Altova RaptorXML+XBRL Server 2018

Manual de referencia y del usuario

Manual del usuario y referencia de Altova RaptorXML+XBRL Server 2018

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: 2018

© 2018 Altova GmbH

Contenido

1 Introducción a RaptorXML+XBRL Server 3

1.1 Ediciones e interfaces ... 5
1.2 Requisitos del sistema ... 7
1.3 Características ... 8
1.4 Especificaciones compatibles ... 10

2 Instalar RaptorXML+XBRL Server 14

2.1 Instalación y configuración en Windows ... 15
2.1.1 Instalación en Windows ... 15
2.1.2 Asignación de licencias en Windows ... 17
2.2 Instalación y configuración en Linux ... 22
2.2.1 Instalación en Linux ... 22
2.2.2 Asignación de licencias en Linux ... 25
2.2.3 Notas sobre configuración del entorno ... 28
2.3 Instalación y configuración en macOS ... 29
2.3.1 Instalación en macOS ... 29
2.3.2 Asignación de licencias en macOS ... 32
2.3.3 Notas sobre configuración del entorno ... 34
2.4 Catálogos XML ... 35
2.4.1 Cómo funcionan los catálogos ... 35
2.4.2 Mecanismo de catalogación XML de Altova ... 37
2.4.3 Variables para ubicaciones de sistemas Windows ... 39
2.5 Recursos globales ... 41
2.6 Problemas de seguridad ... 43

3 Interfaz de la línea de comandos (ILC) 46

3.1 Comandos para validar XML, DTD, XSD ... 50
3.1.1 valxml-withdtd (xml) ... 50
3.1.2 valxml-withxsd (xsi) ... 54
3.1.3 valdtd (dtd) ... 61
3.1.4 valxsd (xsd) ... 64
3.2 Comandos para comprobar el formato ... 71

3.2.1 wfxml ... 71
3.2.2 wfdtd ... 75
3.2.3 wfany ... 78
3.3 Comandos de validación XBRL ... 81
3.3.1 valxbrl (xbrl) ... 81
3.3.2 valinlinexbrl (ixbrl) ... 95
3.3.3 valxbrltaxonomy (dts) ... 110
3.3.4 valtaxonomypackage (taxpkg) ... 119
3.3.5 valxbrlver ... 124
3.4 Comandos XSLT ... 130
3.4.1 xslt ... 130
3.4.2 valxslt ... 138
3.5 Comandos XQuery ... 145
3.5.1 xquery ... 145
3.5.2 xqueryupdate ... 152
3.5.3 valxquery ... 161
3.5.4 valxqueryupdate ... 167
3.6 Comandos XML Signature ... 173
3.6.1 xmlsignature-sign ... 173
3.6.2 xmlsignature-verify ... 178
3.6.3 xmlsignature-update ... 181
3.6.4 xmlsignature-remove ... 184
3.7 Comandos JSON/Avro ... 186
3.7.1 avroextractschema ... 186
3.7.2 valavro (avro) ... 189
3.7.3 valavrojson (avrojson) ... 192
3.7.4 valavroschema (avroschema) ... 196
3.7.5 valjsonschema (jsonschema) ... 199
3.7.6 valjson (json) ... 202
3.7.7 wfjson ... 206
3.8 Comando valany ... 210
3.9 Comando script ... 212
3.10 Comandos de ayuda y licencias ... 213
3.10.1 help ... 213
3.10.2 licenseserver ... 214
3.10.3 assignlicense ... 215
3.10.4 verifylicense ... 216
3.11 Comandos de localización ... 218
3.11.1 exportresourcestrings ... 218
3.11.2 setdeflang ... 219

3.12 Opciones ... 221
3.12.1 Catálogos, recursos globales, archivos ZIP ... 221
3.12.2 Mensajes, errores, ayuda, tiempo de espera y versión ... 222
3.12.3 Procesamiento ... 223
3.12.4 XBRL ... 224
3.12.5 XML ... 232
3.12.6 XSD ... 234
3.12.7 XQuery ... 236
3.12.8 XSLT ... 238
3.12.9 JSON/Avro ... 240

4 Interfaz HTTP 242

4.1 Preparar el servidor ... 244
4.1.1 Iniciar el servidor ... 244
4.1.2 Probar la conexión ... 246
4.1.3 Configurar el servidor ... 247
4.1.4 Configuración HTTPS ... 252
4.1.5 Configurar el cifrado SSL ... 253
4.2 Solicitudes cliente ... 257
4.2.1 Iniciar trabajos con POST ... 259
Ejemplo nº1 (con llamadas): validar XML ... 263
Ejemplo nº2: usar un catálogo para buscar el esquema ... 265
Ejemplo nº3: usar archivos ZIP ... 266
Pruebas con CURL ... 267
4.2.2 Respuesta del servidor a solicitudes POST ... 270
4.2.3 Obtener el documento de resultados ... 272
4.2.4 Obtener los documentos de salida/errores ... 276
4.2.5 Liberar espacio tras el procesamiento ... 278

5 Interfaces API de Python y .NET 280

5.1 API de Python ... 282
5.1.1 Versiones de las API de Python ... 283
5.1.2 RaptorXML+XBRL Server como paquete Python ... 285
5.2 API de .NET Framework ... 288

6 Interfaz Java 290

6.1 Ejemplo de proyecto Java ... 292
6.2 Interfaces de RaptorXML para Java ... 295

6.2.1 RaptorXMLFactory ... 295
6.2.2 XMLValidator ... 299
6.2.3 XSLT ... 304
6.2.4 XQuery ... 310
6.2.5 XBRL ... 315
6.2.6 RaptorXMLException ... 324
6.3 Enumeraciones de RaptorXML para Java ... 325
6.3.1 RaptorXMLFactory ... 325
6.3.2 XML Validation ... 325
6.3.3 XSLT y XQuery ... 329
6.3.4 XBRL ... 330

7 Interfaces COM y .NET 334

7.1 Notas sobre la interfaz COM ... 335
7.2 Notas sobre la interfaz NET ... 336
7.3 Lenguajes de programación ... 338
7.3.1 Ejemplo de COM: VBScript ... 339
7.3.2 Ejemplo de .NET: C# ... 341
7.3.3 Ejemplo de .NET: Visual Basic .NET ... 343
7.4 Referencia de la API ... 346
7.4.1 Interfaces ... 346
IServer ... 347
IXMLValidator ... 350
IXSLT ... 355
IXQuery ... 360
IXBRL ... 365
7.4.2 Enumeraciones ... 373
ENUMAssessmentMode ... 374
ENUMErrorFormat ... 374
ENUMLoadSchemalocation ... 375
ENUMQueryVersion ... 376
ENUMSchemaImports ... 377
ENUMSchemaMapping ... 378
ENUMTableOutputFormat ... 378
ENUMValidationType ... 379
ENUMWellformedCheckType ... 381
ENUMXBRLValidationType ... 382
ENUMIXBRLVersion ... 383
ENUMIXBRLUriStrategy ... 383

ENUMXMLValidationMode ... 384
ENUMXQueryVersion ... 385
ENUMXQueryUpdatedXML ... 386
ENUMXSDVersion ... 386
ENUMXSLTVersion ... 387

8 Información adicional 390

8.1 Códigos de salida ... 391
8.2 Sugerencias sobre ubicación de esquemas ... 392
8.3 Parámetros de XBRL Formula ... 393
8.3.1 Formatos de los parámetros de fórmula ... 393
8.3.2 Usar parámetros de fórmula ... 394

9 Información sobre motores XSLT y XQuery 400

9.1 XSLT 1.0 ... 401
9.2 XSLT 2.0 ... 402
9.3 XSLT 3.0 ... 404
9.4 XQuery 1.0 ... 405
9.5 XQuery 3.1 ... 409

10 Funciones XSTL y XPath/XQuery 412

10.1 Funciones de extensión de Altova ... 414
10.1.1 Funciones XSLT ... 415
10.1.2 Funciones XPath/XQuery: de fecha y hora ... 418
10.1.3 Funciones XPath/XQuery: de geoubicación ... 433
10.1.4 Funciones XPath/XQuery: relacionadas con imágenes ... 441
10.1.5 Funciones XPath/XQuery: numéricas ... 446
10.1.6 Funciones XPath/XQuery: de secuencia ... 449
10.1.7 Funciones XPath/XQuery: de cadena ... 456
10.1.8 Funciones XPath/XQuery varias ... 462
10.1.9 Funciones de extensión para gráficos ... 463
Estructura XML de los datos de gráficos ... 468
Ejemplo: funciones para gráficos ... 473
10.1.10 Funciones para códigos de barras ... 476
10.2 Funciones de extensión varias ... 479
10.2.1 Funciones de extensión Java ... 479
Archivos de clases definidos por el usuario ... 481

Archivos JAR definidos por el usuario ... 484
Constructores ... 485
Métodos estáticos y campos estáticos ... 485
Métodos de instancia y campos de instancia ... 486
Tipos de datos: conversión de XPath/XQuery en Java ... 487
Tipos de datos: conversión de Java en XPath/XQuery ... 488
10.2.2 Funciones de extensión .NET ... 489
Constructores ... 491
Metodos estáticos y campos estáticos ... 492
Métodos de instancia y campos de instancia ... 493
Tipos de datos: conversión de XPath/XQuery en .NET ... 494
Tipos de datos: conversión de .NET en XPath/XQuery ... 495
10.2.3 Funciones XBRL para XSLT ... 495
10.2.4 Scripts MSXSL para XSLT ... 496

11 Altova LicenseServer 500

11.1 Información de red ... 502
11.2 Actualizar LicenseServer ... 504
11.3 Instalación en Windows ... 505
11.4 Instalación en Linux ... 507
11.5 Instalación en macOS ... 510
11.6 Altova ServiceController ... 512
11.7 Asignación de licencias ... 513
11.7.1 Iniciar LicenseServer ... 514
11.7.2 Abrir la página de configuración de LicenseServer (Windows) ... 515
11.7.3 Abrir la página de configuración de LicenseServer (Linux) ... 518
11.7.4 Abrir la página de configuración de LicenseServer (macOS) ... 520
11.7.5 Cargar licencias en LicenseServer ... 523
11.7.6 Registrar productos con LicenseServer ... 526
Registrar productos de escritorio de Altova ... 527
Registrar FlowForce Server ... 528
Registrar MapForce Server ... 532
Registrar MobileTogether Server ... 534
Registrar RaptorXML(+XBRL) Server ... 534
Registrar StyleVision Server ... 536
11.7.7 Asignar licencias a productos registrados ... 537
11.8 Referencia de la página de configuración ... 544
11.8.1 Repertorio de licencias ... 544
11.8.2 Gestión de clientes ... 551

11.8.3 Supervisión de clientes 556
11.8.4 Opciones de configuración 557
11.8.5 Mensajes, Cerrar sesión 565
11.9 Restaurar la contraseña 566

Índice

Altova RaptorXML+XBRL Server 2018

Introducción a RaptorXML+XBRL Server

1 Introducción a RaptorXML+XBRL Server

Altova RaptorXML+XBRL Server (en adelante RaptorXML) es el rapidísimo motor XML y XBRL de tercera generación de Altova, optimizado para los estándares más recientes y para entornos de informática en paralelo. RaptorXML es compatible con múltiples plataformas y aprovecha la omnipresencia actual de equipos multinúcleo para ofrecer rapidísimas funciones de procesamiento de datos XML y XBRL.

