForce disponibles en la ventana Bibliotecas [23]. Las funciones están organizadas por biblioteca. Las funciones disponibles en la ventana **Bibliotecas** dependen del lenguaje de transformación que elija para su asignación de datos. Para saber más sobre la lista de lenguajes de transformación disponibles, consulte este tema [19].

La información sobre la compatibilidad de las funciones y los lenguajes de transformación se proporciona en las subsecciones siguientes.

### core (funciones básicas)

Las listas siguientes resumen la compatibilidad de las funciones básicas con los lenguajes de transformación.

*core | aggregate (funciones de 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 (funciones de 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 (funciones de ruta de archivos)*
Todas las funciones de ruta de archivos son compatibles con todos los lenguajes de transformación.

*core | generator (funciones del generador)*
La función **auto-number** está disponible para todos los lenguajes de transformación.

*core | logical (funciones lógicas)*
Todas las funciones lógicas son compatibles con todos los lenguajes de transformación.

*core | math (funciones matemáticas)*

- **add**, **ceiling**, **divide**, **floor**, **modulus**, **multiply**, **round**, **subtract**: todos los lenguajes de transformación
- **round-precision**: C#, C++, Java, Built-In.

*core | node (funciones de nodo)*


© 2019-2025 Altova GmbH

- **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 (funciones QName)*
Las funciones `QName` son compatibles con todos los lenguajes de transformación excepto XSLT1.0.

*core | sequence (funciones de 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 (funciones de 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.

## bson (solo MapForce Enterprise Edition)

Todas las funciones BSON son compatibles con el lenguaje de transformación Built-in solamente.

## db (MapForce Professional y Enterprise Edition)

Las funciones `db` son compatibles con C#, C++, Java, Built-In.

## edifact (solo MapForce Enterprise Edition)

Las funciones `edifact` son compatibles con C#, C++, Java, Built-In.

## lang (MapForce Professional y Enterprise Edition)

Las listas siguientes resumen la compatibilidad de las funciones `lang` con los lenguajes de transformación.
*lang | datetime (funciones fechaHora)*
Las funciones `lang | datetime` son compatibles con C#, C++, Java, Built-In.

*lang | file*
Las funciones **read-binary-file** y **write-binary-file** solo son compatibles con Built-In.

*lang | generator (funciones del generador)*
La función **create-guid** está disponible para C#, C++, Java, Built-In.

*lang | logical (funciones lógicas)*
Las funciones `lang | logical` están disponibles para C#, C++, Java, Built-In.

*lang | math (funciones matemáticas)*
Las funciones `lang | math` están disponibles para C#, C++, Java, Built-In.

*lang | QName (funciones QName)*
Las funciones `lang | Qname` son compatibles con C#, C++, Java, Built-In.

*lang | string (funciones de cadena)*

- **`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.

## mime (solo MapForce Enterprise Edition)

Las funciones `mime` solo están disponibles para Built-In.

## xbrl (solo MapForce Enterprise Edition)

Las funciones `xbrl` son compatibles con C#, C++, Java, Built-In.

## xlsx (solo MapForce Enterprise Edition)

Las funciones `xlsx` son compatibles con XSLT 2.0, XSLT 3.0, C#, Java y Built-In.

## xpath2

Todas las funciones `xpath2` son compatibles con XSLT 2.0, XSLT 3.0 y XQuery 1.0.

## xpath3

Todas las funciones `xpath3` son compatibles con el lenguaje de transformación XSLT 3.0 solamente.

## xslt10

Las listas siguientes resumen la compatibilidad de las funciones `xslt10` con los lenguajes de transformación.
*xslt10 | xpath*

- **`local-name`**, **`name`**, **`namespace-uri`**: XSLT 1.0, XSLT 2.0, y XSLT 3.0.
- **`lang`**, **`last`**, **`position`**: XSLT 1.0.

*xslt10 | xslt*

- **`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.

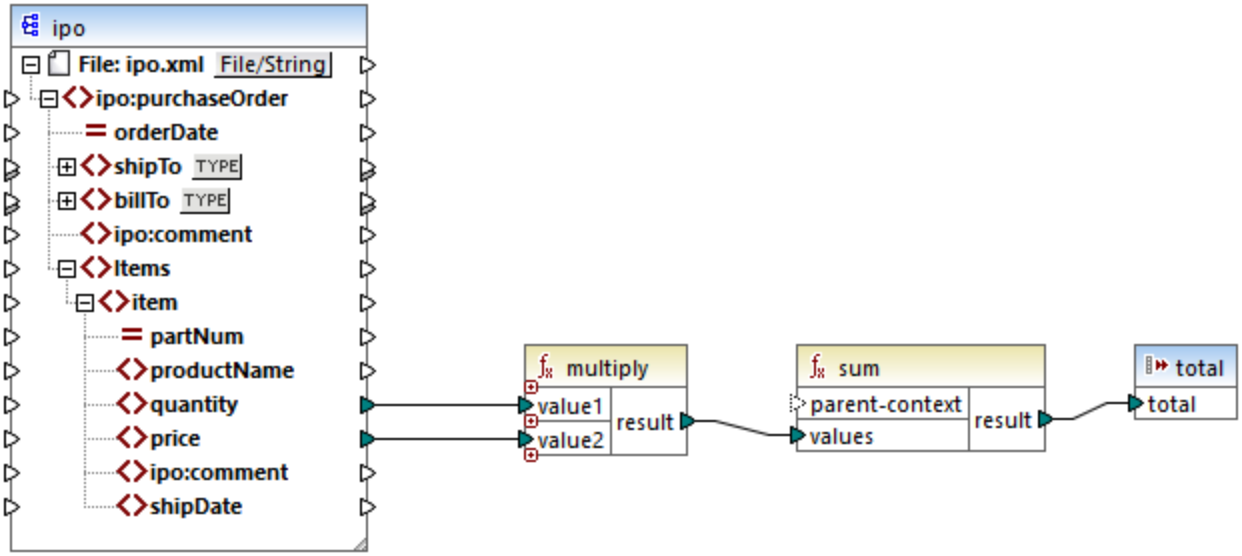

© 2019-2025 Altova GmbH

## 5.9.7.1 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 85.
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. Encontrará esta asignación en la carpeta **`MapForceExamples`**.

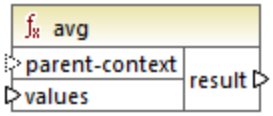



*SimpleTotal.mfd*

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.

### 5.9.7.1.1 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.

## Parámetros

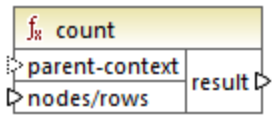
| Argumento | Descripción |
|---|---|
| **parent-context** | Argumento opcional. Suministra el contexto primario. Véase también el apartado Ejemplo: cambiar el contexto primario 85.<br><br>**parent-context** es un argumento opcional de algunas funciones de agregado de la biblioteca de funciones **core** de MapForce (como, por ejemplo, de las funciones `min`, `max`, `avg` y `count`). En un componente de origen que tiene varias secuencias jerárquicas, el contexto primario determina en qué conjunto de nodos debe operar la función. |
| **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 603.

### 5.9.7.1.2 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.


© 2019-2025 Altova GmbH

## Parámetros

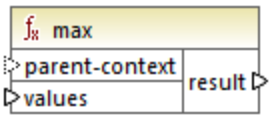
| Argumento | Descripción |
|---|---|
| **parent-context** | Argumento opcional. Suministra el contexto primario. Véase también el apartado Ejemplo: cambiar el contexto primario 85.<br><br>**parent-context** es un argumento opcional de algunas funciones de agregado de la biblioteca de funciones **core** de MapForce (como, por ejemplo, de las funciones `min`, `max`, `avg` y `count`). En un componente de origen que tiene varias secuencias jerárquicas, el contexto primario determina en qué conjunto de nodos debe operar la función. |
| **nodes/rows** | Este argumento debe estar conectado al elemento de origen que se debe contar. |

## Ejemplo

Consulte el apartado Ejemplo: cambiar el contexto primario 85, Ejemplo: contar filas de tabla de BD 393.

### 5.9.7.1.3 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 contexto primario. Véase también el apartado Ejemplo: cambiar el contexto primario 85.<br><br>**parent-context** es un argumento opcional de algunas funciones de agregado de la biblioteca de funciones **core** de MapForce (como, por ejemplo, de las funciones `min`, `max`, `avg` y `count`). En un componente de origen que tiene varias secuencias jerárquicas, el contexto primario determina en qué conjunto de nodos debe operar la función. |

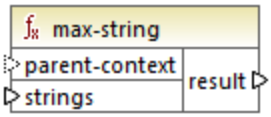
| Argumento | Descripción |
|---|---|
| **values** | Este argumento debe estar conectado a un elemento de entrada que suministra los datos propiamente dichos. Recuerde que el valor de argumento dado debe ser numérico. Para obtener el máximo de una secuencia de cadenas, utilice la función max-string 550. |

## Ejemplo

Consulte el apartado Ejemplo: agrupar registros por clave 603.

### 5.9.7.1.4 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.

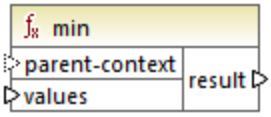
## Parámetros

| Argumento | Descripción |
|---|---|
| **parent-context** | Argumento opcional. Suministra el contexto primario. Véase también el apartado Ejemplo: cambiar el contexto primario 85.<br><br>**parent-context** es un argumento opcional de algunas funciones de agregado de la biblioteca de funciones **core** de MapForce (como, por ejemplo, de las funciones `min`, `max`, `avg` y `count`). En un componente de origen que tiene varias secuencias jerárquicas, el contexto primario determina en qué conjunto de nodos debe operar la función. |
| **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`. |

### 5.9.7.1.5 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.

 © 2019-2025 Altova GmbH




## Lenguajes

Built-in, C++, C#, Java, XQuery, XSLT 2.0, XSLT 3.0.

## Parámetros

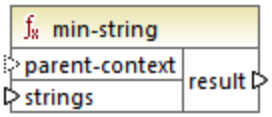
| Argumento | Descripción |
|---|---|
| **parent-context** | Argumento opcional. Suministra el contexto primario. Véase también el apartado Ejemplo: cambiar el contexto primario 85.<br><br>**parent-context** es un argumento opcional de algunas funciones de agregado de la biblioteca de funciones **core** de MapForce (como, por ejemplo, de las funciones `min`, `max`, `avg` y `count`). En un componente de origen que tiene varias secuencias jerárquicas, el contexto primario determina en qué conjunto de nodos debe operar la función. |
| **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 mínimo de una secuencia de cadenas, utilice la función min-string 551. |

## Ejemplo

Consulte el apartado Ejemplo: agrupar registros por clave 603.

### 5.9.7.1.6 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

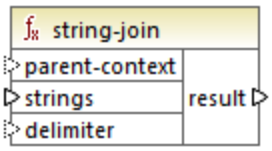
Built-in, C++, C#, Java, XQuery, XSLT 2.0, XSLT 3.0.

## Parámetros

| Argumento | Descripción |
|---|---|
| **parent-context** | Argumento opcional. Suministra el contexto primario. Véase también el apartado Ejemplo: cambiar el contexto primario 85.<br><br>**parent-context** es un argumento opcional de algunas funciones de agregado de la biblioteca de funciones **core** de MapForce (como, por ejemplo, de las funciones **min**, **max**, **avg** y **count**). En un componente de origen que tiene varias secuencias jerárquicas, el contexto primario determina en qué conjunto de nodos debe operar la función. |
| **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`. |

### 5.9.7.1.7 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 La función devuelve una cadena vacía si el argumento **strings** es un conjunto vacío.




## Lenguajes

Built-in, C++, C#, Java, XQuery, XSLT 2.0, XSLT 3.0.

## Parámetros

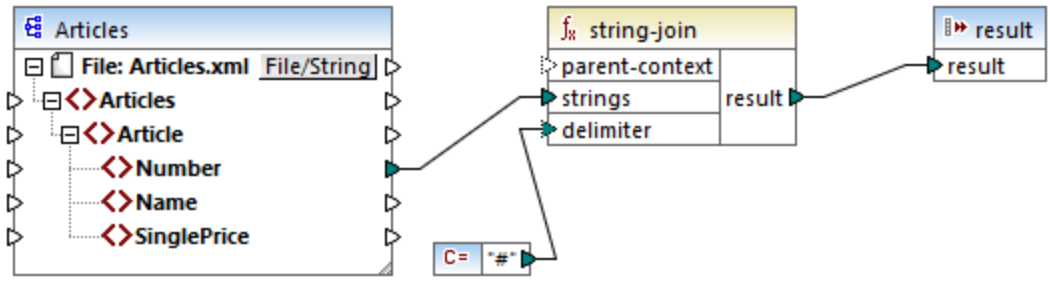
| Argumento | Descripción |
|---|---|
| **parent-context** | Argumento opcional. Suministra el contexto primario. Véase también el apartado Ejemplo: cambiar el contexto primario 85.<br><br>**parent-context** es un argumento opcional de algunas funciones de agregado de la biblioteca de funciones **core** de MapForce (como, por ejemplo, de las funciones **min**, **max**, **avg** y **count**). En un componente de origen que tiene varias secuencias jerárquicas, el contexto primario determina en qué conjunto de nodos debe operar la función. |
| **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 |


© 2019-2025 Altova GmbH

| Argumento | Descripción |
|---|---|
| | varios) de `xs:string`. |
| **delimiter** | Argumento opcional. Indica el delimitador que se introduce entre dos cadenas consecutivas. |

## Ejemplo

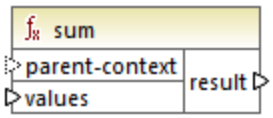
El ejemplo de la siguiente imagen muestra cuatro elementos **Article** con los números: 1, 2, 3, y 4.




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**.

## 5.9.7.1.8 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 contexto primario. Véase también el apartado Ejemplo: cambiar el contexto primario 85.<br><br>**parent-context** es un argumento opcional de algunas funciones de agregado de la biblioteca de funciones **core** de MapForce (como, por ejemplo, de las funciones **min**, |

| Argumento | Descripción |
|---|---|
| | **max**, **avg** y **count**). En un componente de origen que tiene varias secuencias jerárquicas, el contexto primario determina en qué conjunto de nodos debe operar la función. |
| **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

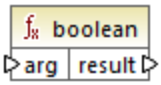
Consulte el apartado Ejemplo: sumar valores de nodo 511.

## 5.9.7.2 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 563 para forzar una comparación numérica.

### 5.9.7.2.1 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 432. 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.

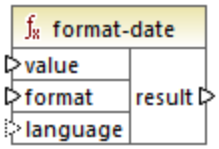
### Parámetros

| Argumento | Descripción |
|---|---|
| **arg** | Argumento obligatorio. Indica el valor que se va a convertir. |

 © 2019-2025 Altova GmbH

### 5.9.7.2.2 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

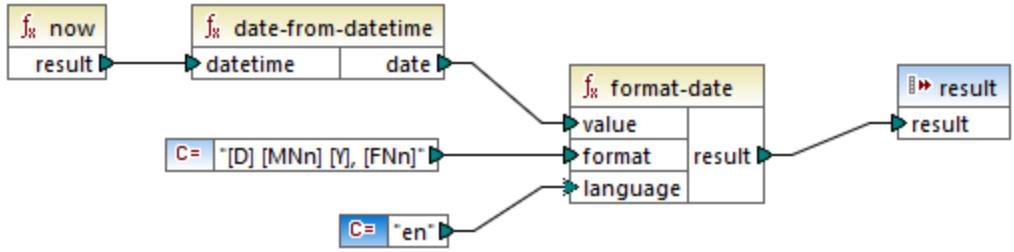
La función está disponible para XSLT 2.0, XSLT 3.0, Java, C#, C++ y el motor de ejecución integrado.

## Parámetros

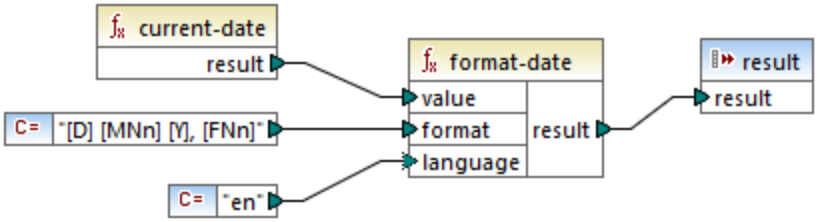
| Argumento | Descripción |
|---|---|
| **value** | Fecha a la que se debe aplicar formato. |
| **format** | Cadena de formato que identifica el formato que se debe aplicar a la fecha. Este argumento se usa igual que el argumento **format** de la format-dateTime 556 . |
| **language** | Argumento opcional. Devuelve el nombre del mes y el día de la semana en el idioma seleccionado. Son valores válidos:<br><br>**de** alemán<br>**en (predet.)** inglés<br>**es** español<br>**fr** francés<br>**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**.

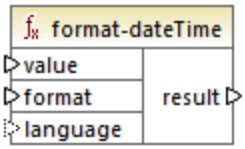



Tenga en cuenta que la asignación anterior se ha diseñado para Java, C#, C++ y el motor de ejecución integrado. Para obtener el mismo resultado en XSLT 2.0 use esta asignación:




### 5.9.7.2.3 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

Built-in, C++, C#, Java, XSLT 2.0, XSLT 3.0.

## 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`. Véase *Formato* más abajo. |


© 2019-2025 Altova GmbH

| Argumento | Descripción |
|---|---|
| `language` | Argumento opcional. Devuelve el nombre del mes y el día de la semana en el idioma seleccionado. Los valores válidos son:<br><br>`de` alemán<br>`en` (predet.) inglés<br>`es` español<br>`fr` francés<br>`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 41 del componente de destino.

## Formato

El argumento `format` consiste en una cadena que contiene *marcadores de variables* entre corchetes (por ejemplo, **`[Y]/[M]/[D]`**). 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 consta de las siguientes partes:

- un especificador que identifica qué parte del tipo de datos debe mostrarse (es decir, la fecha o la hora),
- un modificador de formato opcional,
- un modificador de presentación opcional, y
- un modificador del ancho opcional, que debe ir precedido de una coma si está presente.

```
[marcador de variable] = [especificador(formato)?(presentación)?(ancho)?]

donde

ancho := , min-width ("-" max-width)?
```

*Especificadores*
Los especificadores disponibles se describen en la tabla siguiente.

| Especificador | Descripción | Presentación predeterminada |
|---|---|---|
| **Y** | año (valor absoluto) | cuatro dígitos (2010) |
| **M** | mes del año | 1-12 |

| Especificador | Descripción | Presentación predeterminada |
|---|---|---|
| **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 |
| **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 |

*Modificadores de formato*
El modificador de formato puede ser cualquiera de estos:

| Modificador de formato | 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 |
| **n** | nombre del componente, todo en minúsculas | lunes, martes |
| **Nn** | nombre del componente, mayúsculas o minúsculas | Lunes, Martes |

**Nota importante**
los modificadores **N**, **n** y **Nn** solamente son compatibles con estos especificadores: **M**, **F** y **P**. Los modificadores **N**, **n** y **Nn** también son compatibles con el especificador **z**, pero solo en el lenguaje de transformación Built-in y en la generación de código (*Professional y Enterprise Edition*).

Si necesita un modificador del ancho, antepóngale una coma. El modificador del ancho es un dígito que expresa el ancho mínimo. Opcionalmente, puede agregar un guion y un dígito que exprese el ancho máximo.

 © 2019-2025 Altova GmbH

Por ejemplo:

- **[D,2]** es el día del mes, con ceros a la izquierda (dos dígitos).
- **[MNn,3-3]** es el nombre del mes, escrito con tres caracteres, por ejemplo, *Ene*, *Feb*, *Mar*, etc.

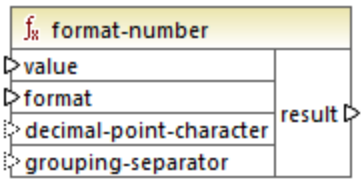
## Ejemplos

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 qué devuelve la función.

| Valor | Formato | Resultado |
|---|---|---|
| 2003-11-03T00:00:00 | `[D]/[M]/[Y]` | 3/11/2003 |
| 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 |
| 2010-06-02T08:02:12.054 | `[Y] [MNn] [D01] [FNn,3-3] [d] [H]:[m]:[s].[f]` | 2010 June 02 Wed 153 8:02:12.054 |
| 2010-06-02T08:02:12.054+02:00 | `[Y] [MNn] [D01] [FNn,3-3] [d] [H]:[m]:[s].[f] [z]` | 2010 June 02 Wed 153 8:02:12.054 GMT+02:00 |
| 2010-06-02T08:02:12.054+02:00 | `[Y] [MNn] [D1] [FNn] [H]:[m]:[s].[f] [Z]` | 2010 June 2 Wednesday 8:02:12.054 +02:00 |
| 2010-06-02T08:02:12.054 | `[Y] [MNn] [D] [FNn,3-3] [H01]:[m]:[s]` | 2010 June 2 Wed 08:02:12 |

## 5.9.7.2.4 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 ( . ). |
| **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 41 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 |

 © 2019-2025 Altova GmbH

| Carácter especial | Predeterminado | Descripción |
|---|---|---|
| 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. 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.

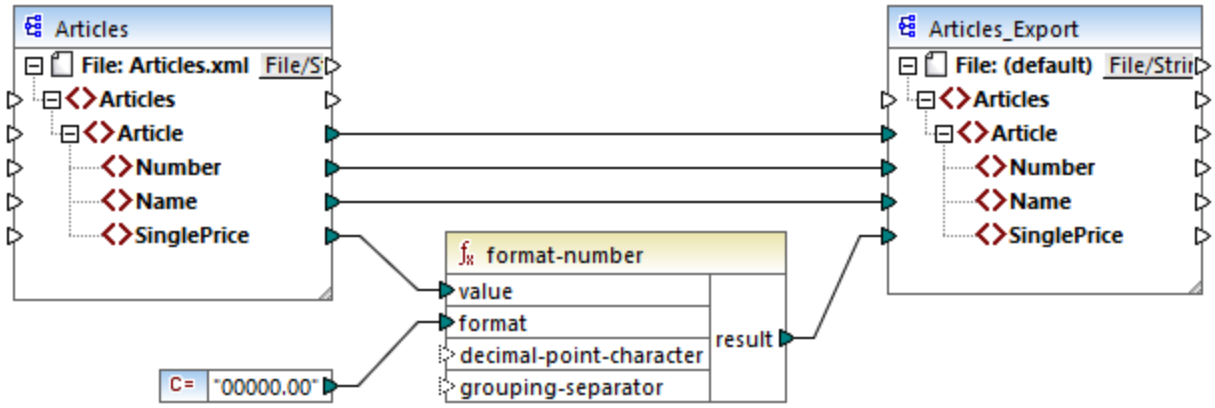
| Número | Cadena de formato | Resultado |
|---|---|---|
| 1234,5 | #,##0.00 | 1.234,50 |
| 123,456 | #,##0.00 | 123,46 |
| 1000000 | #,##0.00 | 1.000.000,00 |
| -59 | #,##0.00 | -59,00 |
| 1234 | ###0.0### | 1234,0 |
| 1234,5 | ###0.0### | 1234,5 |
| .00025 | ###0.0### | 0,0003 |
| .00035 | ###0.0### | 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*

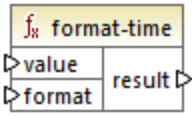
En consecuencia, los valores de destino paran a ser:

```
00025,00
00002,30
00034,00
00057,50
```

Puede encontrar el archivo de diseño de esta asignación en la ruta:
**<Documentos>\Altova\MapForce2025\MapForceExamples\PreserveFormatting.mfd**.

### 5.9.7.2.5 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.


© 2019-2025 Altova GmbH

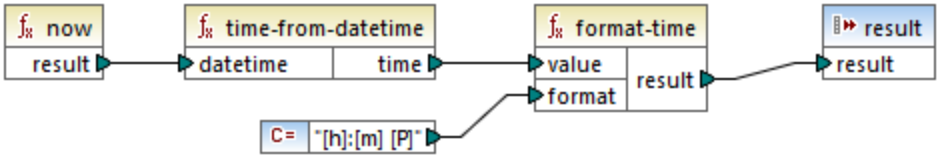
## Parámetros

| 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 556 . |

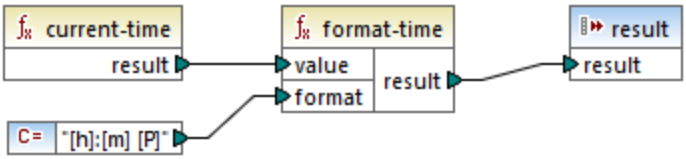
## Ejemplo

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."

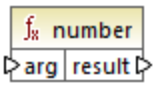



Tenga en cuenta que la asignación anterior se ha diseñado para Java, C#, C++ y el motor de ejecución integrado. Para obtener el mismo resultado en XSLT 2.0 use esta asignación:




### 5.9.7.2.6 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

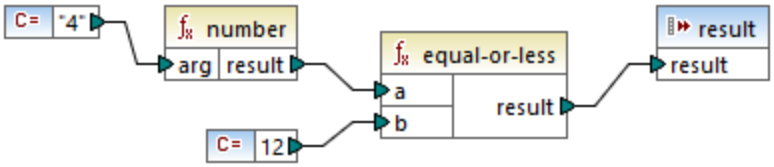
Built-in, C++, C#, Java, XQuery#, XSLT 1.0 y, XSLT 1.0, XSLT 2.0, XSLT 3.0.

## Parámetros

| Argumento | Descripción |
|---|---|
| **arg** | Argumento obligatorio. Indica el valor que se va a convertir. |

## Ejemplo

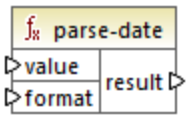
En el ejemplo siguiente la primera constante es de tipo `string` y contiene la cadena "4". La segunda constante contiene la constante numérica 12. Para que los dos valores se puedan comparar como números deben ser del mismo tipo.




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".

### 5.9.7.2.7 parse-date

Convierte una cadena en una fecha. Esta función se basa en la función parse-dateTime 565 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. |

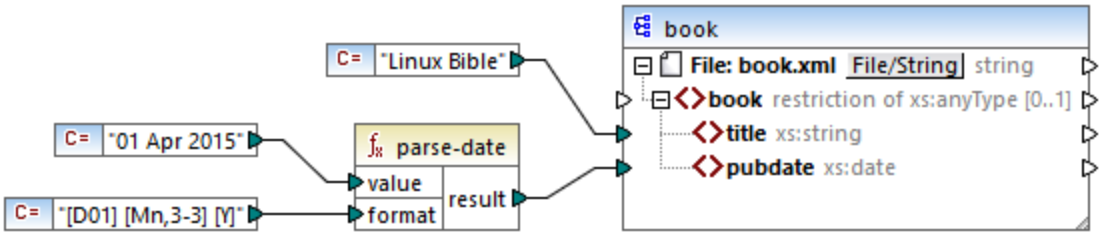

© 2019-2025 Altova GmbH

| Argumento | Descripción |
|---|---|
| **format** | Argumento obligatorio. Indica el formato de la cadena. Este argumento se usa igual que el argumento **format** de la parse-dateTime 565 . |

## 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
- **`[Y]`** es el año

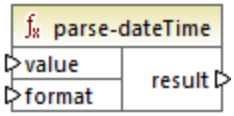



El resultado es el siguiente (sin incluir las declaraciones XML y de espacio de nombres):

```
<book>
   <title>Linux Bible</title>
   <pubdate>2015-04-01</pubdate>
</book>
```

## 5.9.7.2.8 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) |
| **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 |


© 2019-2025 Altova GmbH

| Modificador | Descripción | Ejemplo |
|---|---|---|
| **Nn** | nombre del componente, primera letra en mayúsculas | Febrero, Marzo |

**Nota:** los modificadores *N*, *n* y *Nn* solo admiten el componente *M* (mes).

Si necesita un modificador del ancho, antepóngale una coma. El modificador del ancho es un dígito que expresa el ancho mínimo. Opcionalmente, puede agregar un guion y un dígito que exprese el ancho máximo. Por ejemplo:

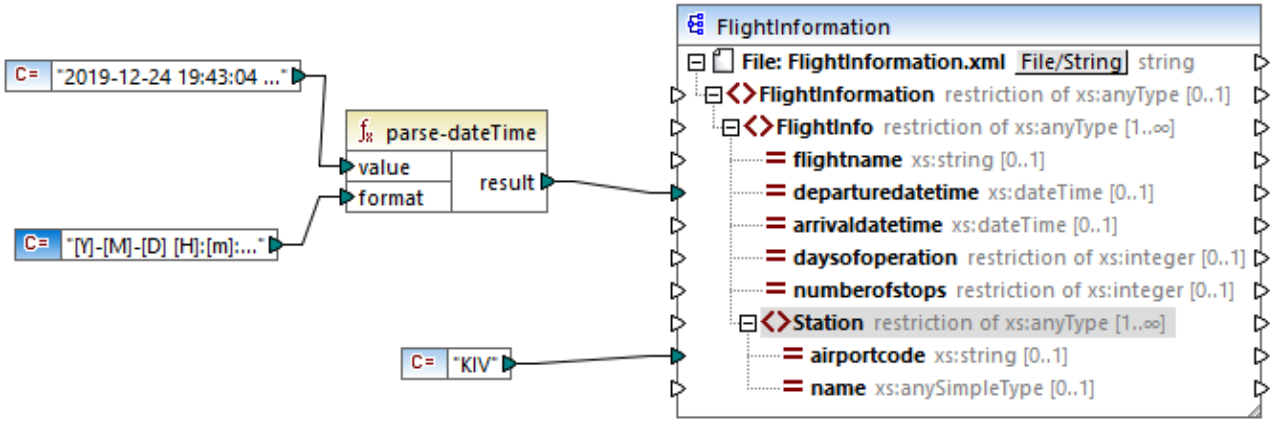
- **`[D,2]`** es el día del mes, con ceros a la izquierda (dos dígitos).
- **`[MNn,3-3]`** es el nombre del mes, escrito con tres caracteres, por ejemplo, *Ene*, *Feb*, *Mar*, etc.

A continuación puede ver algunos ejemplos más:

| 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]`**.



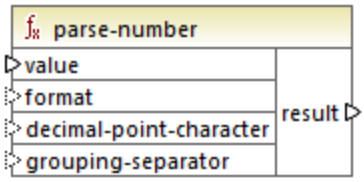

El resultado es el siguiente (sin incluir las declaraciones XML y de espacio de nombres):

```
<FlightInformation>
 <FlightInfo departuredatetime="2019-12-24T19:43:04+02:00">
```

```
 <Station airportcode="KIV"/>
 </FlightInfo>
</FlightInformation>
```

### 5.9.7.2.9 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.

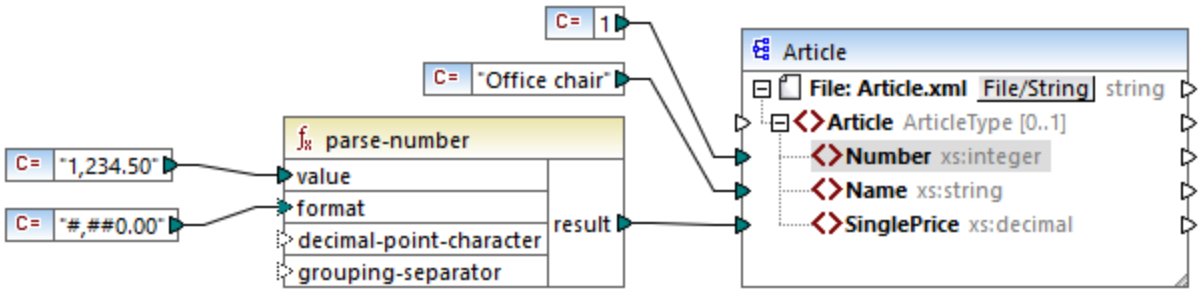
## Parámetros

| 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 [559].<br><br>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.


© 2019-2025 Altova GmbH

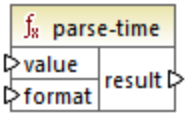



El resultado de la asignación es el siguiente (sin incluir las declaraciones XML y de espacio de nombres):

```
<Article>
   <Number>1</Number>
   <Name>Office chair</Name>
   <SinglePrice>1234.5</SinglePrice>
</Article>
```

### 5.9.7.2.10 parse-time

Convierte una cadena en un valor `xs:time`. Esta función se basa en la función parse-dateTime 565 e ignora el componente fecha.




## Lenguajes

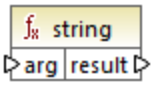
Disponible para Java, C#, C++ y el motor de ejecución integrado.

## 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 565 . |

### 5.9.7.2.11 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

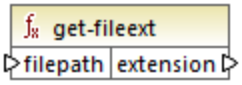
| Argumento | Descripción |
|---|---|
| **arg** | Argumento obligatorio. Indica el valor que se va a convertir. |

## 5.9.7.3 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.

### 5.9.7.3.1 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. |

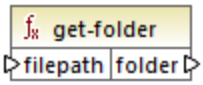

© 2019-2025 Altova GmbH

## Ejemplo

P. ej. c:\data\Sample.mfd devuelve ".mfd"

### 5.9.7.3.2 get-folder

Devuelve el nombre de la carpeta de la ruta de archivo, incluida la barra diagonal final o la barra diagonal inversa final.




## 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.

### 5.9.7.3.3 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.

### 5.9.7.3.4 mfd-filepath

Si se le llama desde la asignación principal, la función devuelve el mismo resultado que la función main-mfd-filepath 571 , es decir, la ruta completa del archivo mfd que contiene la asignación principal. Si el archivo mfd

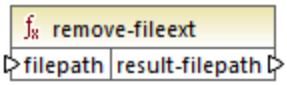
archivo está sin guardar, la función devuelve una cadena vacía. Si se le llama desde dentro de una función definida por el usuario que es *importada* por un archivo mfd, la función mfd-filepath devuelve la ruta completa del archivo mfd *importado* que contiene la definición de la función definida por el usuario.

### Lenguajes

Built-in, C++, C#, Java, XQuery#, XSLT 1.0 y, XSLT 1.0, XSLT 2.0, XSLT 3.0.

## 5.9.7.3.5 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

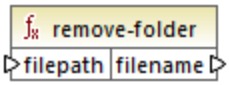
| Argumento | Descripción |
|---|---|
| **filepath** | Argumento obligatorio. Suministra la ruta de acceso al archivo que se quiere procesar. |

### Ejemplo

P. ej. c:/data/Sample.mfd devuelve c:/data/Sample.

## 5.9.7.3.6 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.


© 2019-2025 Altova GmbH

## Parámetros

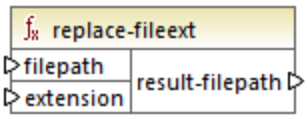
| Argumento | Descripción |
|---|---|
| **filepath** | Argumento obligatorio. Suministra la ruta de acceso al archivo que se quiere procesar. |

## Ejemplo

P. ej. c:\data\Sample.mfd devuelve ".mfd"

### 5.9.7.3.7 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

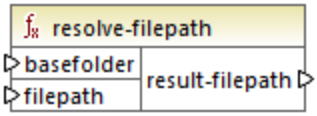
| Argumento | Descripción |
|---|---|
| **filepath** | Argumento obligatorio. Suministra la ruta de acceso al archivo que se quiere procesar. |
| **extension** | Argumento obligatorio. Suministra la estensión nueva que se debe usar. |

## 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`.

### 5.9.7.3.8 resolve-filepath

Convierte una ruta de acceso relativa en una carpeta base relativa o absoluta. La función admite "." (directorio actual) y ".." (directorio primario).




## Lenguajes

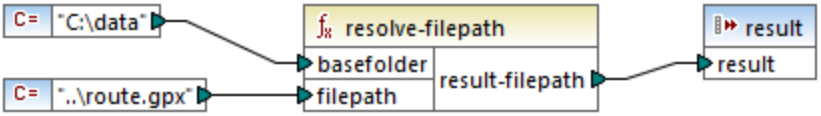
Built-in, C++, C#, Java, XQuery#, XSLT 1.0 y, XSLT 1.0, XSLT 2.0, XSLT 3.0.

## 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**.



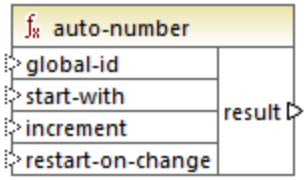

El resultado de la asignación es `C:\route.gpx`.

# 5.9.7.4 core | generator functions (generador)

La biblioteca de funciones **core / generator** incluye funciones que generan valores.

## 5.9.7.4.1 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.





© 2019-2025 Altova GmbH

El orden exacto en que el código de asignación generado llama a las funciones no está definido. MapForce puede necesitar almacenar en 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 616 en su lugar.

## Lenguajes

Built-in, C++, C#, Java, XQuery#, XSLT 1.0 y, XSLT 1.0, XSLT 2.0, XSLT 3.0.

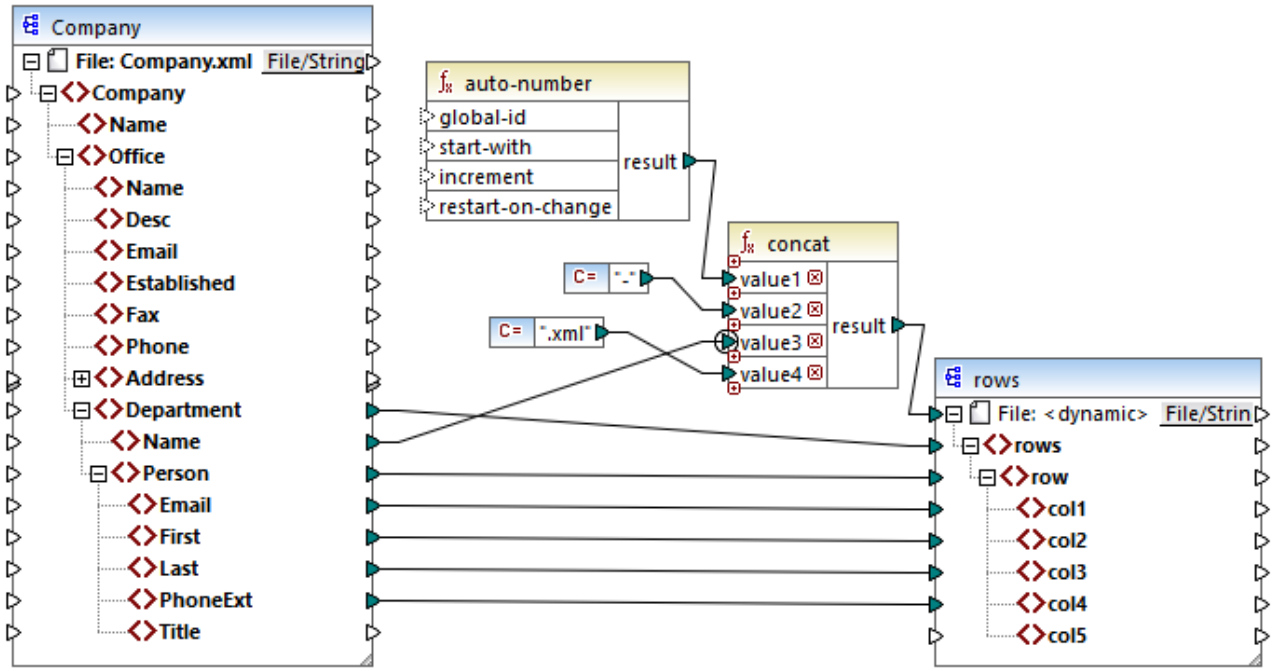
## Parámetros

| 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** descrita en Ejemplo: cambiar el contexto primario 85.

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 esta razón, cada nombre de archivo generado debe comenzar con un número secuencial, por ejemplo **1-Administración.xml**, **2-Mercadeo.xml**, y así sucesivamente.




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 destacar que al tercer parámetro de la función **concat** (el nombre del departamento) se le aplica un contexto de prioridad [89]. Esto sirve para llamar a la función **auto-number** en el contexto de cada departamento y produce los valores secuenciales necesarios. Si no se utilizara el contexto de prioridad, la función **auto-number** seguiría generando el número 1 (en ausencia de contexto) y, como consecuencia, se generarían nombres de archivo duplicados.

## 5.9.7.5 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, [432] 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

 © 2019-2025 Altova GmbH

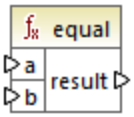
de tipos distintos (p. ej. un `integer` y una `cadena` o una `cadena` y una `fecha`), entonces el tipo de datos utilizado para la comparación es el tipo de datos de entrada más general y menos limitado de los dos.

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) 146.

### 5.9.7.5.1 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

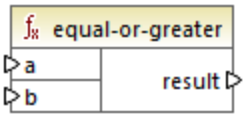
Built-in, C++, C#, Java, XQuery#, XSLT 1.0 y, XSLT 1.0, XSLT 2.0, XSLT 3.0.

## Parámetros

| Argumento | Descripción |
|---|---|
| a | Parámetro obligatorio. Suministra el primer valor de la comparación. |
| b | Parámetro obligatorio. Suministra el segundo valor de la comparación. |

### 5.9.7.5.2 equal-or-greater

El resultado es **true** si *a* es igual/mayor que *b*. De lo contrario, el resultado es **false**.




#### Lenguajes

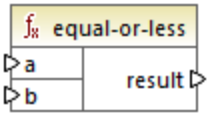
Built-in, C++, C#, Java, XQuery#, XSLT 1.0 y, XSLT 1.0, XSLT 2.0, XSLT 3.0.

#### Parámetros

| Argumento | Descripción |
|---|---|
| **a** | Parámetro obligatorio. Suministra el primer valor de la comparación. |
| **b** | Parámetro obligatorio. Suministra el segundo valor de la comparación. |

### 5.9.7.5.3 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.

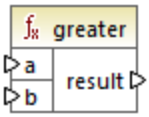
#### Parámetros

| Argumento | Descripción |
|---|---|
| **a** | Parámetro obligatorio. Suministra el primer valor de la comparación. |
| **b** | Parámetro obligatorio. Suministra el segundo valor de la comparación. |


© 2019-2025 Altova GmbH

### 5.9.7.5.4 greater

El resultado es **true** si *a* es mayor que *b*. De lo contrario devuelve **false**.




#### Lenguajes

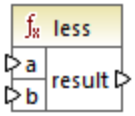
Built-in, C++, C#, Java, XQuery#, XSLT 1.0 y, XSLT 1.0, XSLT 2.0, XSLT 3.0.

#### Parámetros

| Argumento | Descripción |
|---|---|
| **a** | Parámetro obligatorio. Suministra el primer valor de la comparación. |
| **b** | Parámetro obligatorio. Suministra el segundo valor de la comparación. |

### 5.9.7.5.5 less

El resultado es **true** si *a* es menor que *b*. De lo contrario devuelve **false**.




#### Lenguajes

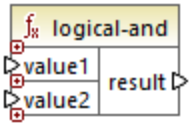
Built-in, C++, C#, Java, XQuery#, XSLT 1.0 y, XSLT 1.0, XSLT 2.0, XSLT 3.0.

#### Parámetros

| Argumento | Descripción |
|---|---|
| **a** | Parámetro obligatorio. Suministra el primer valor de la comparación. |
| **b** | Parámetro obligatorio. Suministra el segundo valor de la comparación. |

### 5.9.7.5.6 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 463.




## Lenguajes

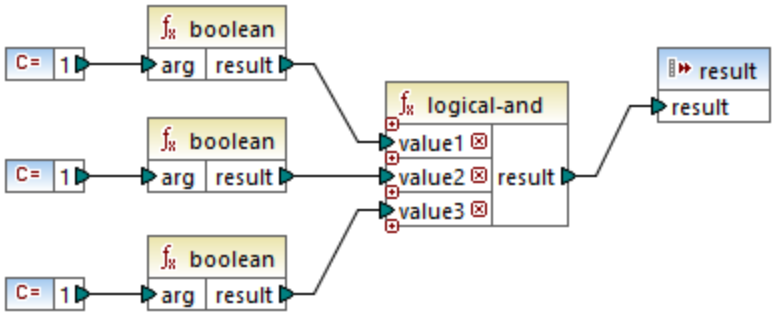
Built-in, C++, C#, Java, XQuery#, XSLT 1.0 y, XSLT 1.0, XSLT 2.0, XSLT 3.0.

## 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**.

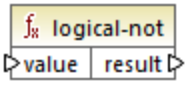



Véase también el apartado Ejemplo: búsqueda y concatenación 503.

### 5.9.7.5.7 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**.

 © 2019-2025 Altova GmbH




## Lenguajes

Built-in, C++, C#, Java, XQuery#, XSLT 1.0 y, XSLT 1.0, XSLT 2.0, XSLT 3.0.

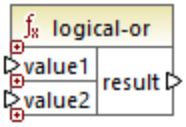
## Parámetros

| Argumento | Descripción |
|---|---|
| **value** | Parámetro obligatorio. Suministra el valor de entrada. |

### 5.9.7.5.8 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 463.




## Lenguajes

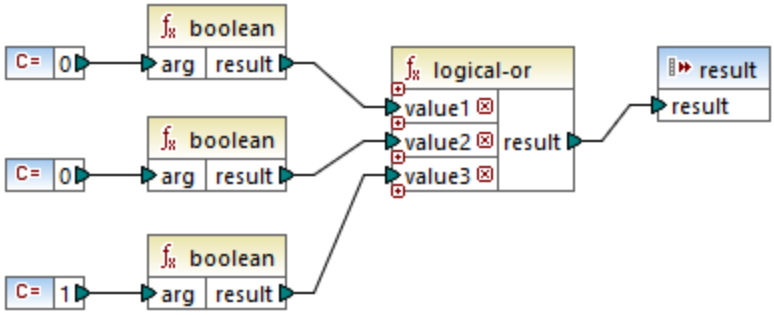
Built-in, C++, C#, Java, XQuery#, XSLT 1.0 y, XSLT 1.0, XSLT 2.0, XSLT 3.0.

## Parámetros

| Argumento | Descripción |
|---|---|
| **value1** | Parámetro obligatorio. Suministra el primer valor de la comparación. |
| **value2** | Parámetro obligatorio. Suministra el segundo valor de la comparación. |

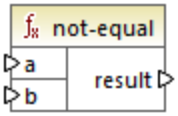
## Ejemplo

El resultado de la asignación siguiente es **true** porque al menos uno de los argumentos de la función es **true**.




### 5.9.7.5.9 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

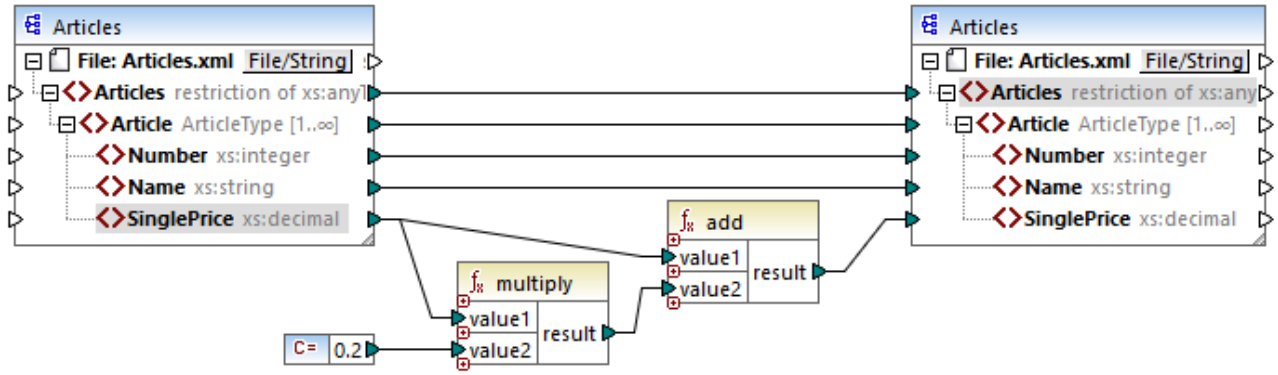
| Argumento | Descripción |
|---|---|
| **a** | Parámetro obligatorio. Suministra el primer valor de la comparación. |
| **b** | Parámetro obligatorio. Suministra el segundo valor de la comparación. |

## 5.9.7.6 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`

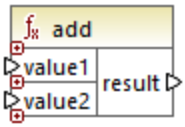

© 2019-2025 Altova GmbH




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.

### 5.9.7.6.1 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 463.




#### Lenguajes

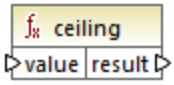
Built-in, C++, C#, Java, XQuery#, XSLT 1.0 y, XSLT 1.0, XSLT 2.0, XSLT 3.0.

#### Parámetros

| Argumento | Descripción |
|---|---|
| **value1** | Parámetro obligatorio. Suministra el primer operando |
| **value2** | Parámetro obligatorio. Suministra el segundo operando |

### 5.9.7.6.2 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

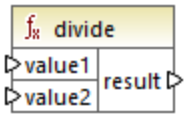
| Argumento | Descripción |
|---|---|
| **value** | Parámetro obligatorio. Suministra el valor de entrada de la función. |

### 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).

## 5.9.7.6.3 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 586 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 |

## 5.9.7.6.4 floor

El resultado es el entero mayor que sea mayor o igual que **value**.

© 2019-2025 Altova GmbH

### Lenguajes

Built-in, C++, C#, Java, XQuery#, XSLT 1.0 y, XSLT 1.0, XSLT 2.0, XSLT 3.0.

### Parámetros

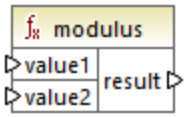
| Argumento | Descripción |
|---|---|
| **value** | Parámetro obligatorio. Suministra el valor de entrada de la función. |

### 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).

## 5.9.7.6.5 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 |

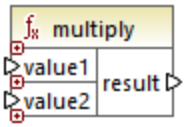
### Ejemplo

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.

### 5.9.7.6.6 multiply

El resultado es el valor decimal resultante de multiplicar **value1** por **value2**.




#### Lenguajes

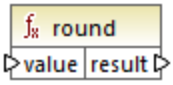
Built-in, C++, C#, Java, XQuery#, XSLT 1.0 y, XSLT 1.0, XSLT 2.0, XSLT 3.0.

#### Parámetros

| Argumento | Descripción |
|---|---|
| **value1** | Parámetro obligatorio. Suministra el primer operando |
| **value2** | Parámetro obligatorio. Suministra el segundo operando |

### 5.9.7.6.7 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.

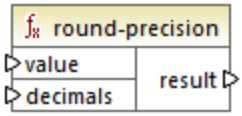
#### Parámetros

| Argumento | Descripción |
|---|---|
| **value** | Parámetro obligatorio. Suministra el valor de entrada de la función. |

### 5.9.7.6.8 round-precision

El resultado es el valor decimal del número redondeado a las cifras decimales definidas por el parámetro **decimals**.


© 2019-2025 Altova GmbH




## Lenguajes

Disponible para Java, C#, C++ y el motor de ejecución integrado.

## Parámetros

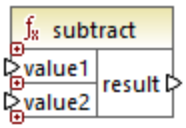
| Argumento | Descripción |
|---|---|
| **value** | Parámetro obligatorio. Suministra el valor de entrada de la función. |
| **decimals** | Parámetro obligatorio. Indica el número de decimales a los que se debe redondear. |

## 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**.

### 5.9.7.6.9 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 |

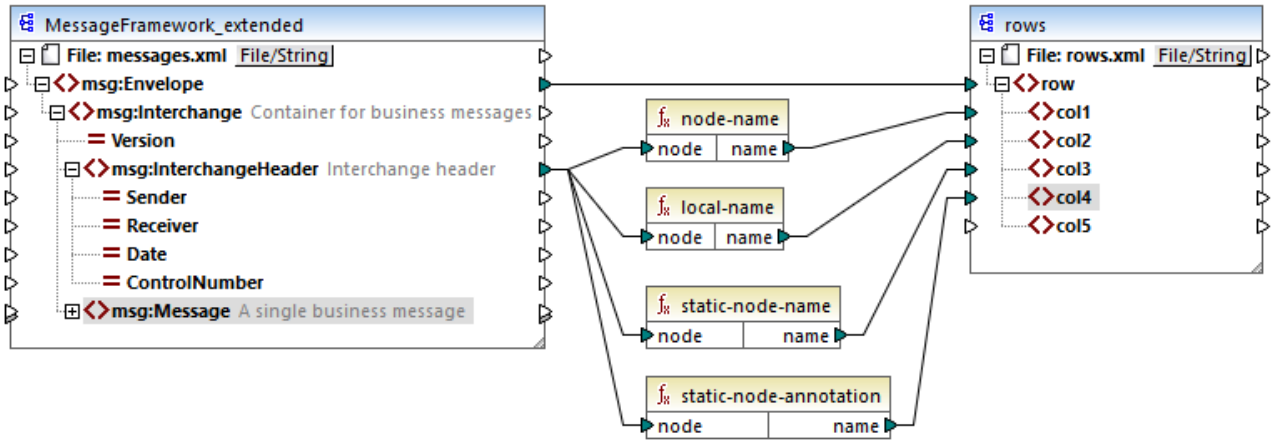
## 5.9.7.7 core | node functions (nodo)

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) 146.

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) 763.

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.



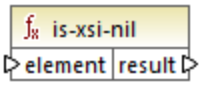

El resultado de la asignación es el siguiente (sin incluir las declaraciones XML y de espacio de nombres):

```
<row>
 <col1>msg:InterchangeHeader</col1>
 <col2>InterchangeHeader</col2>
 <col3>msg:InterchangeHeader</col3>
 <col4>Interchange header</col4>
</row>
```

 © 2019-2025 Altova GmbH

### 5.9.7.7.1 is-xsi-nil

Devuelve **true** si el nodo **element** del componente de origen tiene un atributo `xsi:nil` con el valor **true**.




## Lenguajes

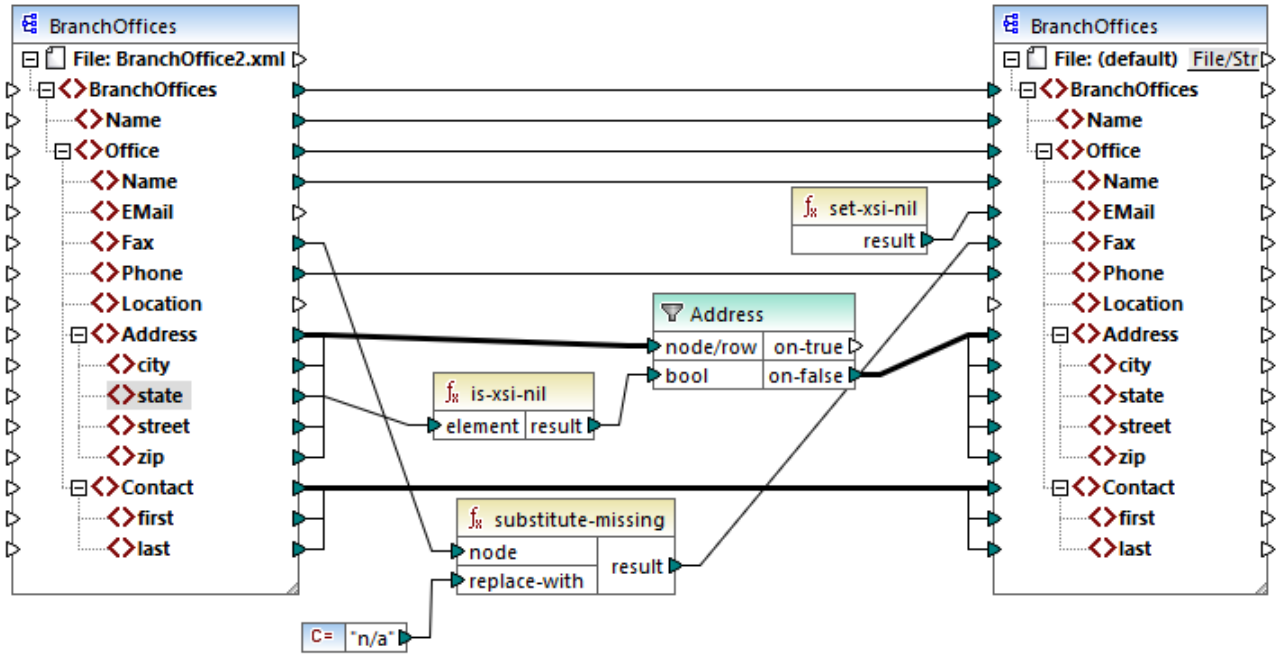
Built-in, C++, C#, Java, XQuery#, XSLT 1.0 y, XSLT 1.0, XSLT 2.0, XSLT 3.0.

## 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\MapForce2025\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 sólo 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:

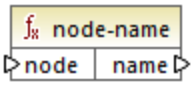
```
<BranchOffices>
 <Name>Nanonull</Name>
 <Office>
 <Name>Nanonull Research Outpost</Name>
 <EMail xsi:nil="true"/>
 <Fax>n/a</Fax>
 <Phone>+8817 3141 5926</Phone>
 <Contact>
 <first>Scott</first>
 <last>Amundsen</last>
 </Contact>
 </Office>
</BranchOffices>
```

### 5.9.7.7.2 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.

 © 2019-2025 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

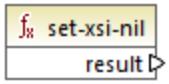
Built-in, C++, C#, Java, XQuery, XSLT 2.0, XSLT 3.0.

## Parámetros

| Argumento | Descripción |
|---|---|
| **nodo** | Parámetro obligatorio. Conecta la entrada con el nodo cuyo nombre se quiere obtener. |

### 5.9.7.7.3 set-xsi-nil

Establece el nodo de destino en xsi:nil.



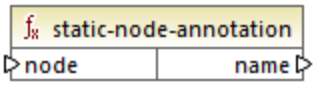

## Lenguajes

Built-in, C++, C#, Java, XQuery, XSLT 2.0, XSLT 3.0.

### 5.9.7.7.4 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 489 directamente conectada a un parámetro 492, 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.

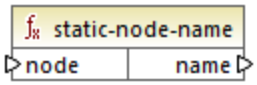
### Parámetros

| Argumento | Descripción |
|---|---|
| **nodo** | Parámetro obligatorio. Conecta el valor de entrada con el nodo cuyo anotación se quiere obtener. |

## 5.9.7.7.5 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 489 directamente conectada a un parámetro 492, 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

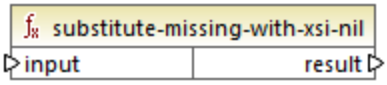
Built-in, C++, C#, Java, XQuery#, XSLT 1.0 y, XSLT 1.0, XSLT 2.0, XSLT 3.0.

### Parámetros

| Argumento | Descripción |
|---|---|
| **nodo** | Parámetro obligatorio. Conecta la entrada con el nodo cuyo nombre se quiere obtener. |

## 5.9.7.7.6 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.




 © 2019-2025 Altova GmbH

### Lenguajes

Built-in, C++, C#, Java, XQuery, XSLT 2.0, XSLT 3.0.

### Parámetros

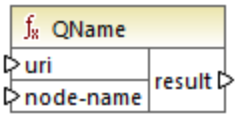
| Argumento | Descripción |
|---|---|
| **input** | Parámetro obligatorio. Conecta la entrada con el nodo cuyo nombre se quiere obtener. |

## 5.9.7.8 core | QName functions (QName)

Las funciones QName permiten manipular los nombres completos (QName) en documentos XML.

### 5.9.7.8.1 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

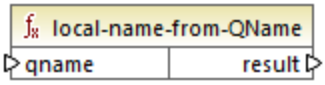
Built-in, C++, C#, Java, XQuery, XSLT 2.0, XSLT 3.0.

### Parámetros

| Nombre | Descripción |
|---|---|
| **uri** | Obligatorio. Suministra el URI. |
| **node-name** | Obligatorio. Suministra el nombre del nodo. |

### 5.9.7.8.2 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`. |

### 5.9.7.8.3 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. |

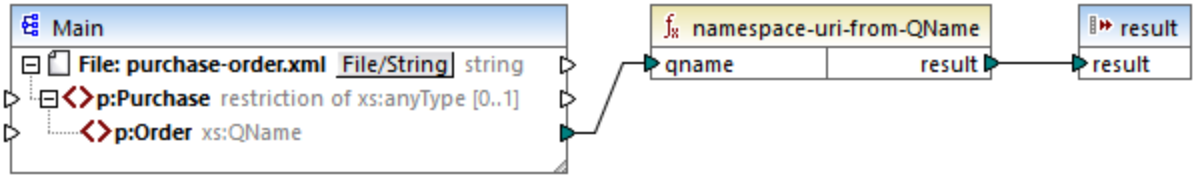
## Ejemplo

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>
```

 © 2019-2025 Altova GmbH

Esta asignación procesa el valor Qname y obtiene el URI de espacio de nombre:



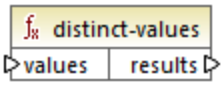

El resultado de esta asignación es `http://NamespaceTest.com/Order`.

## 5.9.7.9 core | sequence functions (secuencia)

Las funciones de secuencia permiten procesar secuencias [81] de entrada y agrupar su contenido.

### 5.9.7.9.1 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 [81] 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. |

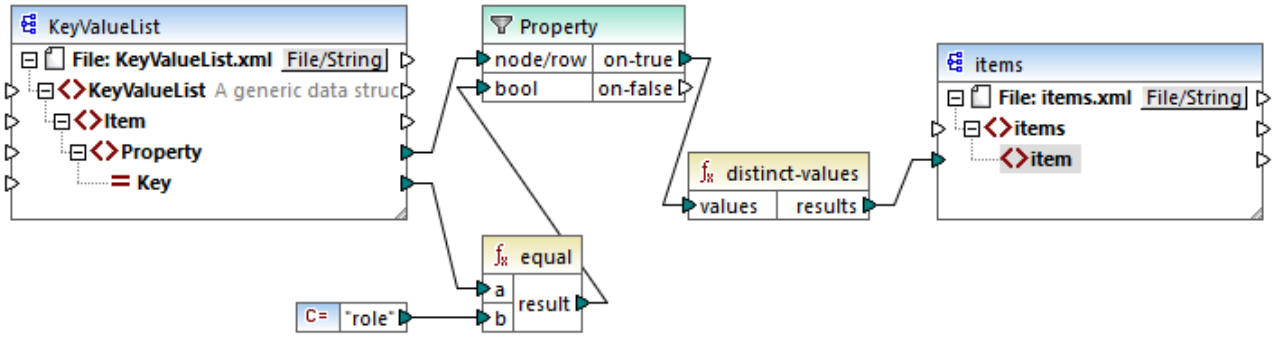
#### Ejemplo

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 sólo 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.

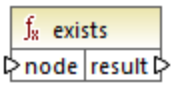
El resultado de la asignación es el siguiente:

 © 2019-2025 Altova GmbH

```
<items>
 <item>Manager</item>
 <item>Programmer</item>
 <item>Support</item>
</items>
```

### 5.9.7.9.2 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 432 por registros que tienen (o que no tienen) un elemento o atributo secundario.




## Lenguajes

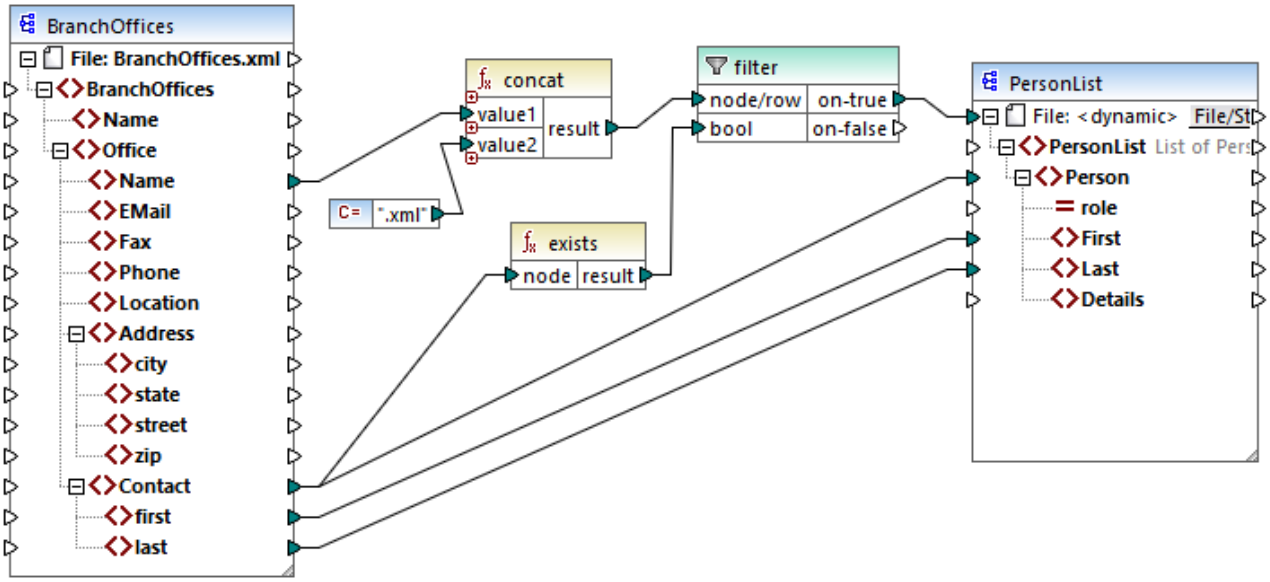
Built-in, C++, C#, Java, XQuery#, XSLT 1.0 y, XSLT 1.0, XSLT 2.0, XSLT 3.0.

## 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\MapForce2025\MapForceExamples**.




*PersonListsForAllBranchOffices.mfd*

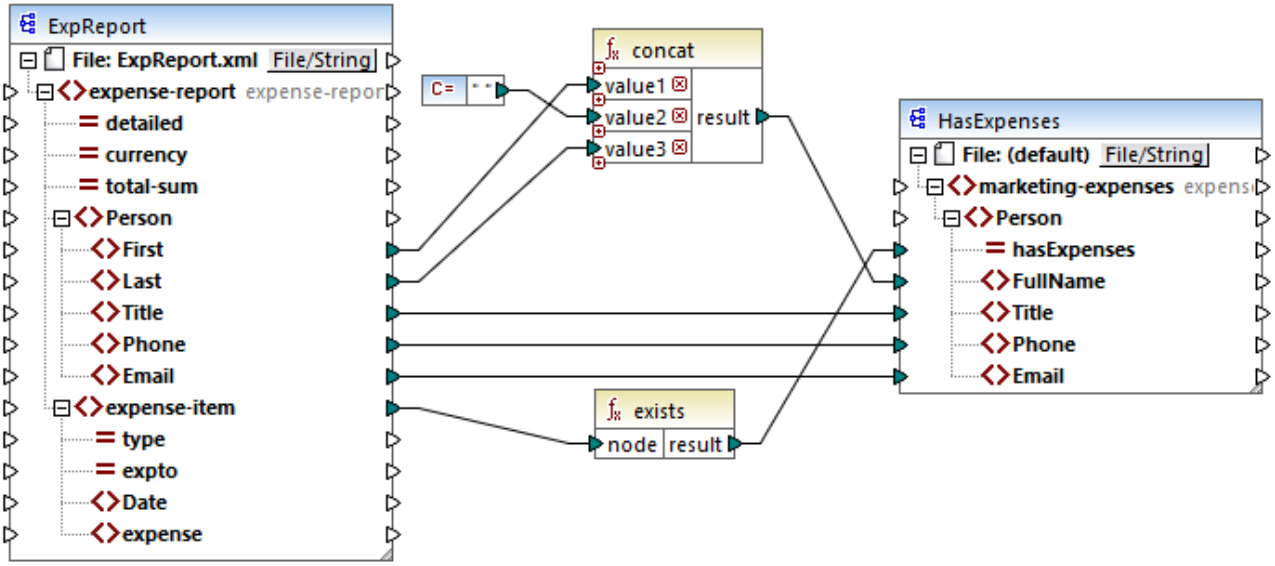
En el archivo de origen **BranchOffices.xml** hay tres elementos **Office**. En concreto, uno de ellos no tienen ningún elemento secundario **Contact**. La asignación tiene varios objetivos:

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 784 .

Otro ejemplo sería esta asignación:
**<Documentos>\Altova\MapForce2025\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.

 © 2019-2025 Altova GmbH

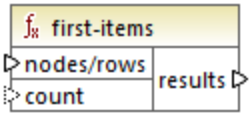



*HasMarketingExpenses.mfd*

Véase también el apartado Ejemplo: excepción cuando un nodo no existe 458.

### 5.9.7.9.3 first-items

Devuelve los *X* últimos nodos de la secuencia nodes/rows, siendo *X* el número dado por el parámetro **count**.




## Lenguajes

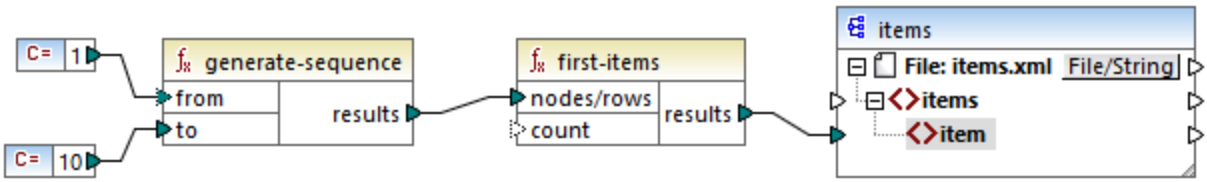
Built-in, C++, C#, Java, XQuery, XSLT 2.0, XSLT 3.0.

## Parámetros

| Nombre | Descripción |
|---|---|
| **nodos/filas** | Esta entrada debe recibir una conexión desde un elemento de la asignación que suministre una secuencia 81 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.



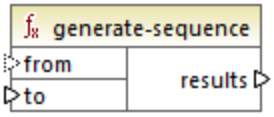

Como el argumento **count** no tiene ningún valor, se aplica el valor predeterminado **1**. El resultado es que en la salida de la asignación sólo se genera el primer valor de la secuencia:

```
<items>
 <item>1</item>
</items>
```

Para ver un ejemplo más realista consulte la asignación **FindHighestTemperatures.mfd** de Pasar parámetros a la asignación 369.

### 5.9.7.9.4 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.

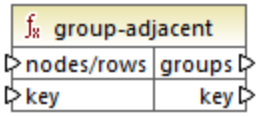
## Parámetros

| 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). |


© 2019-2025 Altova GmbH

### 5.9.7.9.5 group-adjacent

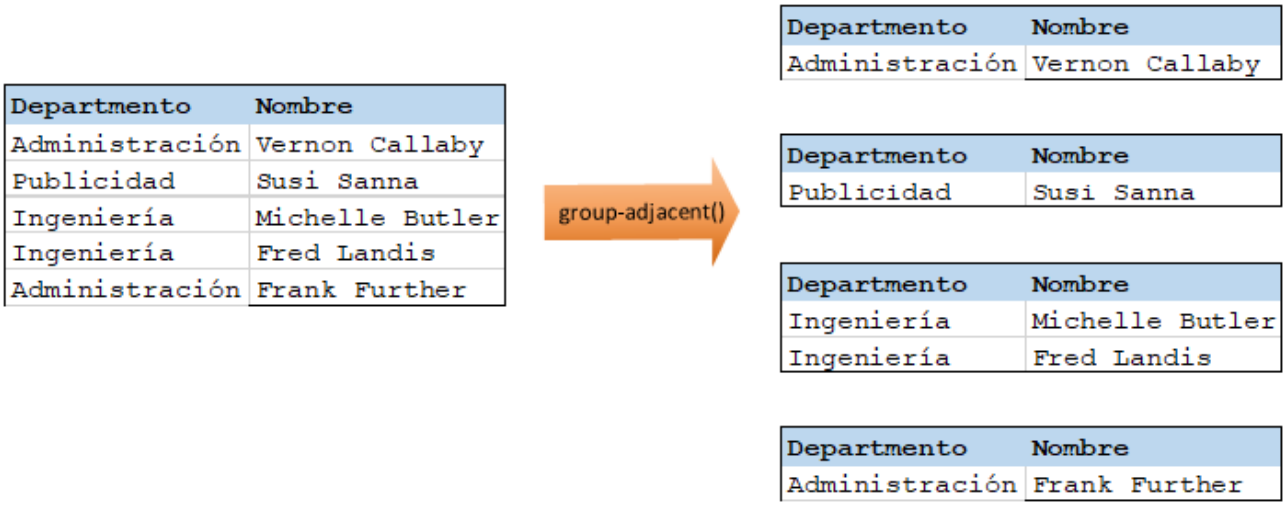
La función **group-adjacent** agrupa los elementos conectados a los datos de entrada de **nodes/rows** con la clave conectada a los datos de entrada de **key**. Observe que si los elementos que comparten la misma clave no son adyacentes, esta clave los coloca en grupos separados. Si varios elementos 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

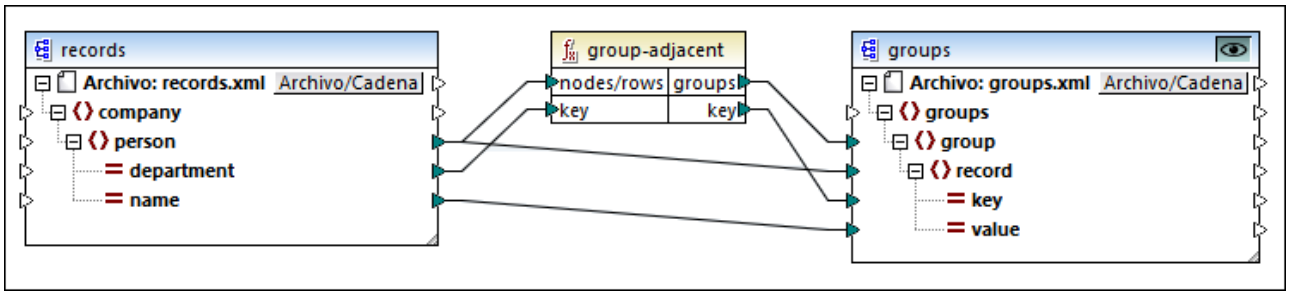
| Nombre | Descripción |
|---|---|
| **nodos/filas** | Esta entrada debe recibir una conexión desde un elemento de la asignación que suministre una secuencia 81 de cero o más valores. Por ejemplo, la conexión puede provenir de un elemento XML de origen, un campo CSV, un registro de BD, etc. |
| **key** | 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>
```

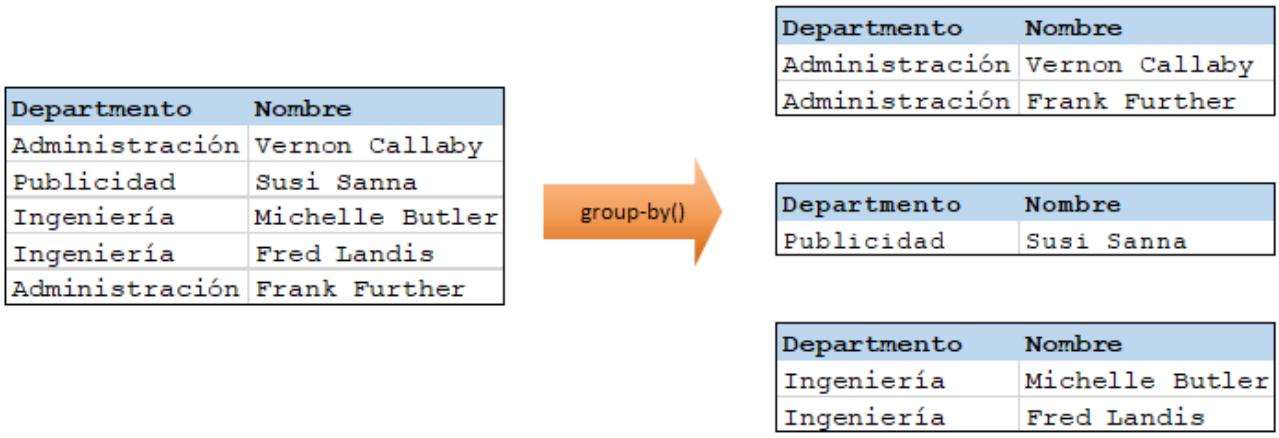

© 2019-2025 Altova GmbH

```
 <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\MapForce2025\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.

### 5.9.7.9.6 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

Built-in, C++, C#, Java, XSLT 2.0, XSLT 3.0.

## Parámetros

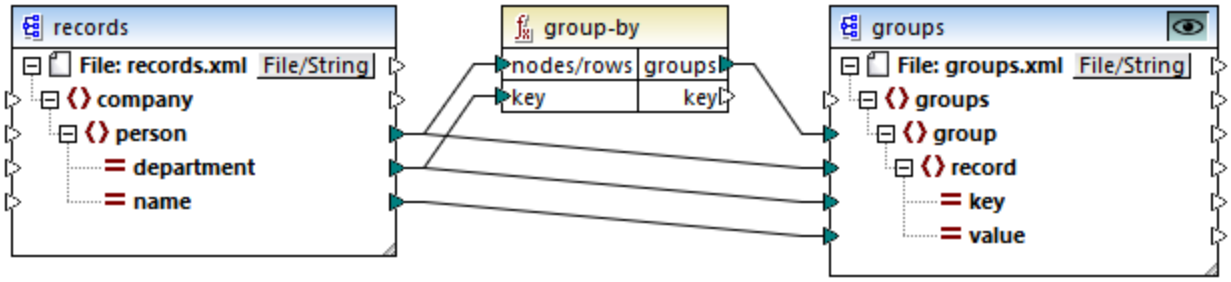
| Nombre | Descripción |
|---|---|
| **nodos/filas** | Esta entrada debe recibir una conexión desde un elemento de la asignación que suministre una secuencia[81] de cero o más valores. Por ejemplo, la conexión puede provenir de un elemento XML de origen, un campo CSV, un registro de BD, etc. |
| **key** | La clave conforme a la cual se agrupan los elementos. |

## Ejemplo 1

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. 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\MapForce2025\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.


© 2019-2025 Altova GmbH

## Ejemplo 2

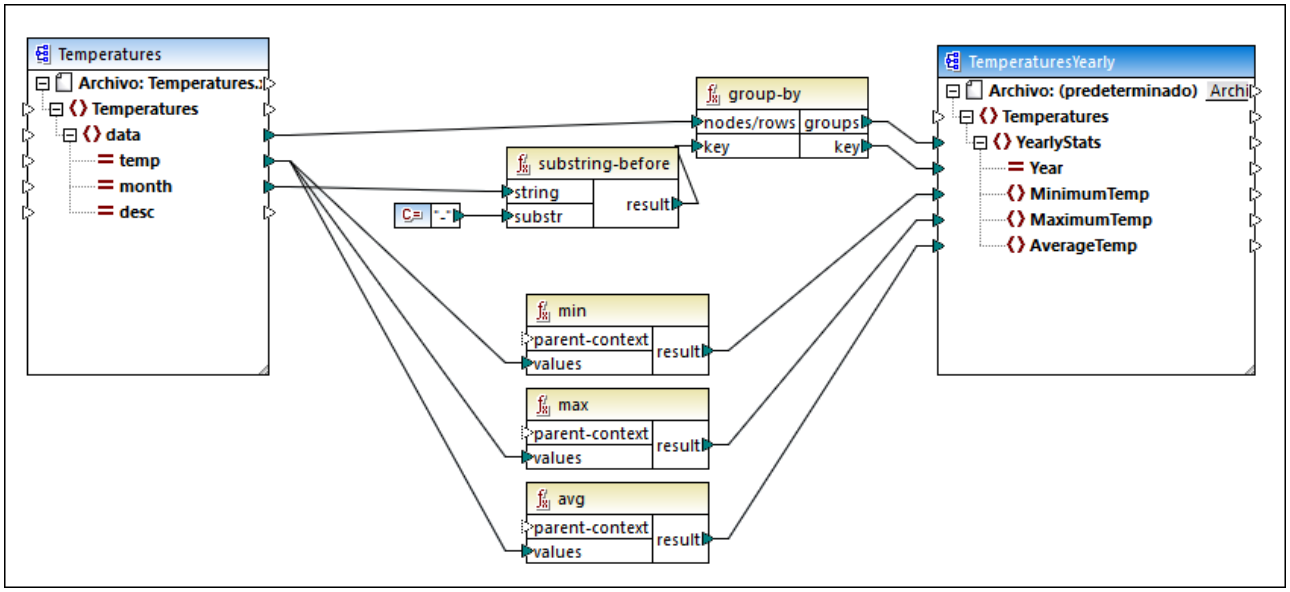
Este ejemplo muestra cómo agrupar registros con la ayuda de la función **group-by** e ilustra cómo añadir datos. Este ejemplo viene acompañado por una asignación de muestra que encontrará en la siguiente ruta: **<Documentos>\Altova\MapForce2025\MapForceExamples\GroupTemperaturesByYear.mfd**. Esta asignación 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" />
   <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, usamos la función **group-by**. Para conseguir el segundo requisito, usamos funciones agregadas **min** 550, **max** 549 y **avg** 547.




*GroupTemperaturesByYear.mfd*

La manera de ejecutar asignaciones de datos con MapForce Server (y el método recomendado para empezar a leerlas) consiste en observar el elemento de mayor nivel del componente de destino. En este ejemplo se creará un elemento **YearlyStats** por cada grupo que devuelva la función **group-by**. La función **group-by** 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** 631 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 del resultado de la asignación, haga clic en la pestaña **Resultados**. Observe que el número de grupos coincide con el número de años obtenido leyendo el archivo fuente, 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>
```

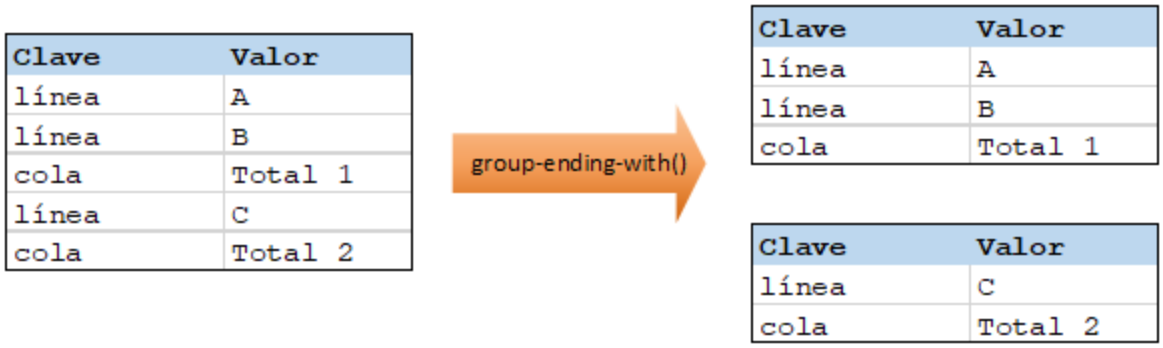
 © 2019-2025 Altova GmbH

```
    </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.

### 5.9.7.9.7 group-ending-with

La función **group-ending** toma como argumento una condición booleana. Si la condición booleana se cumple, entonces 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

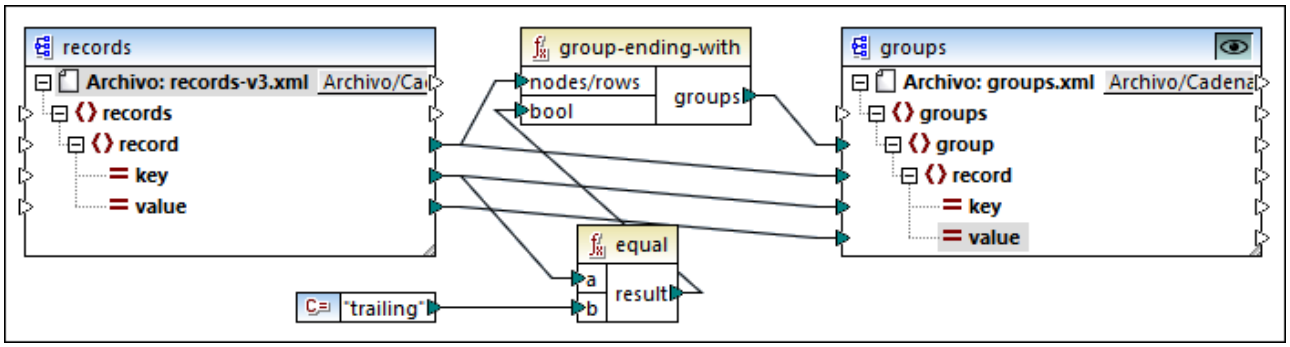
| Nombre | Descripción |
|---|---|
| **nodos/filas** | Esta entrada debe recibir una conexión desde un elemento de la asignación que suministre una secuencia 81 de cero o más valores. Por ejemplo, la conexión puede provenir de un elemento XML de origen, un campo CSV, un registro de BD, etc. |
| **bool** | 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>
```

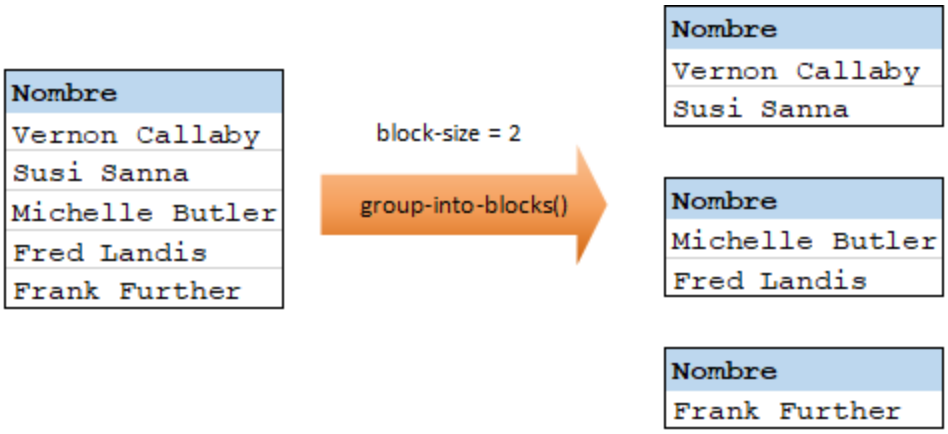
Este ejemplo, junto con otros ejemplos de agrupación, es parte de esta asignación: **<Documentos>\Altova\MapForce2025\MapForceExamples\Tutorial\GroupingFunctions.mfd**. Antes de hacer clic en la pestaña **Resultados** recuerde hacer clic en el botón **Vista previa** de la función que quiere comprobar.

### 5.9.7.9.8 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

 © 2019-2025 Altova GmbH

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 sólo puede contener uno.




## Lenguajes

La función está disponible para XSLT 2.0, XSLT 3.0, Java, C#, C++ y el motor de ejecución integrado.

## Parámetros

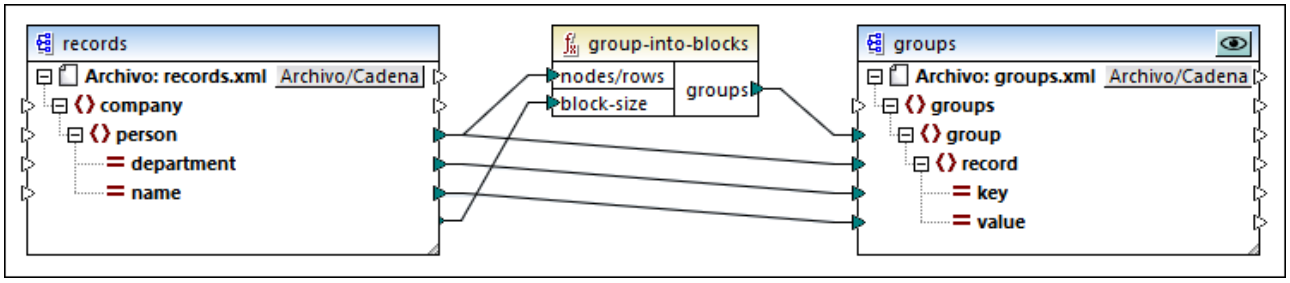
| Nombre | Descripción |
|---|---|
| **nodos/filas** | Esta entrada debe recibir una conexión desde un elemento de la asignación que suministre una secuencia [81] de cero o más valores. Por ejemplo, la conexión puede provenir de un elemento XML de origen, un campo CSV, un registro de BD, etc. |
| **block-size** | 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>
```

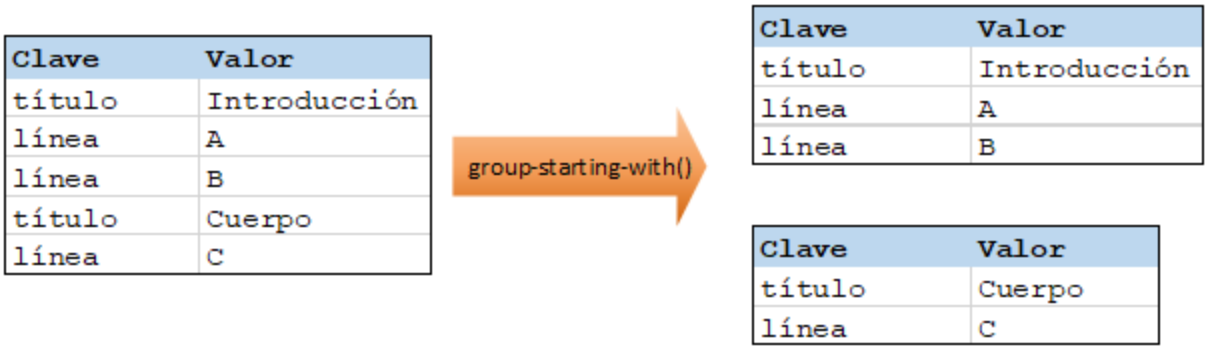
Observe que el último grupo contiene un único elemento, ya que el número total de elementos (5) no se puede dividir entre 2.

Este ejemplo, junto con otros ejemplos de agrupación, es parte de esta asignación: **<Documentos>\Altova\MapForce2025\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.

### 5.9.7.9.9 group-starting-with

La función **`group-starting-with`** toma como argumento una condición booleana. Si la condición booleana se cumple, entonces 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:

 © 2019-2025 Altova GmbH




**Nota:** si existen registros antes del primero que cumple la condición, entonces 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

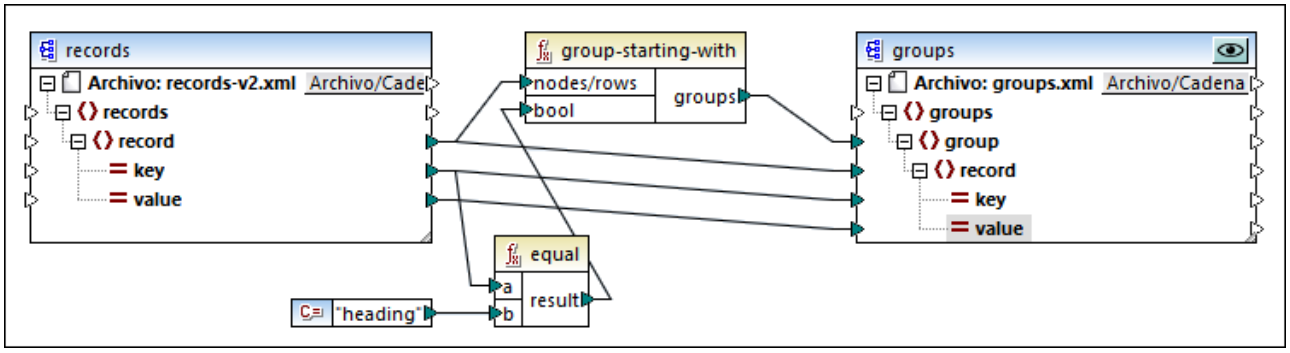
| Nombre | Descripción |
|---|---|
| **nodos/filas** | Esta entrada debe recibir una conexión desde un elemento de la asignación que suministre una secuencia [81] de cero o más valores. Por ejemplo, la conexión puede provenir de un elemento XML de origen, un campo CSV, un registro de BD, etc. |
| **bool** | 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.



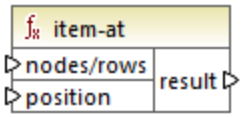

El resultado de la asignación es el siguiente:

```
<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\MapForce2025\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.

### 5.9.7.9.10 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.

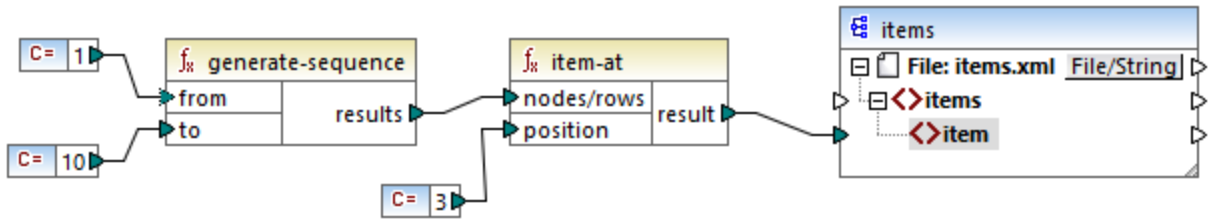

© 2019-2025 Altova GmbH

## Parámetros

| Nombre | Descripción |
|---|---|
| **nodos/filas** | Esta entrada debe recibir una conexión desde un elemento de la asignación que suministre una secuencia 81 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.

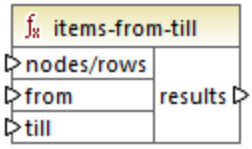



El argumento **position** tiene el valor **3**, por lo que sólo el tercer valor de la secuencia pasa al archivo de destino. En consecuencia, la salida de la asignación es la siguiente (sin incluir las declaraciones de esquema XML):

```
<items>
 <item>3</item>
</items>
```

### 5.9.7.9.11 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

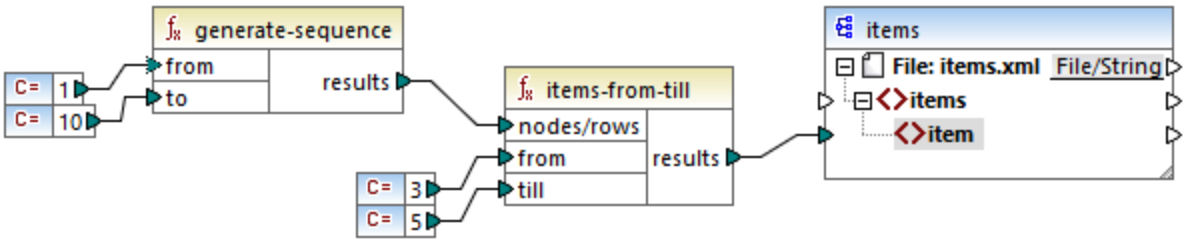
Built-in, C++, C#, Java, XQuery, XSLT 2.0, XSLT 3.0.

## Parámetros

| Nombre | Descripción |
|---|---|
| **nodos/filas** | Esta entrada debe recibir una conexión desde un elemento de la asignación que suministre una secuencia 81 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.

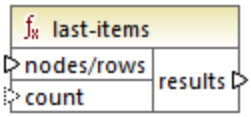



Los argumentos **from** y **till** tienen los valores **3** y **5**, por lo que sólo se pasan al componente de destino el subconjunto de valores de **3** a **5**. En consecuencia, la salida de la asignación es la siguiente (sin incluir las declaraciones de esquema XML):

```
<items>
 <item>3</item>
 <item>4</item>
 <item>5</item>
</items>
```

## 5.9.7.9.12 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.




 © 2019-2025 Altova GmbH

## Lenguajes

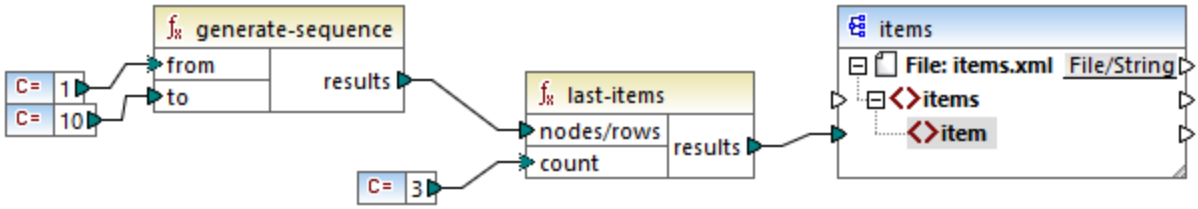
Built-in, C++, C#, Java, XQuery, XSLT 2.0, XSLT 3.0.

## Parámetros

| Nombre | Descripción |
|---|---|
| **nodos/filas** | Esta entrada debe recibir una conexión desde un elemento de la asignación que suministre una secuencia [81] 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.

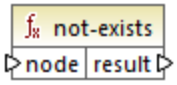



El argumento count es **3**, por lo que sólo los tres últimos valores de la secuencia se pasan al archivo de destino. En consecuencia, la salida de la asignación es la siguiente (sin incluir las declaraciones de esquema XML):

```
<items>
 <item>8</item>
 <item>9</item>
 <item>10</item>
</items>
```

### 5.9.7.9.13 not-exists

Si el nodo existe, la función devuelve **false**. De lo contrario, devuelve **true**. Esta función es la contraria a la función exists [597] pero funciona de la misma manera.



### Lenguajes

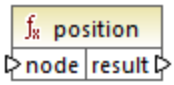
Built-in, C++, C#, Java, XQuery#, XSLT 1.0 y, XSLT 1.0, XSLT 2.0, XSLT 3.0.

### Parámetros

| Nombre | Descripción |
|---|---|
| **nodo** | El nodo cuya existencia se quiere comprobar. |

## 5.9.7.9.14 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.

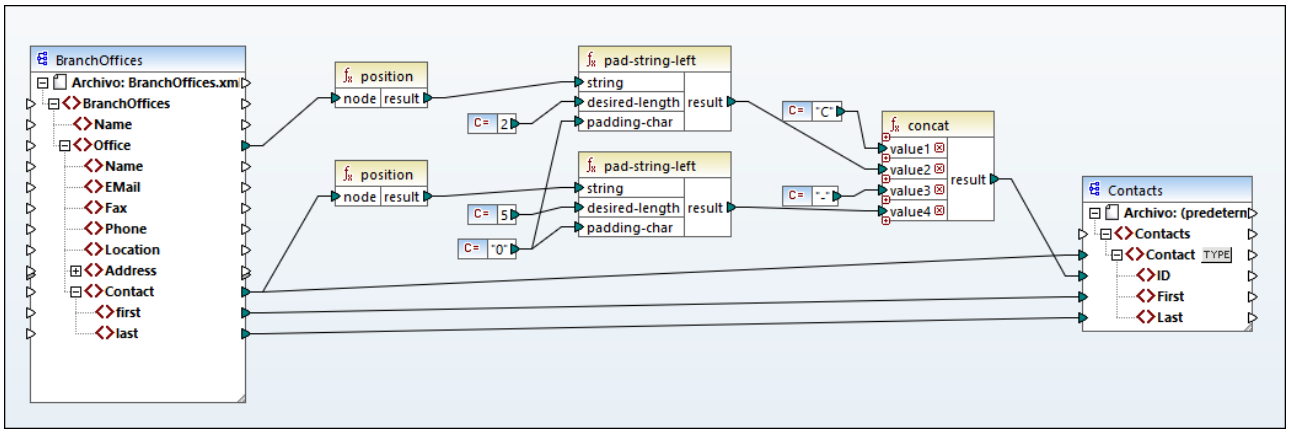
### Parámetros

| Nombre | Descripción |
|---|---|
| **node** | Esta entrada debe recibir una conexión desde un elemento de la asignación que suministre una secuencia 81 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\MapForce2025\MapForceExamples\ContactsFromBranchOffices.mfd**.


© 2019-2025 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.

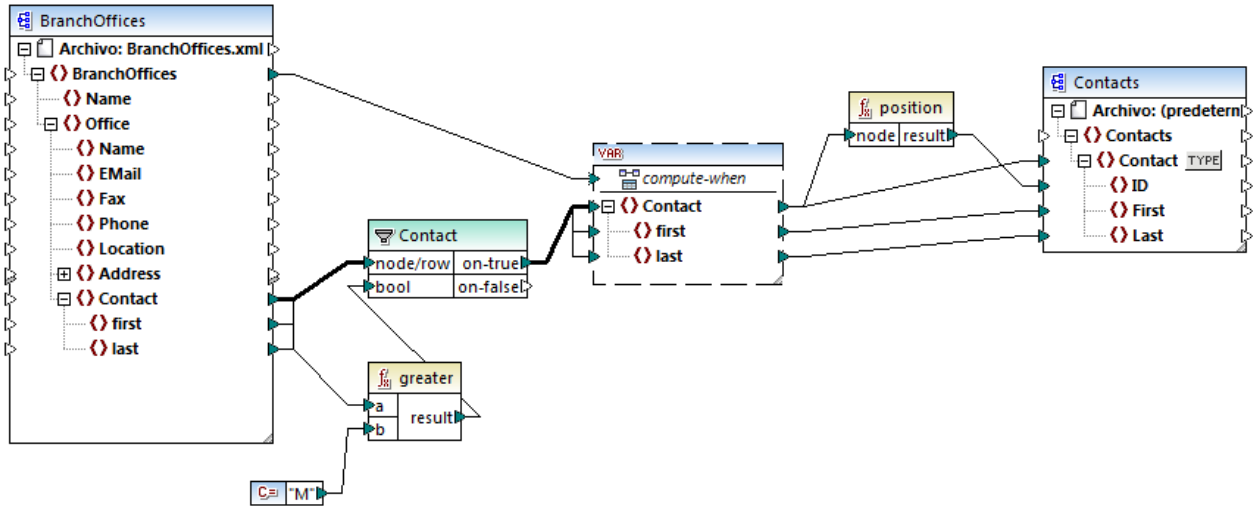
Al usar la función **`position`**, es importante considerar el contexto de asignación 82 actual. Concretamente, al ejecutar la asignación se establece el contexto inicial desde el elemento raíz del componente de destino al elemento de origen al que está conectado (aunque sea de forma indirecta mediante 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`** que aparece más abajo genera números secuenciales correspondientes 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[432]. 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 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\MapForce2025\MapForceExamples\PositionInFilteredSequence.mfd**.

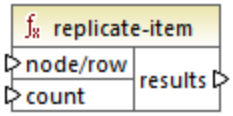



Los resultados de los componentes de variable siempre son secuencias. Por tanto, en la asignación anterior la función **position** recorre la secuencia creada por la variable y devuelve la posición de cada uno de los elementos de esa secuencia. Esta asignación se describe más detalladamente en el apartado Ejemplo: filtrar y numerar nodos[394].

 © 2019-2025 Altova GmbH

### 5.9.7.9.15 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

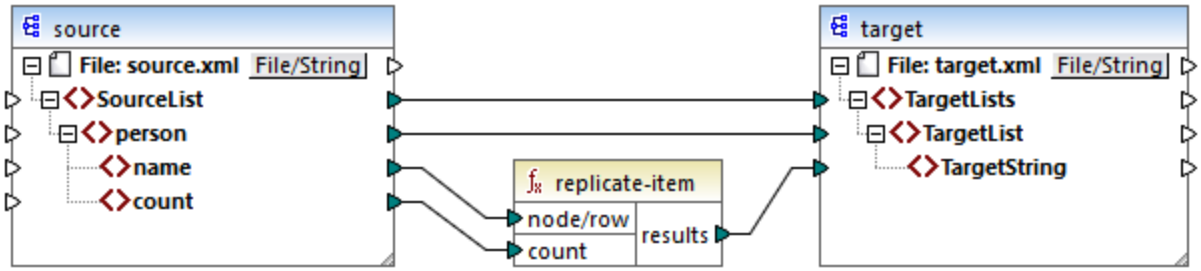
| Nombre | Descripción |
|---|---|
| **nodo/fila** | Esta entrada debe recibir una conexión desde un elemento de la asignación que suministre una secuencia 81 de cero o más valores. Por ejemplo, la conexión puede provenir de un elemento XML de origen, un campo CSV, un registro de BD, etc. |
| **count** | 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**:

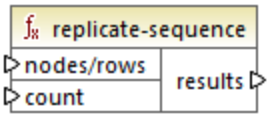



Este sería el resultado:

```
<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>
```

### 5.9.7.9.16 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.

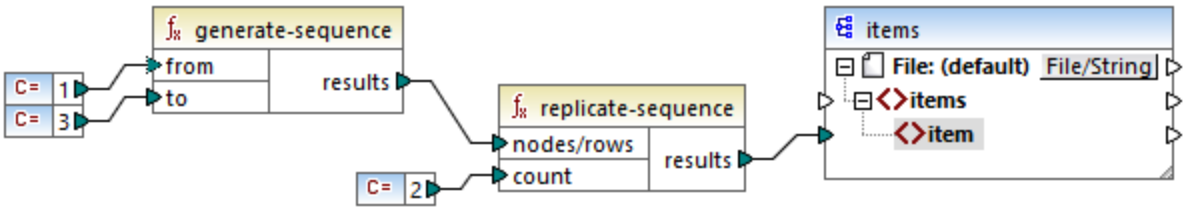

© 2019-2025 Altova GmbH

## Parámetros

| Nombre | Descripción |
|---|---|
| **nodos/filas** | Esta entrada debe recibir una conexión desde un elemento de la asignación que suministre una secuencia 81 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.

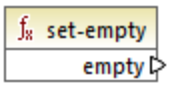



El argumento **count** tiene el valor **2**, por lo que la secuencia se replica dos veces y después pasa al archivo de destino. En consecuencia, la salida de la asignación es la siguiente (sin incluir las declaraciones de esquema XML):

```
<items>
 <item>1</item>
 <item>2</item>
 <item>3</item>
 <item>1</item>
 <item>2</item>
 <item>3</item>
</items>
```

## 5.9.7.9.17 set-empty

Devuelve una secuencia vacía.

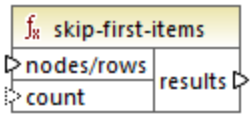



## Lenguajes

Built-in, C++, C#, Java, XQuery, XSLT 2.0, XSLT 3.0.

### 5.9.7.9.18 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

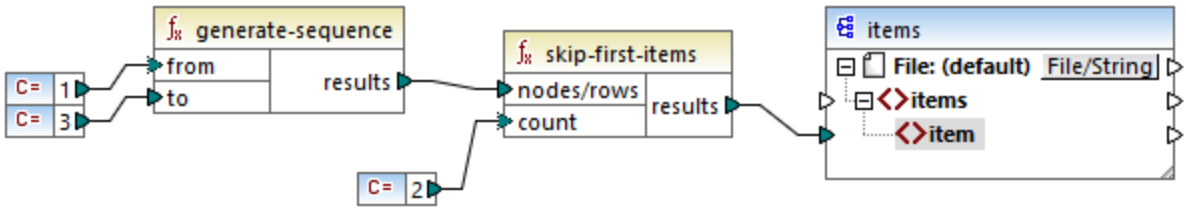
Built-in, C++, C#, Java, XQuery, XSLT 2.0, XSLT 3.0.

## Parámetros

| Nombre | Descripción |
|---|---|
| **nodos/filas** | Esta entrada debe recibir una conexión desde un elemento de la asignación que suministre una secuencia 81 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.



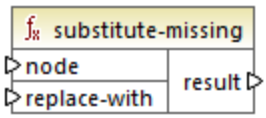

El argumento **count** tiene el valor **2**, por lo que los primeros dos elementos se omiten y el resto se pasan al archivo de destino. En consecuencia, la salida de la asignación es la siguiente (sin incluir las declaraciones de esquema XML):

```
<items>
 <item>3</item>
</items>
```


© 2019-2025 Altova GmbH

### 5.9.7.9.19 substitute-missing

Esta función es una combinación de la función exists[597] y la condición if-else.[436]. 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

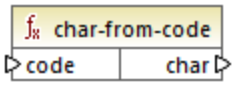
| Nombre | Descripción |
|---|---|
| **nodo** | Esta entrada debe recibir una conexión desde un elemento de la asignación que suministre una secuencia[81] de cero o más valores. Por ejemplo, la conexión puede provenir de un elemento XML de origen, un campo CSV, un registro de BD, etc. |
| **replace-with** | Este elemento de entrada estar conectado a un elemento de la asignación que proporcione el valor de reemplazo. |

## 5.9.7.10 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.

### 5.9.7.10.1 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`[625] .




#### Lenguajes

Built-in, C++, C#, Java, XQuery, XSLT 2.0, XSLT 3.0.

## Parámetros

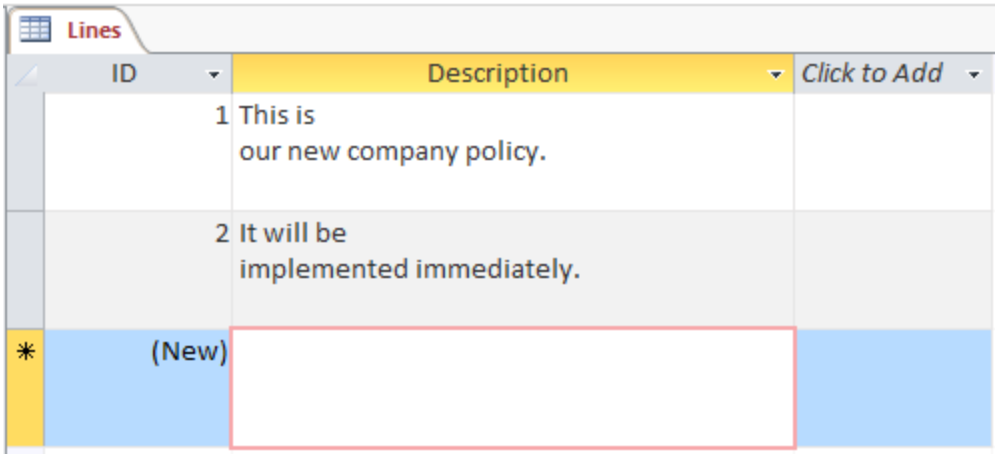
| Nombre | Descripción |
|---|---|
| **código** | El valor Unicode como número decimal. |

## Ejemplo 1

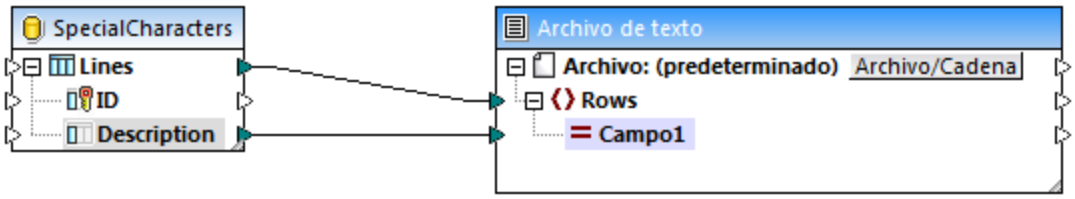
De acuerdo con los gráficos de la página web de Unicode (https://www.unicode.org/charts/), el carácter de exclamación tiene el valor hexadecimal `0021`. El valor correspondiente en formato decimal es **33**. Por tanto, si el argumento de la función `char-from-code` es `33`, el resultado será el carácter `!`.

## Ejemplo 2 (ediciones Professional y Enterprise)

En este ejemplo se puede ver cómo reemplazar caracteres especiales en una base de datos con caracteres de espacio. Imagine que tiene una base de datos SQLite 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:



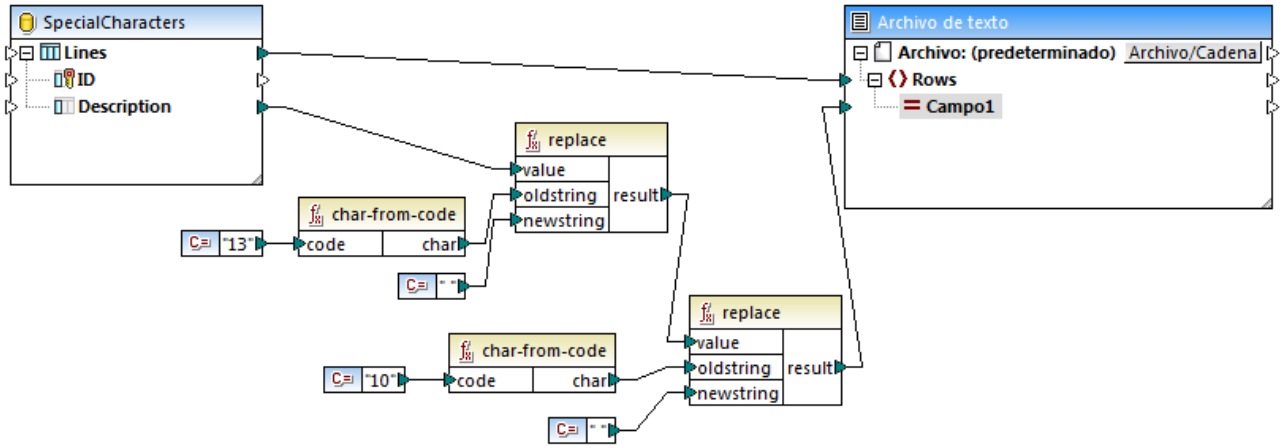

Sin embargo, como cada fila "Description" de Access contiene varias líneas separadas por caracteres CR/LF, el resultado de la asignación incluye también saltos de línea y este no es el resultado que queremos.


© 2019-2025 Altova GmbH

```
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. 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
```

## 5.9.7.10.2 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

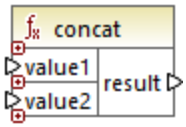
| Nombre | Descripción |
|---|---|
| **char** | El valor string de entrada. |

## 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).

### 5.9.7.10.3 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 sólo dos parámetros, pero puede añadir más. Haga clic en el icono **Agregar parámetro** ( ⊞ ) o **Eliminar parámetro** ( ⊠ para añadir o borrar parámetros.




**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 623 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.

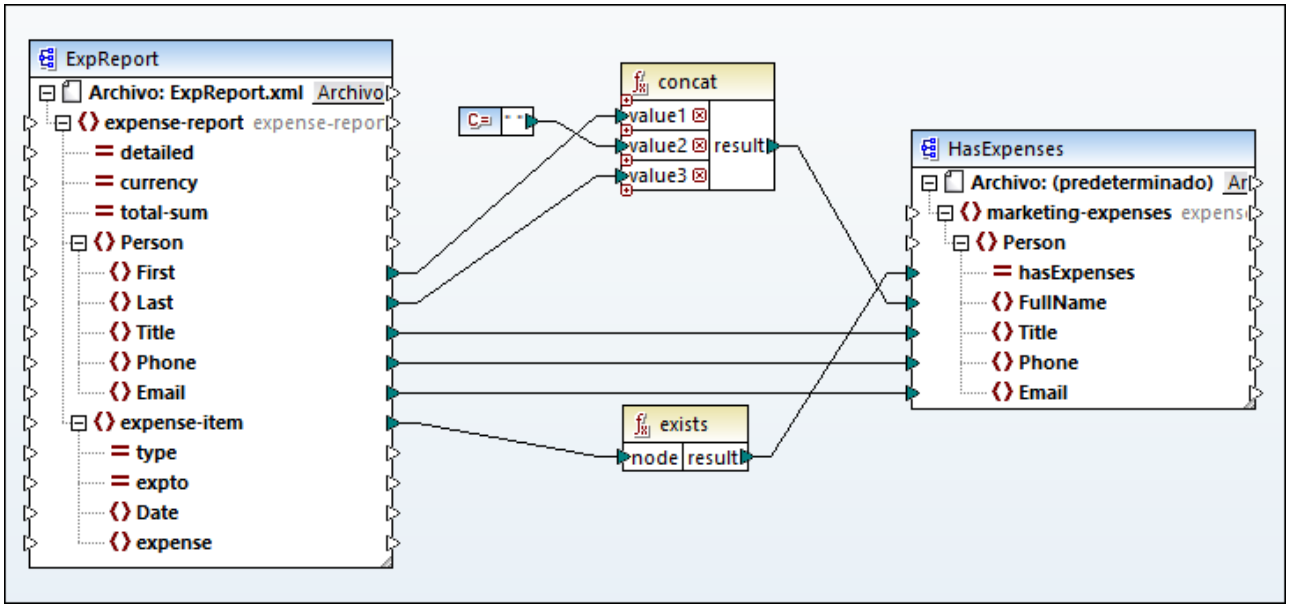
## Parámetros

| Nombre | Descripción |
|---|---|
| **value1** | El primer valor de la entrada. |
| **value2** | El segundo valor de la entrada. |
| **valueN** | El valor de *n* entrada. |


© 2019-2025 Altova GmbH

## Ejemplo

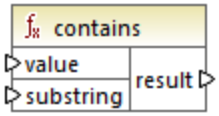
En la asignación siguiente, la función **concat** junta el nombre, la constante "" y el apellido. El valor de retorno se escribe en el elemento de destino **FullName**. La asignación de esta función se encuentra en: **<Documentos>\Altova\MapForce2025\MapForceExamples\HasMarketingExpenses.mfd**.




*HasMarketingExpenses.mfd*

## 5.9.7.10.4 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

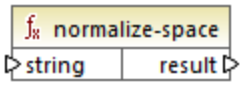
| Nombre | Descripción |
|---|---|
| **value** | El valor de entrada (el "pajar"). |
| **substring** | La subcadena que se busca (la "aguja"). |

## Ejemplo

El resultado es true si la entrada **value** es "category" y **substring** es "cat".

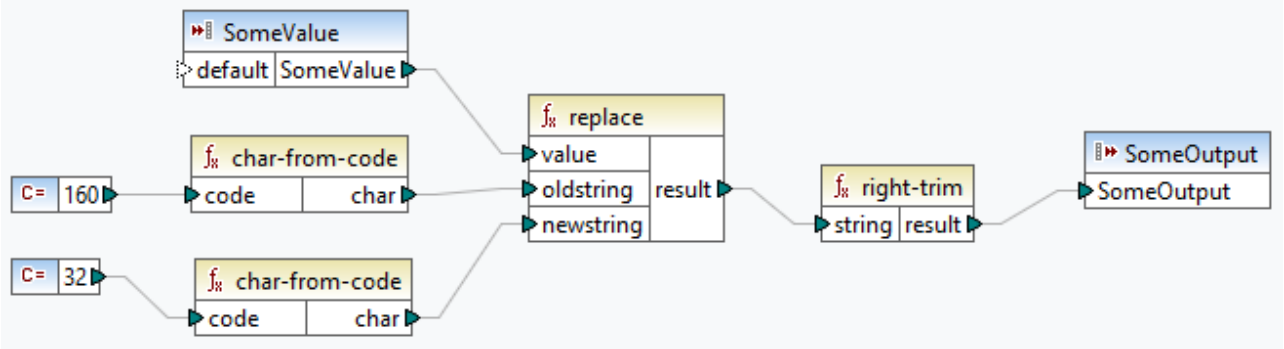
### 5.9.7.10.5 normalize-space

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. |

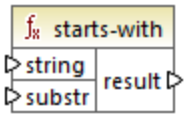
## Ejemplo

Si la cadena de entrada es **`El veloz zorro marrón`**, la función devuelve **`El veloz zorro marrón`**.


© 2019-2025 Altova GmbH

### 5.9.7.10.6 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

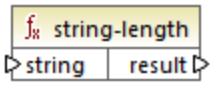
| Nombre | Descripción |
|---|---|
| **string** | La cadena de entrada. |
| **substr** | La subcadena que se debe comprobar. |

#### Ejemplo

Si la cadena de entrada **string** es `category` y la subcadena es `cat`, la función devuelve **true**.

### 5.9.7.10.7 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

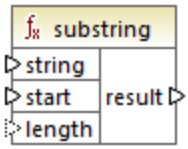
| Nombre | Descripción |
|---|---|
| **string** | La cadena de entrada. |

## 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**.

## 5.9.7.10.8 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. |

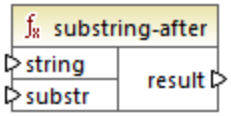
## Ejemplo

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`.

## 5.9.7.10.9 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.


© 2019-2025 Altova GmbH




## Lenguajes

Built-in, C++, C#, Java, XQuery#, XSLT 1.0 y, XSLT 1.0, XSLT 2.0, XSLT 3.0.

## Parámetros

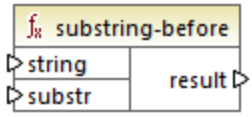
| Nombre | Descripción |
|---|---|
| **string** | La cadena de entrada. |
| **substr** | La subcadena. El resultado de la función son los caracteres que haya después de la primera ocurrencia de **substr**. |

## 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`.

### 5.9.7.10.10 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

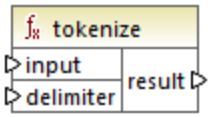
| Nombre | Descripción |
|---|---|
| **string** | La cadena de entrada. |
| **substr** | La subcadena. Los caracteres que hay antes de la primera aparición de **substr** son el resultado de la función. |

## 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`.

## 5.9.7.10.11 tokenize

El resultado es la cadena input dividida en una secuencia formada por caracteres delimitados por el parámetro delimiter.




## Lenguajes

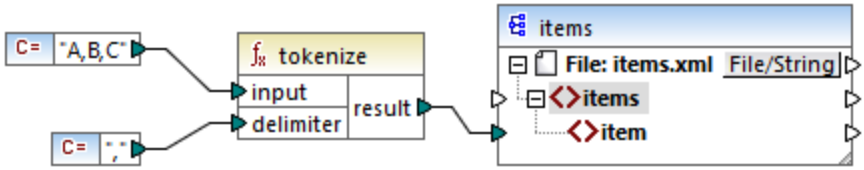
Built-in, C++, C#, Java, XQuery, XSLT 2.0, XSLT 3.0.

## Parámetros

| Nombre | Descripción |
|---|---|
| **input** | La cadena de entrada. |
| **delimiter** | El delimitador que se usa. |

## Ejemplo

Si la cadena de entrada es `A,B,C` y el delimitador es `,` entonces la función devuelve una secuencia de tres cadenas: El resultado será `A`, `B` y `C`.

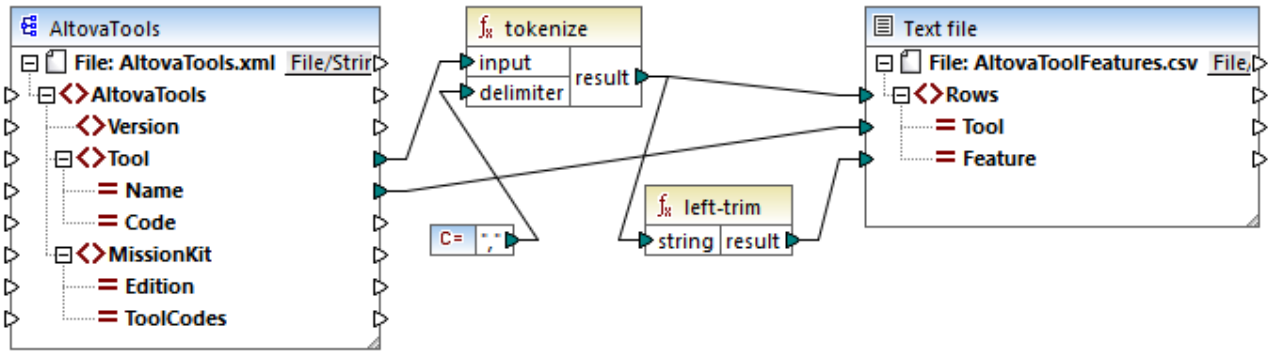



En la asignación de prueba de más arriba, el resultado de la función es una secuencia de cadenas. Según las reglas 81 generales de asignación de datos, por cada elemento de la secuencia de origen, se crea un nuevo **item** en el componente de destino. En consecuencia, el resultado de la asignación tiene este aspecto:

```
<items>
   <item>A</item>
   <item>B</item>
   <item>C</item>
</items>
```


© 2019-2025 Altova GmbH

En el diseño de asignación de datos **tokenizeString1.mfd** de la carpeta **<Documentos>\Altova\MapForce2025\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**. Los datos del elemento **Tool** consisten en 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 trozos. (p. ej. el primer trozo es "XML editor").
- Para cada trozo resultante de la función **tokenize**, se genera una nueva fila en el destino. 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.

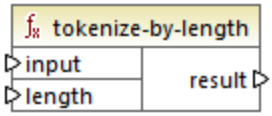
- El archivo de salida del componente de destino se definió como archivo CSV (**AltovaToolFeatures.csv**) con el delimitador de campos punto y coma (haga doble clic en el componente para ver su configuración).

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
```

### 5.9.7.10.12 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.

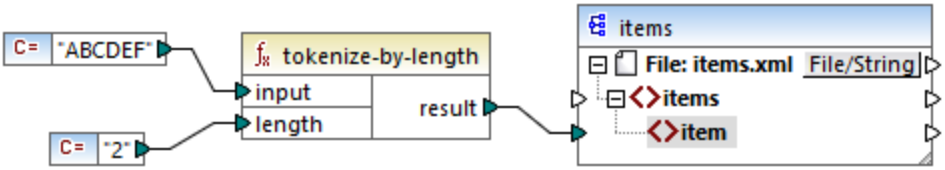
## Parámetros

| Nombre | Descripción |
|---|---|
| **input** | La cadena de entrada. |
| **length** | Determina la longitud de cada una de las cadenas que forman la secuencia de cadenas generada. |


© 2019-2025 Altova GmbH

## Ejemplo

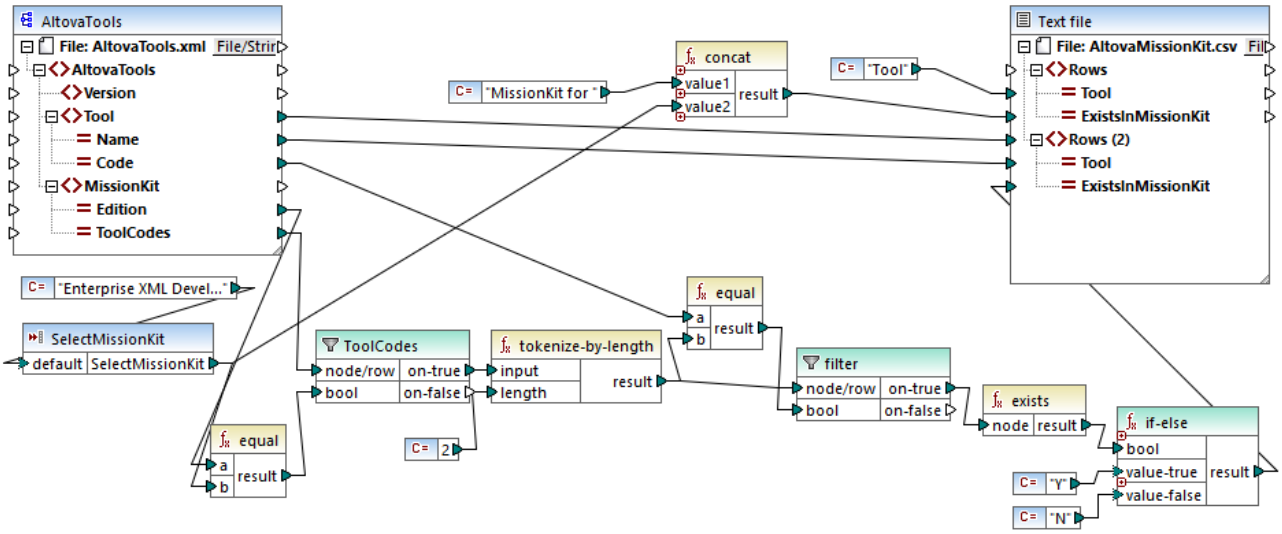
P. ej., si la cadena input es **ABCDEF** y el parámetro length es **2**, entonces la función devuelve una secuencia de tres cadenas: **AB**, **CD** y **EF**.




En la asignación de prueba de más arriba, el resultado de la función es una secuencia de cadenas. Según las reglas 81 generales de asignación de datos, por cada elemento de la secuencia de origen, se crea un nuevo **item** 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 **<Documentos>\Altova\MapForce2025\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 pasa al parámetro **on-true** y, posteriormente, al parámetro **input** de la función `tokenize-by-length` .
- La función `tokenize-by-length` divide el valor `XSMFSVDDSASW` en varios trozos de dos caracteres cada uno. (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. Tenga en cuenta que *todos* los trozos producidos por la función `tokenize-by-length` se pasan 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
```

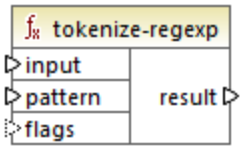
 © 2019-2025 Altova GmbH

```
StyleVision;Y
UModel;N
DatabaseSpy;N
DiffDog;Y
SchemaAgent;Y
SemanticWorks;Y
Authentic;N
```

### 5.9.7.10.13 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

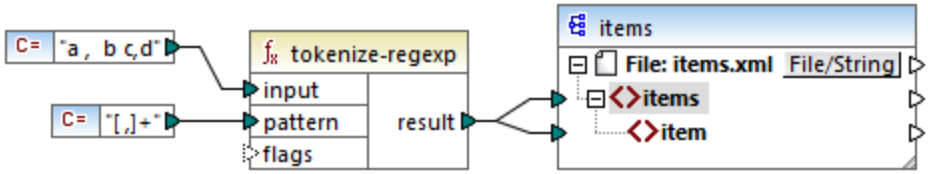
Built-in, C++, C#, Java, XQuery, XSLT 2.0, XSLT 3.0.

## 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 540. |
| **flags** | Parámetro opcional. Indica expresiones regulares flags 543 . 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:

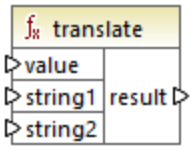
- La expresión regular pattern define una clase de caracteres `[ ,]`, de los cuales se usará un solo carácter (es decir, o bien el espacio, o bien la coma).
- El cuantificador `+` especifica "una o más" apariciones de la clase de caracteres/cadena. Sin este cuantificador, en la secuencia resultante se 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>
```

### 5.9.7.10.14 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

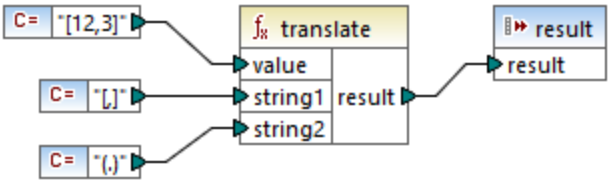
| Nombre | Descripción |
|---|---|
| **value** | La cadena de entrada. |

 © 2019-2025 Altova GmbH

| Nombre | Descripción |
|---|---|
| **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** `(.)`

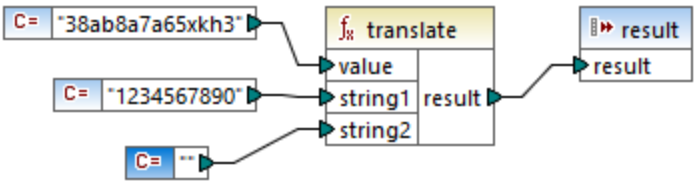
Tenga en cuenta que tanto **string1** como **string2** tienen el mismo número de caracteres. Cada carácter de **string1** se reemplaza con el carácter que esté en esa misma posición en **string2**. Por tanto, en este caso lo que ocurrirá es que:

- 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:

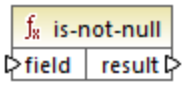
```
abaaxkh
```

## 5.9.7.11 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.

### 5.9.7.11.1 is-not-null

Si el campo es NULL, la función devuelve **false**. De lo contrario devuelve **true**.




#### Lenguajes

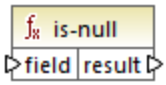
Disponible para Java, C#, C++ y el motor de ejecución integrado.

#### Parámetros

| Nombre | Descripción |
|---|---|
| **field** | El campo de la BD. |

### 5.9.7.11.2 is-null

Si el campo es NULL, la función devuelve **true** . De lo contrario devuelve **false** .





© 2019-2025 Altova GmbH

## Lenguajes

Disponible para Java, C#, C++ y el motor de ejecución integrado.

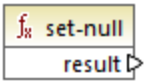
## Parámetros

| Nombre | Descripción |
|---|---|
| **field** | El campo de la BD. |

### 5.9.7.11.3 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.

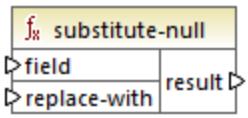



## Lenguajes

Disponible para Java, C#, C++ y el motor de ejecución integrado.

### 5.9.7.11.4 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

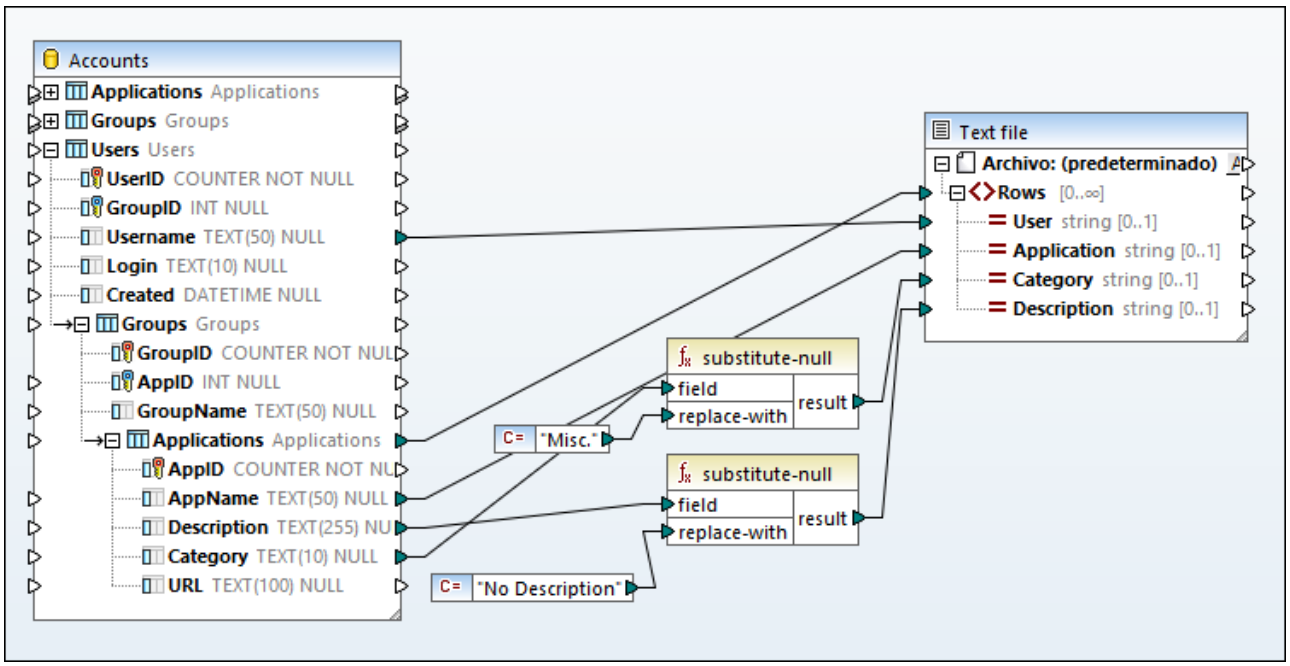
Disponible para Java, C#, C++ y el motor de ejecución integrado.

## 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\MapForce2025\MapForceExamples**.




La asignación lee datos de una BD SQLite 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.

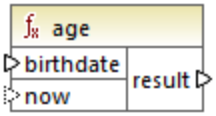
 © 2019-2025 Altova GmbH

## 5.9.7.12 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 sólo están disponibles para Java, C#, C++ o el motor de ejecución integrado.

### 5.9.7.12.1 age

El resultado es la cantidad total de años comprendidos entre la fecha de nacimiento y la fecha actual.




#### Lenguajes

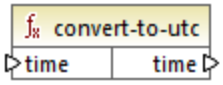
Disponible para Java, C#, C++ y el motor de ejecución integrado.

#### 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. |

### 5.9.7.12.2 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. |

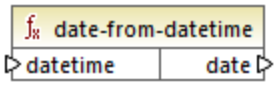
## Ejemplo

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`**.

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.

### 5.9.7.12.3 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. |

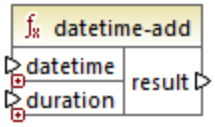
## Ejemplo

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`**.

### 5.9.7.12.4 datetime-add

El resultado es el valor datetime que se obtiene al añadir un valor duration (segundo argumento) a un valor datetime (primer argumento).


© 2019-2025 Altova GmbH




## Lenguajes

Disponible para Java, C#, C++ y el motor de ejecución integrado.

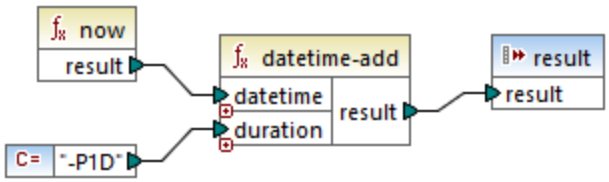
## Parámetros

| 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`.<br><br>Un ejemplo del parámetro `duration` es **`P1Y2M3DT04H05M59S`**, donde:<br><br>• "P" designa el periodo y es obligatorio;<br>• El resto de caracteres representa, en este orden: 1 año, 2 meses, 3 días, T (indica el tiempo), 04 horas, 05 minutos, 59 segundos.<br><br>Si aparece el caracter menos (-) antes de "P" la duración se entiende como negativa, por ejemplo: **`-P1D`**. |

## Ejemplo

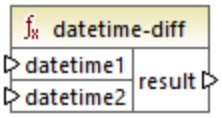
Imagine que el valor de entrada **datetime** es **`2001-12-17T09:30:02+05:00`**. Si **duration** es **`P10D`** (10 días), entonces el resultado de la función es **`2001-12-27T00:00:00+05:00`**.

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).



### 5.9.7.12.5 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.




## Lenguajes

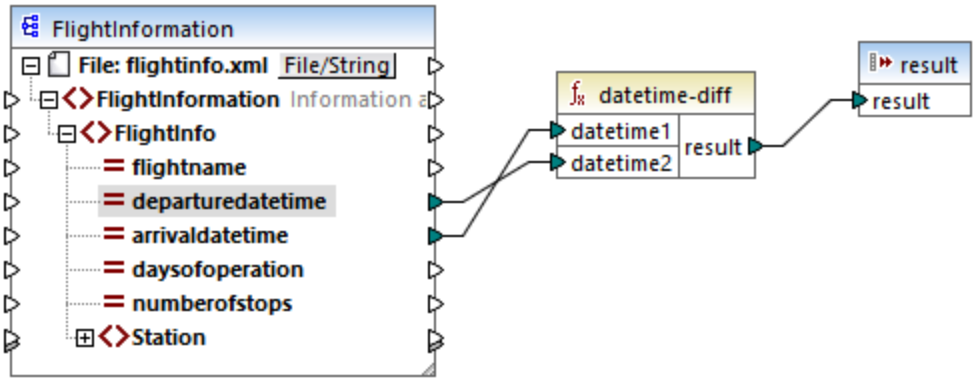
Disponible para Java, C#, C++ y el motor de ejecución integrado.

## 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.



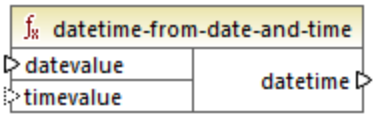

El resultado de la asignación es la diferencia entre las dos horas (que es un periodo de 10 horas).

```
PT10H
```


© 2019-2025 Altova GmbH

### 5.9.7.12.6 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

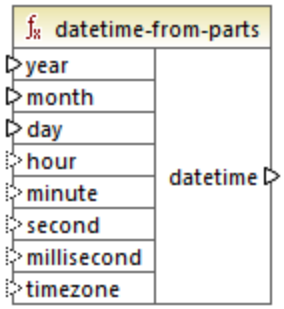
| Nombre | Tipo | Descripción |
|---|---|---|
| **datevalue** | **xs:date** | Obtiene el valor de tipo `xs: date` |
| **timevalue** | **xs:time** | Obtiene el valor de tipo `xs: time` |

#### 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**.

### 5.9.7.12.7 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

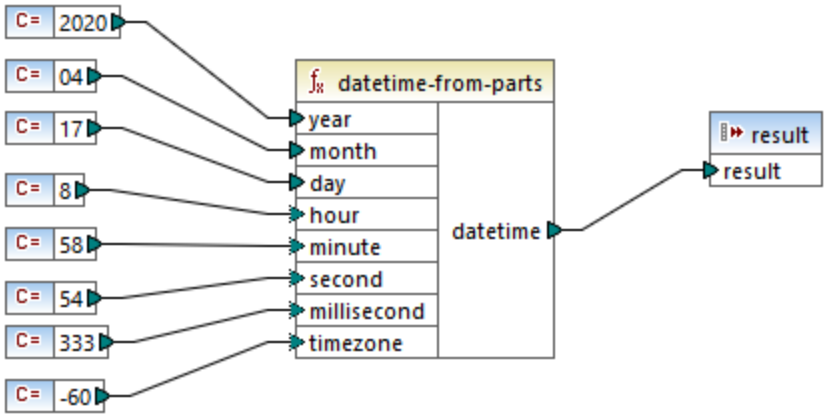
Disponible para Java, C#, C++ y el motor de ejecución integrado.

## Parámetros

| Nombre | Tipo | Descripción |
|---|---|---|
| **year** | `xs:int` | El año. |
| **month** | `xs:int` | El mes. |
| **day** | `xs:int` | El día del mes. |
| **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

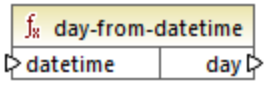
La asignación siguiente genera un valor `xs:dateTime` a partir de partes dadas por constantes.




El resultado de la asignación es **`2020-04-17T08:58:54.333-01:00`**.

### 5.9.7.12.8 day-from-datetime

El resultado es el día del argumento dateTime.





© 2019-2025 Altova GmbH

### Lenguajes

Disponible para Java, C#, C++ y el motor de ejecución integrado.

### Parámetros

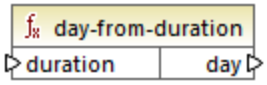
| Nombre | Tipo | Descripción |
|---|---|---|
| **datetime** | `xs:dateTime` | Obtiene el valor de la entrada de tipo `xs: dateTime`. |

### Ejemplo

Si **datetime** es `2001-12-17T10:30:03+01:00`, entonces la función devuelve `17`.

## 5.9.7.12.9 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

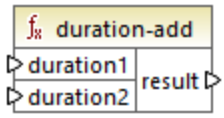
| Nombre | Tipo | Descripción |
|---|---|---|
| **duration** | `xs:duration` | Obtiene el valor de la entrada de tipo `xs:duration`. |

### Ejemplo

Si **duration** es `P1Y2M3DT10H30M`, entonces la función `day-from-datetime` devuelve `3`.

## 5.9.7.12.10 duration-add

Devuelve la duración resultante de sumar dos duraciones.



## Lenguajes

Disponible para Java, C#, C++ y el motor de ejecución integrado.

## 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).

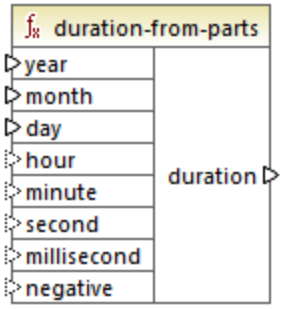
### 5.9.7.12.11 duration-from-parts

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.

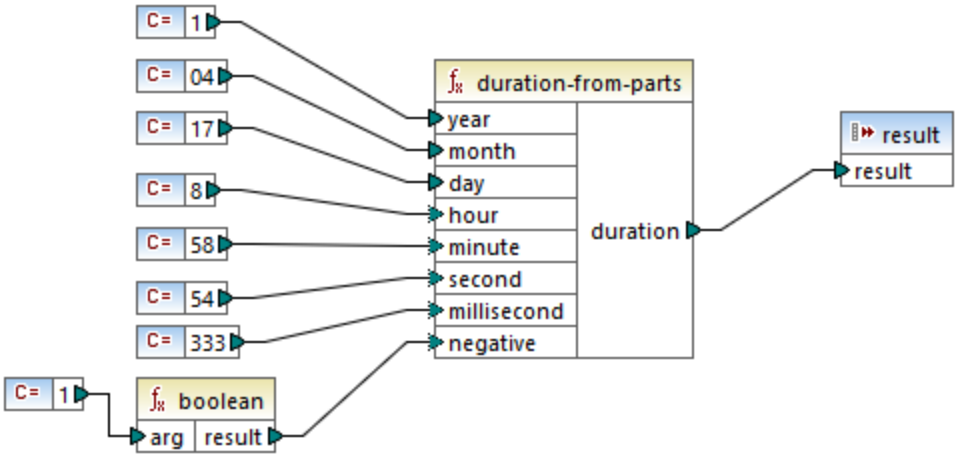

© 2019-2025 Altova GmbH

## Parámetros

| Nombre | Tipo | Descripción |
|---|---|---|
| **year** | `xs:int` | El año. |
| **month** | `xs:int` | El mes. |
| **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`**.

## 5.9.7.12.12 duration-subtract

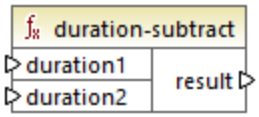
El resultado es la duración que se obtiene al sustraer **duration2** a **duration1**.

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`**.




## Lenguajes

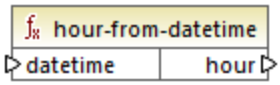
Disponible para Java, C#, C++ y el motor de ejecución integrado.

## 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).

### 5.9.7.12.13 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`. |

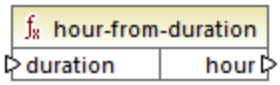
## Ejemplo

Si **datetime** es **`2001-12-17T09:30:02+05:00`**, entonces la función devuelve **`9`**.


© 2019-2025 Altova GmbH

### 5.9.7.12.14 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**.

### 5.9.7.12.15 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

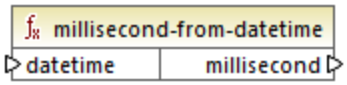
| Nombre | Tipo | Descripción |
|---|---|---|
| **datetime** | `xs:dateTime` | Obtiene el valor de la entrada de tipo `xs: dateTime`. |

#### Ejemplo

Si **datetime** es `2020-04-17T09:30:02+02:00`, la función devuelve **true** porque 2020 es bisiesto.

### 5.9.7.12.16 millisecond-from-datetime

El resultado es el componente millisecond del argumento datetime.




#### Lenguajes

Disponible para Java, C#, C++ y el motor de ejecución integrado.

#### Parámetros

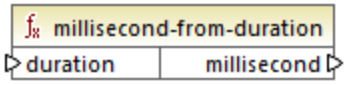
| Nombre | Tipo | Descripción |
|---|---|---|
| **datetime** | `xs:dateTime` | Obtiene el valor de la entrada de tipo `xs: dateTime`. |

#### Ejemplo

Si **datetime** es **`2001-12-17T09:30:02.544+05:00`**, entonces la función devuelve **`544`**.

### 5.9.7.12.17 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`. |

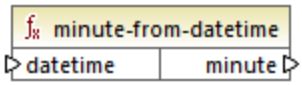
#### Ejemplo

Si **duration** es **`P0Y0M0DT05H07M02.227S`**, entonces la función devuelve **`227`**.


© 2019-2025 Altova GmbH

### 5.9.7.12.18 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

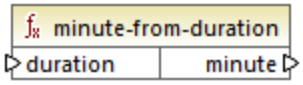
| Nombre | Tipo | Descripción |
|---|---|---|
| **datetime** | `xs:dateTime` | Obtiene el valor de la entrada de tipo `xs: dateTime`. |

#### Ejemplo

Si **datetime** es `2001-12-17T09:30:02.544+05:00`, entonces la función devuelve `30`.

### 5.9.7.12.19 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

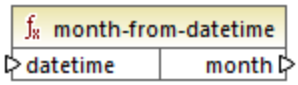
| Nombre | Tipo | Descripción |
|---|---|---|
| **duration** | `xs:duration` | Obtiene el valor de la entrada de tipo `xs:duration`. |

#### Ejemplo

Si **duration** es `P0Y0M0DT05H07M02.227S`, entonces la función devuelve `7`.

### 5.9.7.12.20 month-from-datetime

El resultado es el componente month del argumento dateTime.




#### Lenguajes

Disponible para Java, C#, C++ y el motor de ejecución integrado.

#### Parámetros

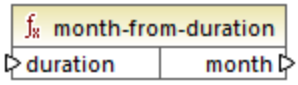
| Nombre | Tipo | Descripción |
|---|---|---|
| **datetime** | `xs:dateTime` | Obtiene el valor de la entrada de tipo `xs: dateTime`. |

#### Ejemplo

Si **datetime** es `2001-12-17T09:30:02.544+05:00`, entonces la función devuelve `12`.

### 5.9.7.12.21 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`. |

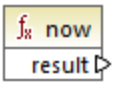
#### Ejemplo

Si **duration** es `P0Y04M0DT05H07M02.227S`, entonces la función devuelve `4`.


© 2019-2025 Altova GmbH

### 5.9.7.12.22 now

El resultado es la fechaHora actual (incluida la zona horaria) como valor `xs:dateTime`.

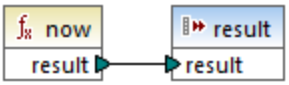



#### Lenguajes

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.

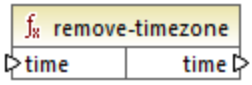



Un ejemplo del resultado sería **`2020-04-17T11:42:34.684+02:00`**.

Para ver un ejemplo de cómo extraer la fecha de ayer consulte la función core | lang | datetime-add 644 .

### 5.9.7.12.23 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

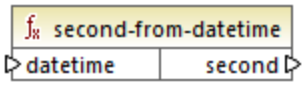
| Nombre | Tipo | Descripción |
|---|---|---|
| **time** | **`xs:dateTime`** | Obtiene el valor de la entrada de tipo `xs: dateTime`. |

#### Ejemplo

Si **time** es **`2001-12-17T09:30:02+05:00`**, entonces la función devuelve **`2001-12-17T09:30:02`**.

### 5.9.7.12.24 second-from-datetime

El resultado es el componente seconds 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 `2`.

### 5.9.7.12.25 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`. |

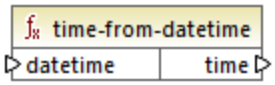
#### Ejemplo

Si **duration** es `P0Y04M0DT05H07M02.227S`, entonces la función devuelve `2`.


© 2019-2025 Altova GmbH

### 5.9.7.12.26 time-from-datetime

Devuelve el componente time del argumento dateTime.




#### Lenguajes

Disponible para Java, C#, C++ y el motor de ejecución integrado.

#### Parámetros

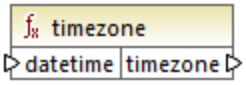
| Nombre | Tipo | Descripción |
|---|---|---|
| **datetime** | `xs:dateTime` | Obtiene el valor de la entrada de tipo `xs: dateTime`. |

#### Ejemplo

Si **datetime** es `2001-12-17T09:30:02.544+05:00`, entonces la función devuelve `09:31:02+05:00`.

### 5.9.7.12.27 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

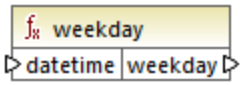
| Nombre | Tipo | Descripción |
|---|---|---|
| **datetime** | `xs:dateTime` | Obtiene el valor de la entrada de tipo `xs: dateTime`. |

#### Ejemplo

Si **datetime** es `2001-12-17T09:30:02.544+05:00`, entonces la función devuelve `300`.

### 5.9.7.12.28 weekday

Devuelve el día de la semana indicado por el valor dateTime, siendo **Lunes=1** y así sucesivamente hasta **Domingo=7**.




#### Lenguajes

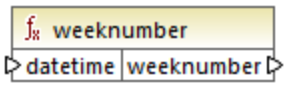
Disponible para Java, C#, C++ y el motor de ejecución integrado.

#### Parámetros

| Nombre | Tipo | Descripción |
|---|---|---|
| **datetime** | `xs:dateTime` | Obtiene el valor de la entrada de tipo `xs: dateTime`. |

### 5.9.7.12.29 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.

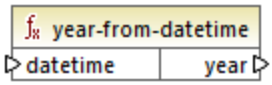
#### Parámetros

| Nombre | Tipo | Descripción |
|---|---|---|
| **datetime** | `xs:dateTime` | Obtiene el valor de la entrada de tipo `xs: dateTime`. |


© 2019-2025 Altova GmbH

### 5.9.7.12.30 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

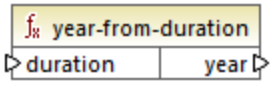
| Nombre | Tipo | Descripción |
|---|---|---|
| **datetime** | `xs:dateTime` | Obtiene el valor de la entrada de tipo `xs: dateTime`. |

#### Ejemplo

Si **datetime** es **`2001-12-17T09:30:02.544+05:00`**, entonces la función devuelve **`2001`**.

### 5.9.7.12.31 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`**.

## 5.9.7.13 lang | file functions

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 sólo 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 sólo 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 20. 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.

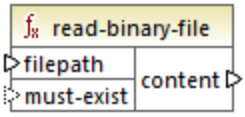
*Leer y escribir archivos binarios*
En MapForce no existe ningún tipo de componente asociado a archivos binarios, como sí ocurre, por ejemplo, con los archivos XML, JSON o de texto. Sin embargo, para lograr los objetivos de los ejemplos del punto anterior puede usar estas funciones integradas de MapForce:

- `read-binary-file` 662
- `write-binary-file` 665

### 5.9.7.13.1 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.

 © 2019-2025 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

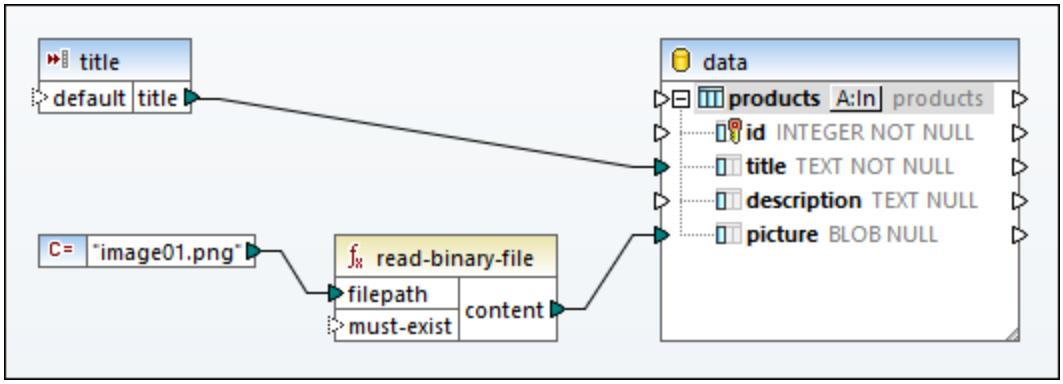
Built-in.

## Parámetros

| Nombre | Tipo | Descripción |
|---|---|---|
| **filepath** | `xs:string` | La ruta de acceso al 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.<br><br>El valor predeterminado es **true**. |

## Ejemplo

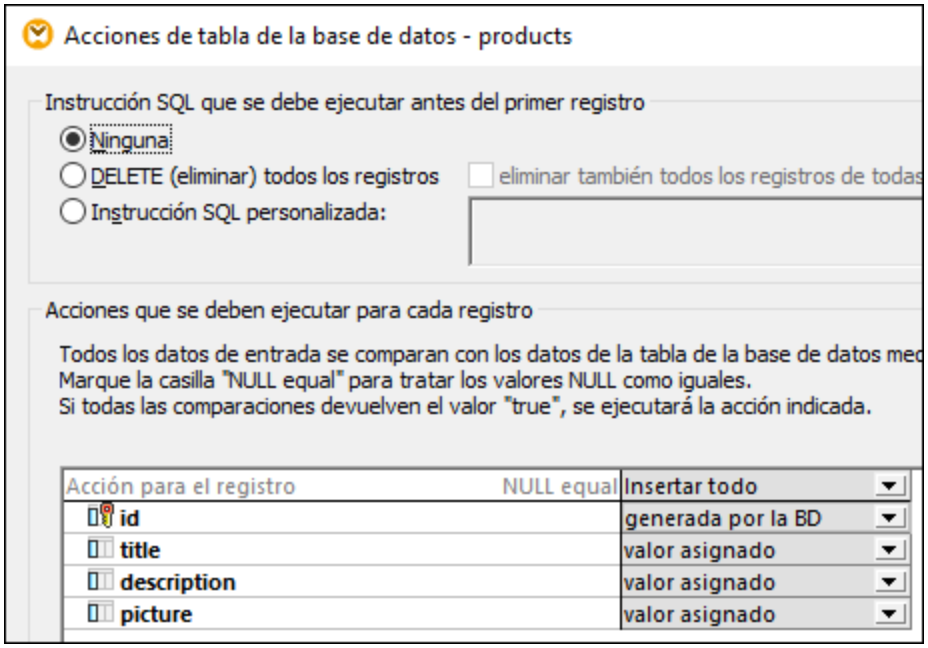
La asignación siguiente lee datos desde un archivo de imagen y los escribe en una tabla de BD. La BD de destino es SQLite. Observe que el tipo de datos del campo de BD **picture** es BLOB.



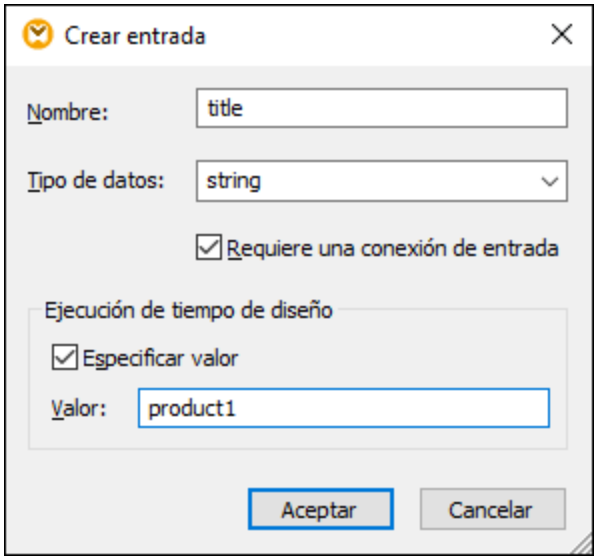
En esta asignación se usó la función `read-binary-file` para extraer contenido binario del archivo. El primer argumento, **filepath**, viene dado por una constante. Tenga en cuenta que, al ser una ruta relativa, MapForce buscará el archivo de imagen en el mismo directorio que la asignación.

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 283.




- **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 369.



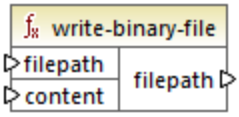

 © 2019-2025 Altova GmbH

- **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**.

### 5.9.7.13.2 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 381. 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

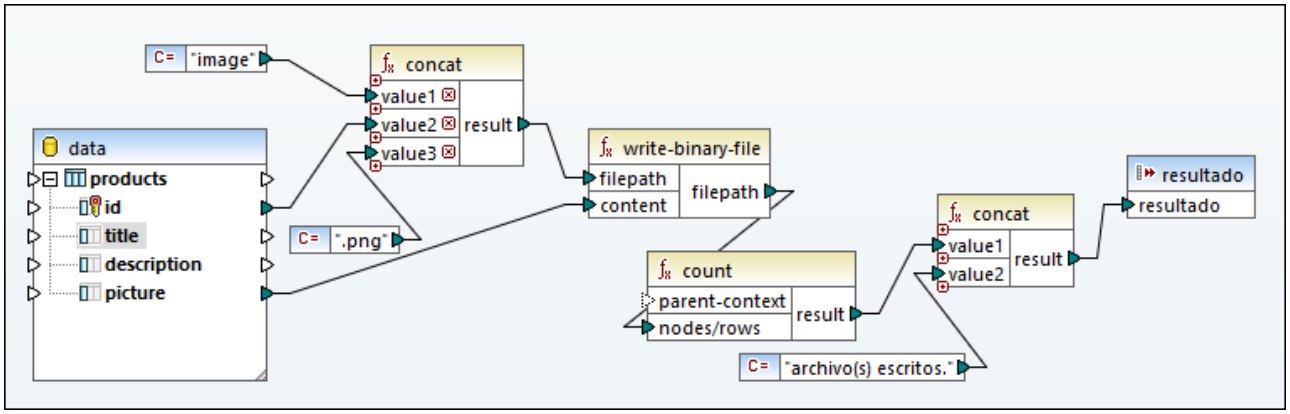
| Nombre | Tipo | Descripción |
|---|---|---|
| **filepath** | `xs:string` | La ruta de acceso al archivo de entrada. |
| **content** | `xs:base64Binary` | El contenido binario de tipo `xs:base64Binary`. |

## Ejemplo

La asignación siguiente lee valores BLOB de una BD SQLite y escribe archivos de imagen en el disco. La base de datos contiene una tabla llamada **products** cuyas columnas (campos) son:

- **id** (*número entero*, el número de serie permanente y único del registro)
- **title** (*texto*, el título del producto)
- **description** (*texto*, la descripción del producto
- **picture** (*blob*, la imagen del producto)

Para este ejemplo sólo son relevantes los campos **id** y **picture**.

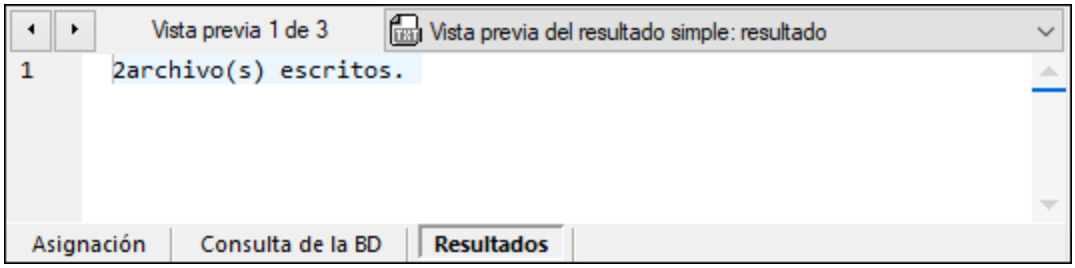



El objetivo de la asignación es extraer todas las imágenes de la tabla **products** y escribirlas en disco como archivos. Como se ve más arriba, para ello se usa la función `write-binary-file`. El primer argumento, **filepath**, recibe la ruta de acceso de las imágenes. La ruta debe ser única para evitar sobrescribir otros archivos, por lo que para generarla se concatenan el **id** de BD único de cada registro con la palabra "image" y luego se añade la extensión de archivo ".png" (se asume que todas las imágenes están en formato PNG). Recuerde que la ruta es relativa, por lo que los archivos se escriben en el mismo directorio que la asignación.

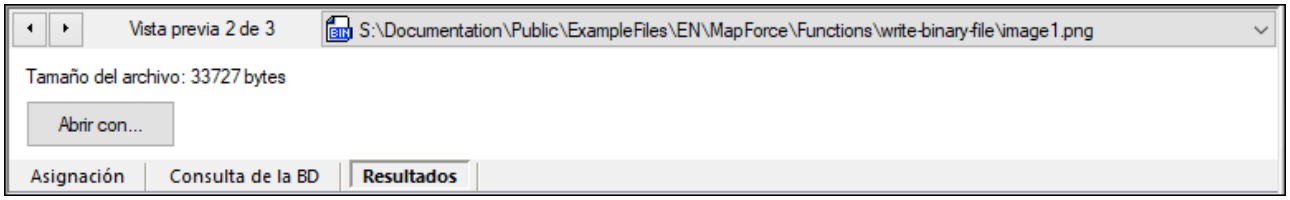
El segundo argumento, **content**, recibe el contenido binario que se almacena en la BD.

La función `count` devuelve el recuento de todos los archivos generados y combina ese número con la cadena de texto "archivo(s) escritos". Así puede saber cuántos archivos ha generado la asignación. En este ejemplo la BD sólo contiene dos registros de producto, que es lo que refleja el resultado:

 © 2019-2025 Altova GmbH




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.



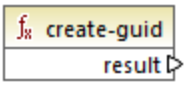

Para guardar todos los archivos haga clic en los botones de la barra de herramientas **Guardar resultado generado** o **Guardar todos los resultados generados**, según el caso.

## 5.9.7.14 lang | generator functions (generador)

Las funciones generadoras de la biblioteca **lang** generan valores (actualmente `create-guid` es la única función de este tipo).

### 5.9.7.14.1 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 686.




#### Lenguajes

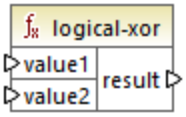
Disponible para Java, C#, C++ y el motor de ejecución integrado.

## 5.9.7.15 lang | logical functions (lógica)

Las funciones lógicas de la biblioteca **lang** incluyen funciones que evalúan distintos tipos de valores con lógica booleana.

### 5.9.7.15.1 logical-xor

El resultado es **true** si **value1** es distinto de **value2**. De lo contrario, el resultado es **false**.




#### Lenguajes

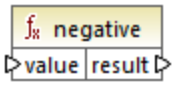
Disponible para Java, C#, C++ y el motor de ejecución integrado.

#### Parámetros

| Nombre | Tipo | Descripción |
|---|---|---|
| **value1** | `xs:boolean` | El primer valor de la entrada. |
| **value2** | `xs:boolean` | El segundo valor de la entrada. |

### 5.9.7.15.2 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.

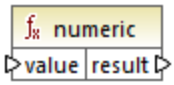
#### Parámetros

| Nombre | Tipo | Descripción |
|---|---|---|
| **value** | `xs:decimal` | El valor de entrada. |


© 2019-2025 Altova GmbH

### 5.9.7.15.3 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

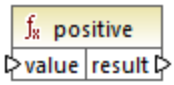
| Nombre | Tipo | Descripción |
|---|---|---|
| **value** | `xs:decimal` | El valor de entrada. |

#### 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** .

### 5.9.7.15.4 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.

#### Parámetros

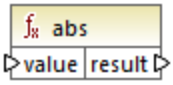
| Nombre | Tipo | Descripción |
|---|---|---|
| **value** | `xs:decimal` | El valor de entrada. |

## 5.9.7.16 lang | math functions (matemáticas)

Las funciones matemáticas de la biblioteca **lang** se pueden usar para ejecutar distintas operaciones matemáticas en la asignación.

### 5.9.7.16.1 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

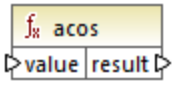
Disponible para Java, C#, C++ y el motor de ejecución integrado.

#### Parámetros

| Nombre | Tipo | Descripción |
|---|---|---|
| **value** | `xs:decimal` | El valor de entrada. |

### 5.9.7.16.2 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.

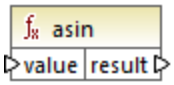
#### Parámetros

| Nombre | Tipo | Descripción |
|---|---|---|
| **value** | `xs:double` | El valor de entrada. |


© 2019-2025 Altova GmbH

### 5.9.7.16.3 asin

El resultado es el arco seno del parámetro de entrada **value** en el rango de `-pi/2` a `pi/2`.




#### Lenguajes

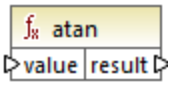
Disponible para Java, C#, C++ y el motor de ejecución integrado.

#### Parámetros

| Nombre | Tipo | Descripción |
|---|---|---|
| **value** | `xs:double` | El valor de entrada. |

### 5.9.7.16.4 atan

El resultado es el arco tangente del parámetro de entrada **value** en el rango de `-pi/2` a `pi/2`.




#### Lenguajes

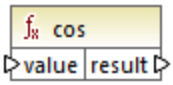
Disponible para Java, C#, C++ y el motor de ejecución integrado.

#### Parámetros

| Nombre | Tipo | Descripción |
|---|---|---|
| **value** | `xs:double` | El valor de entrada. |

### 5.9.7.16.5 cos

El resultado es el coseno del parámetro de entrada **value**. La unidad del valor es el radián.



### Lenguajes

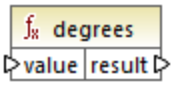
Disponible para Java, C#, C++ y el motor de ejecución integrado.

### Parámetros

| Nombre | Tipo | Descripción |
|---|---|---|
| **value** | `xs:double` | El valor de entrada. |

## 5.9.7.16.6 degrees

El resultado de convertir en grados el valor de value dado en radianes.




### Lenguajes

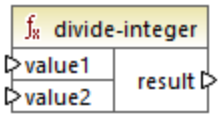
Disponible para Java, C#, C++ y el motor de ejecución integrado.

### Parámetros

| Nombre | Tipo | Descripción |
|---|---|---|
| **value** | `xs:double` | El valor de entrada. |

## 5.9.7.16.7 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.


© 2019-2025 Altova GmbH

## Parámetros

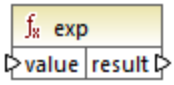
| Nombre | Tipo | Descripción |
|---|---|---|
| **value1** | `xs:decimal` | El primer valor de la entrada. |
| **value2** | `xs:decimal` | El segundo valor de la entrada. |

## Ejemplo

Por ejemplo, **15** divide-integer **2**, el entero resultante es **7**.

### 5.9.7.16.8 exp

El resultado es **e** (el logaritmo natural base) elevado a la potencia **value**.




## Lenguajes

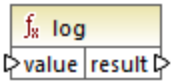
Disponible para Java, C#, C++ y el motor de ejecución integrado.

## Parámetros

| Nombre | Tipo | Descripción |
|---|---|---|
| **value** | `xs:double` | El valor de entrada. |

### 5.9.7.16.9 log

El resultado es el logaritmo natural (en base e) de **value**.




## Lenguajes

Disponible para Java, C#, C++ y el motor de ejecución integrado.

## Parámetros

| Nombre | Tipo | Descripción |
|---|---|---|
| **value** | `xs:double` | El valor de entrada. |

## 5.9.7.16.10 log10

El resultado es el logaritmo decimal (en base 10) de **value**.

## Lenguajes

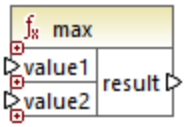
Disponible para Java, C#, C++ y el motor de ejecución integrado.

## Parámetros

| Nombre | Tipo | Descripción |
|---|---|---|
| **value** | `xs:double` | El valor de entrada. |

## 5.9.7.16.11 max

El resultado es el valor numéricamente mayor de value1 comparado con value2. Por defecto, esta función tiene sólo 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 461.




## Lenguajes

Disponible para Java, C#, C++ y el motor de ejecución integrado.

## Parámetros

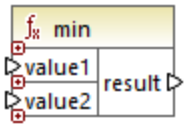
| Nombre | Tipo | Descripción |
|---|---|---|
| **value1** | `xs:decimal` | El primer valor de la entrada. |


© 2019-2025 Altova GmbH

| Nombre | Tipo | Descripción |
|---|---|---|
| **value2** | `xs:decimal` | El segundo valor de la entrada. |
| **valueN** | `xs:decimal` | El valor de *n* entrada. |

### 5.9.7.16.12 min

El resultado es el valor numéricamente menor de value1 comparado con value2. Por defecto, esta función tiene sólo 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 461.




#### 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. |

### 5.9.7.16.13 pi

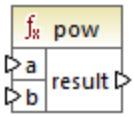
El resultado es el valor de $\pi$.

#### Lenguajes

Disponible para Java, C#, C++ y el motor de ejecución integrado.

### 5.9.7.16.14 pow

El resultado es el valor de **a** elevado a la potencia **b**.




#### Lenguajes

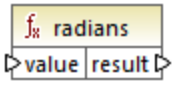
Disponible para Java, C#, C++ y el motor de ejecución integrado.

#### Parámetros

| Nombre | Tipo | Descripción |
|---|---|---|
| **a** | `xs:double` | Da el valor **a** (la base). |
| **b** | `xs:double` | Da el valor **b** (la potencia). |

### 5.9.7.16.15 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.

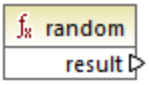
#### Parámetros

| Nombre | Tipo | Descripción |
|---|---|---|
| **value** | `xs:double` | El valor de entrada. |

### 5.9.7.16.16 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.


© 2019-2025 Altova GmbH

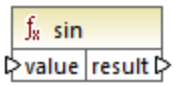



## Lenguajes

Disponible para Java, C#, C++ y el motor de ejecución integrado.

## 5.9.7.16.17 sin

El resultado es el seno de **value**.




## Lenguajes

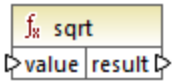
Disponible para Java, C#, C++ y el motor de ejecución integrado.

## Parámetros

| Nombre | Tipo | Descripción |
|---|---|---|
| **value** | `xs:double` | El valor de entrada. |

## 5.9.7.16.18 sqrt

El resultado es la raíz cuadrada de **value**.




## Lenguajes

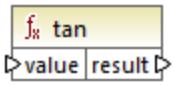
Disponible para Java, C#, C++ y el motor de ejecución integrado.

## Parámetros

| Nombre | Tipo | Descripción |
|---|---|---|
| **value** | `xs:double` | El valor de entrada. |

### 5.9.7.16.19 tan

El resultado es la tangente de **value**.




#### Lenguajes

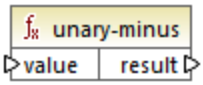
Disponible para Java, C#, C++ y el motor de ejecución integrado.

#### Parámetros

| Nombre | Tipo | Descripción |
|---|---|---|
| **value** | `xs:double` | El valor de entrada. |

### 5.9.7.16.20 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**.

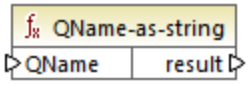
## 5.9.7.17 lang | QName functions (QName)

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 sólo están disponibles para Java, C#, C++ o el motor de ejecución integrado.


© 2019-2025 Altova GmbH

### 5.9.7.17.1 QName-as-string

El resultado es la representación de cadena única del QName.




#### Lenguajes

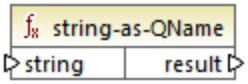
Disponible para Java, C#, C++ y el motor de ejecución integrado.

#### Parámetros

| Nombre | Descripción |
|---|---|
| **QName** | El valor `xs:QName` de entrada. |

### 5.9.7.17.2 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.

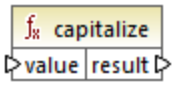
| Nombre | Descripción |
|---|---|
| **string** | El valor `string` de entrada. |

## 5.9.7.18 lang | string functions (cadena)

Las funciones de cadena de la biblioteca `lang` le permiten procesar cadenas (p.ej., recortar, rellenar, reemplazar, convertir cadenas a mayúsculas o minúsculas, etc.).

### 5.9.7.18.1 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

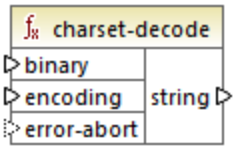
| Nombre | Tipo | Descripción |
|---|---|---|
| **value** | `xs:string` | El valor de entrada. |

#### Ejemplo

Si el valor de entrada es `el veloz zorro marrón`, la función devuelve `El Veloz Zorro Marrón`.

### 5.9.7.18.2 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 682 .




#### Lenguajes

Disponible en el motor de ejecución integrado.

#### Parámetros

| Nombre | Tipo | Descripción |
|---|---|---|
| binary-data | `xs:base64Binary` | Los datos binarios como texto en Base 64. |


© 2019-2025 Altova GmbH

| 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:<br>• **true**: terminar el procesamiento con una excepción donde está el carácter no válido.<br>• **false**: terminar el procesamiento y reemplazar los caracteres no válidos con el carácter de sustitución .<br><br>El valor predeterminado es true. |

## Ejemplo

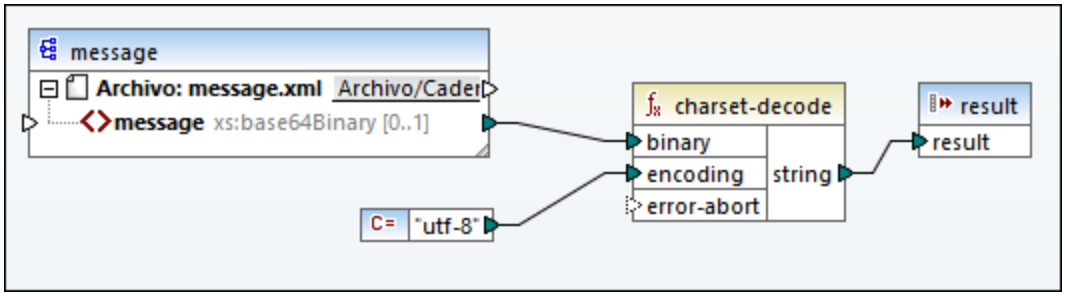
Imagine que quiere descodificar datos binarios con origen en el siguiente archivo XML. Observe que el elemento **message** contiene datos binarios codificados como texto en Base 64.

```
<?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:

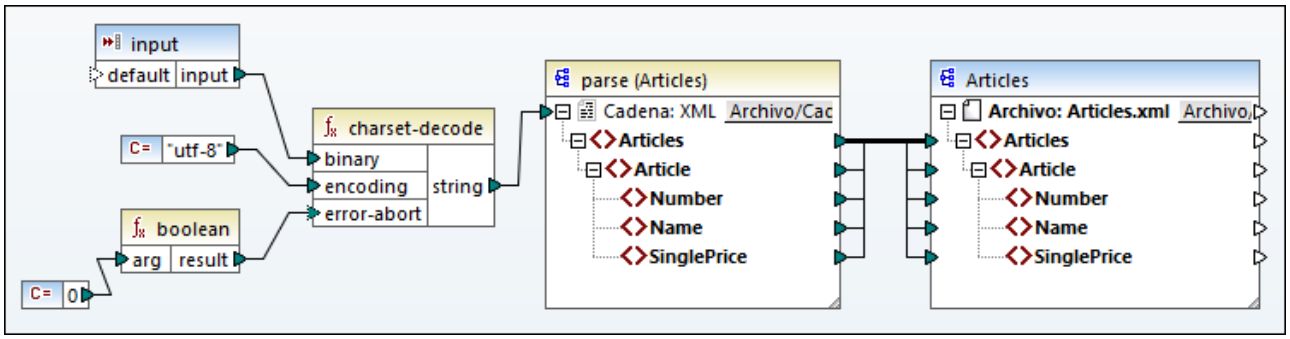
```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
 <xs:element name="message" type="xs:base64Binary"/>
</xs:schema>
```

Este es el aspecto de una asignación que descodifica el mensaje anterior:




La asignación de este mensaje tiene como salida el texto "Lorem ipsum".

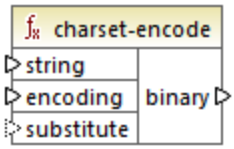
Una asignación también puede procesar texto o archivos XML codificados como datos en Base64 con ayuda de un componente de serialización de MapForce. Por ejemplo, la asignación siguiente tiene un parámetro de entrada que espera datos de texto en Base64. Si asumimos que los datos en Base64 se crearon a partir de un archivo XML, como se muestra en el ejemplo charset-encode 682 , 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 791.

### 5.9.7.18.3 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 680 .



## Lenguajes

Disponible en el motor de ejecución integrado.