* **Nota:** las funciones de procesamiento XBRL solamente están disponibles en RaptorXML+XBRL Server (no están disponibles en RaptorXML Server).

Sitio web de Altova: Servidor de validación XML, Validación XML

Ediciones y sistemas operativos

Altova ofrece dos ediciones diferentes de RaptorXML, diseñadas para satisfacer diferentes requisitos. Estas dos ediciones se describen en el apartado Ediciones e interfaces. RaptorXML está disponible para Windows, Linux y macOS. Para más información consulte el apartado Requisitos del sistema.

Características y especificaciones compatibles

RaptorXML ofrece funciones de validación XML y XBRL, transformación XSLT y ejecución de XQuery dotadas de numerosas y potentes opciones. Para ver la lista de características y funciones clave de RaptorXML, consulte el apartado Características. En el apartado Especificaciones compatibles se enumeran todas las especificaciones con las que cumple RaptorXML. Para más información visite el sitio web de Altova.

Certificación XBRL

XMLSpy ha obtenido la certificación de XBRL Internacional. Para más información acerca de la certificación XBRL, vaya a la página web (en inglés) XBRL Software Certification.

Esta documentación

La presente documentación está incluida en la aplicación y también está disponible en el sitio web de Altova. Tenga en cuenta que el explorador Chrome tiene una restricción que no permite expandir las entradas de la tabla de contenido cuando la documentación se abre localmente. Sin embargo, si abre la documentación desde un servidor web, la tabla de contenido funciona correctamente en Chrome.

La presente documentación se divide en varias secciones:

- Introducción a RaptorXML (la presente sección)
- Instalar RaptorXML
- Interfaz de la línea de comandos
- Interfaz HTTP

- Interfaces API de Python y .NET
- Interfaz Java
- Interfaz COM/.NET
- Información sobre motores XSLT y XQuery
- Funciones XSLT y XPath/XQuery
- Altova LicenseServer

Última actualización: 6/22/2018

 © 2018 Altova GmbH

1.1 Ediciones e interfaces

Altova ofrece dos ediciones distintas de RaptorXML:

- ***RaptorXML Server:*** un rapidísimo motor de procesamiento XML compatible con XML, XML Schema, XSLT, XPath y XQuery, entre otros estándares.

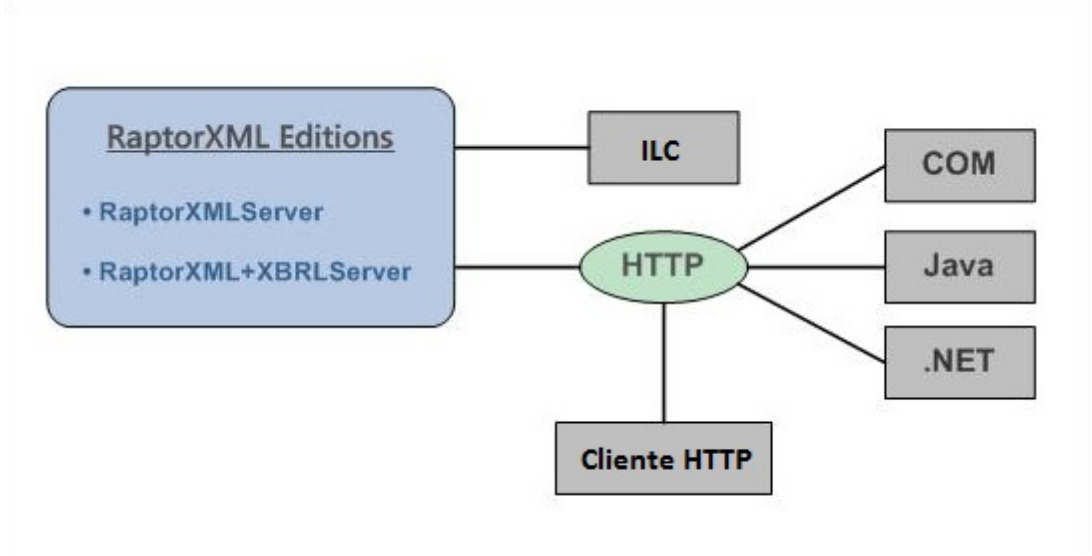
- ***RaptorXML+XBRL Server:*** ofrece todas las características de RaptorXML Server y funciones de procesamiento y validación compatibles con todos los estándares XBRL.

Interfaces

Puede acceder a RaptorXML+XBRL Server a través de varias interfaces:

- Una **interfaz de la línea de comandos** (en adelante ***ILC***).
- Una **interfaz COM** para sistemas Windows.
- Una **interfaz .NET** para sistemas Windows.
- Una **interfaz Java** para sistemas Windows, Linux y Mac OS.
- Una **interfaz HTTP** a la que se puede acceder desde un cliente HTTP.
- Una **interfaz Python** con la que puede acceder y procesar partes de documentos mediante scripts Python y con ayuda de las API de Python de RaptorXML. Los scripts se pueden enviar por la interfaz de la línea de comandos o por la interfaz HTTP. Además puede disponer de las funciones de RaptorXML en forma de paquete de Python. Es decir, podrá usar todas las funciones de RaptorXML dentro del código Python junto con paquetes de otros autores.
- Una **API de .NET Framework**.

El diagrama que aparece a continuación muestra cómo se accede a RaptorXML Server y RaptorXML+XBRL Server a través de las diferentes interfaces.



Observe que las interfaces COM, Java y .NET usan el protocolo HTTP para conectarse a RaptorXML. Los scripts Python se pueden enviar a RaptorXML a través de la interfaz HTTP y de la

línea de comandos.

Interfaz de la línea de comandos (ILC)
Permite validar XML (y otros documentos), transformar XSLT y ejecutar XQuery desde la línea de comandos. Para más información y aprender a usarla consulte la sección *Interfaz de la línea de comandos*.

Interfaz HTTP
Puede acceder a todas las funciones de las ediciones servidor a través de una interfaz HTTP. Las solicitudes cliente se hacen en formato JSON. Cada solicitud se asigna a un directorio de trabajo en el servidor (en este directorio se guardan los archivos de salida) y las respuestas del servidor al cliente incluyen toda la información relacionada con el trabajo. Para más información consulte la sección *Interfaz HTTP*.

API de Python
Junto con un comando de la ILC o una solicitud HTTP, puede suministrar un script Python que acceda a documentos especificados en el comando o en la solicitud. El acceso al documento se consigue a través de las API Python para XML, XSD y XBRL. Para más información sobre esta interfaz y aprender a usarla consulte la sección *API de Python*. Las funciones de RaptorXML también están disponibles en forma de paquete de Python. Es decir, podrá usar todas las funciones de RaptorXML dentro del código Python junto con paquetes de otros autores.

API de .NET Framework
La API de .NET Framework permite incrustar RaptorXML+XBRL Server en cualquier aplicación compatible con .NET Framework. Para más información sobre esta API consulte la sección *API de .NET Framework*.

Interfaz COM
Puede usar RaptorXML a través de la interfaz COM y, por tanto, puede ser utilizada por aplicaciones y lenguajes de script compatibles con COM. La compatibilidad con al interfaz COM está implementada para interfaces sin formato e interfaces de envío. Los datos de entrada se pueden suministrar como archivos o como cadenas de texto en scripts y en los datos de la aplicación.

Interfaz Java
Las funciones de RaptorXML también están disponibles como clases Java que se pueden usar en programas Java. Por ejemplo, hay clases Java que ofrecen características de validación XML, transformación XSLT y ejecución de XQuery.

Interfaz .NET
RaptorXML ofrece un archivo DLL construido como contenedor de RaptorXML que permite a los usuarios de .NET conectarse a las funciones de RaptorXML. Además, RaptorXML ofrece un ensamblado de interoperabilidad principal firmado por Altova. Los datos de entrada se pueden suministrar como archivos o como cadenas de texto en scripts y en los datos de la aplicación.

 © 2018 Altova GmbH

1.2 Requisitos del sistema

RaptorXML+XBRL Server es compatible con estos sistemas operativos:

- *Windows*

  Windows 7 SP1 con actualización de la plataforma, Windows 8, Windows 10

- *Windows Server*

  Windows Server 2008 R2 SP1 con actualización de la plataforma o superior

- *Linux*

  - CentOS 6 o superior
  - RedHat 6 o superior
  - Debian 7 o superior
  - Ubuntu 12.04 o superior

  Las bibliotecas que aparecen a continuación son un requisito previo para la instalación y ejecución de la aplicación. Si los paquetes que aparecen en esta tabla no están en su equipo Linux, ejecute el comando `yum` (o `apt-get` si procede) para instalarlos.

Requisito para	CentOS, RedHat	Debian	Ubuntu
LicenseServer	krb5-libs	libgssapi-krb5-2	libgssapi-krb5-2
RaptorXML+XBRL Server	qt4, krb5-libs, qt-x11	libqtcore4, libqtgui4, libgssapi-krb5-2	libqtcore4, libqtgui4, libgssapi-krb5-2

- (*Mac) OS X, macOS*

  OS X 10.10 o superior

RaptorXML es compatible con equipos de 32 y 64 bits. Estos son los núcleos basados en conjuntos de instrucciones x86 y amd64 (x86-64): Intel Core i5, i7, XEON E5.

Para usar RaptorXML+XBRL Server a través de una interfaz COM el usuario debe tener privilegios para usar la interfaz COM, es decir, para registrar la aplicación y ejecutar las aplicaciones y scripts pertinentes.

1.3 Características

RaptorXML ofrece todas las funciones que aparecen a continuación. La mayoría de las funciones corresponden a la línea de comandos y a la interfaz COM. La principal diferencia es que la interfaz COM en Windows permite construir documentos a partir de cadenas de texto con el código de aplicación o de script (en lugar de hacer referencia a archivos XML, XBRL, DTD, esquemas XML, XSLT o XQuery).

Validación XML y XBRL

- Valida documentos XML y XBRL con esquemas XML y DTD internos o externos.
- Revisa el formato de documentos XML, DTD, XML Schema, XSLT y XQuery.
- Valida taxonomías XBRL y documentos XBRL con taxonomías XBRL.
- Ejecuta fórmulas XBRL y aserciones de validación.
- Representa tablas XBRL.
- Compatible con las especificaciones XBRL 2.1, Dimensions 1.0 y Formula 1.0 y con la recomendación Table Linkbase 1.0.
- Compatible con Inline XBRL
- Compatible con paquetes de taxonomías XBRL

Transformaciones XSLT

- Transforma XML usando documentos XSLT 1.0, 2.0 o 3.0 suministrados por el usuario.
- Los documentos XML y XSLT se pueden suministrar en forma de archivpo (por su URL) o, en el caso de la interfaz COM, en forma de cadena de texto.
- Los resultados se devuelven en forma de archivo (en la ubicación elegida por el usuario) o, en el caso de la interfaz COM, en forma de cadena de texto.
- Los parámetros XSLT se pueden suministrar a través de la línea de comandos o de la interfaz de COM.
- Las funciones de extensión de Altova, así como las funciones de extensión XBRL, Java y .NET, permiten un procesamiento más especializado. Por ejemplo, permiten crear ciertas características como gráficos y códigos de barras en los documentos de salida.