## Parámetros

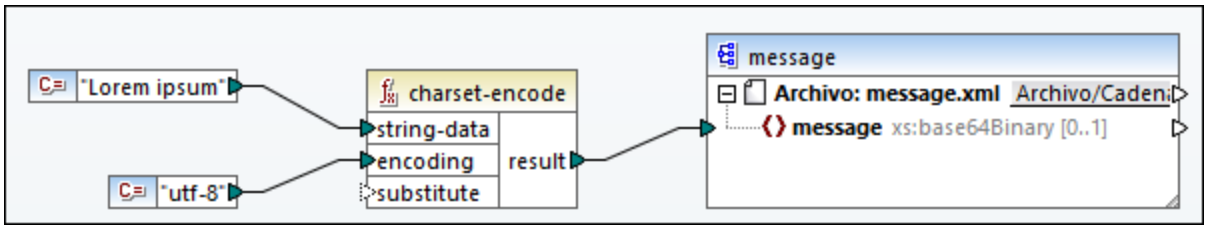
| Nombre | Tipo | Descripción |
|---|---|---|
| string-data | `xs:string` | El datos de cadena que se debe cifrar. |
| encoding | `xs:string` | El conjunto de caracteres usado para codificar (por ejemplo, "utf-8"). |
| substitute | `xs:string` | Argumento opcional que indica qué carácter de reemplazo 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>
```

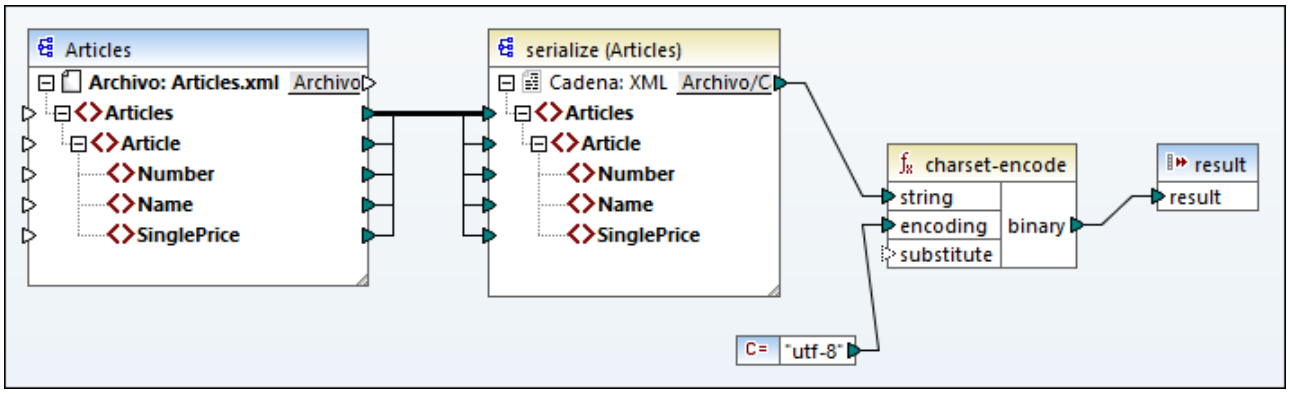
Este es el aspecto de una asignación que cifra en Base64:




Esta asignación produce resultados XML como los del extracto de código siguiente (se han omitido las referencias al esquema y la declaración XML):

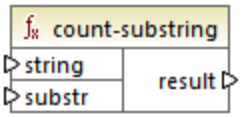
```
<message>TG9yZW0gaXBzdW0=</message>
```

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) 380. Para más información consulte el apartado Redistribuir y serializar cadenas de texto 791.




## 5.9.7.18.4 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.

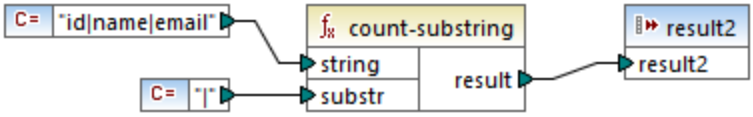
### Parámetros

| Nombre | Tipo | Descripción |
|---|---|---|
| **string** | `xs:string` | La cadena de entrada. |
| **substr** | `xs:string` | La subcadena que se debe comprobar. |

### Ejemplo

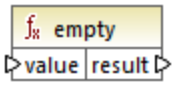
Esta asignación devuelve **2**. El número de veces que el separador aparece dentro de la cadena de entrada **`id|name|email`**.


© 2019-2025 Altova GmbH




### 5.9.7.18.5 empty

El resultado es **true** si la cadena de entrada value está vacía. De lo contrario, el resultado es **false**.




#### Lenguajes

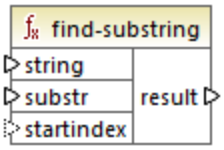
Disponible para Java, C#, C++ y el motor de ejecución integrado.

#### Parámetros

| Nombre | Tipo | Descripción |
|---|---|---|
| **value** | `xs:string` | El valor de entrada. |

### 5.9.7.18.6 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.

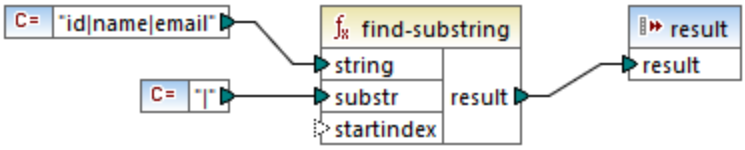
#### Parámetros

| Nombre | Tipo | Descripción |
|---|---|---|
| **string** | `xs:string` | La cadena de entrada. |

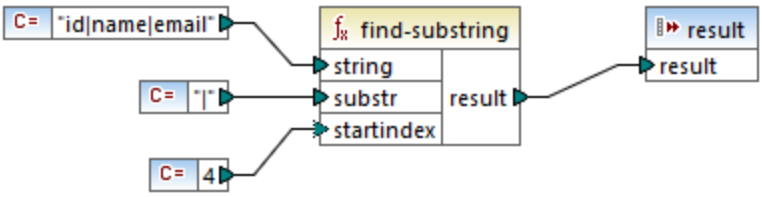
| Nombre | Tipo | Descripción |
|---|---|---|
| **substr** | `xs:string` | La subcadena que se debe buscar. |
| **startindex** | `xs:int` | Opcional. Indica la posición (índice) inicial de la búsqueda. Si no se indica este parámetro, la búsqueda empieza en la posición 1. |

## 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`.

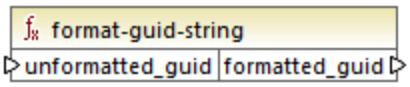



Si indica que el índice inicial es **4**, entonces la función empieza a buscar por el cuarto carácter. Por lo tanto, el resultado de la asignación siguiente sería **8**, que es la primera aparición de la barra vertical a partir del cuarto carácter.




## 5.9.7.18.7 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 667 .




## Lenguajes

Disponible para Java, C#, C++ y el motor de ejecución integrado.

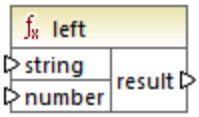
 © 2019-2025 Altova GmbH

## Parámetros

| Nombre | Tipo | Descripción |
|---|---|---|
| **unformatted_guid** | `xs:string` | La cadena de entrada en formato HEX a la que se le quiere dar formato. |

## 5.9.7.18.8 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

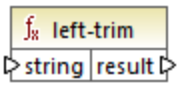
| Nombre | Tipo | Descripción |
|---|---|---|
| **string** | `xs:string` | La cadena de entrada. |
| **number** | `xs:int` | Indica cuántos caracteres se devuelven empezando por el principio de la cadena. |

## Ejemplo

Si la cadena de entrada es `Esto es una frase` y el número es **4**, la función devuelve `Esto`.

## 5.9.7.18.9 left-trim

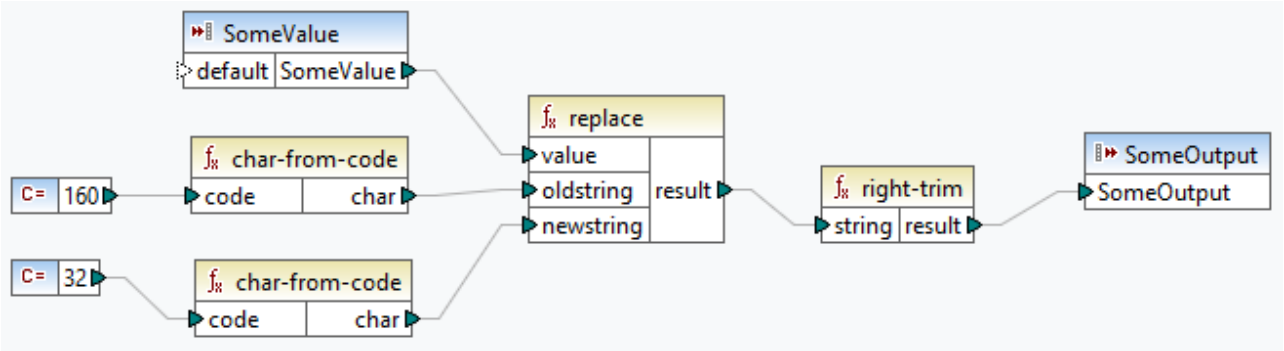
El resultado es la cadena de entrada string después de quitarle todos los espacios en blanco iniciales.




*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

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

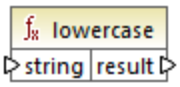
Disponible para Java, C#, C++ y el motor de ejecución integrado.

## Parámetros

| Nombre | Tipo | Descripción |
|---|---|---|
| **string** | `xs:string` | La cadena de entrada. |

## 5.9.7.18.10 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

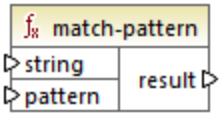
| Nombre | Tipo | Descripción |
|---|---|---|
| **string** | `xs:string` | La cadena de entrada. |


© 2019-2025 Altova GmbH

## 5.9.7.18.11 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 540.

**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

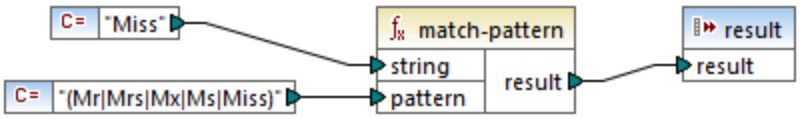
Disponible para Java, C#, C++ y el motor de ejecución integrado.

### Parámetros

| Nombre | Tipo | Descripción |
|---|---|---|
| **string** | `xs:string` | La cadena de entrada. |
| **pattern** | `xs:string` | La expresión regular que debe coincidir. |

### Ejemplo

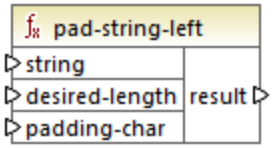
Esta asignación valida títulos extranjeros de personas. Concretamente, el resultado de la asignación será **true** para estos títulos: Mr, Mrs, Mx, Ms, Miss.




Si la cadena de entrada es distinta a esos títulos, el resultado de la asignación será **false**.

## 5.9.7.18.12 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.




## Lenguajes

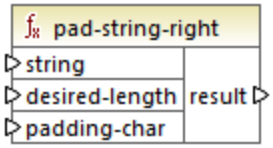
Disponible para Java, C#, C++ y el motor de ejecución integrado.

## 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. |

### 5.9.7.18.13 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

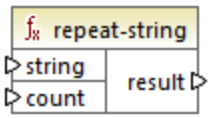
| Nombre | Tipo | Descripción |
|---|---|---|
| **string** | `xs:string` | Especifica la cadena de entrada. |
| **desired-length** | `xs:int` | Define la longitud que debe tener la cadena después de aplicar el relleno. |


© 2019-2025 Altova GmbH

| Nombre | Tipo | Descripción |
|---|---|---|
| **padding-char** | `xs:string` | Define el carácter que se debe usar como carácter de relleno. |

### 5.9.7.18.14 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

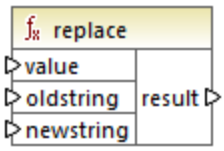
Disponible para Java, C#, C++ y el motor de ejecución integrado.

#### Parámetros

| Nombre | Tipo | Descripción |
|---|---|---|
| **string** | `xs:string` | La cadena de entrada. |
| **count** | `xs:int` | El número de veces que se debe repetir la cadena. |

### 5.9.7.18.15 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.

## Parámetros

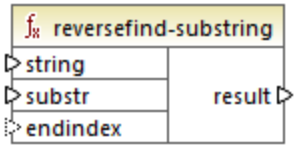
| Nombre | Tipo | Descripción |
|---|---|---|
| **value** | `xs:string` | El valor de entrada. |
| **oldstring** | `xs:string` | La cadena antigua que se va a reemplazar. |
| **newstring** | `xs:string` | La cadena nueva que reemplaza a la antigua. |

## Ejemplo

Consulte el apartado Reemplazar caracteres especiales 623.

## 5.9.7.18.16 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.

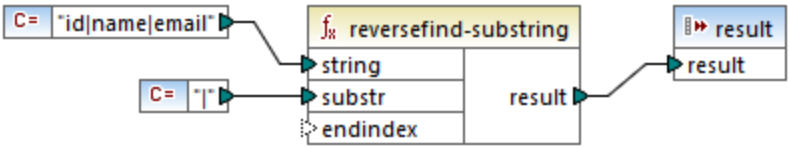
## Parámetros

| 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**. |

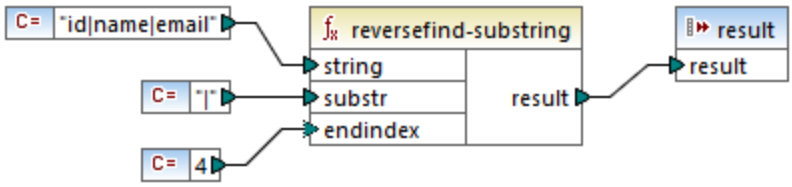

© 2019-2025 Altova GmbH

## 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`.

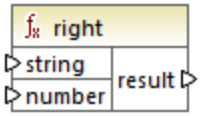



Si indica que el índice final es **4**, entonces la función busca solamente hasta el cuarto carácter. En ese caso, el resultado de la asignación sería **3**.




### 5.9.7.18.17 right

El resultado es una cadena que contiene los últimos caracteres **number** de la cadena de entrada.




## Lenguajes

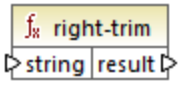
Disponible para Java, C#, C++ y el motor de ejecución integrado.

## Ejemplo

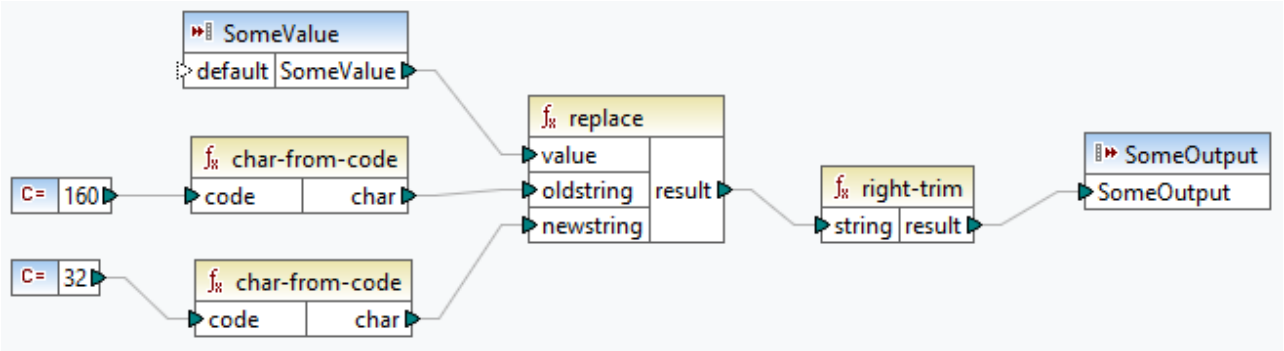
Si la cadena de entrada es `El veloz zorro marrón` y el número es **3**, la función devuelve `zorro`.

### 5.9.7.18.18 right-trim

El resultado es la cadena de entrada string después de eliminar todos los caracteres de espacio en blanco finales.




*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

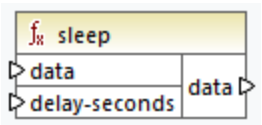
Disponible para Java, C#, C++ y el motor de ejecución integrado.

## Parámetros

| Nombre | Tipo | Descripción |
|---|---|---|
| **string** | `xs:string` | La cadena de entrada. |

## 5.9.7.18.19 sleep

La función **sleep** (*imagen siguiente*) suspende la transmisión de datos por N segundos. Al pasar una secuencia por la función, cada elemento de la secuencia quedará retenido durante un tiempo especificado. La función **sleep** es compatible con estos lenguajes de transformación: Java, C#, C++ y el lenguaje de transformación integrado. Para generar código se pueden usar los lenguajes Java, C# y C++. Para más información sobre la generación de código consulte Generador de código.




 © 2019-2025 Altova GmbH

## Parámetros

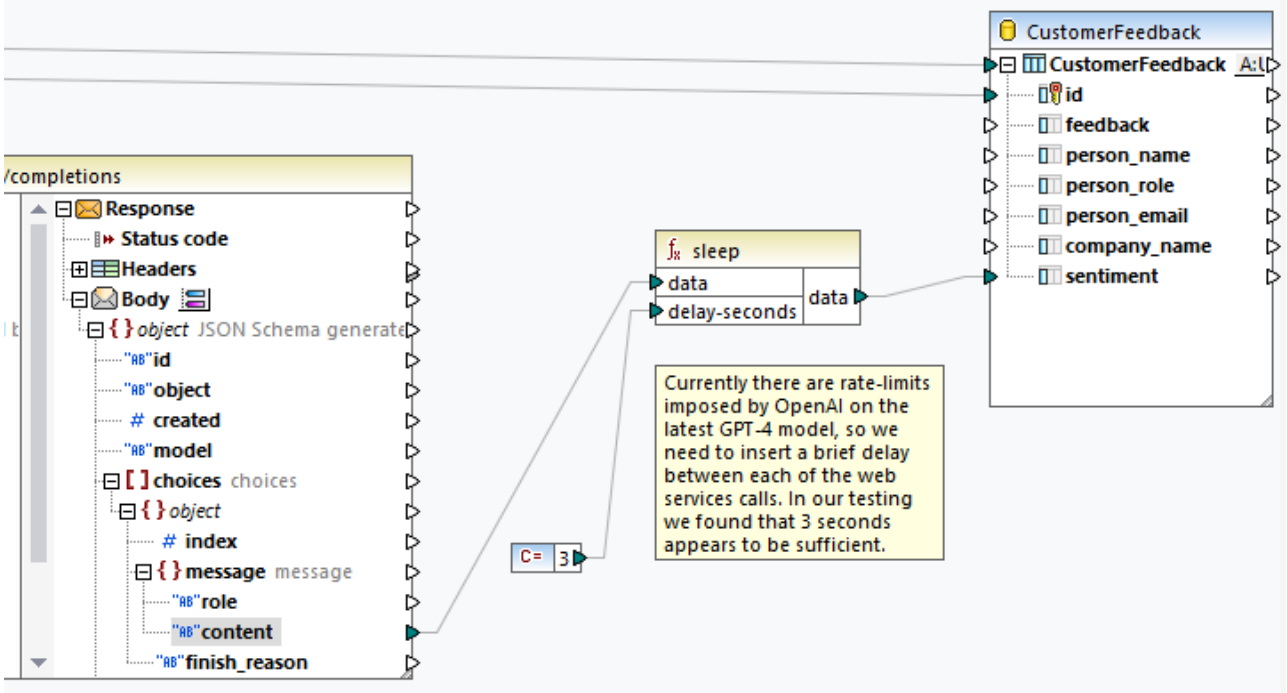
| Nombre | Tipo | Descripción |
|---|---|---|
| datos | cualquier nodo o tipo atómico | El parámetro de entrada de datos acepta cualquier valor, como p.ej. string (cadena). |
| delay-seconds | xs:double | El parámetro de entrada delay-seconds suspende la transmisión de datos por N segundos. También se aceptan segundos fraccionarios. |
| datos | cualquier nodo o tipo atómico | El parámetro de salida de datos recibe datos de la entrada y los transmite a un nodo de destino. |

## Ejemplo

Para ver un caso de uso en el que se utiliza la función de suspensión **sleep**, véase la siguiente asignación: **MapForceExamples\SentimentAnalysis.mfd**. A continuación se puede ver un extracto de la asignación. Para poder probar la asignación, necesitará las credenciales de acceso de su organización.

Dado que OpenAI impone limitaciones de volumen a las solicitudes de API que se pueden realizar, es posible que se produzca el error Demasiadas solicitudes. La función **sleep** permite omitir estas limitaciones de volumen configurando una suspensión.

En la estructura de respuesta de la llamada a servicios web que se muestra a continuación, el nodo content recibe datos como resultado de la solicitud enviada a la API OpenAI. Antes de cada llamada a servicios web, hay un retraso de tres segundos, y luego el valor del nodo content se asigna a la columna sentiment de la BD CustomerFeedback.

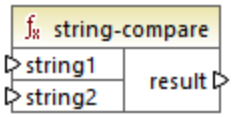



Para más información sobre este ejemplo y las funciones de IA en MapForce, consulte los siguientes artículos:

- *Integración de datos con IA*
- *Análisis de sentimiento de solicitudes de soporte técnico basadas en IA usando MapForce y GPT-4*
- *Clasificación de imágenes de BD basada en IA con Altova MapForce*

## 5.9.7.18.20 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

 © 2019-2025 Altova GmbH

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` 577.

## Lenguajes

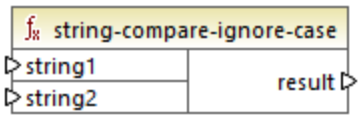
Disponible para Java, C#, C++ y el motor de ejecución integrado.

## Parámetros

| Nombre | Tipo | Descripción |
|---|---|---|
| `string1` | `xs:string` | La primera cadena de entrada. |
| `string2` | `xs:string` | La segunda cadena de entrada. |

## 5.9.7.18.21 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.

## Lenguajes

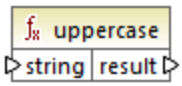
Disponible para Java, C#, C++ y el motor de ejecución integrado.

### Parámetros

| Nombre | Tipo | Descripción |
|---|---|---|
| `string1` | `xs:string` | La primera cadena de entrada. |
| `string2` | `xs:string` | La segunda cadena de entrada. |

### 5.9.7.18.22 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

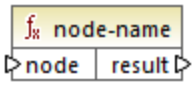
| Nombre | Tipo | Descripción |
|---|---|---|
| **string** | `xs:string` | La cadena de entrada. |

## 5.9.7.19 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.

### 5.9.7.19.1 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`.





© 2019-2025 Altova GmbH

### Lenguajes

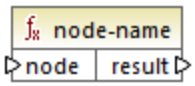
XQuery, XSLT 2.0, XSLT 3.0.

### Parámetros

| Nombre | Tipo | Descripción |
|---|---|---|
| **nodo** | `mf:node` | El nodo de entrada. |

## 5.9.7.19.2 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`** 594 .




### Lenguajes

XQuery, XSLT 2.0, XSLT 3.0.

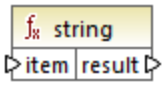
### Parámetros

| Nombre | Tipo | Descripción |
|---|---|---|
| **nodo** | `mf:node` | El nodo de entrada. |

## 5.9.7.19.3 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.)



### Lenguajes

XQuery, XSLT 2.0, XSLT 3.0.

### Parámetros

| Nombre | Tipo | Descripción |
|---|---|---|
| **item** | `mf:item` | El valor de entrada. |

## 5.9.7.20 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.

### 5.9.7.20.1 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.

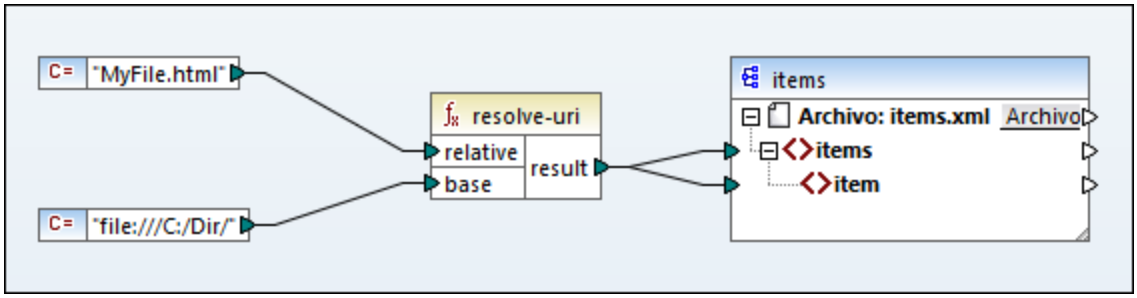
### Parámetros

| 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`**.


© 2019-2025 Altova GmbH

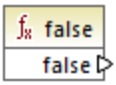



## 5.9.7.21 xpath2 | boolean functions (booleanas)

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.

### 5.9.7.21.1 false

Devuelve el valor booleano **false**.

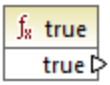



#### Lenguajes

XQuery, XSLT 2.0, XSLT 3.0.

### 5.9.7.21.2 true

Devuelve el valor booleano **true**.




#### Lenguajes

XQuery, XSLT 2.0, XSLT 3.0.

## 5.9.7.22 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** | |

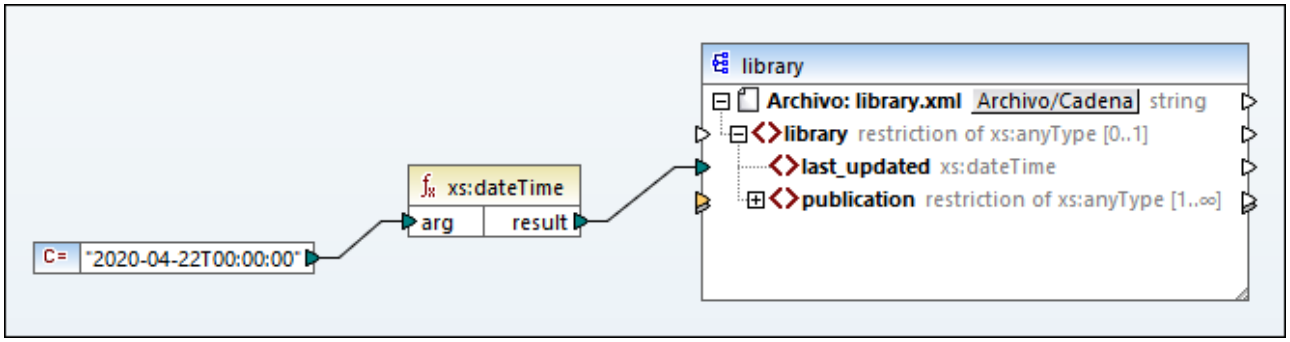
### Lenguajes

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).


© 2019-2025 Altova GmbH




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`.

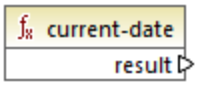
Al pasar el puntero del ratón encima del conector de entrada o de salida aparece el tipo de datos esperado para el elemento de la asignación (incluido el tipo de datos de los argumentos de la función).

## 5.9.7.23 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.

### 5.9.7.23.1 current-date

Devuelve la fecha actual (`xs:date`) del reloj del sistema.

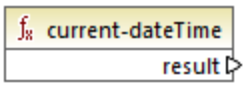



#### Lenguajes

XQuery, XSLT 2.0, XSLT 3.0.

### 5.9.7.23.2 current-dateTime

Devuelve la fecha y hora actual (`xs: dateTime`) del reloj del sistema.

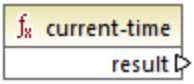


### Lenguajes

XQuery, XSLT 2.0, XSLT 3.0.

## 5.9.7.23.3 current-time

Devuelve la hora actual (`xs:time`) del reloj del sistema.




### Lenguajes

XQuery, XSLT 2.0, XSLT 3.0.

## 5.9.7.23.4 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.

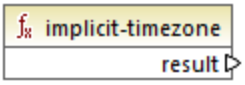
Las comparaciones (incluidas las funciones **`max-string`** y **`min-string`**) se basan en esta intercalación.

### Lenguajes

XQuery, XSLT 2.0, XSLT 3.0.

## 5.9.7.23.5 implicit-timezone

Devuelve el valor de la propiedad "implicit timezone" del contexto de evaluación.



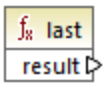

### Lenguajes

XQuery, XSLT 2.0, XSLT 3.0.


© 2019-2025 Altova GmbH

### 5.9.7.23.6 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 [82] actual, como se describe en el ejemplo siguiente.




## Lenguajes

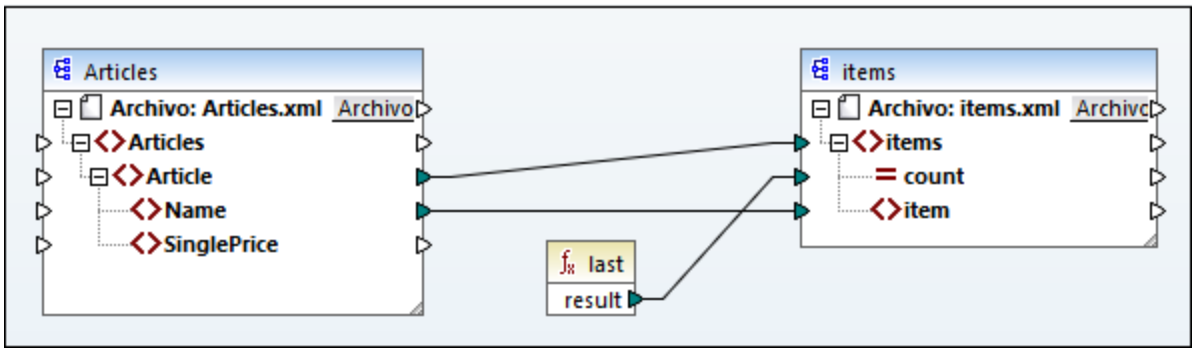
XQuery, XSLT 2.0, XSLT 3.0.

## 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:



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 85 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** 548 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.

## 5.9.7.24 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.

 © 2019-2025 Altova GmbH

- 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**
- **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.

### 5.9.7.24.1 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

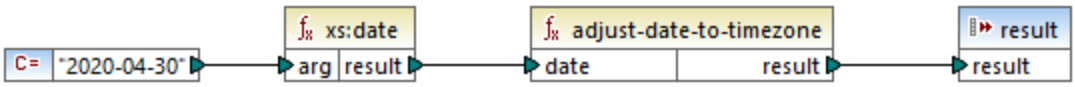
XQuery, XSLT 2.0.

#### 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`**.



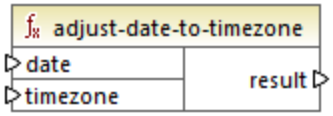

*Asignación XSLT 2.0*

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`**.

### 5.9.7.24.2 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.

 © 2019-2025 Altova GmbH




## Lenguajes

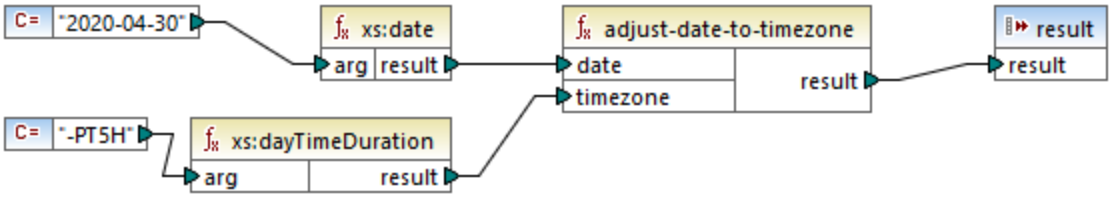
XQuery, XSLT 2.0.

## 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 702 correspondientes. El objetivo de la asignación es ajustar el huso horario a -5 horas. Este huso horario se puede expresar como **`-PT5H`**.

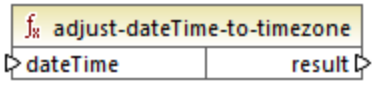



*Asignación XSLT 2.0*

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`**.

### 5.9.7.24.3 adjust-dateTime-to-timezone

Ajusta un valor `xs:dateTime` al huso horario implícito en el contexto de evaluación (el huso horario del sistema).




## Lenguajes

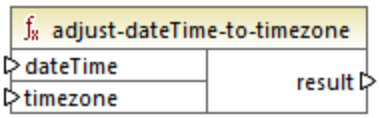
XQuery, XSLT 2.0.

## Parámetros

| Nombre | Tipo | Descripción |
|---|---|---|
| **dateTime** | `xs:dateTime` | El valor de entrada de tipo `xs:dateTime`. |

## 5.9.7.24.4 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.

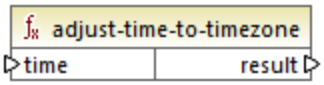
## Parámetros

| 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`. |


© 2019-2025 Altova GmbH

### 5.9.7.24.5 adjust-time-to-timezone

Ajusta un valor `xs:time` al huso horario implícito en el contexto de evaluación (el huso horario del sistema).




#### Lenguajes

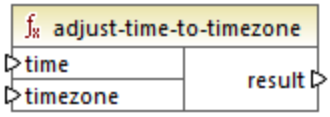
XQuery, XSLT 2.0.

#### Parámetros

| Nombre | Tipo | Descripción |
|---|---|---|
| **time** | `xs:time` | El valor de entrada de tipo `xs:time`. |

### 5.9.7.24.6 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 |

| Nombre | Tipo | Descripción |
|---|---|---|
| | | horas se puede expresar como **-PT5H**. |

### 5.9.7.24.7 day-from-date

Devuelve un elemento `xs:integer` que representa la parte del día del valor `xs:date` dado como argumento.

## Lenguajes

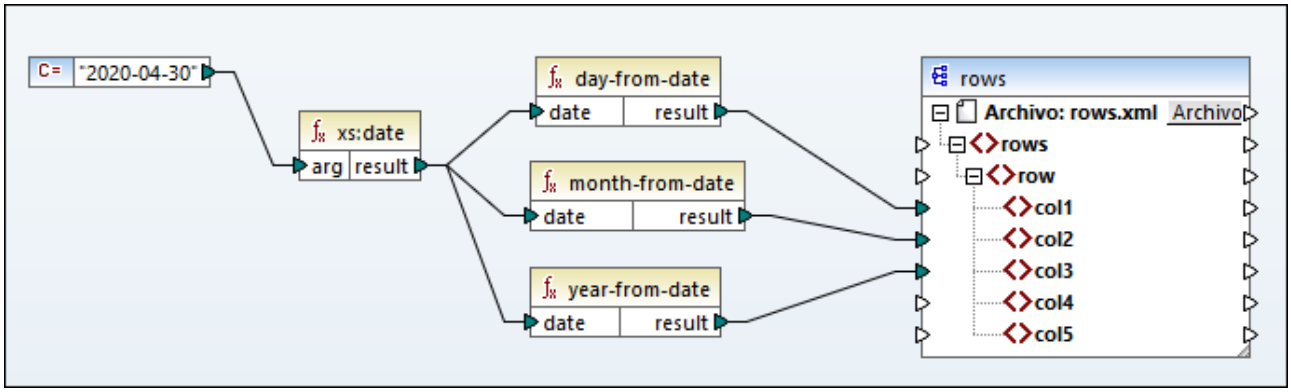
XQuery, XSLT 2.0.

## 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*

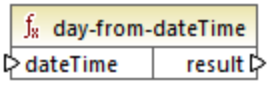
El resultado de la asignación es:


© 2019-2025 Altova GmbH

```
<rows>
   <row>
      <col1>30</col1>
      <col2>4</col2>
      <col3>2020</col3>
   </row>
</rows>
```

### 5.9.7.24.8 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`. |

### 5.9.7.24.9 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

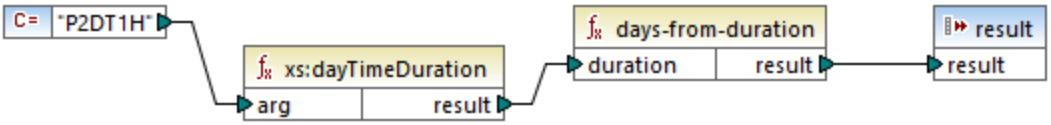
XQuery, XSLT 2.0.

#### Parámetros

| Nombre | Tipo | Descripción |
|---|---|---|
| **duration** | `xs:duration` | El valor de entrada de tipo `xs:duration`. |

## 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).

### 5.9.7.24.10 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`. |

### 5.9.7.24.11 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`. |


© 2019-2025 Altova GmbH

## 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).

### 5.9.7.24.12 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`. |

### 5.9.7.24.13 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`. |

### 5.9.7.24.14 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.

### 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).

## 5.9.7.24.15 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`. |

## 5.9.7.24.16 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`. |


© 2019-2025 Altova GmbH

### 5.9.7.24.17 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`. |

### 5.9.7.24.18 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`. |

### 5.9.7.24.19 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` | |

### 5.9.7.24.20 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`. |

### 5.9.7.24.21 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`. |

### 5.9.7.24.22 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. |


© 2019-2025 Altova GmbH

| Nombre | Tipo | Descripción |
|---|---|---|
| **dateTime2** | `xs:dateTime` | El segundo valor de entrada. |

### 5.9.7.24.23 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

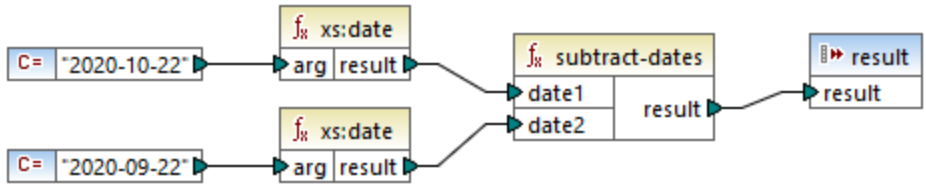
XQuery, XSLT 2.0.

#### 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.




### 5.9.7.24.24 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. |

| Nombre | Tipo | Descripción |
|---|---|---|
| **time2** | `xs:time` | El segundo valor de entrada. |

### 5.9.7.24.25 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`. |

### 5.9.7.24.26 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`. |

### 5.9.7.24.27 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.


© 2019-2025 Altova GmbH

### Lenguajes

XQuery, XSLT 2.0.

### Parámetros

| Nombre | Tipo | Descripción |
|---|---|---|
| **time** | `xs:time` | El valor de entrada de tipo `xs:time`. |

## 5.9.7.24.28 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`. |

## 5.9.7.24.29 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`. |

### 5.9.7.24.30 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`. |

## 5.9.7.25 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`** 724 ). 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`** 722 ).
- 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`** 722 ).
- Con argumento (que debe ser un nodo): la función se aplica al nodo especificado.

### 5.9.7.25.1 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, la función devuelve **false**.

 © 2019-2025 Altova GmbH

## Lenguajes

XQuery, XSLT 2.0, XSLT 3.0.

## Parámetros

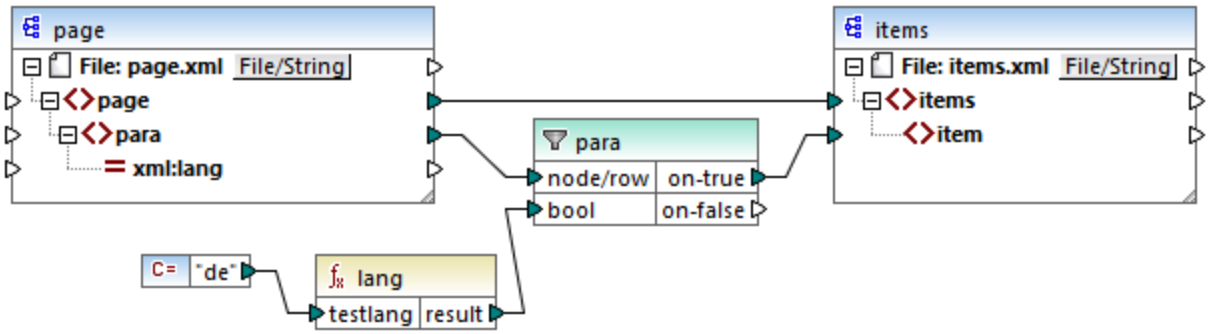
| Nombre | Tipo | Descripción |
|---|---|---|
| **testlang** | `xs:string` | El código del idioma que se comprueba, por ejemplo "en". |

## 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 de datos XSLT 2.0*

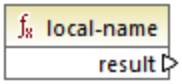
En la asignación anterior, por cada **para** de origen se crea un **item** en el componente de destino, de forma condicional. La condición la da un filtro que pasa al componente de destino solo los nodos en los que la función **`lang`** devuelve **true**. Es decir, solo los nodos cuyo atributo `xml:lang` es "de" (o un subconjunto de "de") cumplen la condición del filtro. En consecuencia, el resultado de la asignación es este:

```
<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** (ver también Contexto de la asignación 82).

### 5.9.7.25.2 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** 724 que toma un nodo de entrada como parámetro.

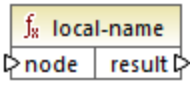



#### Lenguajes

XQuery, XSLT 2.0.

### 5.9.7.25.3 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. |

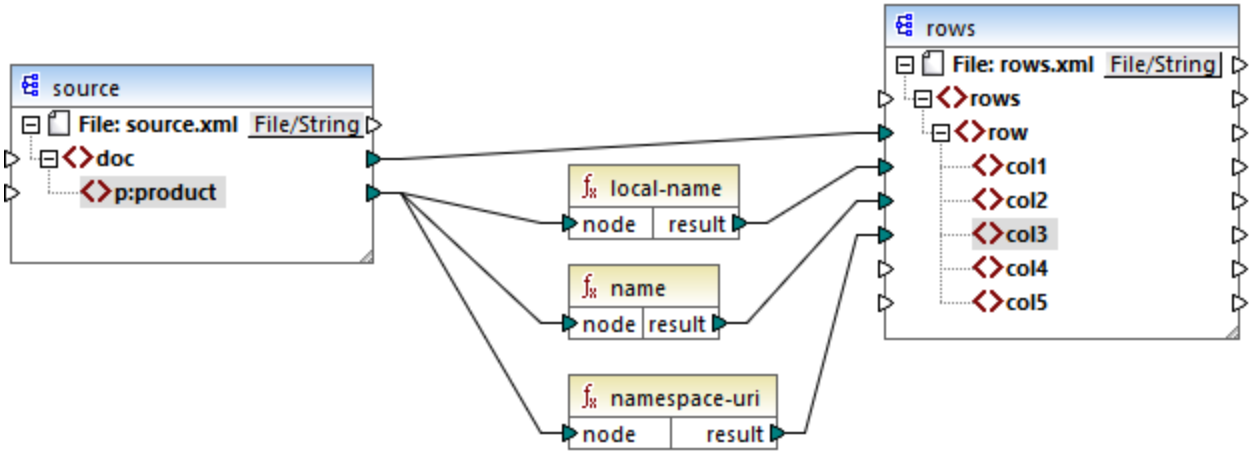
#### Ejemplo

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".


© 2019-2025 Altova GmbH

```
<?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:



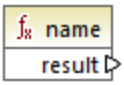

*Asignación XSLT 2.0*

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>
```

### 5.9.7.25.4 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`** 726 que toma un nodo de entrada como parámetro.

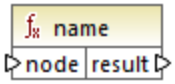


## Lenguajes

XQuery, XSLT 2.0.

## 5.9.7.25.5 name

Devuelve el nombre de un nodo.




## Lenguajes

XQuery, XSLT 2.0.

## Parámetros

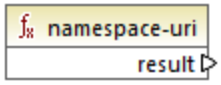
| Nombre | Tipo | Descripción |
|---|---|---|
| **node** | `node()` | El nodo de entrada. |

## Ejemplo

Consulte el ejemplo de la función `local-name` 724.

## 5.9.7.25.6 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` 727 que toma un nodo de entrada como parámetro.




## Lenguajes

XQuery, XSLT 2.0.


© 2019-2025 Altova GmbH

### 5.9.7.25.7 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. |

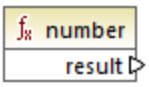
#### Ejemplo

Consulte el ejemplo de la función **local-name** 724.

### 5.9.7.25.8 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** 727 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*).

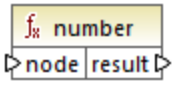



#### Lenguajes

XQuery, XSLT 2.0.

### 5.9.7.25.9 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*).




## Lenguajes

XQuery, XSLT 2.0.

## Parámetros

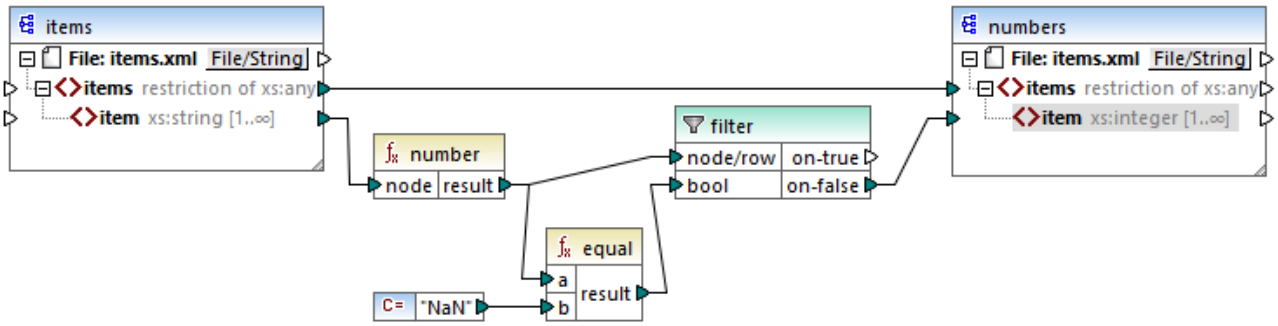
| Nombre | Tipo | Descripción |
|---|---|---|
| **node** | `mf:atomic` | El nodo de entrada. |

## 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>
```

 © 2019-2025 Altova GmbH

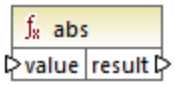
```
   <item>1</item>
   <item>2</item>
</items>
```

## 5.9.7.26 xpath2 | numeric functions (numéricas)

La biblioteca de funciones **xpath2 | numeric** ofrece las funciones numéricas **abs** y **round-half-to-even**.

### 5.9.7.26.1 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.

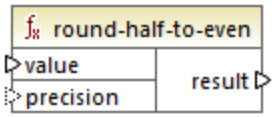
#### Parámetros

| Nombre | Tipo | Descripción |
|---|---|---|
| **value** | `xs:decimal` | El valor de entrada. |

### 5.9.7.26.2 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.

### Parámetros

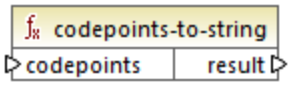
| Nombre | Tipo | Descripción |
|---|---|---|
| **value** | `xs:decimal` | Obligatorio. Indica el valor de entrada que se redondea. |
| **precision** | `xs:integer` | Opcional. Indica el número de decimales del redondeo. El valor predeterminado es **0**. |

## 5.9.7.27 xpath2 | string functions (cadena)

La biblioteca de funciones **xpath2 | string** permiten procesar cadenas (compararlas, convertirlas en mayúsculas o minúsculas, extraer subcadenas, etc.).

### 5.9.7.27.1 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` 738.




### 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

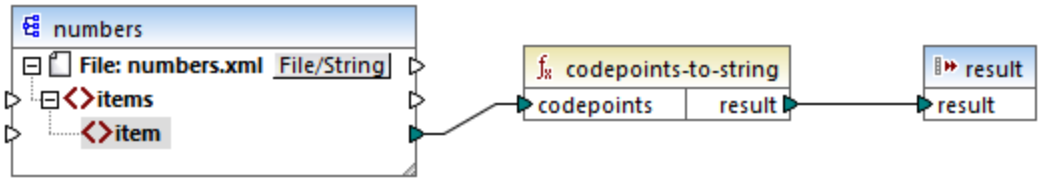
Este XML contiene varios elementos **item** que almacenan valores de puntos de código Unicode.

```
<items>
   <item>77</item>
   <item>97</item>
```

 © 2019-2025 Altova GmbH

```
    <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**.



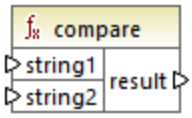

*Asignación XSLT 2.0*

El resultado de la función es **MapForce**.

### 5.9.7.27.2 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 732 de esta función en la que puede usar el elemento collation como argumento.




## Lenguajes

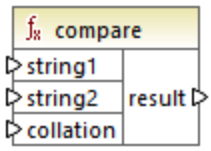
XQuery, XSLT 2.0.

## Parámetros

| Nombre | Tipo | Descripción |
|---|---|---|
| **string1** | xs:string | La primera cadena de entrada. |
| **string2** | xs:string | La segunda cadena de entrada. |

### 5.9.7.27.3 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

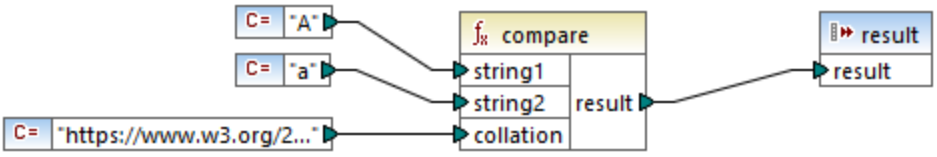
XQuery, XSLT 2.0.

## 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** 704 o puede ser un cotejo como `http://www.w3.org/2005/xpath-functions/collation/html-ascii-case-insensitive`. |

## Ejemplo

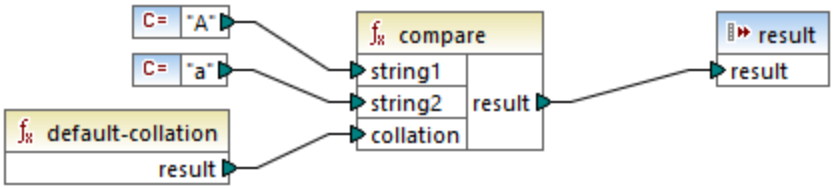
Esta asignación compara las cadenas "A" y "a" usando el elemento collation que no distingue entre mayúsculas y minúsculas `http://www.w3.org/2005/xpath-functions/collation/html-ascii-case-insensitive`, que viene dado por una constante.




*Asignación XSLT 2.0*


© 2019-2025 Altova GmbH

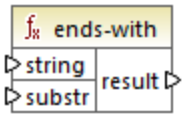
El resultado de la asignación anterior es **0** porque las dos se consideran iguales. Sin embargo, si reemplazamos el elemento collation con el que nos da la función `default-collation`, que usa los puntos de código Unicode predeterminados, el resultado se convierte en **-1** porque "A" es alfabéticamente menos que "a".




## 5.9.7.27.4 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 733 de esta función en la que puede usar el elemento collation como argumento.




### Lenguajes

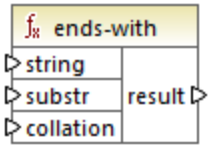
XQuery, XSLT 2.0.

### Parámetros

| Nombre | Tipo | Descripción |
|---|---|---|
| **string** | `xs:string` | La cadena de entrada (el "pajar"). |
| **substr** | `xs:string` | La subcadena (la "aguja"). |

## 5.9.7.27.5 ends-with

Devuelve **true** si **string** termina en **substr**, de lo contrario devuelve **false**. El valor devuelto es de tipo `xs:boolean`.




## Lenguajes

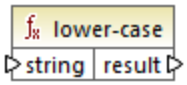
XQuery, XSLT 2.0.

## 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` 704 o puede ser un cotejo como `http://www.w3.org/2005/xpath-functions/collation/html-ascii-case-insensitive`. |

### 5.9.7.27.6 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.

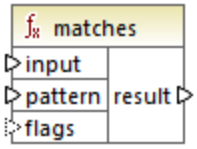
## Parámetros

| Nombre | Tipo | Descripción |
|---|---|---|
| **string** | `xs:string` | El valor de entrada. |


© 2019-2025 Altova GmbH

### 5.9.7.27.7 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

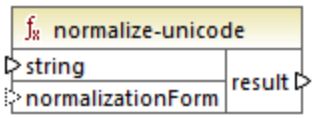
XQuery, XSLT 2.0.

## Parámetros

| Nombre | Tipo | Descripción |
|---|---|---|
| **input** | `xs:string` | La cadena de entrada. |
| **pattern** | `xs:string` | La expresión regular, véase Expresiones regulares 540. |
| **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:<br>`i` Ignorar mayúsculas y minúsculas. El valor predeterminado es no ignorarlas.<br>`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 **$** indican el principio y final de cada línea. El modo predeterminado es de cadena, donde las cadenas empiezan y terminan con los caracteres **^** and **$**.<br>`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.<br>`x` Ignora los espacios en blanco. El modo predeterminado es no ignorarlos. |

### 5.9.7.27.8 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

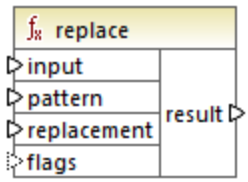
XQuery, XSLT 2.0.

#### 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).<br><br>También son compatibles las formas normales NFC, NFD, NFKC y NFKD. |

### 5.9.7.27.9 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, sólo se reemplaza la primera.





© 2019-2025 Altova GmbH

## Lenguajes

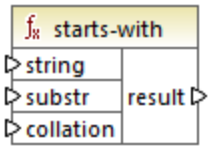
XQuery, XSLT 2.0.

## Parámetros

| Nombre | Tipo | Descripción |
|---|---|---|
| **input** | `xs:string` | La cadena de entrada. |
| **pattern** | `xs:string` | La expresión regular, véase Expresiones regulares 540. |
| **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` 735. |

## 5.9.7.27.10 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.

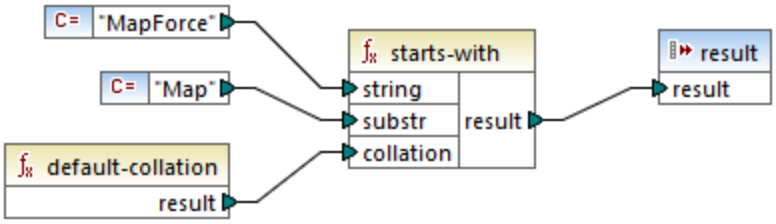
## Parámetros

| 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` 704 o puede ser un cotejo como `http://www.w3.org/2005/xpath` |

| Nombre | Tipo | Descripción |
|---|---|---|
| | | `-functions/collation/html-ascii-case-insensitive`. |

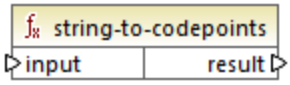
## Ejemplo

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.




## 5.9.7.27.11 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`** 730.




## Lenguajes

XQuery, XSLT 2.0.

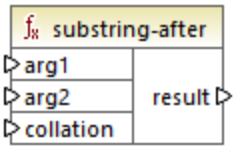
## Parámetros

| Nombre | Tipo | Descripción |
|---|---|---|
| **input** | **`xs:string`** | La cadena de entrada. |


© 2019-2025 Altova GmbH

### 5.9.7.27.12 substring-after

Devuelve la parte de **arg1** que aparece después de la cadena **arg2**.




## Lenguajes

XQuery, XSLT 2.0.

## Parámetros

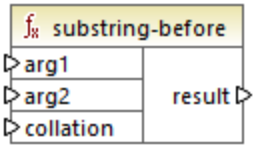
| Nombre | Tipo | Descripción |
|---|---|---|
| **arg1** | `xs:string` | haystack |
| **arg2** | `xs:string` | haystack |
| **collation** | `xs:string` | Indica qué cotejo usar para la comparación de cadenas. Puede provenir de la entrada de la función `default-collation` 704 o puede ser un cotejo como `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` 704, la función devuelve "Force".

### 5.9.7.27.13 substring-before

Devuelve la parte de **arg1** que aparece antes de la cadena **arg2**.



### Lenguajes

XQuery, XSLT 2.0.

### Parámetros

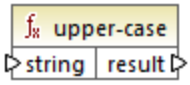
| Nombre | Tipo | Descripción |
|---|---|---|
| **arg1** | `xs:string` | La cadena de entrada (el "pajar"). |
| **arg2** | `xs:string` | La subcadena (la "aguja"). |
| **collation** | `xs:string` | Indica qué cotejo usar para la comparación de cadenas. Puede provenir de la entrada de la función `default-collation` 704 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` 704, la función devuelve "Map".

## 5.9.7.27.14 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. |

## 5.9.7.28 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.


© 2019-2025 Altova GmbH

### 5.9.7.28.1 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.

### 5.9.7.28.2 environment-variable

Devuelve el valor de una variable de entorno del sistema, si existe. El tipo de retorno es `xs: string`.

#### Lenguajes

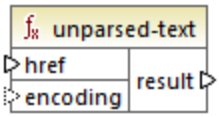
XSLT 3.0.

#### Parámetros

| Nombre | Tipo | Descripción |
|---|---|---|
| **name** | **`xs:string`** | El nombre de la variable de entorno. |

### 5.9.7.28.3 unparsed-text

Lee un recurso externo (por ejemplo un archivo) y devuelve una representación en cadena de ese recurso.




#### Lenguajes

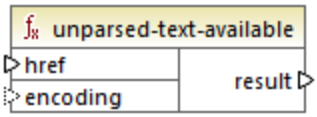
XSLT 3.0.

## 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. |

### 5.9.7.28.4 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.

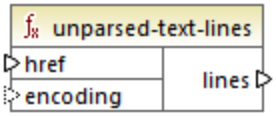
## Parámetros

| 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. |

### 5.9.7.28.5 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.


© 2019-2025 Altova GmbH




### Lenguajes

XSLT 3.0.

### Parámetros

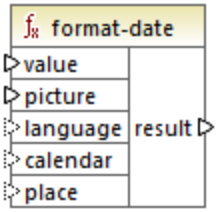
| Nombre | Tipo | Descripción |
|---|---|---|
| **href** | `xs:string` | Una cadena en forma de referencia URI. |
| **encoding** | `xs:string` | Argumento opcional. Indica el nombre del tipo de cifrado, por ejemplo "UTF-8", "UTF-16". Si el cifrado no se puede determinar automáticamente se asume que es UTF-8. |

## 5.9.7.29 xpath3 | formatting functions

Las funciones de formato de la biblioteca **xpath3** se usan para dar formato a los valores date, time e integer.

### 5.9.7.29.1 format-date

Devuelve una cadena que contiene un valor `xs:date` con formato para su visualización.




### Lenguajes

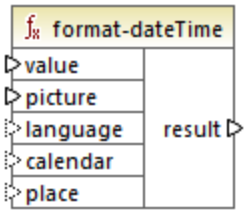
XSLT 3.0.

## 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.<br><br>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.<br><br>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. |

### 5.9.7.29.2 format-dateTime

Devuelve una cadena que contiene un valor `xs: dateTime` con formato para su visualización.




## Lenguajes

XSLT 3.0.

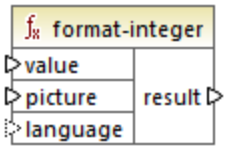

© 2019-2025 Altova GmbH

## Parámetros

| Nombre | Tipo | Descripción |
|---|---|---|
| **value** | `xs:dateTime` | El valor de entrada `xs:dateTime` al que se debe aplicar formato. |
| **picture** | `xs:string` | Parámetro obligatorio.<br><br>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.<br><br>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. |

### 5.9.7.29.3 format-integer

Da formato a un número entero conforme a una cadena de imagen dada y usando las convenciones del lenguaje natural indicado.




## Lenguajes

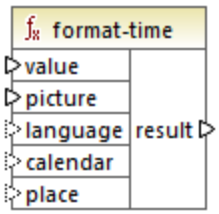
XSLT 3.0.

## 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.<br><br>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.<br><br>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. |

### 5.9.7.29.4 format-time

Devuelve una cadena que contiene un valor `xs: time` con formato para su visualización.




## Lenguajes

XSLT 3.0.


© 2019-2025 Altova GmbH

### Parámetros

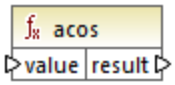
| Nombre | Tipo | Descripción |
|---|---|---|
| **value** | `xs:time` | El valor de entrada `xs: time` al que se debe aplicar formato. |
| **picture** | `xs:string` | Parámetro obligatorio.<br><br>Consulte el apartado 9.8.4.1 de la recomendación "XPath and XQuery Functions and Operators 3.1" W3C Recommendation (https://www.w3.org/TR/xpath-functions-31) para obtener más información. |
| **language** | `xs:string` | Parámetro opcional.<br><br>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. |

## 5.9.7.30 xpath3 | math functions

Las funciones matemáticas de la biblioteca **xpath3** se usan para realizar cálculos trigonométricos, entre otros.

### 5.9.7.30.1 acos

El resultado es el arco coseno de un ángulo en el rango de **0** a **pi**.




### Lenguajes

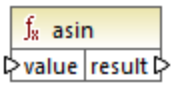
XSLT 3.0.

### Parámetros

| Nombre | Tipo | Descripción |
|---|---|---|
| **value** | `xs:double` | El valor de entrada. |

## 5.9.7.30.2 asin

El resultado es el arcoseno de un ángulo en el rango de **-pi/2** a **pi/2**.




### Lenguajes

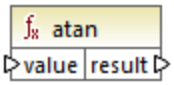
XSLT 3.0.

### Parámetros

| Nombre | Tipo | Descripción |
|---|---|---|
| **value** | `xs:double` | El valor de entrada. |

## 5.9.7.30.3 atan

El resultado es el arco tangente de un ángulo en el rango de **-pi/2** a **pi/2**.




### Lenguajes

XSLT 3.0.

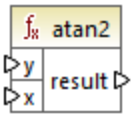
### Parámetros

| Nombre | Tipo | Descripción |
|---|---|---|
| **value** | `xs:double` | El valor de entrada. |


© 2019-2025 Altova GmbH

### 5.9.7.30.4 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

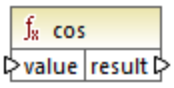
XSLT 3.0.

#### Parámetros

| Nombre | Tipo | Descripción |
|---|---|---|
| **y** | `xs:double` | La coordenada x. |
| **x** | `xs:double` | La coordenada y. |

### 5.9.7.30.5 cos

El resultado es el coseno del ángulo del parámetro de entrada. La unidad del valor es el radián.




#### Lenguajes

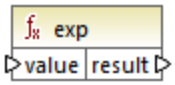
XSLT 3.0.

#### Parámetros

| Nombre | Tipo | Descripción |
|---|---|---|
| **value** | `xs:double` | El valor de entrada. |

### 5.9.7.30.6 exp

El resultado es *e* (el logaritmo natural base) elevado a la potencia de la entrada value.




#### Lenguajes

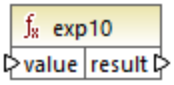
XSLT 3.0.

#### Parámetros

| Nombre | Tipo | Descripción |
|---|---|---|
| **value** | `xs:double` | El valor de entrada. |

### 5.9.7.30.7 exp10

El resultado es 10 elevado a la potencia de la entrada value.




#### Lenguajes

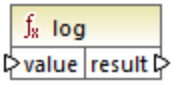
XSLT 3.0.

#### Parámetros

| Nombre | Tipo | Descripción |
|---|---|---|
| **value** | `xs:double` | El valor de entrada. |

### 5.9.7.30.8 log

El resultado es el logaritmo natural (en base e) de un valor.





© 2019-2025 Altova GmbH

## Lenguajes

XSLT 3.0.

## Parámetros

| Nombre | Tipo | Descripción |
|---|---|---|
| **value** | `xs:double` | El valor de entrada. |

## 5.9.7.30.9 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. |

## 5.9.7.30.10 pi

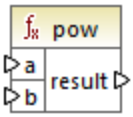
Devuelve una aproximación a la constante matemática **pi**.

## Lenguajes

XSLT 3.0.

### 5.9.7.30.11 pow

El resultado es el valor de **a** elevado a la potencia **b**.




#### Lenguajes

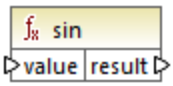
XSLT 3.0.

#### Parámetros

| Nombre | Tipo | Descripción |
|---|---|---|
| **a** | `xs:double` | El valor de entrada **a**. |
| **b** | `xs:double` | El valor de entrada **b**. |

### 5.9.7.30.12 sin

El resultado es el seno del ángulo del parámetro de entrada. La unidad del valor es el radián.




#### Lenguajes

XSLT 3.0.

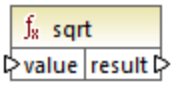
#### Parámetros

| Nombre | Tipo | Descripción |
|---|---|---|
| **value** | `xs:double` | El valor de entrada. |


© 2019-2025 Altova GmbH

### 5.9.7.30.13 sqrt

Devuelve la raíz cuadrada no negativa del argumento.




#### Lenguajes

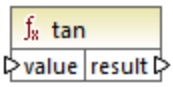
XSLT 3.0.

#### Parámetros

| Nombre | Tipo | Descripción |
|---|---|---|
| **value** | `xs:double` | El valor de entrada. |

### 5.9.7.30.14 tan

El resultado es la tangente del ángulo del parámetro de entrada. La unidad del valor es el radián.




#### Lenguajes

XSLT 3.0.

#### Parámetros

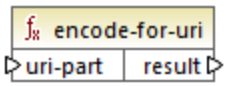
| Nombre | Tipo | Descripción |
|---|---|---|
| **value** | `xs:double` | El valor de entrada. |

## 5.9.7.31 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.

### 5.9.7.31.1 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

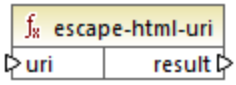
XSLT 3.0.

#### Parámetros

| Nombre | Tipo | Descripción |
|---|---|---|
| **uri-part** | `xs:string` | El valor del URI de entrada al que se debe aplicar el cifrado. |

### 5.9.7.31.2 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.

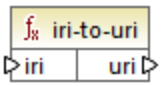
#### Parámetros

| Nombre | Tipo | Descripción |
|---|---|---|
| **uri** | `xs:string` | El valor del URI de entrada al que se deben aplicar las secuencias de escape. |


© 2019-2025 Altova GmbH

### 5.9.7.31.3 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. |

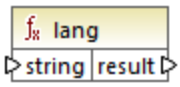
## 5.9.7.32 xslt | xpath functions

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 82 actual, que viene determinado por las conexiones de la asignación.

**Nota:** en la biblioteca de funciones **core** puede encontrar más funciones XPath 1.0.

### 5.9.7.32.1 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.

## Parámetros

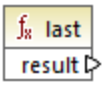
| Nombre | Tipo | Descripción |
|---|---|---|
| **string** | `xs:string` | El código del idioma que se comprueba, por ejemplo "en". |

## Ejemplo

Consulte el ejemplo de la función `lang` 722 de la biblioteca **xpath2**.

### 5.9.7.32.2 last

Devuelve el número de la posición del último nodo en la lista de nodos que se procesa.




## Lenguajes

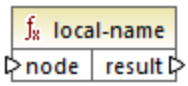
XSLT 1.0.

## Ejemplo

Consulte el ejemplo de la función `last` 705 de la biblioteca **xpath2**.

### 5.9.7.32.3 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. |

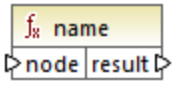

© 2019-2025 Altova GmbH

## Ejemplo

Consulte el ejemplo de la función `local-name` 724 de la biblioteca **xpath2**.

## 5.9.7.32.4 name

Devuelve el nombre del nodo dado como argumento.




## Lenguajes

XSLT 1.0, XSLT 2.0.

## Parámetros

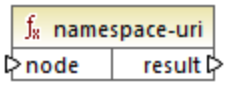
| Nombre | Tipo | Descripción |
|---|---|---|
| **node** | `node()` | El nodo de entrada. |

## Ejemplo

Consulte el ejemplo de la función `local-name` 724 de la biblioteca **xpath2**

## 5.9.7.32.5 namespace-uri

Devuelve el URI de espacio de nombres del nodo dado como argumento.




## Lenguajes

XSLT 1.0, XSLT 2.0.

## Parámetros

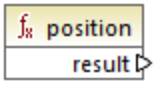
| Nombre | Tipo | Descripción |
|---|---|---|
| **node** | `node()` | El nodo de entrada. |

### Ejemplo

Consulte el ejemplo de la función `local-name` 724 de la biblioteca **xpath2**

### 5.9.7.32.6 position

Devuelve la posición del nodo actual en el conjunto de nodos que se está procesando.




### Lenguajes

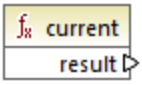
XSLT 1.0.

## 5.9.7.33 xslt | xslt functions

Las funciones de esta biblioteca con funciones XSLT 1.0.

### 5.9.7.33.1 current

Esta función no toma argumentos y devuelve el nodo actual.

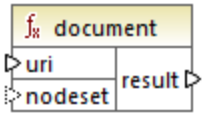



### Lenguajes

XSLT 1.0.

### 5.9.7.33.2 document

Accede a los nodos desde un documento externo. El resultado de la función se envía a un nodo del documento de destino.





© 2019-2025 Altova GmbH

## Lenguajes

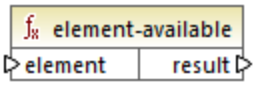
XSLT 1.0.

## 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. |

### 5.9.7.33.3 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

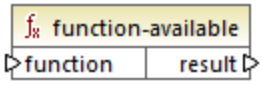
XSLT 1.0.

## Parámetros

| Nombre | Tipo | Descripción |
|---|---|---|
| **element** | `xs:string` | El nombre del elemento. |

### 5.9.7.33.4 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.




### Lenguajes

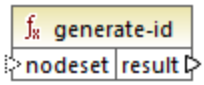
XSLT 1.0.

### Parámetros

| Nombre | Tipo | Descripción |
|---|---|---|
| **Función** | `xs:string` | El nombre de la función. |

## 5.9.7.33.5 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.

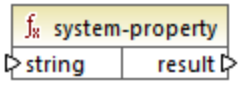
### Parámetros

| Nombre | Tipo | Descripción |
|---|---|---|
| **nodeset** | `nodo()` | Argumento opcional que indica el nodo de entrada. |

## 5.9.7.33.6 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.


© 2019-2025 Altova GmbH




## Lenguajes

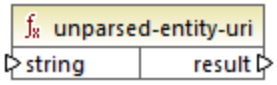
XSLT 1.0, XSLT 2.0, XSLT 3.0.

## 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`. |

### 5.9.7.33.7 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. |

# 6 Asignaciones avanzadas

**Sitio web de Altova:** Herramienta de integración de datos

Esta sección describe diferentes tipos de asignaciones avanzadas y se divide en varios apartados:


- Asignar nombres de nodo 763
- Procesamiento de archivos por lotes 784
- Analizar y serializar cadenas 791



© 2019-2025 Altova GmbH

## 6.1 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 sólo 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** 590 y **static-node-name** 592 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 sólo 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:

- XML
- CSV/FLF*

* Sólo compatible con las ediciones Professional o Enterprise de MapForce.

**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 sólo son compatibles con MapForce Professional o Enterprise Edition.

Para más información consulte el apartado Obtener acceso al nombre de los nodos 764 o el ejemplo del apartado Ejemplo: asignar nombres de elemento a valores de atributo 776.

## 6.1.1 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 sólo 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\MapForce2025\MapForceExamples\Tutorial\Products.xsd**. Este esquema viene

 © 2019-2025 Altova GmbH

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\MapForce2025\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.

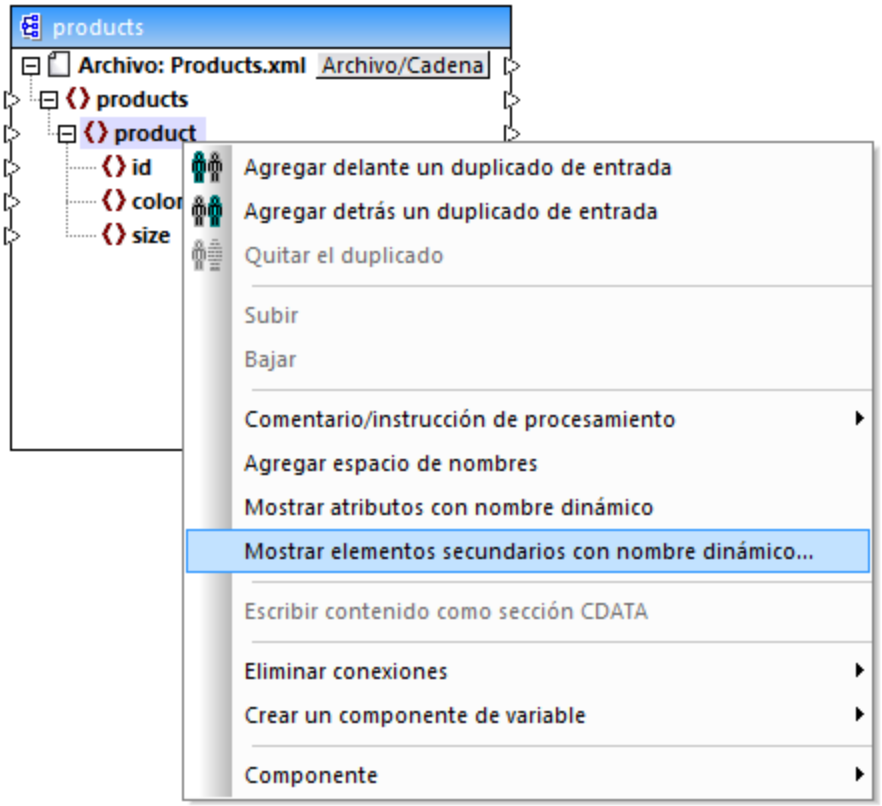



*Fig. 1: Habilitar nombres de nodo dinámicos (para elementos secundarios)*

**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 771).

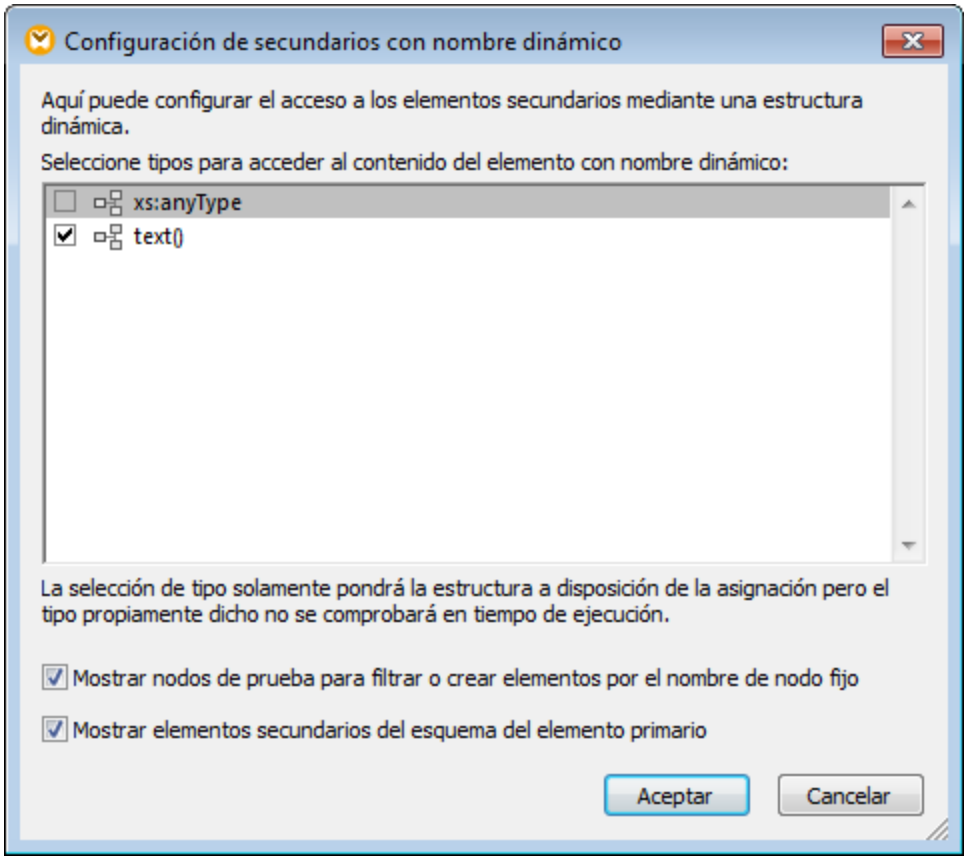



*Fig. 2: Cuadro de diálogo "Configuración de secundarios con nombre dinámico"*

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.

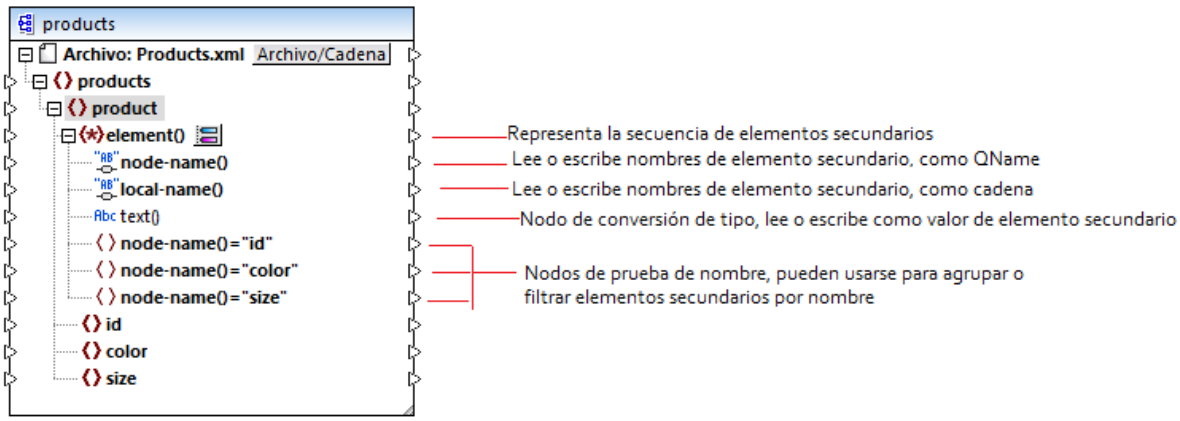



*Fig. 3: Nombres de nodo dinámicos habilitados (para elementos secundarios)*

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.

 © 2019-2025 Altova GmbH

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).

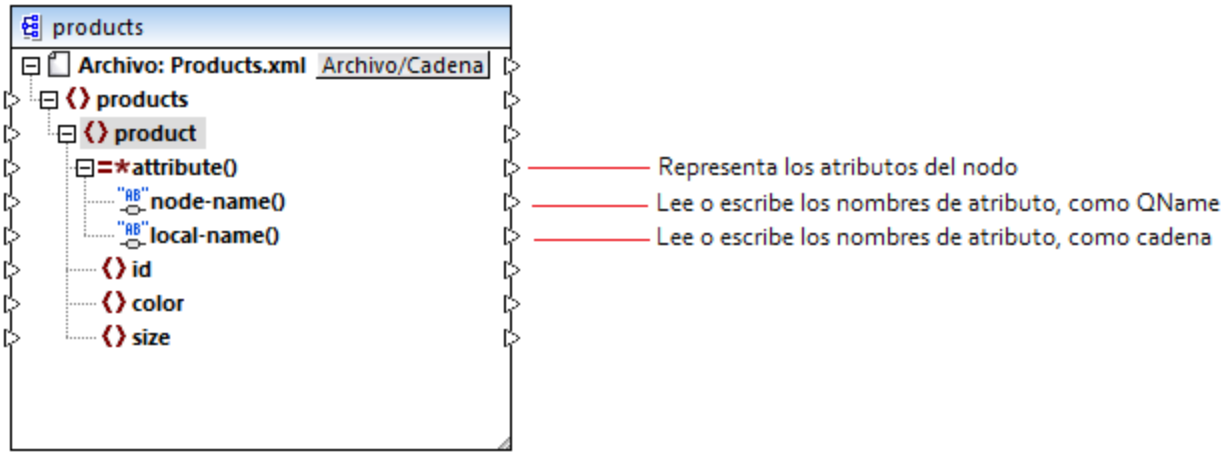



*Fig. 4: Nombres de nodo dinámicos habilitados (para atributos)*

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* 779).

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):

```
<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:

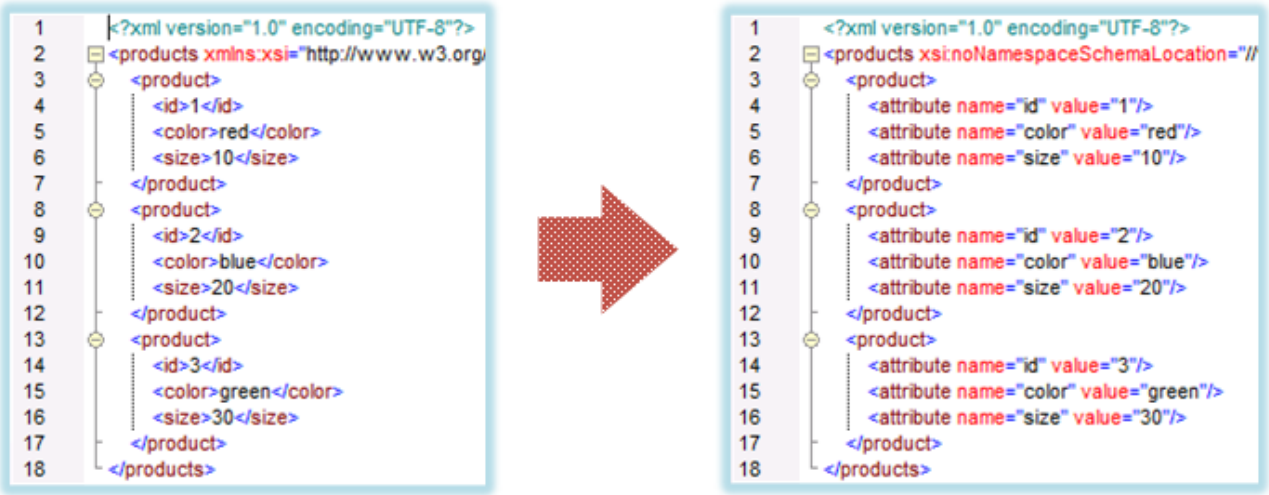



*Fig. 6: Asignación entre nombres de elemento XML y valores de atributo*

Para conseguir este objetivo debemos diseñar esta asignación de datos:

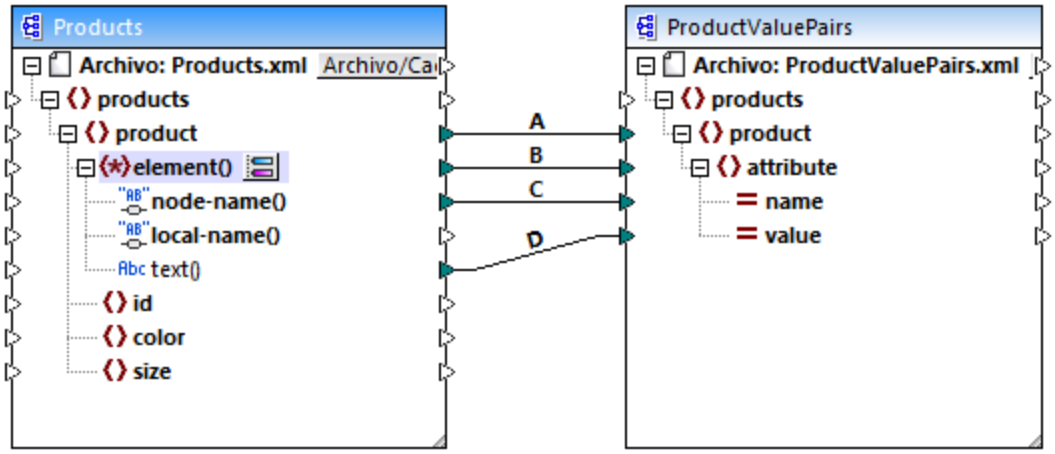



*Fig. 7: Asignar nombres de elemento XML a valores de atributo*

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

 © 2019-2025 Altova GmbH

asignación se describe más detalladamente en el apartado Ejemplo: asignar nombres de elemento a valores de atributo 776.

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()` sólo 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:

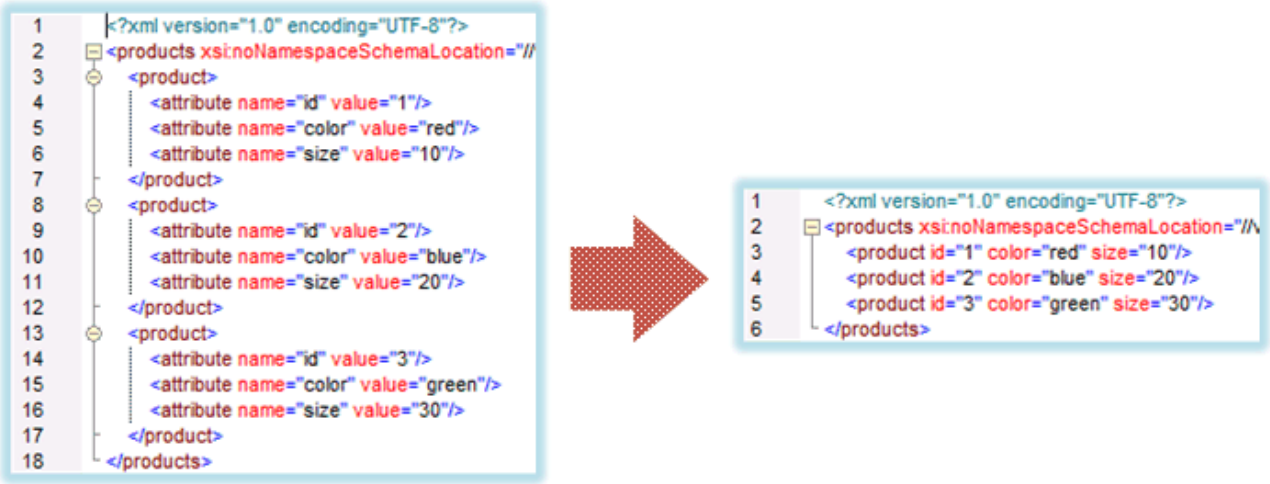



*Fig. 8: Asignar valores a nombres de atributo*

Para conseguir este objetivo debemos diseñar esta asignación de datos:

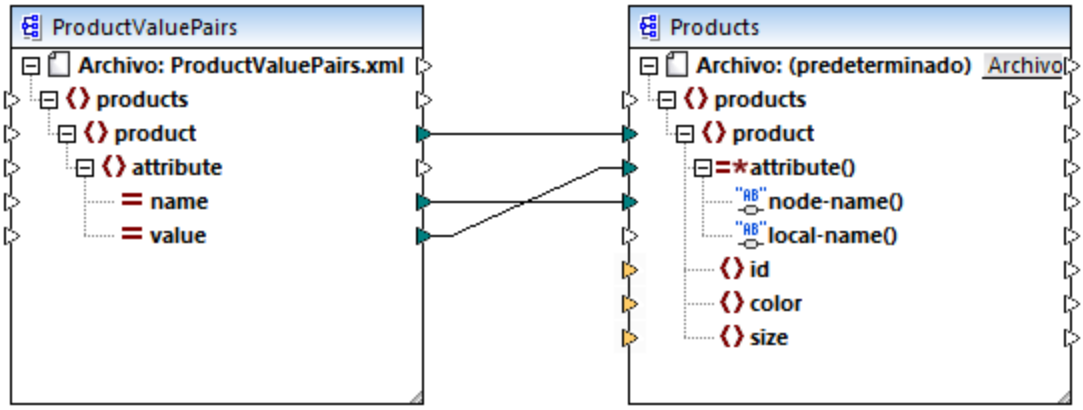



*Fig. 9: Asignar valores de atributo a nombres de atributo*

**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\MapForce2025\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 679.

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`.

 © 2019-2025 Altova GmbH

### 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 771 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* 771). Para ver un ejemplo consulte Ejemplo: agrupar y filtrar nodos según su nombre 779.

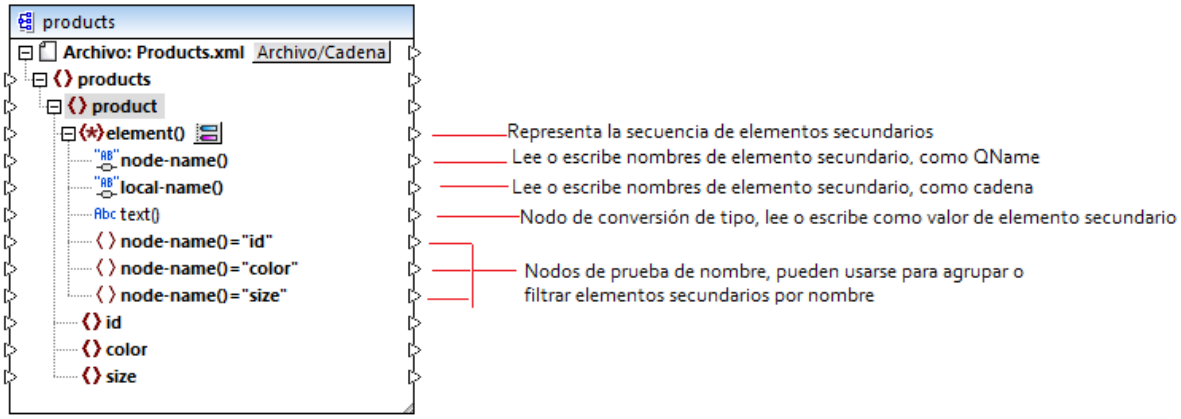
En general los nodos de prueba de nombre funcionan casi igual que los nodos de elemento normales para leer y escribir valores y subestructuras. Sin embargo, como la semántica de la asignación es distinta cuando está habilitado el acceso dinámico, existen algunas restricciones. Por ejemplo, no se puede 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).

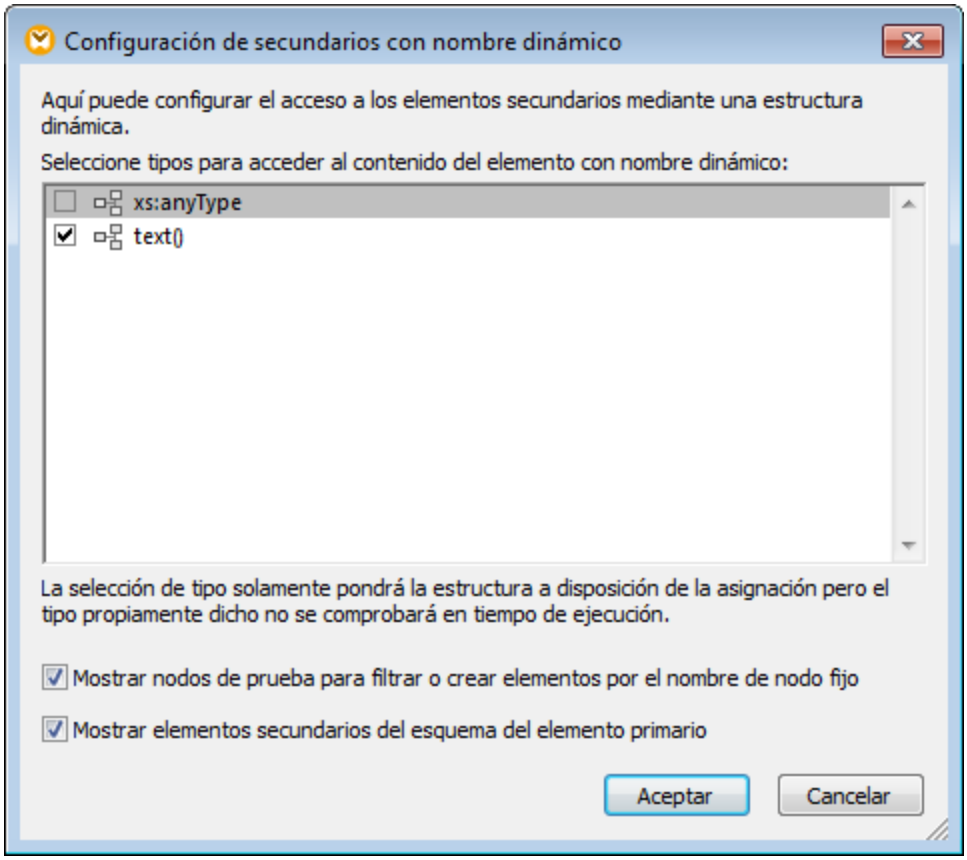
## 6.1.2 Obtener acceso a determinado tipo de nodos

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()`.




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()`.

 © 2019-2025 Altova GmbH




*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 776. 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.

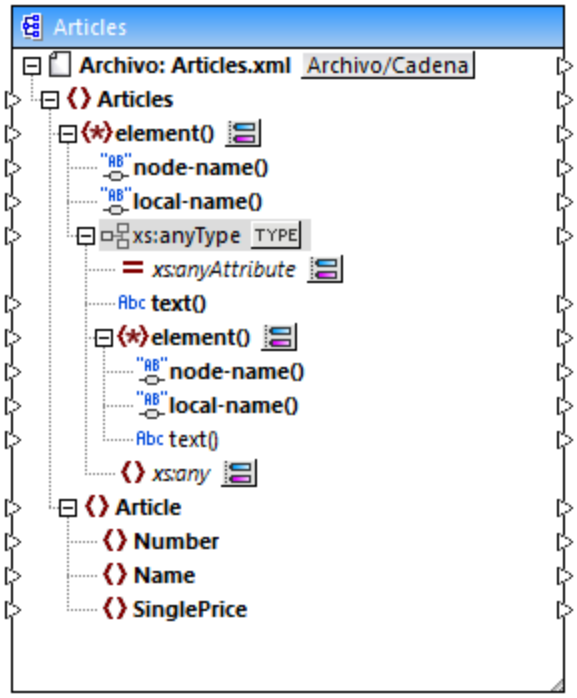
b) Agregar un filtro para leer de la instancia solamente el tipo complejo que debe corresponder.

Esta técnica se describe con más detalle en Ejemplo: agrupar y filtrar nodos según su nombre 779.

## 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 776 esta técnica no es necesaria porque la asignación sólo 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\MapForce2025\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".




 © 2019-2025 Altova GmbH

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* 144).
- 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* 151). Si usa esta opción, asegúrese de que el esquema permite que el atributo o elemento exista de verdad.

### 6.1.3 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\MapForce2025\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 sólo 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

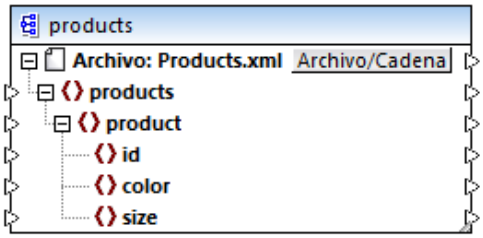
- En el menú **Insertar** haga clic en el comando **Archivo o esquema XML** y navegue hasta el archivo **<Documentos>\Altova\MapForce2025\MapForceExamples\Tutorial\Products.xml**. Este archivo XML apunta al esquema **Products.xsd** que está situado en la misma carpeta.

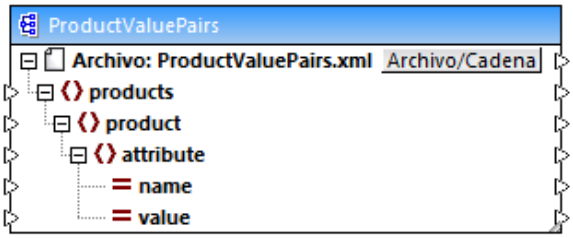
#### Paso nº2: agregar el componente XML de destino a la asignación

- En el menú **Insertar** haga clic en el comando **Archivo o esquema XML** y navegue hasta el archivo de esquema **<Documentos>\Altova\MapForce2025\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:

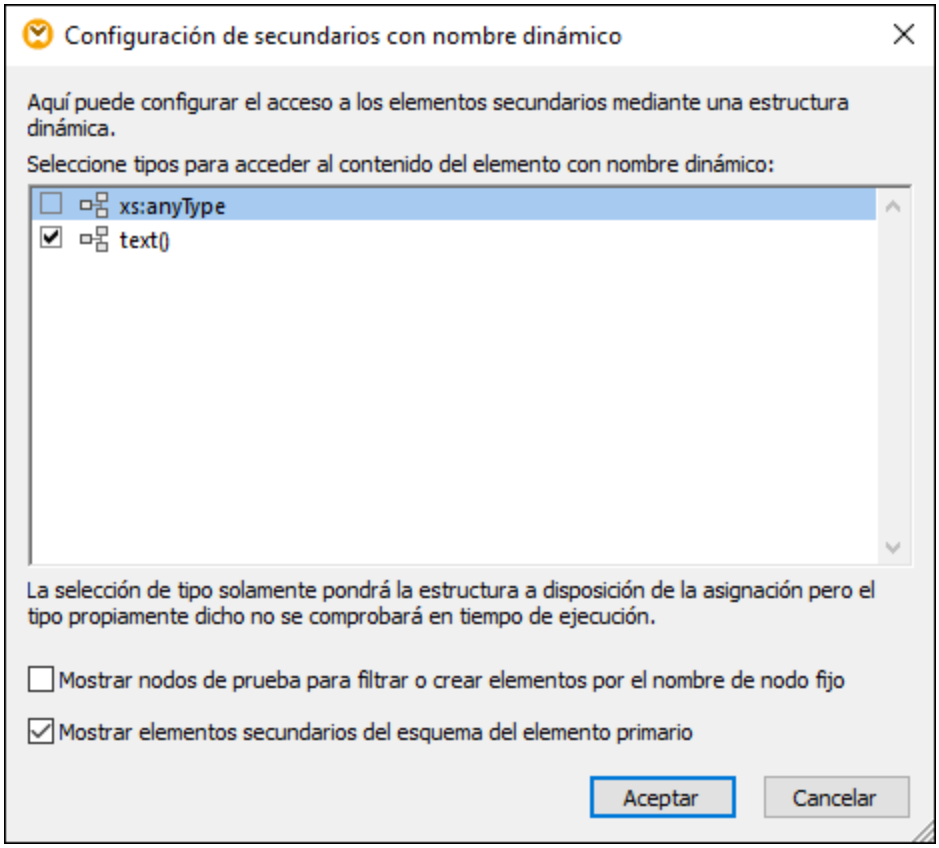
 © 2019-2025 Altova GmbH



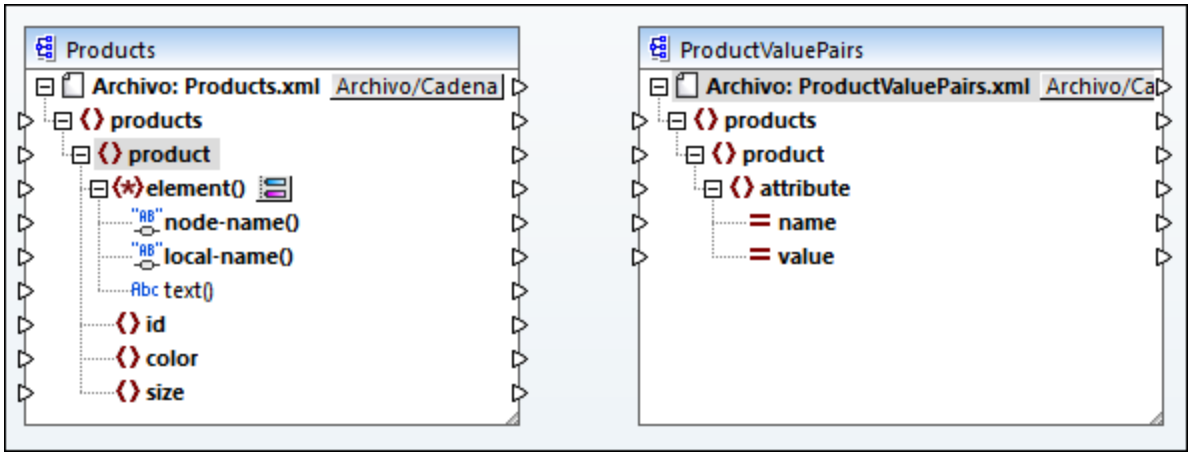




## Paso nº3: habilitar el acceso dinámico a los nodos secundarios

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.

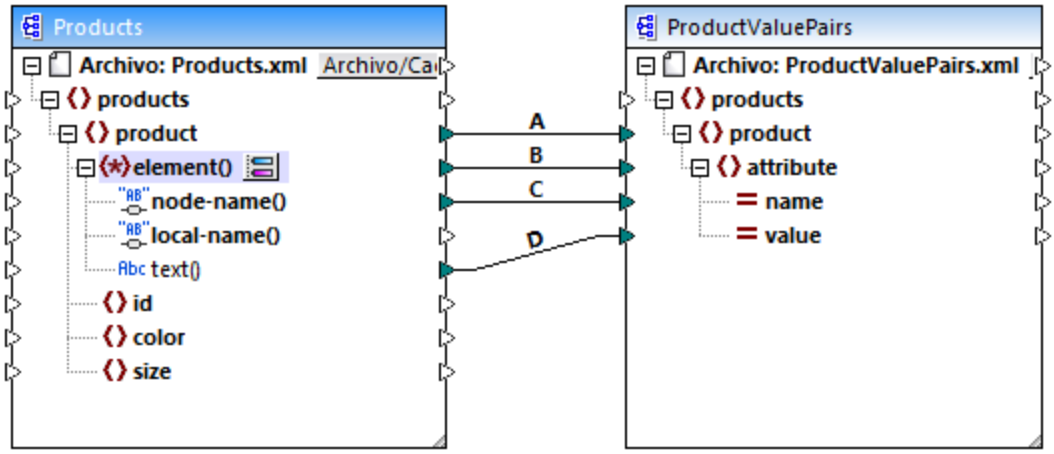



Observe que esto añade un nodo llamado `text()` al componente de origen. Este nodo se encargará de suministrar el contenido de los elementos secundarios a la asignación (en este caso, el valor de "id", "color" y "size").




## 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

 © 2019-2025 Altova GmbH

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*

## 6.1.4 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\MapForce2025\MapForceExamples\Tutorial\ReadPropertyList.mfd**.

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 610 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* 19).
2. En el menú **Insertar** haga clic en el comando **Archivo o esquema XML** y navegue hasta el archivo **<Documentos>\Altova\MapForce2025\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

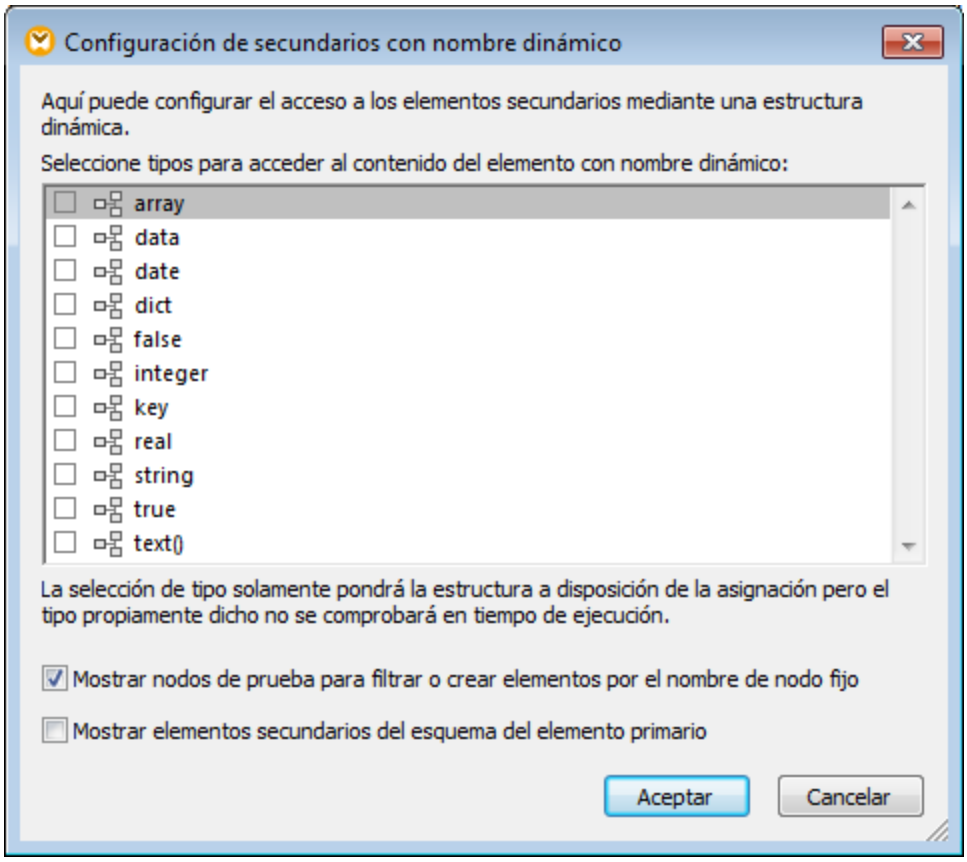
1. En el menú **Insertar** haga clic en el comando **Archivo de texto**. Ahora seleccione la opción *Utilizar procesamiento básico para archivos CSV...* en el cuadro de diálogo.
2. Añada un campo CSV más al componente con el botón **Anexar campo**.

 © 2019-2025 Altova GmbH

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 344 para obtener más información.

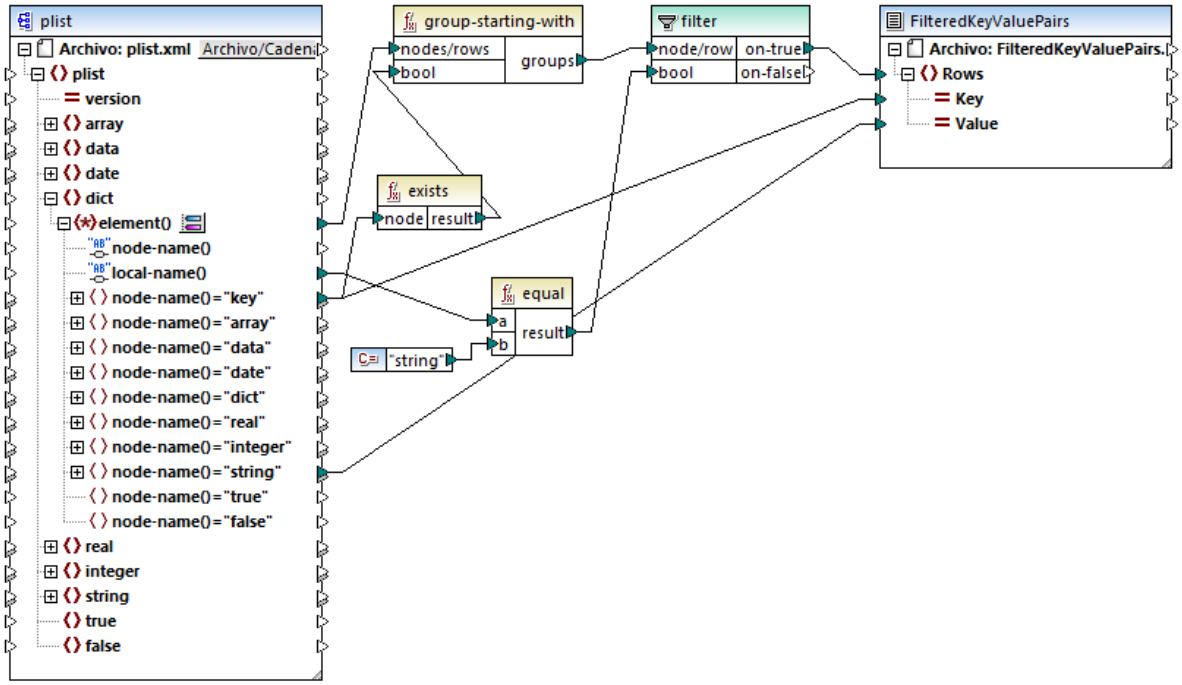
## Paso nº3: agregar el filtro y funciones

1. Arrastre las funciones **`equal`** 577, **`exists`** 597 y **`group-starting-with`** 610 desde la ventana Bibliotecas hasta el área de asignación (*véase Funciones básicas* 461).
2. Para agregar el filtro haga clic en el menú **Insertar** y seleccione el comando **Filtro: nodos/filas** (*véase Filtros y condiciones* 432).
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.




5. Dibuje las conexiones que aparecen en la imagen siguiente.


© 2019-2025 Altova GmbH




*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`.

## 6.2 Procesar archivos por lotes

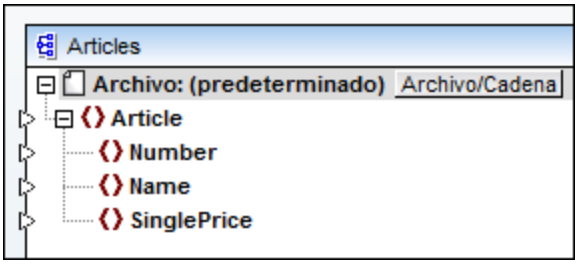
MapForce puede configurarse para procesar varios archivos simultáneamente (p. ej. todos los archivos de un directorio) cuando se ejecute la asignación. Gracias a esta característica podrá:

- 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).

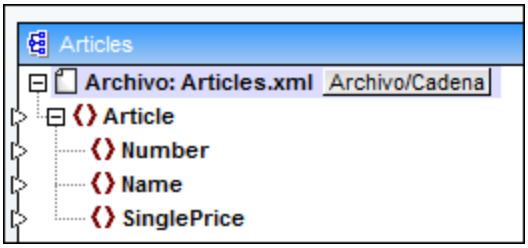
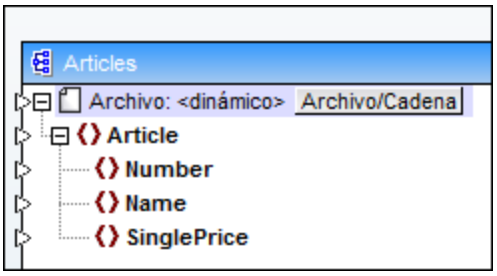
Hay dos maneras de configurar componentes de MapForce para que procese varios archivos a la vez:

- 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* 41). 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".<br><br>Si selecciona esta opción, el nodo raíz no tiene un conector de entrada porque no es relevante.<br><br>  |

 © 2019-2025 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.<br><br>Si el nombre de la entrada es idéntico al archivo de salida, entonces aparece como nombre del nodo raíz.<br><br>  <br><br>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.<br><br>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.<br><br>Cuando se selecciona esta opción, el nombre del nodo raíz aparece como **`Archivo: <dinámico>`**.<br><br>  <br><br>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 791 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 791 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:*

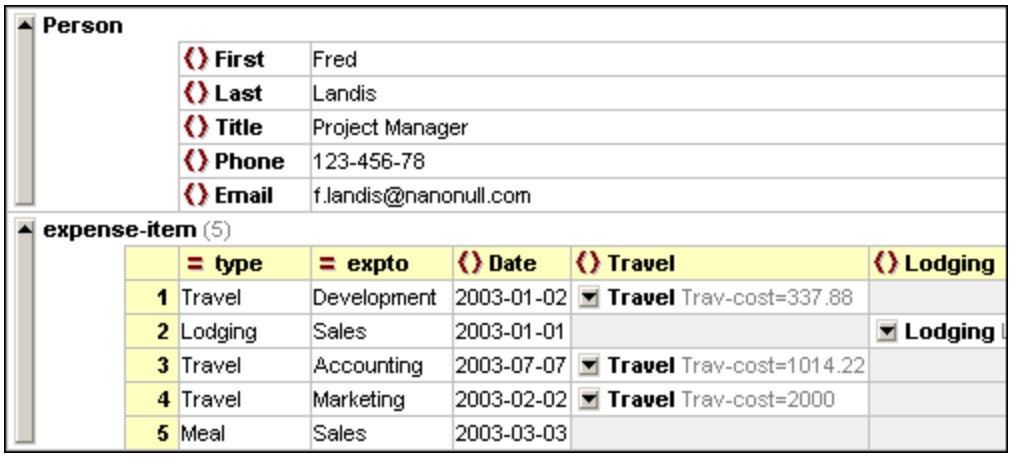
 © 2019-2025 Altova GmbH

| | |
|---|---|
| * | 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. |

## 6.2.1 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\MapForce2025\MapForceExamples\Tutorial\Tut-ExpReport-dyn.mfd**.

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 |

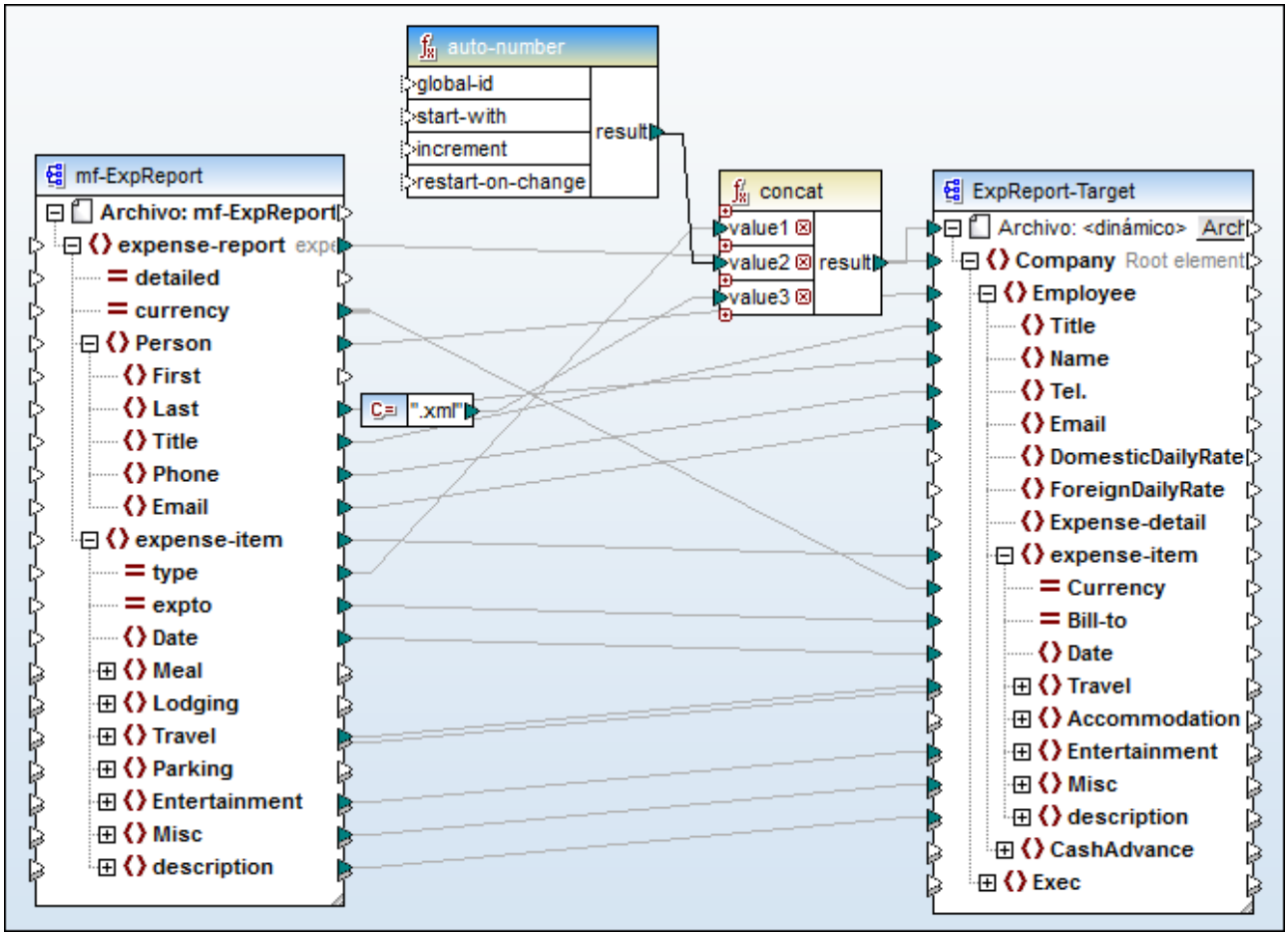
expense-item (5)

| | 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".

## 6.2.2 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\MapForce2025\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.

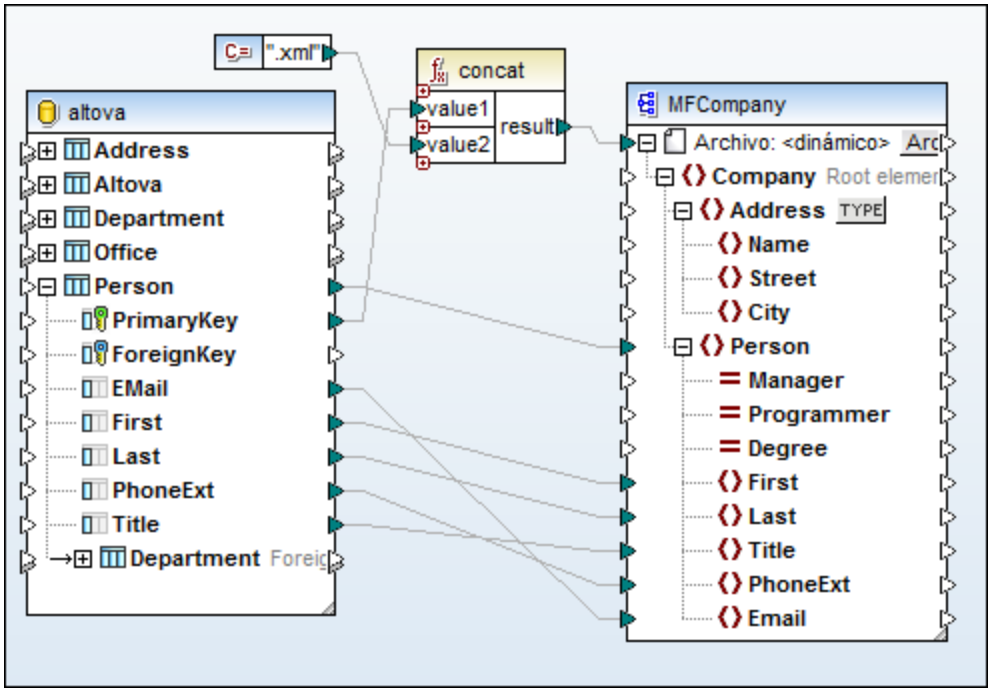
 © 2019-2025 Altova GmbH

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".


*© 2019-2025 Altova GmbH*

## 6.3 Analizar 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:

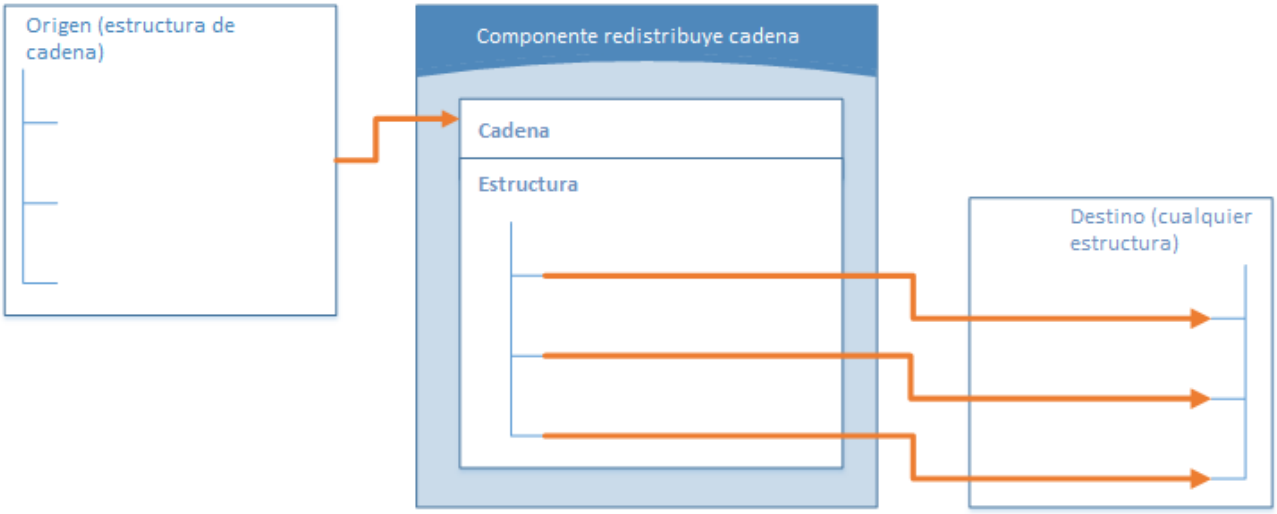
- Componentes de texto (texto CSV, texto de longitud fija)
- Archivos de esquema XML

La redistribución y serialización de cadenas de texto es compatible con el lenguaje de destino integrado de MapForce. Además, la redistribución de cadenas de texto a JSON o la serialización de JSON a partir de cadenas de texto es compatible con estos lenguajes de transformación: BUILT-IN, C# y Java.

### 6.3.1 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.

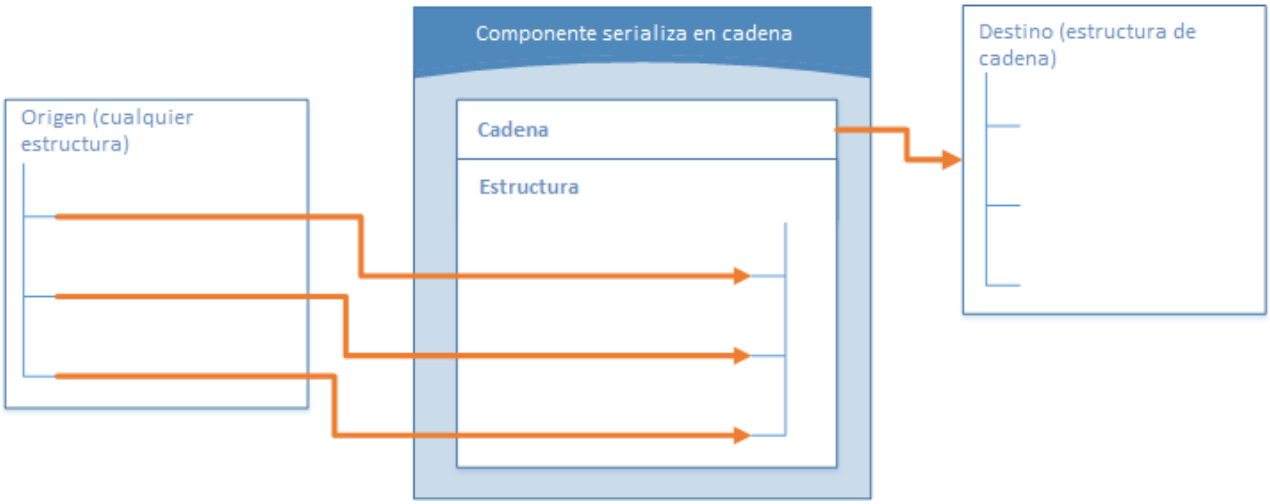
Puede utilizar un componente de redistribución/serialización de cadenas cuando necesite pasar a otro formato una cadena de texto que tiene una estructura (p. ej. datos XML guardados como cadena de texto en una BD). La redistribución de la cadena de origen a través del componente de redistribución/serialización 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.



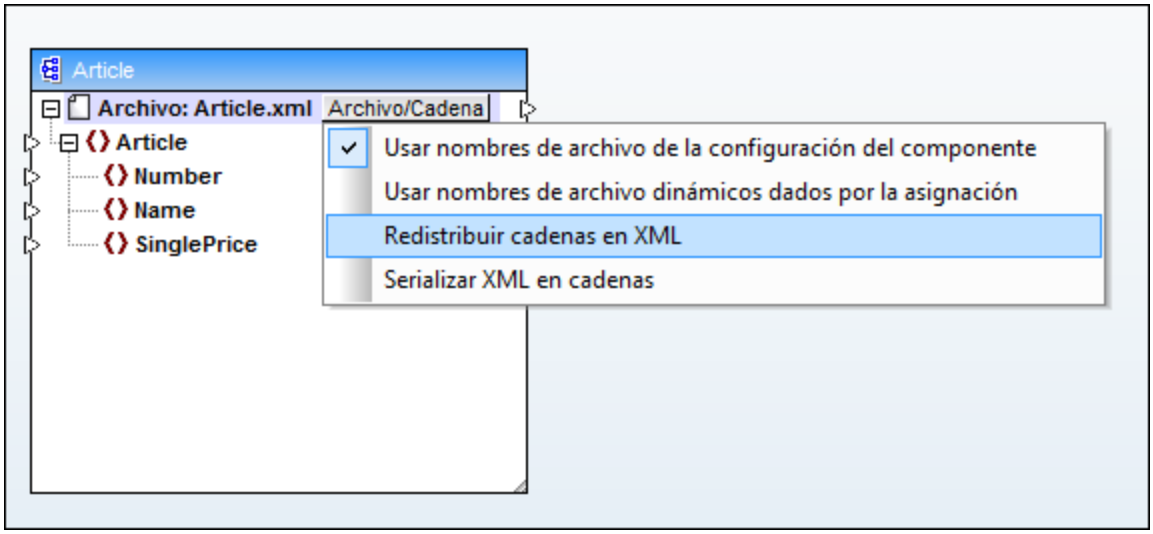

*Componente de serialización de cadena genérico*

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

 © 2019-2025 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) 794.

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* 41). Recuerde que no todas las opciones de configuración están disponibles si el componente está en modo de redistribución/serialización de cadenas.

## 6.3.2 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\MapForce2025\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>
```

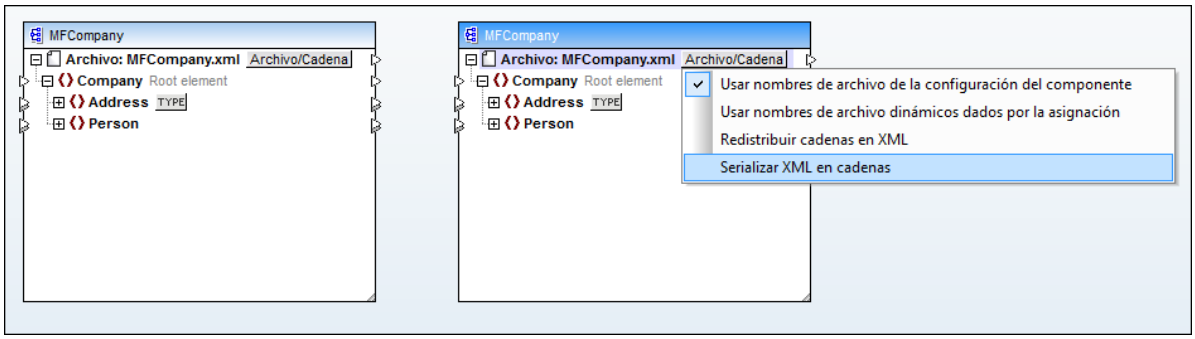
Nuestro objetivo es extraer cada elemento `<Person>` del archivo XML e insertarlo literalmente (incluidas las etiquetas XML) en un nuevo registro de BD de la tabla `PEOPLE` de una base de datos SQLite. La tabla `PEOPLE` solamente contiene dos columnas: `ID` y `PERSON`. Esta sería su definición completa:

```
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.

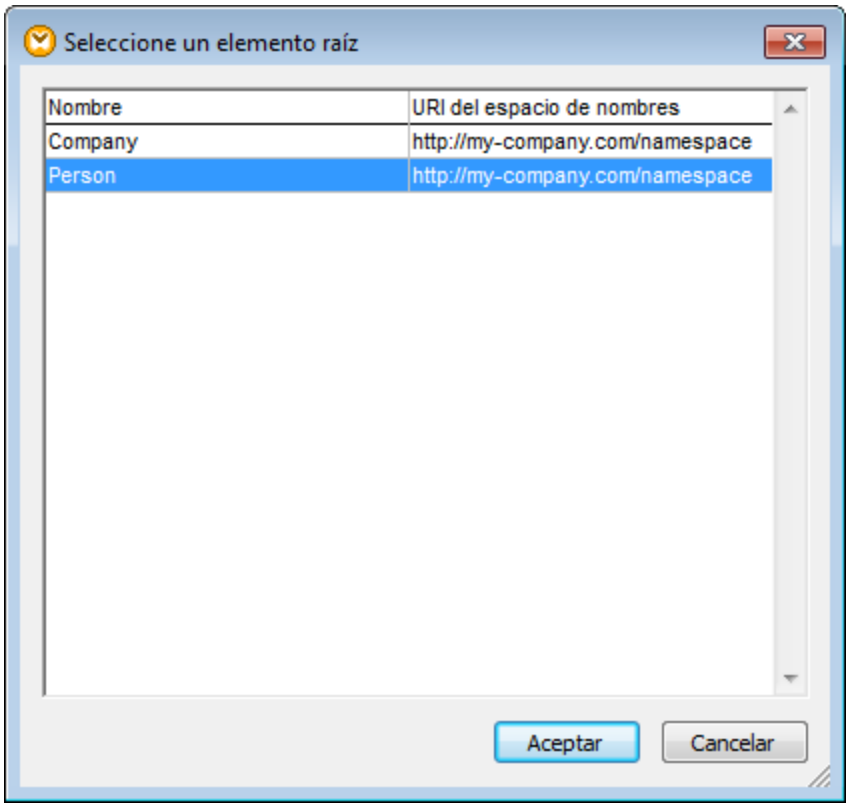
Para conseguir este objetivo debe seguir estos pasos:

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\MapForce2025\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.



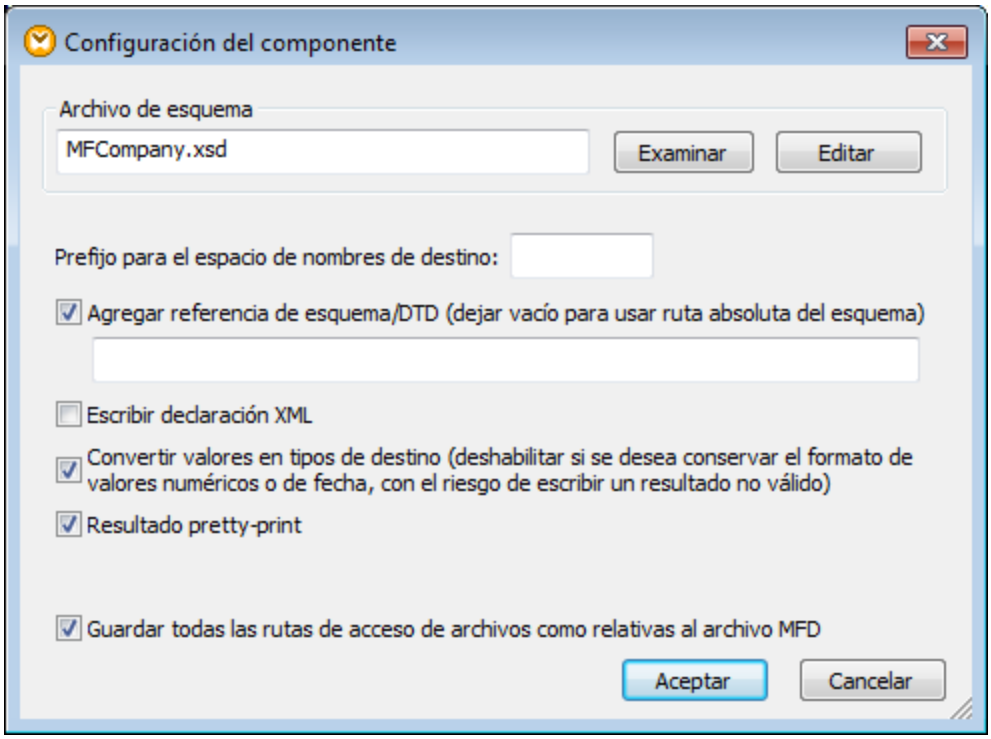

4. Haga clic con el botón derecho en la copia del componente XML y seleccione **Cambiar de elemento raíz** en el menú contextual. Ahora cambie el elemento raíz por `<Person>`.


© 2019-2025 Altova GmbH



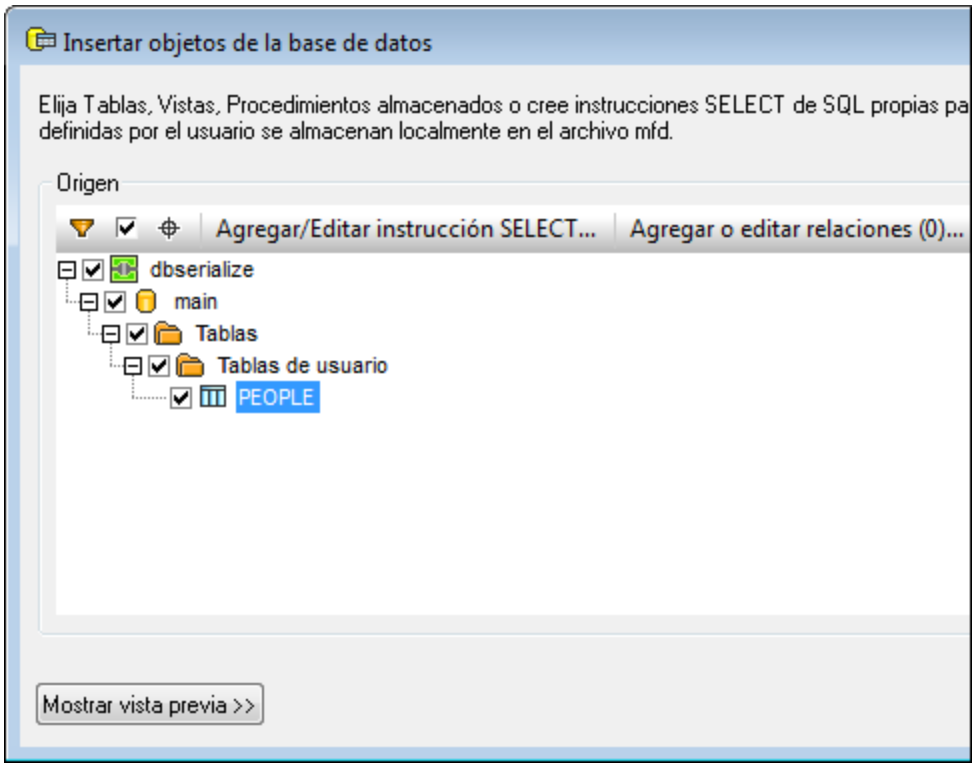

Por lo general, el elemento raíz puede cambiarse por cualquier elemento que tenga una declaración global (no local) en el esquema XML. Los elementos que no estén definidos globalmente en el esquema no aparecerán en el cuadro de diálogo "Seleccione un elemento raíz".

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>`.

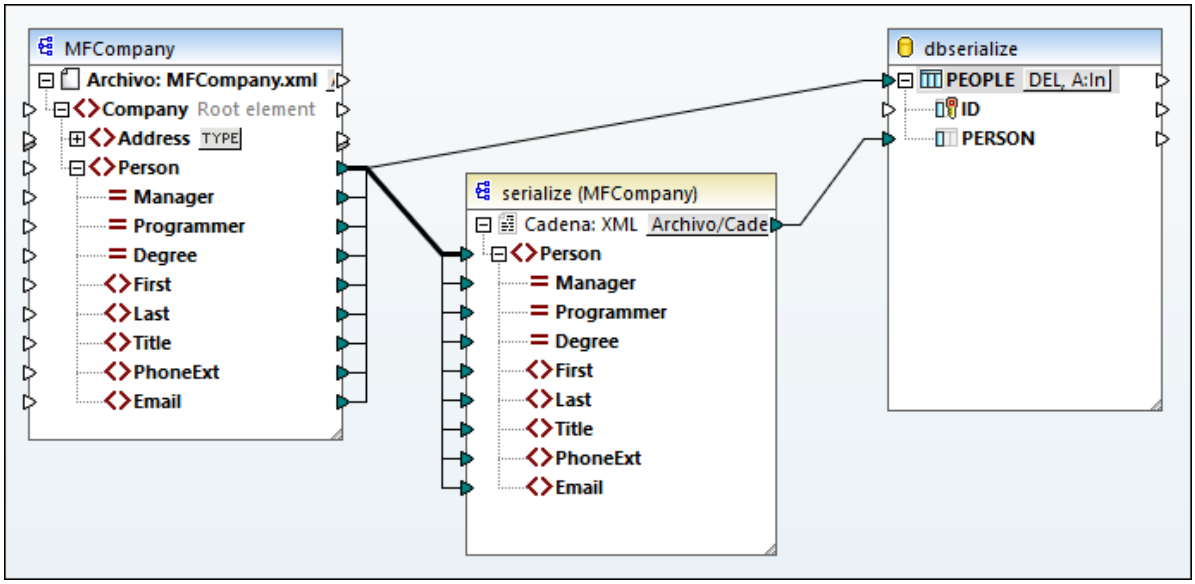



6. Añada el componente de base de datos SQLite de destino al área de asignación (la base de datos está en esta ubicación: **<Documentos>\Altova\MapForce2025\MapForceExamples\Tutorial\\dbserialize.db**). Para añadir el componente de base de datos seleccione el comando de menú **Insertar | Base de datos** (*véase también* *Conectarse a una base de datos* 178). Cuando solicite un objeto de base de datos, seleccione la tabla PEOPLE.

 © 2019-2025 Altova GmbH

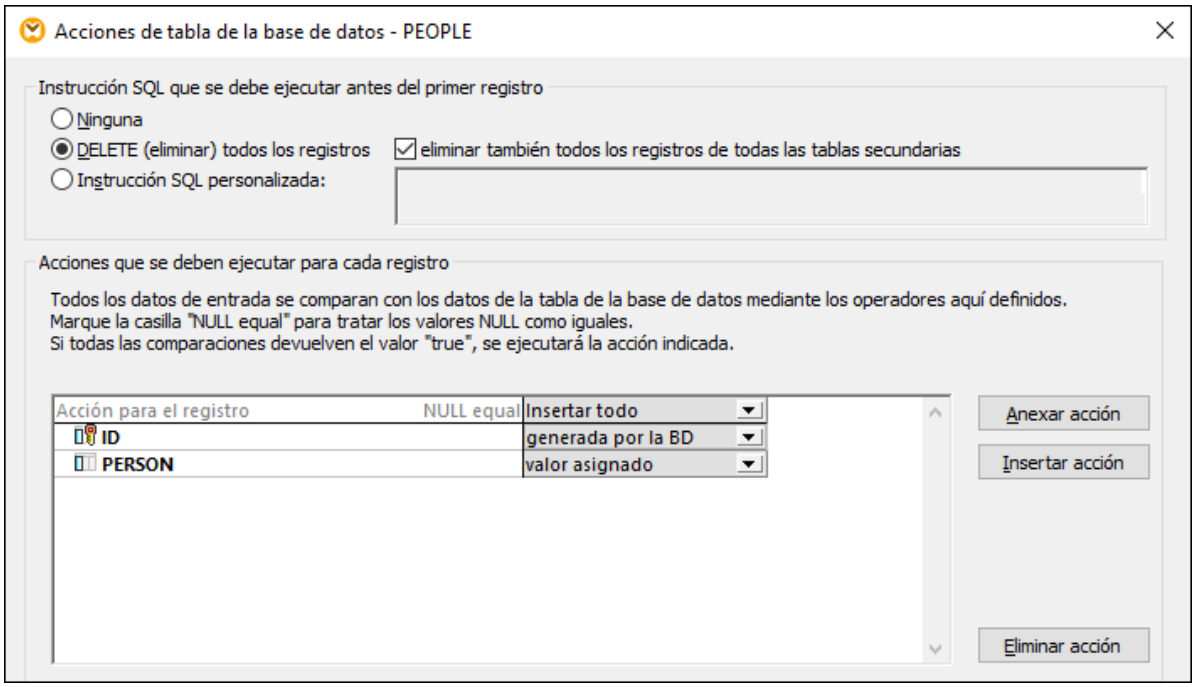



7. Conecte los componentes tal y como puede ver en la imagen siguiente. En el lado izquierdo de la asignación, el elemento `<Person>` se asigna al componente de serialización. En el lado derecho el valor de cadena serializado se inserta en la columna `PERSON` de la tabla de BD `PEOPLE`. Por último, el 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.

 © 2019-2025 Altova GmbH

# 7 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.

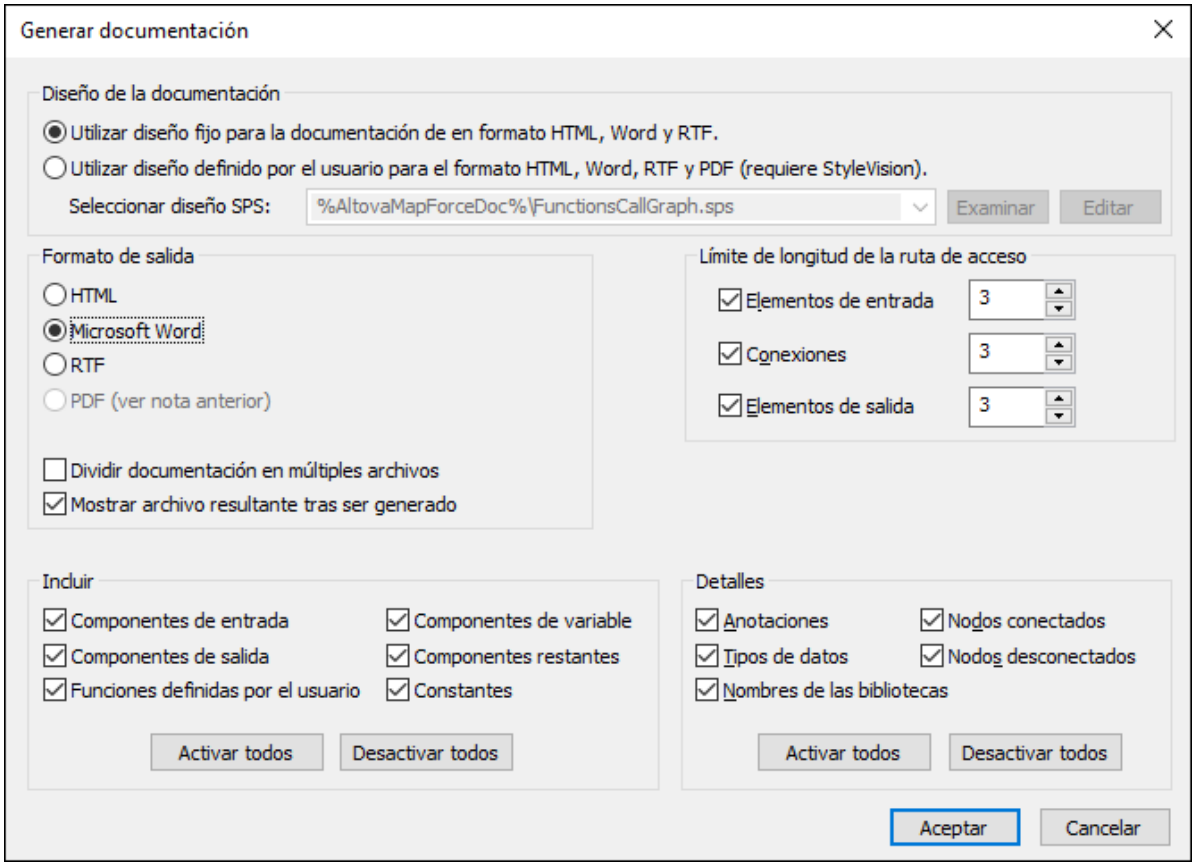
**Requisitos**

- 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\MapForce2025\Documentation\MapForce\`. For details, see Predefined StyleVision Power Stylesheets 801.
- 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.

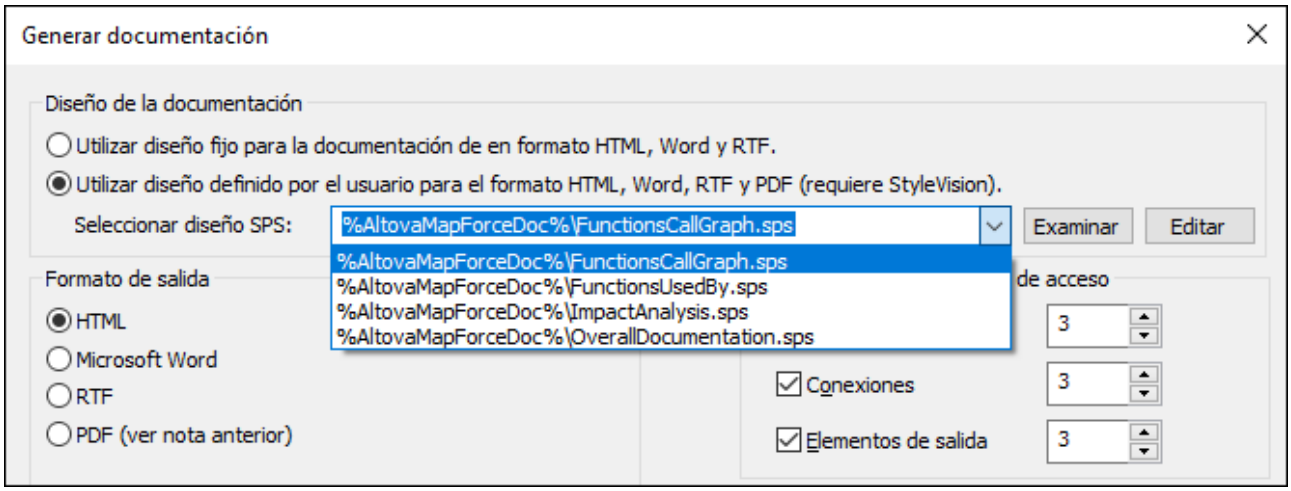
 © 2019-2025 Altova GmbH

## 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):

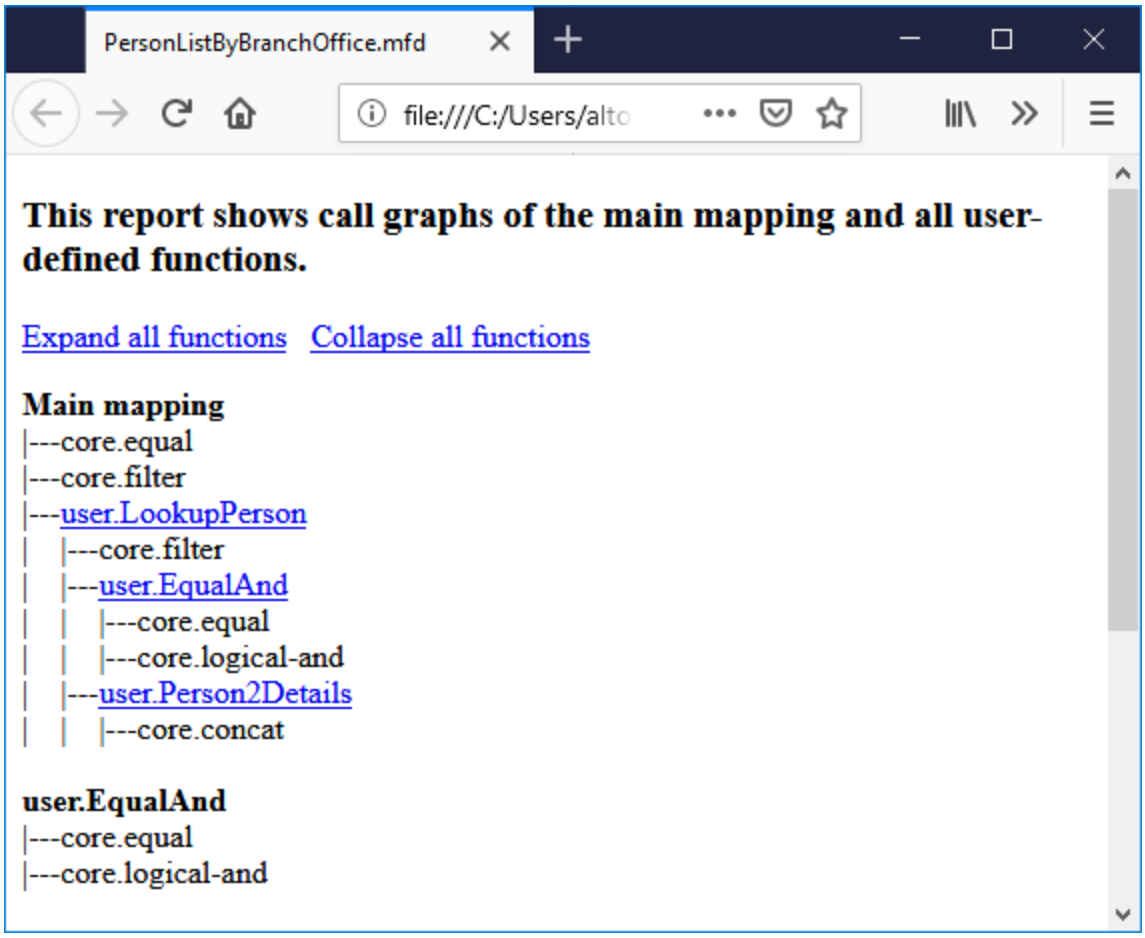
- **FunctionCallGraph.sps**: muestra el gráfico de llamadas de la asignación principal y las funciones definidas por el usuario que haya.
- **FunctionsUsedBy.sps**: muestra qué funciones usa, directa o indirectamente, la asignación.
- **ImpactAnalysis.sps**: enumera todos los nodos de origen y de destino, 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 **...\MapForce2025\Documentation\MapForce**.

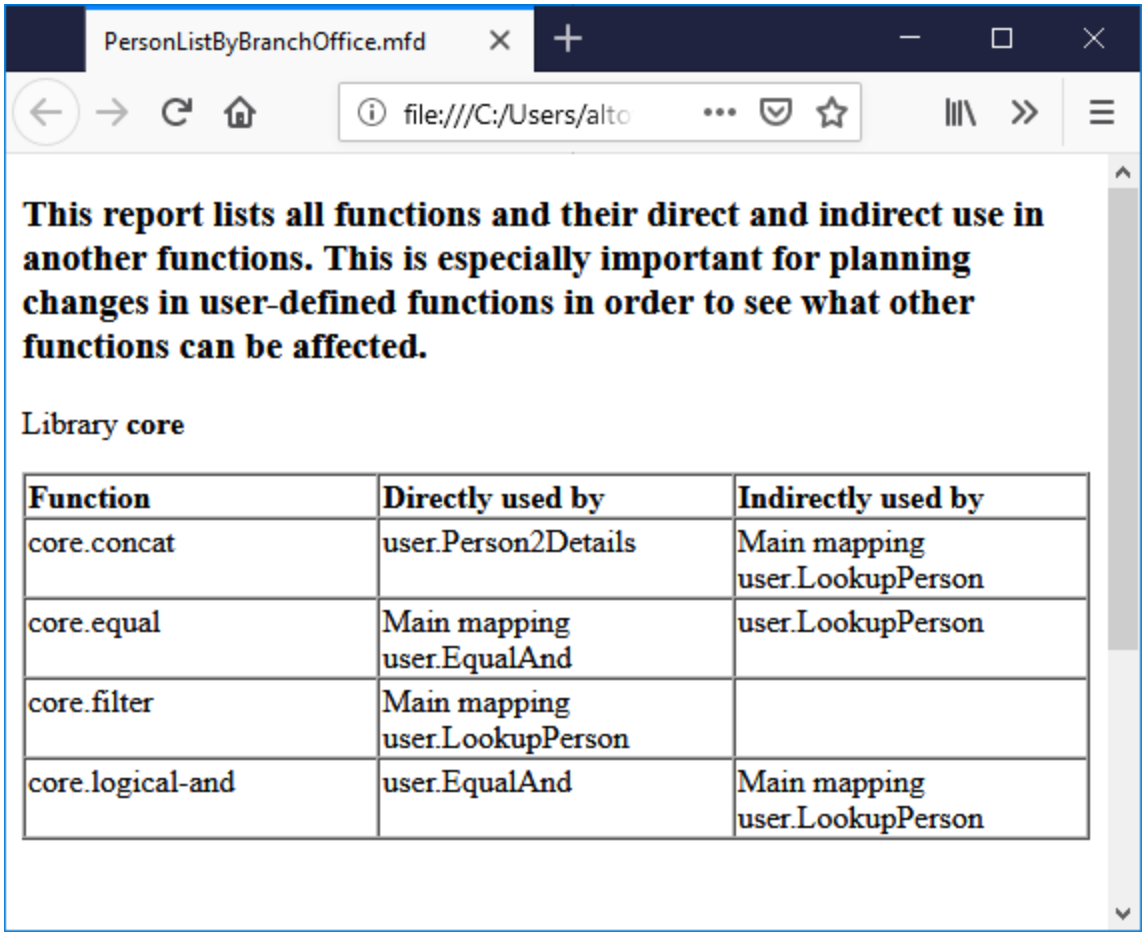



Los ejemplos siguientes ilustran los resultados producidos por cada una de estas hojas de estilos. Los ejemplos se generaron a partir de una de las asignaciones de ejemplo que vienen con MapForce: **PersonListByBranchOffice.mfd**. Aunque estos ejemplos muestran resultados HTML, el diseño de otros formatos es parecido. Para más información sobre cómo crear y personalizar archivos SPS consulte Diseño personalizado 806.

## Hoja de estilo "FunctionCallGraph.sps"



## Hoja de estilo "FunctionsUsedBy.sps"

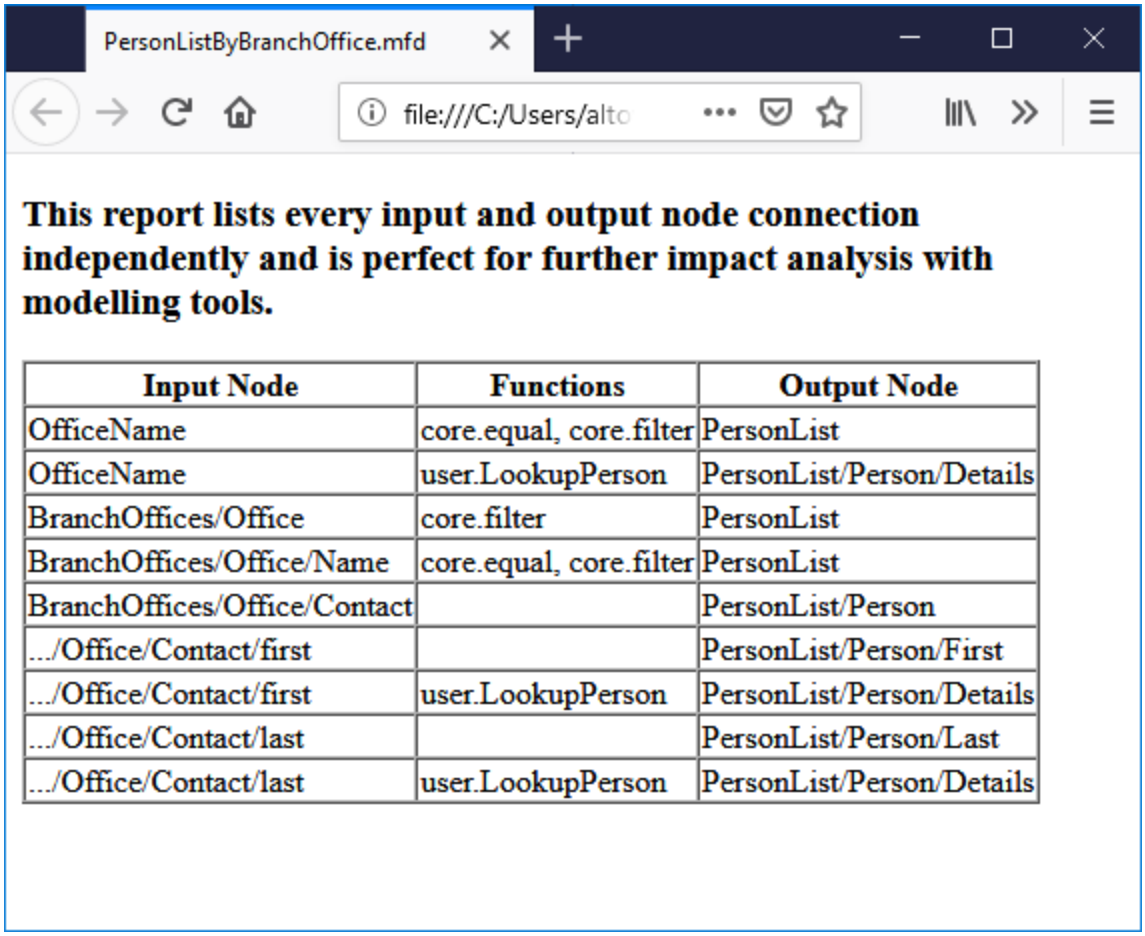



**This report lists all functions and their direct and indirect use in another functions. This is especially important for planning changes in user-defined functions in order to see what other functions can be affected.**

Library **core**

| Function | Directly used by | Indirectly used by |
|---|---|---|
| core.concat | user.Person2Details | Main mapping<br>user.LookupPerson |
| core.equal | Main mapping<br>user.EqualAnd | user.LookupPerson |
| core.filter | Main mapping<br>user.LookupPerson | |
| core.logical-and | user.EqualAnd | Main mapping<br>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 |


© 2019-2025 Altova GmbH

## Hoja de estilo "OverallDocumentation.sps"

PersonListByBranchOffice.mfd

file:///C:/Users/altova

Mapping **PersonListByBranchOffice.mfd**

Input **OfficeName**

| Nodes | Connections | |
|---|---|---|
| OfficeName<br>Type: string<br>Default:<br>core.constant("Nanonull, Inc.") | core.equal => a \| result =><br>core.filter => bool \| on-true => | PersonList<br>Type: restriction of xs:anyType [0..1]<br>Annotation: List of Persons |
| | user.LookupPerson =><br>Office_Name \| result => | PersonList/Person/Details<br>Type: xs:string [0..1] |

Input **BranchOffices** (BranchOffices.xsd)

| Nodes | Connections |
|---|---|
| File: BranchOffices.xml<br>Type: string | |
| BranchOffices<br>Type: restriction of xs:anyType [0..1] | |
| BranchOffices/Name<br>Type: restriction of xs:string | |

## 7.2 Hojas de estilos personalizadas

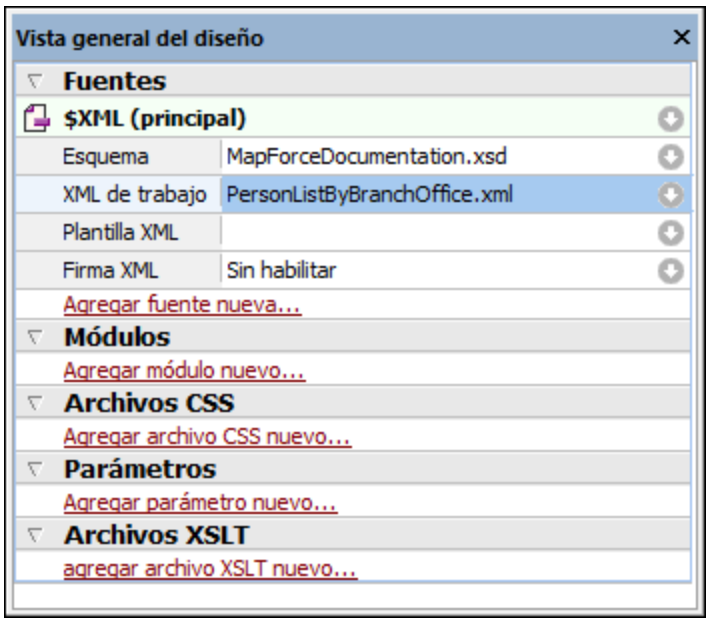
Además de usar el diseño fijo integrado, también puede crear hojas de estilos personalizadas a partir de la documentación generada a partir de la asignación con StyleVision. También puede cambiar cualquiera de las hojas de estilos predefinidas 801, por ejemplo ajustando el tipo de fuente u otros estilos.

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:

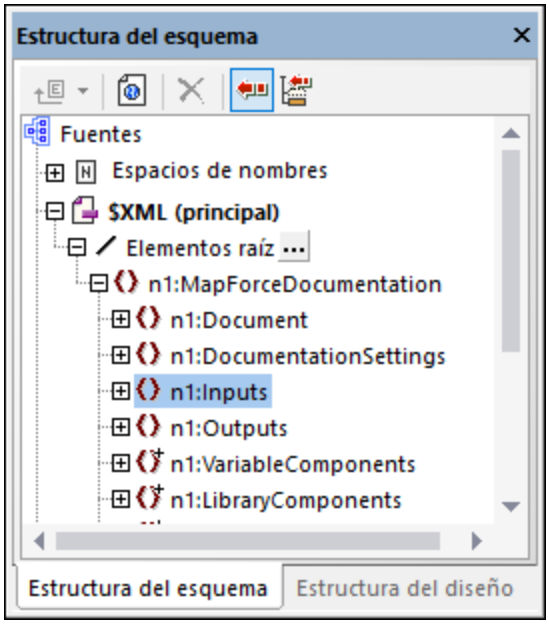
1. El esquema XML que determina la estructura de la documentación generada de MapForce. Este esquema se llama **MapForceDocumentation.xsd** y está incluido en los archivos que se instalan con MapForce. Lo encontrará en la carpeta **... \Documents\Altova\MapForce2025\Documentation\MapForce**. Tenga en 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\MapForce2025\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.

 © 2019-2025 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

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.

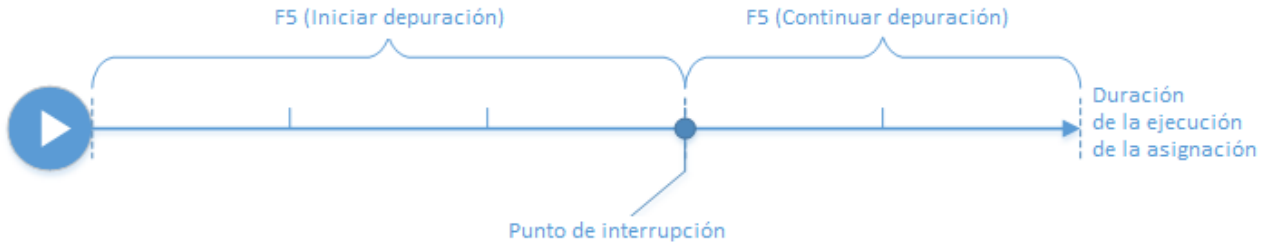
El depurador de asignaciones solamente está disponible si el lenguaje de transformación de la asignación es BUILT-IN. Si comienza a depurar una asignación configurada para otro tipo de lenguaje, MapForce solicitará que cambie el lenguaje por BUILT-IN. También puede convertir una asignación al lenguaje BUILT-IN seleccionando el comando de menú **Resultados | Motor de ejecución integrado**. En ambos casos la conversión a BUILT-IN sólo 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).

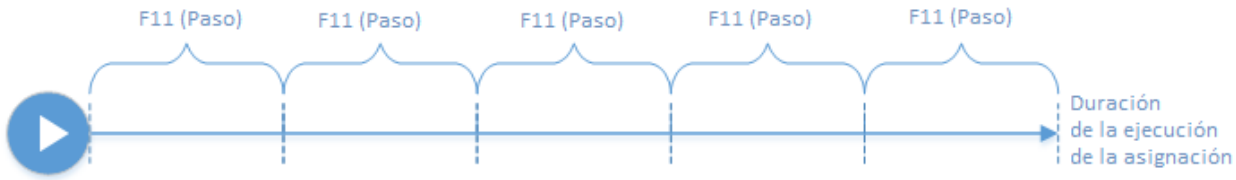



 © 2019-2025 Altova GmbH

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.

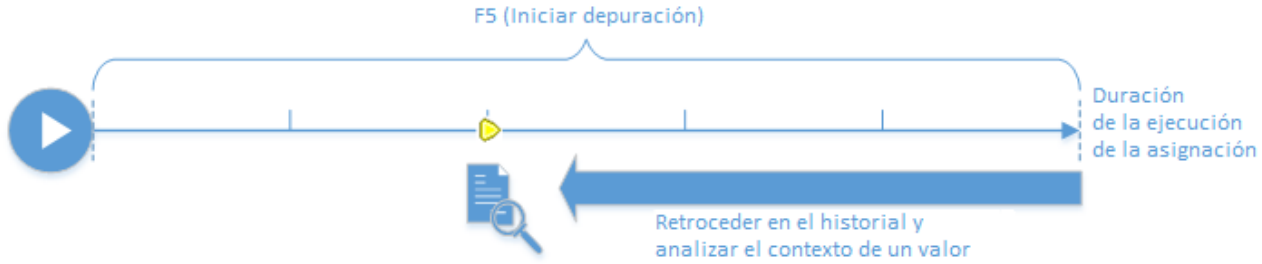
## Depuración por pasos

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.



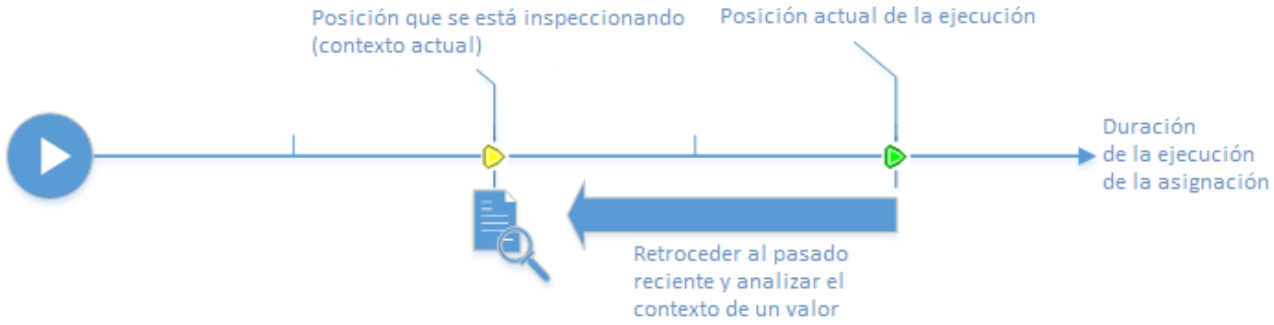

## Analizar el registro de valores

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).



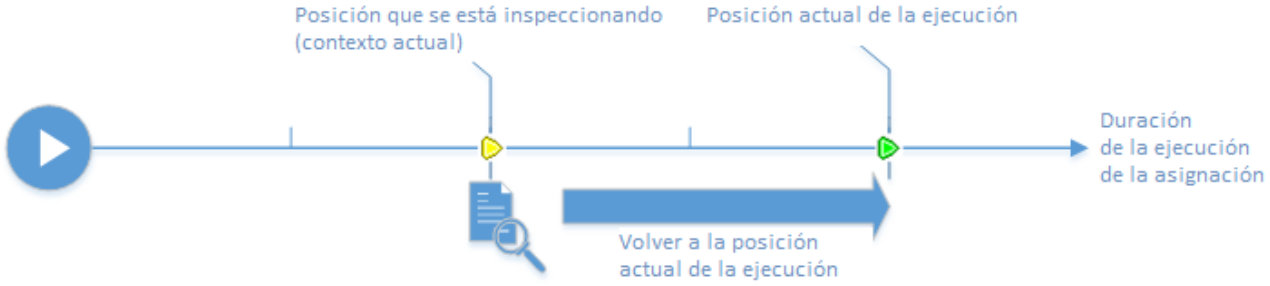

## Establecer el contexto en un valor relacionado con la posición actual de la ejecución

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.

 © 2019-2025 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* 1130).

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 encadenada 119, 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* 815).

## 8.2 Información sobre el modo depurador

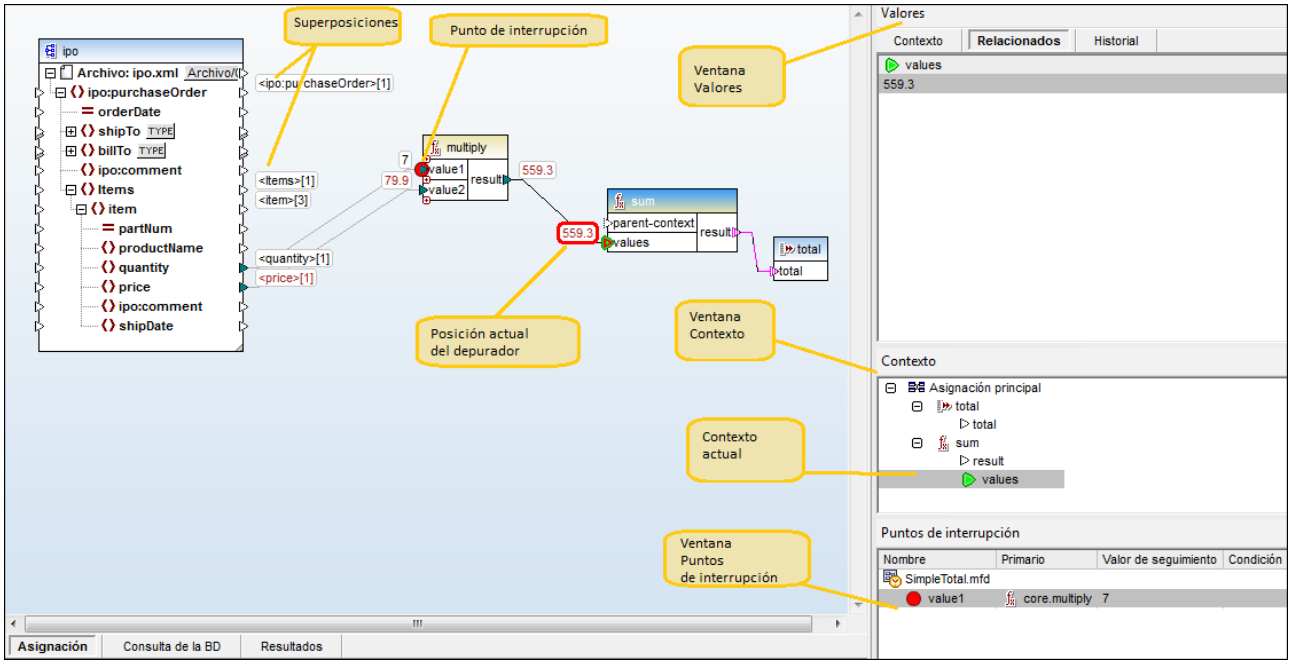
Cuando se inicia la depuración (pulsando **F5**, **F11** o **Ctrl + F11**), MapForce ejecuta la asignación en modo depurador.

> Mientras MapForce esté en modo depurador, la asignación es de sólo 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 sólo 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\MapForce2025\MapForceExamples**.




*Entorno de desarrollo MapForce en modo depurador*

A continuación describimos las características de la interfaz del depurador de MapForce.


© 2019-2025 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 815 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[820]).

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 (*consulte Ver el valor actual de un conector*[825]). |
| | 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 (*consulte Establecer un contexto en un valor*[829]). |

## 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[818] 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[820] 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[822] para obtener más información.


© 2019-2025 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.

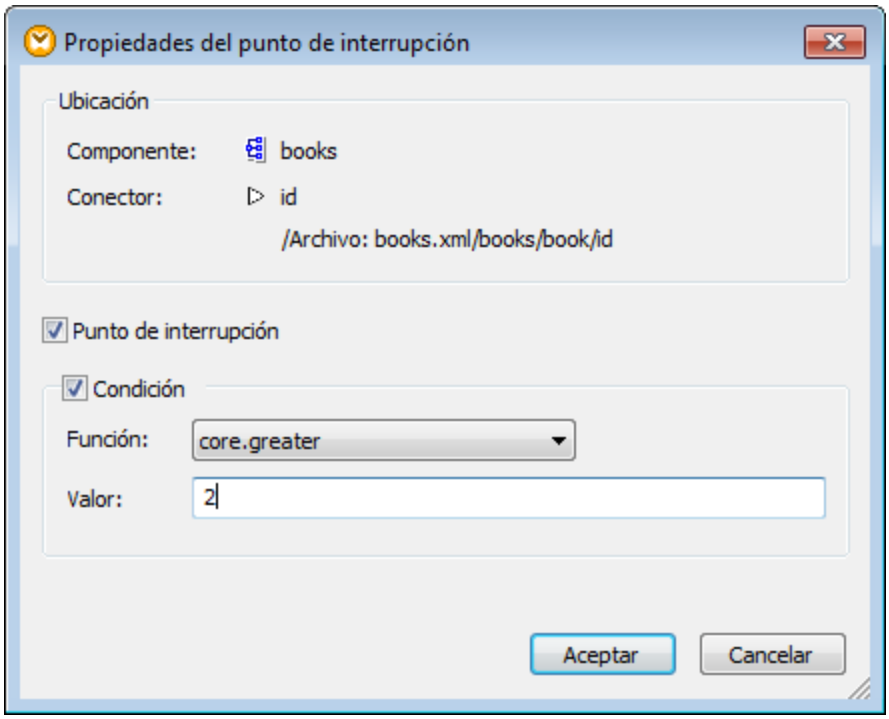
Los puntos de interrupción pueden ser simples o condicionales. Los simples detienen la ejecución de la asignación sin condiciones. Los condicionales detienen la ejecución de la asignación cuando se cumple una condición asignada previamente al punto de interrupción. Las condiciones pueden ser funciones 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

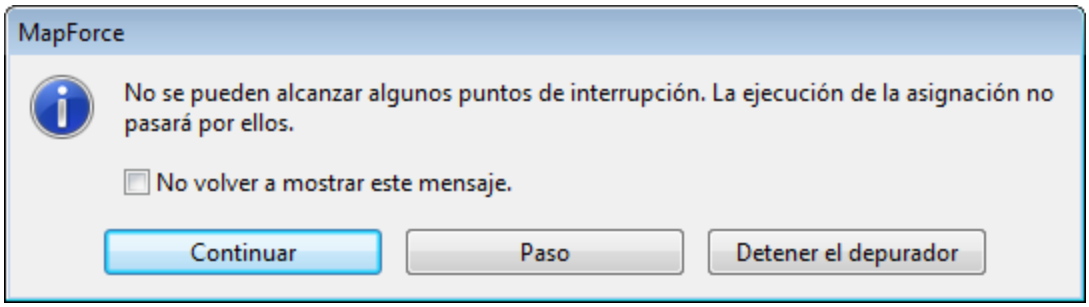
Para quitar un punto de interrupción haga clic con el botón derecho en el conector que lo contiene y seleccione **Punto de interrupción del depurador**. Otra opción es hacer clic en el conector de entrada o salida donde está definido el punto de interrupción y pulsar **F9**.

También puede eliminar puntos de interrupción desde la ventana Puntos de interrupción 822.

## Puntos de interrupción inalcanzables

En algunos casos MapForce emite este mensaje:

 © 2019-2025 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 810).

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:

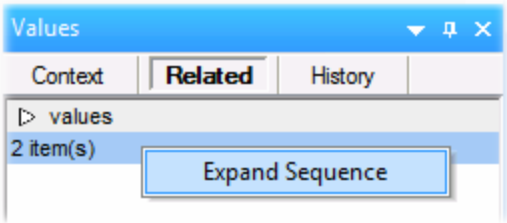
| | |
|---|---|
| | El triángulo verde es *el presente*. Indica la posición actual de la ejecución (*consulte* *Ver el valor actual de un conector* 825). |
| | El triángulo amarillo es *el pasado*. Indica que se trata de un conector del pasado, relativo a la posición actual de la ejecución. Esto puede ocurrir después de definirse un contexto a mano (*consulte* *Establecer un contexto en un valor* 829). |

## 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* 826).

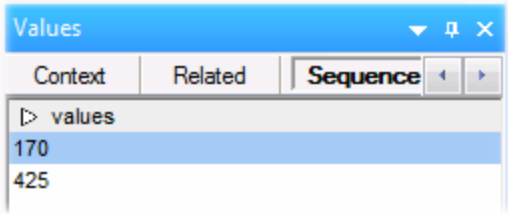
## Pestaña *Secuencia*

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 (p.ej. 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*.


© 2019-2025 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* 827).

## 8.5 Ventana Contexto

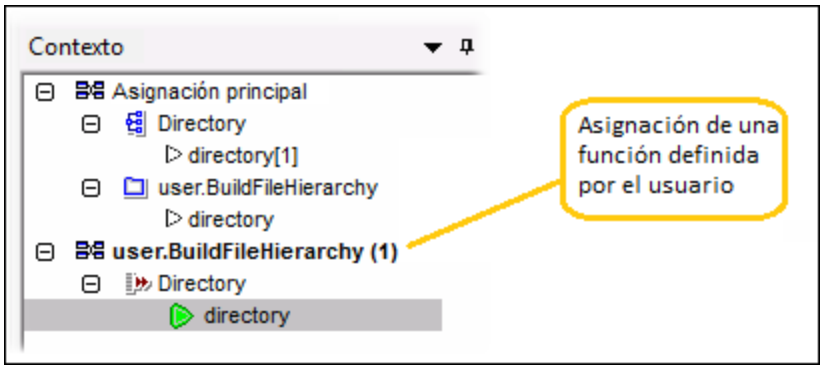
Cuando MapForce está en modo depurador, la ventana Contexto muestra una estructura de conectores que corresponden a la posición actual del depurador. En otras palabras, ofrece el contexto de asignación responsable de producir el valor actual de la asignación.

MapForce genera el contexto actual de la siguiente manera:

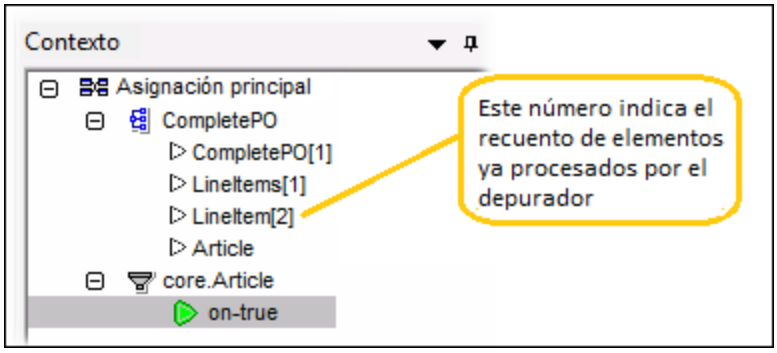
1. Empezar por el nodo raíz de la estructura de destino.
2. Descender hasta el nodo de destino actual.
3. Desde el nodo de destino actual, avanzar hacia la izquierda por los componentes que conduzcan hasta la posición actual. Estos componentes pueden ser filtros o componentes de ordenación, funciones 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. |

 © 2019-2025 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.<br><br>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* 829). |

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:

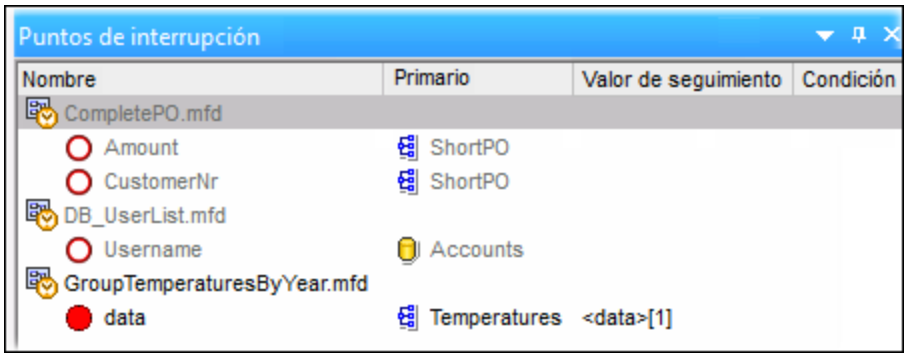
1. Desde el destino del conector de salida de la función definida por el usuario hasta el conector de entrada de la función definida por el usuario.
2. Desde esa posición hacia la izquierda.

**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:** 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:

 © 2019-2025 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**.

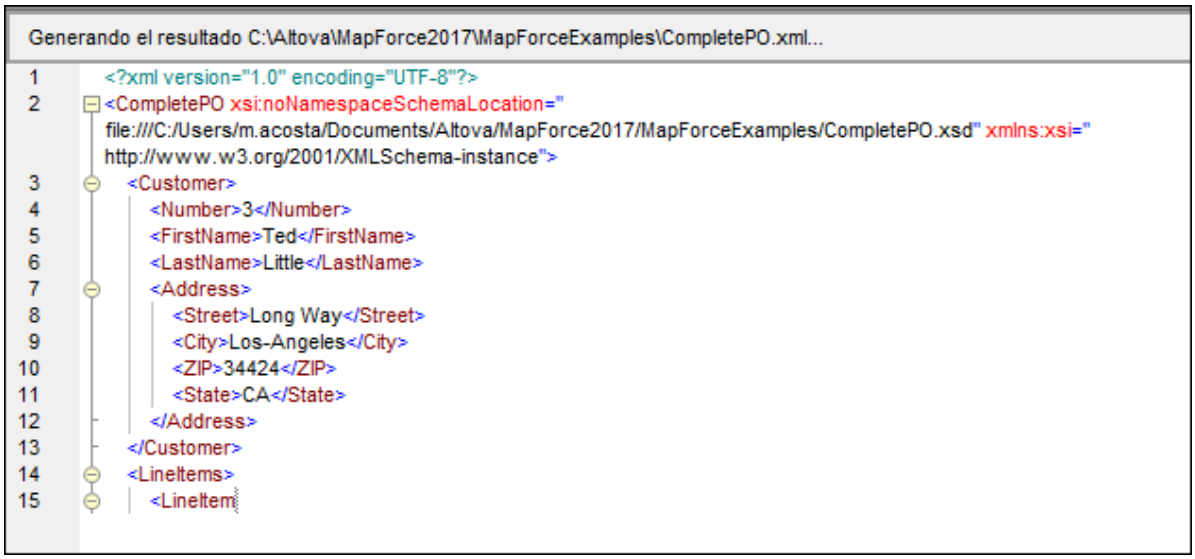
El comando del menú contextual **Eliminar todos los puntos de interrupción** quita todos los puntos de interrupción de la ventana, independientemente de la asignación de datos a la que pertenezcan.

Consulte también el apartado Agregar y quitar puntos de interrupción 815.

# 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 admite 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

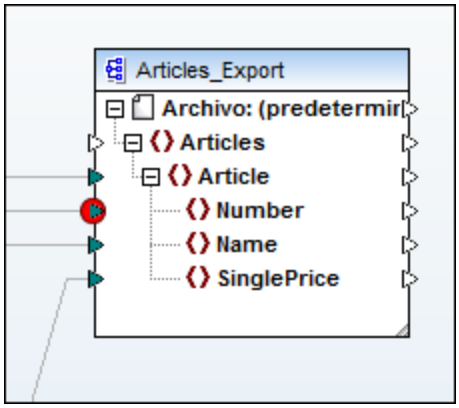
- El nodo de destino que se está calculando en ese momento no siempre aparece en el panel *Resultados*. Por ejemplo, los atributos XML se recopilan de forma interna y se escriben de una sola vez.
- Si el resultado produce varios archivos de salida, solamente se puede obtener una vista previa del archivo que se escribe en ese momento dado. Es decir, está deshabilitada la opción de cambiar a otro archivo de salida.


© 2019-2025 Altova GmbH

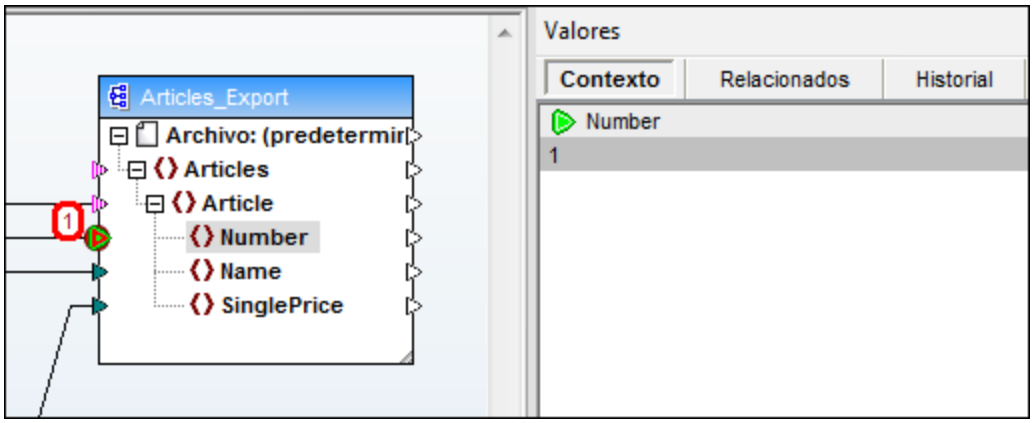
## 8.8 Ver el valor actual de un conector

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 820.

Para comprender mejor este funcionamiento del depurador abra la asignación **PreserveFormatting.mfd** del directorio **<Documentos>\Altova\MapForce2025\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.



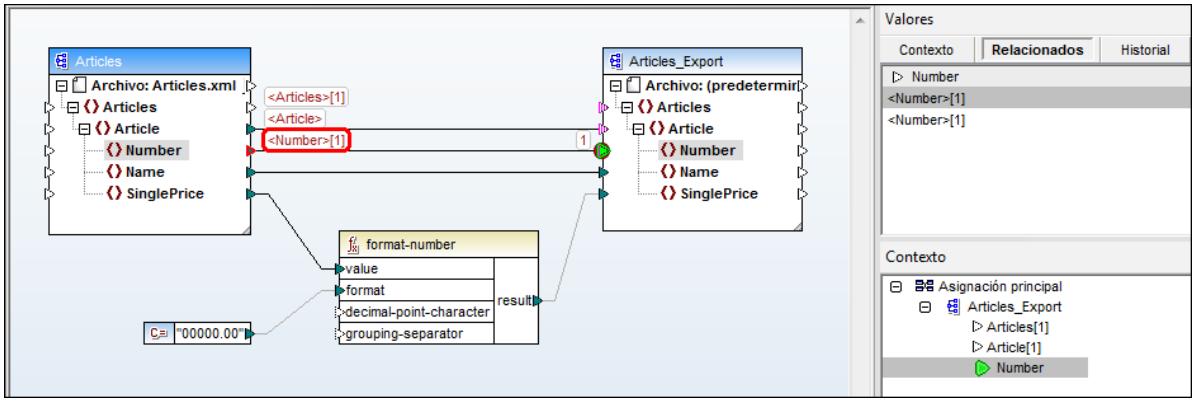

Tal y como se aprecia en la imagen anterior, la posición actual del depurador (y el punto de interrupción ) está en el nodo `Number` del componente de destino. La ventana Valores indica que este nodo procesa el valor "1" (este valor también se resalta con un borde grueso rojo en la asignación).

# 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\MapForce2025\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* 829).

 © 2019-2025 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* 1130), 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 sólo es relevante si se trata de conectores que procesaron valores desde el comienzo de la ejecución hasta la posición actual de la depuración.

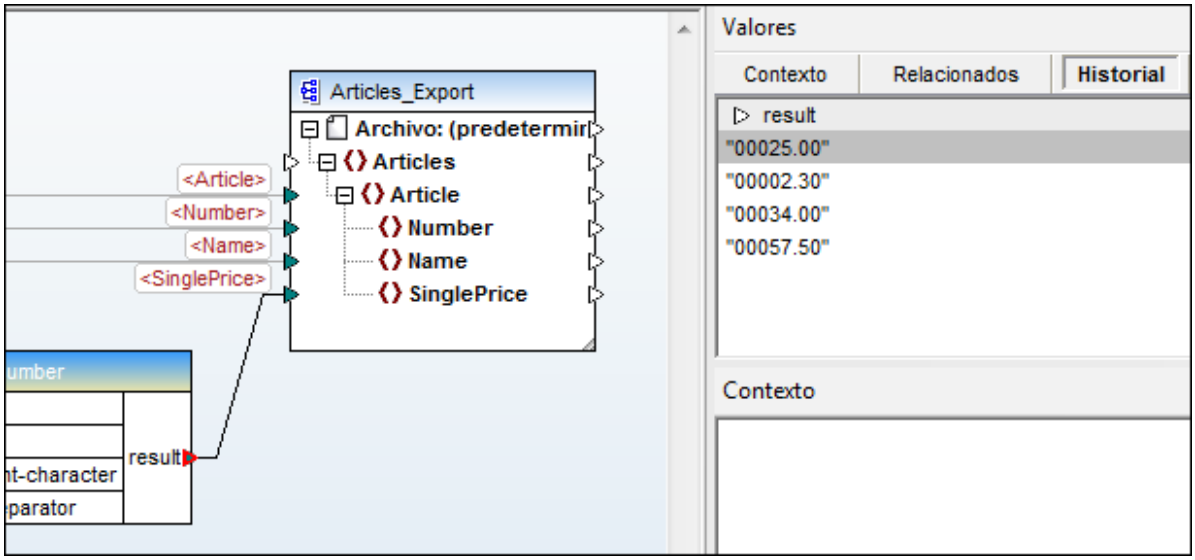
Para comprender mejor este funcionamiento del depurador vamos a depurar de principio a fin y sin puntos de interrupción una asignación y después veremos el historial de valores que fueron procesados por un conector determinado.

Para empezar abra la asignación **PreserveFormatting.mfd** del directorio **<Documentos>\Altova\MapForce2025\MapForceExamples**. Si ya tiene abierta esta asignación, asegúrese de:

- Eliminar todos los puntos de interrupción previamente definidos (*véase* *Agregar y quitar puntos de interrupción* 815).
- 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.

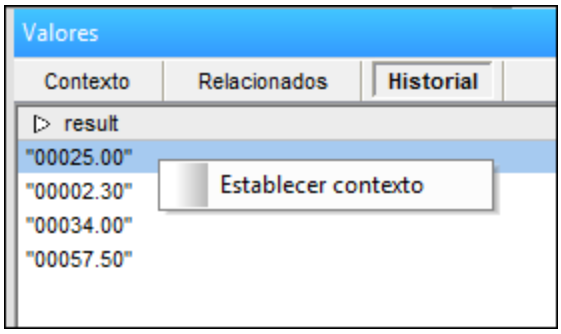



Como se aprecia en la imagen anterior, este nodo concreto (`result`) procesó un total de cuatro valores. Si necesita conocer más información sobre alguno de ellos, recuerde que puede recrear el contexto que lo produjo (*véase* *Establecer el contexto en un valor* 829).

© 2019-2025 Altova GmbH

## 8.11 Establecer el contexto en un valor

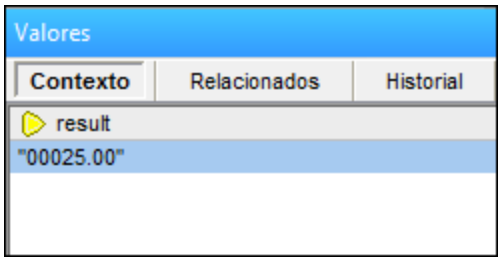
Establecer el contexto en un valor es una acción comparable con retroceder un paso hacia el pasado con el objetivo de obtener más información sobre el contexto de asignación que produjo dicho valor. Puede establecer el contexto en cualquier valor de los que aparecen en la ventana Valores (en cualquiera de sus pestañas). Si habilitó la opción *Guardar historial de seguimiento completo* (*véase* *Configurar el depurador* 1130), 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 812 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 820.




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 asignaciones con 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* 831).
- *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* 832).

** 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 851 para obtener más información.


© 2019-2025 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:

- 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 832.

**Nota:** También puede obtener una vista previa del código XSLT 68 y del código XQuery con el motor integrado.

## 9.2 Automatización con MapForce Server

MapForce Server 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 19). 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* 839.

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* 842.

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 sólo son compatibles con Windows (*para información sobre otros sistemas operativos consulte Preparar asignaciones para ejecutarlas en servidores* 833).

Para más información sobre MapForce Server consulte la documentación de este producto (https://www.altova.com/es/documentation).

 © 2019-2025 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* 66).

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 839.
- 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:

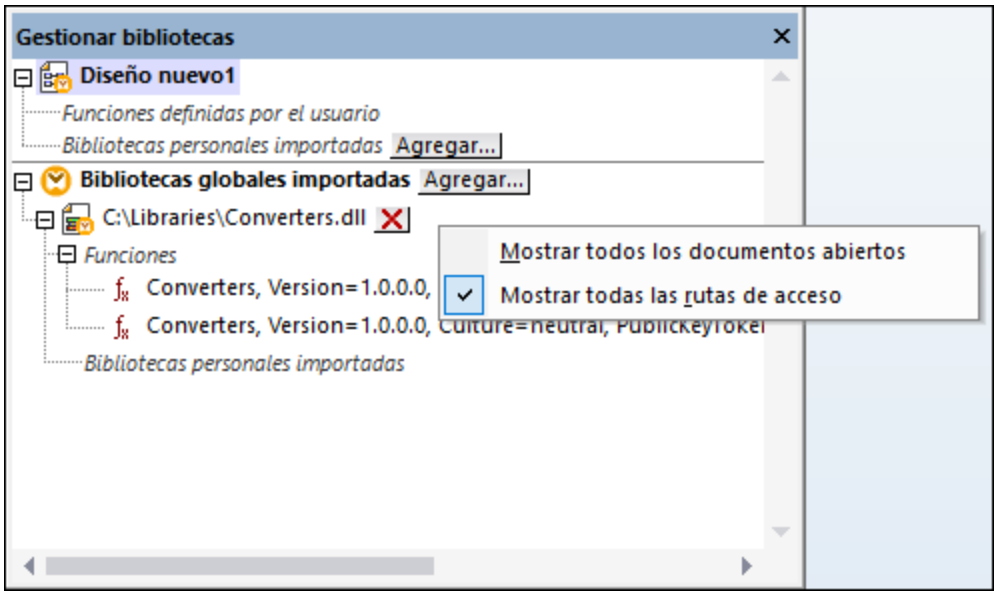
- En el equipo de origen, necesitará la edición MapForce Enterprise o Professional para diseñar la asignación e implementarla en el equipo de destino, como se explica en el apartado Implementar asignaciones en FlowForce Server 874.
- Tanto MapForce Server como FlowForce Server deben tener asignada una 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[874].

- 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[784]). 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*[1128]).
- 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*).

 © 2019-2025 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* 44).

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* 46). 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* 197)
- 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* 251)

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* 175). 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` |


© 2019-2025 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* 78).
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* 44). 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* 44).

## 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 874 y Recursos globales en FlowForce Server 874..

 © 2019-2025 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 832). 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 ejecutar la asignación invocando el método `run` de la API de MapForce Server. Consulte la documentación de MapForce Server (https://www.altova.com/es/documentation) 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. Para más información consulte el apartado Implementar asignaciones en FlowForce Server 842.

## Requisitos

Consulte el apartado anterior (Preparar asignaciones para ejecutarlas en servidores 833).

### Para compilar una asignación en un archivo de ejecución de MapForce Server (.mfx):

1. Abra una asignación de datos en MapForce (por ejemplo, **myMapping.mfd**).
2. En el menú **Archivo**, haga clic en **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. Por último, haga clic en **Guardar**. El archivo de ejecución de MapForce Server **myMapping.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\MapForce2025\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\MapForce2025\MapForce.exe" "C:
\Usuarios\altova\Documentos\Altova\MapForce2025\MapForceExamples\SimpleTotal.mfd" /COMPIL
E "C:\Usuarios\altova\Desktop"
```

Consulte la documentación de la interfaz de la línea de comandos de MapForce 851 para obtener más información.

## Qué incluye el archivo .mfx

El archivo .mfx incluye esta información:

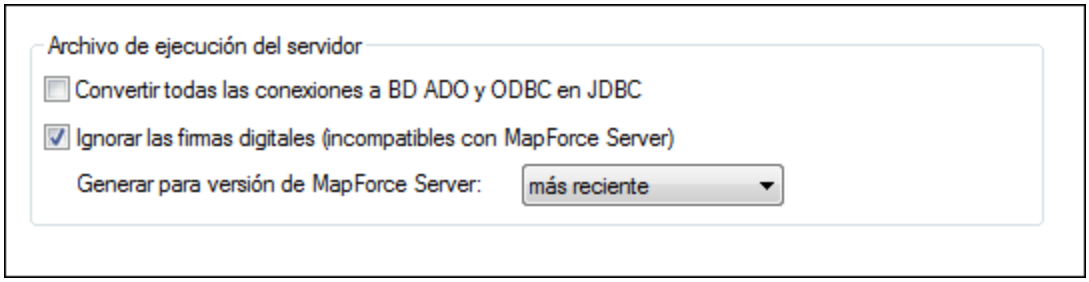
- El algoritmo de asignación, incluidas todas las funciones definidas por el usuario importadas de otras asignaciones.
- Nombres de archivos de entrada y salida a los que se hace referencia en componentes de la asignación. Las rutas de acceso son absolutas o relativas dependiendo de la configuración de la asignación, véase Rutas de acceso según el entorno de ejecución 46.
- Si la asignación contiene componentes XML, se incluye la información sobre el esquema XML, que es necesaria para poder ejecutar la asignación, y se codifica en el algoritmo de asignación.
- Si la asignación incluye conexiones a la base de datos, se describen también los detalles de esta conexión. Las contraseñas están cifradas.

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. Para más información consulte el apartado Preparar asignaciones para ejecución en servidor 833.