Ejecución de XQuery

- Ejecuta documentos XQuery 1.0 y 3.0.
- Los documentos XQuery y XML se pueden suministrar en forma de archivo (por su URL) o, en el caso de la interfaz COM, en forma de cadena de texto.
- Los resultados se devuelven en forma de archivo (en la ubicación elegida por el usuario) o, en el caso de la interfaz COM, en forma de cadena de texto.
- Las variables XQuery externas se pueden suministrar a través de la línea de comandos o de la interfaz de COM.
- Opciones de serialización: codificación de salida, método de codificación (es decir, si el resultado es en XML, XHTML, HTML o texto), omisión de la declaración XML y sangría.

 © 2018 Altova GmbH

Validación y conversión de datos JSON y Avro

- Validación de documentos de esquema JSON y Avro.
- Validación de instancias JSON con esquemas JSON y esquemas Avro.
- Validación de binarios Avro.
- Conversión de binarios Avro en esquemas Avro y de datos Avro en formato JSON.
- Conversión de datos Avro JSON en binarios Avro.

Características de alto rendimiento

- Optimizaciones de código de altísimo rendimiento.
  - Implementaciones nativas de conjuntos de instrucciones.
  - Versión de 32 bits.
- Bajísima superficie de memoria.
  - Representación en memoria de XML Information Set extremadamente compacta.
  - Validación de instancias por transmisión por secuencias.
- Características compatibles con múltiples plataformas.
- Código altamente adaptable para informática en paralelo y equipos multi-CPU/ multinúcleo.
- Carga, validación y procesamiento en paralelo.

Características para desarrolladores

- Avanzadas funciones de generación de informes de errores.
- Modo servidor Windows y modo demonio Unix (a través de opciones de la línea de comandos).
- Intérprete Python 3.x para scripting.
- Todas las funciones de RaptorXML en un paquete de Python para importar las funciones como biblioteca Python.
- API de .NET Framework para acceder al modelo de datos XML y XBRL subyacente.
- API de COM en la plataforma Windows.
- API de Java en todas las plataformas.
- Funciones de extensión XPath, Java, .NET, XBRL, etc.
- Serialización de secuencias de datos.
- Servidor HTTP integrado con API de validación REST.

Para más información consulte el apartado Especificaciones compatibles y visite el sitio web de Altova.

1.4 Especificaciones compatibles

RaptorXML es compatible con todas estas especificaciones.

Recomendaciones del W3C

Sitio web: World Wide Web Consortium (W3C)

- Extensible Markup Language (XML) 1.0 (Fifth Edition)
- Extensible Markup Language (XML) 1.1 (Second Edition)
- Namespaces in XML 1.0 (Third Edition)
- Namespaces in XML 1.1 (Second Edition)
- XML Information Set (Second Edition)
- XML Base (Second Edition)
- XML Inclusions (XInclude) Version 1.0 (Second Edition)
- XML Linking Language (XLink) Version 1.0
- XML Schema Part 1: Structures Second Edition
- XML Schema Part 2: Datatypes Second Edition
- W3C XML Schema Definition Language (XSD) 1.1 Part 1: Structures
- W3C XML Schema Definition Language (XSD) 1.1 Part 2: Datatypes
- XPointer Framework
- XPointer xmlns() Scheme
- XPointer element() Scheme
- XML Path Language (XPath) Version 1.0
- XSL Transformations (XSLT) Version 1.0
- XML Path Language (XPath) 2.0 (Second Edition)
- XSL Transformations (XSLT) Version 2.0
- XQuery 1.0: An XML Query Language (Second Edition)
- XQuery 1.0 and XPath 2.0 Functions and Operators (Second Edition)
- XSLT 2.0 and XQuery 1.0 Serialization (Second Edition)
- XML Path Language (XPath) 3.0
- XML Path Language (XPath) 3.1
- XQuery 3.0: An XML Query Language
- XQuery Update Facility 1.0
- XPath and XQuery Functions and Operators 3.0
- XSLT and XQuery Serialization 3.0

Borradores y recomendaciones candidatas del W3C

Sitio web: World Wide Web Consortium (W3C)

- XSL Transformations (XSLT) Version 3.0 (subset)
- XQuery 3.1: An XML Query Language
- XPath and XQuery Functions and Operators 3.1
- XQuery Update Facility 3.0
- XSLT and XQuery Serialization 3.1

 © 2018 Altova GmbH

Estándares OASIS

Sitio web: OASIS Standards

- XML Catalogs V 1.1 - OASIS Standard V1.1

Estándares JSON/Avro

Sitios web: JSON Schema y Apache Avro

- JSON Schema Draft 4
- Apache Avro Schema

Recomendaciones XBRL

Sitio web: Extensible Business Reporting Language (XBRL)

- XBRL 2.1
- Dimensions 1.0
- Extensible Enumerations 1.0 (y la propuesta de recomendación obsoleta 1.1).
- Formula Specification 1.0
  - Filtros de aspecto cover
  - Nivel de gravedad de aserciones
  - Filtros booleanos
  - Filtros de concepto
  - Filtros de relación de conceptos
  - Aserciones de coherencia
  - Implementación de funciones personales
  - Filtros de dimensión
  - Filtros de entidad
  - Aserciones de existencia
  - Fórmula
  - Registro de funciones
  - Filtros genéricos
  - Mensajes genéricos
  - Filtros implicit
  - Filtros match
  - Filtros de período
  - Filtros relativos
  - Filtros de segmento
  - Filtros de tupla
  - Filtros de unidad
  - Validación
  - Mensajes de validación
  - Aserciones de valor
  - Filtros de valor
  - Variables
- Table Linkbase 1.0
- Function Registry 1.0

- Generic Links 1.0
  - Referencias genéricas
  - Etiquetas genéricas
- Units Registry 1.0
- Inline XBRL 1.0 y 1.1
- Taxonomy Packages 1.0
- XBRL Versioning 1.0

 © 2018 Altova GmbH

Altova RaptorXML+XBRL Server 2018

Instalar RaptorXML+XBRL Server

2 Instalar RaptorXML+XBRL Server

Esta sección describe el procedimiento para instalar y configurar RaptorXML+XBRL Server correctamente. Incluye varios apartados con información sobre:

- Cómo instalar RaptorXML y asignarle licencias en sistemas Windows, Linux y macOS
- Cómo usar los catálogos XML.
- Cómo trabajar con los recursos globales de Altova.
- Problemas de seguridad relacionados con RaptorXML.

RaptorXML tiene opciones especiales que admiten el uso de catálogos XML y de recursos globales de Altova, características que mejoran la portabilidad y modularidad del entorno en el que se trabaja.

Nota: el apartado Problemas de seguridad explica cómo configurar importantes soluciones de seguridad.

 © 2018 Altova GmbH

2.1 Instalación y configuración en Windows

Esta sección explica cómo instalar RaptorXML+XBRL Server y asignarle licencias en sistemas Windows.

Instalación en Windows

- Requisitos del sistema
- Instalar RaptorXML+XBRL Server
- Altova LicenseServer
- Versiones de LicenseServer
- Licencia de prueba
- Ubicación de la carpeta de aplicación

Asignación de licencias en Windows

- Iniciar el controlador de servicios ServiceController
- Iniciar LicenseServer
- Iniciar RaptorXML+XBRL Server
- Registrar RaptorXML+XBRL Server
- Asignar licencias

2.1.1 Instalación en Windows

El proceso de instalación y configuración de RaptorXML+XBRL Server en Windows se describe a continuación.

Requisitos del sistema

Windows

Windows 7 SP1 con actualización de la plataforma, Windows 8, Windows 10

Windows Server

Windows Server 2008 R2 SP1 con actualización de la plataforma o superior

Instalar RaptorXML+XBRL Server

RaptorXML+XBRL Server se puede instalar en sistemas Windows:

- **Como producto servidor independiente llamado RaptorXML+XBRL Server:** para instalar RaptorXML+XBRL Server como producto independiente, descargue y ejecute el instalador de RaptorXML+XBRL Server y siga las instrucciones en pantalla.

- **Como parte del paquete de instalación de FlowForce Server:** para instalar RaptorXML+XBRL Server como parte del paquete de FlowForce Server descargue y

ejecute el instalador de FlowForce Server. Siga las instrucciones en pantalla y marque la opción para instalar RaptorXML+XBRL Server.

Los programas de instalación de RaptorXML+XBRL Server y FlowForce Server están disponibles en el centro de descargas de Altova (http://www.altova.com/es/download.html).

Una vez completada la instalación, el ejecutable de RaptorXML+XBRL Server estará en esta ubicación predeterminada:

```
<CarpetaArchivosPrograma>\Altova\RaptorXMLXBRLServer2018\bin
\RaptorXMLXBRL.exe
```

El programa de instalación creará todos los registros necesarios para usar RaptorXML +XBRL Server desde la interfaz de COM, como interfaz Java y en el entorno .NET. El programa de instalación realizará todos los registros necesarios para utilizar RaptorXML +XBRL Server a través de una interfaz COM, como interfaz Java y en el entorno .NET. Esto incluye el registro del ejecutable de RaptorXML+XBRL Server como objeto servidor de COM, instalar `RaptorXMLLib.dll` (para su uso en la interfaz Java) en el directorio `WINDIR \system32\` y añadir el archivo `Altova.RaptorXML.dll` a la biblioteca de referencia .NET.

Altova LicenseServer

- Para que RaptorXML+XBRL Server funcione debe tener asignada una licencia desde un servidor Altova LicenseServer de la red.
- El programa de instalación de RaptorXML+XBRL Server o de FlowForce Server para sistemas Windows ofrece una opción para descargar e instalar Altova LicenseServer junto con RaptorXML+XBRL Server o con FlowForce Server.
- Si en la red ya hay instalado un servidor Altova LicenseServer, no necesita instalar otro LicenseServer a no ser que se necesite una versión más reciente (ver el siguiente apartado *versiones de LicenseServer*).
- Durante el proceso de instalación de RaptorXML+XBRL Server o de FlowForce Server, puede seleccionar si también se instala Altova LicenseServer.

Para más información sobre cómo registrar RaptorXML+XBRL Server y asignarle licencias con Altova LicenseServer, consulte la sección Asignación de licencias en Windows.

Versiones de LicenseServer

- Los productos servidor de Altova deben tener una licencia con la versión de LicenseServer correspondiente a la versión de RaptorXML+XBRL Server instalada o con una versión posterior de LicenseServer.
- La versión de LicenseServer correspondiente a la versión de RaptorXML+XBRL Server aparece en pantalla durante la instalación de RaptorXML+XBRL Server. Puede instalar esta versión de LicenseServer junto con RaptorXML+XBRL Server o puede instalar LicenseServer por separado.
- Antes de instalar una versión nueva de LicenseServer, es necesario desinstalar versiones anteriores. El programa de instalación de LicenseServer se encarga de esto automáticamente si detecta versiones más recientes en el sistema.