## 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.




Cuando instale una versión más reciente de MapForce Server, recuerde que deberá cambiar esta opción según corresponda. Si no quiere compilar para una versión específica de MapForce Server, también puede seleccionar la versión más reciente en la lista desplegable (es la opción predeterminada). En este caso 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\MapForce2025\MapForce.exe" /COMPILE /MFXVERSION:2025
```

 © 2019-2025 Altova GmbH

Consulte la documentación de la interfaz de la línea de comandos de MapForce 851 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 251).<br><br>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. Consulte la documentación de la interfaz de la línea de comandos de MapForce 851 para obtener más información.

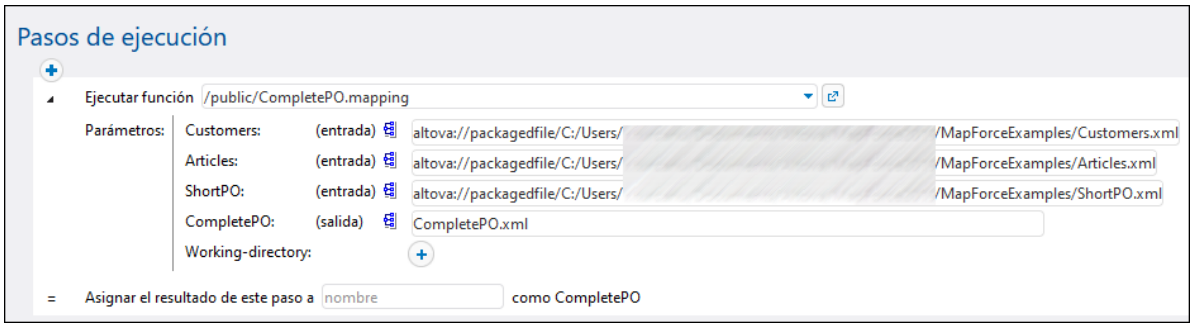
# 9.5 Implementar asignaciones en FlowForce Server

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. La implementación de asignaciones de MapForce en FlowForce Server permite automatizar su ejecución a través de trabajos. Tras implementar la asignación, podrá crear un trabajo de FlowForce Server basado en la asignación implementada y sacar partido a todas las características de los trabajos de FlowForce (p. ej. definir condiciones desencadenantes, exponer el trabajo 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 propiamente dicha. Tras la implementación, la asignación de datos estará disponible en la interfaz de administración de FlowForce Server en forma de función de asignación (.mapping) y en la ruta de acceso que se especificara. Los componentes de origen se convierten en argumentos de entrada y los componentes de destino se convierten en argumentos de salida de dicha función.




- Todos los archivos de instancia de entrada (XML, CSV, Text) que se usan en la asignación.

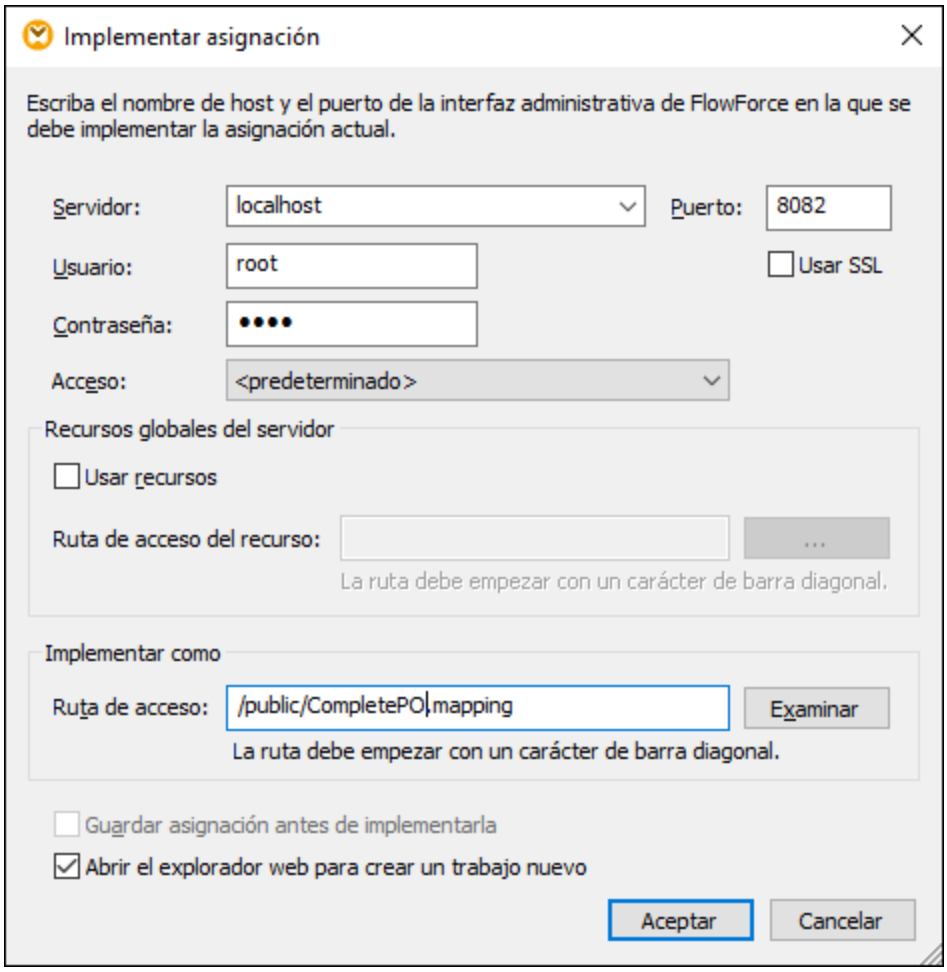
## Requisitos

Consulte Preparar asignaciones para ejecutarlas en servidores 833.

## Implementar la asignación en FlowForce Server

1. Ejecute MapForce y asegúrese de que el lenguaje de transformación está configurado en el lenguaje integrado BUILT-IN.
2. En el menú **Archivo** haga clic en **Implementar en FlowForce Server**. Aparece el cuadro de diálogo **Implementar asignación** (*imagen siguiente*).

 © 2019-2025 Altova GmbH




3. Seleccione las opciones de implementación (ver más abajo) y haga clic en Aceptar. Si marca la casilla *Abrir el explorador web para crear un trabajo nuevo* la interfaz de administración de FlowForce Server se abre en el explorador, donde podrá empezar a crear el trabajo de FlowForce.

En la siguiente 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 (o dirección IP) y el puerto del servidor de FlowForce Server. Puede usar **localhost** y **8082** si FlowForce Server se está ejecutando en el mismo equipo en el puerto predeterminado. Si no sabe qué dirección y puerto usar, inicie sesión en la interfaz de administración web de FlowForce Server y consulte qué dirección IP y puerto aparece en la barra de dirección del explorador web.<br><br>Si tiene problemas de conectividad, compruebe que el equipo en el que se ejecuta FlowForce Server está configurado para admitir conexiones entrantes en el puerto y en la dirección designados. |

| Opción | Descripción |
|---|---|
| | Para implementar la asignación a través de una conexión con cifrado SSL debe marcar la casilla **Usar SSL**. Al hacerlo, la aplicación entiende que FlowForce ya está 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 del recurso | Marque la casilla **Usar recursos** si quiere que la función de asignación use Recursos 856 una vez se haya implementado en el servidor. Si marca esta casilla también debe introducir en la caja de texto **Ruta de acceso del recurso** la ruta de acceso del recurso correspondiente en el servidor. Para navegar hasta el recurso haga clic en el botón de **puntos suspensivos**.<br><br>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 875.<br><br>Si no marca la casilla **Usar recursos** entonces el recurso que se resuelva dependerá de la configuración seleccionada. En el servidor la función de asignación ya no necesita recursos globales, pero sí usará el valor resuelto. |
| 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.<br><br>En el cuadro de diálogo también puede crear y eliminar contenedores y asignaciones de datos (siempre y cuando disponga de los permisos correspondientes). |
| Guardar asignación antes de | Esta opción está habilitada si la asignación que se intenta implementar |

 © 2019-2025 Altova GmbH

| Opción | Descripción |
|---|---|
| implementarla | está sin guardar. Marque esta casilla para guardarla antes de implementarla. |
| Anexar archivos MFD para recuperarlos después | Esta opción le permite implementar el archivo MFD junto con sus archivos de entrada correspondientes (p.ej. archivo(s) XML de origen). Sin embargo, no se permite adjuntar archivos que definen la estructura (p.ej. esquemas XSD). Al abrir la asignación implementada en FlowForce Server, la sección *Archivos implementados* mostrará todos los archivos que se pueden descargar. |
| Abrir el explorador web para crear un trabajo nuevo | Si marca esta casilla, la interfaz de administración web de FlowForce Server se abrirá en el explorador una vez terminada la implementación y podrá empezar a crear un trabajo de FlowForce Server inmediatamente. |

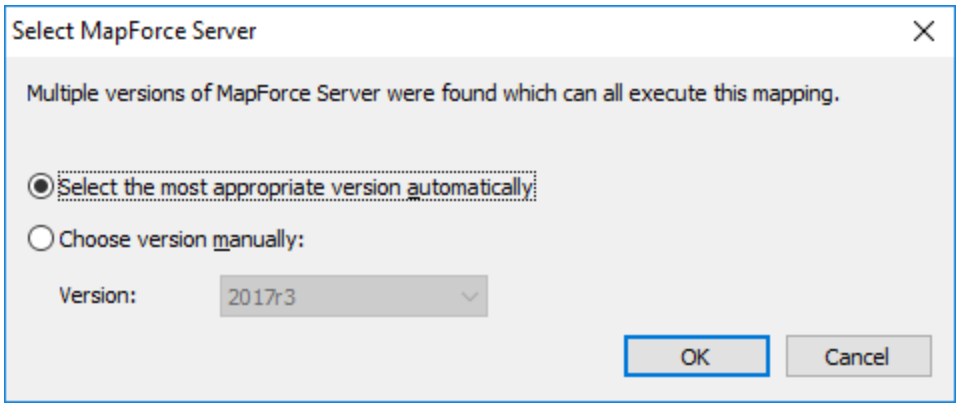
## Solución de errores

En la siguiente 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:<br><br>`No se puede realizar la operación de entrada/salida en el archivo...`<br>`Error I/O 28: No se puede establecer la conexión con el puerto de <server> 8082. Tiempo de espera agotado`<br>`Error de 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.<br><br>El servicio *FlowForce Server* también debe estar en ejecución para que se pueda realizar la implementación. |
| Al implementar la asignación se genera este error:<br><br>`No se puede realizar la operación de entrada/salida en el archivo...`<br>`Error I/O 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 (sólo Windows)

Si el servidor donde implementa la asignación ejecuta varias versiones de MapForce Server bajo la dirección de FlowForce Server (sólo si se trata de servidores Windows), entonces la aplicación solicitará que especifique la versión de MapForce Server en la que tiene pensado ejecutar la asignación:

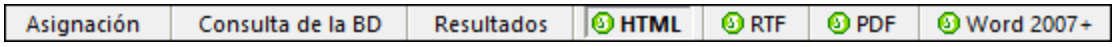



**Nota:** El cuadro de diálogo aparece si el directorio de instalación de FlowForce Server contiene archivos .tool para cada una de las versiones de MapForce Server que se está ejecutando bajo el control de FlowForce Server. Cuando se instala MapForce Server junto con FlowForce Server, se añade un archivo .tool a este directorio automáticamente. La ruta de acceso donde se almacenan los archivos .tool es: **C:\Archivos de programa\Altova\FlowForceServer2025\tools**. Si tiene otras versiones de MapForce Server y desea ejecutarlas bajo la dirección de FlowForce Server, deberá copiar a mano sus archivos .tool en el directorio mencionado. El archivo .tool de MapForce Server está en este directorio: **C:\Archivos de programa\Altova\MapForceServer2025\etc**.

 © 2019-2025 Altova GmbH

# 9.6 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*).




**Importante**

- 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 Asignación encadenada 119.

Para generar una vista previa de los datos de la asignación en los paneles de resultados de StyleVision necesita:

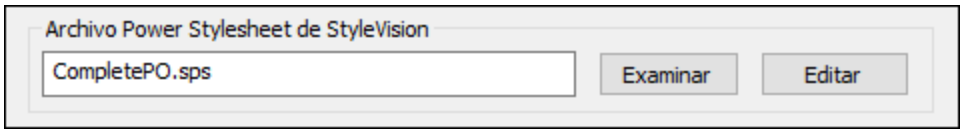
- Tener instalado Altova StyleVision en su equipo, bien como instalación independiente o como parte de Altova MissionKit.
- El componente de destino debe tener asociado un archivo SPS (hoja de estilos StyleVision Power). El archivo de la hoja de estilos se puede crear o editar con StyleVision. Esta hoja 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 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 de acceso al archivo hoja de estilos StyleVision Power puede ser absoluta o relativa. Para más detalles consulte Rutas de acceso relativas y absolutas 44.

*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:

 © 2019-2025 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** |

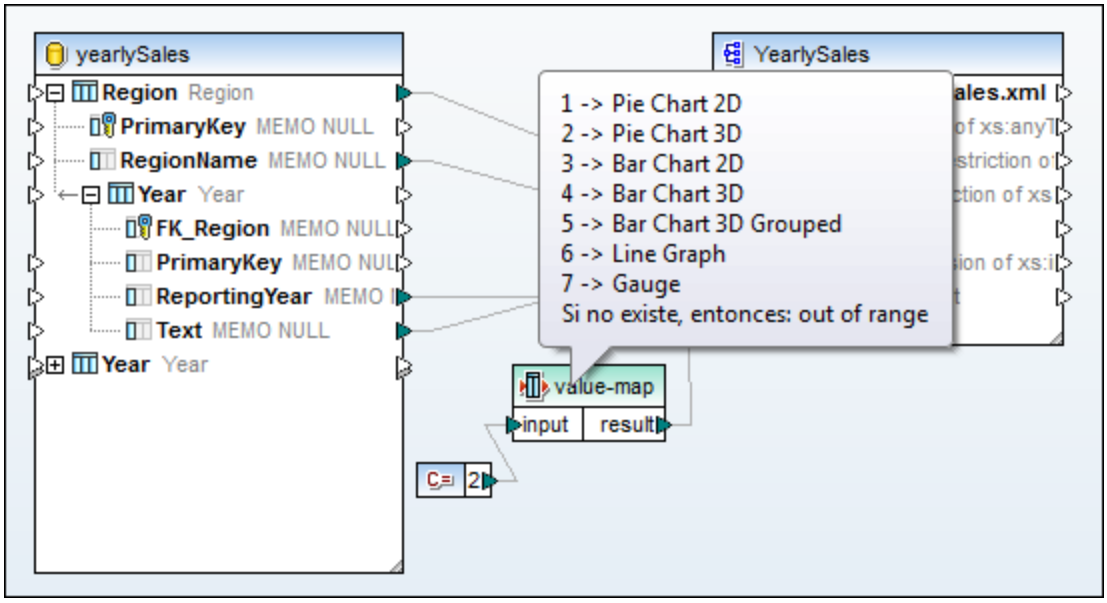
**Other Comments:** _____

_____ _____

Authorized Signature Date

Mapping | XSLT2 | DB Query | Output | HTML | RTF | PDF | Word 2007+

Otro ejemplo es **`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.


© 2019-2025 Altova GmbH

# 9.7 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 <nombreArchivo> [/{destino} [[<dirSalida>] [/opciones]]]
```

Siga leyendo para obtener más información sobre cada parámetro.

*Sintaxis de la línea de comandos*
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 |

`<nombreArchivo>`

El archivo del diseño de la asignación (**`.mfd`**) o del proyecto de la asignación (**`.mfp`**) (*ediciones Professional y Enterprise*) a partir del cual se genera el código. Para generar código para todo el proyecto, indique el destino `/GENERATE` (*véase /{target} a continuación*) e introduzca la ruta del proyecto en el campo `<nombreArchivo>` (p.ej., **`MapForceExamples.mfp`**).

`/{destino}`

Indica el lenguaje o entorno de destino para el que se genera el código. Son compatibles los siguientes destinos de generación de código:

- `/XSLT`
- `/XSLT2`
- `/XSLT3`
- El comando `/COMPILE[:compileoptions]` Compila una asignación en un archivo de ejecución de MapForce Server (**`.mfx`**). También puede elegir las siguientes opciones separadas por una coma:
  - La opción `JDBC` transforma todas las conexiones DB en JDBC usando el driver JDBC y la URL definidos en el cuadro de diálogo Configuración de componentes de BD 251.
  - La opción `NOXMLSIGNATURES` elimina la generación de firmas digitales en el archivo de ejecución de MapForce Server. Tenga en cuenta que no se admiten las firmas digitales en MapForce Server.
- El comando `/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[101]). Si selecciona este destino, asegúrese de que indica un proyecto de MapForce (**.mfp**) en el campo `<filename>`.

- `/XQuery`
- `Java`
- El comando `/CS` genera código C#. Este comando permite configurar las siguientes opciones de generación de código:

  ```
  /CS[:{VS2013|VS2015|VS2017|VS2019|VS2022|DOTNETCORE31|DOTNET50|DOTNET60}]
  ```

  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.

- El comando `/CPP` genera código C++. Este comando permite configurar las siguientes opciones de generación de código:

  ```
  /CPP[:{VS2013|VS2015|VS2017|VS2019|VS2022|DOTNETCORE31|DOTNET50|DOTNET60},
  {MSXML|XERCES3},{LIB|DLL},{MFC|NoMFC}]
  ```

  El primer grupo de opciones indica el entorno de desarollo (p.ej., `VS2022` significa Visual Studio 2022).

  El segundo grupo de opciones indica la biblioteca XML de destino para el código generado. Los siguientes valores son válidos:

  o `MSXML` (genera código para MSXML 6.0)
  o `XERCES3` (genera código para Xerces 3)

  El tercer grupo de opciones indica si deben generarse bibliotecas estáticas o dinámicas. Los valores válidos son los siguientes:

  o `LIB` (genera bibliotecas LIB estáticas)
  o `DLL` (genera bibliotecas DLL)

  El cuarto grupo de opciones indica si el código generado debe ser compatible o no con MFC. Los valores válidos son los siguientes:

  o `MFC` (habilita la compatibilidad con MFC)
  o `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.

`<dirSalida>`

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 `<dirSalida>` no está definido, el lenguaje de generación de código y la ruta de salida de cada asignación, los determinan las opciones definidas para cada carpeta del proyecto (véase Gestionar las carpetas de proyecto[101]).


© 2019-2025 Altova GmbH

Cuando el destino es `/GENERATE` y el parámetro `<dirSalida>` está definido,el valor `<dirSalida>` indicado en la línea de comandos tiene prioridad sobre el directorio de salida definido en el nivel raíz del proyecto. Sin embargo, no tiene prioridad sobre las opciones de generación de código definidas en cada carpeta del proyecto.

`/opciones`

Las opciones `/opciones` no se excluyen mutuamente. Se pueden indicar una o más de las siguientes opciones:

- La opción `/GLOBALRESOURCEFILE <nombreArchivo>` se puede aplicar si la asignación de datos usa los recursos globales para resolver un archivo de entrada o de salida, rutas a directorios o bases de datos. Para más información, consulte Recursos globales de Altova 856. La opción `/GLOBALRESOURCEFILE` indica la ruta al archivo XML de los recursos globales. Observe si está indicado `/GLOBALRESOURCEFILE` también ha de estarlo `/GLOBALRESOURCECONFIG`.

- La opción `/GLOBALRESOURCECONFIG <config>` especifica el nombre de la configuración de los recursos globales (*véase la opción anterior*). Si está indicado `/GLOBALRESOURCEFILE` también ha de estarlo `/GLOBALRESOURCECONFIG`.

- La opción `/LOG <nombrearchivoregistro>` genera un archivo de registro en la ruta indicada. La ruta `<nombrearchivoregistro>` puede ser absoluta. Si indica una ruta completa, el directorio debe existir para que se genere el archivo de registro. Si sólo indica el nombre del archivo, este se guardará en el directorio actual del símbolo del sistema de Windows.

- La opción `[/MFXVERSION[:<versión>]` se puede aplicar si el destino es `/COMPILE`. Con esta opción se compila un archivo de ejecución de MapForce Server (`.mfx`) para una versión específica de MapForce Server. Puede introducir cualquiera de las versiones de MapForce Server a partir de la versión 2013r2. Consulte también Compilar asignaciones en archivos de ejecución de MapForce Server 840.

- La opción `/LIBRARY <nombrebiblioteca> (...)` 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. Consulte también Gestionar bibliotecas de funciones 464.

*Notas*

- Las rutas relativas lo son al directorio de trabajo, que es el directorio de la aplicación que llama a MapForce en ese momento. Esto se aplica a la ruta del nombre de archivo **`.mfd`**, **`.mfp`**, directorio de salida, nombre de archivo de registro y nombre de archivo de recurso global.
- No use barras diagonales inversas y comillas de cierre en la línea de comandos (p.ej., **`"C:\Mi directorio\"`**). El analizador sintáctico interpreta estos dos caracteres como comillas dobles literales. Se recomienda evitar el uso de espacios y comillas. Si hay espacios en la línea de comando y necesita las comillas, use la doble barra inversa (p.ej., **`"C:\Mi Directorio\\"`**).

## Ejemplos

1) Para iniciar MapForce y abrir la asignación `<nombreArchivo>.mfd`, use este comando:

```
MapForce.exe <nombreArchivo>.mfd
```

2) Para generar código XSLT 2.0 y crear también un archivo de registro con el nombre `<nombrearchivoregistro>`, use este comando:

```
MapForce.exe <nombreArchivo>.mfd /XSLT2 <dirSalida> /LOG <nombrearchivoregistro>
```

3) Para generar código XSLT 2.0 teniendo en cuenta la configuración del recurso global `<nombreconfigrecglo>` del archivo de recurso global `<nombrearchivorecglo>`, use este comando:

```
Mapforce.exe <nombreArchivo>.mfd /XSLT2 <dirSalida> /GLOBALRESOURCEFILE
<nombrearchivorecglo> /GLOBALRESOURCECONFIG <nombreconfigrecglo>
```

*Ejemplos para las ediciones Professional y Enterprise*

1) Para generar una aplicación en código C# para Visual Studio 2022 y generar un archivo de registro, use este comando:

```
MapForce.exe <nombreArchivo>.mfd /CS:VS2022 <dirSalida> /LOG <nombrearchivoregistro>
```

2) Para generar una aplicación en C++ usando las opciones de generación de código definidas en **Herramientas | Opciones** y generar un archivo de registro, use este comando:

```
MapForce.exe <nombreArchivo>.mfd /CPP <dirSalida> /LOG <nombrearchivoregistro>
```

3) Para generar una aplicación en C++ para Visual Studio 2022, MSXML, con bibliotecas estáticas, compatible con MFC y sin archivo de registro, use este comando:

```
MapForce.exe <nombreArchivo>.mfd /CPP:VS2022,MSXML,LIB,MFC
```

4) Para generar una aplicación en C++ para Visual Studio, Xerces, con bibliotecas dinámicas, sin compatibilidad con MFC y con archivo de registro, use este comando:

```
MapForce.exe <nombreArchivo>.mfd /CPP:VS2022,XERCES,DLL,NoMFC <dirSalida> /LOG
<nombrearchivoregistro>
```

5) Para generar una aplicación en código Java y generar un archivo de registro, use este comando:

```
MapForce.exe <nombreArchivo>.mfd /JAVA <dirSalida> /LOG <nombrearchivoregistro>
```

6) 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 este comando:

```
MapForce.exe <nombreArchivo>.mfp /GENERATE /LOG <nombrearchivoregistro>
```

7) Para generar código Java para todas las asignaciones del archivo del proyecto, use este comando:

```
MapForce.exe <nombreArchivo>.mfp /JAVA /LOG <nombrearchivoregistro>
```

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.

8) Para indicar los archivos de entrada y de salida en la línea de comandos para una asignación en Java compilada con anterioridad, use este comando:

 © 2019-2025 Altova GmbH

```
java -jar <archivoasignación>.jar /InputFileName
<nombrearchivoentrada> /OutputFileName <nombrearchivosalida>
```

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 369).

9) Para compilar una asignación en un archivo de ejecución de MapForce Server para la versión de MapForce 2025 y eliminar las firmas XML, use este comando:

```
MapForce.exe <nombreArchivo>.mfd /COMPILE:NOXMLSIGNATURES
<dirSalida> /MFXVERSION:2025 /LOG <nombrearchivoregistro>
```

# 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 1 857
- Configurar recursos globales, parte 2 859
- Archivos XML y recursos globales 863
- Carpetas como recursos globales 865
- Bases de datos como recursos globales 867
- Resultados de transformación como recursos globales 869
- Recursos globales en entornos de ejecución 873


 © 2019-2025 Altova GmbH

# 10.1 Configurar recursos globales - parte 1

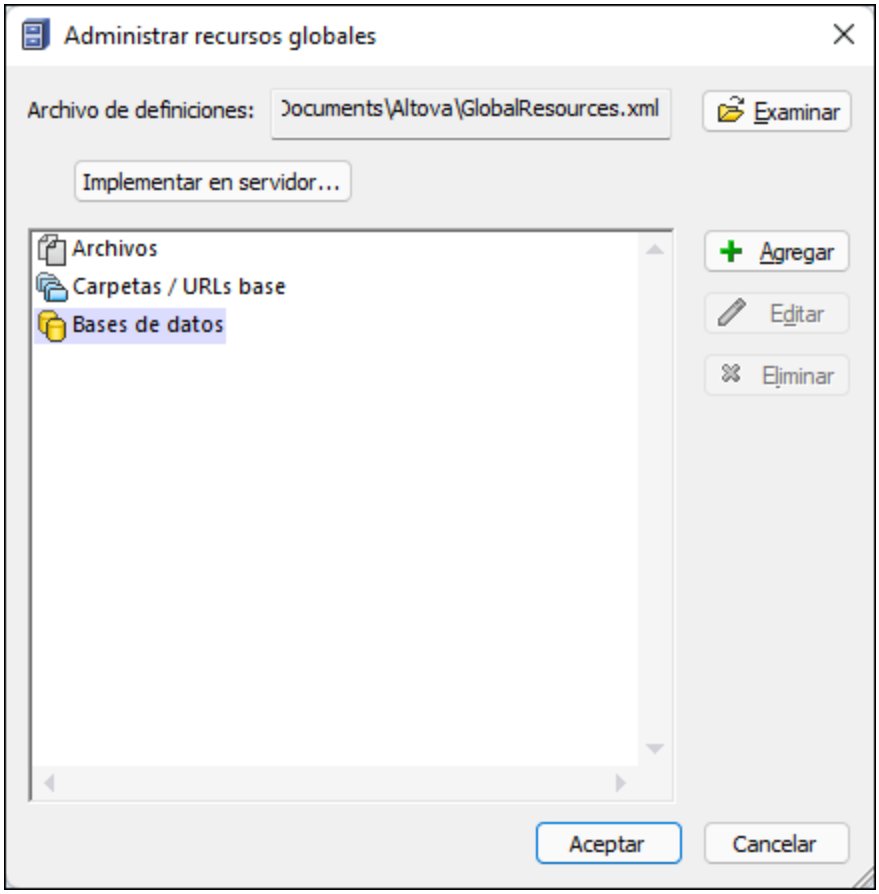
La configuración de los recursos globales tiene dos pasos: (I) crear el recurso global en el cuadro de diálogo **Administrar recursos globales** (*imagen siguiente*) y (ii) definir las propiedades de este recurso global en el cuadro de diálogo **Recursos globales**. La segunda parte se explica en el apartado siguiente 859.

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*).



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.


© 2019-2025 Altova GmbH

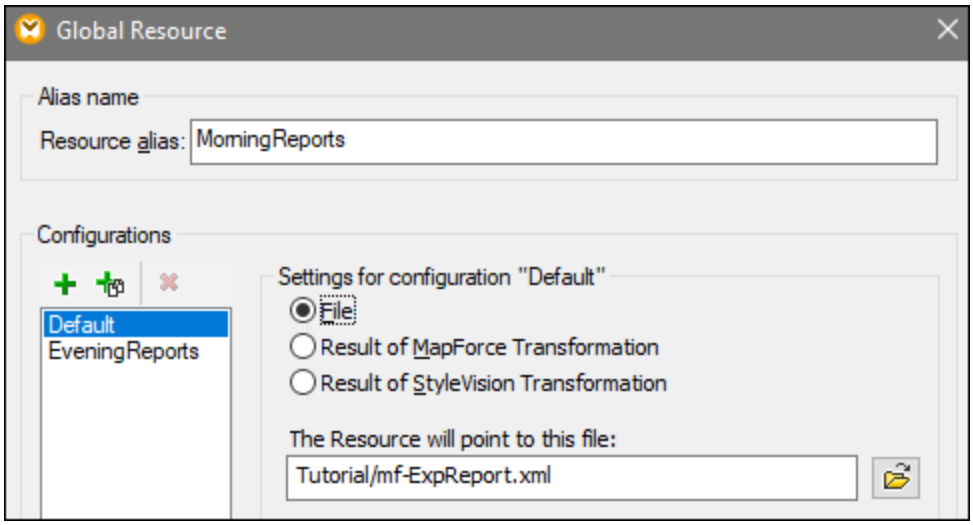
# 10.2 Configurar recursos globales - parte 2

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 857.

To find out more about setting up different types of global resources, see the following examples: XML Files as Global Resources 863, Folders as Global Resources 865, Databases as Global Resources 867.

## Files

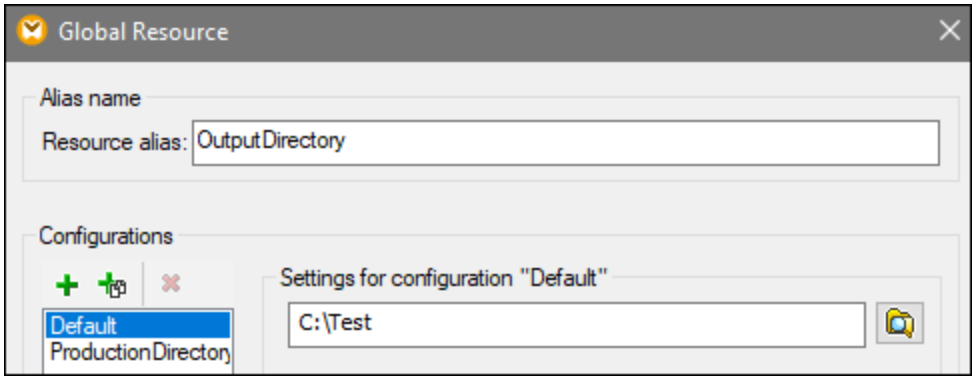
The file-specific properties are shown in the **Global Resource** dialog box below. The setup has three major parts: (i) the name of the file, (ii) the location of this file, and (iii) the list of configurations defined for this file alias.




The settings *Result of MapForce Transformation* and *Result of StyleVision Transformation* are discussed in Transformation Results as Global Resources 869.

## Folders

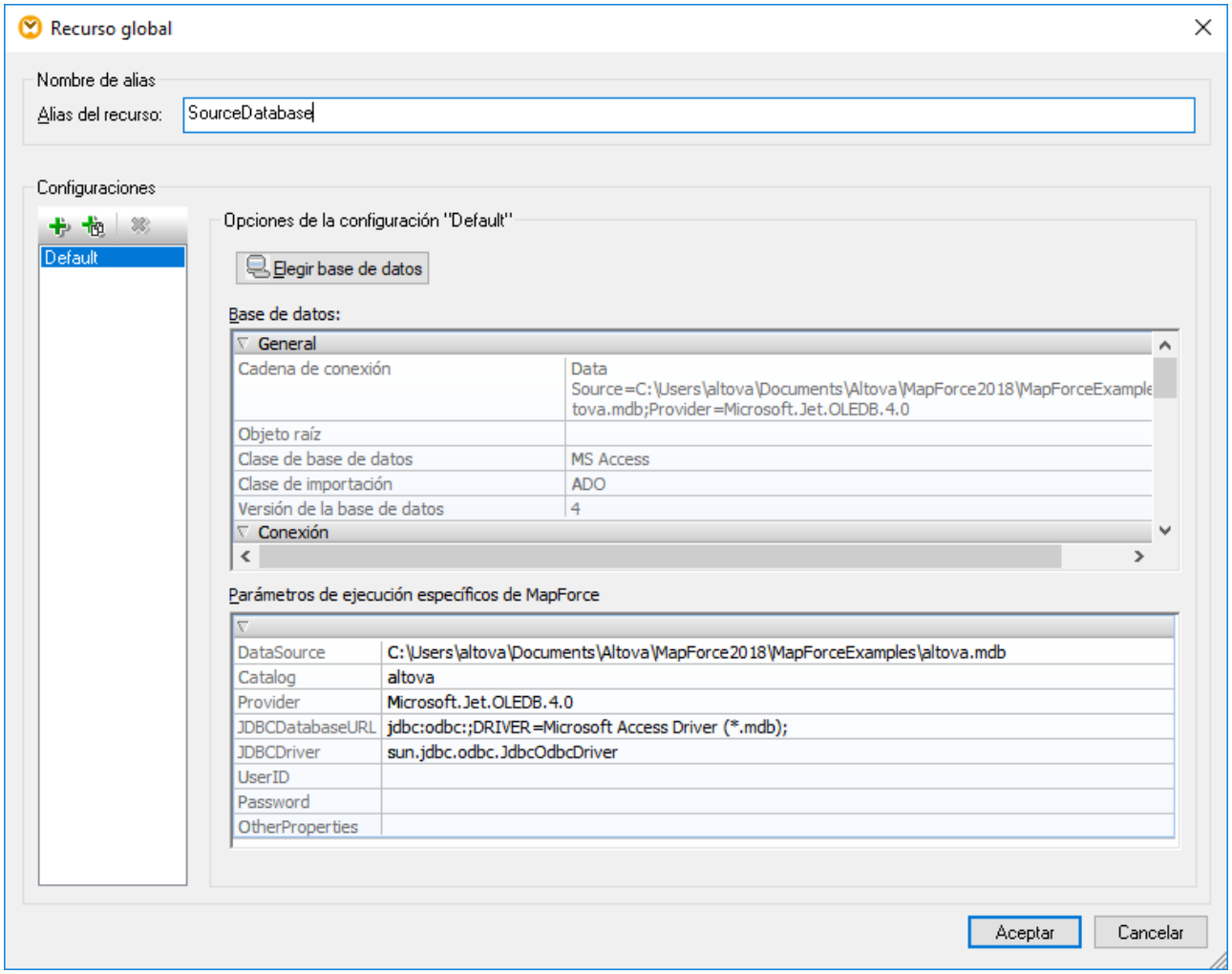
The folder-specific properties are shown in the **Global Resource** dialog box below. The setup has three major parts: (i) the name of the folder, (ii) the location of this folder, and (iii) the list of configurations defined for this folder alias.




## 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 179. Once you complete the wizard, the database connection parameters are displayed in the **Global Resource** dialog box (*see screenshot below*).


© 2019-2025 Altova GmbH




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 175.

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 178). 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.

 © 2019-2025 Altova GmbH

# 10.3 Archivos XML como recursos globales

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**.
- 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**.


*© 2019-2025 Altova GmbH*

# 10.4 Carpetas como recursos globales

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:

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**.


*© 2019-2025 Altova GmbH*

# 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.sqlite`**.
2. La configuración `ReleaseDatabase` apuntará a **`Tutorial\Altova.sqlite`**.

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 **SQLite** y navegue hasta **`MapForceExamples\Altova.sqlite`**.
4. Haga clic en y asigne un nombre a la segunda configuración.
5. Haga clic en **Elegir base de datos**, seleccione **SQLite** y navegue hasta **`Tutorial\Altova.sqlite`**.

## 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**.

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.


© 2019-2025 Altova GmbH

# 10.6 Resultados de transformación 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.

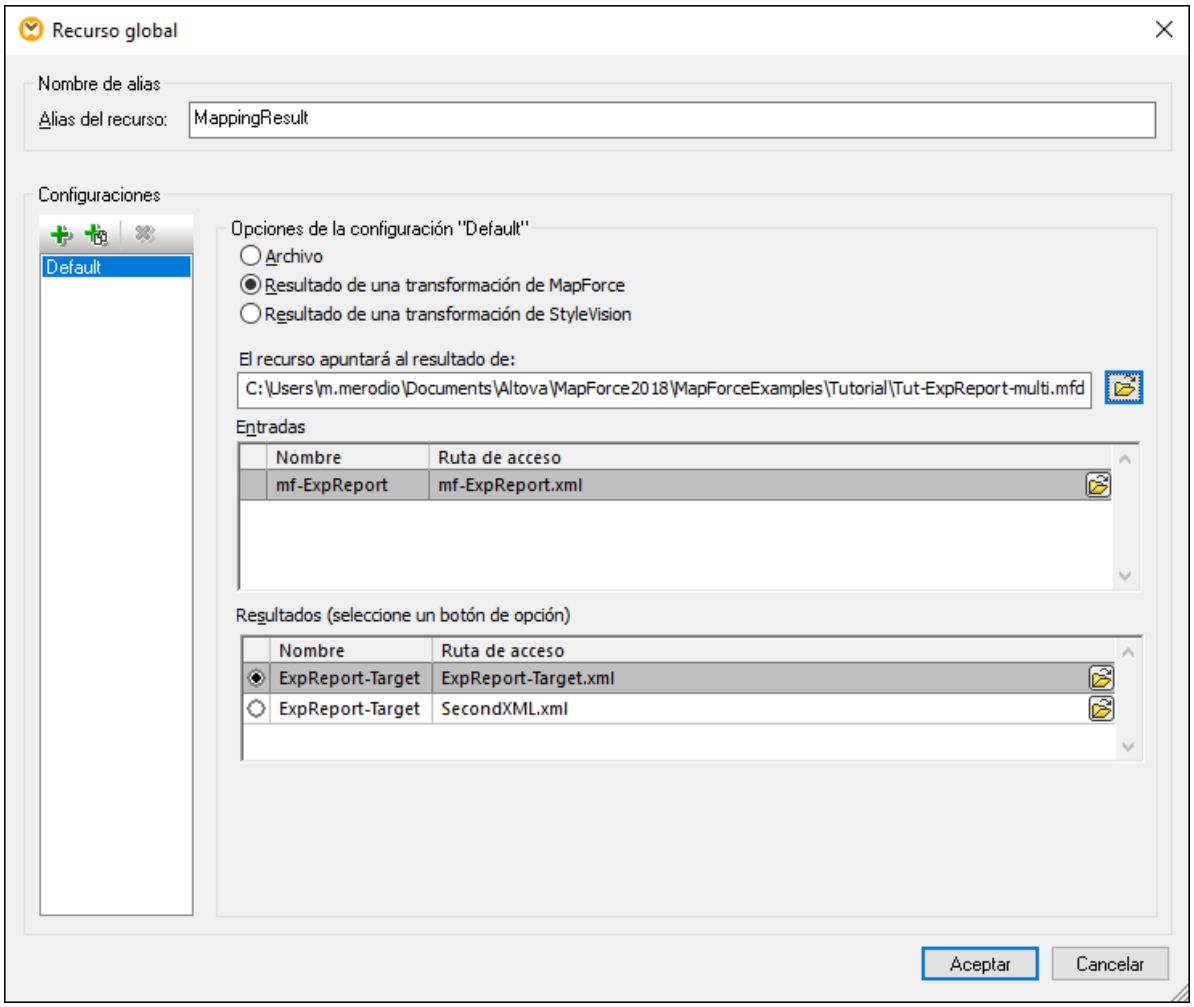
## Ejemplo: Resultados de transformaciones de MapForce

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

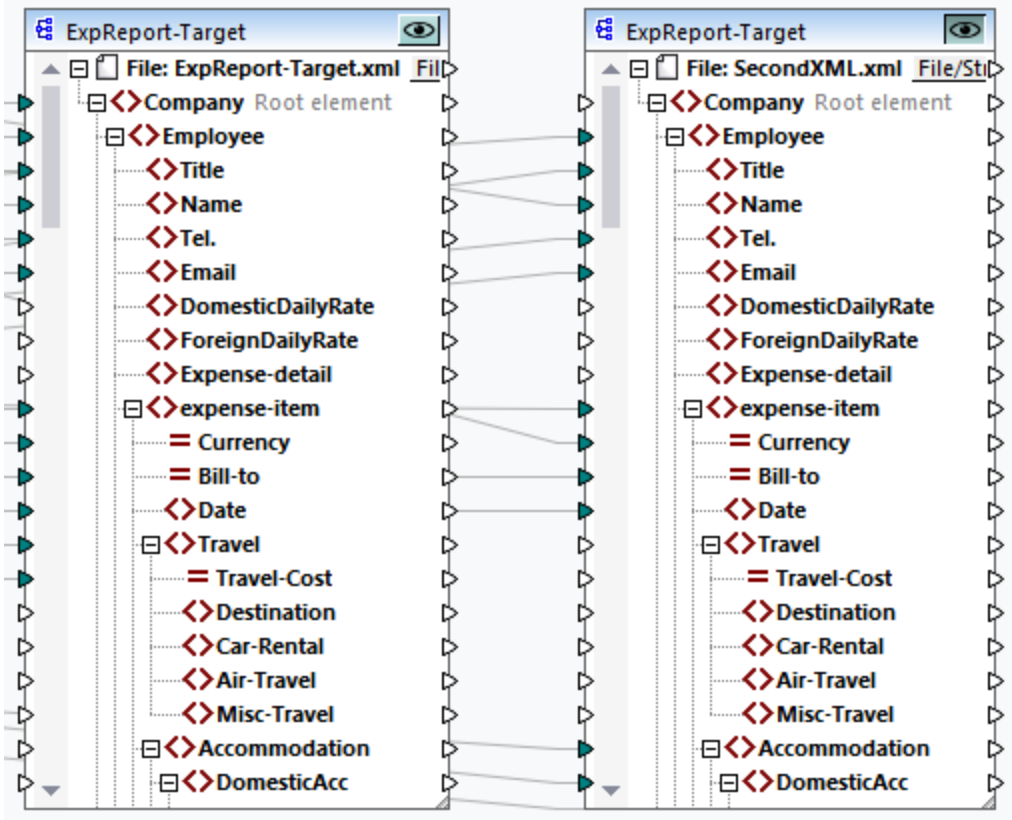
Para este paso puede usar MapForce o XMLSpy.

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.




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*).

 © 2019-2025 Altova GmbH




## 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

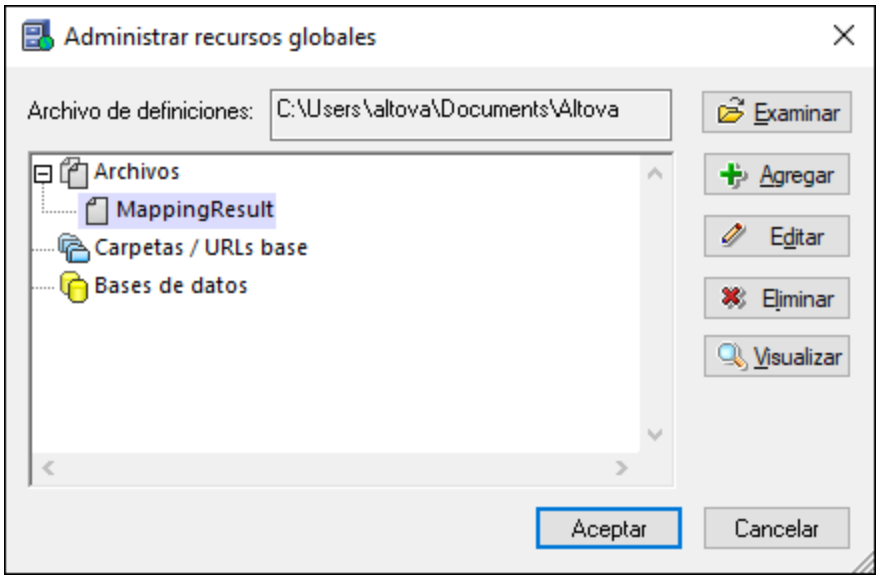
A continuación explicamos cómo usar el recurso global que creamos en el paso anterior.

*Configuración predeterminada*

Para usar la configuración predeterminada en XMLSpy siga las instrucciones siguientes.

1. Ejecute XMLSpy.

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`**.

 © 2019-2025 Altova GmbH

# 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 873
- Recursos globales en MapForce Server 874
- Recursos globales en FlowForce Server 874

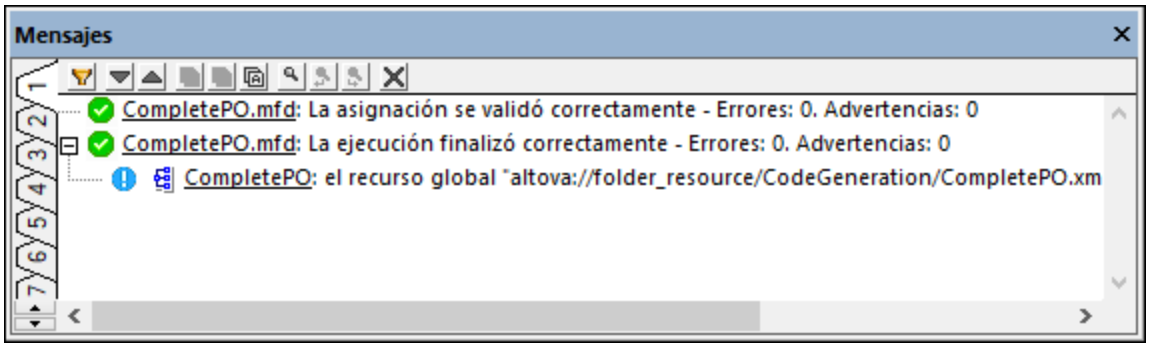

## 10.7.1 Recursos globales en el código generado

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 851 para obtener más información.

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 entrada o salida.

## 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 1 857.
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 2 859.

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\MapForceServer2025\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.

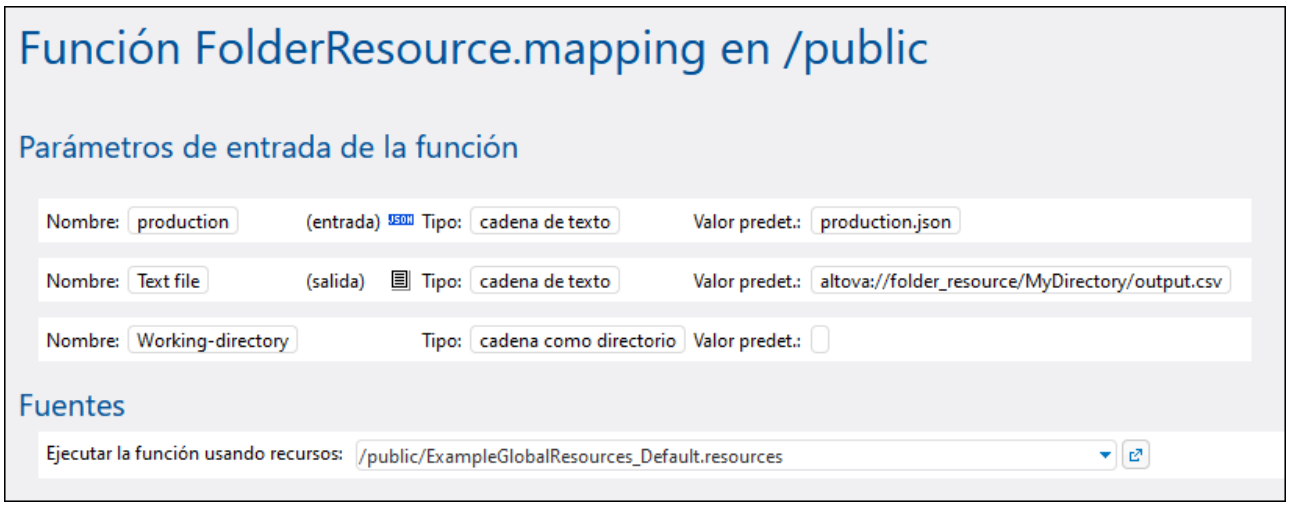
## 10.7.3 Recursos globales en FlowForce Server

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

 © 2019-2025 Altova GmbH

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.

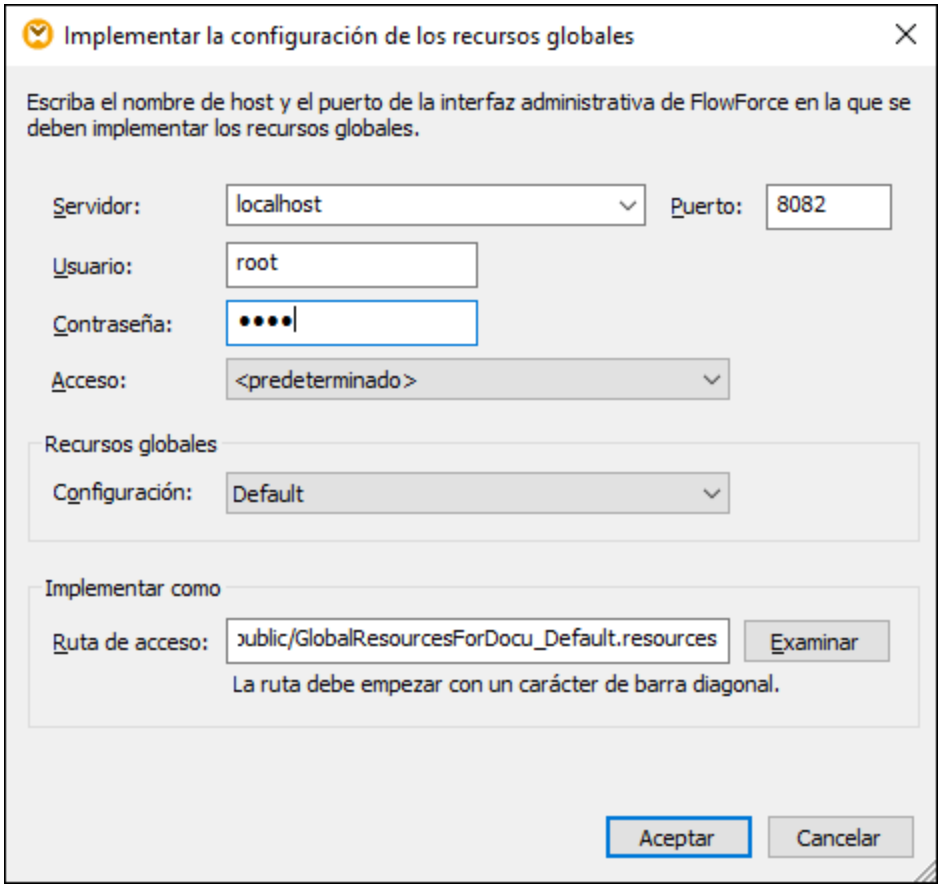
## Implementar recursos en FlowForce Server

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.

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 842.
5. Seleccione una configuración de la lista **Configuraciones**. Esta lista incluye todas las configuraciones del archivo de recursos globales actual 857. Sólo 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 869.


© 2019-2025 Altova GmbH

# 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 878
- Estructura de los catálogos en MapForce 880
- Personalizar catálogos 882
- Memoria insuficiente 884


Para más información consulte la especificación XML Catalogs.

# 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"
```

 © 2019-2025 Altova GmbH

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.

## 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\Common2025\Schemas |
| %ApplicationWritableDataFolder% | C:\ProgramData\Altova |


© 2019-2025 Altova GmbH

*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.

## 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.

 © 2019-2025 Altova GmbH

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.

## 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:

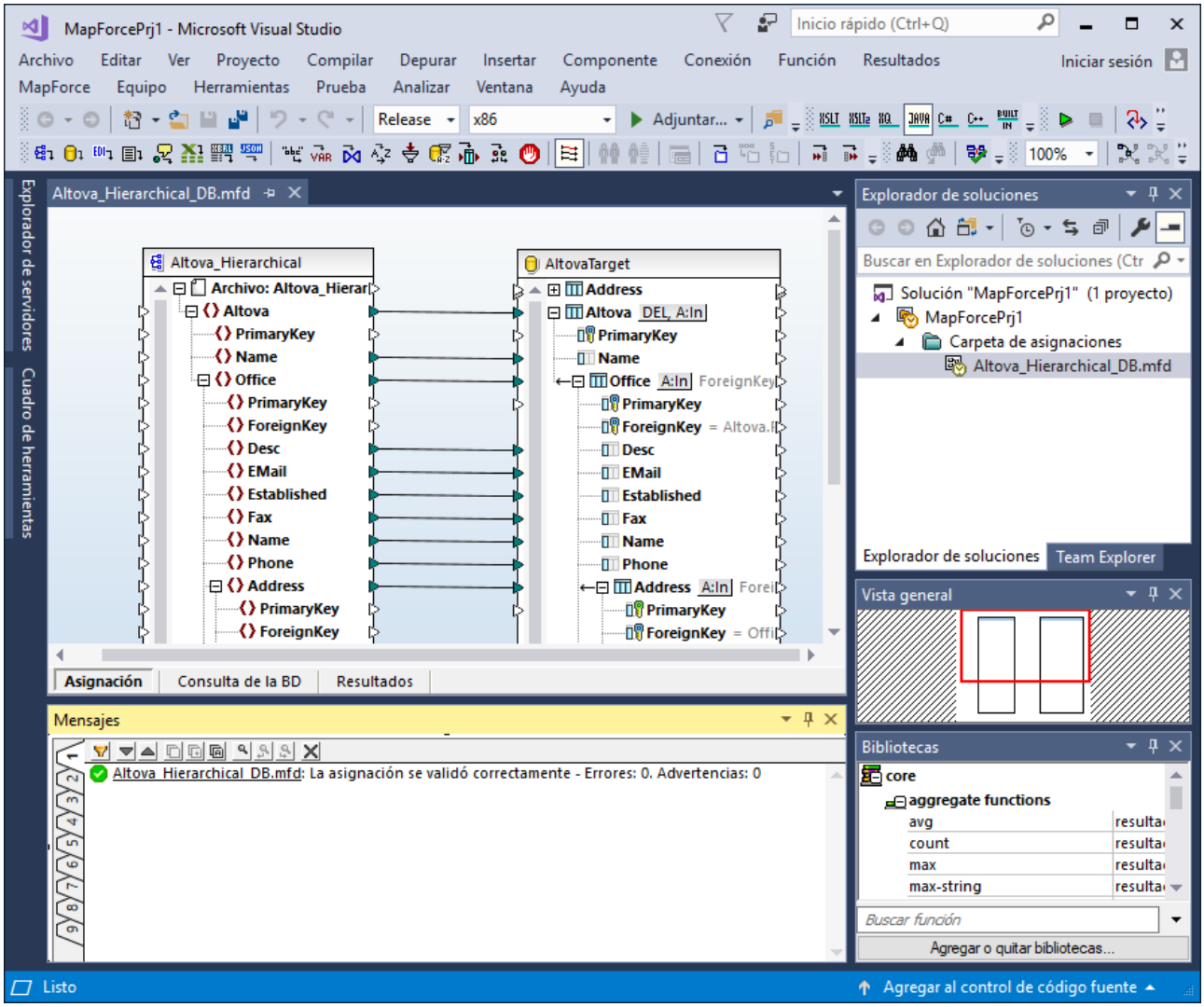
| | |
|---|---|
| `%PersonalFolder%` | Ruta de acceso completa de la carpeta personal del usuario actual, por ejemplo `C:\Usuarios\<nombre>\Documentos` |
| `%CommonSchemasFolder%` | `C:\ProgramData\Altova\Common2025\Schemas` |
| `%ApplicationWritableDataFolder%` | `C:\ProgramData\Altova` |
| `%AltovaCommonFolder%` | C:\Archivos de programa\Altova\Common2025 |
| `%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. |


© 2019-2025 Altova GmbH

| | |
|---|---|
| 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. |

# 12 Complemento de MapForce para Visual Studio

Puede integrar MapForce 2025 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.

 © 2019-2025 Altova GmbH

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 2025). 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 sólo tiene que buscar los tipos de archivos relacionados con MapForce.

- Recursos globales

  Los recursos globales 856 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 1119 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**

- 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.

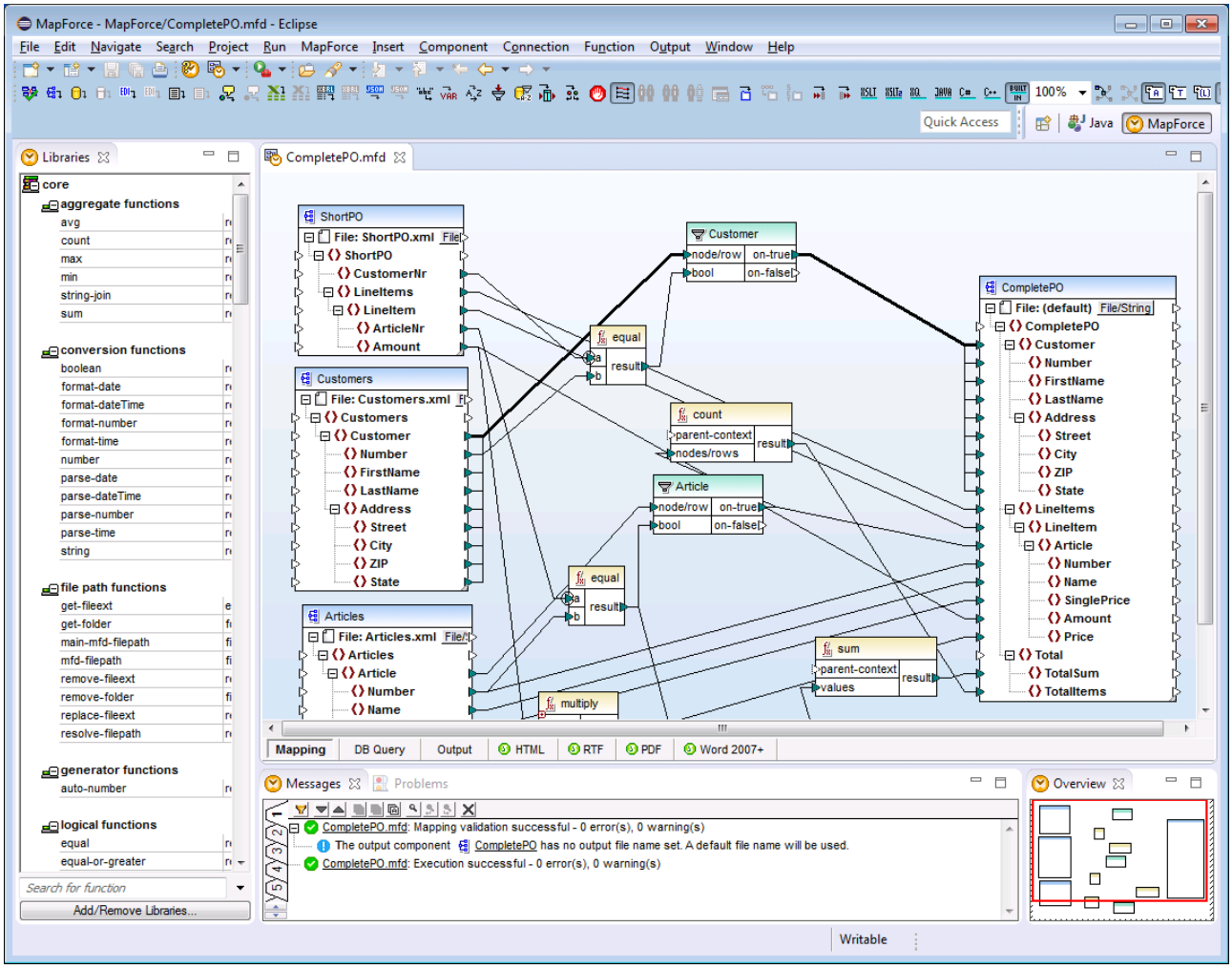
- Ayuda y soporte técnico

  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**


© 2019-2025 Altova GmbH

# 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 2024-12 (4.34), 2024-09 (4.33), 2024-06 (4.32), 2024-03 (4.31) 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 890
- Perspectiva MapForce 892
- Acceso a menús y funciones frecuentes 895
- Trabajar con asignaciones y proyectos 899
- Ampliar el complemento de MapForce para Eclipse 909

# 13.1 Instalación del complemento de MapForce para Eclipse

## Requisitos

- Eclipse 2024-12 (4.34), 2024-09 (4.33), 2024-06 (4.32), 2024-03 (4.31) (http://www.eclipse.org), de 64 bits.
- 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:

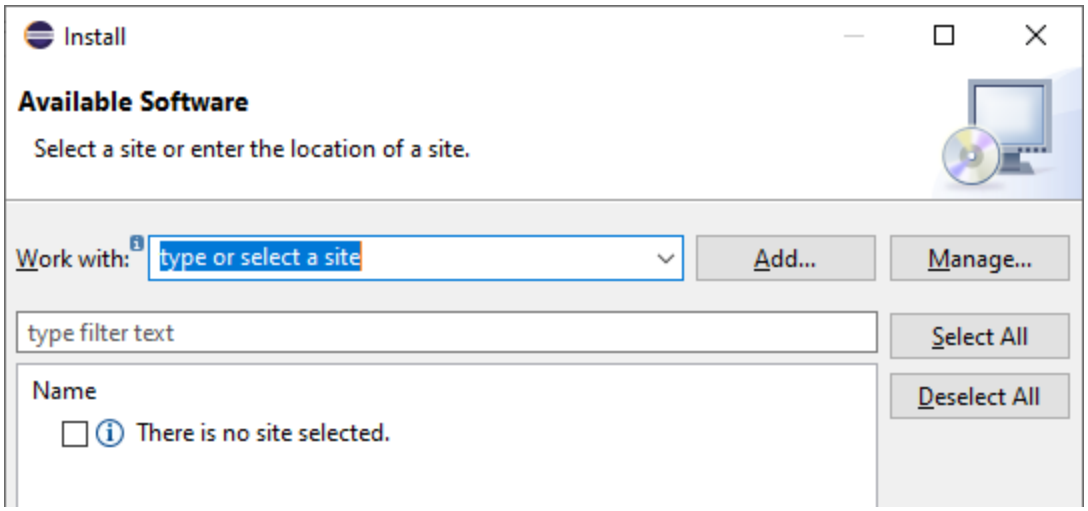
1. Ejecute el paquete de integración de MapForce para iniciar el asistente de instalación.
2. Cuando el instalador se lo pida seleccione *Instalar el complemento para Eclipse* y haga clic en **Aceptar**.
3. Cuando le pida que escoja cómo quiere integrar el complemento de MapForce en Eclipse seleccione *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.

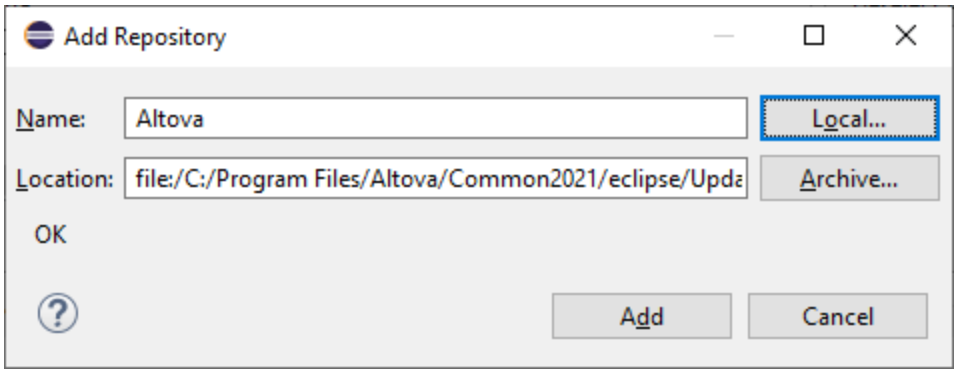
## Integrar el complemento de MapForce para Eclipse manualmente

1. En Eclipse seleccione el comando de menú **Help | Install new software**.
2. En el cuadro de diálogo Instalar haga clic en **Add** (Agregar).

 © 2019-2025 Altova GmbH




3. En el cuadro de diálogo "Agregar repositorio" haga clic en **Local**. Navegue hasta la carpeta `C:\Archivos de programa\Altova\Common2025\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:\Archivos de programa\Altova\MapForce2025\eclipse\UpdateSite` y un nombre como "Altova MapForce".
5. En el cuadro de diálogo seleccione *Sólo 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 (p.ej. iconos que faltan), pruebe a iniciar Eclipse desde la línea de comandos con el elemento flag `-clean`.

# 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. Permiten cambiar de un archivo a otro sin necesidad de abrir un entorno de desarrollo nuevo (el entorno asociado está disponible en la perspectiva)

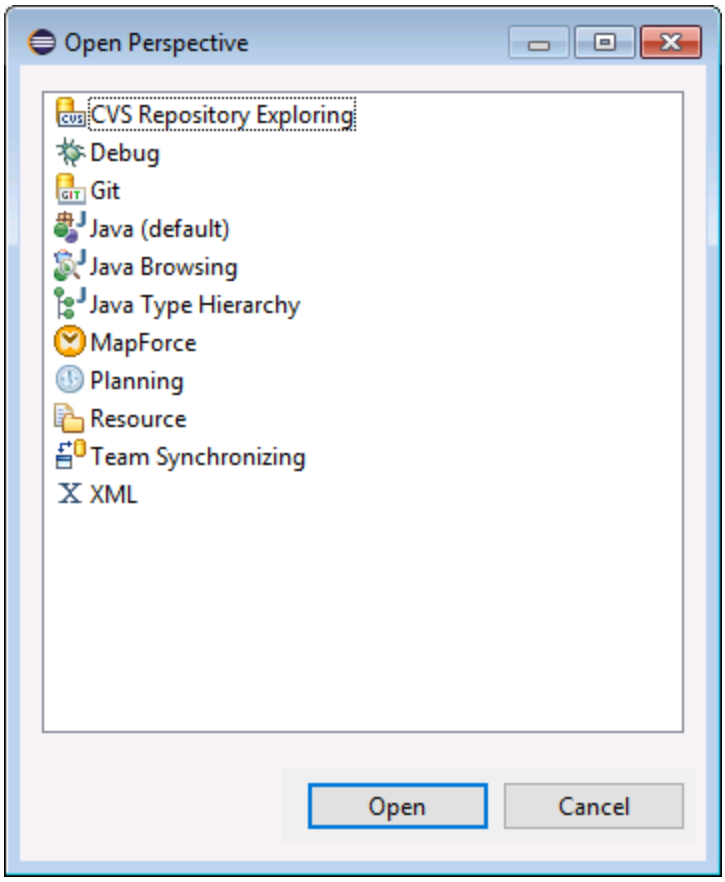
Para trabajar con la perspectiva MapForce es necesario:

- 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** o **Open**.


© 2019-2025 Altova GmbH




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 *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 **Yes**.

## 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

- 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* (Teclas de acceso rápido) del cuadro de diálogo Customize Perspective (Personalizar perspectiva) 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 and Close** para terminar de personalizar la perspectiva y que los cambios surtan efecto.


© 2019-2025 Altova GmbH

# 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 892).

**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 1103.

## 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) sólo están disponibles cuando dicho tipo de archivo está activo en el editor de Eclipse.

## Recursos globales

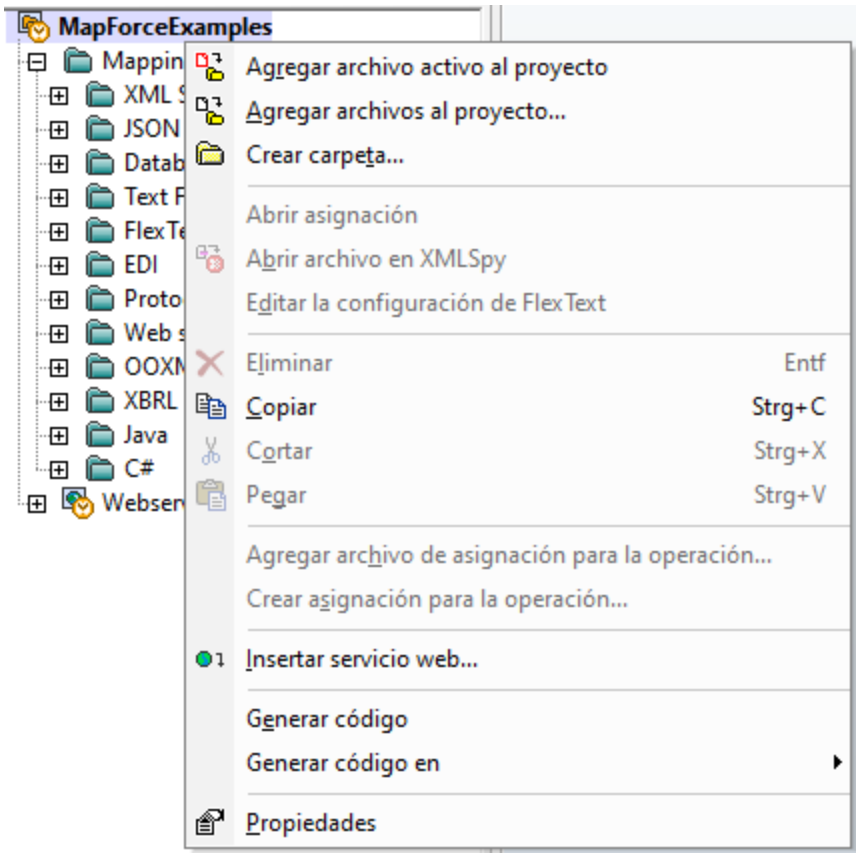
Hay dos maneras de acceder a los recursos globales y gestionarlos:

- 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.

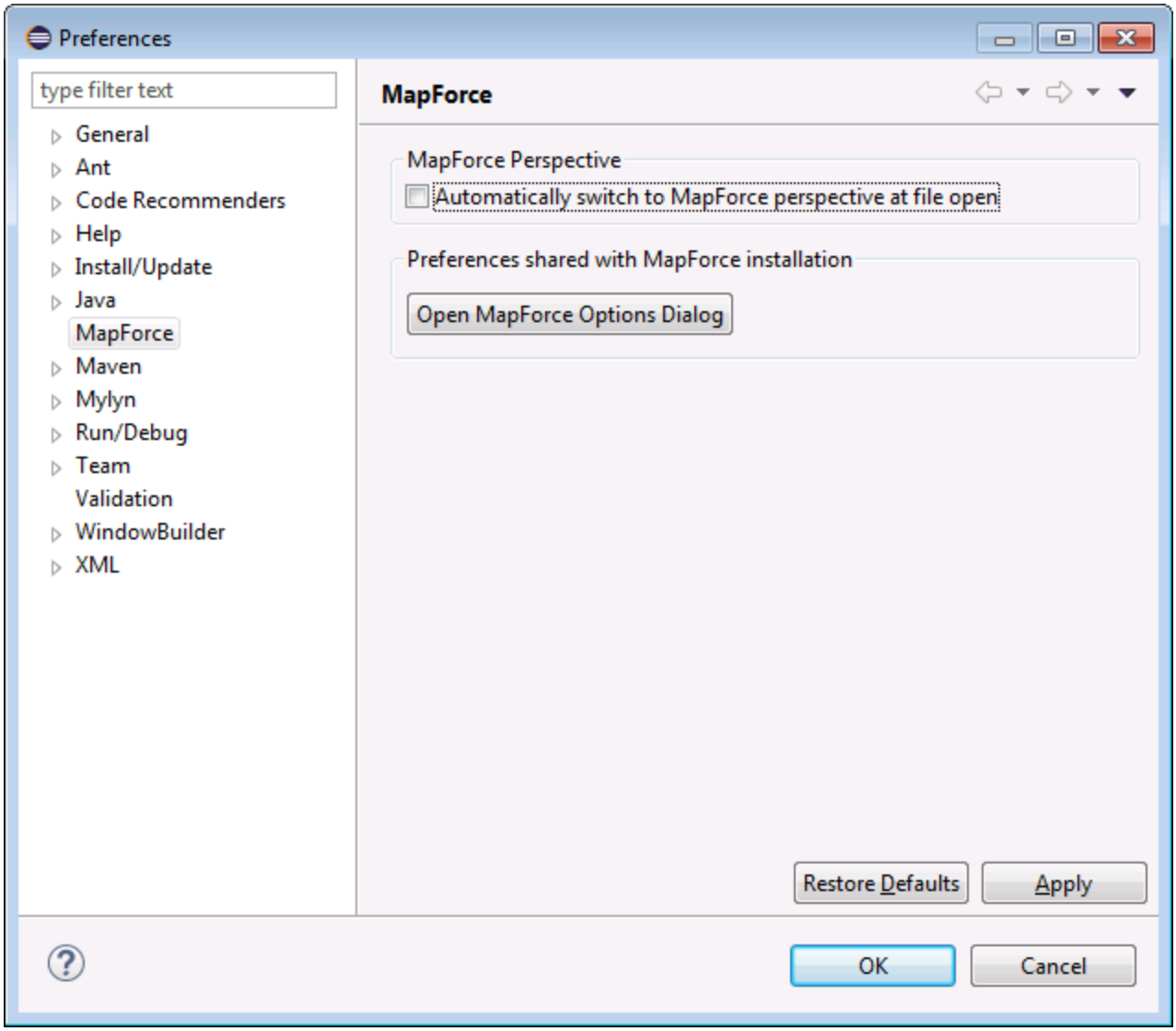



Tenga en cuenta que, además de proyectos estándar de MapForce (.mfp), en Eclipse también puede crear proyectos de tipo MapForce/Eclipse. Este tipo de proyectos son de carácter mixto y pueden configurarse para la generación automática de código de MapForce. Para más información consulte el apartado Trabajar con asignaciones y proyectos 899.

## 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**.

 © 2019-2025 Altova GmbH




*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 892.

## 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.

## Ayuda y soporte técnico

Los menús Ayuda, Soporte técnico, Buscar actualizaciones y Acerca de están en el menú **Help | Ayuda de MapForce** de Eclipse.


© 2019-2025 Altova GmbH

# 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.

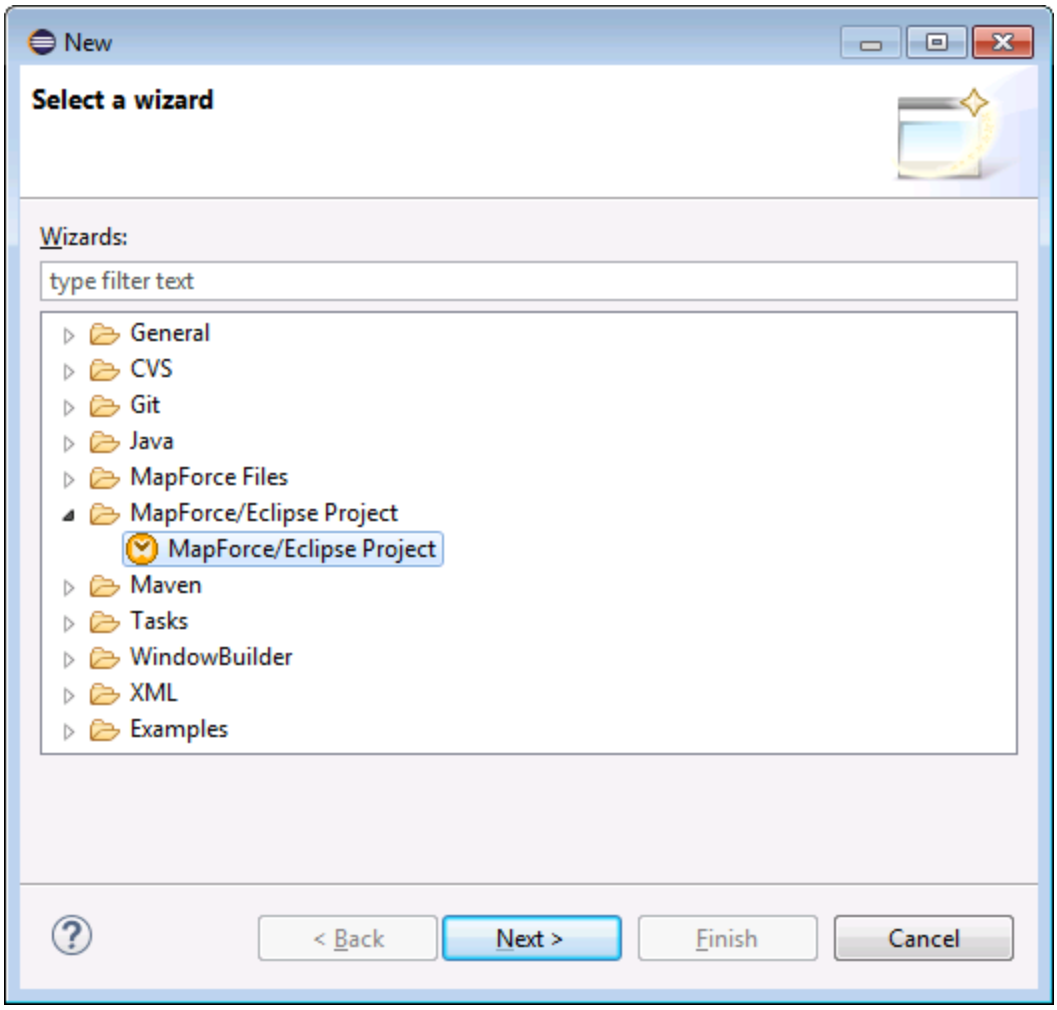
Esta sección incluye varios apartados:


- Crear un proyecto MapForce/Eclipse 899
- Crear asignaciones nuevas 901
- Importar asignaciones a un proyecto de Eclipse 903
- Configurar la generación automática de código MapForce 906


## 13.4.1 Crear un proyecto MapForce/Eclipse

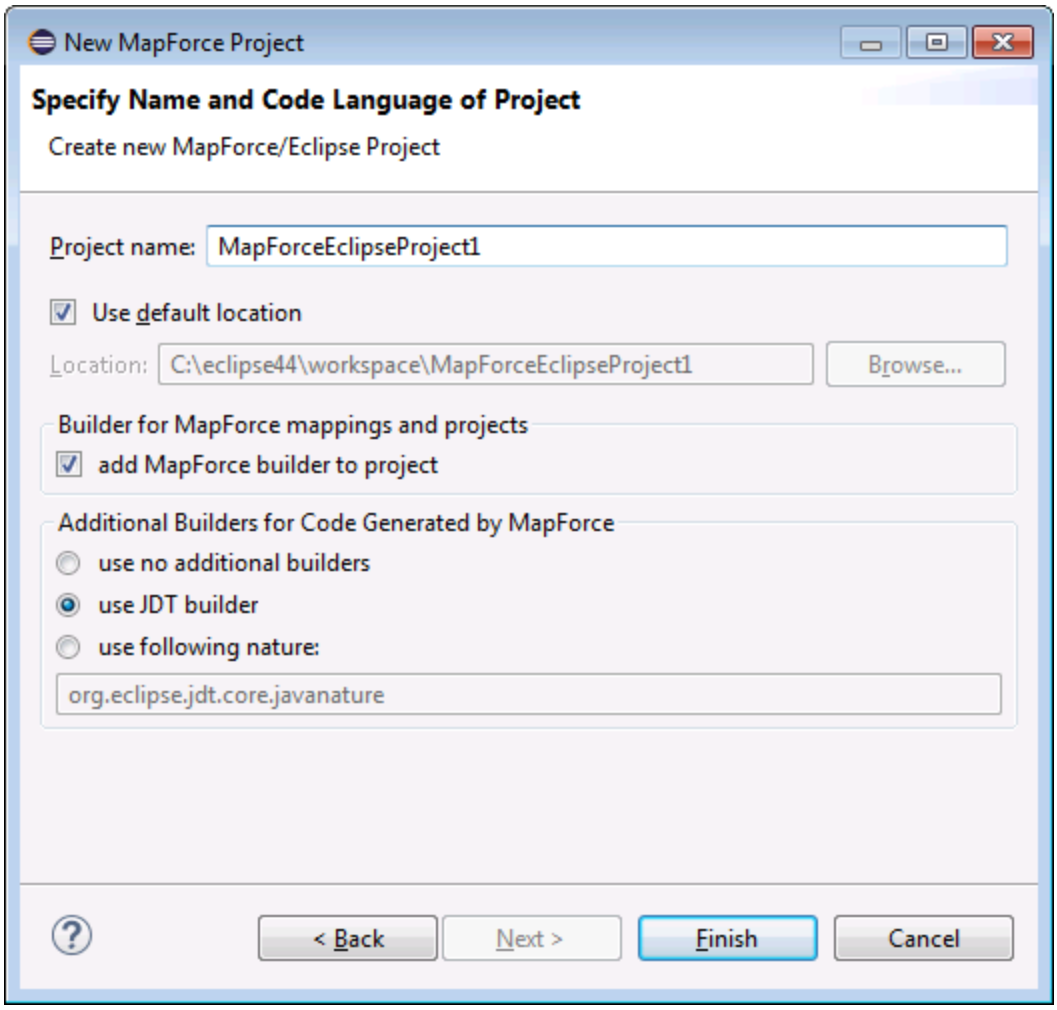
Para crear un proyecto MapForce/Eclipse:

1. En el menú **File** haga clic en **New | Other**.
2. Seleccione la categoría **MapForce/Eclipse Project**.




3. Haga clic en **Next** para continuar.


© 2019-2025 Altova GmbH




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.

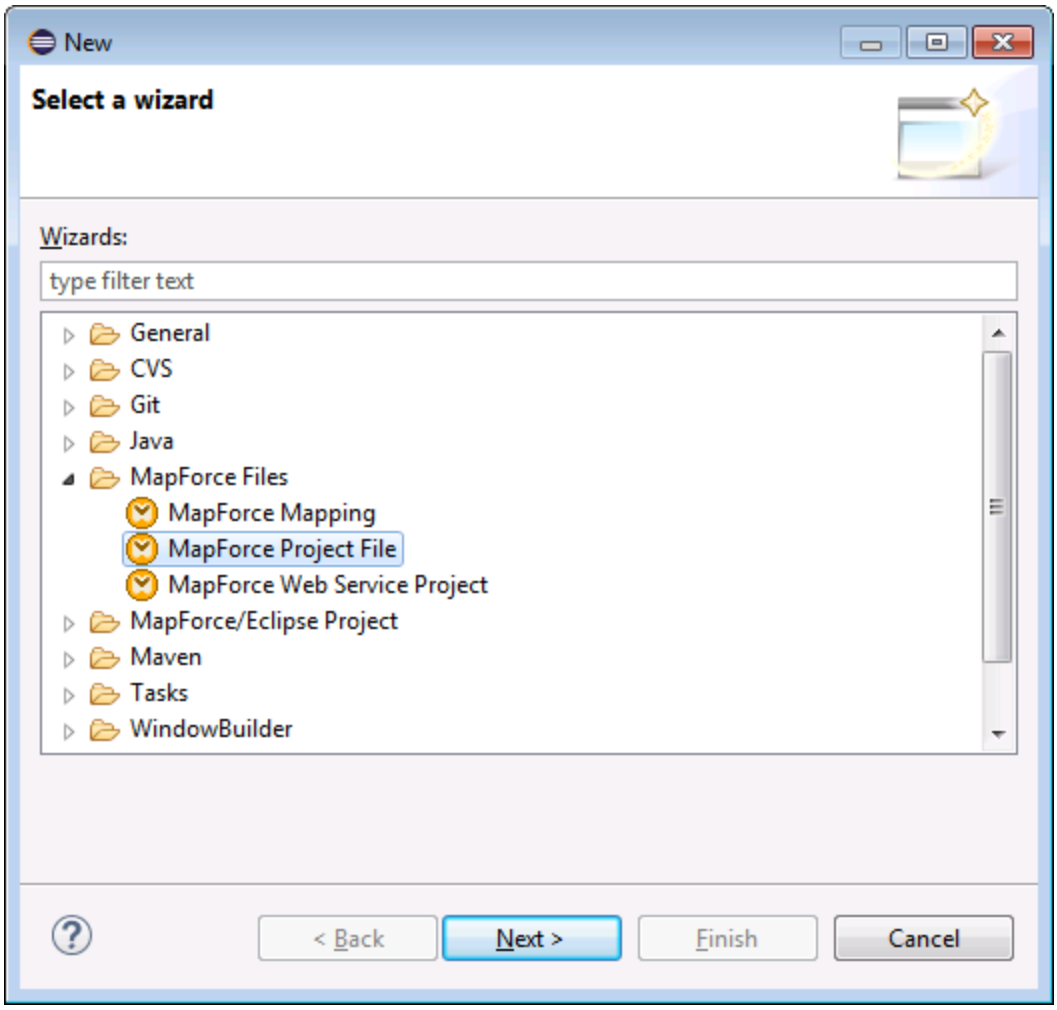
## 13.4.2 Crear asignaciones nuevas

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 (sólo en la edición MapForce Enterprise)

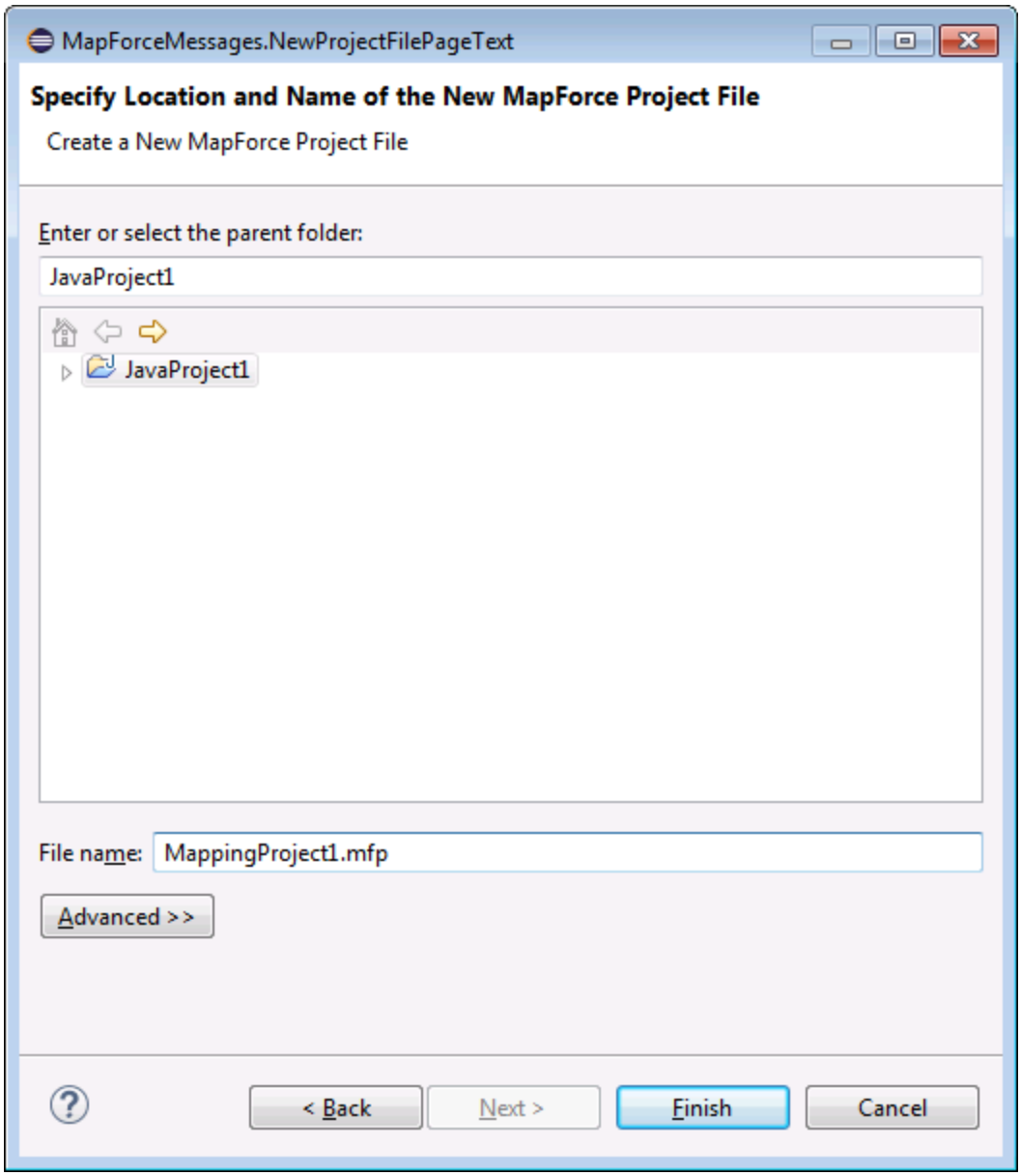
**Para crear este tipo de archivos dentro de un proyecto de Eclipse:**

1. Cree un proyecto de Eclipse nuevo o abra un proyecto actual.
2. En el menú **File** haga clic en **New | Other**.




3. Seleccione el tipo de archivo que desea crear en el cuadro de diálogo y haga clic en **Next** para continuar.


© 2019-2025 Altova GmbH

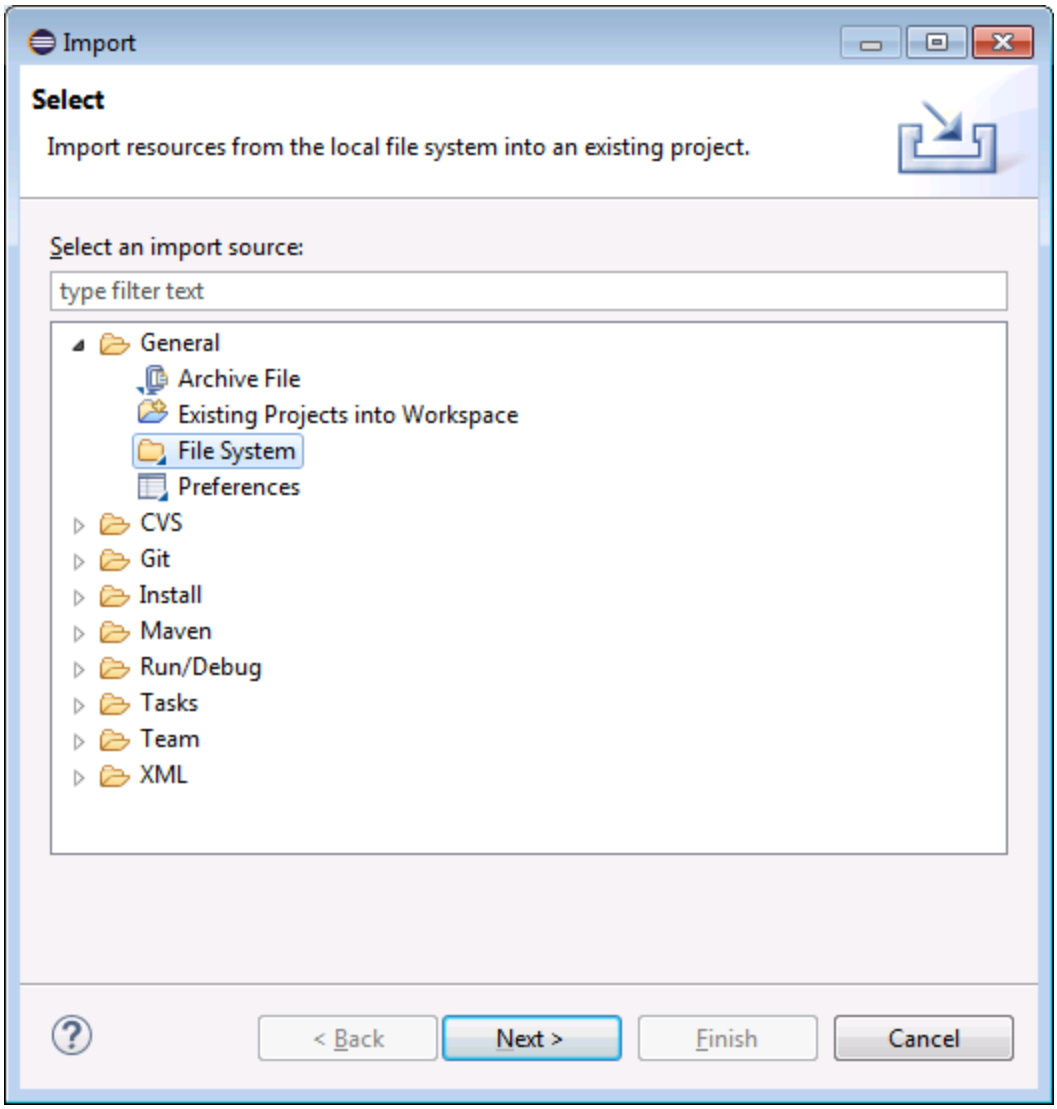



4. Seleccione una carpeta raíz del proyecto actual y haga clic en **Finish** para finalizar.

## 13.4.3 Importar asignaciones a un proyecto de Eclipse

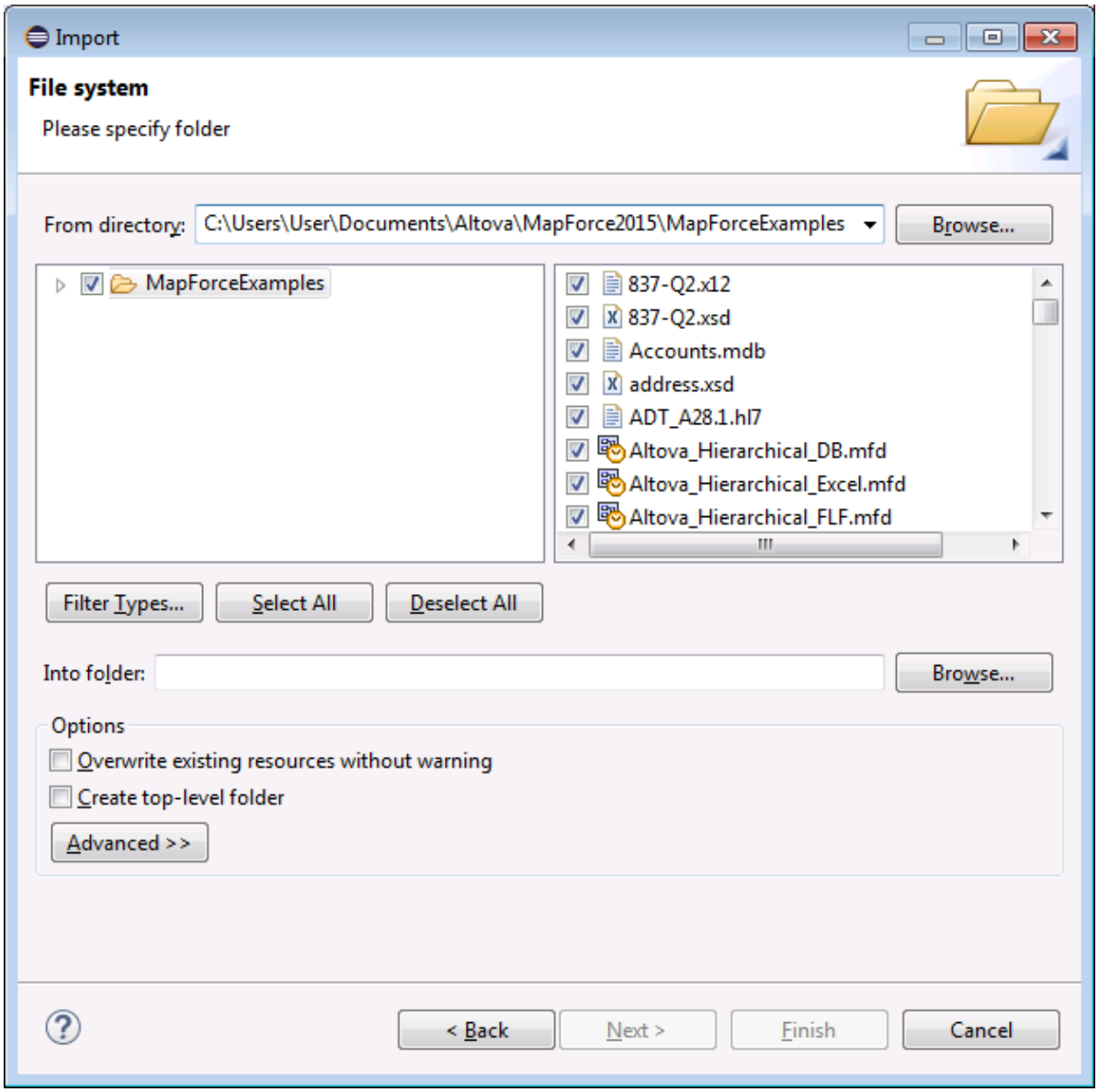
Para importar asignaciones de datos de MapForce y sus archivos dependientes en un proyecto de Eclipse:

1. Abra el proyecto en el que desea importar archivos.
2. En el menú **File** haga clic en **Import**.

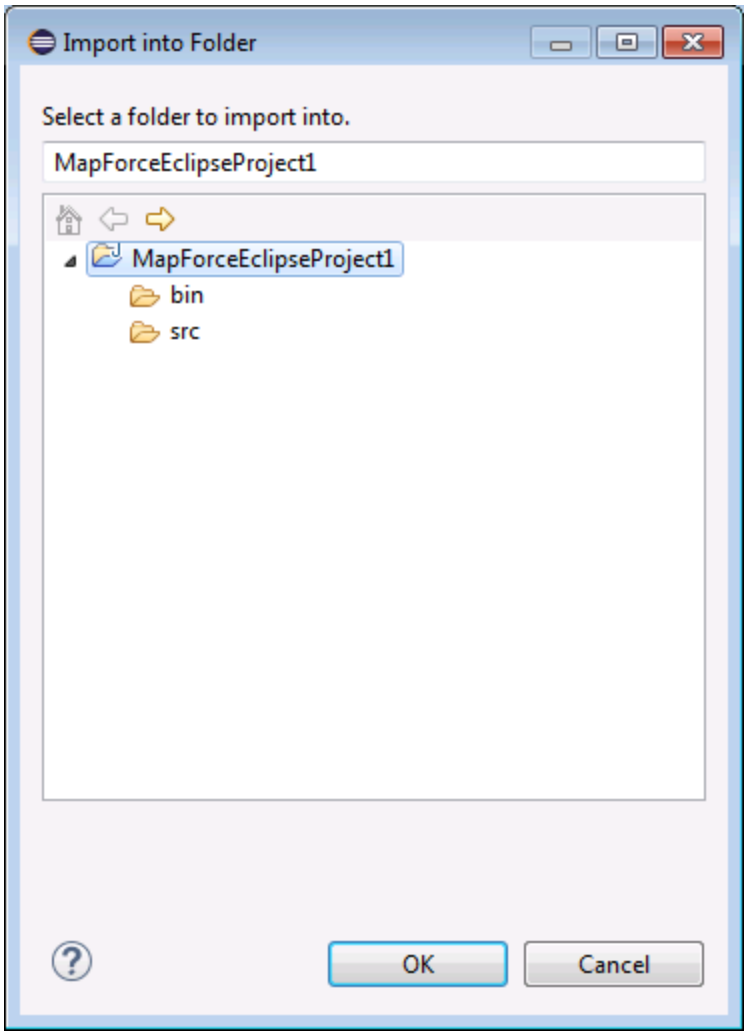



3. Seleccione **File System** y después haga clic en **Next** para continuar.


© 2019-2025 Altova GmbH




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*).




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* 899). 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:

 © 2019-2025 Altova GmbH

```
<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>
```

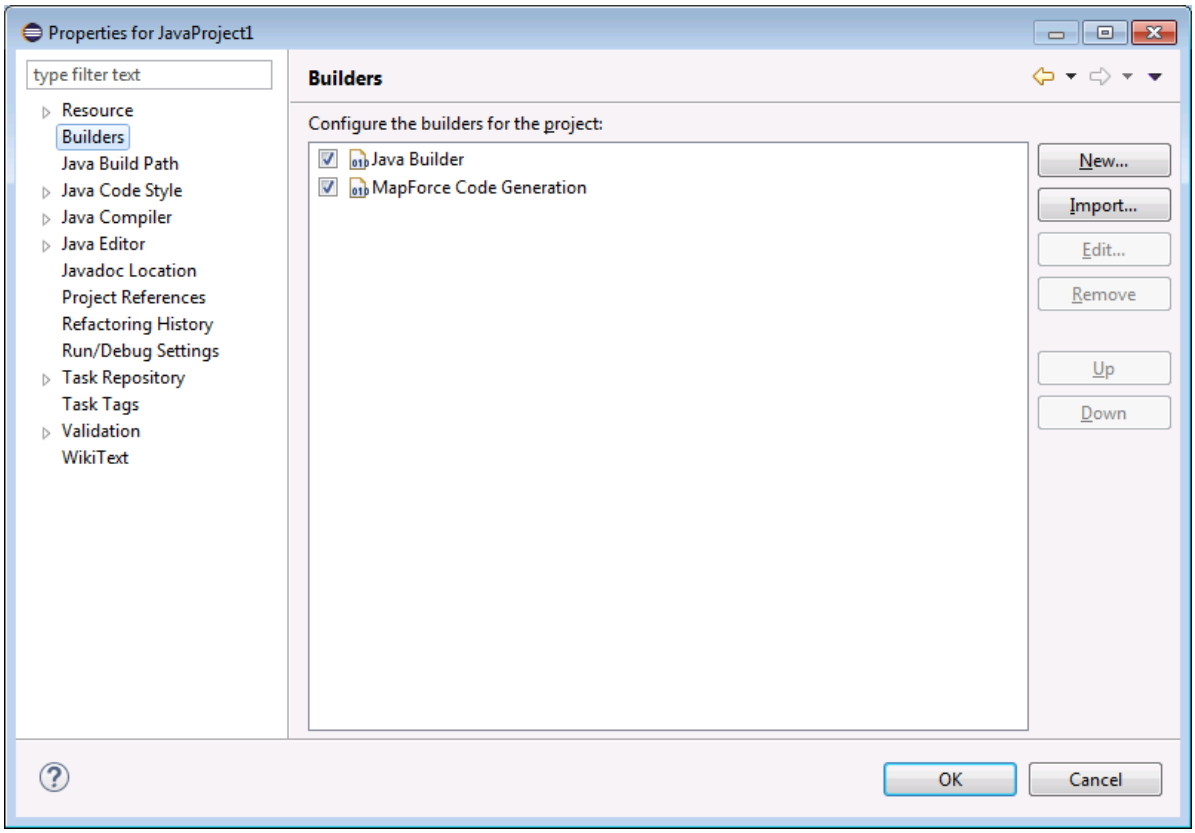
**Consejo:** puede abrir el archivo **.project** rápidamente desde la vista de navegación *Navigator* de Eclipse (para habilitar esta vista, haga clic en el comando de menú **Window | Show View | Navigator**).

### 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**.




3. Desactive la casilla **MapForce Code Generation** (Generación de código de MapForce).


© 2019-2025 Altova GmbH

## 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[1041] y de la API de MapForce.

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\MapForce2025\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\MapForce2025\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.

7. Abra cualquier asignación de MapForce en el nuevo área de trabajo y observe que se abre con un nivel de zoom del 70%.


© 2019-2025 Altova GmbH

# 14 Generador de código

El generador de código es una función integrada en MapForce que le permite generar código a partir de archivos de asignación. 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 sólo iniciar la aplicación recién generada. También podrá importar el código generado en otras aplicaciones o ampliarlo con más funciones.

*Información de compatibilidad*
En la siguiente tabla puede ver un resumen de la información de compatibilidad con los lenguajes de programación C++, C# y Java.

| 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<br><br>Marcos de destino:<br>• .NET Framework<br>• .NET Core 3.1<br>• NET 5.0<br>• NET 6.0<br>• NET 8.0 | Java SE JDK 8, 11, 17, 21 (incluido OpenJDK)<br>Eclipse 4.4 o superior<br>Apache Ant (archivo build.xml) |
| **Implementaciones XML DOM** | MSXML 6.0<br>Apache Xerces 3 | System.Xml | JAXP |
| **API de la base de datos** | ADO | ADO.NET | JDBC |

**Nota:** Sólo 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.

Para ello debe cumplir estas condiciones:

- Para compilar el código C++ generado debe tener Windows SDK instalado en su equipo.
- Para usar Xerces 3 para C++ debe instalarlo y compilarlo según las instrucciones en el sitio web de Apache Xerces. Asegúrese de que añade la variable de entorno XERCES3 que apunta al directorio en el que instaló Xerces (p.ej. `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 (p.ej. `%XERCES3%\bin`).

- 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. Para ello siga estos pasos:

  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 (*ver tabla anterior*).

*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*).

## Generar, compilar, ejecutar e integrar código

Para obtener instrucciones sobre cómo generar, compilar y ejecutar código, consulte el apartado Generar, compilar y ejecutar código 913. Para obtener más información sobre cómo integrar código generado con MapForce en su código personalizado, consulte Integrar código generado 919.

## Plantillas de generación de código

El código generado se puede personalizar con un archivo de plantilla llamado SPL 1020 (Spy Programming Language o lenguaje de programación de Spy). Puede personalizar la plantilla utilizada para la generación de código. Esta opción puede resultar útil, por ejemplo, cuando desea personalizar el código de acuerdo con las convenciones de escritura de su empresa o reemplazar bibliotecas específicas en el código generado.

## Ejemplos

Para ver ejemplos de generación de código, consulte los apartados Ejemplo: biblioteca 942 y Ejemplo: orden de compra 965.

 © 2019-2025 Altova GmbH

# 14.1 Generar, compilar y ejecutar código

En este apartado, se explica cómo generar código a partir de una asignación de datos y de un proyecto, así como cómo compilar y ejecutar el código generado. Existen situaciones en las que puede ser necesario modificar el código C#/C++/Java generado para poder integrarlo en su código personalizado. Para más detalles consulte Integrar código generado 919.

## Generar código a partir de una asignación de datos

Para generar código a partir de un diseño de asignación (**`.mfd`**), siga estos pasos:

1. En el cuadro de diálogo **Opciones** (para C# y C++) o en las opciones de la asignación 78, vaya a la sección *Generación* y seleccione las opciones de generación de código relevantes. Para más detalles sobre la configuración de la generación de código en el cuadro de diálogo **Opciones**, véase el apartado Generación 1131.
2. Haga clic en **Archivo | Generar código en** y seleccione el lenguaje de transformación correspondiente. Otra opción es seleccionar **Archivo | Generar código en el idioma seleccionado**. En este caso, el código se generará en el idioma que haya seleccionado en la barra de herramientas.
3. Seleccione el directorio de destino para los archivos que se deben generar y haga clic en **Aceptar** para confirmar. MapForce genera el código y muestra el resultado de la operación en la ventana Mensajes 27.

## Generar código a partir de un proyecto (ediciones Professional y Enterprise)

Puede generar código C# a partir de proyectos de asignación (**`.mfp`**) que consistan en varios archivos de diseño de asignación (**`.mfd`**). Tenga en cuenta que 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 1150.

Para generar código a partir de un proyecto de asignación, siga las instrucciones a continuación.

1. Abra el proyecto de asignación correspondiente a partir del cual quiere generar el código.
2. Haciendo clic con el botón derecho en el nombre del proyecto en la ventana Proyecto y entonces eligiendo el comando de menú **Propiedades** del menú contextual. Otra opción es hacer clic en el nombre del proyecto y seleccionar el comando **Propiedades** del menú **Proyecto**.
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 **Generar código para todo el proyecto**.

Independientemente del idioma seleccionado en el cuadro de diálogo **Propiedades del proyecto**, siempre puede optar por generar el código del proyecto en un idioma diferente, seleccionando el comando de menú **Proyecto | Generar código en | <idioma>**.

Los progresos y resultados de la generación de código aparecen en la ventana Mensajes. Por defecto, el nombre de la aplicación que se ha generado es igual al nombre de proyecto. Si el nombre del proyecto contiene espacios, éstos se convertirán en guiones bajos en el código generado. El código se genera por defecto en el directorio donde está ubicado el proyecto de MapForce, en el subdirectorio **`output`**.

Puede modificar el directorio de salida y/o el nombre del proyecto en el cuadro de diálogo **Propiedades del proyecto**. Si su proyecto de MapForce tiene carpetas, podrá cambiar las opciones de generación de código

para cada una de las carpetas por separado: Haga clic con el botón derecho en una carpeta de interés y seleccione **Propiedades** del menú contextual. Sino todas las carpetas del proyecto heredan por defecto las opciones de generación de código del nivel superior. Para obtener más información sobre los proyectos, su configuración y los procedimientos relacionados con ellos, consulte el apartado Proyectos 98.

## Información sobre los lenguajes de transformación

En este subapartado se describen las particularidades de generar código en diferentes lenguajes de transformación. Después se explica cómo compilar el código generado en C++, C# y Java y ejecutar la aplicación. También puede generar código en XSLT 1-3 y XQuery. Para más información consulte el apartado Generación de código 70.

*Código C++ y C#*
Por lo general, generar, compilar y ejecutar los códigos C++ y C# funciona de la misma manera. En las subsecciones más abajo se describen los procedimientos generales.

Una vez haya generado el código C++ o C#, la solución incluirá los siguientes componentes:

- Los archivos de solución (**`.sln`**) y de proyecto (**`.vcproj`** para C++ y **`.csproj`** para C#) que se pueden abrir con Visual Studio.
- Varias bibliotecas firmadas por Altova que la asignación de datos requiere (todas empiezan con el prefijo `Altova`).
- El proyecto de asignación principal (por defecto se llama `Mapping`) que incluye la aplicación de asignación de datos y sus archivos independientes.

Recuerde que puede cambiar el nombre predeterminado del proyecto de asignación en el cuadro de diálogo Configurar la asignación 78.

Después de haber generado el código C++/C#, los siguientes pasos consistirán en compilar el código y ejecutar la aplicación. Hay dos métodos principales para compilar el código generado: (i) en Visual Studio y (ii) en la línea de comandos (*ver detalles más abajo*). Tenga en cuenta que para crear código C# debe tener instalado el SDK correspondiente y una versión compatible de Visual Studio. Para obtener el paquete de instalación correspondiente a su sistema operativo y plataforma, consulte el sitio web de Microsoft.

***Compilar el código generado en Visual Studio***
Para compilar el código C++/C# generado, siga estos pasos:

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.
2. Seleccione la configuración de compilación correspondiente (p.ej. Depuración). Nota sobre el código C#: Recuerde que solamente las opciones Unicode admiten el conjunto de caracteres Unicode completo en XML y en otros archivos. Las compilaciones que no son Unicode trabajan con la página de código local de su instalación Windows.
3. En el menú **Compilar** haga clic en **Compilar solución**.

Como resultado de la compilación del código, se creará una aplicación de línea de comandos llamada **`Mapping.exe`** junto con sus archivos relacionados. La aplicación de asignación se encontrará en uno de los subdirectorios relativos al archivo **`.sln`**. El nombre del subdirectorio depende de la configuración de compilación seleccionada: p.ej. **`Depuración`** (C++), **`bin\Depuración`** (C#).

 © 2019-2025 Altova GmbH

***Compilar el código generado en la línea de comandos***

Para compilar el código generado en la línea de comandos, cambie al directorio con el código generado y ejecute el siguiente comando:

```
devenv Mapping.sln /Build "Debug|AnyCPU" /Project Mapping
```

Este comando llama a Visual Studio y especifica el nombre del archivo de solución a compilar (en nuestro caso es **`Mapping.sln`**), la configuración deseada (en nuestro caso es Depuración y cualquier CPU), y el nombre del proyecto al que pertenece el archivo de solución (`Mapping`). Como resultado de la compilación del código, se creará una aplicación de línea de comandos llamada **`Mapping.exe`** junto con sus archivos relacionados. La aplicación de asignación se encontrará en uno de los subdirectorios relativos al archivo **`.sln`**. El nombre del subdirectorio depende de la configuración de compilación seleccionada: p.ej. **`Depuración`** (C++), **`bin\Depuración`** (C#).

*Notas sobre la compilación de código C#*

Además de llamar a Visual Studio, también puede llamar a .NET para compilar el código C# generado. Para ello siga estos pasos:

1. Antes de generar código C++ o C#, asegúrese de seleccionar la plataforma correcta en la sección *Generación* del cuadro de diálogo **Opciones** (en el menú **Herramientas**). Para más información consulte el apartado Generación 1131.
2. Si su destino es .NET/.NET Core, configure las variables de entorno PATH y DOTNET_ROOT para que apunten a la ubicación en la que está instalado .NET/.NET Core. Esto evitará posibles problemas con los comandos de la línea de comandos.
3. Abra un símbolo del sistema y cambie al directorio con el código generado.
4. Ejecute el siguiente comando:

   ```
   dotnet build Mapping\Mapping.sln --configuration Release
   ```

   Este comando llama a .NET para que construya el archivo de la solución llamado `Mapping.sln` con la configuración de `Release` y crea un ejecutable llamado **`Mapping.exe`** junto con sus archivos relacionados en la carpeta de destino. El nombre de la carpeta de destino depende de la configuración seleccionada y la versión de .NET utilizada. En nuestro ejemplo el ejecutable se guardará en la carpeta **`bin\Release\net8.0`**.

***Ejecutar la aplicación***

Una vez haya compilado el código, o en Visual Studio directamente o en la línea de comandos, puede proceder a ejecutar la aplicación. Para ejecutar la aplicación, haga doble clic en **`Mapping.exe`** o llame al ejecutable desde la línea de comandos. Una vez haya ejecutado el ejecutable, el resultado de la transformación de la asignación se mostrará en la carpeta de destino (por defecto es la carpeta donde está almacenado el ejecutable).

Si compila el código generado en Linux, el ejecutable generado se llamará `Mapping` sin ninguna extensión. Para ejecutar el ejecutable, puede que tenga que usar este comando:

```
./Mapping
```

*Código Java*

Una vez haya generado el código Java, el proyecto Java incluirá los siguientes componentes:

- Varias bibliotecas Java firmadas por Altova que la asignación de datos requiere (todas empiezan con el prefijo `com.altova`)
- El paquete `com.mapforce` incluye la aplicación de asignación y sus archivos dependientes, entre los que se encuentran los dos archivos más importantes que especifican los puntos de entrada de la aplicación:
  - La aplicación de asignación de datos Java como aplicación de diálogo (`MappingApplication.java`)
  - La aplicación de asignación de datos Java como aplicación de consola (`MappingConsole.java`)
- El archivo `build.xml` que se puede ejecutar con Apache Ant para compilar el proyecto y generar archivos JAR

Los nombres predeterminados de la aplicación de asignación y sus archivos dependientes en el paquete `com.mapforce` llevan el prefijo **`Mapping`**. Puede modificar el nombre predeterminado así como otras opciones en la configuración de la asignación 78.

Después de haber generado el código Java, los siguientes pasos consistirán en compilar y ejecutar el código. Hay dos métodos principales para compilar el código generado y ejecutar la aplicación: (i) en Eclipse y (ii) en la línea de comandos, usando Apache Ant. En las subsecciones más abajo se describen los procedimientos generales.

***Compilar el código generado y ejecutar la aplicación en Eclipse***

En este método se utiliza el flujo de trabajo de Eclipse. Para ello debe cumplir estas condiciones:

- Debe tener instalado en su sistema un Java Development Kit (JDK), Eclipse y Apache Ant. Eclipse suele incluir una versión de Ant, pero también se puede instalar Ant por separado.
- Para ejecutar Eclipse con OpenJDK, debe configurar la variable de entorno PATH de modo que incluya la ruta al directorio **`bin`** de JDK (p.ej. **`C:\Java\jdk-11.0.1\bin`**).
- La variable de entorno JAVA_HOME debe apuntar a la ubicación de JDK.
- La variable de entorno ANT_HOME debe apuntar a la ubicación de Apache Ant.

Una vez haya generado el código Java, el siguiente paso consiste en importarlo a Eclipse. Para ello siga estos pasos:

1. En el menú **File** haga clic en el comando **Import** y después seleccione **General | Existing Projects into Workspace** (proyectos actuales al espacio de trabajo).
2. Haga clic en **Next** para continuar.
3. 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**.

Tenga en cuenta que, por defecto, el código se compila automáticamente cada vez que se detecta un cambio. También puede desactivar esta función y compilar el código cuando sea necesario (consulte la documentación de Eclipse para más información).

Una vez haya importado y compilado el código, el siguiente paso consiste en ejecutar la aplicación. En este subapartado explicamos algunos de los posibles métodos para solucionar problemas en la ejecución de una aplicación.

 © 2019-2025 Altova GmbH

*Método 1: Ejecutar el proyecto como aplicación*
Este método le permite ejecutar su proyecto Java como aplicación de interfaz gráfica. Para ello siga estos pasos:

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.

*Método n°2: Ejecutar el proyecto como aplicación de consola*
Este método le permite ejecutar su proyecto Java como aplicación de consola (línea de comandos). Para ello siga estos pasos:

1. En la vista 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.

Como resultado de haber ejecutado la aplicación, independientemente del método seleccionado, la aplicación Java ejecutará la transformación de asignación de datos y generará el archivo o los archivos de salida en la ruta de acceso de destino.

***Compilar el código generado y ejecutar la aplicación en la línea de comandos***
Para poder compilar y ejecutar el código generado en la línea de comandos, debe tener instalados los siguientes componentes y configurar las variables de entorno:

- Debe tener instalado en su sistema un Java Development Kit (JDK) y Apache Ant.
- La ubicación del directorio **`bin`** de Ant (p.ej. **`C:\apache-ant-1.10.5\bin`**) debe añadirse a la variable de entorno PATH. Esto le permitirá ejecutar Ant cómodamente sin tener que escribir la ruta completa al ejecutable en la línea de comandos.
- La variable de entorno JAVA_HOME debe apuntar a la ubicación de JDK.
- La variable de entorno ANT_HOME debe apuntar a la ubicación de Apache Ant.

Para compilar el código generado con Apache Ant, siga estos pasos:

1. Abra un símbolo del sistema y vaya al directorio donde se encuentra el código generado, incluido el archivo de compilación (**`build.xml`**).
2. Ejecute el siguiente comando:

   ```
   ant jar
   ```

   Este comando compilará el código generado y creará un archivo JAR (que tiene el nombre predeterminado **`Mapping.jar`**). El archivo JAR sirve para empaquetar archivos Java **`.class`** junto con sus metadatos y recursos relacionados. En nuestro caso, el archivo JAR está pensado para ser utilizado como un programa de Java ejecutable. Por tanto, el archivo de manifiesto del archivo incluye el punto de entrada de la aplicación (por defecto es `com.mapforce.MappingConsole`).

Para ejecutar la aplicación Java, ejecute el siguiente comando en el directorio donde se encuentra el archivo JAR:

```
java com.mapforce.MappingConsole Mapping.jar
```

Este comando inicia la máquina virtual Java, lanza la clase principal llamada `com.mapforce.MappingConsole` que hace referencia al punto de entrada de la aplicación Java y ejecuta el programa incluido en el archivo JAR llamado **`Mapping.jar`**. Como resultado, la aplicación Java ejecutará la transformación de asignación de datos y generará el archivo o los archivos de salida en la ruta de acceso de destino. Si desea lanzar la aplicación como una aplicación de interfaz gráfica, introduzca el siguiente valor como argumento de la clase principal: `com.mapforce.MappingApplication`. Ahora se abrirá una ventana emergente en la que podrá iniciar la transformación de asignación de datos.

### *Prevención de posibles problemas de memoria insuficiente*

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 evitar las excepciones de memoria insuficiente, siga estos pasos:

1. 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: `-server -Xmx512m -Xms512m`.
2. 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`** a `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`**.

### *Prevención de posibles problemas con las conexiones JDBC*

Si ha generado código Java a partir de una asignación que se conecta a una base de datos mediante JDBC, puede que tenga que añadir el controlador JDBC como entrada classpath en la configuración actual. Si no lo hace, la ejecución de la aplicación podría resultar en un error. Para más detalles consulte el apartado Bases de datos 176.

 © 2019-2025 Altova GmbH

## 14.2 Integrar código generado

Aunque el resultado de la generación de código es una aplicación completa y totalmente funcional, es posible que tenga que adaptar el código generado con MapForce para poder integrarlo en su código personalizado. A continuación, se exponen algunos ejemplos típicos de modificación del código:

- Modificar archivos de origen y destino para la aplicación de asignación de datos 919
- Definir códigos de control de errores personalizados 919
- Cambiar el tipo de datos de la entrada de la asignación en el código C# o Java generado (p.ej. cambiar tipos string por tipos stream) 922
- Generar bibliotecas contenedoras de esquemas que se pueden integrar en otras aplicaciones para leer, modificar o escribir documentos XML mediante programación 928

### 14.2.1 Modificar datos de entrada/salida, definir gestión de errores

En este apartado se explica cómo modificar los archivos de origen y destino y definir la gestión de errores para la aplicación de asignaciones en código Java, C# y C++. Para ilustrar estos procedimientos, utilizamos la asignación de muestra llamada **`MapForceExamples\CompletePO.mfd`**. La asignación consta de tres componentes de origen (**`ShortPO.xml`**, **`Customers.xml`** y **`Articles.xml`**) y un componente de destino (**`CompletePO.xml`**).

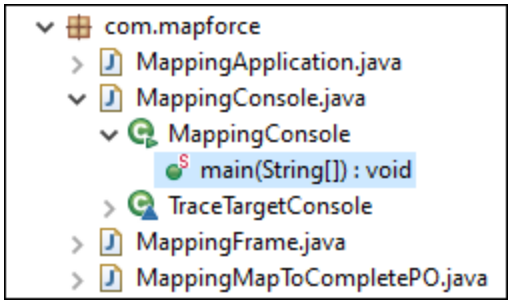
En el código generado estos orígenes y destinos se convertirán en tres parámetros de entrada y uno de salida que se aportan al método **`Run`** para ejecutar la asignación de datos. Tenga en cuenta estos aspectos básicos de la generación de código en MapForce:

- 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.

#### Java

El entorno IDE de Java utilizado para este ejemplo es Eclipse. Para empezar, genere código Java a partir de la asignación de muestra **`MapForceExamples\CompletePO.mfd`** y, a continuación, importe el proyecto en Eclipse. Para obtener más información sobre cómo generar código Java e importarlo a Eclipse, consulte el apartado Generar, compilar y ejecutar código 915.

Para editar la aplicación de consola Java generada, busque el método **`main`** de la aplicación generada en el Explorador de proyectos de Eclipse (*imagen siguiente*). Por defecto, este método se encuentra en la clase `MappingConsole` del paquete `com.mapforce`. Si, por el contrario, modificó el paquete base, entonces estará en la clase `MappingConsole` de su paquete personalizado.




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. Por defecto, el método **run** se invoca desde la clase `MappingFrame.java` del paquete `com.mapforce` (*imagen anterior*).

***Modificar datos de origen y de destino***

La lista de código que aparece a continuación es un extracto del método **main** de la aplicación de consola Java generada. Las rutas de acceso a los archivos de origen y destino que se pueden ver a continuación están definidos como parámetros del método **run**. Si necesita cambiar los orígenes y/o destinos, cambie los valores de los parámetros que se muestran a continuación.

```
com.altova.io.Input Customers2Source =
com.altova.io.StreamInput.createInput("C:/Usuarios/<usuario>/Documentos/Altova/MapForce20
24/MapForceExamples/Customers.xml");
                    com.altova.io.Input Articles2Source =
com.altova.io.StreamInput.createInput("C:/Usuarios/<usuario>/Documentos/Altova/MapForce20
24/MapForceExamples/Articles.xml");
                    com.altova.io.Input ShortPO2Source =
com.altova.io.StreamInput.createInput("C:/Usuarios/<usuario>/Documentos/Altova/MapForce20
24/MapForceExamples/ShortPO.xml");
                    com.altova.io.Output CompletePO2Target = new
com.altova.io.FileOutput("CompletePO.xml");
```

***Definir una gestión de errores personalizada***

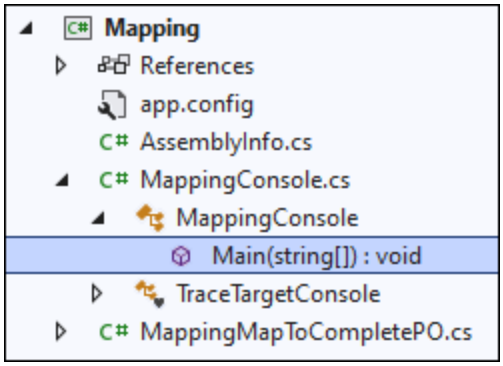
Si necesita añadir su código personalizado de gestión de errores, modifique la instrucción `catch` o en el método **main** (aplicación de consola) o en `MappingFrame.java` (aplicación IGU).

## C#

El entorno IDE de C# utilizado para este ejemplo es Microsoft Visual Studio. Para empezar, genere código C# a partir de la asignación de muestra **MapForceExamples\CompletePO.mfd** y, a continuación, abra la solución en Visual Studio. Para obtener más información sobre cómo generar código e importarlo a Visual Studio, consulte el apartado Generar, compilar y ejecutar código 914.

Para editar la aplicación C# generada, busque el método **Main** de la aplicación generada en el Explorador de Solución de Visual Studio (*imagen siguiente*). 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.

 © 2019-2025 Altova GmbH




***Modificar datos de origen y de destino***

La lista de código que aparece a continuación es un extracto del método **Main** de la aplicación C# generada. Las rutas de acceso a los archivos de origen y destino que se pueden ver a continuación están definidos como parámetros del método **Run**. Si necesita cambiar los orígenes y/o destinos, cambie los valores de los parámetros que se muestran a continuación.

```
Altova.IO.Input Customers2Source =
Altova.IO.StreamInput.createInput("C:/Usuarios/<usuario>/Documentos/Altova/MapForce2024/M
apForceExamples/Customers.xml");
                            Altova.IO.Input Articles2Source =
Altova.IO.StreamInput.createInput("C:/Usuarios/<usuario>/Documentos/Altova/MapForce2024/M
apForceExamples/Articles.xml");
                            Altova.IO.Input ShortPO2Source =
Altova.IO.StreamInput.createInput("C:/Usuarios/<usuario>/Documentos/Altova/MapForce2024/M
apForceExamples/ShortPO.xml");
                            Altova.IO.Output CompletePO2Target = new
Altova.IO.FileOutput("CompletePO.xml");
```

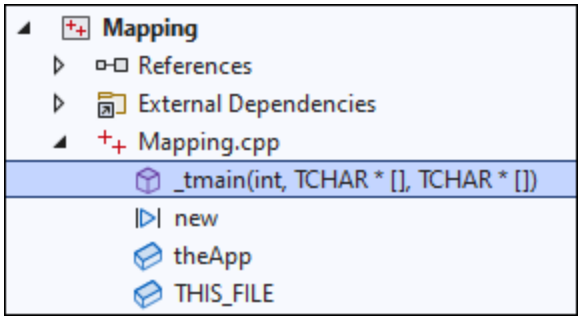
***Definir una gestión de errores personalizada***

Si necesita añadir su código personalizado de gestión de errores, modifique la instrucción `catch` en el método **Main**.

## C++

El entorno IDE de C++ utilizado para este ejemplo es Microsoft Visual Studio. Para empezar, genere código C++ a partir de la asignación de muestra **MapForceExamples\CompletePO.mfd** y, a continuación, abra la solución en Visual Studio. Para obtener más información sobre cómo generar código e importarlo a Visual Studio, consulte el apartado Generar, compilar y ejecutar código 914.

Para editar la aplicación C++ generada, busque el método **_tmain** de la aplicación generada en el Explorador de Solución de Visual Studio (*imagen siguiente*). 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.




***Modificar datos de origen y de destino***
La lista de código que aparece a continuación es un extracto del método **Main** de la aplicación C++ generada. Las rutas de acceso a los archivos de origen y destino que se pueden ver a continuación están definidos como parámetros del método **Run**. Si necesita cambiar los orígenes y/o destinos, cambie los valores de los parámetros que se muestran a continuación.

```
MappingMapToCompletePO MappingMapToCompletePOObject;
                      MappingMapToCompletePOObject.Run(

_T("C:/Usuarios/<usuario>/Documentos/Altova/MapForce2024/MapForceExamples/Customers.xml")
,

_T("C:/Usuarios/<usuario>/Documentos/Altova/MapForce2024/MapForceExamples/Articles.xml"),

_T("C:/Usuarios/<usuario>/Documentos/Altova/MapForce2024/MapForceExamples/ShortPO.xml"),
               _T("CompletePO.xml"));
```

***Definir una gestión de errores personalizada***
Si necesita añadir su código personalizado de gestión de errores, modifique la instrucción `catch` en el método **_tmain**.

## 14.2.2 Cambiar tipo de datos de entrada/salida

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. Después de generar código Java o C# [913] 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.

 © 2019-2025 Altova GmbH

| | 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 78. 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 784, 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)`<br>`Altova.IO.FileOutput(string nombreArchivo)` |
| **Secuencias** | `Altova.IO.StreamInput(System.IO.Stream secuencia)`<br>`Altova.IO.StreamOutput(System.IO.Stream secuencia)` |

| | |
|---|---|
| **Cadenas** | `Altova.IO.StringInput(string contenido)`<br>`Altova.IO.StringOutput(System.Text.StringBuilder generadorCadenas)` |
| **Escritor/lector** | `Altova.IO.ReaderInput(System.IO.TextReader lector)`<br>`Altova.IO.WriterOutput(System.IO.TextWriter escritor)` |
| **Documentos DOM** | `Altova.IO.DocumentInput(System.Xml.XmlDocument documento)`<br>`Altova.IO.DocumentOutput(System.Xml.XmlDocument documento)` |

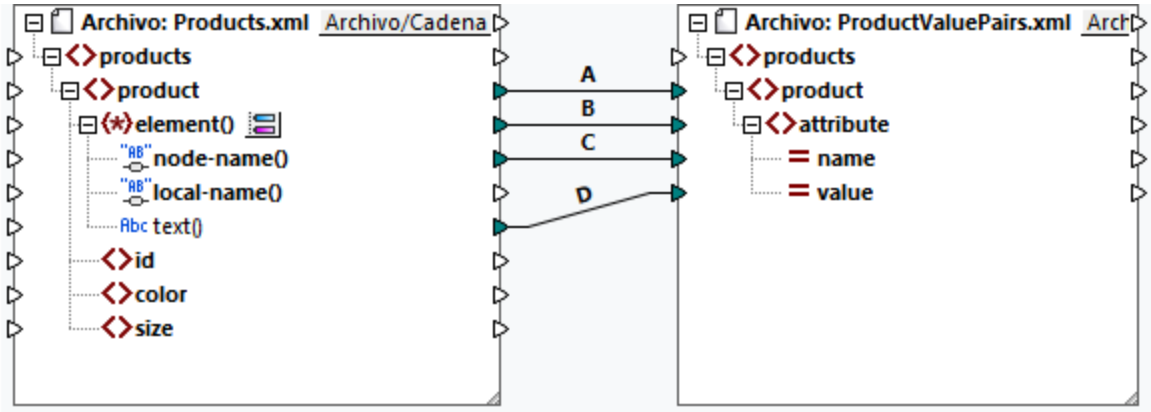
En Java, los tipos que puede usar como parámetros del método **`run`** son clases que se definen en el paquete `com.altova.io`. Las clases base son `com.altova.io.Input` y `com.altova.io.Output` respectivamente.

*Tipos Java*

| | |
|---|---|
| **Archivos** | `com.altova.io.FileInput(String nombreArchivo)`<br>`com.altova.io.FileOutput(String nombreArchivo)` |
| **Secuencias** | `com.altova.io.StreamInput(java.io.InputStream secuencia)`<br>`com.altova.io.FileOutput(String nombreArchivo)` |
| **Cadenas** | `Altova.IO.StringInput(string contenido)`<br>`com.altova.io.StringOutput()` |
| **Escritor/lector** | `com.altova.io.ReaderInput(java.io.Reader lector)`<br>`com.altova.io.WriterOutput(java.io.Writer escritor)` |
| **Documentos DOM** | `com.altova.io.DocumentOutput(org.w3c.dom.Document documento)`<br>`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**2025\MapForceExamples\Tutorials**.



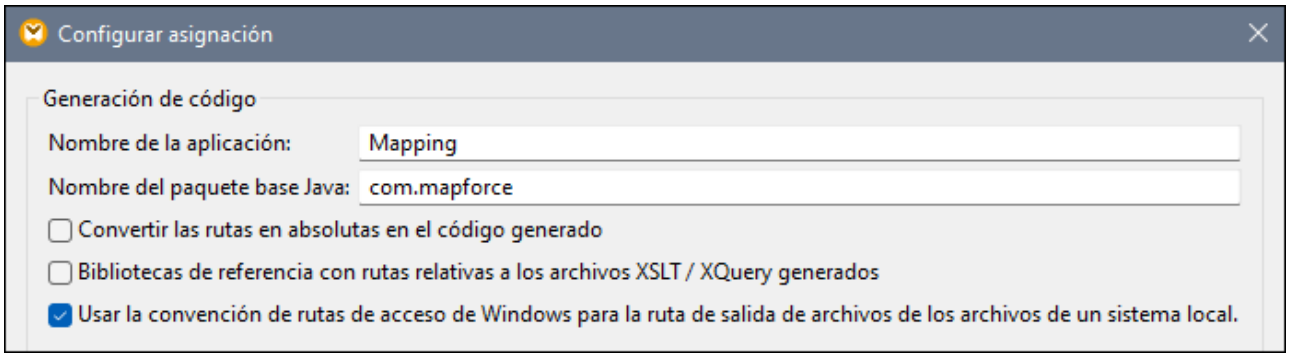

*ConvertProducts.mfd*

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:

 © 2019-2025 Altova GmbH

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");
//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());
```


© 2019-2025 Altova GmbH

```
   } 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*

```
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\MapForce2025\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.

## 14.2.3 Generar código 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 las opciones del generador de código 1131. 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:

 © 2019-2025 Altova GmbH

| 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. |
| *[SuEsquema]* | *`com.SuEsquema`* | 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.<br><br>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).<br><br>El código C# generado utiliza la biblioteca System.XML estándar de .NET como implementación DOM subyacente.<br><br>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 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* 939).

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 1020, 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 1020.

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++) 972
- Referencia de las clases generadas (C#) 988
- Referencia de las clases generadas (Java) 1004

Para más información sobre la conversión de tipos y otros detalles propios de cada lenguaje consulte estos apartados:

- Bibliotecas contenedoras de esquemas (C++) 931
- Bibliotecas contenedoras de esquemas (C#) 934
- Bibliotecas contenedoras de esquemas (Java) 937

## 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

 © 2019-2025 Altova GmbH

- 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[1020], pero las clases generadas no lo exigen
- Valores predeterminados y fijos para atributos. Están disponibles en el SPL[1020], 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`)

## 14.2.3.1 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 973 | |
| xs:duration | altova::Duration 976 | |
| 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 982
altova::meta::ComplexType 980

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.

 © 2019-2025 Altova GmbH

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] 983.

**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 986
[SuEsquema]::ElementoMiembro 987

**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 942.

## 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. |
| `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` 979
`altova::meta::ComplexType` 980
`altova::meta::Element` 981
`altova::meta::SimpleType` 982

Los métodos que devuelven una de las clases de metadatos devuelven un objeto NULL si no existe la propiedad correspondiente.

## 14.2.3.2 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` | |


© 2019-2025 Altova GmbH

| XML Schema | C# | Observaciones |
|---|---|---|
| xs:dateTime, date, time, gYearMonth, gYear, gMonthDay, gDay, gMonth | Altova.Types.DateTime 988 | |
| xs:duration | Altova.Types.Duration 994 | |
| 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 998
Altova.Xml.Meta.ComplexType 997

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 [SuEsquema].[Doc] 999 .

**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 1002
[SuTipoEsquema].ElementoMiembro 1003

**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` 996
`Altova.Xml.Meta.ComplexType` 997
`Altova.Xml.Meta.Element` 998
`Altova.Xml.Meta.SimpleType` 998

Las propiedades que devuelven una de las clases de metadatos devuelven un objeto NULL si no existe la propiedad correspondiente.

 © 2019-2025 Altova GmbH

### 14.2.3.3 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` 1005 | |
| `xs:duration` | `com.altova.types.Duration` 1010 | |
| `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.

#### 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` 1015
`com.altova.xml.meta.ComplexType` 1014

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]` 1016.

**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` 1019
`com.[SuEsquema].[SuTipoEsquema].ElementoMiembro` 1019

**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.

 © 2019-2025 Altova GmbH

com.altova.xml.meta.Attribute 1013
com.altova.xml.meta.ComplexType 1014
com.altova.xml.meta.Element 1014
com.altova.xml.meta.SimpleType 1015

Las propiedades que devuelven una de las clases de metadatos devuelven un objeto NULL si no existe la propiedad correspondiente.

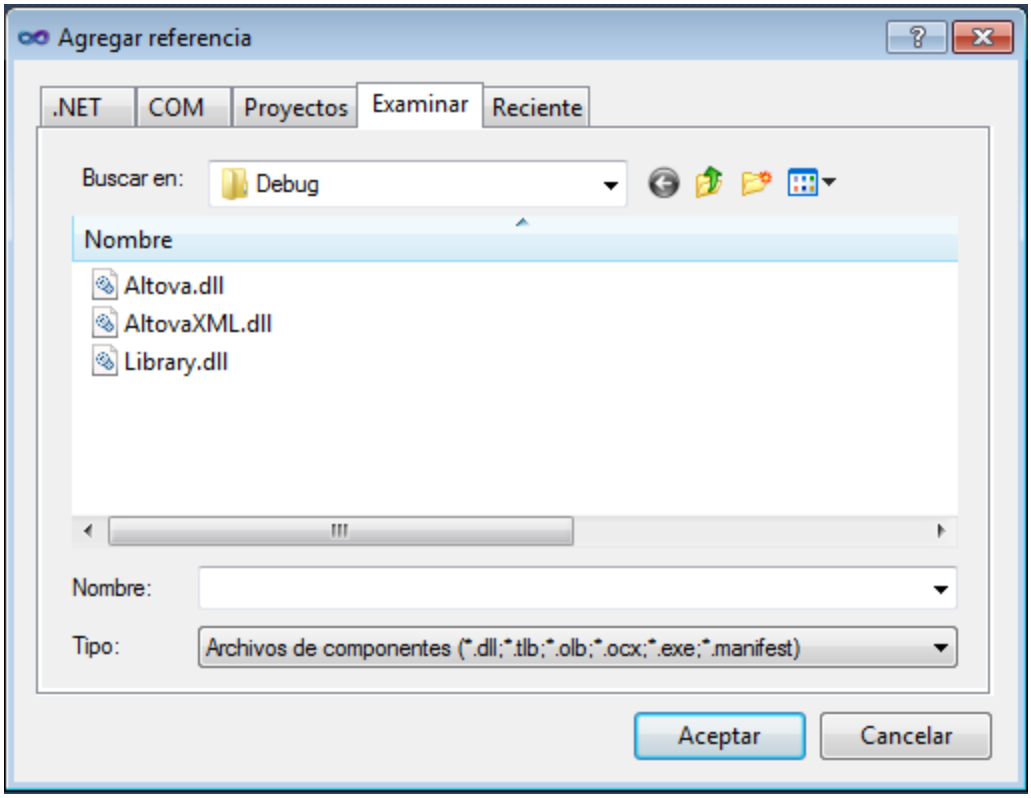
## 14.2.3.4 Integrar las bibliotecas contenedoras de esquemas

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. Enla pestana Examinar, busque las siguientes bibliotecas: **Altova.dll**, **AltovaXML.dll**, and **YourSchema.dll** ubicadas en el directorio de salida de los proyectos generados (p.ej. **bin\Debug**).



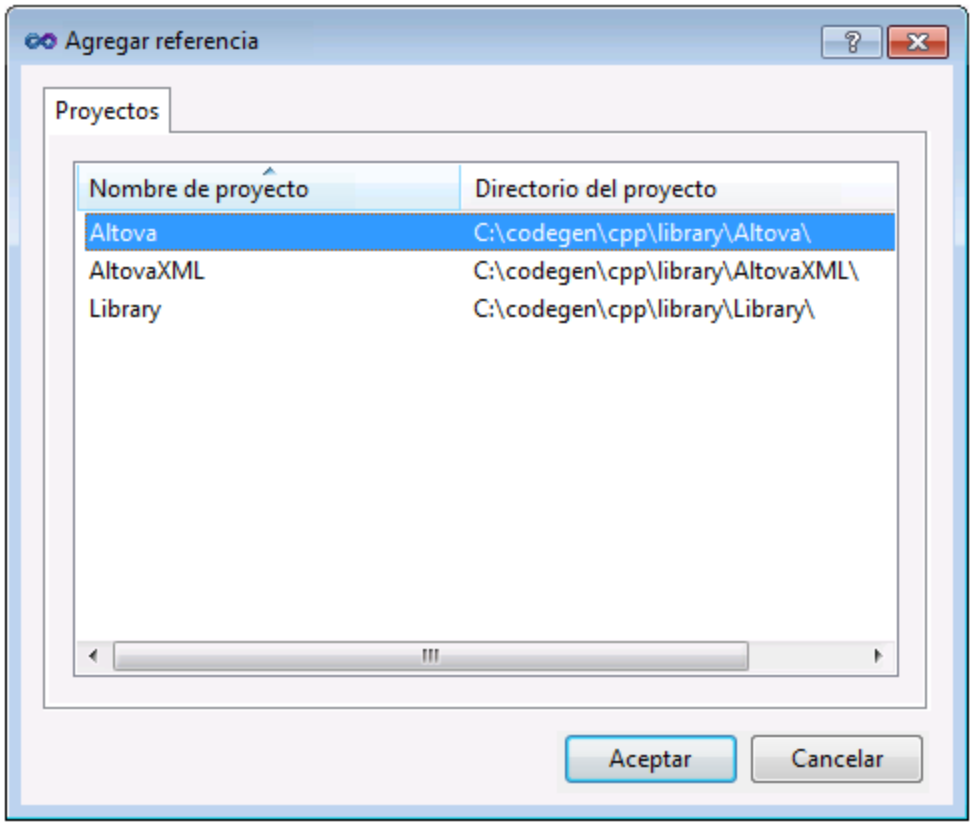
## C++

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. Haga clic con el botón derecho en su proyecto y seleccione el comando **Propiedades**. En el cuadro de diálogo "Páginas de propiedades" seleccione **Propiedades comunes | Framework y referencias** y después haga clic en **Agregar referencia nueva**. Ahora seleccione y añada cada uno de 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 (también se puede aplicar a su versión de Visual Studio):


© 2019-2025 Altova GmbH

- 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

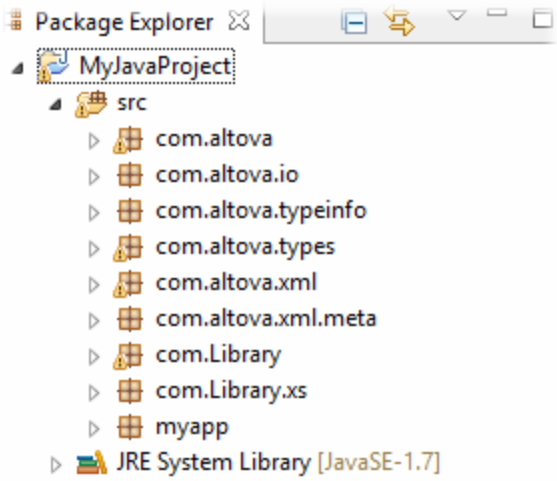
La opción de generar bibliotecas estáticas o dinámicas se encuentra entre las opciones de generación de código (véase Generación 1131).

## Java

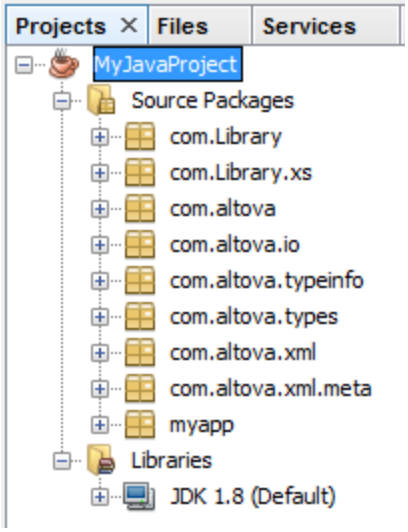
Puede integrar sus paquetes de Altova en su proyecto Java copiando el directorio **com** del código generado en el directorio que almacena los paquetes fuente de su proyecto Java (p.ej. **C:\Workspace\MyJavaProject\src**). Por ejemplo, imagine que generó código en el directorio **c:\codegen\java\library**. 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**.

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*

## 14.2.3.5 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.

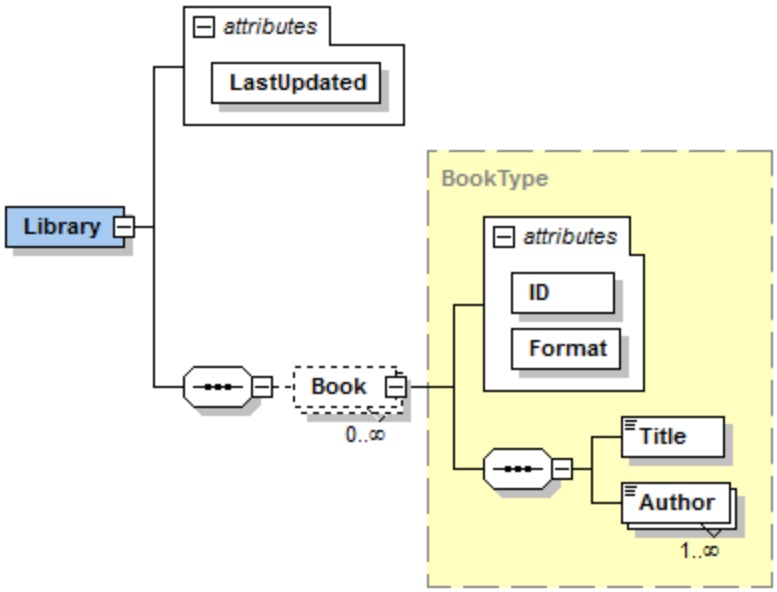
El esquema que se utiliza en este ejemplo describe una biblioteca de libros. Su definición completa aparece más abajo. Si quiere obtener los mismos resultados que se obtienen en esta sección, guarde este fragmento de código como **`Library.xsd`**. Este esquema se necesita para generar las bibliotecas 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>
```

 © 2019-2025 Altova GmbH

```
      <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>
```

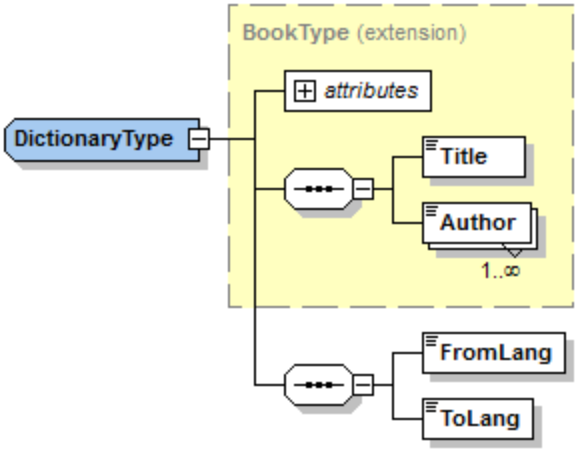
**Library** es un elemento raíz de un `complexType` cuya representación gráfica aparece a continuación (tomada de la vista Esquema de XMLSpy):




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 928).

 © 2019-2025 Altova GmbH

### 14.2.3.5.1 Leer y escribir documentos XML (C++)

Tras generarse el código a partir del esquema de ejemplo 942 se crea una aplicación de prueba C++ junto con varias bibliotecas Altova secundarias.

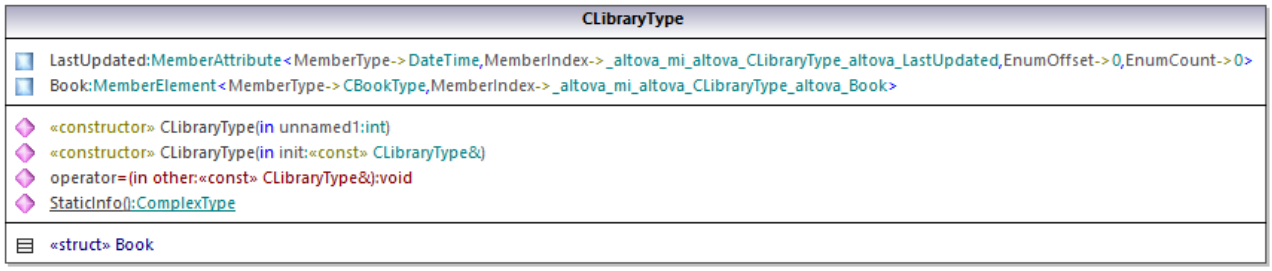
#### Información sobre las bibliotecas C++ generadas

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. 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]` 983).

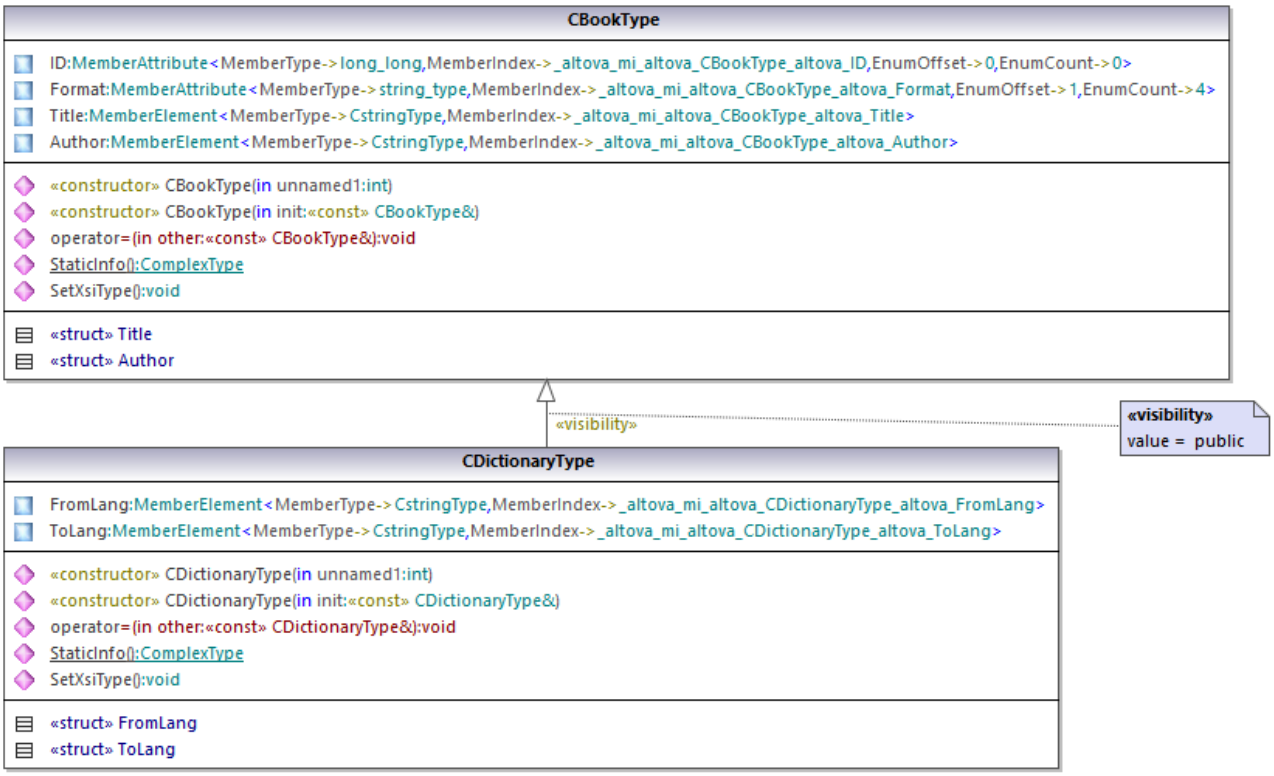
```
CDoc
Library:MemberElement<MemberType->CLibraryType,MemberIndex->_altova_mi_altova_CDoc_altova_Library>
«constructor» CDoc(in unnamed1:int)
«constructor» CDoc(in init:«const» CDoc&)
operator=(in other:«const» CDoc&):void
StaticInfo():ComplexType
SetXsiType():void
LoadFromFile(in fileName:«const» 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` 986 y `[SuEsquema]::ElementoMiembro` 987.

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.).




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

 © 2019-2025 Altova GmbH

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* 939).

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

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();
```


© 2019-2025 Altova GmbH

```
   // 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;

      // 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.

### 14.2.3.5.2 Leer y escribir documentos XML (C#)

Tras generarse el código a partir del esquema de ejemplo 942 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. 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]` 999).

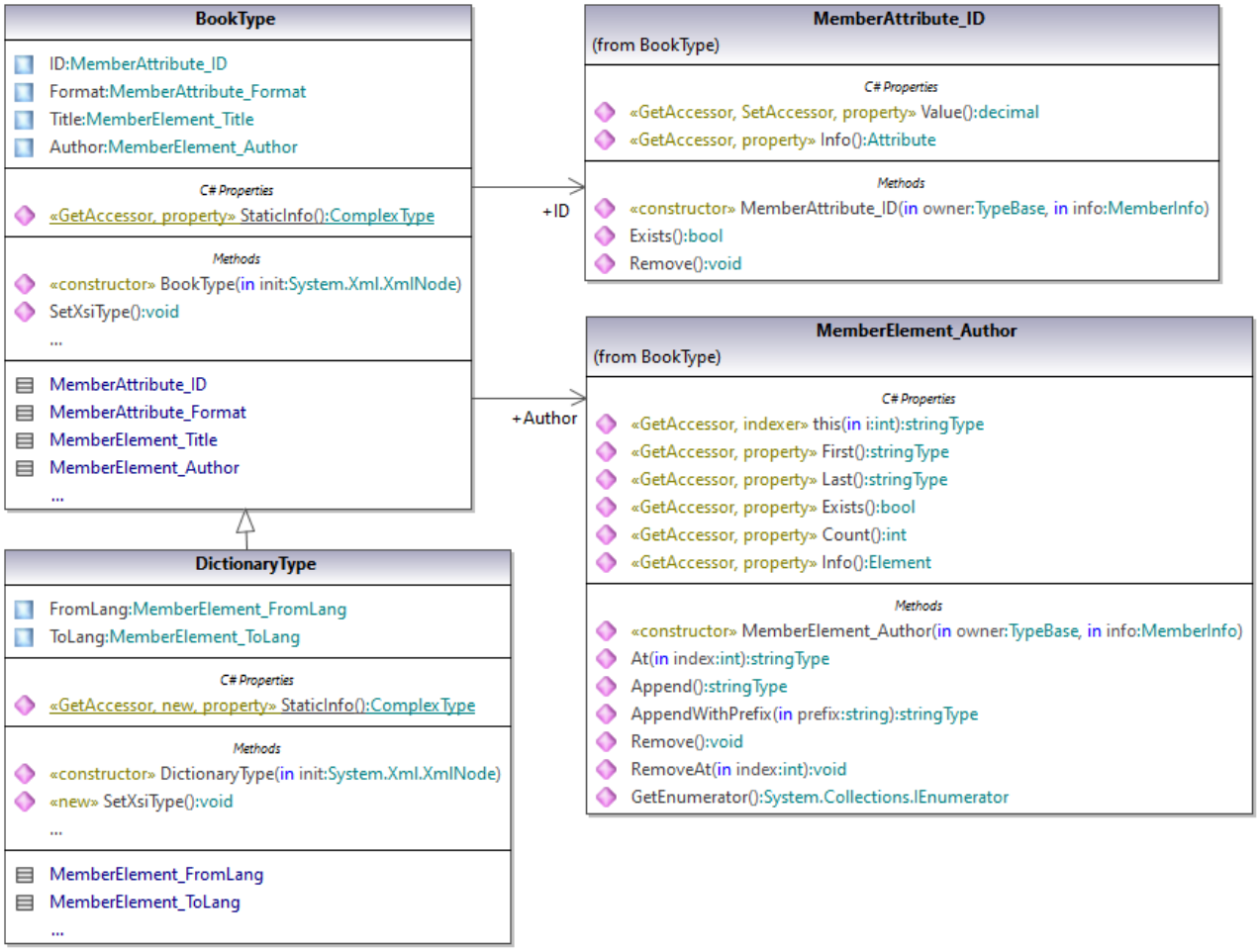
```
Doc2
Library:MemberElement_Library
C# Properties
«GetAccessor, property» StaticInfo():ComplexType
Methods
LoadFromFile(in filename:string):Doc2
LoadFromString(in xmlstring:string):Doc2
LoadFromBinary(in binary:byte[*]):Doc2
SaveToFile(in filename:string, in prettyPrint:bool):void
SaveToFile(in filename:string, 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#) 934, 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` 1002 y `[SuTipoEsquema].MemberElement` 1003.

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`.


© 2019-2025 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* 939).

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**).


© 2019-2025 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()`.

 © 2019-2025 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:

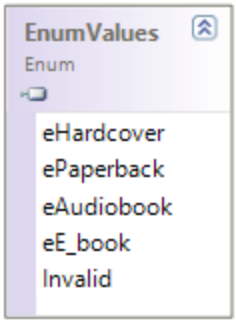
```
// Anexar el elemento raíz a la biblioteca
LibraryType root = doc.Library.Append();
```

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` 988 o `Altova.Types.Duration` 994 (bien desde `System.DateTime` o utilizando partes como horas y minutos como en `Altova.Types.DateTime` 988 y `Altova.Types.Duration` 994).
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` 988 o `Altova.Types.Duration` 994.
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` 988 y `Altova.Types.Duration` 994.

## 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";
```


© 2019-2025 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);

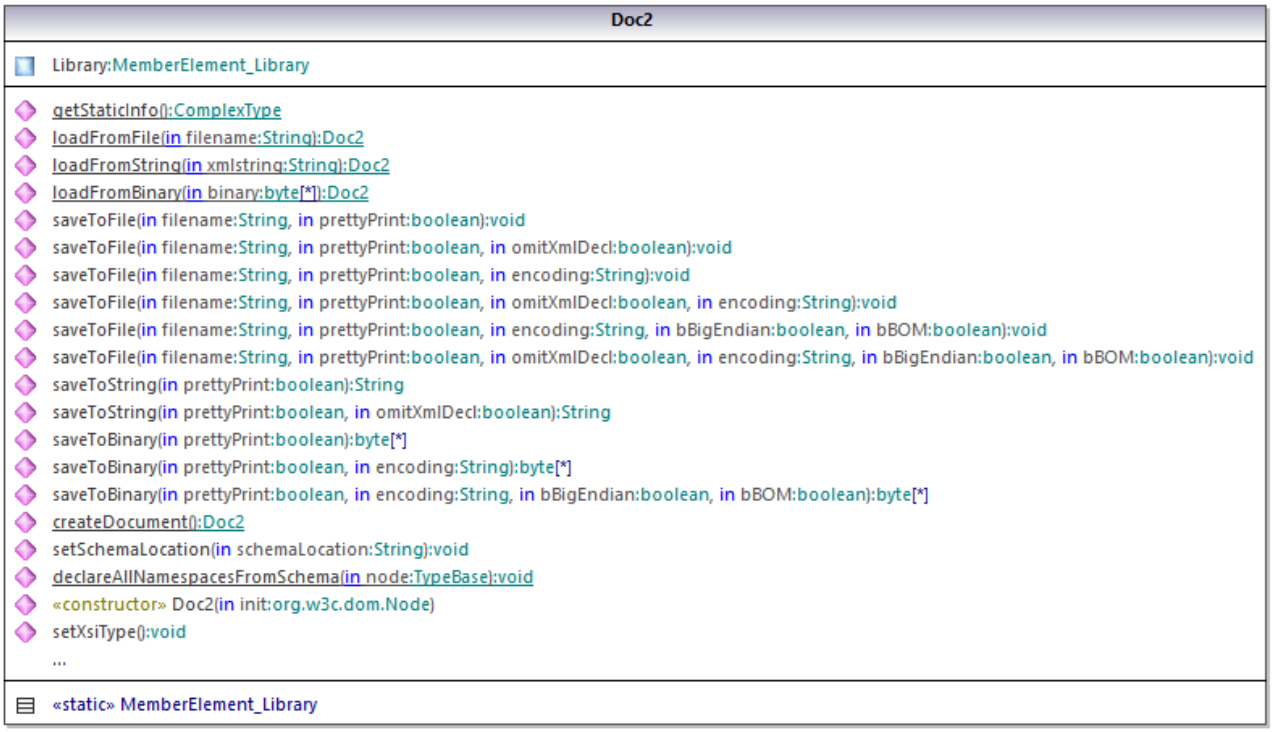
          // Si este libro tiene DictionaryType
          if (namespaceUri == "http://www.nanonull.com/LibrarySample" &&
xsiTypeValue.Equals(prefix + ":DictionaryType"))
          {
             // generar campos adicionales
             DictionaryType dictionary = new DictionaryType(book.Node);
             Console.WriteLine("Language from: " + dictionary.FromLang.First.Value);
             Console.WriteLine("Language to: " + dictionary.ToLang.First.Value);
          }
          else
          {
              throw new Exception("Unexpected book type");
          }
      }
```

### 14.2.3.5.3 Leer y escribir documentos XML (Java)

Tras generarse el código a partir del esquema de ejemplo 942 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. 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]` 1016.

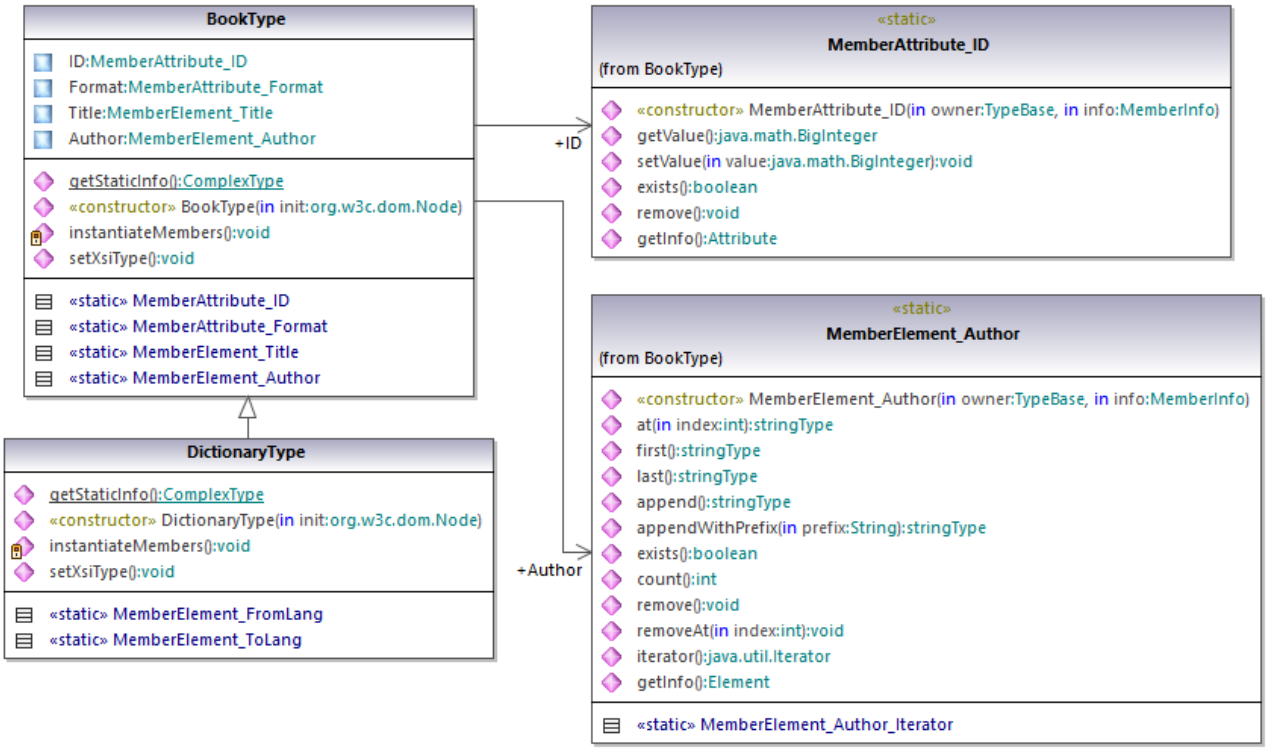



El miembro `Library` de la clase `Doc2` representa la raíz real del documento.

De acuerdo con las reglas de generación de código mencionadas en el apartado Bibliotecas contenedoras de esquemas (Java) 937, 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` 1019 y `com.[SuEsquema].[SuTipoEsquema].MemberElement` 1019.

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`.

 © 2019-2025 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* 939).

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"
```


© 2019-2025 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()`.

 © 2019-2025 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` 1005 o `com.altova.types.Duration` 1010.
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` 1005 o `com.altova.types.Duration` 1010.
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` 1005 y `com.altova.types.Duration` 1010.

## 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

 © 2019-2025 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.");
            }
         }
```

## 14.2.3.6 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>
```

 © 2019-2025 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 [928]. 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++) [968]
   - Espacios de nombres y prefijos XML (C#) [970]
   - Espacios de nombres y prefijos XML (Java) [971]

### 14.2.3.6.1 Espacios de nombres y prefijos XML (C++)

Una vez haya generado código a partir del esquema de ejemplo [965] 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() [983]. 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

 © 2019-2025 Altova GmbH

**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() 985. 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(). 987 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() 987 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 965. 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();
}
```

### 14.2.3.6.2 Espacios de nombres y prefijos XML (C#)

Una vez haya generado código a partir del esquema de ejemplo 965 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() 983: 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() 1001: 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() 1003: 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() 1003 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 965. 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
```

 © 2019-2025 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);
}
```

### 14.2.3.6.3 Espacios de nombres y prefijos XML (Java)

Una vez haya generado código a partir del esquema de ejemplo 965 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() 1016: 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() 1017: 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() 1019: 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() 987 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[965]. 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);
}
```

## 14.2.4 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[928]). Puede integrar estas clases en su código para leer, modificar y escribir documentos XML.

**Nota:** El código generado puede incluir otras clases compatibles que no se enumeran en este apartado y que pueden cambiar en el futuro.

 © 2019-2025 Altova GmbH

### 14.2.4.1 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. |
| `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. |

| Nombre | Descripción |
|---|---|
| `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()
{
   // 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
```


© 2019-2025 Altova GmbH

```
   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; }
}
```

## 14.2.4.2 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* 978). |
| `Duration(const YearMonthDuration& ym)` | Inicializa una instancia nueva de la clase `Duration` en la duración definida por el argumento `ym` (véase *altova::YearMonthDuration* 979). |
| `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* 979 *y altova::DayTimeDuration* 978). |

### 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* 978). |
| `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. |


© 2019-2025 Altova GmbH

| Nombre | Descripción |
|---|---|
| `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. |
| `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* 979). |
| `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;
```

```
}
```

### 14.2.4.3 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. |


© 2019-2025 Altova GmbH

## 14.2.4.4 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. |

## 14.2.4.5 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.

### 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. |

## 14.2.4.6 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. |


© 2019-2025 Altova GmbH

| 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. |

## 14.2.4.7 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. |

### Operadores

| Nombre | Descripción |
|---|---|
| `bool operator()` | Devuelve false si este no es el elemento NULL. |

| Nombre | Descripción |
|---|---|
| `bool operator!()` | Devuelve true si este es el elemento NULL. |

## 14.2.4.8 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. |
| `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:<br>• Whitespace_Unknown<br>• Whitespace_Preserve<br>• Whitespace_Replace<br>• Whitespace_Collapse |


© 2019-2025 Altova GmbH

### Operadores

| Nombre | Descripción |
|---|---|
| `bool operator()` | Devuelve true si este no es el SimpleType NULL. |
| `bool operator!()` | Devuelve true si este es el SimpleType NULL. |

## 14.2.4.9 [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(ElementType& 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. |
| `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,` | Guarda un documento XML en una matriz de bytes (con formato pretty-print opcional) con la codificación especificada. |

| Nombre | Descripción |
|---|---|
| `const string_type & encoding)` | |
| `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 con la codificación especificada. 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.<br><br>Este método sólo está disponible si generó código para la biblioteca XML Xerces3 (véase Generación 1131). |
| `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.<br><br>Este método sólo está disponible si generó código para la biblioteca XML Xerces3 (véase Generación 1131). |
| `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. |


© 2019-2025 Altova GmbH

| Nombre | Descripción |
|---|---|
| | Este método sólo está disponible si generó código para la biblioteca XML Xerces3 (véase Generación 1131). |
| `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.<br><br>Este método sólo está disponible si generó código para la biblioteca XML Xerces3 (véase Generación 1131). |
| `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. |

## 14.2.4.10 [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 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.<br><br>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 |

| Nombre | Descripción |
|---|---|
| | argumento está vacío, el método crea o sobrescribe la declaración de espacio de nombres predeterminada del elemento.<br><br>Por ejemplo, imagine que el documento XML tiene un elemento XML llamado "purchase". Si llama a<br><br>`purchase.DeclareNamespace(_T("ord"), _T("http://OrderTypes"));`<br><br>entonces el documento XML se convierte en<br><br>`<purchase xmlns:ord="http://OrderTypes" />`<br><br>Otro ejemplo. Si llama a:<br><br>`purchase.DeclareNamespace(_T(""), _T("http://OrderTypes"));`<br><br>entonces el documento XML se convierte en<br><br>`<purchase xmlns="http://OrderTypes" />`<br><br>**Nota:** el espacio de nombres declarado se usa al anexar elementos secundarios o atributos posteriormente, conforme a estas normas:<br><br>1. Si el espacio de nombres secundario es el predeterminado, use un prefijo vacío.<br>2. Si el espacio de nombres secundario es igual que el del elemento matriz, use el prefijo matriz.<br>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.<br>4. Si no se encuentra ningún prefijo para el espacio de nombres del elemento, entonces use un prefijo vacío. |

## 14.2.4.11 [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.


© 2019-2025 Altova GmbH

### 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` 979). |
| `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. |

## 14.2.4.12 [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.

### 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 965. |
| `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. |

| Nombre | Descripción |
|---|---|
| `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` 981). |
| `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. |

## 14.2.5 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 928). Puede integrar estas clases en su código para leer, modificar y escribir documentos XML.

**Nota:** El código generado puede incluir otras clases compatibles que no se enumeran en este apartado y que pueden cambiar en el futuro.

### 14.2.5.1 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. |


© 2019-2025 Altova GmbH

| | Nombre | Descripción |
|---|---|---|
| | `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`. |
| | `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. |

| | Nombre | Descripción |
|---|---|---|
| | `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`:<br><br>`2015-01-01T23:23:23`<br>`2015-01-01`<br>`2015-11`<br>`23:23:23`<br><br>Si la cadena no se puede convertir en un objeto `DateTime`, entonces se genera una excepción.<br><br>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 992.<br><br>Si la cadena no se puede convertir en un objeto `DateTime`, entonces se genera una excepción.<br><br>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 992. |

## 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. |


© 2019-2025 Altova GmbH

| Nombre | Descripción |
|---|---|
| >= | 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)
   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));
}
```

### 14.2.5.2 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`<br>(si la fecha es el día 12 del mes) |
| **W3_gMonth** | Da al valor el formato de gMonth de esquema. | `--11`<br>(si el mes es noviembre) |
| **W3_gMonthDay** | Da al valor el formato de gMonthDay de esquema. | `--11-12`<br>(si la fecha es el día 12 de noviembre) |
| **W3_gYear** | Da al valor el formato de gYear de esquema. | `2015`<br>(si el año es 2015) |


© 2019-2025 Altova GmbH

| Valor | Descripción | Ejemplo |
|---|---|---|
| **W3_gYearMonth** | Da al valor el formato de gYearMonth de esquema. | `2015-11`<br>(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` |

## 14.2.5.3 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. |


© 2019-2025 Altova GmbH

| | 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:<br>**DURATION** Analizar duración asumiendo que existen las partes año, mes, día y duración.<br>**YEARMONTH** Analizar duración asumiendo que solo existen las partes año y mes.<br>**DAYTIME** Analizar duración asumiendo que solo existen las partes día y hora.<br>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);
```

```
   // 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);
}
```

### 14.2.5.4 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.


© 2019-2025 Altova GmbH

### 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. |

## 14.2.5.5 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. |

| | 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. |

## 14.2.5.6 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. |

## 14.2.5.7 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.


© 2019-2025 Altova GmbH

## 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. |

## 14.2.5.8 [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.

## 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. |


© 2019-2025 Altova GmbH

| | 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. |

## 14.2.5.9 [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 |

| | Nombre | Descripción |
|---|---|---|
| | | 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.<br><br>Por ejemplo, imagine que el documento XML tiene un elemento XML llamado "purchase". Si llama a<br><br>`purchase.DeclareNamespace("ord", "http://OrderTypes");`<br><br>entonces el documento XML se convierte en<br><br>`<purchase xmlns:ord="http://OrderTypes" />`<br><br>Otro ejemplo. Si llama a:<br><br>`purchase.DeclareNamespace("", "http://OrderTypes");`<br><br>entonces el documento XML se convierte en<br><br>`<purchase xmlns="http://OrderTypes" />`<br><br>**Nota:** el espacio de nombres declarado se usa al anexar elementos secundarios o atributos posteriormente, conforme a estas normas:<br><br>1. Si el espacio de nombres secundario es el predeterminado, use un prefijo vacío.<br>2. Si el espacio de nombres secundario es igual que el del elemento matriz, use el prefijo matriz.<br>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.<br>4. Si no se encuentra ningún prefijo para el espacio de nombres del elemento, entonces use un prefijo vacío. |

## 14.2.5.10 [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.


© 2019-2025 Altova GmbH

### 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`* 996). |
| | `AttributeType Value` | Establece u obtiene el valor del atributo. |

## 14.2.5.11 [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. |
| | `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 |

| | Nombre | Descripción |
|---|---|---|
| | | ejemplo consulte Ejemplo: orden de compra 965. |
| | `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` 998 ). |
| | `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). |

## 14.2.6 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 928). Puede integrar estas clases en su código para leer, modificar y escribir documentos XML.


© 2019-2025 Altova GmbH

**Nota:** El código generado puede incluir otras clases compatibles que no se enumeran en este apartado y que pueden cambiar en el futuro.

## 14.2.6.1 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`:<br><br>`2015-11-24T12:54:47.969+01:00` |

| | Nombre | Descripción |
|---|---|---|
| | | `2015-11-24T12:54:47`<br>`2015-11-24` |
| | **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:<br><br>`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:<br><br>`-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:<br>`CalendarBase.TZ_MISSING` No está definida la diferencia de zona horaria.<br>`CalendarBase.TZ_UTC` La zona horaria es UTC.<br>`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 |


© 2019-2025 Altova GmbH

| | Nombre | Descripción |
|---|---|---|
| | | establecer la zona horaria en UTC (hora universal coordinada). Los valores válidos para el argumento nHasTZ son:<br>`CalendarBase.TZ_MISSING` Establecer la diferencia de zona horaria como indefinida.<br>`CalendarBase.TZ_UTC` Establecer la zona horaria en UTC.<br>`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:<br>`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:
```

 © 2019-2025 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());
}
```

## 14.2.6.2 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:<br>**ParseType.DAYTIME** Puede utilizarse cuando la cadena **s** está compuesta por cualquiera de estos: días, horas, minutos, segundos, partes fraccionarias de |


© 2019-2025 Altova GmbH

| | Nombre | Descripción |
|---|---|---|
| | | segundo. Por ejemplo - P4DT4H4M4.774S.<br>**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.<br>**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. |
| | `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. |

| | Nombre | Descripción |
|---|---|---|
| | `String toString()` | Devuelve la representación de cadena de la instancia `Duration`. Por ejemplo:<br><br>`-P4DT4H4M4.774S` |
| | `String toYearMonthString()` | Devuelve la representación de cadena de la parte YearMonth de la instancia `Duration` actual. Por ejemplo:<br><br>`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);
   Duration dr7 = Duration.parse("-P4DT4H4M4.774S", ParseType.DAYTIME);
}
```

Este ejemplo muestra cómo se puede obtener y establecer el valor de objetos `Duration`:


© 2019-2025 Altova GmbH

```
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());
}
```

### 14.2.6.3 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. |

| | Nombre | Descripción |
|---|---|---|
| | `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. |

## 14.2.6.4 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. |

## 14.2.6.5 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.


© 2019-2025 Altova GmbH

### 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. |

## 14.2.6.6 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. |
| | `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. |

| | Nombre | Descripción |
|---|---|---|
| | `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` |

## 14.2.6.7 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. |
| 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. |


© 2019-2025 Altova GmbH

| | Nombre | Descripción |
|---|---|---|
| | **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** 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** setSchemaLocation(String schemaLocation) | Agrega un atributo xsi:schemaLocation o xsi:noNamespaceSchemaLocation al elemento raíz. Es obligatorio que ya exista un elemento raíz. |

## 14.2.6.8 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.<br><br>Por ejemplo, imagine que el documento XML tiene un elemento XML llamado "purchase". Si llama a<br><br>`purchase.declareNamespace("ord", "http://OrderTypes");`<br><br>entonces el documento XML se convierte en<br><br>`<purchase xmlns:ord="http://OrderTypes" />`<br><br>Otro ejemplo. Si llama a:<br><br>`purchase.declareNamespace("", "http://OrderTypes");`<br><br>entonces el documento XML se convierte en<br><br>`<purchase xmlns="http://OrderTypes" />`<br><br>**Nota:** el espacio de nombres declarado se usa al anexar elementos secundarios o atributos posteriormente, conforme a estas normas:<br><br>1. Si el espacio de nombres secundario es el predeterminado, use un prefijo vacío.<br>2. Si el espacio de nombres secundario es igual que el del elemento matriz, use el prefijo matriz.<br>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. |


© 2019-2025 Altova GmbH

| | Nombre | Descripción |
|---|---|---|
| | | 4. Si no se encuentra ningún prefijo para el espacio de nombres del elemento, entonces use un prefijo vacío. |

## 14.2.6.9 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` 1013 ). |
| | `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 valores a este método para establecer el valor. |
| | **void** `setValue(AttributeType value)` | Establece el valor del atributo. |

## 14.2.6.10 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 965. |
| | `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` 1014 ). |
| | `MemberType getValue()` | Obtiene el contexto del elemento (solo se genera si el elemento puede tener contenido simple o mixto). |
| | `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). |

## 14.2.7 Referencia de la SPL

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

 © 2019-2025 Altova GmbH

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 **spl** de la aplicación. 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 MapForce. 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 del lenguaje de destino (es decir, los archivos de código no compilados), 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.

*Nota sobre los 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 ello siga estos pasos:

1. Navegue hasta el directorio de instalación del programa, p.ej. `C:\Archivos de programa\Altova\MapForce2025`.
2. En el subdirectorio **spl** busque el directorio que corresponde al lenguaje de programación, p.ej. **..\spl\java**.
3. Abra el archivo **settings.spl** e inserte una línea nueva en la sección de `reserva`, p.ej. **reserve "miPalabraReservada"**.
4. Vuelva a generar el código de programa.

### 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]
```

## 14.2.7.1 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 "**:**".

Por ejemplo:

```
[$x = 42
$x = $x + 1]
```

o

```
[$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 salida1027).

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: **]**.

## 14.2.7.2 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 ... to ...

```
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
```

 © 2019-2025 Altova GmbH

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 ... to ...

**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.

Ejemplo:

```
map type "float" to "CSchemaFloat"
```

## default ... is ...

**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 |

| Nombre de configuración | Explicación |
|---|---|
| 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

**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

incluye el archivo indicado como SPL de origen. Esto permite dividir su plantilla en varios archivos para editarlos y gestionarlos más fácilmente.

**include** *nombrearchivo*

Ejemplo:

```
include "Module.cpp"
```

## 14.2.7.3 Variables

Los archivos SPL importantes exigen el uso de variables. Algunas variables vienen predefinidas 1025 por el generador de código y se pueden crear variables nuevas con solo asignarles valores.

El carácter **$** se usa cuando se **declara** o **usa** una variable y un nombre de variable siempre tiene el prefijo **$**. 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

 © 2019-2025 Altova GmbH

- string
- objeto, que viene dado por MapForce
- iterador, ver instrucción foreach 1030

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 equivalen a una línea nueva y a una tabulación, respectivamente. **\"** es una comilla doble literal y **** es una barra diagonal inversa. Las constantes de cadena también puede abarcar varias líneas.

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**, a las que puede acceder con el operador **.** 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**.

## 14.2.7.4 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.

| Nombre | Tipo | Descripción |
|---|---|---|
| $schematype | integer | 1 para DTD, 2 para XML Schema |
| $TheLibrary | Biblioteca 1034 | La biblioteca que se deriva del esquema XML o de la DTD |

| Nombre | Tipo | Descripción |
|---|---|---|
| $module | string | Nombre del esquema de origen sin la extensión |
| $outputpath | string | 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 | integer | 1 para MSXML, 2 para Xerces |
| $libtype | integer | 1 para static LIB, 2 para DLL |
| $mfc | booleano | True si se marcó la casilla Compatibilidad con MFC |
| $VSVersion | integer | Especifica la versión de Visual Studio. Valores válidos:<br>**0** Sin proyecto de Visual Studio<br>**2010** Visual Studio 2010<br>**2013** Visual Studio 2013<br>**2015** Visual Studio 2015<br>**2017** Visual Studio 2017<br>**2019** Visual Studio 2019 |

Para generación de código C# solamente:

| Nombre | Tipo | Descripción |
|---|---|---|
| $VSVersion | integer | Especifica la versión de Visual Studio. Valores válidos:<br>**0** Sin proyecto de Visual Studio<br>**2010** Visual Studio 2010<br>**2013** Visual Studio 2013<br>**2015** Visual Studio 2015<br>**2017** Visual Studio 2017<br>**2019** Visual Studio 2019 |


© 2019-2025 Altova GmbH

## 14.2.7.5 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

```
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

```
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 ... to ...

```
filecopy origen to destino
```

copia el archivo de origen al archivo de destino, sin interpretarlo.

Ejemplo:

```
filecopy "java/mapforce/mapforce.png" to $outputpath & "/" & $JavaPackageDir &
"/mapforce.png"
```

## 14.2.7.6 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 |


© 2019-2025 Altova GmbH

<= 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

## 14.2.7.7 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 1028 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]
```

### 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.

## 14.2.7.8 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 iterator in colección
   instrucciones
siguiente
```

Ejemplo n°1:

```
[foreach $class in $classes
   if not $class.IsInternal
      ]  class [=$class.Name];
[  endif
next]
```

Ejemplo n°2:

```
[foreach $i in 1 To 3
      Write "// Step " & $i & "\n"
      ‘ Do some work
next]
```

En la primera línea:

**$classes** es el **objeto global**[1025] 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]
```

## 14.2.7.9 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 recurrente (las subrutinas se pueden llamar a sí mismas)

### 14.2.7.9.1 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:

- 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
]
```


© 2019-2025 Altova GmbH

### 14.2.7.9.2 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")
```