 © 2018 Altova GmbH

- Las versiones de LicenseServer son compatibles y funcionan con versiones más antiguas de RaptorXML+XBRL Server.
- Si instala una versión nueva de RaptorXML+XBRL Server y la versión de LicenseServer que está instalada es anterior a la que le corresponde, instale la versión más reciente que está siempre disponible en el sitio web de Altova.
- Cuando se desinstala LicenseServer, todos los datos de registro y asignación de licencias almacenados en la versión antigua de LicenseServer se guardan en una base de datos en el equipo servidor. Estos datos se importan de forma automática a la siguiente versión que se instale en el equipo.
- El número de versión de LicenseServer siempre aparece al final de la página de configuración de LicenseServer.

Versión actual: 2.7

Licencia de prueba

Durante el proceso de instalación tendrá la opción de solicitar una licencia de prueba de 30 días para RaptorXML+XBRL Server. Altova le enviará un correo electrónico con la licencia de prueba a la dirección de correo que usted indique en el formulario.

Ubicación de la carpeta de la aplicación

La aplicación se instalará en esta carpeta:

Windows 7, 8 y 10	`C:\Archivos de programa\Altova\`
Versión de 32 bits en sistemas operativos de 64 bits	`C:\Archivos de programa (x86)\Altova\`

2.1.2 Asignación de licencias en Windows

Para poder trabajar con RaptorXML+XBRL Server es necesario asignarle una licencia con Altova LicenseServer. La asignación de licencias es un proceso de dos pasos:

1. El primero consiste en **registrar RaptorXML+XBRL Server** con LicenseServer desde RaptorXML+XBRL Server.
2. El segundo paso consiste en **asignar una licencia** a RaptorXML+XBRL Server desde LicenseServer.

A continuación encontrará información más detallada al respecto.

Iniciar el controlador de servicios ServiceController

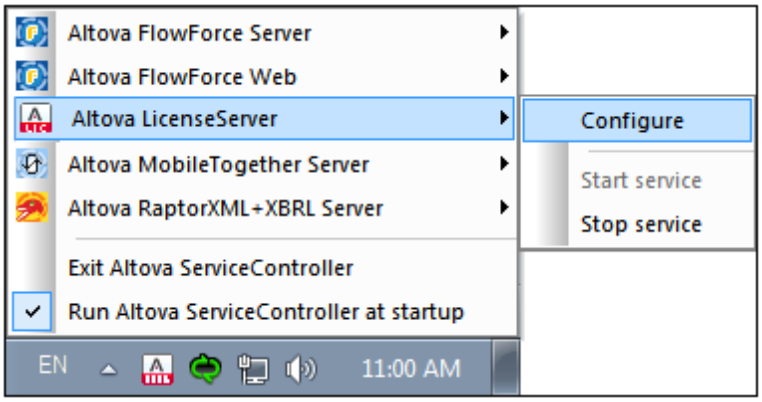
Altova ServiceController se inicia para arrancar Altova LicenseServer y Altova RaptorXML +XBRL Server.

Altova ServiceController (en adelante *ServiceController*) es una práctica aplicación que sirve

para iniciar, detener y configurar los servicios de Altova **en sistemas Windows**.

ServiceController se instala con Altova LicenseServer y con *los productos servidor de Altova que se instalan como servicios* (FlowForce Server, RaptorXML(+XBRL) Server y Mobile Together Server). Se puede iniciar haciendo clic en **Inicio | Altova LicenseServer | Altova ServiceController**. (Este comando también está en las carpetas del menú **Inicio** de *los productos servidor de Altova que se instalan como servicios* (FlowForce Server, RaptorXML(+XBRL) Server y Mobile Together Server).) Una vez iniciado, podrá acceder a ServiceController desde la bandeja del sistema (*imagen siguiente*).

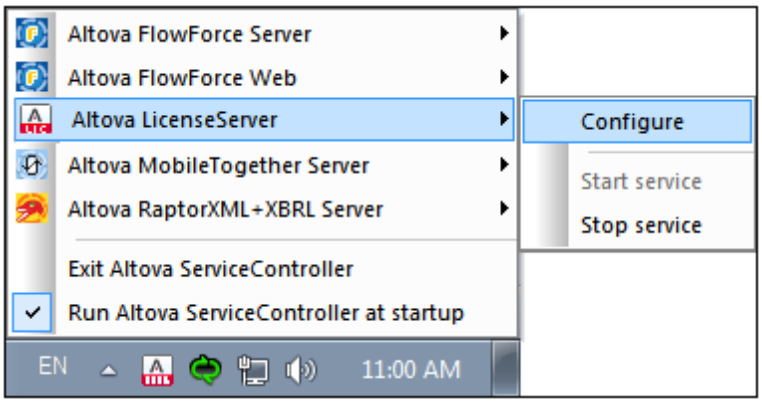
Si quiere que ServiceController se inicie automáticamente nada más iniciar sesión en el sistema, haga clic en el icono de ServiceController de la bandeja del sistema para abrir el menú de opciones de **ServiceController** (*imagen siguiente*) y active la opción **Run Altova ServiceController at Startup** (*Ejecutar Altova ServiceController al inicio*), que de todas maneras es la opción predeterminada. Para cerrar ServiceController haga clic en el icono de ServiceController de la bandeja del sistema y en el menú haga clic en la opción **Exit Altova ServiceController** (Salir de Altova ServiceController).



Iniciar LicenseServer

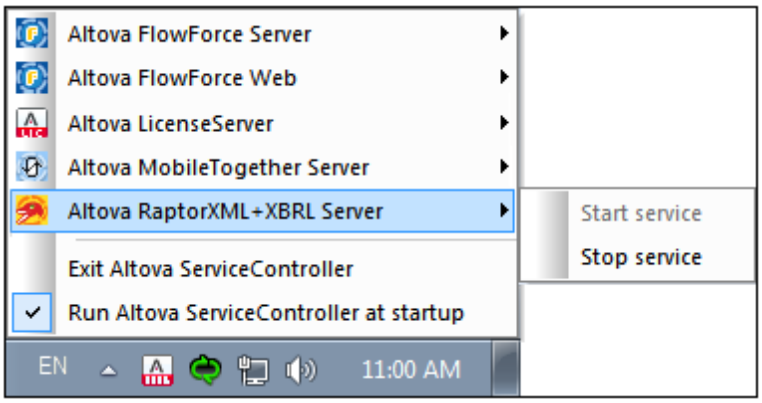
Para iniciar LicenseServer haga clic en el icono de **ServiceController** en la bandeja del sistema, pase el puntero del ratón por encima de la opción **Altova LicenseServer** del menú emergente (*imagen siguiente*) y seleccione el comando **Start service** en el submenú. Si LicenseServer ya está en ejecución, este comando estará deshabilitado.

 © 2018 Altova GmbH



Iniciar RaptorXML+XBRL Server

Para iniciar RaptorXML+XBRL Server haga clic en el icono de **ServiceController** en la bandeja del sistema, pase el puntero del ratón por encima de la opción **Altova RaptorXML +XBRL Server** del menú emergente (*imagen siguiente*) y seleccione el comando **Start service** en el submenú. Si RaptorXML+XBRL Server ya está en ejecución, este comando estará deshabilitado.



Nota: si RaptorXML+XBRL Server tiene una licencia para ejecutar solamente ejecuciones de subprocesamiento simple (esto suele deberse a que su equipo tiene varios núcleos, pero su licencia es para un solo núcleo), entonces solamente podrá usar una instancia de RaptorXML+XBRL Server a la vez: bien como servicio, bien desde la línea de comandos. El motivo es que las licencias para un solo núcleo se asignan automáticamente a la primera instancia que se inicia y que está en ejecución. No podrá iniciar una segunda instancia hasta que la primera deje de ejecutarse.

- Si desea usar RaptorXML+XBRL Server desde la línea de comandos pero el servicio ya está en ejecución, detenga el servicio antes de usar la línea de comandos.
- Si desea iniciar RaptorXML+XBRL Server como servicio, compruebe que no se está ejecutando ninguna acción de la línea de comandos. Si es así no podrá iniciar el servicio.

- Registrar RaptorXML+XBRL Server

  Para registrar RaptorXML+XBRL Server desde la interfaz de la línea de comandos utilice el comando `licenseserver`:

  `RaptorXMLXBRL licenseserver [opciones] NombreServidor-O--Dirección-IP`

  Por ejemplo, si `localhost` es el nombre del servidor donde está instalado LicenseServer:

  `RaptorXMLXBRL licenseserver localhost`

  Si RaptorXML+XBRL Server se instaló durante la instalación de FlowForce Server y registró FlowForce Server con LicenseServer, entonces RaptorXML+XBRL Server también se registró automáticamente. Estos son los pasos que debe seguir: (i) Inicie Altova FlowForce Web como servicio con el controlador de servicios de Altova ServiceController (*véase el punto anterior*); (ii) Introduzca su contraseña para acceder a la página de configuración web; (iii) Seleccione el nombre o la dirección del servidor LicenseServer y haga clic en el botón **Registrarse con LicenseServer**. Para más información consulte el apartado Registrar FlowForce Server.

  Una vez finalizado el proceso de registro, abra la pestaña de gestión de servidores de la página de configuración de LicenseServer para asignar una licencia a RaptorXML+XBRL Server.

- Asignar licencias

  Tras registrarse con LicenseServer, RaptorXML+XBRL Server aparecerá en la lista de la pestaña de gestión de servidores **Server Management** de la página de configuración de LicenseServer. En esta pestaña puede asignar una licencia a RaptorXML+XBRL Server.

  Nota sobre núcleos y licencias
  La asignación de licencias a productos servidor de Altova depende de cuántos núcleos de procesador tiene el equipo donde se ejecuta el producto servidor de Altova. Por ejemplo, un procesador dual tiene dos núcleos, un procesador *quad* tiene cuatro núcleos, un procesador *hexa-core* tiene seis núcleos, y así sucesivamente. El número de núcleos de la licencia asignada a un producto debe ser mayor o igual al número de núcleos disponibles en dicho equipo servidor, ya sea un servidor físico o un equipo virtual.

  Por ejemplo, si un servidor tiene ocho núcleos (un procesador *octa-core*), deberá comprar una licencia para ocho núcleos. También puede combinar varias licencias para alcanzar el número de núcleos necesario. Es decir, puede usar dos licencias para cuatro núcleos para un servidor *octa-core* en lugar de una licencia para ocho núcleos, por ejemplo.

  Si usa un equipo servidor con gran cantidad de núcleos, pero tiene un bajo volumen de procesamiento, también puede crear un equipo virtual que tenga adjudicados menos núcleos y comprar una licencia para ese menor número de núcleos. No obstante, dicha implementación será menos rápida que si utilizara todos los núcleos disponibles en el servidor.

  Nota: cada licencia de los productos servidor de Altova se puede usar de forma simultánea en un equipo como máximo (en el equipo donde está instalado el producto servidor de Altova), incluso si la capacidad de la licencia no está agotada. Por ejemplo, si utiliza una licencia para 10 núcleos para un equipo cliente que tiene 6 núcleos, los 4 núcleos

restantes de la licencia no se pueden usar simultáneamente en otro equipo cliente.

Licencias de MobileTogether Server
Las licencias de MobileTogether Server funciona en base al número de núcleos CPU que tenga el equipo donde se ejecuta MobileTogether Server. Las licencias basadas en el número de núcleos permiten conectar un número ilimitado de dispositivos al servidor. Sin embargo, si marca la casilla *Limit to single thread execution*, entonces solo se podrá conectar a MobileTogether Server un máximo de un dispositivo. Esto puede ser útil en tareas de evaluación y de pruebas a pequeña escala. No obstante, si estando marcada esta casilla, se conecta otro dispositivo a MobileTogether Server, este otro dispositivo se hará con la licencia. El primer dispositivo ya no se podrá conectar y recibirá un mensaje de error a tal efecto.