### 14.2.7.9.3 Ejemplo de subrutinas

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
]
```

## 14.2.7.10 Tipos integrados

Esta sección describe las propiedades de los tipos integrados usados en las variables predefinidas 1025 que describen el esquema analizado.

### 14.2.7.10.1 Library (biblioteca)

Este objeto representa toda la biblioteca generada a partir del esquema XML o de la DTD.

| Propiedad | Tipo | Descripción |
|---|---|---|
| SchemaNamespaces | Colección Namespace 1034 | Los espacios de nombres de esta biblioteca |
| SchemaFilename | string | Nombre del XSD o DTD del que se deriva esta biblioteca |
| SchemaType | integer | 1 para DTD, 2 para XML Schema |
| Guid | string | Un ID único global |
| CodeName | string | El nombre de la biblioteca generada (derivado del nombre del archivo de esquema) |

### 14.2.7.10.2 Namespace (espacio de nombres)

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) |


© 2019-2025 Altova GmbH

| Propiedad | Tipo | Descripción |
|---|---|---|
| LocalName | string | Prefijo del espacio de nombres |
| NamespaceURI | string | URI del espacio de nombres |
| Tipos | Colección Type 1035 | Todos los tipos incluidos en este espacio de nombres |
| Biblioteca | Biblioteca 1034 | La biblioteca que contiene este espacio de nombres |

### 14.2.7.10.3 Type (tipo)

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 | El nombre original en el esquema |
| Namespace | Namespace 1034 | Espacio de nombres que contiene este tipo |
| Attributes | Colección Member 1036 | Atributos que contiene este tipo* |
| Elements | Colección Member 1036 | 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 | Tipo | El tipo base de este tipo (si IsDerived es true) |

| Propiedad | Tipo | Descripción |
|---|---|---|
| IsDocumentRootType | boolean | True si este tipo representa al documento propiamente dicho |
| Biblioteca | Biblioteca 1034 | 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 1037 | El NativeBinding para este tipo |
| Facets | Facetas 1038 | Las facetas de este tipo |
| Espacios en blanco | 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.

### 14.2.7.10.4 Member (miembro)

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 | Nombre para el código generado (derivado del nombre local o de la declaración de objeto primario) |
| LocalName | string | El nombre original en el esquema. Aparece vacía para el miembro especial que representa el contenido de texto de tipos complejos. |


© 2019-2025 Altova GmbH

| Propiedad | Tipo | Descripción |
|---|---|---|
| NamespaceURI | string | El URI de espacio de nombres de este elemento/atributo dentro de documentos/secuencias de instancia XML. |
| DeclaringType | Tipo 1035 | El tipo que originalmente declara el miembro (igual a containingType para miembros no heredados) |
| ContainingType | Tipo 1035 | El tipo del cual este es miembro |
| DataType | Tipo 1035 | El tipo de datos del contenido de este miembro |
| Biblioteca | Biblioteca 1034 | La biblioteca que contiene el DataType de este miembro |
| IsAttribute | boolean | True para atributos, false 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 |

### 14.2.7.10.5 NativeBinding (tipo nativo)

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 |

| Propiedad | Tipo | Descripción |
|---|---|---|
| ValueHandler | string | Instancia de clase de formateador |

### 14.2.7.10.6 Facets (facetas)

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 | Tipo | El tipo en el que están declaradas las facetas |
| Espacios en blanco | 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 |
| Lista | 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 | Tipo 1035 | El tipo en el que se declara esta faceta |
| FacetCheckerName | string | El nombre del Facet Checker (de la asignación schemafacet) |


© 2019-2025 Altova GmbH

| Propiedad | Tipo | Descripción |
|---|---|---|
| FacetValue | string o integer | El valor real de esta faceta |

# 15 API COM de MapForce

La API basada en COM de MapForce permite a otras aplicaciones utilizar las funciones de MapForce. Esto permite automatizar una amplia gama de tareas, por ejemplo, la definición de diferentes opciones y la generación de resultados, entre muchas otras.

La API COM de MapForce sigue las especificaciones comunes para servidores de automatización establecidas por Microsoft. MapForce se registra automáticamente como objeto servidor COM durante la instalación. Una vez registrado el objeto servidor COM, podrá invocarlo desde aplicaciones y lenguajes de programación que dispongan de funciones de programación para llamadas COM. Esto hace posible acceder a la API MapForce no sólo desde entornos de desarrollo que utilicen .NET, C++ y Visual Basic, sino también desde lenguajes de scripting como JScript y VBScript.

## Documentación de la API de MapForce

Puede acceder a la documentación de la API de MapForce aquí:
https://www.altova.com/manual/en/api/mapforceapi/index.html.


© 2019-2025 Altova GmbH

# 16 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#, and VB.NET. 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/es/components/download). Ensure that you install MapForce first, and then the MapForce Integration Package. Other prerequisites apply, depending on language and platform (see Prerequisites 1042).

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 1046 and Integration at Document Level 1049 describe the key steps at these respective levels. The ActiveX Integration Examples 1052 section provides examples in C# and Java. Looking through these examples will help you to make the right decisions quickly. The Object Reference 1082 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 886.

# 16.1 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/es/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 1044).

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 1052.

## 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):

 © 2019-2025 Altova GmbH

- MapForce
- The MapForce Integration Package
- The custom integration code or application.

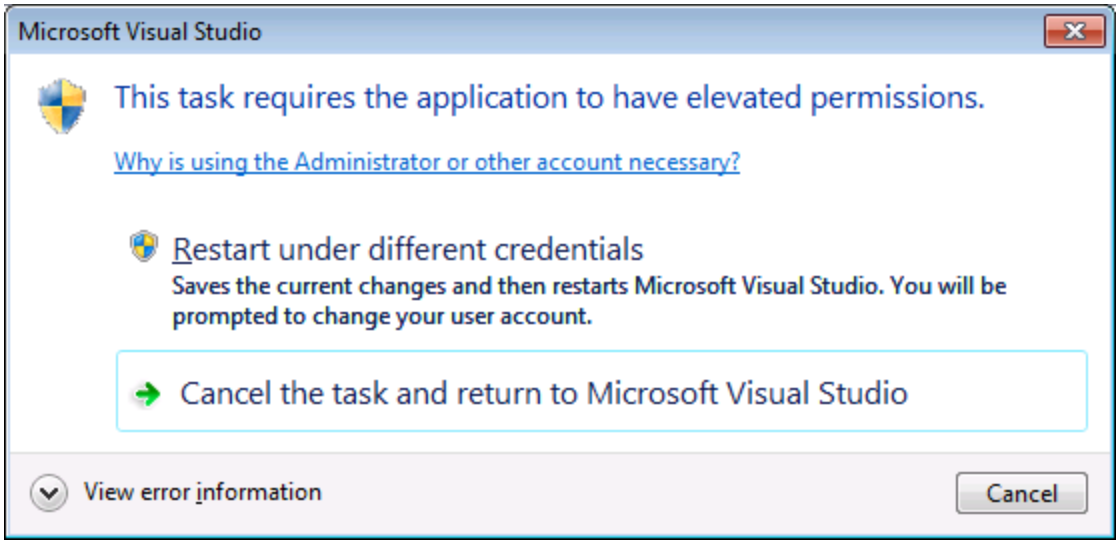
## 16.2 Adding the ActiveX Controls to the Toolbox

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.

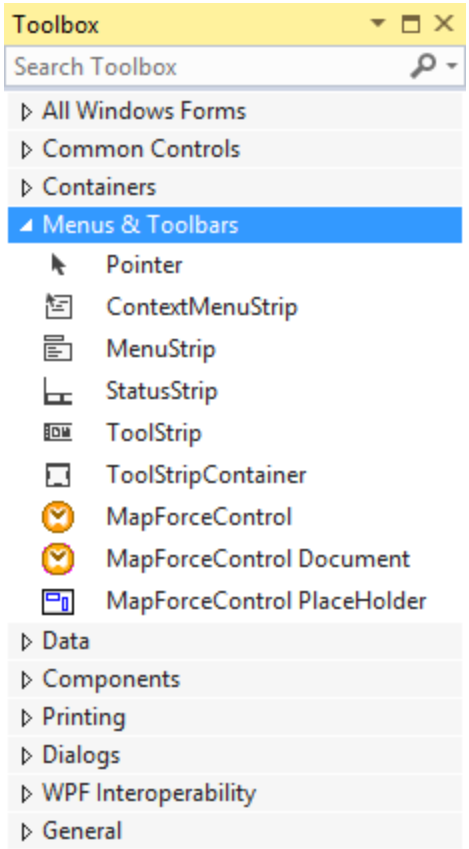
In case the controls above are not available, follow the steps below:

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 1042.
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.


© 2019-2025 Altova GmbH




**Note:** For an application-level integration, only the **MapForceControl** ActiveX control is used (see Integration at Application Level 1046). The **MapForceControl Document** and **MapForceControl Placeholder** controls are used for document-level integration (see Integration at Document Level 1049).

## 16.3 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.

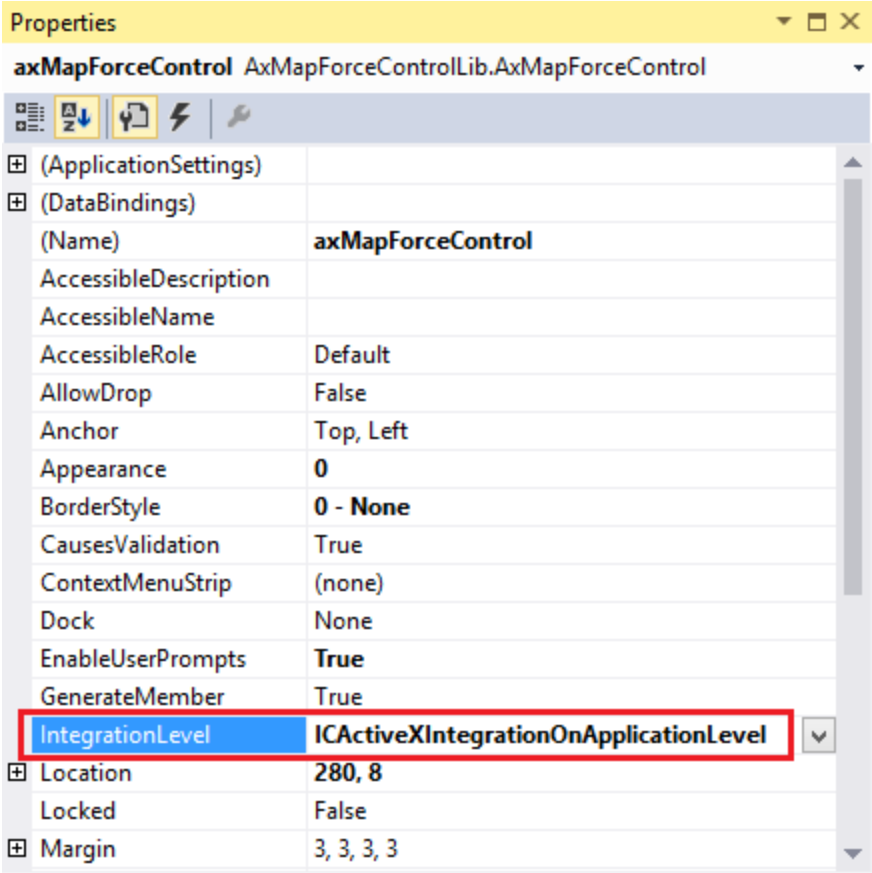
The only ActiveX control you need to integrate is `MapForceControl`[1085]. Do not instantiate or access `MapForceControlDocument`[1093] or `MapForceControlPlaceHolder`[1099] ActiveX controls when integrating at application-level.

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`[1085]. Consider using `MapForceControl.Application`[1086] 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:

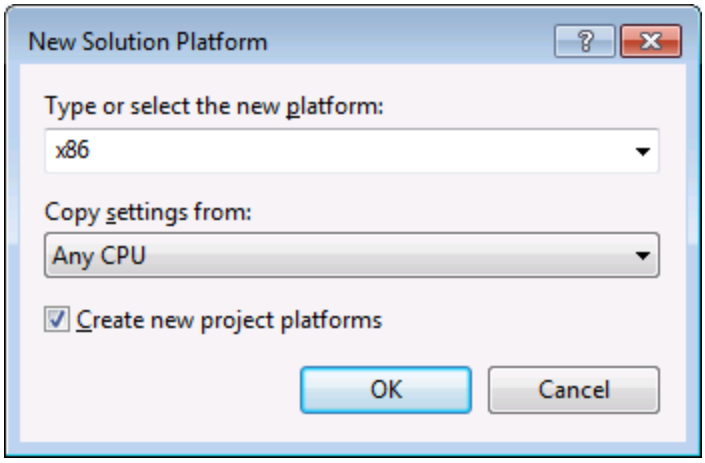
1. Check that all prerequisites are met (see Prerequisites[1042]).
2. Create a new Visual Studio Windows Forms project with a new empty form.
3. If you have not done that already, add the ActiveX controls to the toolbox (see Adding the ActiveX Controls to the Toolbox[1044]).
4. Drag the **MapForceControl** 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**.


© 2019-2025 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).


*© 2019-2025 Altova GmbH*

## 16.4 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 1046, 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 1085
- MapForceControl Document 1093
- MapForceControl Placeholder 1099

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 1044).

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 1087 property to ICActiveXIntegrationOnDocumentLevel (or "1"). To hide the control from the user, set its Visible property to False. Note that, 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.

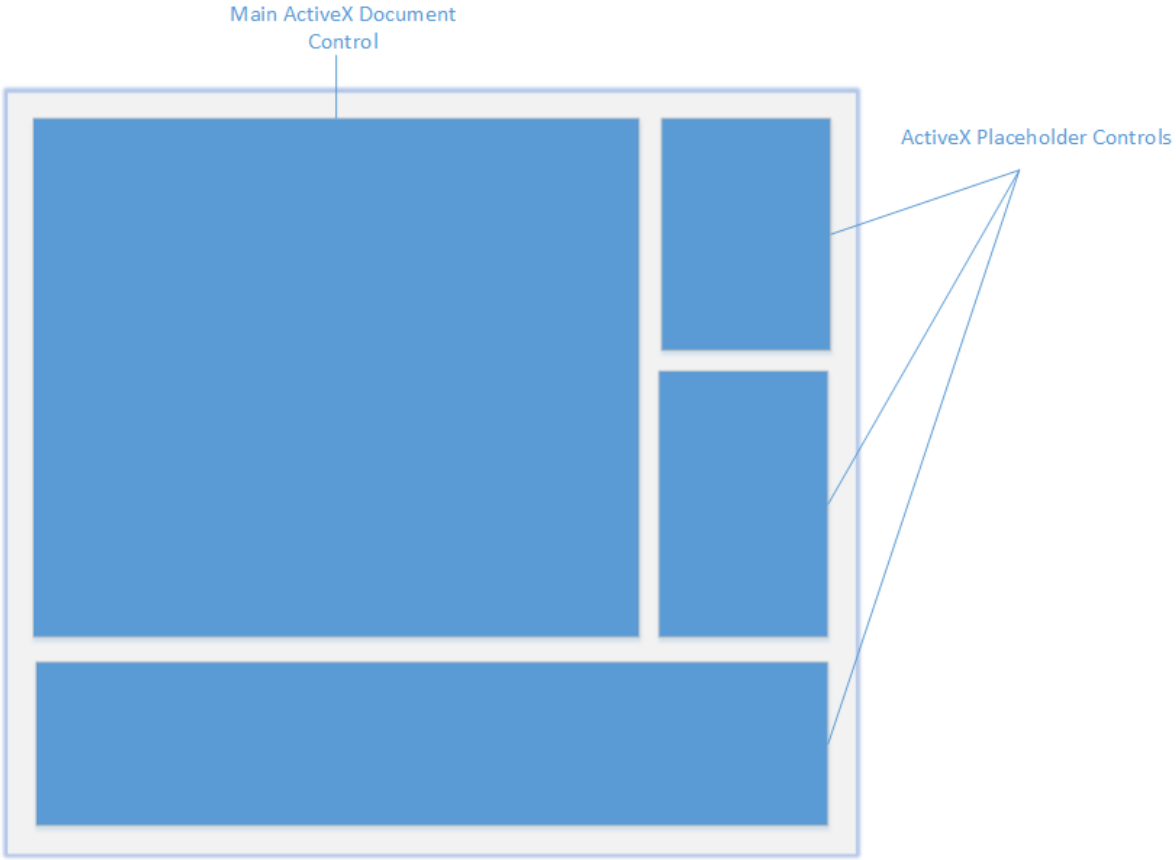
2. Create at least one instance of MapForceControl Document in your application. This control supplies the document editing window of MapForce to your application and can be instantiated multiple times if necessary. Use the method Open to load any existing file. To access document-related functionality, use the Path and Save or methods and properties accessible via the property Document. Note that 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 1102. 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 1044).
3. Drag the MapForceControl 1085 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 1093 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 1099 controls to the form (one for each additional window type that your application needs, for example, the Project window). You will typically want to place such additional placeholder controls either below or to the right or left of the main document control, for example:




7. Set the PlaceholderWindowID property of each MapForceControl Placeholder control to a valid window identifier. For the list of valid values, see MapForceControlPlaceholderWindow 1102.
8. Add commands to your application (at minimum, you will need to open, save and close documents), as shown below.


© 2019-2025 Altova GmbH

## 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[1087] property of the MapForceControl.
- To retrieve commands organized according to their menu structure, use the MainMenu[1088] property.
- To retrieve commands organized by the toolbar in which they appear, use the Toolbars[1088] property.
- To send commands to MapForce, use the Exec[1089] method.
- To query if a command is currently enabled or disabled, use the QueryStatus[1090] 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[1072] section.

## General considerations

To automate the behaviour of MapForce, use the properties, methods, and events described for the MapForceControl[1085], MapForceControl Document[1093], and MapForceControl Placeholder[1099].

For more complex access to MapForce functionality, consider using the following properties:

- `MapForceControl.Application`[1086]
- `MapForceControlDocument.Document`[1094]
- `MapForceControlPlaceHolder.Project`[1100]

These properties give you access to the MapForce automation interface (MapForceAPI)

**Note:** To open a document, always use MapForceControlDocument.Open[1096] or MapForceControlDocument.New[1096] on the appropriate document control. To open a project, always use MapForceControlPlaceHolder.OpenProject[1101] 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[1052].

# 16.5 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.

## 16.5.1 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 1052).

### 16.5.1.1 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 1042.

#### 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 1093 and several instances of MapForceControlPlaceHolder 1099 controls. Double-check that the following properties of these controls are set as shown in the table below:

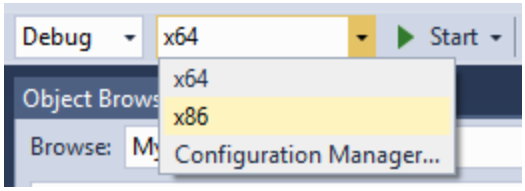
| Control name | Property | Property value |
|---|---|---|
| axMapForceControl | IntegrationLevel | ICActiveXIntegrationOnDocumentLevel |
| axMapForceControlLibrary | PlaceholderWindowID | 0 |
| axMapForceControlOutput | PlaceholderWindowID | 2 |
| axMapForceControlPreview | PlaceholderWindowID | 1 |

Here is how you can view or set the properties of an ActiveX control:

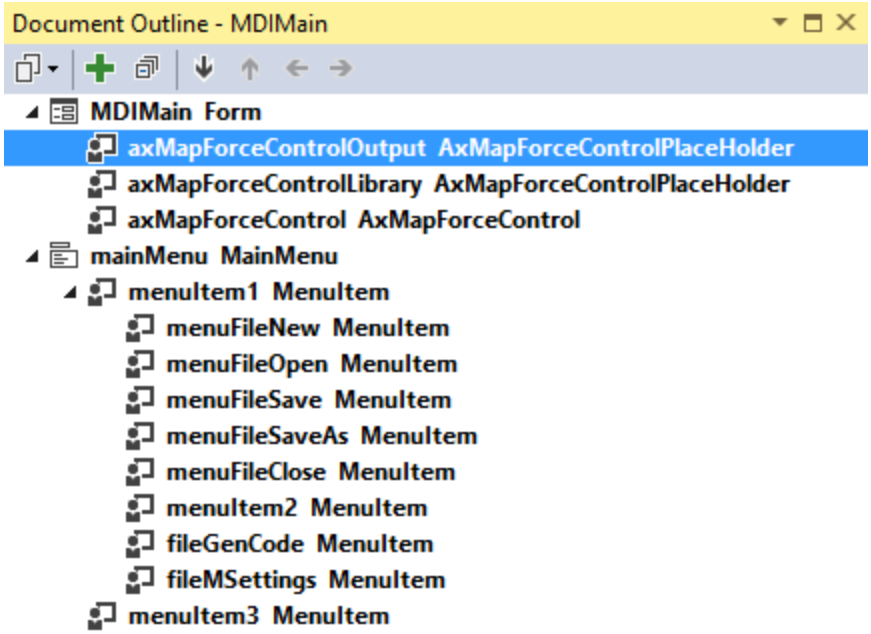
 © 2019-2025 Altova GmbH

1. Open the **MDIMain.cs** form in the designer window.

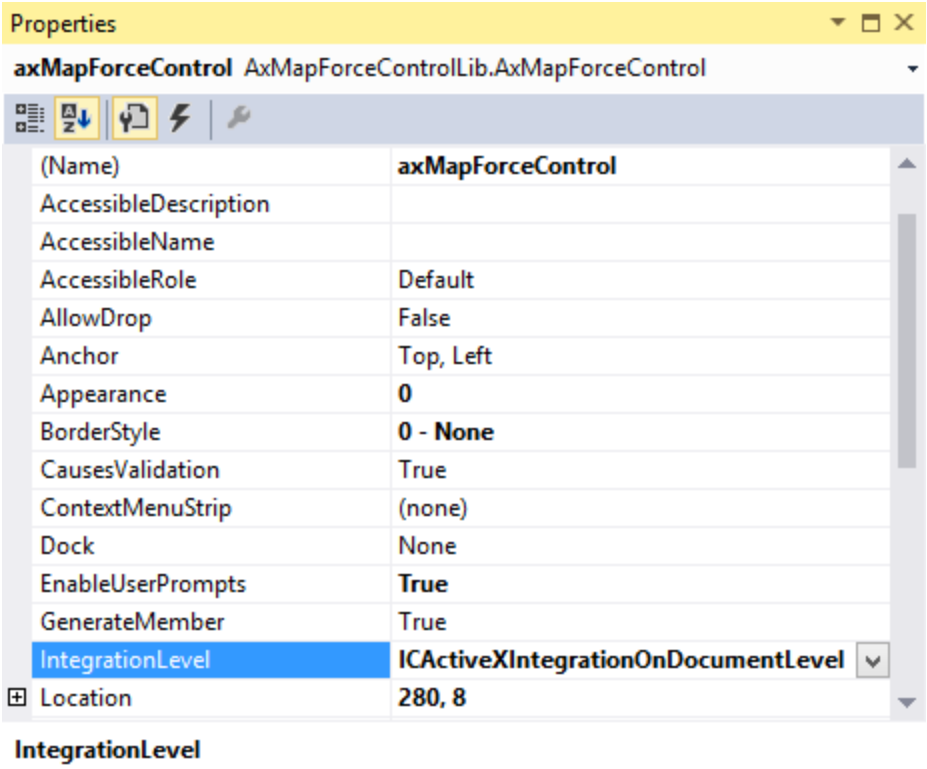
**Note:** On 64-bit Windows, it may be necessary to change the build configuration of the Visual Studio solution to "x86" **before** opening the designer window. If you need to build the sample as a 64-bit application, see Prerequisites[1042].




2. Open the **Document Outline** window of Visual Studio (On the **View** menu, click **Other Windows | Document Outline**).



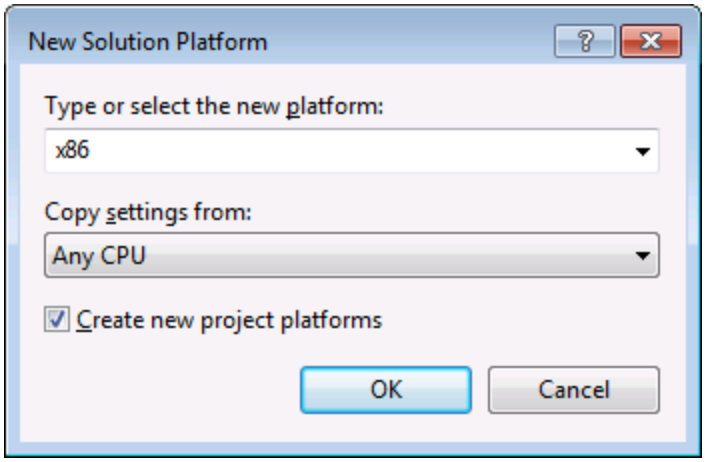

3. Click an ActiveX control in the **Document Outline** window, and edit its required property in the **Properties** window, for example:




## Step 4: Set the build platform

- Create a build platform configuration that matches the platform under which you want to build (x86, x64). Here is how you can create the build configuration:

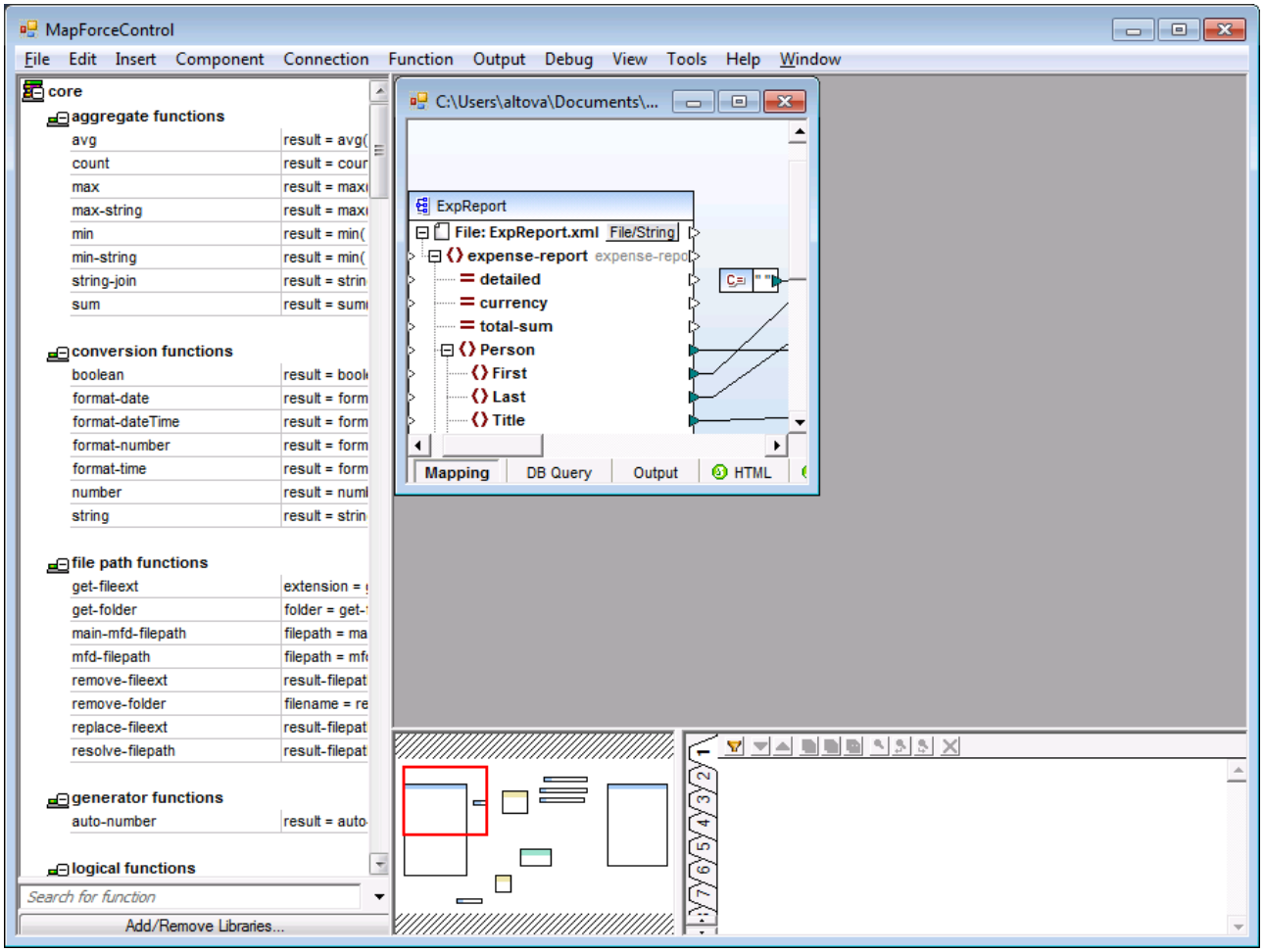
  a. Right-click the solution in Visual Studio, and select **Configuration Manager**.
  b. Under **Active solution platform**, select **New...** and then select the x86 or x64 configuration (in this example, **x86**).




You are now ready to build and run the solution in Visual Studio. Remember to build using the configuration that matches your target platform (x86, x64); otherwise, runtime errors might occur.


© 2019-2025 Altova GmbH

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.

## 16.5.1.2 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;
}
```

 © 2019-2025 Altova GmbH

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).

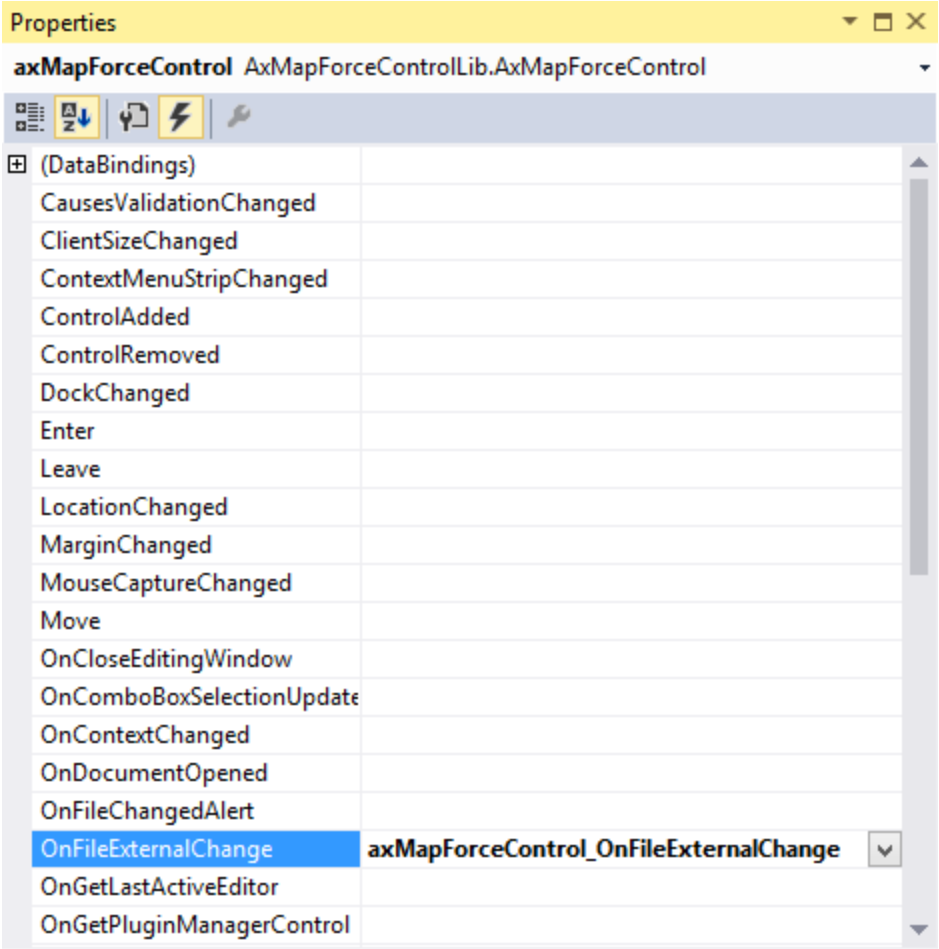
The method **ProcessCommand** delegates the execution either to the MapForceControl itself or to any active MapForce document loaded in a **MapForceControlDocument** control. This is necessary because the MapForceControl has no way to know which document is currently active in the hosting application.

```
private void ProcessCommand(int nID)
{
   MapForceDoc docMapForce = GetCurrentMapForceDoc();

   if(docMapForce != null)
      docMapForce.axMapForceControlDoc.Exec(nID);
   else
      axMapForceControl.Exec(nID);
}
```

### 16.5.1.3 Handling Events

Because all events in the MapForce library are based on connection points, you can use the C# delegate mechanism to provide the custom event handlers. You will always find a complete list of events on the property page of each control of the MapForce library. The image below shows the events of the main MapForceControl:




As you can see, the example project only overrides the `OnFileExternalChange` event. The creation of the C# delegate is done for you by the C# Framework. All you need to do is fill in the empty event handler.

For example, the handler implementation shown below turns off any file reloading and displays a message box to inform the user that a file loaded by the MapForceControl has been changed from outside:

```
private void axMapForceControl_OnFileExternalChange(object sender,
AxMapForceControlLib._DMapForceControlEvents_OnFileExternalChangeEvent e)
{
            MessageBox.Show("Attention: The file " + e.strPath + " has been changed from
outside\nbut reloading is turned off in the sample application!");

            // This turns off any file reloading:
            e.varRet = false;
}
```

 © 2019-2025 Altova GmbH

## 16.5.2 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[1042]).

- `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[1060].

### Rules for mapping the ActiveX Control names to Java

For the documentation of ActiveX controls, see Object Reference[1082]. Note that the object naming conventions are slightly different in Java compared to other languages. Namely, the rules for mapping between the ActiveX controls and the Java wrapper are as follows:

*Classes and class names*
For every component of the MapForce ActiveX interface a Java class exists with the name of the component.

*Method names*
Method names on the Java interface are the same as used on the COM interfaces but start with a small letter to conform to Java naming conventions. To access COM properties, Java methods that prefix the property name with get and set can be used. If a property does not support write-access, no setter method is available. Example: For the `IntegrationLevel` property of the MapForceControl, the Java methods `getIntegrationLevel` and `setIntegrationLevel` are available.

*Enumerations*
For every enumeration defined in the ActiveX interface, a Java enumeration is defined with the same name and values.

*Events and event handlers*
For every interface in the automation interface that supports events, a Java interface with the same name plus 'Event' is available. To simplify the overloading of single events, a Java class with default implementations for all events is provided. The name of this Java class is the name of the event interface plus '`DefaultHandler`'. For example:

`MapForceControl`: Java class to access the application
`MapForceControlEvents`: Events interface for the MapForceControl
`MapForceControlEventsDefaultHandler`: Default handler for `MapForceControlEvents`

### Exceptions to mapping rules

There are some exceptions to the rules listed above. These are listed below:

| Interface | Changes in Java class |
|---|---|
| `MapForceControlDocument`, method `New` | Renamed to `newDocument` |
| `MapForceControlDocument`, method `OpenDocument` | Removed. Use the `Open` method |
| `MapForceControlDocument`, method `NewDocument` | Removed. Use the `newDocument` method |
| `MapForceControlDocument`, method `SaveDocument` | Removed. Use the `Save` method |

### This section

This section shows how some basic MapForce ActiveX functionality can be accessed from Java code. It is organized into the following sub-sections:


- Example Java Project 1060
- Creating the ActiveX Controls 1062
- Loading Data in the Controls 1063
- Basic Event Handling 1063
- Menus 1064
- UI Update Event Handling 1066
- Creating a MapForce Mapping Table 1066


## 16.5.2.1 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) |


© 2019-2025 Altova GmbH

| | |
|---|---|
| `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 1062: Starts MapForce, which is registered as an automation server, or activates MapForce if it is already running.
- Loading Data in the Controls 1063: Locates one of the example documents installed with MapForce and opens it.
- Basic Event Handling 1063: Changes the view of all open documents to Text View. The code also shows how to iterate through open documents.
- Menus 1064: Validates the active document and shows the result in a message box. The code shows how to use output parameters.
- UI Update Event Handling 1066: Shows how to handle MapForce events.
- Creating a MapForce Mapping Table 1066: 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\
\MapForce2025\\MapForceExamples\\";
```

## Running the example from the command line

To run the example from the command line:

1. Check that all prerequisites are met (see Prerequisites 1042).
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 1042).
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**.

## 16.5.2.2 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;
```


© 2019-2025 Altova GmbH

```
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 ) );
```

## 16.5.2.3 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\\MapForce2025\\MapForceExamples\\";
3       mapForceProjectToolWindow = new
MapForceControlPlaceHolder( MapForceControlPlaceholderWindow.MapForceXProjectWindow.getValu
e(), strExamplesFolder + "MapForceExamples.mfp" ) ;
```

## 16.5.2.4 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   }
```

## 16.5.2.5 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 1066).

```
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    }
```

## 16.5.2.6 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.getKey() )
> 2 );
11   }
12  /**
13    * Call-back from the MapForceControl.
14    * Usually called while enabling/disabling commands due to UI updates
15    */
16   @Override
17   public boolean onIsActiveEditor( String i_strFilePath ) throws AutomationException
18   {
19     try {
20       return
MapForceContainer.mapForceDocument.getDocument().getFullName().equalsIgnoreCase( i_strFileP
ath );
21     } catch ( Exception e ) {
22       return false;
23     }
24   }
```

## 16.5.2.7 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.


© 2019-2025 Altova GmbH

```
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();
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(); }

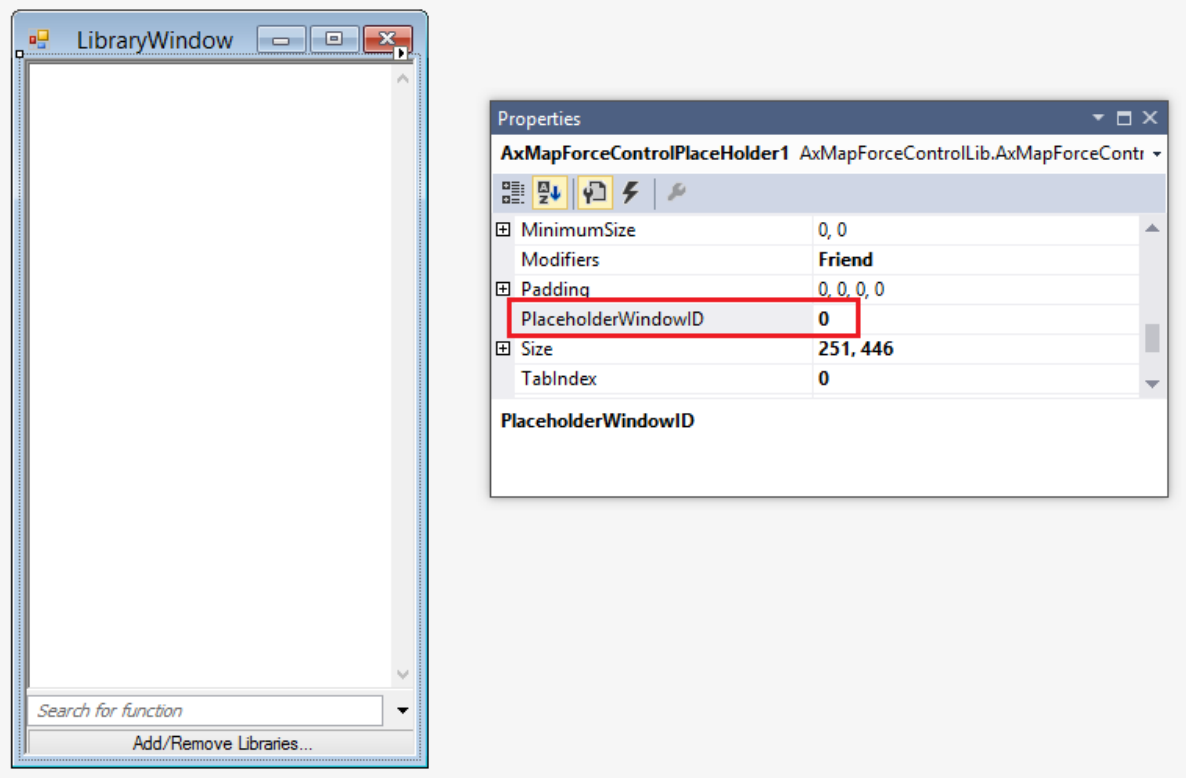
81      } );
82
83      // Set width
84      for( index = 0 ; index < columnNames.length ; ++index )
85        myTable.getColumnModel().getColumn( index ).setMinWidth( 80 );
86      myTable.getColumnModel().getColumn( 5 ).setMinWidth( 400 );
87    }
88    catch (Exception e)
89    {
90      e.printStackTrace();
91    }
92  }
93
94 }
```

## 16.5.3 VB.NET

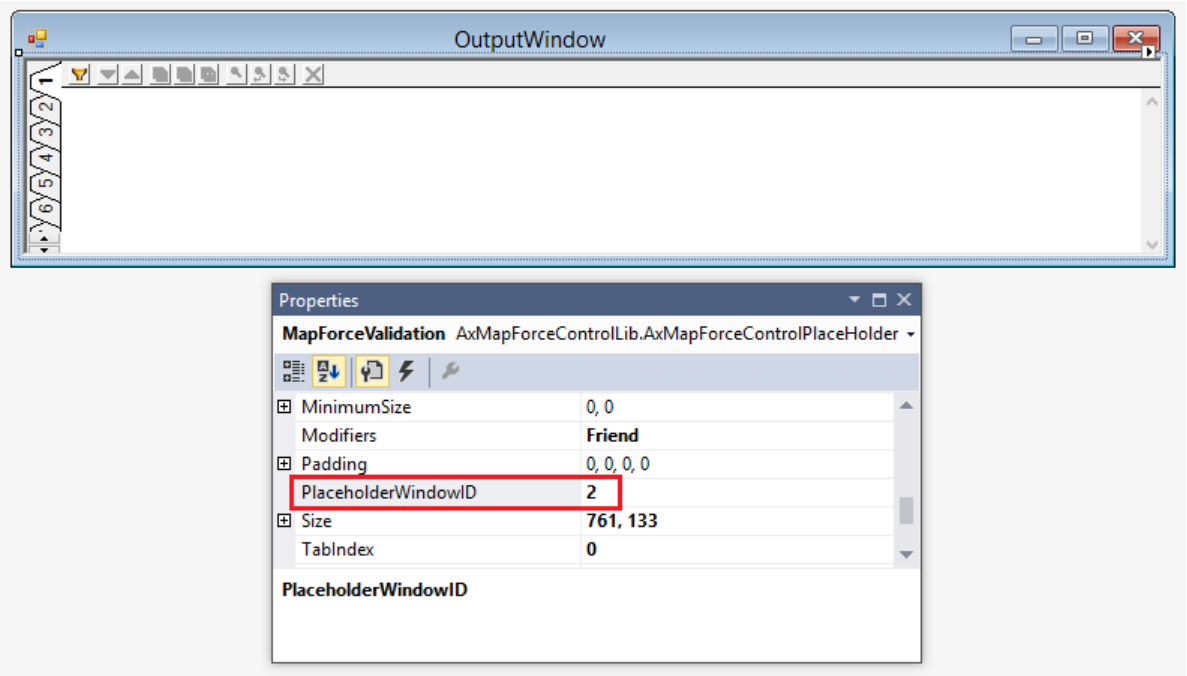
Source code which illustrates integration of MapForceControl into a VB.NET application can be found in the folder `<ApplicationFolder>\Examples\ActiveX\VB.NET` of your MapForce installation. The solution consists of three windows, as follows:

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).




Before attempting to build and run this solution, note the following steps:

## Step 1: Check the prerequisites

For the list of prerequisites, see Prerequisites 1042.

## Step 2: Copy the sample to a directory where you have write permissions

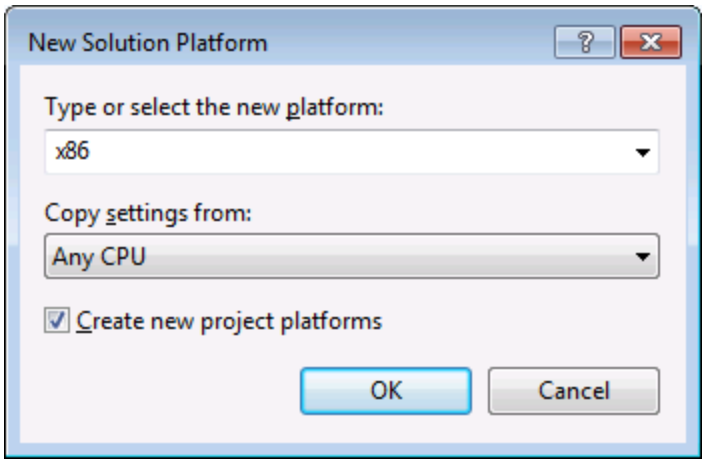
To avoid running Visual Studio as an Administrator, copy the source code to a directory where you have write permissions, instead of running it from the default location.

## 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:

1. Right-click the solution in Visual Studio, and select **Configuration Manager**.
2. Under **Active solution platform**, select **New...** and then select the x86 or x64 configuration (in this example, **x86**).

 © 2019-2025 Altova GmbH




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.

# 16.6 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` 1089 or the `MapForceControlDocument.Exec` 1095 methods. To query the status of a command, use the `MapForceControl.QueryStatus` 1090 or `MapForceControlDocument.QueryStatus` 1096 methods.

Depending on the edition of MapForce you have installed, some of these commands might not be supported.

## 16.6.1 "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 |

 © 2019-2025 Altova GmbH

| Menu item | Command name | ID |
|---|---|---|
| Generate Code in Selected Language | ID_FILE_GENERATE_SELECTED_CODE | 32362 |
| 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 |

## 16.6.2 "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 |

## 16.6.3 "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 |

## 16.6.4 "Project" Menu

The "Project" menu has the following commands:

| Menu item | Command name | ID |
|---|---|---|
| Reload Project | ID_PROJECT_RELOAD | 32476 |


© 2019-2025 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 |

## 16.6.5 "Component" Menu

The "Component" menu has the following commands:

| Menu item | Command name | ID |
|---|---|---|
| Change Root Element... | ID_COMPONENT_CHANGEROOTELEMENT | 32334 |

| Menu item | Command name | ID |
|---|---|---|
| Edit Schema Definition in XMLSpy | ID_COMPONENT_EDIT_SCHEMA | 32337 |
| Edit FlexText Configuration | ID_COMPONENT_EDIT_MFT | 32301 |
| Add/Remove/Edit Database Objects... | ID_COMPONENT_SELECTTABLES | 32346 |
| Create Mapping to EDI X12 997 | ID_COMPONENT_CREATE_MAPPING_TO_997 | 32483 |
| Create Mapping to EDI X12 999 | ID_COMPONENT_CREATE_MAPPING_TO_999 | 32484 |
| Refresh | IDC_COMMAND_REFRESH_COMPONENT | 32373 |
| Add Duplicate Input Before | ID_COMPONENT_CREATE_DUPLICATE_ICON_BEFORE | 32503 |
| Add Duplicate Input After | ID_COMPONENT_CREATE_DUPLICATE_ICON | 32335 |
| Remove Duplicate | ID_COMPONENT_REMOVE_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 |

## 16.6.6 "Connection" Menu

The "Connection" menu has the following commands:

 © 2019-2025 Altova GmbH

| Menu item | Command name | ID |
|---|---|---|
| Auto Connect Matching Children | ID_CONNECTION_AUTOCONNECTCHILDREN | 32342 |
| Settings for Connect Matching Children | ID_CONNECTION_SETTINGS | 32344 |
| Connect Matching Children... | ID_CONNECTION_MAPCHILDELEMENTS | 32343 |
| Target Driven (Standard) | ID_POPUP_NORMALCONNECTION | 32401 |
| Copy-All (Copy Child Items) | ID_POPUP_NORMALWITHCHILDREN_CONNECTION | 32460 |
| Source Driven (Mixed Content) | ID_POPUP_ORDERBYSOURCECONNECTION | 32403 |
| Properties | ID_POPUP_CONNECTION_SETTINGS | 32398 |

## 16.6.7 "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 |

## 16.6.8 "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 |

| Menu item | Command name | ID |
|---|---|---|
| C# (Sharp) | ID_SELECT_LANGUAGE_CSHARP | 32430 |
| C++ | ID_SELECT_LANGUAGE_CPP | 32429 |
| Built-In Execution Engine | ID_SELECT_LANGUAGE_BUILTIN | 32490 |
| Validate Output File | ID_XML_VALIDATE | 32458 |
| Save Output File... | IDC_FILE_SAVEGENERATEDOUTPUT | 32321 |
| Save All Output Files... | IDC_FILE_SAVEALLGENERATEDOUTPUT | 32374 |
| Regenerate Output | ID_REGENERATE_PREVIEW_OUTPUT | 32480 |
| Run SQL-Script | ID_TRANSFORM_RUN_SQL | 32442 |
| Insert/Remove Bookmark | ID_TOGGLE_BOOKMARK | 32317 |
| Next Bookmark | ID_GOTONEXTBOOKMARK | 32315 |
| Previous Bookmark | ID_GOTOPREVBOOKMARK | 32314 |
| Remove All Bookmarks | ID_REMOVEALLBOOKMARKS | 32313 |
| Pretty-Print XML Text | ID_PRETTY_PRINT_OUTPUT | 32363 |
| Text View Settings | ID_TEXTVIEWSETTINGSDIALOG | 32472 |

## 16.6.9 "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 |


© 2019-2025 Altova GmbH

## 16.6.10 "View" Menu

The "View" menu has the following commands:

| Menu item | Command name | ID |
|---|---|---|
| Show Annotations | ID_SHOW_ANNOTATION | 32435 |
| Show Types | ID_SHOW_TYPES | 32437 |
| Show Library in Function Header | ID_VIEW_SHOWLIBRARYINFUNCTIONHEADER | 32448 |
| Show Tips | ID_SHOW_TIPS | 32436 |
| XBRL Display Options | ID_VIEW_XBRL_DISPLAY_OPTIONS | 32473 |
| Show Selected Component Connectors | ID_VIEW_AUTOHIGHLIGHTCOMPONENTCONNECTIONS | 32443 |
| Show Connectors from Source to Target | ID_VIEW_RECURSIVEAUTOHIGHLIGHT | 32447 |
| Zoom... | ID_VIEW_ZOOM | 32451 |
| Back | ID_CMD_BACK | 32479 |
| Forward | ID_CMD_FORWARD | 32478 |
| Status Bar | ID_VIEW_STATUS_BAR | 59393 |
| Library Window | ID_VIEW_LIBRARY_WINDOW | 32445 |
| Messages | ID_VIEW_VALIDATION_OUTPUT | 32450 |
| Overview | ID_VIEW_OVERVIEW_WINDOW | 32446 |
| 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 |

## 16.6.11 "Tools" Menu

The "Tools" menu has the following commands:

| Menu item | Command name | ID |
|---|---|---|
| Global Resources | IDC_GLOBALRESOURCES | 37401 |

| Menu item | Command name | ID |
|---|---|---|
| <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 |

## 16.6.12 "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 |

## 16.6.13 "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 |


© 2019-2025 Altova GmbH

| Menu item | Command name | ID |
|---|---|---|
| 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 |

## 16.7 Object Reference

**Objects:**
MapForceCommand 1082
MapForceCommands 1084
MapForceControl 1085
MapForceControlDocument 1093
MapForceControlPlaceHolder 1099

To give access to standard MapForce functionality, objects of the **MapForce automation interface** can be accessed as well. See MapForceControl.Application 1086, MapForceControlDocument.Document 1094 and MapForceControlPlaceHolder.Project 1100 for more information.

### 16.7.1 MapForceCommand

**Properties:**
`ID` 1083
`Label` 1083
`Name` 1083
`IsSeparator` 1083
`ToolTip` 1084
`StatusText` 1084
`Accelerator` 1083
`SubCommands` 1084

**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<br>• `IsSeparator` is false<br>• `SubCommands` is empty |
| A command container | • `ID` is zero<br>• `IsSeparator` is false<br>• `SubCommands` contains a collection of `Command` objects. |
| Separator | • `ID` is zero<br>• `IsSeparator` is true |


© 2019-2025 Altova GmbH

### 16.7.1.1 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`.

### 16.7.1.2 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`[1089]) or query its status (using `QueryStatus`[1090]). If the command is a container for other commands (for example, a top-level menu), or a separator, the ID is 0.

### 16.7.1.3 IsSeparator

**Property:** `IsSeparator` as `boolean`

**Description:**
The property returns `true` if the command object is a menu separator; `false` otherwise. See also Command[1082].

### 16.7.1.4 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.

### 16.7.1.5 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.

### 16.7.1.6 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.

### 16.7.1.7 SubCommands

**Property:** `SubCommands` as Commands 1084

**Description:**
The `SubCommands` property gets the collection of `Command` 1082 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`.

### 16.7.1.8 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.

## 16.7.2 MapForceCommands

**Properties:**
`Count` 1085
`Item` 1085

**Description:**
Collection of `Command` 1082 objects to get access to command labels and IDs of the MapForceControl. Those commands can be executed with the `Exec` 1089 method and their status can be queried with `QueryStatus` 1090.


© 2019-2025 Altova GmbH

### 16.7.2.1 Count

**Property:** `Count` as `long`

**Description:**
Number of `Command` 1082 objects on this level of the collection.

### 16.7.2.2 Item

**Property:** `Item` (`n` as `long`) as `Command` 1082

**Description:**
Gets the command with the index `n` in this collection. Index is 1-based.

## 16.7.3 MapForceControl

**Properties:**
`IntegrationLevel` 1087
`Appearance` 1086
`Application` 1086
`BorderStyle` 1086
`CommandsList` 1087
`EnableUserPrompts` 1087
`MainMenu` 1088
`Toolbars` 1088

**Methods:**
`Open` 1090
`Exec` 1089
`QueryStatus` 1090

**Events:**
`OnUpdateCmdUI` 1092
`OnOpenedOrFocused` 1092
`OnCloseEditingWindow` 1091
`OnFileChangedAlert` 1091
`OnDocumentOpened` 1091
`OnValidationWindowUpdated` 1092

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
```

## 16.7.3.1 Properties

The following properties are defined:

IntegrationLevel 1087
EnableUserPrompts 1087
Appearance 1086
BorderStyle 1086

Command related properties:
CommandsList 1087
MainMenu 1088
Toolbars 1088

Access to MapForceAPI:
Application 1086

### 16.7.3.1.1 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`.

### 16.7.3.1.2 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.

### 16.7.3.1.3 BorderStyle

**Property:** `BorderStyle` as `short`

**Dispatch Id:** *-504*

**Description:**
A value of `1` displays the control with a thin border. Default value is `0`.

 © 2019-2025 Altova GmbH

### 16.7.3.1.4 CommandsList

**Property:** `CommandList` as `Commands` 1084 (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` 1088. To get toolbar commands, use `Toolbars` 1088.

```
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*

### 16.7.3.1.5 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*.

### 16.7.3.1.6 IntegrationLevel

**Property:** `IntegrationLevel` as `ICActiveXIntegrationLevel` 1102

**Dispatch Id:** *1000*

**Description:**
The `IntegrationLevel` property determines the operation mode of the control. See also Integration at Application Level 1046 and Integration at Document Level 1049 for more information.

**Note:** It is important to set this property immediately after the creation of the `MapForceControl` object.

### 16.7.3.1.7 MainMenu

**Property:** `MainMenu as Command` [1082] (read-only)

**Dispatch Id:** *1003*

**Description:**
This property provides information about the structure and commands available in the MapForceControl main menu, as a `Command` object. The `Command` object contains all available submenus of MapForce (for example "File", "Edit", "View" etc.). To access the submenu objects, use the `SubCommands` property of the `MainMenu` property. Each submenu is also a `Command` object. For each submenu, you can then further iterate through their `SubCommands` property in order to get their corresponding child commands and separators (this technique may be used, for example, to create the application menu programmatically). Note that some menu commands act as containers ("parents") for other menu commands, in which case they also have a `SubCommands` property. To get the structure of all menu commands programmatically, you will need a recursive function, as illustrated for C# in Retrieving Command Information [1055].

```
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*

### 16.7.3.1.8 Toolbars

**Property:** `Toolbars as Commands` [1084] (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).


© 2019-2025 Altova GmbH

```
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*

## 16.7.3.2 Methods

The following methods are defined:

Open 1090
Exec 1089
QueryStatus 1090

### 16.7.3.2.1 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` 1087. To retrieve the status of any command, use `QueryStatus` 1090.

#### 16.7.3.2.2 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` 1096 and `MapForceControlPlaceHolder.OpenProject` 1101.

#### 16.7.3.2.3 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.

### 16.7.3.3 Events

The MapForceControl ActiveX control provides the following connection point events:

`OnUpdateCmdUI` 1092
`OnOpenedOrFocused` 1092
`OnCloseEditingWindow` 1091
`OnFileChangedAlert` 1091
`OnDocumentOpened` 1091
`OnValidationWindowUpdated` 1092


© 2019-2025 Altova GmbH

#### 16.7.3.3.1 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.

#### 16.7.3.3.2 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`[1098] instead.

#### 16.7.3.3.3 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.

#### 16.7.3.3.4 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).

### 16.7.3.3.5 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.

### 16.7.3.3.6 OnToolWindowUpdated

**Event:** `OnToolWindowUpdated(pToolWnd` as `long )`

**Dispatch Id:** *1006*

**Description:**
This event is triggered when the tool window is updated.

### 16.7.3.3.7 OnUpdateCmdUI

**Event:** `OnUpdateCmdUI()`

**Dispatch Id:** *1003*

**Description:**
Called frequently to give integrators a good opportunity to check status of MapForce commands using `MapForceControl.QueryStatus` 1090. Do not perform long operations in this callback.

### 16.7.3.3.8 OnValidationWindowUpdated

**Event:** `OnValidationWindowUpdated()`

**Dispatch Id:** *3*

 © 2019-2025 Altova GmbH

**Description:**
This event is triggered whenever the validation output window is updated with new information.

## 16.7.4 MapForceControlDocument

**Properties:**
Appearance 1094
BorderStyle 1094
Document 1094
IsModified 1094
Path 1095
ReadOnly 1095

**Methods:**
Exec 1095
New 1096
Open 1096
QueryStatus 1096
Reload 1096
Save 1097
SaveAs 1097

**Events:**
OnDocumentOpened 1098
OnDocumentClosed 1098
OnModifiedFlagChanged 1099
OnFileChangedAlert 1098
OnActivate 1097

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
```

### 16.7.4.1 Properties

The following properties are defined:

ReadOnly 1095
IsModified 1094
Path 1095
Appearance 1094
BorderStyle 1094

Access to MapForceAPI:

Document 1094

### 16.7.4.1.1 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`.

### 16.7.4.1.2 BorderStyle

**Property:** `BorderStyle` as `short`

**Dispatch Id:** *-504*

**Description:**
A value of `1` displays the control with a thin border. Default value is `0`.

### 16.7.4.1.3 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.

### 16.7.4.1.4 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.


© 2019-2025 Altova GmbH

#### 16.7.4.1.5 Path

**Property:** `Path` as `string`

**Dispatch Id:** *1005*

**Description:**
Sets or gets the full path name of the document loaded into the control.

#### 16.7.4.1.6 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.

### 16.7.4.2 Methods

The following methods are defined:

Document handling:
`New` 1096
`Open` 1096
`Reload` 1096
`Save` 1097
`SaveAs` 1097

Command Handling:
`Exec` 1095
`QueryStatus` 1096

#### 16.7.4.2.1 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` 1088 property of MapForceControl. To get toolbar commands, use the `Toolbars` 1088 property of the MapForceControl.

### 16.7.4.2.2 New

**Method:** `New` () as `boolean`

**Dispatch Id:** *1000*

**Description:**
This method initializes a new document inside the control.

### 16.7.4.2.3 Open

**Method:** `Open` (`strFileName` as `string`) as `boolean`

**Dispatch Id:** *1001*

**Description:**
`Open` loads the file `strFileName` as the new document into the control.

### 16.7.4.2.4 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.

### 16.7.4.2.5 Reload

**Method:** `Reload()` as `boolean`

**Dispatch Id:** *1002*


© 2019-2025 Altova GmbH

**Description:**
`Reload` updates the document content from the file system.

### 16.7.4.2.6 Save

**Method:** `Save()` as `boolean`

**Dispatch Id:** *1003*

**Description:**
`Save` saves the current document at the location `Path`[1095].

### 16.7.4.2.7 SaveAs

**Method:** `SaveAs` (`strFileName` as `string`) as `boolean`

**Dispatch Id:** *1004*

**Description:**
`SaveAs` sets `Path`[1095] to *strFileName* and then saves the document to this location.

## 16.7.4.3 Events

The MapForceControlDocument ActiveX control provides following connection point events:

`OnDocumentOpened`[1098]
`OnDocumentClosed`[1098]
`OnModifiedFlagChanged`[1099]
`OnFileChangedAlert`[1098]
`OnActivate`[1097]
`OnSetEditorTitle`[1099]

### 16.7.4.3.1 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.

### 16.7.4.3.2 OnDocumentClosed

**Event:** `OnDocumentClosed` (`objDocument` as `Document`)

**Dispatch Id:** *1001*

**Description:**
This event is triggered whenever the document loaded into this control is closed. The argument `objDocument` is a `Document` object from the MapForce automation interface and should be used with care.

### 16.7.4.3.3 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.

### 16.7.4.3.4 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.

### 16.7.4.3.5 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.


© 2019-2025 Altova GmbH

#### 16.7.4.3.6 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.

#### 16.7.4.3.7 OnSetEditorTitle

**Event:** OnSetEditorTitle ()

**Dispatch Id:** *1006*

**Description:**
This event is being raised when the contained document is being internally renamed.

### 16.7.5 MapForceControlPlaceHolder

**Properties available for all kinds of placeholder windows:**
`PlaceholderWindowID` 1100

**Properties for project placeholder window:**
`Project` 1100

**Methods for project placeholder window:**
`OpenProject` 1101
`CloseProject` 1101

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
```

#### 16.7.5.1 Properties

The following properties are defined:

`PlaceholderWindowID` 1100

Access to MapForceAPI:
`Project` 1100

#### 16.7.5.1.1 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.

#### 16.7.5.1.2 PlaceholderWindowID

**Property:** `PlaceholderWindowID as MapForceControlPlaceholderWindow` 1102

**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` 1102 enumeration. The control changes its state immediately and shows the new MapForce window.

#### 16.7.5.1.3 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` 1100 with a value of `MapForceXProjectWindow (=3)`. The property is read-only.

### 16.7.5.2 Methods

The following method is defined:

`OpenProject` 1101
`CloseProject` 1101


© 2019-2025 Altova GmbH

#### 16.7.5.2.1 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`**[1100] different to `MapForceXProjectWindow (=3)`.

#### 16.7.5.2.2 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`[1100] different to `MapForceXProjectWindow (=3)`.

### 16.7.5.3 Events

The MapForceControlPlaceholder ActiveX control provides following connection point events:

`OnModifiedFlagChanged`[1101]

#### 16.7.5.3.1 OnModifiedFlagChanged

**Event:** `OnModifiedFlagChanged` (`i_bIsModified` as `boolean`)

**Dispatch Id:** *1*

**Description:**
This event gets triggered only for placeholder controls with a `PlaceholderWindowID`[1100] 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.

#### 16.7.5.3.2 OnSetLabel

**Event:** `OnSetLabel(i_strNewLabel as string)`

**Dispatch Id:** *1000*

**Description:**
Raised when the title of the placeholder window is changed.

## 16.7.6 Enumerations

The following enumerations are defined:

ICActiveXIntegrationLevel 1102
MapForceControlPlaceholderWindow 1102

### 16.7.6.1 ICActiveXIntegrationLevel

Possible values for the IntegrationLevel 1087 property of the MapForceControl.

```
ICActiveXIntegrationOnApplicationLevel = 0
ICActiveXIntegrationOnDocumentLevel = 1
```

### 16.7.6.2 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
```


© 2019-2025 Altova GmbH

# 17 Comandos de menú

En esta sección se describirán los comandos de menú en MapForce. Estos son los comandos de menú disponibles:


- Archivo 1104
- Editar 1107
- Insertar 1108
- Proyecto 1111
- Componente 1113
- Conexión 1115
- Función 1116
- Resultados 1117
- Depuración 1119
- Vista 1121
- Herramientas 1123
- Ventana 1142
- Ayuda 1143

# 17.1 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 el apartado Proyectos 98.

- Abrir

  Abre el archivo de diseño de asignación guardado anteriormente (`.mfd`). En las ediciones Professional y Enterprise también abre proyectos de asignación de datos (`.mfp`). Para más detalles consulte el apartado Proyectos 98.

- 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 de asignación que están abiertos.

- Volver a cargar

  Vuelve a cargar el archivo de asignación activo. Al volver a cargarlo, se revierten los últimos cambios.

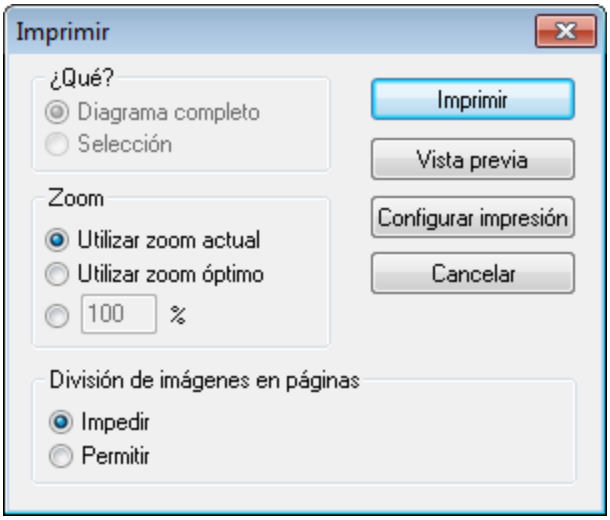
- Cerrar, Cerrar todos

  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 guardar los archivos que están sin guardar.

- 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.


© 2019-2025 Altova GmbH




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

  El comando **Validar asignación** comprueba si todas las asignaciones (conectores) son válidas y emite advertencias y errores de validación. Para más detalles consulte Validación 66.

- Configurar asignación

  Abre el cuadro de diálogo Cambiar configuración de la asignación 78, donde puede definir las opciones de configuración del documento activo.

- 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

  El comando **Generar código en el lenguaje seleccionado** genera código en el lenguaje seleccionado en la barra de herramientas. El comando **Generar código en <idioma>** permite generar código en XSLT 1-3 (*todas las ediciones*), XQuery, Java, C# y C++ (*ediciones Professional y Enterprise*). Al hacer clic en estos comandos se abre el cuadro de diálogo **Buscar carpeta**, donde puede seleccionar la ubicación de los archivos generados. Los nombres de los archivos generados se define en el cuadro de diálogo Configurar asignación 78.

  Para más información sobre los lenguajes de transformación disponibles consulte Lenguajes de transformación 19. Para más información sobre cómo generar código, consulte Generación de código 68.

- 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 839.

- Implementar en FlowForce Server (*ediciones Professional y Enterprise*)

  Implementa la asignación activa en FlowForce Server. Para más detalles véase Implementar una asignación de MapForce 842.

- Generar documentación (*ediciones Professional y Enterprise*)

  Genera documentación para los proyectos de asignación en varios formatos de salida. Para más información consulte Generar documentación de la asignación 799).

- Archivos recientes

  Muestra una lista con archivos abiertos recientemente.

- Salir

  Cierra la aplicación. Antes aparece un aviso preguntando si desea guardar los archivos que estén sin guardar.


© 2019-2025 Altova GmbH

## 17.2 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**.

# 17.3 Insertar

En este apartado puede consultar todos los comandos del menú **Insertar**.

- Archivo o esquema XML

  Agrega un esquema XML o archivo XML de instancia a la asignación. Si selecciona un archivo XML sin una referencia de esquema, MapForce puede generar un esquema XML coincidente 138. Si selecciona un archivo de esquema XML, aparece un aviso solicitando 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 (*Professional y Enterprise Editions*)

  Agrega un componente de base de datos. Consulte Bases de datos 174. También puede agregar un componente de BD con el comando de la barra de herramientas. En MapForce Enterprise Edition, también puede agregar bases de datos NoSQL como componentes.

- EDI (*Enterprise Edition*)

  Agrega un documento EDI. También puede agregar un componente EDI con el comando de la barra de herramientas.

- Archivo de texto (*Professional y Enterprise Editions*)

  Agrega un documento de archivo plano como un archivo CSV o un archivo de texto de longitud fija. Para más detalles, consulte Archivos CSV y archivos de texto 344. También puede agregar un archivo de texto con el comando de la barra de herramientas. MapForce Enterprise Edition también le permite procesar archivos de texto con FlexText.

- Función de servicio web (*Enterprise Edition*)

  Agrega 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*)

  Agrega un archivo de Microsoft Excel 2007+ (`.xlsx`). 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**. También puede agregar un archivo de Excel con el comando de la barra de herramientas.

- Documento XBRL (*Enterprise Edition*)

  Agrega un documento de instancia XBRL o una taxonomía XBRL. También puede agregar un componente XBRL con el comando de la barra de herramientas.

- Archivo o esquema JSON (*Enterprise Edition*)


© 2019-2025 Altova GmbH

Agrega un archivo o esquema JSON. También puede agregar un componente JSON con el comando de la barra de herramientas.

- Archivo de Protocol Buffers (*Enterprise Edition*)

  Agrega un archivo binario cifrado en formato Protocol Buffers. También puede agregar un archivo Protocol Buffers con el comando de la barra de herramientas.

- Insertar un documento PDF (*Enterprise Edition*)

  Agrega un documento PDF. También puede insertar un documento PDF desde la barra de herramientas.

- Insertar componente de entrada

  Los componentes de entrada simple pueden utilizarse como parámetros de entrada relevantes para toda la asignación de datos o solo en el contexto de las funciones definidas por el usuario. Para más información, consulte Entrada simple 369 y Parámetros en funciones definidas por el usuario 492. También puede insertar un componente de entrada simple con el comando de la barra de herramientas.

- Insertar componente de salida

  Los componentes de salida simple pueden utilizarse como componentes de salida en las asignaciones de datos y como parámetros de salida en funciones definidas por el usuario. Para más información, consulte Salida simple 380 y Parámetros en funciones definidas por el usuario 492. También puede insertar 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. Puede elegir entre estos tipos de datos: `cadena`, `número`, `demás opciones`. También puede insertar una constante con el comando de la barra de herramientas .

- Variable

  Inserta una variable, que equivale a una función definida por el usuario normal (no inline). Una variable es un tipo especial de componente que sirve para almacenar el resultado intermedio de una asignación de datos para procesarlo más tarde. También puede agregar una variable con el comando de la barra de herramientas.

- Combinación (*Professional y Enterprise Editions*)

  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 397.

- Componente de ordenación: nodos/filas

  Inserta un componente que permite ordenar nodos (véase Ordenar datos 426). También puede agregar un componente de ordenación 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 las bases de datos. Consulte la sección Filtros y condiciones 432 para obtener más información. También puede agregar un filtro con el comando de la barra de herramientas.

- SQL/NoSQL-WHERE/ORDER (*Professional y Enterprise Editions*)

  Inserta un componente que filtra datos de la base de datos de forma condicional. Para más información véase Componente WHERE/ORDER de SQL 437. 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 a nombres de meses). Para más información, consulte el apartado Asignación de valores 445. También puede insertar una asignación de valores con el comando de la barra de herramientas.

- Condición If-Else

  Inserta una condición If-Else para cuando necesite procesar un valor simple de forma condicional. Consulte la sección Filtros y condiciones 432 para obtener más información. También puede agregar una condición If-Else con el comando de la barra de herramientas.

- Excepción (*Professional y Enterprise Editions*)

  Este componente permite interrumpir el proceso de asignación de datos cuando se cumple determinada condición. También puede agregar un componente de excepción con el comando de la barra de herramientas. En MapForce Enterprise Edition, este componente también le permite definir mensajes de error en proyectos de asignación de datos WSDL. Para más información sobre los componentes de Excepción, consulte Excepciones 457.

- Comentario

  Este comando de menú le permite insertar comentarios parecidos a una nota adhesiva como componentes independientes. Para más detalles, consulte Comentarios 36. También puede agregar un comentario con el comando de la barra de herramientas.


© 2019-2025 Altova GmbH

## 17.4 Proyecto

MapForce le permite agrupar sus asignaciones de datos en proyectos de asignación 98. 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 activo.

- 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. Véase Carpetas de proyecto 101.

- Crear enlace web

  Este comando le permite crear un enlace a un recurso web externo y agregar este enlace a su proyecto. Para más detalles, consulte Proyectos 98.

- 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*)

  Permite agregar un archivo de asignación guardado anteriormente a la operación WSDL activa.

- Insertar servicio web (*Enterprise Edition*)

  Permite insertar un servicio web basado en un archivo WSDL ya existente.

- Abrir en XMLSpy

  Abre el archivo WSDL seleccionado en Altova XMLSpy.

- Generar código para todo el proyecto

  Genera código para todo el proyecto activo en la ventana **Proyecto**. El código se genera en 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 contextual.

- Propiedades

  Abre un cuadro de diálogo en el que puede configurar proyectos 100.

- Proyectos recientes

  Muestra una lista con proyectos abiertos recientemente.


© 2019-2025 Altova GmbH

## 17.5 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

  Para para poder editar un esquema en Altova XMLSpy, debe hacer clic en un esquema XML y seleccionar la opción **Editar la definición del esquema en 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 174.

- 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. MapForce puede generar automáticamente un componente X12 999 y crear automáticamente las conexiones de asignación necesarias.

- 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 del elemento seleccionado antes o después del elemento seleccionado. Los elementos de entrada duplicados no se pueden utilizar como origen de datos. Para más información consulte Duplicar elementos de entrada 43.

- Quitar el duplicado

  Quita el elemento duplicado seleccionado.

- Comentario/Instrucción de procesamiento

  Esta opción permite insertar comentarios e instrucciones de procesamiento 149 en componentes XML.

- Escribir contenido como sección CDATA

Este comando crea una sección CDATA 150 trata como unidades completas partes del documento que normalmente se interpretarían como marcado.

- Acciones de tablas de BD (*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 tablas de BD 276 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 en el campo. Así 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.

- Editar comentario

  Si en su asignación hay un componente de comentario, puede editarlo haciendo clic en él y seleccionando el comando **Editar comentario**. También puede hacer doble clic dentro del componente de comentario y editar el texto directamente en la caja del comentario. Para más información sobre los componentes de comentarios y sus tipos, consulte Comentario 36.

- Propiedades

  Abre un cuadro de diálogo que muestra las opciones de configuración del componente seleccionado. Consulte Cambiar la configuración de los componentes 41.


© 2019-2025 Altova GmbH

## 17.6 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**. Para más información sobre conexiones y sus tipos consulte Conexiones 48.

- Configurar la conexión de secundarios equivalentes

  Le ayuda a definir las conexiones de secundarios equivalentes. Para más detalles, consulte Conexiones de secundarios equivalentes 55.

- 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. Las opciones definidas en este cuadro de diálogo se aplican cuando esté activo el icono (**Conexión automática de secundarios**) de la barra de herramientas. Para más detalles, consulte Conexiones de secundarios equivalentes 55.

- Basada en destino (estándar)

  Cambia el tipo de conector a una asignación estándar. Para más información consulte Conexiones basadas en el destino vs. conexiones basadas en el origen 52.

- Copia total (copia de los elementos secundarios)

  Crea conexiones para todos los elementos secundarios equivalentes. La ventaja principal de las conexiones de copia total es que simplifican visualmente el área de trabajo: En vez de múltiples conexiones se crea una sola conexión, representada por una línea gruesa. Consulte también Conexiones de copia total 57.

- 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 52.

- Propiedades

  Abre el cuadro de diálogo **Configuración de la conexión** donde puede definir los tipos de conexión y la configuración de la anotación. Para más información consulte Configuración de la conexión 59.

# 17.7 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 485 (UDF por sus siglas en inglés). También puede crear una función definida por el usuario con el comando de la barra de herramientas .

- Crear una 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 489. 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 491.

- 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* 370). 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* 492). 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* 381). 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 485. También puede insertar un componente de salida simple con el comando de la barra de herramientas .


© 2019-2025 Altova GmbH

## 17.8 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 [19]. También puede seleccionar un lenguaje de transformación a través de la barra de herramientas.

- Validar archivo de salida

  Valida el archivo XML de salida con el esquema al que se hace referencia. Consulte Validación [66].

- 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 [784]. Consulte también el Tutorial 4 [128].

- Volver a generar archivo de salida

  Vuelve a generar los datos visibles en el panel **Resultados**

- Ejecutar script SQL/NoSQL (*ediciones Professional y Enterprise*)

  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 [174].

- Insertar/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 en pretty-print

  Ajusta el formato del documento XML en el panel **Resultados** 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 [71] del grupo de opciones *Tabulaciones*.

- Configurar la vista Texto

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 71.

 © 2019-2025 Altova GmbH

# 17.9 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 808.

- 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 808.

- 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 808.

- 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 808.

- 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 808.

- 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 808.

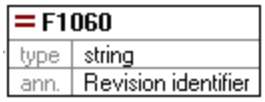
© 2019-2025 Altova GmbH

# 17.10 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 60.




- 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 50.

- Zoom

  Abre el cuadro de diálogo de **zoom**, donde puede indicar el factor de zoom en formato numérico o con un control deslizante.

- 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 [27]. 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 [26]. Arrastre el rectángulo rojo por la ventana de vista general para navegar por el panel **Asignación**.

- Ventana del proyecto (*ediciones Professional y Enterprise*)

  Muestra/oculta la ventana **Proyecto**. Para saber más sobre proyectos consulte Proyectos [98].

- Ventanas del depurador (*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 [812].


© 2019-2025 Altova GmbH

## 17.11 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 configuraciones que son aplicables en varias aplicaciones de Altova. Para más información, consulte Recursos globales de Altova 856.

- Configuración activa

  Permite seleccionar la configuración de recurso global que está activa en una lista que enumera todas las configuraciones definidas previamente. Para crear y configurar diferentes tipos de recursos globales, consulte Recursos globales de Altova 856.

- Crear asignación invertida

  Crea una asignación invertida a partir de la asignación activa que será la base de una asignación nueva. Esto significa que el componente de origen pasa a ser el componente de destino y viceversa. Tenga en cuenta que 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. Por lo tanto, puede que tenga que editar a mano la asignación nueva.

  Estos son los datos que se conservan:

  - Las conexiones directas entre los componentes
  - Las conexiones directas entre los componentes en una asignación encadenada
  - El tipo de conexión 52: estándar, mixto, copia total
  - La configuración de componentes de paso a través
  - Los componentes de BD (*en las ediciones Professional y Enterprise*)

  Estos son los datos que *no* se conservan:

  - Las conexiones a través de funciones, filtros, etc.
  - Las funciones definidas por el usuario
  - Los componentes de servicio web (*Enterprise Edition*)

- Gestor de taxonomías XBRL (*Enterprise Edition*)

  El Gestor de taxonomías XBRL es una herramienta que permite instalar y administrar taxonomías XBRL.

- Gestor de esquemas XML

  El Gestor de esquemas XML es una herramienta de Altova que ofrece una forma centralizada de instalar y gestionar esquemas XML (DTDs para XML y Esquemas XML) para usarlos en todas las aplicaciones de Altova compatibles con XBRL. Para más información, consulte el Gestor de esquemas 157.

- Personalizar

  Esta opción permite personalizar la interfaz gráfica del usuario de MapForce, incluidas las barras de herramientas que aparecen en la interfaz. También permite personalizar los menús 1124 y las teclas de

acceso rápido[1125].

- Restaurar barras de herramientas y ventanas

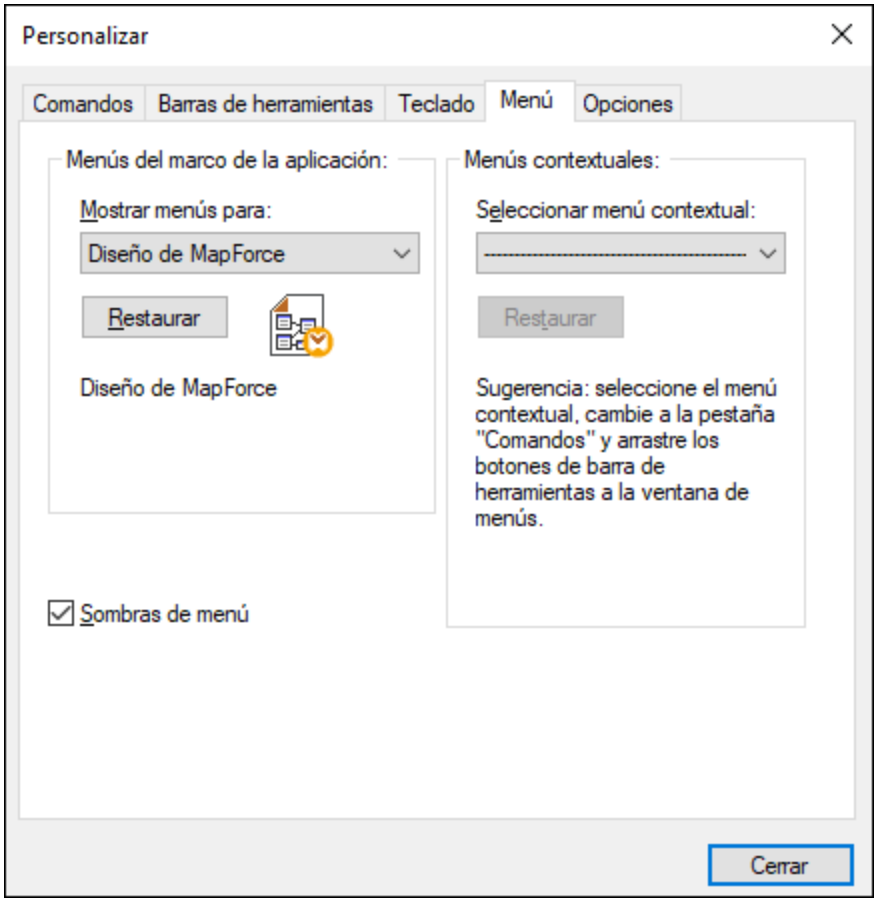
  Restaura las barras de herramientas, las ventanas de ayuda de entrada, las ventanas acopladas, etc. a sus valores predeterminados. Es necesario reiniciar MapForce para que los cambios surtan efecto.

- Opciones

  Al hacer clic, se abre el cuadro de diálogo **Opciones** que permite cambiar la configuración predeterminada de MapForce. Para más información consulte Opciones[1128].

## 17.11.1 Personalizar menús

Puede personalizar los menús estándar de MapForce, así como los menús contextuales (p.ej., 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*).




 © 2019-2025 Altova GmbH

*Personalizar menús*
La 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 menú 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**.

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.

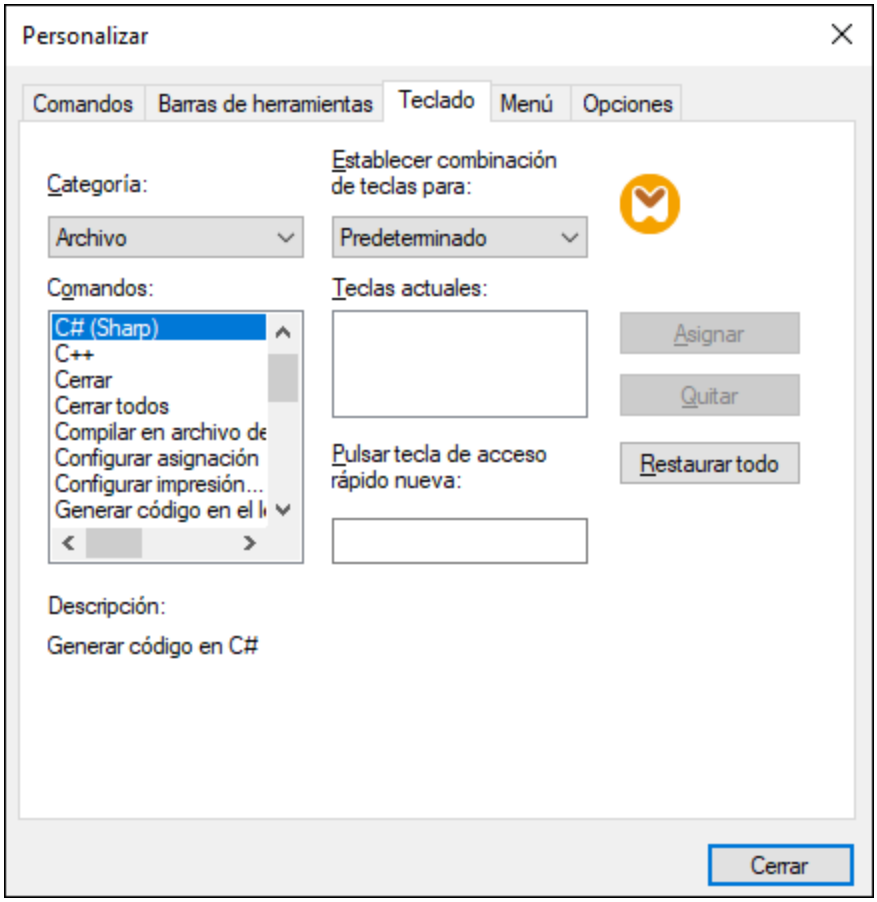
Puede restaurar cualquier menú contextual a su estado original de instalación; para ello selecciónela en la lista desplegable **Seleccionar menú contextual** y haga clic en el botón **Restaurar**.

*Sombras de menú*
Marque la casilla *Sombras de menú* para que todos los menús tengan sombra.

## 17.11.2 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**). Para eliminar una combinación de teclas de la lista *Teclas actuales*, haga clic en ella y pulse **Quitar**.

**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:

| | |
|---|---|
| F1 | Menú Ayuda |
| F2 | Marcador siguiente (panel *Resultados*) |
| F3 | Buscar Siguiente |
| F10 | Activar barra de menú |
| + del teclado numérico | Expandir el nodo actual |

 © 2019-2025 Altova GmbH

| | |
|---|---|
| - del teclado numérico | Contraer el nodo actual |
| * del teclado numérico | Expandir todo a partir del nodo actual |
| | |
| CTRL + TAB | Cambiar de una asignación a otra |
| CTRL + F6 | Recorrer todas las ventanas abiertas |
| CTRL + F4 | Cerrar la asignación activa |
| | |
| Alt + F4 | Cerrar MapForce |
| Alt + F, F, 1 | Abrir el último archivo |
| Alt + F, T, 1 | Abrir 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 | Confirmar una selección |

**Teclas del panel Resultados**

| | |
|---|---|
| CTRL + F2 | Insertar/Quitar marcador |
| F2 | Siguiente marcador |
| Mayús + F2 | Marcador anterior |
| CTRL + Mayús + F2 | Quitar todos los marcadores |

**Teclas de zoom**

| | |
|---|---|
| CTRL + rueda del ratón hacia adelante | Acercarse |
| CTRL + rueda del ratón hacia detrás | Alejarse |
| CTRL + 0 (cero) | Restablecer el nivel de zoom |

## 17.11.3 Opciones

Puede cambiar las opciones generales y preferencias de MapForce con el comando **Herramientas | Opciones**. A continuación, describimos todas 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.

- En la sección *Visualización de asignaciones* puede configurar estas opciones:

  - Habilita y deshabilita el fondo degradado del panel Asignación (*Mostrar fondo degradado*).
  - Puede limitar la visualización de anotaciones a N líneas (*Limitar la presentación de anotaciones a*) Por ejemplo, si ha configurado esta opción con el valor 2 y el texto de la anotación tiene 3 líneas, en la asignación solamente se verán las dos primeras líneas. Esta opción también afecta a las instrucciones SELECT que estén visibles en los componentes.
  - También puede limitar la visualización de comentarios de componentes 34 a N líneas (*Limitar la visualización de comentarios de componentes a N líneas*) Por ejemplo, si ha limitado la visualización de comentarios a una línea y un comentario contiene más de una línea, en la caja sólo se verá la primera línea. Si configura la propiedad con el valor `0` no se verá ningún comentario. Recuerde que la opción *Limitar la visualización de comentarios* no afecta a los comentarios de los componentes 36.

- *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 41).

  *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 el panel 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 833.

 © 2019-2025 Altova GmbH

*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 66.

Edición

Estas son las opciones disponibles en esta página:

- *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 42).

- *Eliminación inteligente de componentes:* MapForce permite conservar conexiones aunque se hayan eliminado algunos componentes de transformación 34. 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 63.

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 Generación de código 68, Generación 1131 y Compilar asignaciones de datos en archivos de ejecución de MapForce Server 839 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 1132.

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 1133.

- Proxy de red

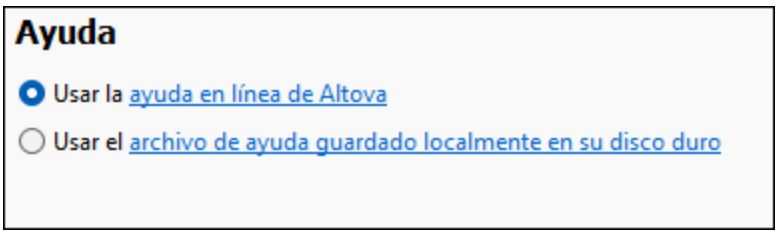
  La sección *Proxy de red* permite personalizar la configuración del proxy de red. Esta opción afecta a cómo se conecta la aplicación a Internet. Por defecto, la aplicación usa la configuración del proxy del sistema, por lo que no necesita configurarlo para que funcione. Para más detalles consulte Configuración del proxy de red 1139.

- Ayuda

  MapForce contiene la ayuda (el manual del usuario) en dos formatos:

  - La ayuda en línea, en formato HTML, que puede encontrar en el sitio web de Altova. Para acceder a la ayuda en línea necesita tener acceso a Internet.
  - Un archivo PDF de ayuda que se instala en el equipo al instalar MapForce. La versión local es un PDF llamado `MapForce.pdf` que puede encontrar en la carpeta de la aplicación (en el directorio Archivos de programa). Si no tiene acceso a Internet siempre puede abrir el archivo local de ayuda.

  La opción Ayuda (*imagen siguiente*) permite seleccionar cuál de los dos formatos se abre al hacer clic en el comando **Ayuda (F1)** del menú **Ayuda**.



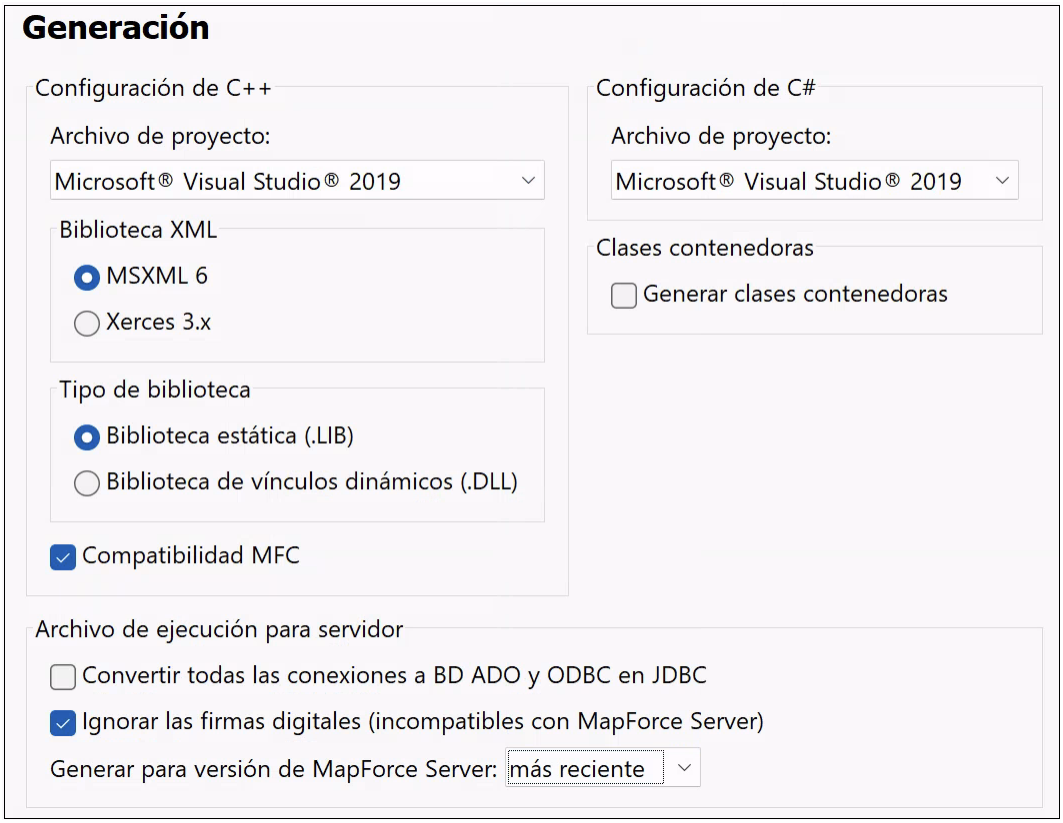

  Puede cambiar esta opción en cualquier momento. Los enlaces de esta sección (*imagen anterior*) abren

 © 2019-2025 Altova GmbH

la ayuda en el formato que haya seleccionado.

## 17.11.3.1 Generación de código

La sección *Generación* del cuadro de diálogo **Opciones** le permite configurar varias opciones de generación de código y de ejecución de servidor (*ver a continuación*).




A continuación describimos las opciones disponibles.

- Configuración de C++

  Esta sección 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#

Si la solución generada necesita apuntar a una plataforma concreta, seleccione la opción correspondiente de *Microsoft .NET <version>* en la lista desplegable. Si necesita usar la plataforma .NET Framework para un proyecto de Visual Studio concreto, seleccione la opción correspondiente de *Microsoft Visual Studio <version>*.

- Clases contenedoras

  Permite generar clases contenedoras para esquemas XML. Para más detalles consulte el apartado Generar código a partir de esquemas XML o DTD 928.

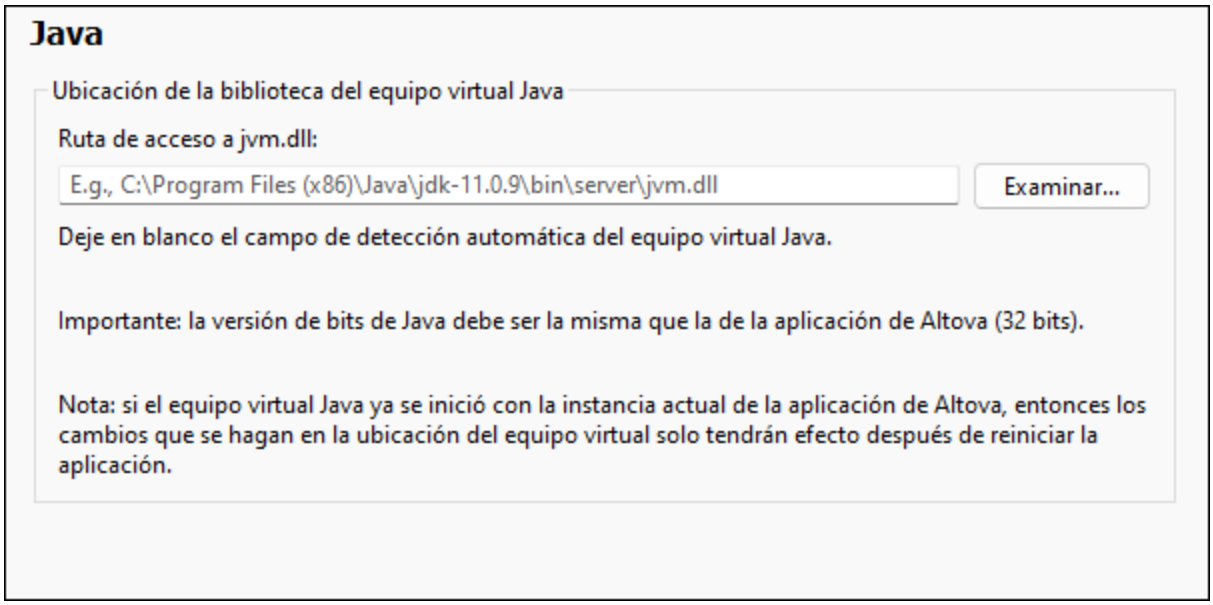
- Archivo de ejecución del servidor

  Estas opciones son relevantes cuando se compilan asignaciones en archivos de ejecución de MapForce Server. Para más información consulte el apartado Compilar asignaciones en archivos de ejecución de MapForce Server 839.

## 17.11.3.2 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.




 © 2019-2025 Altova GmbH

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.

## 17.11.3.3 Base de datos

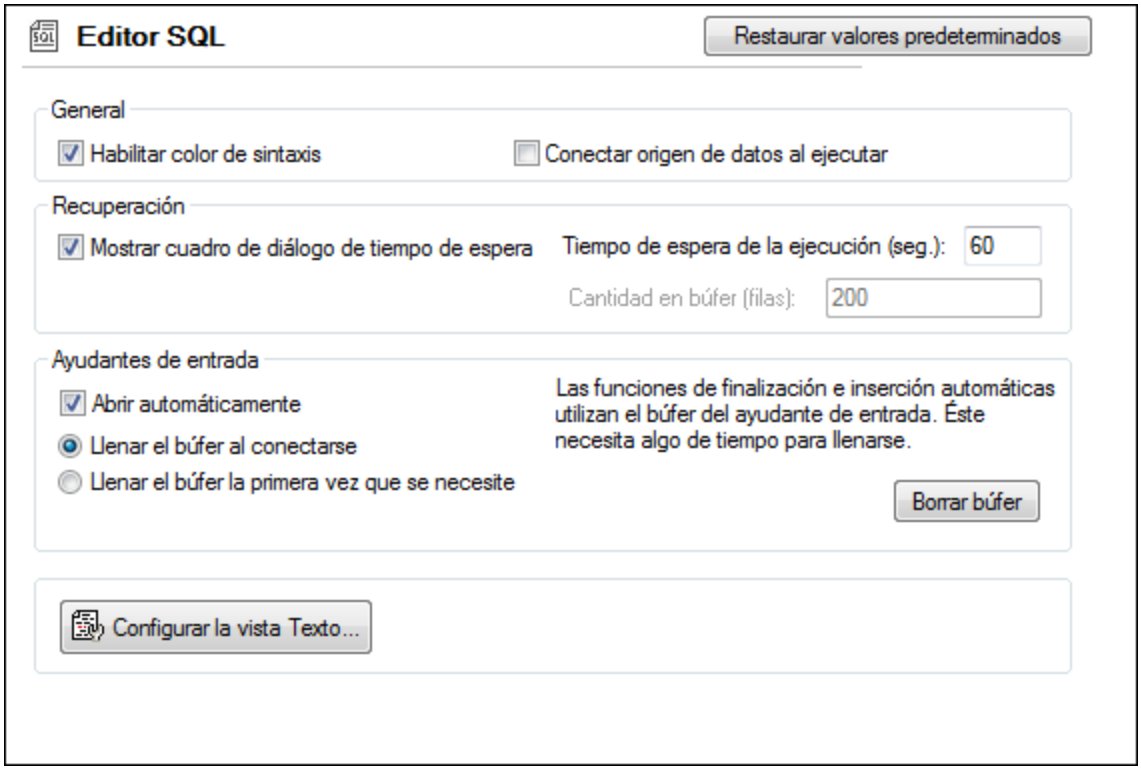
En esta sección explicamos cómo configurar varias de las opciones del editor SQL. Puede acceder a estas opciones de varias formas:

- 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.




- *General:* 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.

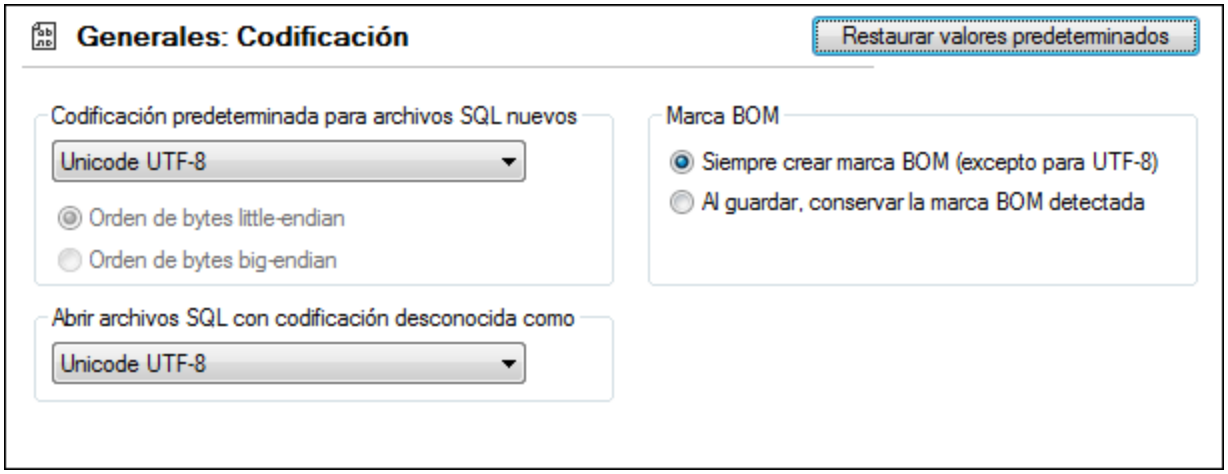
- *Ayudantes de entrada:* Para habilitar la finalización automática cuando escriba instrucciones SQL, marque la casilla *Abrir automáticamente* (véase también Finalización automática 251). Puede elegir cuándo rellenar el búfer del ayudante de entrada: al conectarse a un origen de datos o cuando el búfer 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 71.

## Codificación

En la sección *Codificación* puede especificar las opciones de codificación para los archivos SQL creados o abiertos con el editor SQL (*imagen siguiente*).

 © 2019-2025 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.

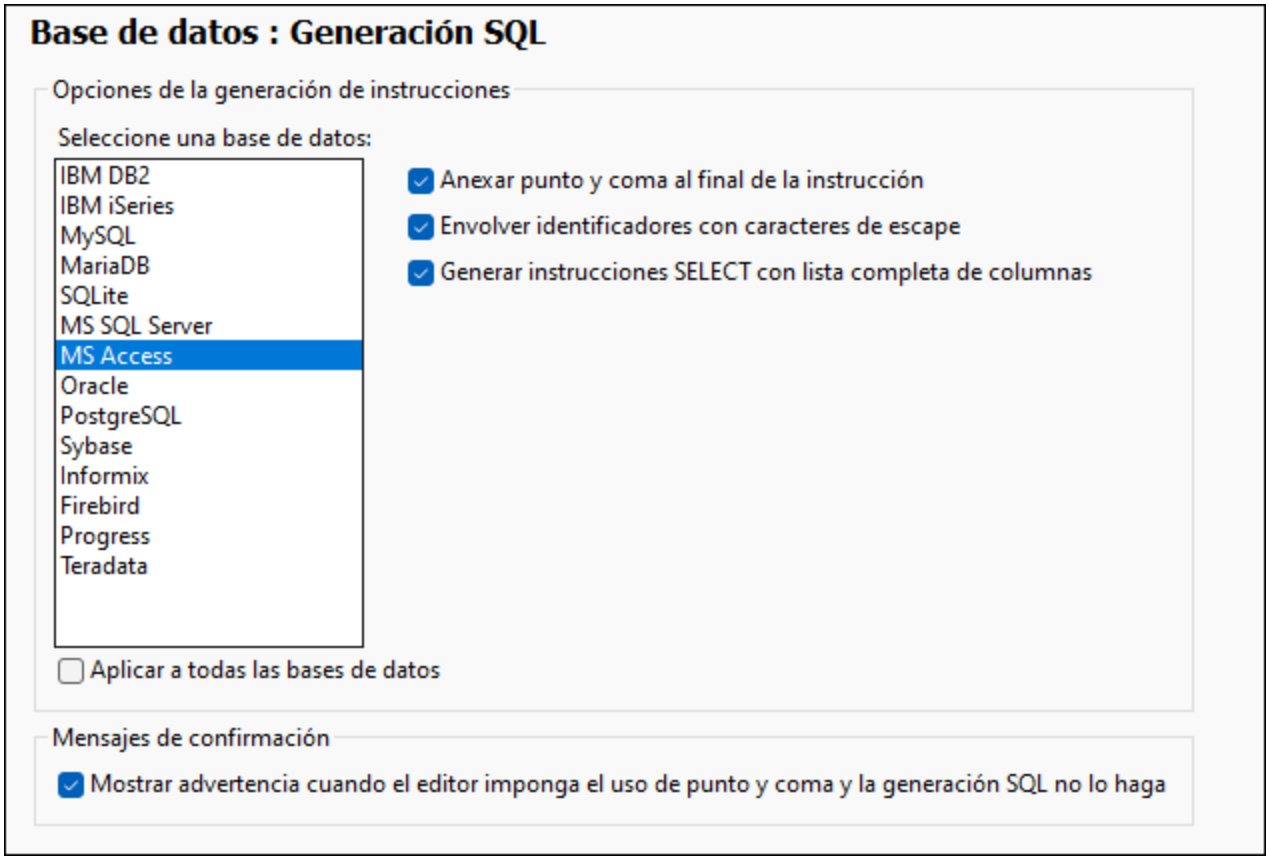
- *Abrir archivos SQL con codificación desconocida como:* Puede seleccionar la codificación con la que quiere abrir un archivo SQL con codificación desconocida.

- *Marca 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 SQL

En la sección *Generación SQL* puede indicar la sintaxis de generación de las instrucciones SQL para varios tipos de BD (*imagen siguiente*).

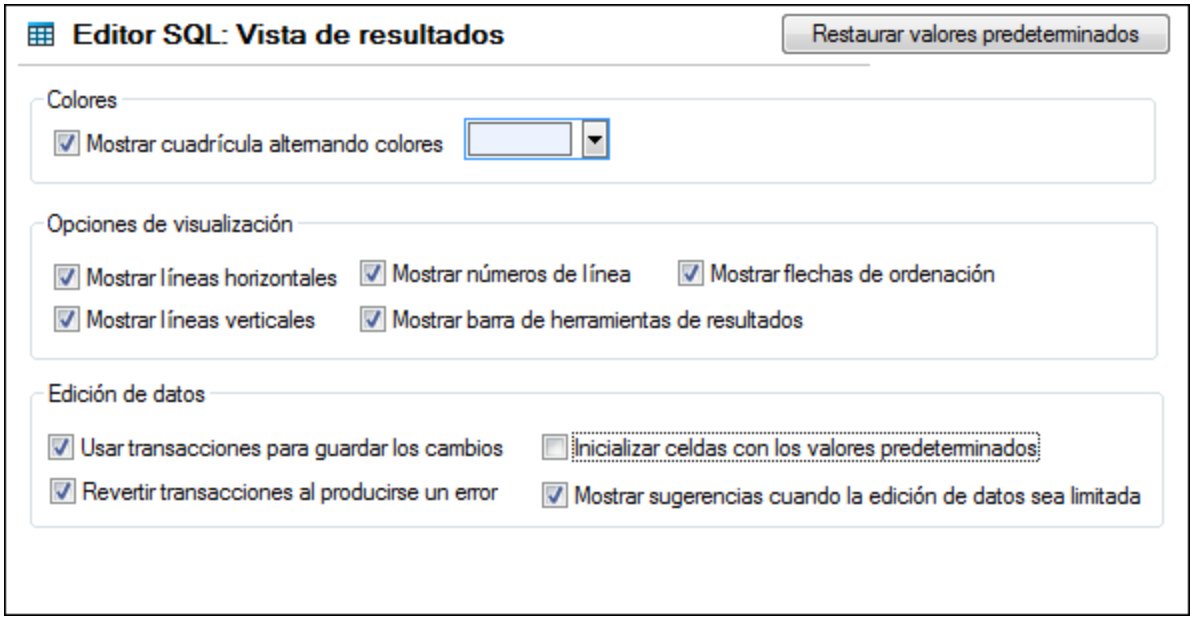



Para definir las preferencias de sintaxis para una BD en concreto, selecciónela en la lista y después habilite o deshabilite las opciones de la derecha. También puede aplicar las mismas opciones a todas las bases de datos (con la casilla *Aplicar a todas las bases de datos*). Recuerde que si usa la misma 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*).

 © 2019-2025 Altova GmbH




Marque la casilla *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**.

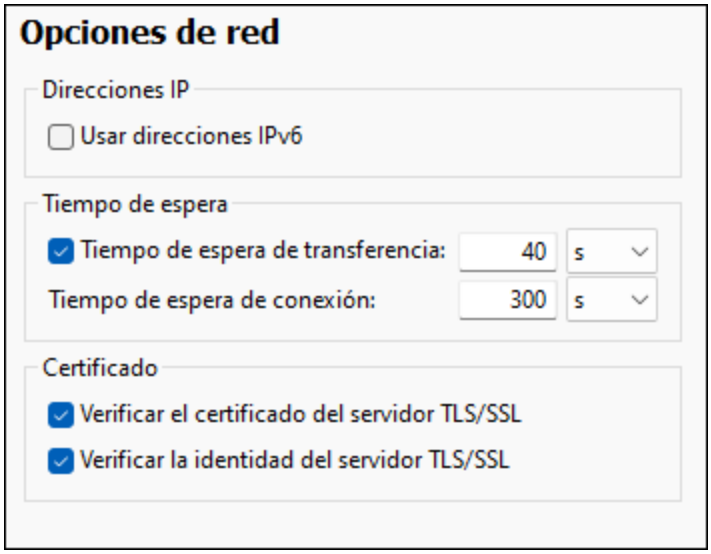
El grupo de opciones *Edición de datos* sirve para configurar las transacciones, para definir si las celdas se deben rellenar con valores predeterminados y para especificar si la aplicación debe mostrar una sugerencia cuando la edición de datos sea limitada.

## Fuentes de la vista Texto

En la *Fuentes de la vista 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 de la vista Texto** desde el panel **Consulta de BD** puede hacer clic en el botón **Restaurar valores predeterminados** para volver a la configuración original.

## 17.11.3.4 Opciones de red

La sección **Opciones de red** (*imagen siguiente*) permite configurar las opciones de red.




*Direcciones IP*
Cuando los nombres de host se resuelven en más de una dirección en redes mixtas IPv4/IPv6, marcar esta casilla indica que se deben usar las direcciones IPv6. Si no se marca esta casilla en dichos entornos y hay

direcciones IPv4 disponibles, se usan direcciones IPv4.

*Tiempo de espera*

- *Tiempo de espera de transferencia:* Si se alcanza este límite al transferir dos paquetes de datos consecutivos (enviados o recibidos), se anula la transferencia al completo. Puede indicar los valores en segundos [s] o milisegundos [ms]; el valor predeterminado son 40 segundos. Si no se marca esta opción no existe ningún límite de tiempo para anular la transferencia.
- *Tiempo de espera de conexión:* Este es el límite de tiempo en el que debe establecerse la conexión, incluido el tiempo que se tarda en establecer la comunicación (handshake). Puede indicar los valores en segundos [s] o milisegundos [ms]; el valor predeterminado son 300 segundos. Este tiempo de espera no se puede deshabilitar.

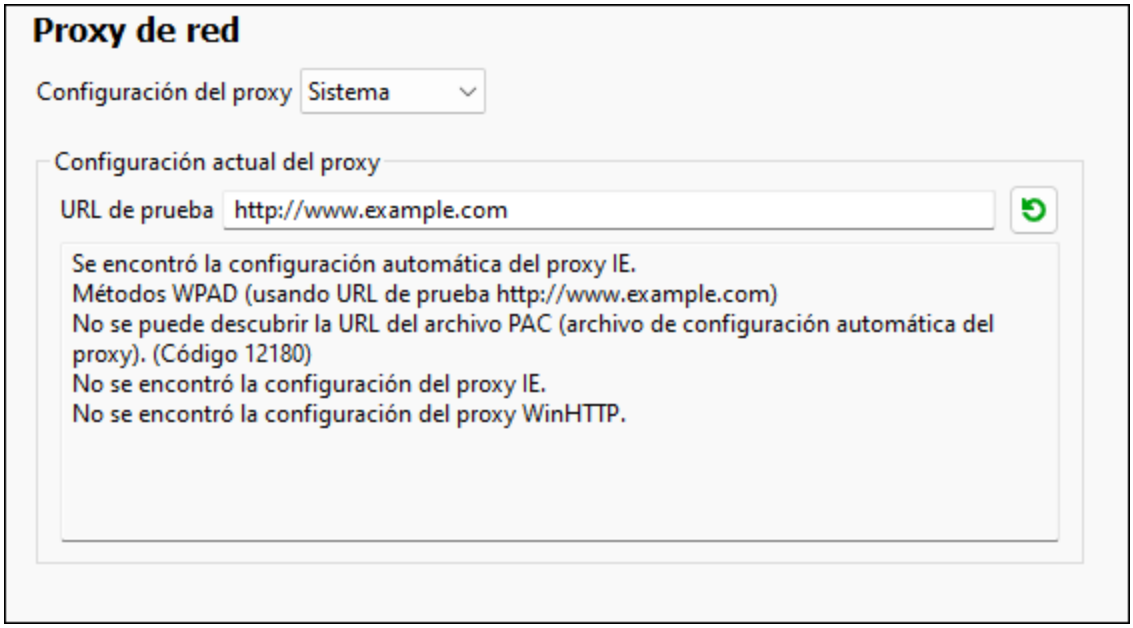
*Certificado*

- *Verificar el certificado del servidor TLS/SSL:* Si se marca esta opción se comprueba la autenticidad del certificado del servidor, para lo que se comprueba la cadena de firmas digitales hasta que se alcanza un certificado raíz de confianza. Esta opción está marcada por defecto. Si no se marca esta opción, la comunicación no es segura y no se detecta si hay ataques (por ejemplo, ataques de suplantación de identidad). Tenga en cuenta que esta opción no comprueba si el certificado pertenece al servidor con el que se está comunicando. Para habilitar la seguridad al completo debe marcar las casillas de certificado y de identidad (*véase la opción siguiente*).
- *Verificar la identidad del servidor TLS/SSL:* Si marca esta opción, se comprueba si el certificado pertenece al servidor con el que se quiere establecer la conexión. Para ello se comprueba si el nombre del servidor de la URL es el mismo que el del certificado. Esta opción está marcada por defecto. Si no se marca esta opción, no se comprueba la identidad del servidor. Recuerde que esta opción no habilita la verificación del certificado del servidor. Para habilitar la seguridad al completo debe marcar tanto la casilla de certificado como la de identidad (*véase la opción anterior*).

### 17.11.3.5 Proxy de red

La sección *Proxy de red* permite personalizar la configuración del proxy de red. Esta configuración afecta a cómo la aplicación se conecta a Internet (para validar un documento XML, por ejemplo). Por defecto, la aplicación usa la configuración del proxy del sistema, por lo que no necesita configurarlo para que funcione. No obstante, si es necesario, puede configurar un proxy de red alternativo seleccionando, en el cuadro combinado *Configuración del proxy*, la opción *Automática* o *Manual*.

**Nota:** la configuración del proxy de red es común a todas las aplicaciones de Altova MissionKit. Por consiguiente, cualquier cambio que haga en la configuración del proxy de una de las aplicaciones afectará al resto.




*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:

- *Opciones 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:* Especifique una URL HTTP de un script de configuración automática del proxy (**`.pac`**) que se utilizará para la configuración del proxy.
- *Volver a cargar:* Restaura y vuelve a cargar la configuración actual automática 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`**). No es necesario que el esquema sea el mismo que el protocolo correspondiente si el proxy es compatible con el esquema.

 © 2019-2025 Altova GmbH

Existen las siguientes opciones:

- *Proxy HTTP:* Utiliza el nombre de host y el puerto especificados para el protocolo HTTP. Si está seleccionada la opción *Utilizar este servidor proxy para todos los protocolos*, entonces se usa el proxy HTTP especificado para todos los protocolos.
- *Proxy SSL:* Utiliza el nombre de host y el puerto especificados para el protocolo SSL.
- *Ningún proxy para:* Una lista separada por punto y coma (`;`) de nombres de host totalmente cualificados, nombres de dominio o direcciones IP para los hosts que deben utilizarse sin 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 usar el servidor proxy para direcciones locales:* Si está marcada, añade `<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 un carácter de 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. Puede actualizarse con el botón **Actualizar** situado a la derecha del campo *URL de prueba* (por ejemplo, al cambiar la URL de prueba o cuando se ha modificado la configuración del proxy).

- *URL de prueba:* Se puede utilizar una URL de prueba para ver qué proxy se utiliza para esa URL específica. No se realiza ninguna E/S con esta URL. Este campo no debe estar vacío si se utiliza la configuración automática del proxy (ya sea a través de la opción *Utilizar la configuración del proxy del sistema* o *Configuración automática del proxy*).

## 17.12 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

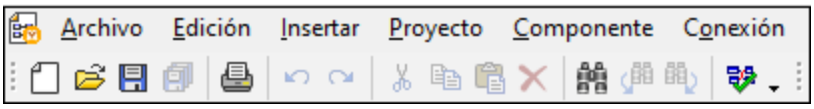
  Este comando reorganiza todas las ventanas abiertas **en forma de mosaico horizontal**, mostrando todas las ventanas a la vez.

- En mosaico vertical

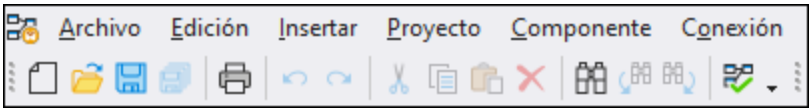
  Este comando reorganiza todas las ventanas abiertas **en forma de mosaico vertical**, mostrando todas las ventanas a la vez.

- Tema clásico/claro/oscuro

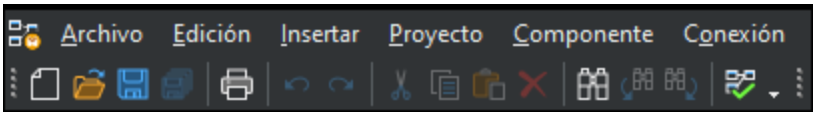
  MapForce permite elegir entre estos temas: *clásico*, *claro* y *oscuro*. A continuación puede ver ejemplos de los tres temas. La opción predeterminada es el tema clásico.




*Tema clásico*




*Tema claro*




*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.

- Ventanas

  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.


© 2019-2025 Altova GmbH

# 17.13 Ayuda

En este apartado puede consultar todos los comandos del menú **Ayuda**.

Ayuda (F1)

El comando **Ayuda (F1)** abre la documentación de ayuda de la aplicación (el manual del usuario). La ayuda que se abre de forma predeterminada es la ayuda en línea en formato HTML.

Si no tiene acceso a Internet o por algún motivo no quiere usar la ayuda en línea, siempre puede usar la versión local del manual del usuario. La versión local es un PDF llamado `MapForce.pdf` que puede encontrar en la carpeta de la aplicación (en el directorio Archivos de programa).

Si quiere cambiar el formato predeterminado (ayuda en línea o PDF local) puede hacerlo en la sección Ayuda del cuadro de diálogo Opciones (comando de menú **Herramientas | Opciones**).

Activación del software

*Asignar una licencia al producto*
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 de evaluación gratuita***. Cuando inicie el software por primera vez, tras haberlo descargado e instalado, aparecerá el cuadro de diálogo **Activación del software**. Este cuadro de diálogo incluye un botón para solicitar una licencia de evaluación gratuita. Haga clic en este botón para obtener su licencia. Al hacer clic en este botón, se generará un código hash para el ID de su equipo que se enviará a Altova a través de HTTPS. La información de la licencia se devolverá al equipo a través de una respuesta HTTP. Una vez la licencia se haya creado con éxito, aparecerá un cuadro de diálogo al respecto en su aplicación de Altova. Al hacer clic en **Aceptar** en este cuadro de diálogo se activará el software durante 30 días **en ese equipo particular**.

- ***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 sus 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. Si adquiere una licencia instalada para *N* ordenadores, la licencia permite utilizar el software hasta en esta cantidad de ordenadores. De la misma manera, una licencia de usuario concurrente para *N* usuarios concurrentes permite a *N* usuarios ejecutar el software de forma concurrente. (El software puede instalarse en 10N ordenadores). 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 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 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. El archivo de la licencia 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**. Puede escribir la ruta de acceso o navegar hasta el archivo de licencia, y luego hacer clic en **Aceptar**.
- Guardar el archivo de licencia (**`.altova_licenses`**) en su equipo y cargarlo desde esa ubicación a su Altova LicenseServer. Puede elegir entre estas dos opciones: (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.*

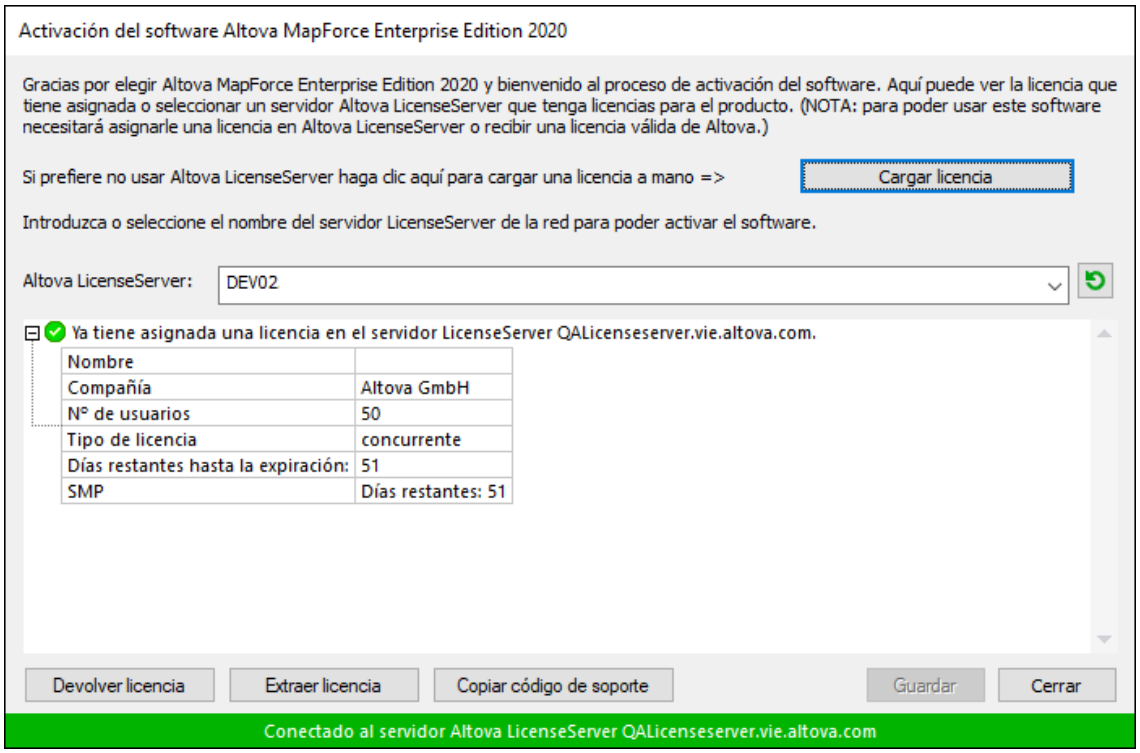
El cuadro de diálogo **Activación del software** (*imagen siguiente*) se abre con el comando **Ayuda | Activación del software**.

*Activar el software*
Puede activar el software registrando la licencia en el cuadro de diálogo "Activación del software" o asignando una licencia a través de Altova LicenseServer (*ver detalles más abajo*).

- *Registrando la licencia en el cuadro de diálogo "Activación del software"*. En el cuadro de diálogo, 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 **Activación del software**. 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 (*ver imagen siguiente donde figura el cuadro de diálogo en Altova XMLSpy*). Haga clic en el botón **Guardar** para adquirir la licencia.

 *© 2019-2025 Altova GmbH*




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 de forma automática al de licencias una vez ha finalizado el periodo de extracción. 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: (I) En el cuadro de diálogo **Activación del software** haga clic en el botón **Extraer licencia** (*imagen anterior*). (Ii) Aparece el cuadro de diálogo **Extracción de licencias**. Seleccione el periodo de extracción deseado y haga clic en **Extraer**. Así se extraerá la licencia. Ahora, después de haber extraído una licencia, ocurren dos cosas: (i) El cuadro de diálogo **Activación del software** muestra información sobre la extracción de la licencia, incluida la fecha y la 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 en cualquier momento dado, 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.

Si la licencia que extrae es una licencia de tipo instalado o una licencia de usuario concurrente, entonces esta se extrae al equipo y está disponible para el usuario que extrajo la licencia. Si la licencia que extrae es una licencia de usuario designado, entonces esta se extrae a la cuenta de Windows del usuario designado. Se pueden extraer licencias en equipos virtuales pero no para escritorios virtuales (en una virtualización de escritorio). Tenga en cuenta que al extraer una licencia de usuario designado, los datos que identifican esa extracción de licencia se almacenan en el perfil del usuario. Para que funcione la extracción de licencias, el perfil del usuario debe estar almacenado en el equipo local que se utilizará para trabajar sin conexión. Si el perfil del usuario se encuentra en una ubicación no local (como un archivo compartido), la extracción se considerará no válida a la hora de iniciar la aplicación de Altova.

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 las características administrativas que ofrece para la gestión de grandes volúmenes de licencias de Altova. Altova LicenseServer puede descargarse gratis del sitio web de Altova. Para más información sobre Altova LicenseServer, 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 **Formulario de pedido**, que le lleva directamente a la tienda en línea 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.

## Soporte técnico


© 2019-2025 Altova GmbH

Es un enlace al centro de soporte técnico de Altova en Internet. 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.

- Descargar herramientas gratis y componentes

  Es un enlace al centro de descargas de componentes del sitio web de Altova. Aquí puede descargar una variedad de 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. Aquí 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 y muestra el número de versión del producto. Si usa la versión de 64 bits de MapForce, esto se ve en el nombre de la aplicación, que lleva el sufijo (x64). La versión de 32 bits no lleva ningún sufijo.

# 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 1149
- Información sobre motores 1153
- Datos técnicos 1258
- Información sobre licencias 1261



© 2019-2025 Altova GmbH

# 18.1 Notas sobre compatibilidad

MapForce es una aplicación de 32/64 bits para Windows compatible con los sistemas operativos siguientes:

- Windows 10, Windows 11
- Windows Server 2016 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 886.
- Eclipse 2024-12 (4.34), 2024-09 (4.33), 2024-06 (4.32), 2024-03 (4.31). Véase Complemento de MapForce para Eclipse 889.

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 174.
- Puede generar la documentación de la asignación en Word 2000 o versiones posteriores. Véase Generar la documentación de la asignación 799.

## 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 175.

## 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.

| Característica | XSLT 1.0 | XSLT 2.0 | XSLT 3.0 | XQuery | C++ | C# | Java | BUILT-IN |
|---|---|---|---|---|---|---|---|---|
| Pasar parámetros a la asignación 369 | ● | ● | ● | ● | ● | ● | ● | ● |
| Suministrar los nombres de los archivos de entrada dinámicamente desde la asignación 784 | ● | ● | ● | ● | ● | ● | ● | ● |
| Suministrar nombres de archivos comodín como entrada de la asignación 784 1 | | ● | ● | ● | ● | ● | ● | ● |
| Generar los nombres de los archivos de salida dinámicamente desde la asignación 784 | | ● | ● | | ● | ● | ● | ● |


© 2019-2025 Altova GmbH

| Característica | XSLT 1.0 | XSLT 2.0 | XSLT 3.0 | XQuery | C++ | C# | Java | BUILT-IN |
|---|---|---|---|---|---|---|---|---|
| Obtener valores de cadena de una asignación 380 | ● | ● | ● | | ● | ● | ● | ● |
| Variables | | ● | ● | ● | ● | ● | ● | ● |
| Ordenar componentes 426 | | ● | ● | ● | | | | ● |
| Agrupar funciones 595 | | ● | ● | | ● | ● | ● | ● |
| Filtros 432 | ● | ● | ● | ● | ● | ● | ● | ● |
| Combinar componentes 397 | | | | | | | | ● |
| Componentes Asignación de valores 445 | ● | ● | ● | ● | ● | ● | ● | ● |
| Valores predeterminados y funciones 469 | | | | | | | | ● |
| Asignaciones: excepciones 457 | | ● | ● | ● | ● | ● | ● | ● |
| Redistribuir y serializar cadenas de texto 791 2 | | | | | | | | ● |
| Nombres de nodo dinámicos 763 | | ● | ● | ● | ● | ● | ● | ● |
| BD: inserciones masivas 276 | | | | | | | | ● |
| Instrucciones SQL SELECT para BD sin parámetros de entrada 259 | | | | | ● | ● | ● | ● |
| Instrucciones SQL SELECT para BD con parámetros de entrada 259 | | | | | | | | ● |
| BD: procedimientos almacenados 315 | | | | | | | | ● |

| Característica | XSLT 1.0 | XSLT 2.0 | XSLT 3.0 | XQuery | C++ | C# | Java | BUILT-IN |
|---|---|---|---|---|---|---|---|---|
| Gestionar excepciones de BD 288 3 | | | | | ● | ● | ● | ● |
| Seguimiento de BD y registro de errores 251 | | | | | | | | ● |
| Generar archivos de ejecución de MapForce Server 839 | | | | | | | | ● |
| Implementar asignaciones en FlowForce Server 842 | | | | | | | | ● |
| Leer datos de archivos binarios 662 | | | | | | | | ● |
| Escribir datos en archivos binarios 665 | | | | | | | | ● |

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.


© 2019-2025 Altova GmbH

# 18.2 Información sobre motores

En esta sección, ofrecemos 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 1153
- XSLT 2.0 1154
- XQuery 1.0 1155


### 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. Tenga en cuenta la información sobre la implementación que se ve a continuación.

#### 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 1154
- Compatibilidad con versiones antiguas 1154
- Espacios de nombres 1154
- Compatibilidad con esquemas 1155
- Comportamiento propio de esta implementación 1155


### 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 es relevante cuando se utiliza el motor XSLT 2.0 para procesar una hoja de estilos o instrucción 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.

 © 2019-2025 Altova GmbH

- 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 (p.ej. `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 `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 de algunas 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`. Ejemplo: `xs:base64Binary(unparsed-text('chart.png', 'x-binarytobase64'))`.

**`unparsed-text-available`**
El argumento `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 1156
- Compatibilidad con esquemas 1156
- Codificación 1156
- Espacios de nombres 1156
- Fuentes XML y validación 1157
- Comprobación de tipos estática y dinámica 1157

- Módulos biblioteca 1157
- Funciones externas 1158
- Intercalaciones 1158
- Precisión de datos numéricos 1158
- Compatibilidad con instrucciones XQuery 1158
- Comportamiento propio de esta implementación 1158


## 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 es **compatible con 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` |
| Instancia de esquema | `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 (véase `fn:`) es el espacio de nombres de funciones predeterminado de XQuery, no es necesario usar el prefijo `fn:` cuando se invocan funciones integradas (p.ej. `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 (p.ej.: `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. (p.ej.: `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

 © 2019-2025 Altova GmbH

consulta. (p.ej.: `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 esquema, p.ej. `fn:string` y `fn:boolean`. (Se definen tanto `xs:string` como `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 aquí1159. Para usar una intercalación concreta, indique su URI tal y como aparece en la lista de intercalaciones compatibles1159. 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.

## Comportamiento propio de esta implementación

A continuación puede ver una descripción de cómo enfocan los motores XQuery y XQuery Update 1.0 los aspectos relativos a la implementación de ciertas funciones.

**`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`. Ejemplo: `xs:base64Binary(unparsed-text('chart.png', 'x-binarytobase64'))`.

**`unparsed-text-available`**

El argumento `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`.


© 2019-2025 Altova GmbH

## 18.2.2 Funciones XSLT 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.

En esta sección describimos las funciones de extensión XPath/XQuery que han sido creadas por Altova para proporcionar operaciones adicionales, así como otras funciones de extensión 1239. Estas funciones 1160 pueden ser calculadas por los motores XSLT y XQuery de Altova basándose en las reglas descritas en esta sección. Para obtener información sobre las funciones XPath/XQuery regulares, consulte la Referencia de funciones.

### Aspectos generales

Es necesario tener en cuenta estos puntos generales:

- A las funciones de las bibliotecas de funciones principales 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 `http://www.w3.org/2005/xpath-functions`, que es el que se especifica en las especificaciones de las funciones XPath y XQuery. 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.
- Se usa la colación de punto de código de Unicode para todas las comparaciones de cadenas de texto.
- Los resultados que son QName se serializan de esta forma `[prefijo:]nombrelocal`.

*Precisión de xs:decimal*

La precisión se refiere a la cantidad de dígitos del número; la especificación requiere un mínimo de 18 dígitos. Para operaciones de división que dan un resultado de tipo `xs:decimal`, la precisión es de 19 dígitos tras el punto decimal sin redondeos.

*Zona horaria implícita*

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. La zona horaria implícita se toma del reloj del sistema y para probar cuál es su valor puede utilizar la función `implicit-timezone()`.

*Intercalaciones*

La colación predeterminada es la colación de punto de código de Unicode, que compara cadenas de texto basándose en su punto de código. El motor usa el algoritmo de colación de 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 `max` y `min`) se harán según la intercalación especificada. Si no se ha indicado ninguna colación se usará la colación predeterminada de punto de código de Unicode.

| Lenguaje | URIs |
|---|---|
| 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: Japonés | 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 está obsoleto en XPath 2.0. Sin embargo, sí que se admite el uso del espacio de nombres. Para acceder a la información sobre el espacio de nombres con mecanismos de XPath 2.0, utilice las funciones `in-scope-prefixes()`, `namespace-uri()` y `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.

 © 2019-2025 Altova GmbH

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** |

## Cómo usar las funciones de extensión de Altova

Para poder usar las funciones de extensión de Altova debe declarar el espacio de nombre correspondiente (*el primer resaltado en el extracto de código siguiente*) y después usar las funciones de extensión para que se resuelvan como si pertenecieran a ese espacio de nombres (*véase el segundo resaltado*). En el ejemplo siguiente puede ver cómo se usa la función de extensión de Altova **age**.

```
<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:altova="http://www.altova.com/xslt-extensions">
   <xsl:output method="text" encoding="ISO-8859-1"/>
   <xsl:template match="Persons">
      <xsl:for-each select="Person">
         <xsl:value-of select="concat(Name, ': ')"/>
         <xsl:value-of select="altova:age(xs:date(BirthDate))"/>
         <xsl:value-of select="' years&#x0A;'"/>
      </xsl:for-each>
   </xsl:template>
</xsl:stylesheet>
```

## Funciones XSLT 1162

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 XPath/XQuery de fecha y hora 1165
- Funciones XPath/XQuery de geoubicación 1183
- Funciones XPath/XQuery relacionadas con imágenes 1194
- Funciones XPath/XQuery numéricas 1199
- Funciones XPath/XQuery de secuencia 1221
- Funciones XPath/XQuery de cadena 1230
- Funciones XPath/XQuery varias 1237


### 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

 © 2019-2025 Altova GmbH

**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 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 1162 ]

## 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:]


© 2019-2025 Altova GmbH

**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 1162 ]

### 18.2.2.1.2 Funciones XPath/XQuery: 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 1166
- Agregar una duración a xs:date y devolver xs:date 1168
- Agregar una duración a xs:time y devolver xs:time 1170
- Recuperar duraciones y aplicarles formato 1169
- Quitar la zona horaria de las funciones que generan la fecha/hora actual 1171
- Devolver el número de días, horas, minutos y segundos de duraciones 1172
- Devolver el día de la semana de una fecha como número entero 1174
- Devolver el número de semana de una fecha como número entero 1175
- Generar la fecha, la hora y el tipo de duración a partir de los componentes léxicos de cada tipo 1176
- Construir un tipo date, dateTime o a partir de la cadena de entrada 1178
- Funciones para calcular la edad 1179
- Funciones para calcular el tiempo Unix 1181


Funciones por orden alfabético

altova:add-days-to-date 1168
altova:add-days-to-dateTime 1166
altova:add-hours-to-dateTime 1166
altova:add-hours-to-time 1170
altova:add-minutes-to-dateTime 1166
altova:add-minutes-to-time 1170
altova:add-months-to-date 1168
altova:add-months-to-dateTime 1166
altova:add-seconds-to-dateTime 1166
altova:add-seconds-to-time 1170
altova:add-years-to-date 1168
altova:add-years-to-dateTime 1166
altova:age 1179
altova:age-details 1179
altova:build-date 1176
altova:build-duration 1176
altova:build-time 1176
altova:current-dateTime-no-TZ 1171
altova:current-date-no-TZ 1171
altova:current-time-no-TZ 1171
altova:date-no-TZ 1171
altova:dateTime-from-epoch 1181
altova:dateTime-from-epoch-no-TZ 1181
altova:dateTime-no-TZ 1171
altova:days-in-month 1172
altova:epoch-from-dateTime 1181
altova:hours-from-dateTimeDuration-accumulated 1172
altova:minutes-from-dateTimeDuration-accumulated 1172
altova:seconds-from-dateTimeDuration-accumulated 1172
altova:format-duration 1169
altova:parse-date 1178
altova:parse-dateTime 1178
altova:parse-duration 1169
altova:parse-time 1178
altova:time-no-TZ 1171
altova:weekday-from-date 1174
altova:weekday-from-dateTime 1174
altova:weeknumber-from-date 1175
altova:weeknumber-from-dateTime 1175

[ Subir 1165 ]

## 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

 © 2019-2025 Altova GmbH

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-

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 1165 ]

## 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


© 2019-2025 Altova GmbH

- **altova:add-years-to-date**(xs:date("2014-01-15"), -4) devuelve 2010-01-15

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 1165 ]

## 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:]

**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 1165 ]

## 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

 © 2019-2025 Altova GmbH

**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 1165 ]

## 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*

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 1165 ]

## 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

 © 2019-2025 Altova GmbH

- **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 **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"


© 2019-2025 Altova GmbH

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 1165 ]

## 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`
- `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 1165 ]

## 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


© 2019-2025 Altova GmbH

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.

*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 1165 ]

## 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 1165 ]

## 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


© 2019-2025 Altova GmbH

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 1165 ]

## 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 1165 ]


© 2019-2025 Altova GmbH

### 18.2.2.1.3 Funciones XPath/XQuery: 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

comprendidos entre +180 y -180 (E a W).

| 1 |
|---|
| Grados, minutos y segundos decimales + orientación como sufijo (N/S, E/W)<br>**D°M'S.SS"N/S D°M'S.SS"E/W**<br>*Ejemplo*: **33°55'11.11"N 22°44'66.66"W** |

| 2 |
|---|
| Grados decimales + orientación como sufijo (N/S, E/W)<br>**D.DDN/S D.DDE/W**<br>*Ejemplo*: **33.33N 22.22W** |

| 3 |
|---|
| Grados, minutos y segundos decimales + prefijo (+/-). El signo + para (**N/E**) es opcional<br>**+/-D°M'S.SS" +/-D°M'S.SS"**<br>*Ejemplo*: **33°55'11.11" -22°44'66.66"** |

| 4 |
|---|
| Grados decimales + prefijo (+/-). El signo + para (**N/E**) es opcional<br>**+/-D.DD +/-D.DD**<br>*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 1194 y el atributo @Geolocation 1194 de los metadatos Exif se pueden utilizar para suministrar la cadena de entrada de la geoubicación (*ver ejemplos*).


© 2019-2025 Altova GmbH

*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 (48.858222222222, 24.2945)
- **altova:parse-geolocation**('48°51''29.6"N  24°17''40.2"') devuelve la secuencia de dos xs:decimals (48.858222222222, 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). En los ejemplos de esta sección, las comillas para delimitar la cadena de entrada está resaltada en amarillo (**"**) mientras los indicadores de unidades de escape están resaltados en azul (**""**).

- 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-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 1194 y el atributo de metadatos Exif @Geolocation 1194 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). En los ejemplos de esta sección, las comillas para delimitar la cadena de entrada está resaltada en amarillo (") mientras los indicadores de unidades de escape están resaltados en azul (").

- 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**


© 2019-2025 Altova GmbH

- 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 1194 y el atributo de metadatos Exif @Geolocation 1194 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:*

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). En los ejemplos de esta sección, las comillas para delimitar la cadena de entrada está resaltada en amarillo (") mientras los indicadores de unidades de escape están resaltados en azul (").

- 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:

 © 2019-2025 Altova GmbH

| 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` 1194 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). En los ejemplos de esta sección, las comillas para delimitar la cadena de entrada está resaltada en amarillo (**"**) mientras los indicadores de unidades de escape están resaltados en azul (**""**).

- 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**

*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)<br>**D°M'S.SS"N/S D°M'S.SS"E/W**<br>*Ejemplo*: **33°55'11.11"N 22°44'66.66"W** |

| 2 |
|---|
| Grados decimales + orientación como sufijo (N/S, E/W)<br>**D.DDN/S D.DDE/W**<br>*Ejemplo*: **33.33N 22.22W** |

| 3 |
|---|
| Grados, minutos y segundos decimales + prefijo (+/-). El signo + para (**N/E**) es opcional<br>**+/-D°M'S.SS" +/-D°M'S.SS"** |


© 2019-2025 Altova GmbH

| |
|---|
| *Ejemplo*: **33°55'11.11" -22°44'66.66"** |

| 4 |
|---|
| Grados decimales + prefijo (+/-). El signo + para (**N/E**) es opcional<br>**+/-D.DD +/-D.DD**<br>*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 1194 y el atributo de metadatos Exif @Geolocation 1194 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). En los ejemplos de esta sección, las comillas para delimitar la cadena de entrada está resaltada en amarillo (**"**) mientras los indicadores de unidades de escape están resaltados en azul (**""**).

- 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 |


© 2019-2025 Altova GmbH

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` 1194 y el atributo de metadatos Exif `@Geolocation` 1194 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). En los ejemplos de esta sección, las comillas para delimitar la cadena de entrada está resaltada en amarillo (**"**) mientras los indicadores de unidades de escape están resaltados en azul (**""**).

- 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 |

[ Subir 1183 ]

## 18.2.2.1.4 Funciones XPath/XQuery: 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 |

 © 2019-2025 Altova GmbH

| | |
|---|---|
| *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:

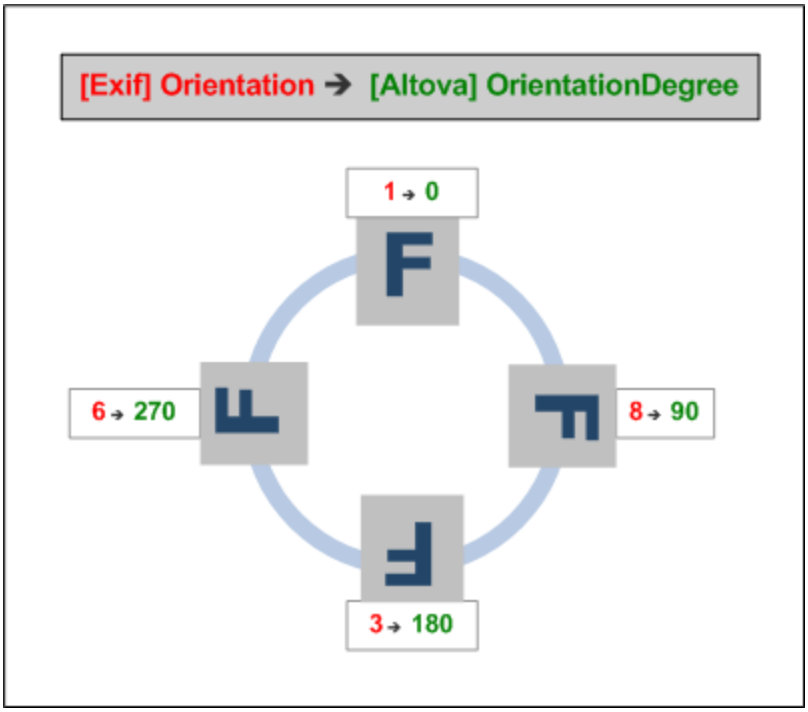
| GPSLatitude | GPSLatitudeRef | GPSLongitude | GPSLongitudeRef | Geolocation |
|---|---|---|---|---|
| 33 51 21.91 | S | 151 13 11.73 | E | 33°51'21.91"S 151° 13'11.73"E |

## *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**.

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


© 2019-2025 Altova GmbH

- BitsPerSample
- Compression
- PhotometricInterpretation
- Orientation
- SamplesPerPixel
- PlanarConfiguration
- YCbCrSubSampling
- YCbCrPositioning
- XResolution
- YResolution
- ResolutionUnit
- StripOffsets
- RowsPerStrip
- StripByteCounts
- JPEGInterchangeFormat
- JPEGInterchangeFormatLength
- TransferFunction
- WhitePoint
- PrimaryChromaticities
- YCbCrCoefficients
- ReferenceBlackWhite
- DateTime
- ImageDescription
- Make
- Model
- Software
- Artist
- Copyright

---

- ExifVersion
- FlashpixVersion
- ColorSpace
- ComponentsConfiguration
- CompressedBitsPerPixel
- PixelXDimension
- PixelYDimension
- MakerNote
- UserComment
- RelatedSoundFile
- DateTimeOriginal
- DateTimeDigitized
- SubSecTime
- SubSecTimeOriginal
- SubSecTimeDigitized
- ExposureTime
- FNumber
- ExposureProgram
- SpectralSensitivity
- ISOSpeedRatings
- OECF
- ShutterSpeedValue
- ApertureValue
- BrightnessValue
- ExposureBiasValue

- MaxApertureValue
- SubjectDistance
- MeteringMode
- LightSource
- Flash
- FocalLength
- SubjectArea
- FlashEnergy
- SpatialFrequencyResponse
- FocalPlaneXResolution
- FocalPlaneYResolution
- FocalPlaneResolutionUnit
- SubjectLocation
- ExposureIndex
- SensingMethod
- FileSource
- SceneType
- CFAPattern
- CustomRendered
- ExposureMode
- WhiteBalance
- DigitalZoomRatio
- FocalLengthIn35mmFilm
- SceneCaptureType
- GainControl
- Contrast
- Saturation
- Sharpness
- DeviceSettingDescription
- SubjectDistanceRange
- ImageUniqueID

---

- GPSVersionID
- GPSLatitudeRef
- GPSLatitude
- GPSLongitudeRef
- GPSLongitude
- GPSAltitudeRef
- GPSAltitude
- GPSTimeStamp
- GPSSatellites
- GPSStatus
- GPSMeasureMode
- GPSDOP
- GPSSpeedRef
- GPSSpeed
- GPSTrackRef
- GPSTrack
- GPSImgDirectionRef
- GPSImgDirection
- GPSMapDatum
- GPSDestLatitudeRef
- GPSDestLatitude
- GPSDestLongitudeRef


© 2019-2025 Altova GmbH

- GPSDestLongitude
- GPSDestBearingRef
- GPSDestBearing
- GPSDestDistanceRef
- GPSDestDistance
- GPSProcessingMethod
- GPSAreaInformation
- GPSDateStamp
- GPSDifferential

[ Subir 1194 ]

### 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")`.

### 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 1199 ]

## 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


© 2019-2025 Altova GmbH

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 1199 ]
[ Subir 1199 ]

### 18.2.2.1.6 Funciones XPath/XQuery: 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 |
| *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

 © 2019-2025 Altova GmbH

*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 Definition

| 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 | Tipo de la propiedad | Valor de la propiedad |
|---|---|---|

| 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 | |


© 2019-2025 Altova GmbH

| | | |
|---|---|---|
| | ("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 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 | Sólo 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 | (Sólo 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" |


© 2019-2025 Altova GmbH

| | | |
|---|---|---|
| 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 definition | Simple Type | |
| item type definition | (Sólo para tipos de lista) Simple Type | |
| member type definitions | (Sólo 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.

 © 2019-2025 Altova GmbH

**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 Definition

| 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) |


© 2019-2025 Altova GmbH

| | | |
|---|---|---|
| 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 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 | Sólo 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" |


© 2019-2025 Altova GmbH

| | | |
|---|---|---|
| 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 | (Sólo 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 definition | Simple Type | |
| item type definition | (Sólo para tipos de lista) Simple Type | |
| member type definitions | (Sólo 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) | |


© 2019-2025 Altova GmbH

| | | |
|---|---|---|
| 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 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 Definition

| 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": | |


© 2019-2025 Altova GmbH

| | | |
|---|---|---|
| | "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 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 |


© 2019-2025 Altova GmbH

| | | |
|---|---|---|
| fixed | Booleano | |
| typed-value | Sólo 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 | (Sólo 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 definition | Simple Type | |
| item type definition | (Sólo para tipos de lista) Simple Type | |
| member type definitions | (Sólo para tipos de unión) Secuencia de elementos Simple Type | |

Type Alternative


© 2019-2025 Altova GmbH

| 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: 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()*

**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


© 2019-2025 Altova GmbH

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**:

- **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)
```


© 2019-2025 Altova GmbH

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*

- **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')


© 2019-2025 Altova GmbH

### 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)`

### 18.2.2.1.8 Funciones XPath/XQuery: 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

 © 2019-2025 Altova GmbH

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**.
- **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


© 2019-2025 Altova GmbH

"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 ''


© 2019-2025 Altova GmbH

### 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

### 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


© 2019-2025 Altova GmbH

`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 |

### decode-string [altova:]

**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.


© 2019-2025 Altova GmbH

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 1237 ]

## 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.

Los motores XSLT y XQuery de los productos de Altova admiten el uso de funciones de extensión en Java[1240] y .NET[1249], así como scripts MSXSL para XSLT[1255].

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[1240]
- Funciones de extensión .NET[1249]
- Scripts MSXSL para XSLT[1255]

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:

- Archivos de clases definidos por el usuario[1242]
- Archivos JAR definidos por el usuario[1245]
- Java: Constructores[1246]
- Java: Métodos estáticos y campos estáticos[1246]
- Java: Métodos de instancia y campos de instancia[1247]
- Tipos de datos: Conversión de XPath/XQuery en Java[1248]
- Tipos de datos: Conversión de Java en XPath/XQuery[1249]

*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**.


© 2019-2025 Altova GmbH

- 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 apartados Archivos de clases definidos por el usuario 1242 y Archivos Jar definidos por el usuario 1245. 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* 1242)
- 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* 1243)
- El archivo de clases está en un paquete. El archivo XSLT/XQuery está en una carpeta cualquiera. (*ver ejemplo* 1243)
- El archivo de clases no está en un paquete. El archivo XSLT/XQuery está una carpeta cualquiera. (*ver ejemplo* 1244)

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>
```

 © 2019-2025 Altova GmbH

```
</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"
       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.


© 2019-2025 Altova GmbH

### 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.

### 18.2.2.2.1.3 Java: 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[1249], 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*[1247]):
```
<xsl:value-of select="date:toString(date:new())" xmlns:date="java:java.util.Date" />
```

### 18.2.2.2.1.4 Java: 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

 © 2019-2025 Altova GmbH

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 Java: 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`.

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)`.

 © 2019-2025 Altova GmbH

- 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:


- .NET: Constructores 1251
- .NET: Métodos estáticos y campos estáticos 1252
- .NET: Métodos de instancia y campos de instancia 1253
- Tipos de datos: Conversión de XPath/XQuery en .NET 1254
- Tipos de datos: Conversión de .NET en XPath/XQuery 1255


#### Formato de la función de extensión

La función de extensión de la expresión XPath/XQuery debe tener este formato `prefijo:nombreFunción()`.

- 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:

 © 2019-2025 Altova GmbH

```
 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.

### *18.2.2.2.2.1 .NET: 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 [1249], 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* [1247]):
```
<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 .NET: 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"/>
```


© 2019-2025 Altova GmbH

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 .NET: 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)"
               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.

 © 2019-2025 Altova GmbH

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*).

#### 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"`.

## 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 1258
- Motores de Altova 1258
- Compatibilidad con Unicode 1259
- Uso de Internet 1259


## 18.3.1 Requisitos de SO y memoria

### Sistema operativo

Las aplicaciones de software de Altova están disponibles en estas plataformas:

- Windows 10, Windows 11
- Windows Server 2016 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.


© 2019-2025 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.


*© 2019-2025 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.


© 2019-2025 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 inglés) en: https://www.altova.com/es/legal/eula
- Encontrará la Política de privacidad de Altova en: https://www.altova.com/es/privacy

## 18.4.4 Empaquetar archivos de licencia con el instalador de MapForce

Si desea efectuar una instalación silenciosa de MapForce, puede modificar la base de datos MSI para que incluya sus archivos de licencia. De este modo, el instalador no solo instalará el producto, sino que, además, le asignará una licencia. Para más información sobre cómo conseguirlo, descargue este archivo ZIP del sitio web de Altova y abra el documento PDF que contiene.

# Índice

## A

**A - Z,**
componente de ordenación, 426
**abs,**
como función de MapForce (en lang | math functions), 670
como función de MapForce (en xpath2 | numeric functions), 729
**acos,**
como función de MapForce (en lang | math functions), 670
**ActiveX,**
integration at application level, 1046
integration at document level, 1049
integration prerequisites, 1042
**ActiveX controls,**
adding to the Visual Studio Toolbox, 1044
**add,**
como función de MapForce (en core | math functions), 583
**ADO,**
como interfaz de conexión de datos, 178
configurar una conexión, 185
**ADO.NET,**
configurar una conexión, 191
**age,**
como función de MapForce (en lang | dateTime functions), 643
**Agregar,**
biblioteca personal, 526
**Análisis de impacto,**
hoja de estilos SPS, 801
**Any,**
xs:any, 151
**API COM de MapForce, 1040**
**Archivo,**
Abrir, 1104
Abrir el gestor de credenciales, 1104
Archivos recientes, 1104
Cerrar, 1104
Cerrar todos, 1104
como botón en componentes, 784
como botón en un componente, 41
Compilar en archivo de ejecución de MapForce Server, 1104
Configurar asignación, 1104
Configurar impresión, 1104
Generar código, 1104
Generar documentación, 1104
Guardar, 1104
Guardar como, 1104
Guardar todos, 1104
Implementar en FlowForce Server, 1104
Imprimir, 1104
Nuevo, 1104
Salir, 1104
Validar asignación, 1104
Vista previa de impresión, 1104
Volver a cargar, 1104
**Archivo de biblioteca,**
mff, 526
**Archivo mff,**
configurar, 526
**Archivo plano,**
asignación, 344
**Archivo/Cadena,**
como botón en componentes, 784
como botón en un componente, 41
**Archivo: (predeterminado),**
como nombre de nodo raíz, 784
**Archivo: <dinámico>,**
como nombre de nodo raíz, 784
**Archivos CSV,**
agregar o quitar campos, 353
como componente de destino, 353
como componente de origen, 353
configurar codificación, 353
vista previa, 353
**Archivos de proyecto (.mfp), 98**
**Archivos de texto,**
agregar o quitar campos, 361
asignar datos, 357
como componente de destino, 361
como componente de origen, 361
configurar codificación, 361
establecer carácter de relleno, 357
establecer el tamaño del campo fijo, 357
vista previa, 361
**archivos XML,**
como recursos globales, 863
generar a partir de registros de BD, 788
generar a partir de un solo origen XML, 787

© 2019-2025 Altova GmbH

**Asignación,**
basada en origen (contenido mixto), 52
componentes, 32
conectores, 32
conexiones, 32
configuración del archivo de salida, 78
crear, 32
depuración, 808
destino, 18
documentar, 799
Estándares SQL/XML, 262
estructura de esquema plana, 262
fundamentos, 32
hojas de estilos SPS predefinidas para la documentación, 801
opciones de configuración, 78
origen, 18
partes, 32
terminología, 32
términos, 32
tipos, 19
tipos de componentes, 32
Versión de XML Schema, 78
**Asignación de datos,**
agregar un componente, 107
agregar una función, 110
crear, 106
formato de archivo plano, 344
generar código, 112
guardar, 106
guardar resultados, 112
nombres de archivo dinámicos, 134
seleccionar el idioma de transformación, 106
validar, 106
ver la estructura, 107
**Asignación de licencias,**
empaquetar archivos de licencia con el instalador, 1263
**Asignación de valores,**
como componente de asignación, 445
pasar datos sin modificarlos, 450
tabla de búsqueda (propiedades), 453
**Asignaciones,**
procesamiento automatizado, 832
**asin,**
como función de MapForce (en lang | math functions), 671
**Aspectos fundamentales, 32, 34, 48, 66**
contexto, 82
contexto de prioridad, 89
estrategias de asignación, 80
orden de procesamiento, 82
reglas de asignación, 80
secuencias, 81
varios componentes de destino, 94
**atan,**
como función de MapForce (en lang | math functions), 671
**ATTLIST,**
DTD y URI de espacio de nombres, 138
**Automático/a,**
carga de bibliotecas, 526
**Automatizado,**
procesamiento, 832
**auto-number,**
como función de MapForce (en core | generator functions), 574
**avg,**
como función de MapForce (en core | aggregate functions), 547
**Ayuda,**
Acerca de MapForce, 1143
Activación del software, 1143
Buscar actualizaciones, 1143
Búsqueda, 1143
Cursos de MapForce, 1143
Descargar herramientas gratis y componentes, 1143
Formulario de pedido, 1143
Índice, 1143
MapForce en Internet, 1143
Preguntas frecuentes en Internet, 1143
Registro, 1143
Soporte técnico, 1143
Tabla de contenido, 1143
**Azure SQL, 223**

# B

**Barras de herramientas,**
Estado de la aplicación, 23
Herramientas, 23
Menú, 23
**Basada en origen,**
asignar contenido mixto, 52
**Base de datos,**
asignar esquema XML a un campo, 305
Codificación, 1133

**Base de datos,**
Configuración del editor SQL, 1133
Editor SQL, 1133
escribir archivos XML en, 308
Fuentes del editor de texto, 1133
generación de código SQL, 1133
generar varios archivos XML a partir de, 788
Vista de resultados, 1133
**Base de datos Access,**
actualización basada en condición IF, 357
**base de datos Oracle,**
conectarse por JDBC, 231
conectarse por ODBC, 232
**Bases de datos, 296**
acciones de tablas, 274, 276, 283, 294
Actualizar si, 276
Administración de transacciones, 251, 276
agregar, 246
agregar objetos de base de datos, 246, 259
archivo de seguimiento, 251
asignaciones en distintos entornos de ejecución, 174
campos NULL, 273
casos de uso, 262
como destino de datos, 274, 276
como recursos globales, 867
configuración, 246
configuraciones de acceso, 251
Configuraciones específicas de ADO/OLEDB, 251
Configuraciones específicas de JDBC, 251
configurar componentes, 251, 259
conos de columna, 246
consejos, 274
consulta, 294
contexto de asignación actual, 274
edición, 246
editar instrucciones SELECT, 259
editar objetos de base de datos, 246, 259
eliminar instrucciones SELECT, 259
eliminar registros secundarios, 276, 283
Eliminar si, 276, 283
Eliminar todo, 283
excepción, 276
excepciones, 251, 288
filtro, 246
Finalización automática SQL, 246
funciones, 273
generar valores secuenciales, 273
generar valores únicos, 273
guardar todas las rutas de acceso de archivos como relativas al archivo MFD, 251
inserción masiva, 276
INSERT, 294
insertar objetos de base de datos, 246
Insertar Restantes, 283
Insertar Todo, 276, 283
Instrucción SELECT, 259
Instrucción SELECT con parámetros, 259
Instrucción SQL personalizada, 283
Instrucciones SQL en Resultados, 274
JDBC, 251
max() +1, 276
MERGE, 294
modificar, 294
nivel de seguimiento, 251
nombre de la conexión, 251
Null equal, 276
objetos, 246
Omitir si, 276, 283
origen de datos, 251
parámetros, 259
personalizar instrucciones SQL, 259
problemas con el controlador JDBC, 174
quitar, 246
registro de errores, 251, 276
relación de clave foránea, 262
relaciones, 246, 262
relaciones locales, 269
reversión, 288
revertir transacciones, 276, 288
seguimiento, 251, 276
tabla principal, 276
tablas secundarias, 276
tiempo de espera para la ejecución de instrucciones, 251
transacciones, 288
transferencia masiva, 276
UPDATE, 294
usar conexión de BD compartida, 251
uso eficiente, 274
valor asignado, 276
valor generado por la BD, 276
valores NULL, 273
**base-uri,**
como función de MapForce function (en xpath2 | accessors library), 698
**BD,**

**BD,**
ORDER BY, 437
**Biblioteca, 1035**
agregar personal, 526
carga automática, 526
**Biblioteca personal,**
agregar, 526
**boolean,**
como función de MapForce (en core | conversion functions), 554
**Buscar,**
elementos en los componentes de asignación, 41

# C

**C#,**
integration of MapForce, 1052
referencia de las clases generadas, 988
**C++,**
referencia de las clases generadas, 972
**Cadena,**
redistribuir datos de, 791
redistribuir datos desde, 791
serializar datos a, 791, 794
**Campo,**
claves en archivos de texto, 349
**Campos vacíos,**
en archivos CSV, 353
en archivos de texto planos, 361
**capitalize,**
como función de MapForce (en lang | string functions), 680
**Carácter de relleno,**
en archivos de texto planos, 361
**Carpetas,**
como recursos globales, 865
**Catálogos, 877**
**Catalogs,**
customize, 882
environment variables, 884
in DTD, 878
in XML Schema, 878
structure, 880
**CDATA, 150**
**ceiling,**
como función de MapForce (en core | math functions), 583
**Clave,**
campos en archivos de texto, 349
clave de ordenación, 426
**Clave de ordenación,**
componente de ordenación, 426
**code-from-char,**
como función de MapForce (en core | string functions), 625
**Codificación,**
cambiar en archivos CSV, 353
cambiar en archivos de texto planos, 361
**Código,**
SPL, 1020
tipos integrados, 1034
**Código generado,**
agregar, 457
**Comandos de menú, 1103**
Archivo, 1104
Ayuda, 1143
Componente, 1113
Conexión, 1115
Depuración, 1119
Edición, 1107
Función, 1116
Herramientas, 1123
Herramientas | Opciones, 1128, 1131
Herramientas | Opciones | Base de datos, 1133
Herramientas | Opciones | Java, 1132
Herramientas | Opciones | Proxy de red, 1139
Herramientas | Personalizar, 1124
Herramientas | Teclado, 1125
Insertar, 1108
Personalizar, 1124, 1125
Proyecto, 1111
Resultados, 1117
Vista, 1121
Windows, 1142
**COMBINACIÓN EXTERNA IZQUIERDA,**
en componentes de combinación, 410
**COMBINACIÓN INTERNA,**
en componentes de combinación, 410
**Comentarios,**
agregar a archivos de destino, 149
**Comillas,**
en archivos CSV, 353
**Command reference, 1072**
**Comodines,**
xs:any - xs:anyAtrribute, 151
**Compatibilidad con Unicode,**

**Compatibilidad con Unicode,**
de los productos de Altova, 1259
**Complemento de MapForce para Eclipse,**
acceso a menús y funciones frecuentes, 895
ampliar características, 909
cambiar a la perspectiva MapForce, 892
configuración para la generación automática de código, 906
crear asignaciones nuevas, 901
crear un proyecto de MapForce/Eclipse, 899
importar asignaciones a un proyecto de Eclipse, 903
información, 889, 892, 899
instalación, 890
punto de extensión, 909
trabajar con asignaciones y proyectos, 899
**Complemento de MapForce para Visual Studio,**
acerca de, 886
habilitar, 886
información sobre menús y funciones, 886
trabajar con asignaciones y proyectos, 886
**Complemento para Visual Studio,**
ejecutar MapForce como, 886
**Componente,**
ordenar datos, 426
**Componentes,**
Acciones de tablas de BD, 1113
Actualizar, 1113
agregar a la asignación, 38
Agregar delante un duplicado de entrada, 1113
Agregar detrás un duplicado de entrada, 1113
Agregar, quitar o editar objetos de la base de datos, 1113
ajustes, 41
alinear, 41
Alinear la estructura a la derecha, 1113
Alinear la estructura a la izquierda, 1113
Archivo de Protocol Buffers, 1108
Archivo de texto, 1108
Archivo Excel 2007+, 1108
Archivo o esquema JSON, 1108
Archivo o esquema XML, 1108
Asignación de valores, 1108
Base de datos, 1108
bases de datos, 174, 246, 251, 259, 262, 269, 273, 274, 276
buscar, 41
Cambiar de elemento raíz, 1113
cambiar la configuración, 41
comandos de menú, 1113
Combinar, 1108
comentario, 34
Componente de ordenación: nodos/filas, 1108
Condición If-Else, 1108
Constante, 1108
Consultar BD, 1113
Crear asignación a EDI X12 997, 1113
Crear asignación a EDI X12 999, 1113
Documento XBRL, 1108
EDI, 1108
Editar la configuración de FlexText, 1113
Editar la definición del esquema en XMLSpy, 1113
elementos eliminados, 62
eliminar, 63
Entrada simple, 1108
Escribir contenido como sección CDATA, 1113
esquema, 41
Esquema XML, 139
estructurales, 34, 137, 138, 174, 246, 251, 259, 262, 269, 273, 274, 276
Excepción, 1108
Filtro: nodos/filas, 1108
Función de servicio web, 1108
Insertar componente de entrada, 1108
Insertar componente de salida, 1108
Propiedades, 1113
Quitar el duplicado, 1113
referencia de los iconos, 34
resumen, 34
Salida simple, 1108
transformación, 34, 368
Variable, 1108
WHERE/ORDER de SQL, 1108
XML, 139
XML y XML Schema, 139, 146
**Componentes estructurales,**
bases de datos, 174, 246, 251, 259, 262, 269, 273, 274, 276
Esquema XML, 138
XML, 138
XML y XML Schema, 138
**concat,**
(como función) ejemplo de uso, 357
como función de MapForce (en core | string functions), 626
**Condiciones If-Else,**
agregar a la asignación, 432
**Conector,**
ver historial de valores procesados, 818
**Conexión de base de datos,**

**Conexión de base de datos,**
configuración, 178
ejemplos de configuración, 207
iniciar el asistente, 179
reutilizar desde recursos globales, 206
**Conexiones,**
ajustes, 59
anotación, 59
basada en destino, 52
Basada en destino (estándar), 1115
basada en origen, 52
Basada en origen (contenido mixto), 1115
cambiar, 48
Conectar automáticamente los secundarios equivalentes, 1115
Conectar los secundarios equivalentes, 1115
conexiones antecesoras ausentes, 48
Configurar la conexión de secundarios equivalentes, 1115
conservar conexiones tras eliminación de componentes, 63
Copia total (copiar elementos secundarios), 1115
copiar, 48
crear, 48
de copia total, 52
eliminar, 48
entradas obligatorias, 48
estándar, 52
menú contextual, 60
mixta, 52
mover, 48, 62
Propiedades, 1115
reparar, 62
reparar después de editar el esquema, 62
resaltado selectivo, 48
secundarios equivalentes, 52
tipos, 52, 59
ver la información rápida de las conexiones, 48
**Conexiones defectuosas,**
después de cambiar el esquema, 62
en archivos XML, 62
en bases de datos, 62
**Conexiones nativas,**
Azure CosmosDB, 205
CouchDB, 205
MariaDB, 205
MongoDB, 205
PostgreSQL, 205
SQLite, 205
**Configurar,**
archivo mff, 526
**Conservar datos,**
cuando se use una asignación de valores, 450
**Conservar datos sin modificarlos,**
al pasar por una asignación de valores, 450
**Constantes,**
agregar, 461
**contains,**
como función de MapForce (en core | string functions), 627
**Contenido mixto,**
asignar, 52
con conexiones basadas en destino, 52
con conexiones estándar, 52
**Contexto (ventana),**
información, 812, 820
**Contexto de prioridad,**
ejemplo, 91
**Contexto primario,**
ejemplo, 85
**Contrato de licencia para el usuario final, 1261, 1263**
**Controladores de BD,**
resumen, 181
**Controladores ODBC,**
comprobar disponibilidad de, 201
**convert-to-utc,**
como función de MapForce (en lang | dateTime functions), 643
**Cortar/Copiar/Pegar/Eliminar,**
Buscar, 1107
Buscar anterior, 1107
Buscar siguiente, 1107
Deshacer, 1107
Rehacer, 1107
Seleccionar todo, 1107
**cos,**
como función de MapForce (en lang | math functions), 671
**count,**
como función de MapForce (en core | aggregate functions), 548
**count-substr,**
como función de MapForce (en lang | string functions), 684
**create-guid,**
como función de MapForce (en lang | generados functions), 667
**Criterio de ordenación,**
cambiar, 426
**CSV,**

**CSV,**
como origen de la asignación, 344
crear jerarquías (claves), 349
crear varias filas, 346
**current,**
como función de MapForce (en xslt | xslt functions), 758
**current- dateTime,**
como función de MapForce (en xpath2 | context functions), 703
**current- time,**
como función de MapForce (en core | context functions), 704
**current-date,**
como función de MapForce (en core | context functions), 703
**char-from-code,**
como función de MapForce (en core | string functions), 623
**charset-decode,**
como función de MapForce (en lang | string functions), 680, 682

# D

**date-from-datetime,**
como función de MapForce (en lang | datetime functions), 644
**datetime- diff,**
como función de MapForce (en lang | dateTime functions), 646
**datetime-add,**
como función de MapForce (en lang | dateTime functions), 644
**datetime-from-date-and-time,**
como función de MapForce (en lang | datetime functions), 647
**datetime-from-parts,**
como función de MapForce (en lang | dateTime functions), 647
**Datos de tabla,**
ordenar, 426
**day-from- duration,**
como función de MapForce (en lang | dateTime functions), 649
**day-from-datetime,**
como función de MapForce (en lang | datetime functions), 648
**de SPL,**
foreach, 1030
variables, 1024
**default-collation,**
como función de MapForce (en xpath2 | context functions), 704
**degrees,**
como función de MapForce (en lang | math functions), 672
**Delimitador,**
cambiar en archivos CSV, 353
cambiar en archivos de texto planos, 361
**Depuración,**
como menú de aplicación, 1119
con puntos de interrupción, 808
Detener, 1119
información, 812
Iniciar, 1119
Paso a paso para salir, 1119
Paso a paso por instrucciones, 1119
Paso a paso por procedimientos, 1119
Paso mínimo, 1119
por pasos, 808
preparar, 811
restricciones, 808
**distinct-values,**
como función de MapForce (en core | sequence functions), 595
**Distribución,**
de productos de software de Altova, 1261
**divide,**
como función de MapForce (en core | math functions), 584
**divide-integer,**
como función de MapForce (en lang | math functions), 672
**document,**
como función de MapForce (en xslt | xslt), 758
**Documentación,**
definir hojas de estilos SPS, 806
**Documentación general,**
hoja de estilos SPS, 801
**Documentar,**
asignaciones, 799
**Document-level,**
examples of integration of MapForce, 1052
**DoTransform.bat,**
ejecutar con RaptorXML Server, 831
**DSN de archivo,**
configuración, 201
**DSN de sistema,**
configuración, 201
**DSN de usuario,**

**DSN de usuario,**
configuración, 201
**DTD,**
origen y destino, 138
**duration- add,**
como función de MapForce (en lang | dateTime functions), 649
**duration- subtract,**
como función de MapForce (en lang | dateTime functions), 651
**Duration-from-parts,**
como función de MapForce (en lang | dateTime functions), 650

# E

**Edición,**
como menú de aplicación, 1107
**Editor SQL,**
agregar comentarios SQL, 298
agregar marcadores, 298
botones de la barra de herramientas, 298
cargar desde scripts, 298
comentario de bloque, 298
comentario de línea, 298
crear instrucciones, 298
ejecutar instrucciones SQL, 298
exportar scripts SQL, 298
importar scripts SQL, 298
insertar regiones SQL, 298
**element-available,**
como función de MapForce (en xslt | xslt functions), 759
**Elementos,**
ausentes, 62
**Elementos ausentes, 62**
**Eliminación inteligente de componentes, 63**
**Eliminar,**
elementos ausentes, 62
**empty,**
como función de MapForce (en lang | string functions), 685
**Entrada,**
duplicar, 41
**Entrada de la asignación,**
especificar varios archivos como, 784
**Enumerations,**
in MapForceControl, 1102
**equal-or-greater,**
como función de MapForce (en core | logical functions), 578
**equal-or-less,**
como función de MapForce (en core | logical functions), 578
**Errores,**
errores de memoria, 38
resolución de problemas, 38
**Espacios de nombres,**
declarar personalizados, 154
y comodines (xs:any), 151
**Esquema,**
estándar del sector, 138
generar, 138
paquete, 138
**Excepciones,**
agregar, 457
**exists,**
como función de MapForce (en core | sequence functions), 597
**exp,**
como función de MapForce (en lang | math functions), 673
**Explorador de la BD,**
buscar objetos de base de datos, 296
diseño de árbol de BD, 296
disposición en carpetas, 296
dividir tablas en tablas del sistema y del usuario, 296
filtrar objetos de base de datos, 296
localizador de objetos, 296
**Expresiones regulares,**
en asignaciones, 540
**Extensiones de Altova,**
funciones para gráficos, 1160

# F

**false,**
como función de MapForce (en lang | boolean functions), 701
**Fija,**
archivos de longitud fija (asignación de), 344
**Filas,**
asignación de datos de archivos de texto, 349
**Filtrar,**
datos de componentes, 432
en bases de datos, 441
tablas de base de datos, 432

**Filtros,**
agregar a la asignación, 432
**find-substring,**
como función de MapForce (en lang | string functions), 685
**Firebird,**
Conexión a través de JDBC, 207
Conexión a través de ODBC, 208
**first-items,**
como función de MapForce (en core | sequence functions), 599
**floor,**
como función de MapForce (en core | math functions), 584
**FlowForce Server,**
implementar asignaciones en, 842
implementar recursos globales en, 842, 874
Recursos globales en, 874
**format-date,**
como función de MapForce (en core | conversion functions), 555
**format-dateTime,**
como función de MapForce (en core | conversion functions), 556
**format-guid- string,**
como función de MapForce (en lang | string functions), 686
**format-number,**
como función de MapForce (en core | conversion functions), 559, 563
**format-time,**
como función de MapForce (en core | conversion functions), 562
**Función,**
como menú de aplicación, 1116
**Funciones, 461**
agregar, 461
agregar parámetros, 461
buscar, 461
buscar en la ventana Bibliotecas, 461
buscar ocurrencias en la asignación activa, 461
constantes, 461
descripción, 461
eliminar parámetros, 461
esquema, 461
lang | string functions (cadena), 679, 694
nodo, 469, 471
parámetros, 461
tipo de datos del argumento, 461
**Funciones de extensión .NET,**
campos de instancia, 1253
campos estáticos, 1252
constructores, 1251
conversiones de tipos de datos, 1254, 1255
métodos de instancia, 1253
métodos estáticos, 1252
para XSLT y XQuery, 1249
tipos de datos .NET en XPath/XQuery, 1255
tipos de datos XPath/XQuery en .NET, 1254
**Funciones de extensión en scripts MSXSL,**
msxsl:script, 1255
**Funciones de extensión Java,**
archivos de clases definidos por el usuario, 1242
archivos JAR definidos por el usuario, 1245
campos de instancia, 1247
campos estáticos, 1246
constructores, 1246
conversiones de tipos de datos, 1248, 1249
métodos de instancia, 1247
métodos estáticos, 1246
para XSLT y XQuery, 1240
tipos de datos Java en XPath/XQuery, 1249
tipos de datos XPath/XQuery en Java, 1248
**Funciones de extensión para XSLT y XQuery,**
funciones de extensión .NET, 1249
funciones de extensión Java, 1240
ver Funciones de extensión .NET, 1249
ver Funciones de extensión Java, 1240
**Funciones de nodo,**
crear, 471
**Funciones definidas por el usuario,**
agregar parámetros, 492
búsqueda, 503
búsqueda recursiva, 497
copiar y pegar, 487
crear, 487
de tipo complejo, 492
de tipo simple, 492
edición, 487
ejemplo, 485
ejemplos, 497, 503
eliminar, 487
Estructuras de tipo complejo, 492
importar, 487
inline, 487
llamada, 487
llamadas recursivas, 497
navegación, 487

© 2019-2025 Altova GmbH

**Funciones definidas por el usuario,**
orden de los parámetros, 492
parámetros, 492
parámetros de entrada, 487
parámetros de salida, 487
recursivamente, 497
regular, 487
resumen, 485
ventajas de las, 485
y contexto de asignación, 500
**Funciones utilizadas por, 801**
**function-available,**
como función de MapForce (en xslt | xslt functions), 759
**Functions,**
Configuración de la función, 1116
Crear función definida por el usuario a partir de la selección, 1116
Crear una función definida por el usuario, 1116
Insertar componente de entrada, 1116
Insertar componente de salida, 1116
Quitar función, 1116

# G

**Generación de código,**
C#, 68, 911, 913, 919, 922
C++, 68, 911, 913, 919
cambiar tipo de datos, 922
compilar, 68, 911
compilar el código, 68, 911, 913
ejecutar aplicación, 68, 911
ejecutar código, 913
generar, 68, 911
generar código, 68, 911, 913
integrar código generado, 919, 922
Java, 68, 911, 913, 919, 922
modificar código generado, 919, 922
modificar entrada/salida, 919
XQuery, 68, 911
XSLT, 68, 911
**generate-id,**
como función de MapForce (en xslt | xslt), 760
**generate-sequence,**
como función de MapForce (en core | sequence functions), 600
**Gestor de esquemas,**
aplicar un parche a un esquema, 165
cómo ejecutar, 160
desinstalar un esquema, 166
estado de los esquemas en el, 163
help (comando ILC), 167
info (comando ILC), 168
initialize (comando ILC), 168
instalar un esquema, 165
install (comando ILC), 169
introducción, 157
introducción a la ILC, 167
list (comando ILC), 170
lista de esquemas por estado, 163
reset (comando ILC), 170
restaurar, 166
uninstall (comando ILC), 171
update (comando ILC), 172
upgrade (comando ILC), 173
**Gestor de esquemas XML, 138**
**get-fileext,**
como función de MapForce (en core | file path functions), 570
**get-folder,**
como función de MapForce (en core | file path functions), 571
**goup-ending-with,**
como función de MapForce (en core | sequence functions), 607
**greater,**
como función de MapForce (en core | logical functions), 579
**group-adjacent,**
como función de MapForce (en core | sequence functions), 601
**group-by,**
como función de MapForce (en core | sequence functions), 603
**group-into-blocks,**
como función de MapForce (en core | sequence functions), 608
**group-starting-with,**
como función de MapForce (en core | sequence functions), 610

# H

**Herramientas,**
comando de menú, 1123

**Herramientas,**
Configuración activa, 1123
Crear asignación invertida, 1123
Gestor de taxonomías XBRL, 1123
Opciones, 1123
Personalizar, 1123
Recursos globales, 1123
Restaurar barras de herramientas y ventanas, 1123
**Herramientas | Opciones,**
Base de datos, 1128
configuración de la generación, 1131
Depurador, 1128
Edición, 1128
Generación, 1128, 1131
generación de código, 1131
Generales, 1128
Java, 1128
Mensajes, 1128
Proxy de red, 1128
XBRL, 1128
**Hojas de estilos,**
definir para la documentación, 806
**hour-from- duration,**
como función de MapForce (en lang | datetime functions), 653
**hour-from-datetime,**
como función de MapForce (en lang | datetime functions), 652
**HTML,**
documentación de la asignación, 799
vista previa del resultado de la asignación como, 847

# I

**IBM DB2,**
conectarse por JDBC, 210
conectarse por ODBC, 212
escribir en campos de tipo XML, 304
leer campos de tipo XML, 304
**IBM DB2 para i,**
conectarse por JDBC, 215
conectarse por ODBC, 216
**IBM Informix,**
conectarse por JDBC, 219
**IGU, 22**
barras de herramientas, 23
Panel de consulta de BD, 294, 298, 301, 303
paneles, 28
Ventana Mensajes, 27
ventanas, 23
**implicit- timezone,**
como función de MapForce (en xpath2 | context functions), 704
**Información general, 1258**
**Información legal, 1261**
**Información sobre derechos de autor, 1261**
**Información técnica, 1258**
**Insertar,**
componente WHERE de SQL, 437
**Instalación silenciosa,**
empaquetar archivos de licencia con el instalador, 1263
modificar archivo MSI, 1263
**Instrucciones de procesamiento,**
agregar a archivos de destino, 149
**Instrucciones de procesamiento y comentarios,**
asignar, 52
**integración,**
con productos de Altova, 20
**Integrating,**
MapForce in applications, 1041
**Intercalación,**
componente de ordenación, 426
intercalación local, 426
punto de código unicode, 426
**Intercalación local, 426**
**Interfaz del usuario, 22**
**is-not-null,**
como función de MapForce (en db functions), 640
**is-null,**
como función de MapForce (en db functions), 640
**is-xsi-nil,**
como función de MapForce (en core | node functions), 589
**item-at,**
como función de MapForce (en core | sequence functions), 612
**items-from-till,**
como función de MapForce (en core | sequence functions), 613

# J

**Java, 1059**

**Java, 1059**
referencia de las clases generadas, 1004
ubicación de la biblioteca VM, 1132
**JDBC,**
como interfaz de conexión de datos, 178
conectarse a Teradata, 240
configurar una conexión (Windows), 197
**Jerarquía,**
a partir de archivos de texto, 349

# L

**lang | string functions (cadena),**
sleep (suspensión), 694
**last,**
como función de MapForce (en xpath2 | context functions), 705
**last-items,**
como función de MapForce (en core | sequence functions), 614
**leapyear,**
como función de MapForce (en lang | dateTime functions), 653
**left,**
como función de MapForce (en lang | string functions), 687
**left-trim,**
como función de MapForce (en lang | string functions), 687
**Lenguajes de transformación,**
BUILT-IN, 19
C#, 19
C++, 19
Java, 19
XQuery, 19
XSLT 1.0, 19
XSLT 2.0, 19
XSLT 3.0, 19
**less,**
como función de MapForce (en core | logical functions), 579
**Licencia, 1263**
información sobre, 1261
**Licencia del producto de software, 1263**
**Licencias, 1143**
**local-name-from-QName,**
como función de MapForce (en lang | QName functions), 593
**log,**
como función de MapForce (en lang | math functions), 673, 674
**logical-and,**
como función de MapForce (en core | logical functions), 580
**logical-not,**
como función de MapForce (en core | logical functions), 580
**logical-or,**
como función de MapForce (en core | logical functions), 581
**logical-xor,**
como función de MapForce (en lang | logical functions), 668
**Lotes,**
automatización de procesamiento, 832
**lowercase,**
como función de MapForce (en lang | string functions), 688
**Llamar a gráficos,**
hoja de estilos SPS, 801

# M

**main-mfd-filepath,**
como función de MapForce (en core | file path functions), 571
**MapForce,**
integration, 1041
introducción, 17
modelo de transformación de datos, 17
resumen, 17
**MapForce Server,**
automatizar asignaciones, 832
compilar asignaciones para, 839
generar excepciones desde, 457
Recursos globales en, 874
**MapForceCommand,**
in MapForceControl, 1082
**MapForceCommands,**
in MapForceControl, 1084
**MapForceControl, 1085**
documentation of, 1041
examples of integration at document level, 1052
integration using C#, 1052
integration using Visual Basic, 1068
object reference, 1082
**MapForceControlDocument, 1093**
**MapForceControlPlaceHolder, 1099**
**MariaDB,**
conectarse por ODBC, 220

© 2019-2025 Altova GmbH

**match-pattern,**
como función de MapForce (en lang | string functions), 689
**max,**
como función de MapForce (en core | aggregate functions), 549
como función de MapForce (en lang | math functions), 674
**max-string,**
como función de MapForce (en core | aggregate functions), 550
**Medición de licencias,**
en los productos de Altova, 1262
**mfd-filepath,**
as MapForce function (in core | file path functions), 571
**mff,**
archivo de biblioteca, 526
archivo mff.xsd, 526
**Microsoft Access,**
conectarse por ADO, 185, 221
**Microsoft Azure SQL, 223**
**Microsoft SharePoint Server,**
agregar archivos como componentes desde, 38
**Microsoft SQL Server,**
conectarse por ADO, 223
conectarse por ODBC, 224
**millisecond-from-datetime,**
como función de MapForce (en lang | datetime functions), 654
**millisecond-from-duration,**
como función de MapForce (en lang | datetime functions), 654
**min,**
como función de MapForce (en core | aggregate functions), 550
como función de MapForce (en lang | math functions), 675
**min-string,**
como función de MapForce (en core | aggregate functions), 551
**minute-from- datetime,**
como función de MapForce (en lang | datetime functions), 655
**minute-from- duration,**
como función de MapForce (en lang | datetime functions), 655
**Mixto,**
asignación basada en origen, 52
asignación de contenido, 52
**modulus,**
como función de MapForce (en core | math functions), 585
**month-from- datetime,**
como función de MapForce (en lang | datetime functions), 656
**month-from- duration,**
como función de MapForce (en lang | datetime functions), 656
**Motores,**
de los productos de Altova, 1258
**multiply,**
como función de MapForce (en core | math functions), 586
**MySQL,**
conectarse por ODBC, 229

# N

**namespace-uri-from-QName,**
como función de MapForce (en lang | QName functions), 594
**negative,**
como función de MapForce (en lang | lógicas functions), 668
**node-name,**
como función de MapForce (en core | node functions), 590
como función de MapForce (en xpath2 | accessors), 699
**node-name (función),**
alternativas de uso, 763
**Nombres de nodo,**
asignar, 763
**normalize-space,**
como función de MapForce (en core | string functions), 628
**Notas generales, 17**
**not-equal,**
como función de MapForce (en core | logical functions), 577, 582
**not-exists,**
como función de MapForce (en core | sequence functions), 615
**Novedades, 13**
Versión 2022, 15
Versión 2023, 14
Versión 2024, 13
**now,**
como función de MapForce (en lang | dateTime functions), 657
**NULL,**
atributo, 146
valores, 146
valores en bases de datos, 146
**numeric,**

**numeric,**
como función de MapForce (en lang | logical functions), 669

# O

**Objetos globales,**
de SPL, 1025
**ODBC,**
como interfaz de conexión de datos, 178
conectarse a MariaDB, 220
conectarse a Teradata, 241
configurar una conexión, 201
**OLE DB,**
como interfaz de conexión de datos, 178
**Opciones de generación de código,**
definir globalmente para todo el proyecto, 100
**OpenAI,**
casos de uso, 694
ejemplo, 694
**OpenJDK,**
como máquina virtual Java, 197
**Oracle,**
escribir en campos de tipo XML, 304
leer campos de tipo XML, 304
**Ordenar,**
componente de ordenación, 426
en bases de datos, 441
**Ordenar datos,**
componente de ordenación, 426
**ORDER BY,**
componente WHERE de SQL, 437

# P

**pad-string-left,**
como función de MapForce (en lang | string functions), 689
**pad-string-right,**
como función de MapForce (en lang | string functions), 690
**Panel de consulta de BD,**
Editor SQL, 294, 298
Explorador de la BD, 294, 296
Pestaña Mensajes, 294, 303
Pestaña Resultados, 294, 301
**Paneles,**
Asignación, 28
Consulta de BD, 28
Resultados, 28
Resultados de StyleVision, 28
XQuery, 28
XSLT, 28
**Parámetros,**
pasar a la asignación, 369, 374
**pares clave-valor,**
usar en la asignación, 445
**parse-date,**
como función de MapForce (en core | conversion functions), 564
**parse-dateTime,**
como función de MapForce (en core | conversion functions), 565
**parse-number,**
como función de MapForce (en core | conversion functions), 568
**parse-time,**
como función de MapForce (en core | conversion functions), 569
**Pasar datos,**
sin modificarlos a través de una asignación de valores, 450
**PDF,**
documentación de la asignación, 799
vista previa del resultado de la asignación como, 847
**Período de evaluación,**
de los productos de software de Altova, 1261
**Personalizar,**
comandos, 1124
eliminar comandos, 1124
menú contextual, 1124
Menú predeterminado vs. diseño de MapForce, 1124
menús, 1124
restaurar barras de herramientas, 1124
sombras de menú, 1124
Teclado, 1125
Teclas de acceso rápido, 1125
**Pestaña Mensajes,**
advertencias, 303
buscar, 303
buscar anterior, 303
buscar siguiente, 303
copiar mensaje en el portapapeles, 303
copiar mensajes, 303
eliminar, 303
errores, 303

© 2019-2025 Altova GmbH

**Pestaña Mensajes,**
filtro, 303
resumen, 303
**Pestaña Resultados,**
buscar, 301
copiar, 301
ordenar, 301
seleccionar, 301
**pi,**
como función de MapForce (en lang | math functions), 675
**Plataformas,**
para productos de Altova, 1258
**Posición del depurador,**
ver valor actual de, 818
**position,**
como función de MapForce (en core | sequence functions), 616
**positive,**
como función de MapForce (en lang | lógicas functions), 669
**PostgreSQL,**
conectarse por ODBC, 235
**pow,**
como función de MapForce (en lang | math functions), 676
**Procedimientos almacenados,**
agregar a una asignación, 319
llamar desde una asignación, 315, 321, 325, 329, 335, 338
notas sobre compatibilidad, 315
**Procedimientos generales, 66**
búsqueda en la vista Texto, 74
configuración de la asignación, 78
configuración del archivo de salida, 78
generación de código, 78
generar código, 68
rutas de acceso en el código generado, 78
validación, 66
validar, 66
validar asignación, 66
validar resultados de la asignación, 66
vista texto, 71
**Procesador XQuery,**
de los productos de Altova, 1258
**Procesadores XSLT,**
de los productos de Altova, 1258
**Procesamiento,**
automatizar asignaciones, 832
**Progress OpenEdge (base de datos),**
conectarse por JDBC, 236
conectarse por ODBC, 237
**Project,**
add mappings to, 98
add Web links, 98
basics, 98
closing, 98
code generation, 98
creating, 98
deleting, 98
generate code, 98
image preview, 98
new, 98
opening, 98
organization, 98
removing, 98
searching, 98
watch video tutorials, 98
**Propiedades,**
tabla de asignación de valores, 453
**Proxy de red,**
configuración, 1139
configuración automática, 1139
manual, 1139
opciones, 1139
sistema, 1139
**Proyecto,**
ajustes, 100
carpetas, 101
opciones de generación de código, 100, 101
propiedades, 101
**Proyectos, 98**
Abrir archivo en XMLSpy, 1111
Abrir asignación de datos, 1111
Agregar archivo activo al proyecto, 1111
Agregar archivo de asignación para la operación, 1111
Agregar archivos al proyecto, 1111
Cerrar, 1111
Crear asignación para la operación, 1111
Crear carpeta, 1111
Generar código en, 1111
Generar código para todo el proyecto, 1111
Guardar, 1111
Insertar servicio web, 1111
Propiedades, 1111
Volver a cargar, 1111
**Punto de código,**
intercalación, 426

© 2019-2025 Altova GmbH

**Puntos de interrupción,**
agregar, 815
eliminar, 815
información, 812
**Puntos de interrupción (ventana),**
información, 812, 822

# Q

**QName,**
como función de MapForce (en lang | QName functions), 593
**QName-as-string,**
-as-string como función de MapForce (en lang | QName functions), 679

# R

**radians,**
como función de MapForce (en lang | math functions), 676
**random,**
como función de MapForce (en lang | math functions), 676
**RaptorXML Server,**
ejecutar una transformación, 831
**read-binary-file,**
como función de MapForce (en lang | string functions), 662
**Recursos globales,**
Archivo de definiciones, 857
Archivos XML como, 863
bases de datos como, 867
carpetas como, 865
configurar, 857
crear, 857
en entornos de ejecución, 873
en FlowForce Server, 874
implementar en FlowForce Server, 842, 874
introducción a, 856
**Referencia del usuario, 1103**
**Relaciones locales,**
y procedimientos almacenados, 333, 335, 338
**remove- timezone,**
como función de MapForce (en lang | dateTime functions), 657
**remove-fileext,**
como función de MapForce (en core | file path functions), 572
**remove-folder,**
como función de MapForce (en core | file path functions), 572
**repeat- string,**
como función de MapForce (en lang | string functions), 691
**replace,**
como función de MapForce (en lang | string functions), 691
**replace-fileext,**
como función de MapForce (en core | file path functions), 573
**replicate-item,**
como función de MapForce (en core | sequence functions), 619
**replicate-sequence,**
como función de MapForce (en core | sequence functions), 620
**Requisitos de memoria, 1258**
**resolve-filepath,**
como función de MapForce (en core | file path functions), 573
**resolver- uri,**
como función de MapForce (en xpath2 | anyURI), 700
**Resultado de la asignación,**
generar varios archivos como, 784
**Resultados, 1117**
C#, 1117
C++, 1117
como menú de aplicación, 1117
Configurar la vista Texto, 1117
Ejecutar script SQL/NoSQL, 1117
Guardar el archivo de salida, 1117
Guardar todos los archivos de salida, 1117
Insertar o quitar marcador, 1117
Java, 1117
Marcador anterior, 1117
Marcador siguiente, 1117
Motor de ejecución integrado, 1117
Quitar todos los marcadores, 1117
Texto XML pretty-print, 1117
Validar archivo de salida, 1117
Volver a generar archivo de salida, 1117
XQuery, 1117
XSLT 1.0, 1117
XSLT 2.0, 1117
XSLT 3.0, 1117
**Retener datos,**
al pasar por una asignación de valores, 450

**reversefind-substring,**
como función de MapForce (en lang | string functions), 692
**right,**
como función de MapForce (en lang | string functions), 693
**right-trim,**
como función de MapForce (en lang | string functions), 693
**round-half-to-even,**
como función de MapForce (en xpath2 | numeric functions), 729
**round-precision,**
como función de MapForce (en core | math functions), 586
**RTF,**
documentación de la asignación, 799
vista previa del resultado de la asignación como, 847
**Rutas de acceso de archivos,**
absolutas, 44
corregir referencias a rutas de acceso rotas, 44
de bases de datos basadas en archivos, 44
en el código generado, 46
en entornos de ejecución, 46
relativas, 44
relativas y absolutas, 46
rotas, 44

# S

**schemanativetype, 1022**
**Search,**
files in the Project window, 98
projects, 98
**Sección,**
CDATA, 150
**Second-from-datetime,**
como función de MapForce (en lang | datetime functions), 658
**second-from-duration,**
como función de MapForce (en lang | datetime functions), 658
**Servidor SQL,**
conectarse por ADO, 185
**set-empty,**
como función de MapForce (en core | sequence functions), 621
**set-null,**
como función de MapForce (en db functions), 641
**set-xsi-nil,**
como función de MapForce (en core | node functions), 591
**Signo de interrogación,**
elementos ausentes, 62
**sin,**
como función de MapForce (en lang | math functions), 677
**skip-first-items,**
como función de MapForce (en core | sequence functions), 622
**SO,**
para productos de Altova, 1258
**SPL, 1020**
bloques de código, 1021
condiciones, 1029
objetos globales, 1025
subrutinas, 1031
usar archivos, 1027
**SPS,**
hojas de estilos definidas por el usuario, 806
hojas de estilos predefinidas para documentar asignaciones, 801
**SQL,**
combinar datos, 410
**SQL Azure, 223**
**SQL Server,**
conectarse por ADO.NET, 191
conectarse por JDBC, 197
escribir en campos de tipo XML, 304, 305
leer campos de tipo XML, 304, 305
**SQL WHERE,**
componente (insertar), 437
ORDER BY, 437
**SQL WHERE/ORDER,**
como componente de MapForce, 441
**SQLite,**
asignar datos a, 794
convertir rutas de acceso de BD absolutas en el código generado, 46
escribir archivos XML en, 308
**sqrt,**
como función de MapForce (en lang | math functions), 677
**starts-with,**
como función de MapForce (en core | string functions), 629
**static-node-annotation,**
como función de MapForce (en core | node functions), 591
**static-node-name,**
como función de MapForce (en core | node functions), 592
**string,**
como función de MapForce (en core | conversion functions), 570

© 2019-2025 Altova GmbH

**string,**
como función de MapForce (en xpath2 | accessors), 699
**string-as- QName,**
como función de MapForce (en lang | QName functions), 679
**string-compare,**
como función de MapForce (en lang | string functions), 696
**string-compare-ignore-case,**
como función de MapForce (en lang | string functions), 697
**string-join,**
como función de MapForce (en core | aggregate functions), 552
**string-length,**
como función de MapForce (en core | string functions), 629
**Stylevision,**
definir hojas de estilos SPS para asignaciones, 806
**substitute-missing,**
como función de MapForce (en core | sequence functions), 623
**substitute-missing-with-xsi-nil,**
como función de MapForce (en core | node functions), 592
**substitute-null,**
como función de MapForce (en db functions), 641
**substring-after,**
como función de MapForce (en core | string functions), 630
como función de MapForce (en core | string functions), 630
**substring-before,**
como función de MapForce (en core | string functions), 631
**subtract,**
como función de MapForce (en core | math functions), 587
**sum,**
como función de MapForce (en core | aggregate functions), 553
**Superposición de datos,**
información, 812
**Sybase,**
conectarse por JDBC, 239
**system-property,**
como función de MapForce (en xslt | xslt functions), 760

# T

**Tabla de búsqueda,**
propiedades, 453
**Tablas de búsqueda,**
usar en la asignación, 445
**tan,**
como función de MapForce (en lang | math functions), 678
**Teradata,**
conectarse por JDBC, 240
conectarse por ODBC, 241
**Texto,**
archivos de, definir campos clave, 349
asignación de datos de archivos de texto, 344
**time-from-datetime,**
como función de MapForce (en lang | datetime functions), 659
**timezone,**
como función de MapForce (en lang | dateTime functions), 659
**Tipo complejo,**
ordenar, 426
**Tipo simple,**
ordenar, 426
**Tipos,**
derivados (xsi:type), 144
integrados, 1034
**Tipos de conexión,**
basada en destino, 52
basada en origen, 52
basadas en destino con contenido mixto, 52
conexiones basadas en el destino vs. conexiones basadas en el destino, 52
de copia total, 57
estándar, 52
estándar con contenido mixto, 52
mixta, 52
secundarios equivalentes, 55
**Tipos derivados,**
asignaciones, 144
**tokenize,**
como función de MapForce (en core | string functions), 632
**tokenize-by-length,**
como función de MapForce (en core | string functions), 634
**tokenize-regexp,**
como función de MapForce (en core | string functions), 637
**Trabajar con conexiones,**
de copia total, 57
secundarios equivalentes, 55
**Transacciones de BD,**
habilitar para procedimientos almacenados, 329
**Transformaciones,**
RaptorXML Server, 831
**translate (in core | string functions),**
como función de MapForce, 638

**Transmisión por secuencias de datos,**
definición, 38
**true,**
como función de MapForce (en xpath2 | boolean functions), 701
**Tutoriales,**
archivos de ejemplo, 103
asignación encadenada, 119
básicos, 104
componentes de paso a través, 119
duplicar elementos de entrada, 114
nombres de archivo dinámicos, 128
transformación, 104
un archivo de origen a un archivo de destino, 104
varios archivo de origen a varios archivos de destino, 128
varios archivos de origen a un solo destino, 114

# U

**unary-minus,**
como función de MapForce (en lang | math functions), 678
**Unicode,**
intercalación de punto de código, 426
**unparsed-entity-uri,**
como función de MapForce (en xslt | xslt functions), 761
**uppercase,**
como función de MapForce (en lang | string functions), 698
**URI,**
en DTDs, 138
**URI de espacio de nombres,**
DTD, 138
**URL,**
agregar archivos como componentes desde, 38
**Uso de Internet,**
en los productos de Altova, 1259

# V

**Valores (ventana),**
información, 812, 818
pestaña Contexto, 818
pestaña Historial, 818
pestaña Relacionados, 818
**Valores predeterminados, 469**
crear, 471
**Valores predeterminados y funciones de nodo,**
anotación, 483
aplicar condicionalmente, 474
ayudas visuales, 471
casos de uso, 474
configurar, 471
configurar reglas, 471
crear, 471
filtro, 474
fractionDigits, 483
iconos de las funciones, 469
lado de entrada, 469
lado de salida, 469
length, 483
maxLength, 483
metadatos, 483
minLength, 483
precision, 483
reglas de bloqueo, 474
scale, 483
sobrescribir reglas heredadas, 474
totalDigits, 483
valores predeterminados para nodos sin conexión, 474
**Variables,**
agregar a la asignación, 386
basadas en DB, 384
cambiar el alcance de, 390
complejas, 384
de SPL, 1024
ejemplos de uso, 393, 394, 395
Variables, 384
**Ventanas,**
Mensajes, 27
**Vista, 1121**
Adelante, 1121
Administrar bibliotecas, 1121
Atrás, 1121
Barra de estado, 1121
Bibliotecas, 1121
comando de menú, 1121
como menú de aplicación, 1121
Depurar ventanas, 1121
Mensajes, 1121
Mostrar anotaciones, 1121
Mostrar biblioteca en el título de la función, 1121
Mostrar información rápida, 1121

© 2019-2025 Altova GmbH

**Vista, 1121**
Mostrar los conectores del componente seleccionado, 1121
Mostrar los conectores desde su origen a su destino, 1121
Mostrar tipos, 1121
Opciones de visualización XBRL, 1121
Ventana proyectos, 1121
Vista general, 1121
Zoom, 1121
**Vista Texto,**
ajuste automático de línea, 71
buscar, 74
formato pretty-print, 71
guías de sangría, 71
marcadores de espacios en blanco, 71
marcadores de final de línea, 71
navegar, 71
números de línea, 71
plegamiento de código, 71
resaltado de texto, 71
ventana Información, 71
zoom, 71
**Visual Basic,**
integration of MapForce, 1068
**Visual Studio,**
adding the MapForce ActiveX Controls to the toolbox, 1044

# W

**WebDAV Server,**
agregar archivos como componentes desde, 38
**weekday,**
como función de MapForce (en lang | dateTime functions), 660
**weeknumber,**
como función de MapForce (en lang | dateTime functions), 660
**WHERE,**
componente WHERE de SQL, 437
**Windows,**
Administrar bibliotecas, 23
Asignación, 23
Bibliotecas, 23
Cascada, 1142
compatibilidad con productos de Altova, 1258
Cuadro de diálogo de Windows, 1142
En mosaico vertical/horizontal, 1142
Proyecto, 23
Tema, 1142
Tema clásico, 1142
Tema ligero, 1142
Tema oscuro, 1142
Ventana de asignaciones múltiples, 23
Vista general, 23
**Word,**
documentación de la asignación, 799
**Word 2007+,**
vista previa del resultado de la asignación como, 847
**write-binary-file,**
como función de MapForce (en lang | string functions), 665

# X

**XML,**
agregar esquema, 139
agregar referencia de la DTD, 139
Archivo del esquema, 139
Archivo hoja de estilos StyleVision Power, 139
archivo XML de entrada, 139
archivo XML de salida, 139
asignar datos CSV a XML, 344
BOM, 139
como destino de la asignación, 344
configurar componentes, 139
configurar el cifrado, 139
convertir valores en tipos de destino, 139
declaración, 139
Declaración XML, 139
escribir en campo de BD, 308
firma digital, 139
formato pretty-print, 139
guardar todas las rutas de acceso de archivos como relativas al archivo MFD, 139
independiente, 139
independiente="yes", 139
min/maxOccurs, 139
nombre del componente, 139
orden de bytes, 139
**XML data,**
escribir en campos de tipo XML, 304
leer campos de tipo XML, 304
**XQuery,**

© 2019-2025 Altova GmbH

**XQuery,**
agregar funciones personales, 514, 515
importar módulos, 515
**xs: any (xs:anyAttribute), 151**
**xsi:nil,**
como atributo en una instancia XML, 146
**xsi:type,**
asignación de tipos derivados, 144
**XSLT,**
agregar funciones personales, 508
espacio de nombres de plantilla, 508
quitar funciones personales, 508

# Y

**year-from- duration,**
como función de MapForce (en lang | datetime functions), 661
**year-from-datetime,**
como función de MapForce (en lang | datetime functions), 661

# Z

**Z to A,**
componente de ordenación, 426

© 2019-2025 Altova GmbH