2.2 Instalación y configuración en Linux

Esta sección explica cómo instalar RaptorXML+XBRL Server y asignarle licencias en sistemas Linux (Debian, Ubuntu, CentOS, RedHat).

Instalación en Linux

- Requisitos del sistema
- Nota sobre el usuario raíz
- Desinstalar versiones antiguas de los productos servidor de Altova
- Descargar el paquete de instalación para Linux
- Instalar RaptorXML+XBRL Server
- Altova LicenseServer
- Versiones de LicenseServer
- Licencia de prueba

Asignación de licencias en Linux

- Nota sobre el usuario raíz
- Iniciar LicenseServer
- Iniciar RaptorXML+XBRL Server
- Registrar RaptorXML+XBRL Server
- Asignar licencias

2.2.1 Instalación en Linux

El proceso de instalación y configuración de RaptorXML+XBRL Server en Linux se describe a continuación.

- Requisitos del sistema
  - *Linux*
    - CentOS 6 o superior
    - RedHat 6 o superior
    - Debian 7 o superior
    - Ubuntu 12.04 o superior

    Las bibliotecas que aparecen a continuación son un requisito previo para la instalación y ejecución de la aplicación. Si los paquetes que aparecen en esta tabla no están en su equipo Linux, ejecute el comando `yum` (o `apt-get` si procede) para instalarlos.

Requisito para	CentOS, RedHat	Debian	Ubuntu
LicenseServer	krb5-libs	libgssapi-krb5-2	libgssapi-krb5-2
RaptorXML+XBRL Server	qt4, krb5-libs, qt-x11	libqtcore4, libqtgui4, libgssapi-krb5-2	libqtcore4, libqtgui4, libgssapi-krb5-2

 © 2018 Altova GmbH

Integración con FlowForce Server

Si está instalando RaptorXML+XBRL Server junto con FlowForce Server, recomendamos instalar primero FlowForce Server. Si no lo hace así, después de instalar RaptorXML+XBRL Server y FlowForce Server, ejecute este comando:

```
cp /opt/Altova/RaptorXMLXBRLServer2018/etc/*.tool /opt/Altova/
FlowForceServer2018/tools
```

Este comando copia el archivo **.tool** del directorio `/etc` de RaptorXML+XBRL Server al directorio `/tools` de FlowForce Server. El archivo **.tool** es un archivo que FlowForce Server necesita porque contiene la ruta de acceso del ejecutable de RaptorXML+XBRL Server. Si instala FlowForce Server antes que RaptorXML+XBRL Server entonces no es necesario ejecutar este comando.

Nota sobre el usuario raíz

Debe tener privilegios de administrador (`root`) para poder instalar RaptorXML+XBRL Server. Por tanto, la instalación debe realizarla el usuario raíz. Si inició sesión como `root`, puede omitir la palabra clave `sudo` de los comandos que aparecen más abajo.

Desinstalar versiones antiguas de los productos servidor de Altova

En la interfaz de la línea de comandos de Linux puede comprobar si ya hay productos servidor de Altova instalados en el equipo. Para ello use este comando:

```
[Debian, Ubuntu]:   dpkg --list | grep Altova
[CentOS, RedHat]:   rpm -qa | grep server
```

Si RaptorXML+XBRL Server no está instalado, continúe con la instalación tal y como se describe más abajo.

Si RaptorXML+XBRL Server ya está instalado y quiere instalar una versión más reciente, antes debe desinstalar la versión previa con este comando:

```
[Debian, Ubuntu]:   sudo dpkg --remove raptorxmlxbrlserver
[CentOS, RedHat]:   sudo rpm -e raptorxmlxbrlserver
```

Si quiere desinstalar una versión previa de Altova LicenseServer, use este comando:

```
[Debian, Ubuntu]:   sudo dpkg --remove licenseserver
[CentOS, RedHat]:   sudo rpm -e licenseserver
```

Descargar el paquete de instalación para Linux

Los paquetes de instalación de RaptorXML+XBRL Server para sistemas Linux se pueden descargar del sitio web de Altova.

Distribución	Extensión del paquete
Debian 7 y superior	`.deb`

Ubuntu12.04 y superior	`.deb`
CentOS 6 y superior	`.rpm`
RedHat 6 y superior	`.rpm`

Tras descargarlo, copie el paquete de instalación en cualquier directorio del sistema Linux. Para ejecutar RaptorXML+XBRL Server es necesario tener instalado Altova LicenseServer, que también se puede descargar del sitio web de Altova.

Instalar RaptorXML+XBRL Server

En una ventana de la Terminal, cambie al directorio donde copió el paquete de instalación para Linux. Por ejemplo, si lo copió en un directorio del usuario llamado `MiAltova` (ubicado en `/home/User` por ejemplo), cambie a ese directorio con esta línea de comandos:

```
cd /home/User/MiAltova
```

Instale RaptorXML+XBRL Server con este comando:

```
[Debian]:   sudo dpkg --install raptorxmlxbrlserver-2018-debian.deb
[Ubuntu]:   sudo dpkg --install raptorxmlxbrlserver-2018-ubuntu.deb
[CentOS]:   sudo rpm -ivh raptorxmlxbrlserver-2018-1.x86_64.rpm
[RedHat]:   sudo rpm -ivh raptorxmlxbrlserver-2018-1.x86_64.rpm
```

La aplicación RaptorXML+XBRL Server se instala en este directorio:

```
/opt/Altova/RaptorXMLXBRLServer2018
```

Altova LicenseServer

Para poder ejecutar los productos servidores de Altova, incluido RaptorXML+XBRL Server, es necesario asignarles una licencia con un servidor Altova LicenseServer de la red.

En los sistemas Linux es necesario instalar Altova LicenseServer por separado. Por tanto, descargue Altova LicenseServer del sitio web de Altova y copie el paquete de instalación en cualquier directorio. Siga las instrucciones anteriores para instalar LicenseServer (*ver apartado anterior*).

```
[Debian]:   sudo dpkg --install licenseserver-2.7-debian.deb
[Ubuntu]:   sudo dpkg --install licenseserver-2.7-ubuntu.deb
[CentOS]:   sudo rpm -ivh licenseserver-2.7-1.x86_64.rpm
[RedHat]:   sudo rpm -ivh licenseserver-2.7-1.x86_64.rpm
```

La aplicación LicenseServer se instala en este directorio:

```
/opt/Altova/LicenseServer
```

Consulte el apartado siguiente Asignación de licencias en Linux para obtener información sobre cómo registrar RaptorXML+XBRL Server con Altova LicenseServer y asignarle licencias. Consulte también la documentación del LicenseServer para información más

 © 2018 Altova GmbH

detallada.

- Versiones de LicenseServer
  - Los productos servidor de Altova deben tener una licencia con la versión de LicenseServer correspondiente a la versión de RaptorXML+XBRL Server instalada o con una versión posterior de LicenseServer.
  - La versión de LicenseServer correspondiente a la versión de RaptorXML+XBRL Server aparece en pantalla durante la instalación de RaptorXML+XBRL Server. Puede instalar esta versión de LicenseServer junto con RaptorXML+XBRL Server o puede instalar LicenseServer por separado.
  - Antes de instalar una versión nueva de LicenseServer, es necesario desinstalar versiones anteriores. El programa de instalación de LicenseServer se encarga de esto automáticamente si detecta versiones más recientes en el sistema.
  - Las versiones de LicenseServer son compatibles y funcionan con versiones más antiguas de RaptorXML+XBRL Server.
  - Si instala una versión nueva de RaptorXML+XBRL Server y la versión de LicenseServer que está instalada es anterior a la que le corresponde, instale la versión más reciente que está siempre disponible en el sitio web de Altova.
  - Cuando se desinstala LicenseServer, todos los datos de registro y asignación de licencias almacenados en la versión antigua de LicenseServer se guardan en una base de datos en el equipo servidor. Estos datos se importan de forma automática a la siguiente versión que se instale en el equipo.
  - El número de versión de LicenseServer siempre aparece al final de la página de configuración de LicenseServer.

    Versión actual: 2.7

2.2.2 Asignación de licencias en Linux

Para poder trabajar con RaptorXML+XBRL Server es necesario asignarle una licencia con Altova LicenseServer. La asignación de licencias es un proceso de dos pasos:

1. El primero consiste en **registrar RaptorXML+XBRL Server** con LicenseServer desde RaptorXML+XBRL Server.
2. El segundo paso consiste en **asignar una licencia** a RaptorXML+XBRL Server desde LicenseServer.

A continuación encontrará información más detallada al respecto.

- Nota sobre el usuario raíz

  Debe tener privilegios de administrador (`root`) para poder instalar RaptorXML+XBRL Server. Por tanto, la instalación debe realizarla el usuario raíz. Si inició sesión como `root`, puede omitir la palabra clave `sudo` de los comandos que aparecen más abajo.

Iniciar LicenseServer

Para poder registrar RaptorXML+XBRL Server con LicenseServer y asignarle una licencia, LicenseServer debe estar en ejecución como servicio. Inicie LicenseServer como servicio con este comando:

[< Debian 8]	**sudo /etc/init.d/licenseserver start**
[≥ Debian 8]	**sudo systemctl start licenseserver**
[< CentOS 7]	**sudo initctl start licenseserver**
[≥ CentOS 7]	**sudo systemctl start licenseserver**
[< Ubuntu 15]	**sudo initctl start licenseserver**
[≥ Ubuntu 15]	**sudo systemctl start licenseserver**
[RedHat]	**sudo initctl start licenseserver**

Si por cualquier motivo necesita detener LicenseServer, use el mismo comando pero sustituya **stop** por **start**. Por ejemplo:

```
sudo /etc/init.d/licenseserver stop
```

Iniciar RaptorXML+XBRL Server

Inicie RaptorXML+XBRL Server como demonio con este comando:

[< Debian 8]	**sudo /etc/init.d/raptorxmlxbrlserver start**
[≥ Debian 8]	**sudo systemctl start raptorxmlxbrlserver**
[< CentOS 7]	**sudo initctl start raptorxmlxbrlserver**
[≥ CentOS 7]	**sudo systemctl start raptorxmlxbrlserver**
[< Ubuntu 15]	**sudo initctl start raptorxmlxbrlserver**
[≥ Ubuntu 15]	**sudo systemctl start raptorxmlxbrlserver**
[RedHat]	**sudo initctl start raptorxmlxbrlserver**

Registrar RaptorXML+XBRL Server

Para registrar RaptorXML+XBRL Server desde la interfaz de la línea de comandos utilice el comando `licenseserver`:

```
sudo /opt/Altova/RaptorXMLXBRLServer2018/bin/raptorxmlxbrl
licenseserver [opciones] NombreServidor-O-Dirección-IP
```

Por ejemplo, si el nombre del servidor donde está instalado LicenseServer es `localhost`:

```
sudo /opt/Altova/RaptorXMLXBRLServer2018/bin/raptorxmlxbrl
licenseserver localhost
```

En el comando anterior `localhost` es el nombre del servidor donde está instalado LicenseServer. Observe también la ubicación del ejecutable de RaptorXML+XBRL Server:

```
/opt/Altova/RaptorXMLXBRLServer2018/bin/
```

 © 2018 Altova GmbH

Una vez completado el registro, abra la pestaña **Server Management** de la página de configuración de LicenseServer para asignar una licencia a RaptorXML+XBRL Server.

Asignar licencias

Tras registrarse con LicenseServer, RaptorXML+XBRL Server aparecerá en la lista de la pestaña de gestión de servidores **Server Management** de la página de configuración de LicenseServer. En esta pestaña puede asignar una licencia a RaptorXML+XBRL Server.

Nota sobre núcleos y licencias
La asignación de licencias a productos servidor de Altova depende de cuántos núcleos de procesador tiene el equipo donde se ejecuta el producto servidor de Altova. Por ejemplo, un procesador dual tiene dos núcleos, un procesador *quad* tiene cuatro núcleos, un procesador *hexa-core* tiene seis núcleos, y así sucesivamente. El número de núcleos de la licencia asignada a un producto debe ser mayor o igual al número de núcleos disponibles en dicho equipo servidor, ya sea un servidor físico o un equipo virtual.

Por ejemplo, si un servidor tiene ocho núcleos (un procesador *octa-core*), deberá comprar una licencia para ocho núcleos. También puede combinar varias licencias para alcanzar el número de núcleos necesario. Es decir, puede usar dos licencias para cuatro núcleos para un servidor *octa-core* en lugar de una licencia para ocho núcleos, por ejemplo.

Si usa un equipo servidor con gran cantidad de núcleos, pero tiene un bajo volumen de procesamiento, también puede crear un equipo virtual que tenga adjudicados menos núcleos y comprar una licencia para ese menor número de núcleos. No obstante, dicha implementación será menos rápida que si utilizara todos los núcleos disponibles en el servidor.

Nota: cada licencia de los productos servidor de Altova se puede usar de forma simultánea en un equipo como máximo (en el equipo donde está instalado el producto servidor de Altova), incluso si la capacidad de la licencia no está agotada. Por ejemplo, si utiliza una licencia para 10 núcleos para un equipo cliente que tiene 6 núcleos, los 4 núcleos restantes de la licencia no se pueden usar simultáneamente en otro equipo cliente.

Licencias de MobileTogether Server
Las licencias de MobileTogether Server funciona en base al número de núcleos CPU que tenga el equipo donde se ejecuta MobileTogether Server. Las licencias basadas en el número de núcleos permiten conectar un número ilimitado de dispositivos al servidor. Sin embargo, si marca la casilla *Limit to single thread execution*, entonces solo se podrá conectar a MobileTogether Server un máximo de un dispositivo. Esto puede ser útil en tareas de evaluación y de pruebas a pequeña escala. No obstante, si estando marcada esta casilla, se conecta otro dispositivo a MobileTogether Server, este otro dispositivo se hará con la licencia. El primer dispositivo ya no se podrá conectar y recibirá un mensaje de error a tal efecto.

2.2.3 Notas sobre configuración del entorno

Carpetas

A continuación enumeramos carpetas importantes de su sistema RaptorXML+XBRL Server.

Conexiones JDBC

Cuando trabaje con este tipo de conexiones debe tener en cuenta estos aspectos:

- Debe tener instalado Java Runtime Environment o el kit de desarrollo de software.
- Debe tener instalados controladores JDBC para la base de datos de destino.
- Debe establecer correctamente estas variables de entorno:
  - `CLASSPATH`: para encontrar los archivos jar.
  - `PATH`: para buscar el entorno JRE, aunque a veces no es necesaria, dependiendo de la instalación.
  - `JAVA_HOME`: a veces no es necesaria, dependiendo de la instalación.

Nota
En servidores Linux solamente son compatibles las conexiones de base de datos de tipo JDBC.

 © 2018 Altova GmbH

2.3 Instalación y configuración en macOS

Esta sección explica cómo instalar RaptorXML+XBRL Server y asignarle licencias en sistemas macOS.

Instalación en macOS

- Requisitos del sistema
- Nota sobre el usuario raíz
- Desinstalar versiones previas de productos servidor de Altova
- Descargar el paquete de instalación para macOS
- Instalar RaptorXML+XBRL Server
- Altova LicenseServer
- Versiones de LicenseServer
- Licencia de prueba

Asignación de licencias en macOS

- Nota sobre el usuario raíz
- Iniciar LicenseServer
- Iniciar RaptorXML+XBRL Server
- Registrar RaptorXML+XBRL Server
- Asignar licencias

2.3.1 Instalación en macOS

El proceso de instalación y configuración de RaptorXML+XBRL Server en macOS se describe a continuación.

- Requisitos del sistema
  - *(Mac) OS X, macOS*

    OS X 10.10 o superior

- Integración con FlowForce Server

  Si instala RaptorXML+XBRL Server junto con FlowForce Server, recomendamos instalar primero FlowForce Server. Si no lo hace así, después de instalar RaptorXML+XBRL Server y FlowForce Server, ejecute este comando:

  ```
  cp /usr/local/Altova/RaptorXMLXBRLServer2018/etc/*.tool /usr/local/Altova/
  FlowForceServer2018/tools
  ```

  Este comando copia el archivo **.tool** del directorio `/etc` de RaptorXML+XBRL Server al directorio `/tools` de FlowForce Server. El archivo **.tool** es un archivo que FlowForce Server necesita porque contiene la ruta de acceso del ejecutable de RaptorXML+XBRL Server. Si

instala FlowForce Server antes que RaptorXML+XBRL Server entonces no es necesario ejecutar este comando.

Nota sobre el usuario raíz

Debe tener privilegios de administrador (`root`) para poder instalar RaptorXML+XBRL Server. Por tanto, la instalación debe realizarla el usuario raíz. Si inició sesión como `root`, puede omitir la palabra clave `sudo` de los comandos que aparecen más abajo.

Desinstalar versiones antiguas de los productos servidor de Altova

Antes de desinstalar RaptorXML+XBRL Server es necesario detener el servicio con este comando:

```
sudo launchctl unload /Library/LaunchDaemons/
com.altova.RaptorXMLXBRLServer2018.plist
```

Para comprobar si el servicio se detuvo correctamente, abra la terminal del Monitor de actividad y compruebe que RaptorXML+XBRL Server no está en la lista. En la terminal de Aplicaciones haga clic con el botón derecho en el icono de RaptorXML+XBRL Server y seleccione **Mover a la papelera**. La aplicación se envía a la papelera pero debe quitar la aplicación de la carpeta `usr`. Para ello puede utilizar este comando:

```
sudo rm -rf /usr/local/Altova/RaptorXMLXBRLServer2018/
```

Si necesita desinstalar una versión antigua de Altova LicenseServer, antes debe detener el servicio con este comando:

```
sudo launchctl unload /Library/LaunchDaemons/
com.altova.LicenseServer.plist
```

Para comprobar si el servicio se detuvo correctamente, abra la terminal del Monitor de actividad y confirme que LicenseServer no está en la lista. Después desinstale LicenseServer siguiendo las instrucciones dadas más arriba para RaptorXML+XBRL Server.

Descargar el archivo de imagen de disco

Descargue el archivo de imagen de disco (`.dmg`) del sitio web de Altova (https://www.altova.com/es/download.html).

Instalar RaptorXML+XBRL Server

Haga clic en el archivo de imagen de disco (`.dmg`) para abrirlo. El programa de instalación de RaptorXML+XBRL Server aparece como unidad virtual en el equipo. En esta unidad virtual nueva haga doble clic en el paquete de instalación (`.pkg`). Siga las instrucciones que aparecen en pantalla y acepte el contrato de licencia. Para expulsar la unidad cuando termine la instalación, haga clic con el botón derecho en la unidad y seleccione **Expulsar**.

El paquete de RaptorXML+XBRL Server se instalará en esta carpeta:
`/usr/local/Altova/RaptorXMLXBRLServer2018` (archivos binarios de la aplicación)

 © 2018 Altova GmbH

/var/Altova/RaptorXMLXBRLServer (archivos de datos como bases de datos y registros)

El demonio de RaptorXML+XBRL Server se inicia automáticamente después de la instalación y de que se reinicie el equipo. Puede iniciar RaptorXML+XBRL Server como demonio con este comando:

```
sudo launchctl load /Library/LaunchDaemons/
com.altova.RaptorXMLXBRLServer2018.plist
```

Altova LicenseServer

Para poder ejecutar los productos servidor de Altova, incluido RaptorXML+XBRL Server, es necesario asignarles una licencia desde un servidor Altova LicenseServer que esté instalado en la red.

El paquete de instalación de Altova LicenseServer está disponible en la unidad virtual creado en el paso anterior. Para instalar Altova LicenseServer haga doble clic en el paquete de instalación que está en la unidad virtual y siga las instrucciones que aparecen en pantalla. También debe aceptar el contrato de licencia para poder continuar con la instalación.

Altova LicenseServer también puede descargarse desde el sitio web de Altova (https://www.altova.com/es/download.html) e instalarse por separado.

El paquete de LicenseServer se instalará en esta carpeta:

/usr/local/Altova/LicenseServer

Para obtener más información sobre cómo registrar RaptorXML+XBRL Server con Altova LicenseServer y asignarle una licencia, consulte el apartado Asignación de licencias en macOS.

Versiones de LicenseServer

- Los productos servidor de Altova deben tener una licencia con la versión de LicenseServer correspondiente a la versión de RaptorXML+XBRL Server instalada o con una versión posterior de LicenseServer.
- La versión de LicenseServer correspondiente a la versión de RaptorXML+XBRL Server aparece en pantalla durante la instalación de RaptorXML+XBRL Server. Puede instalar esta versión de LicenseServer junto con RaptorXML+XBRL Server o puede instalar LicenseServer por separado.
- Antes de instalar una versión nueva de LicenseServer, es necesario desinstalar versiones anteriores. El programa de instalación de LicenseServer se encarga de esto automáticamente si detecta versiones más recientes en el sistema.
- Las versiones de LicenseServer son compatibles y funcionan con versiones más antiguas de RaptorXML+XBRL Server.
- Si instala una versión nueva de RaptorXML+XBRL Server y la versión de LicenseServer que está instalada es anterior a la que le corresponde, instale la versión más reciente que está siempre disponible en el sitio web de Altova.
- Cuando se desinstala LicenseServer, todos los datos de registro y asignación de licencias almacenados en la versión antigua de LicenseServer se guardan en una base de datos en el equipo servidor. Estos datos se importan de forma automática a la siguiente versión que se instale en el equipo.

- El número de versión de LicenseServer siempre aparece al final de la página de configuración de LicenseServer.

  Versión actual: 2.7

2.3.2 Asignación de licencias en macOS

Para poder trabajar con RaptorXML+XBRL Server es necesario asignarle una licencia con Altova LicenseServer. La asignación de licencias es un proceso de dos pasos:

1. El primero consiste en **registrar RaptorXML+XBRL Server** con LicenseServer desde RaptorXML+XBRL Server.
2. El segundo paso consiste en **asignar una licencia** a RaptorXML+XBRL Server desde LicenseServer.

A continuación encontrará información más detallada al respecto.

Nota sobre el usuario raíz

Debe tener privilegios de administrador (`root`) para poder instalar RaptorXML+XBRL Server. Por tanto, la instalación debe realizarla el usuario raíz. Si inició sesión como `root`, puede omitir la palabra clave `sudo` de los comandos que aparecen más abajo.

Iniciar LicenseServer

Para registrar y asignar una licencia correctamente a RaptorXML+XBRL Server, LicenseServer debe estar en ejecución como demonio. Inicie LicenseServer como demonio con este comando:

```
sudo launchctl load /Library/LaunchDaemons/com.altova.LicenseServer.plist
```

Si por cualquier motivo necesita detener LicenseServer, use el mismo comando pero sustituya **`load`** por **`unload`**. Por ejemplo:

```
sudo launchctl unload /Library/LaunchDaemons/
com.altova.LicenseServer.plist
```

Iniciar RaptorXML+XBRL Server

El demonio de RaptorXML+XBRL Server se inicia automáticamente tras instalarlo y reiniciar el equipo. Use este comando para iniciar RaptorXML+XBRL Server como demonio:

```
sudo launchctl load /Library/LaunchDaemons/
com.altova.RaptorXMLXBRLServer2018.plist
```

Si necesita detener RaptorXML+XBRL Server por cualquier motivo, use este comando:

```
sudo launchctl unload /Library/LaunchDaemons/
```

 © 2018 Altova GmbH

com.altova.RaptorXMLXBRLServer2018.plist

- Registrar RaptorXML+XBRL Server

Para registrar RaptorXML+XBRL Server desde la interfaz de la línea de comandos utilice el comando `licenseserver`:

```
sudo /usr/local/Altova/RaptorXMLXBRLServer2018/bin/RaptorXMLXBRL
licenseserver [opciones] NombreServidor-O-Dirección-IP
```

Por ejemplo, si el nombre del servidor donde está instalado LicenseServer es `localhost`:

```
sudo /usr/local/Altova/RaptorXMLXBRLServer2018/bin/RaptorXMLXBRL
licenseserver localhost
```

En el comando anterior `localhost` es el nombre del servidor donde está instalado LicenseServer. Observe también la ubicación del ejecutable de RaptorXML+XBRL Server:

```
/usr/local/Altova/RaptorXMLXBRLServer2018/bin/
```

Una vez completado el registro, abra la pestaña **Server Management** de la página de configuración de LicenseServer para asignar una licencia a RaptorXML+XBRL Server..

- Asignar licencias

Tras registrarse con LicenseServer, RaptorXML+XBRL Server aparecerá en la lista de la pestaña de gestión de servidores **Server Management** de la página de configuración de LicenseServer. En esta pestaña puede asignar una licencia a RaptorXML+XBRL Server.

Nota sobre núcleos y licencias
La asignación de licencias a productos servidor de Altova depende de cuántos núcleos de procesador tiene el equipo donde se ejecuta el producto servidor de Altova. Por ejemplo, un procesador dual tiene dos núcleos, un procesador *quad* tiene cuatro núcleos, un procesador *hexa-core* tiene seis núcleos, y así sucesivamente. El número de núcleos de la licencia asignada a un producto debe ser mayor o igual al número de núcleos disponibles en dicho equipo servidor, ya sea un servidor físico o un equipo virtual.

Por ejemplo, si un servidor tiene ocho núcleos (un procesador *octa-core*), deberá comprar una licencia para ocho núcleos. También puede combinar varias licencias para alcanzar el número de núcleos necesario. Es decir, puede usar dos licencias para cuatro núcleos para un servidor *octa-core* en lugar de una licencia para ocho núcleos, por ejemplo.

Si usa un equipo servidor con gran cantidad de núcleos, pero tiene un bajo volumen de procesamiento, también puede crear un equipo virtual que tenga adjudicados menos núcleos y comprar una licencia para ese menor número de núcleos. No obstante, dicha implementación será menos rápida que si utilizara todos los núcleos disponibles en el servidor.

Nota: cada licencia de los productos servidor de Altova se puede usar de forma simultánea en un equipo como máximo (en el equipo donde está instalado el producto servidor de Altova), incluso si la capacidad de la licencia no está agotada. Por ejemplo, si utiliza una licencia para 10 núcleos para un equipo cliente que tiene 6 núcleos, los 4 núcleos restantes de la licencia no se pueden usar simultáneamente en otro equipo cliente.

Licencias de MobileTogether Server

Las licencias de MobileTogether Server funciona en base al número de núcleos CPU que tenga el equipo donde se ejecuta MobileTogether Server. Las licencias basadas en el número de núcleos permiten conectar un número ilimitado de dispositivos al servidor. Sin embargo, si marca la casilla *Limit to single thread execution*, entonces solo se podrá conectar a MobileTogether Server un máximo de un dispositivo. Esto puede ser útil en tareas de evaluación y de pruebas a pequeña escala. No obstante, si estando marcada esta casilla, se conecta otro dispositivo a MobileTogether Server, este otro dispositivo se hará con la licencia. El primer dispositivo ya no se podrá conectar y recibirá un mensaje de error a tal efecto.

2.3.3 Notas sobre configuración del entorno

Carpetas

A continuación enumeramos carpetas importantes de su sistema RaptorXML+XBRL Server.

Conexiones JDBC

Cuando trabaje con este tipo de conexiones debe tener en cuenta estos aspectos:

- Debe tener instalado Java Runtime Environment o el kit de desarrollo de software.
- Debe tener instalados controladores JDBC para la base de datos de destino.
- Debe establecer correctamente estas variables de entorno:
  - **CLASSPATH**: para encontrar los archivos jar.
  - **PATH**: para buscar el entorno JRE, aunque a veces no es necesaria, dependiendo de la instalación.
  - **JAVA_HOME**: a veces no es necesaria, dependiendo de la instalación.

Nota
En servidores macOS solamente son compatibles las conexiones de base de datos de tipo JDBC.

 © 2018 Altova GmbH

2.4 Catálogos XML

El mecanismo de catalogación XML permite recuperar archivos de carpetas locales, lo cual incrementa la velocidad global de procesamiento y mejora la portabilidad de los documentos (porque solo se tienen que cambiar los identificadores URI de los archivos de catálogo). Para más información consulte el apartado Cómo funcionan los catálogos.

Las herramientas XML de Altova usan un mecanismo de catalogación para acceder rápidamente a los archivos más utilizados, como esquemas XML y DTD. El usuario puede personalizar y ampliar este mecanismo de catalogación, que se describe en el apartado Mecanismo de catalogación XML de Altova. En el apartado Variables para ubicaciones del sistema se enumeran variables Windows para las ubicaciones más corrientes. Estas variables se pueden usar en los archivos de catálogo para encontrar las carpetas más utilizadas.

Esta sección se divide en varios apartados:

- Cómo funcionan los catálogos
- Mecanismo de catalogación XML de Altova
- Variables para ubicaciones de sistemas Windows

Para más información consulte la especificación XML Catalogs.

2.4.1 Cómo funcionan los catálogos

Temas de este apartado:

- Asignar identificadores públicos y de sistema a direcciones URL locales
- Asignar rutas de acceso, direcciones URL web y nombres a direcciones URL locales

La función de los catálogos es redireccionar llamadas a recursos remotos a una URL local. Esto se consigue mediante asignaciones en el archivo de catálogo entre identificadores públicos o de sistemas, identificadores URI o partes de identificadores y la URL local correspondiente.

Asignaciones entre identificadores públicos y de sistema y URL locales

Durante la lectura de la declaración DOCTYPE de una DTD en un archivo XML, el identificador público o de sistema de la declaración encuentra el recurso necesario. Si el identificador selecciona un recurso remoto o si el identificador no es un localizador, entonces se puede asignar a un recurso local mediante una entrada en el catálogo.

Por ejemplo, este archivo SVG:

```
<?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>
  ...
</svg>
```

Su identificador público es: **-//W3C//DTD SVG 1.1//EN**
Su identificador de sistema es: http://www.w3.org/Graphics/SVG/1.1/DTD/svg11.dtd

En el catálogo se puede añadir una entrada para asignar el identificador público a una URL local. Por ejemplo:

```
<public publicId="-//W3C//DTD SVG 1.1//EN" uri="schemas/svg/svg11.dtd"/>
```

O una entrada para asignar el identificador de sistema a una URL local. Por ejemplo:

```
<system systemId="http://www.w3.org/Graphics/SVG/1.1/DTD/svg11.dtd"
uri="schemas/svg/svg11.dtd"/>
```

Si se usa el identificador público o de sistema que aparece en el catálogo, entonces se usa la URL a la que está asignado. Las rutas relativas se resuelven con referencia a un atributo `xml:base` en el elemento de redirección del catálogo. La URL base de reserva es la URL del archivo de catálogo. Por el contrario, si no se usa el identificador público o de sistema que aparece en el catálogo, entonces se usará la URL del documento XML (en nuestro ejemplo sería la URL `http://www.w3.org/Graphics/SVG/1.1/DTD/svg11.dtd`).

Asignaciones entre rutas de archivo relativas/absolutas, direcciones URL o nombres y URL locales

El elemento **`uri`** se puede usar para asignar una ruta de archivo relativa/absoluta, una dirección URL o un nombre a una URL local. Por ejemplo:

- `<uri name="doc.xslt" uri="C:\Docs\doc.xslt"/>`
- `<uri name="U:\Docs\2013\doc.xslt" uri="C:\Docs\doc.xslt"/>`
- `<uri name="http://www.altova.com/schemas/doc.xslt" uri="C:\Docs\doc.xslt"/>`
- `<uri name="foo" uri="C:\Docs\doc.xslt"/>`

Cuando se encuentra el valor de `name`, este se asigna al recurso especificado en el atributo `uri`. Con un catálogo distinto, el mismo nombre se podría asignar a un recurso diferente. Por ejemplo:

```
xsi:schemaLocation="http://www.altova.com/schemas/orgchart OrgChart.xsd"
```

Por lo general, la parte URI del valor del atributo (en negrita) 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 la parte URI del valor del atributo `xsi:schemaLocation` (en vez de `Orgchart.xsd`). El esquema está ubicado dentro del catálogo gracias a la parte de espacio de nombres del valor del atributo `xsi:schemaLocation`. En el ejemplo anterior la parte de espacio de nombres es `http://www.altova.com/schemas/orgchart`.

En el catálogo la entrada siguiente encontraría el esquema por la parte de espacio de nombres.

```
<uri name="http://www.altova.com/schemas/orgchart" uri="C:\MySchemas
```

 © 2018 Altova GmbH

```
\OrgChart.xsd"/>
```

Para más información sobre estos elementos consulte la especificación XML Catalogs.

2.4.2 Mecanismo de catalogación XML de Altova

Temas de este apartado:

- El archivo de catálogo raíz `RootCatalog.xml` contiene los archivos de catálogo en los que busca RaptorXML.
- Los archivos catálogo de extensión `CoreCatalog.xml`, `CustomCatalog.xml` y `Catalog.xml`.
- Subconjunto de catálogos compatible.

RootCatalog.xml

RaptorXML busca por defecto en el archivo `RootCatalog.xml` (*ver más abajo*) la lista de archivos de catálogo que debe usar. El catálogo raíz `RootCatalog.xml` está en esta carpeta:

```
<CarpetaArchivosPrograma>\Altova\RaptorXMLXBRLServer2018\etc
```

Para usar otro archivo como catálogo raíz, utilice la opción `--catalog` de la línea de comandos, el método `setCatalog` de la interfaz Java o el método `Catalog` de la interfaz COM.

```
<?xml version="1.0" encoding="UTF-8"?>
<catalog xmlns="urn:oasis:names:tc:entity:xmlns:xml:catalog"
         xmlns:spy="http://www.altova.com/catalog_ext"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="urn:oasis:names:tc:entity:xmlns:xml:catalog
Catalog.xsd">

  <nextCatalog catalog="%PersonalFolder%/Altova/%AppAndVersionName%/
CustomCatalog.xml"/>
  <nextCatalog catalog="CoreCatalog.xml"/>

  <!-- Incluir todos los catálogos situados en la carpeta Schemas del primer
nivel de directorios -->
  <nextCatalog spy:recurseFrom="%AltovaCommonFolder%/Schemas"
catalog="catalog.xml" spy:depth="1"/>

  <!-- Incluir todos los catálogos situados en la carpeta XBRL del primer nivel
de directorios -->
  <nextCatalog spy:recurseFrom="%AltovaCommonFolder%/XBRL" catalog="catalog.xml"
spy:depth="1"/>
</catalog>
```

El resto de archivos de catálogo se enumeran dentro de un elemento `nextCatalog` y puede añadir tantos archivos como quiera. RaptorXML busca en todos los archivos de catálogo y resuelve las asignaciones que hay en ellos.

En el fragmento de código anterior puede observar una referencia directa a dos catálogos: `CoreCatalog.xml` y `CustomCatalog.xml`. Además se hace referencia a los catálogos llamados `catalog.xml` que están en el primer nivel de subcarpetas de las carpetas `Schemas` y `XBRL`. (El valor de la variable `%AltovaCommonFolder%` se explica en el apartado Variables para ubicaciones de sistema.)

Los archivos de catálogo de `Altova Common Folder` asignan los identificadores públicos y de sistema predefinidos de los esquemas más utilizados (como XML Schema y XHTML) a identificadores URI que apuntan a las copias locales de los esquemas correspondientes. Estos esquemas se instalan en la carpeta `Altova Common Folder` durante la instalación de RaptorXML.

CoreCatalog.xml, CustomCatalog.xml y Catalog.xml

Los archivos de catálogo `CoreCatalog.xml` y `CustomCatalog.xml` se enumeran en `RootCatalog.xml`:

- `CoreCatalog.xml` contiene ciertas asignaciones propias de Altova necesarias para encontrar esquemas en la carpeta `Altova Common Folder`.
- `CustomCatalog.xml` es un archivo esqueleto donde puede crear sus propias asignaciones. En `CustomCatalog.xml` puede crear asignaciones para cualquier esquema que necesite y que no esté en los archivos de catálogo de la carpeta `Altova Common Folder`. Para ello debe utilizar elementos compatibles del mecanismo de catalogación OASIS (*ver más abajo*).
- Hay varios archivos `Catalog.xml` dentro de las carpetas de esquemas o taxonomías XBRL de la carpeta `Altova Common Folder` y cada uno de estos archivos asigna identificadores públicos/de sistema a identificadores URI que apuntan a copias locales de los esquemas correspondientes.

Tanto `CoreCatalog.xml` como `CustomCatalog.xml` están en la carpeta `<CarpetaArchivosPrograma>\Altova\RaptorXMLXBRLServer2018\etc`. Cada archivo `catalog.xml` está en una carpeta de esquema y estas carpetas de esquema están a su vez dentro de las carpetas `%AltovaCommonFolder%\Schemas` y `%AltovaCommonFolder%\XBRL`.

Subconjunto de catálogos compatible

Cuando cree entradas en un archivo de catálogo utilizado por RaptorXML, solamente debería usar los elementos de la especificación OASIS que aparecen a continuación. Consulte la especificación XML Catalogs para obtener más información.

- `<public publicId="IDPúblico del Recurso" uri="URL del archivo local"/>`
- `<system systemId="IDdeSistema del Recurso" uri="URL del archivo local"/>`
- `<uri name="nombreArchivo" uri="URL del archivo identificado por el nombre de archivo"/>`
- `<rewriteURI uriStartString="InicioDeCadena del URI que se debe rescribir" rewritePrefix="Cadena que debe sustituir a InicioDeCadena"/>`
- `<rewriteSystem systemIdStartString="InicioDeCadena del IDdeSistema" rewritePrefix="Cadena de sustitución para encontrar el recurso localmente"/>`

 © 2018 Altova GmbH

Cuando no exista un identificador público, puede asignar el identificador de sistema directamente a una URL con ayuda del elemento `system`. También puede asignar un URI a otro URI usando el elemento `uri`. Los elementos `rewriteURI` y `rewriteSystem` sirven para rescribir la parte inicial de un URI o de un identificador de sistema, respectivamente. Esto permite reemplazar el inicio de una ruta de archivo y, por tanto, apuntar a otro directorio.

Nota: todos los elementos pueden tomar el atributo `xml:base`, que se usa para especificar el URI base del elemento. Si no hay ningún elemento con `xml:base`, el URI base será el URI del archivo de catálogo.

Para obtener más información sobre estos elementos consulte la especificación XML Catalogs.

2.4.3 Variables para ubicaciones de sistemas Windows

En los archivos de catálogo puede usar variables de entorno Shell para señalar la ruta de acceso a varias ubicaciones del sistema Windows. Estas son las variables de entorno Shell compatibles:

`%AltovaCommonFolder%`	`C:\Archivos de programa\Altova\Common2018`
`%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 del directorio de archivos que contiene datos del programa de todos los usuarios.
`%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.
`%CommonFilesFolder%`	Ruta de acceso completa de la carpeta `Common files` del usuario actual.
`%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`.

 © 2018 Altova GmbH

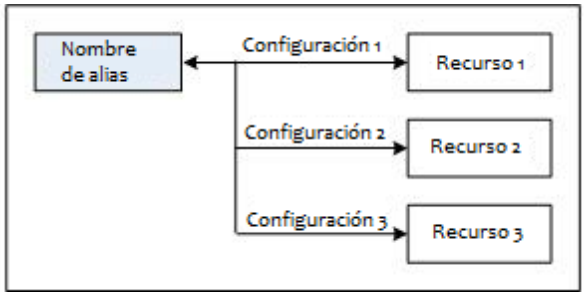
2.5 Recursos globales

Temas de este apartado:

- Recursos globales: ¿qué son?
- Recursos globales: ¿cómo se usan?

¿Qué son los recursos globales?

Un archivo de recurso global de Altova asigna un alias a varios recursos mediante configuraciones diferentes, tal y como muestra el diagrama que aparece a continuación. La idea es poder cambiar de alias para acceder a recursos distintos, dependiendo de la configuración elegida.



Los recursos globales se definen desde las herramientas de Altova (como Altova XMLSpy, por ejemplo) y se guardan en un archivo XML de recursos globales. RaptorXML puede usar estos recursos globales como datos de entrada. Para ello necesita el nombre y la ubicación del archivo de recursos globales, así como el alias y la configuración que debe usar.

La ventaja de usar recursos globales es que puede cambiar de recurso con solo cambiar el nombre de la configuración. En RaptorXML, esto significa que al usar un valor diferente de la opción `--globalresourcesconfig | --gc`, se puede usar un recurso global distinto (*ver ejemplo que aparece más abajo*).

¿Cómo se utilizan los recursos globales con RaptorXML?

Para especificar el uso de un recurso global como entrada para un comando de RaptorXML es obligatorio usar estos parámetros en la interfaz de la línea de comandos:

- El archivo XML de recursos globales (opción `--globalresourcesfile | --gr`)
- La configuración necesaria (opción `--globalresourcesconfig | --gc`)
- El alias, que se puede especificar directamente en la ILC cuando sea necesario un nombre de archivo. También puede estar dentro del archivo XML en el que RaptorXML busca un nombre de archivo (como en un atributo `xsi:schemaLocation`, por ejemplo).

Por ejemplo, si quiere transformar `entrada.xml` con `transform.xslt` en `salida.html`, lo normal sería usar estos comandos en la ILC usando los nombres de archivo:

```
raptorxmlxbrl xslt --input=entrada.xml --output=salida.html transform.xslt
```

No obstante, si tiene una definición de recurso global para el alias `MiEntrada` que apunta al recurso de archivo `PrimeraEntrada.xml` por medio de una configuración llamada `PrimeraConfig`, podría usar el alias `MiEntrada` en la línea de comandos:

```
raptorxmlxbrl xslt --input=altova://file_resource/MiEntrada --gr=C:
\MisRecursosGlobales.xml --gc=PrimeraConfig --output=Salida.html
transform.xslt
```

Ahora imagine que tiene otro recurso de archivo, por ejemplo `SegundaEntrada.xml`, que apunta al alias `MiEntrada` por medio de una configuración llamada `SegundaConfig`, entonces puede usar este otro recurso con solo cambiar la opción `--gc` del comando anterior:

```
raptorxmlxbrl xslt --input=altova://file_resource/MiEntrada --gr=C:
\MisRecursosGlobales.xml --gc=SegundaConfig --output=Salida.html
transform.xslt
```

Nota: en el ejemplo anterior se usó un recurso de archivo. Los recursos de archivo deben llevar el prefijo **`altova://file_resource/`**. También puede usar recursos globales que sean carpetas. Para identificar un recurso de carpeta, utilice el prefijo: **`altova://folder_resource/NombreAlias`**. No olvide que en la interfaz de la línea de comandos puede usar recursos de carpeta como parte de la ruta de acceso. Por ejemplo: **`altova://folder_resource/NombreAlias/entrada.xml`**.

 © 2018 Altova GmbH

2.6 Problemas de seguridad

Temas de este apartado:

- Problemas de seguridad relacionados con la interfaz HTTP
- Trabajar con scripts Python seguros

Algunas características de las interfaces de RaptorXML+XBRL Server plantean algunos problemas de seguridad, que describimos a continuación junto con soluciones para remediarlos.

Problemas de seguridad relacionados con la interfaz HTTP

La interfaz HTTP permite por defecto escribir documentos de resultados en cualquier ubicación indicada por el cliente (y a la que se pueda acceder con el protocolo HTTP). Por tanto, es importante tener en cuenta este aspecto de seguridad cuando instale y configure RaptorXML +XBRL Server.

Si le preocupa que esto pueda comprometer la seguridad de su sistema o que la interfaz se utilice de forma incorrecta, puede configurar el servidor para que escriba los documentos de resultados en un directorio de salida específico del servidor mismo. Esto se consigue estableciendo el valor `false` para la opción `server.unrestricted-filesystem-access` del archivo de configuración del servidor. Si se limita así el acceso, el cliente puede descargar los documentos de resultados del directorio de salida específico mediante solicitudes `GET`. Otra opción es que el administrador copie/cargue los documentos de resultados del servidor en la ubicación de destino.

Trabajar con scripts Python seguros

Cuando se especifica un script Python por HTTP para RaptorXML+XBRL Server, el script solo funciona si está ubicado en el directorio de confianza. El script se ejecuta desde el directorio de confianza. Si especifica un script de cualquier otro directorio, se produce un error. El directorio de confianza se define en la opción `server.script-root-dir` del archivo de configuración del servidor y **es obligatorio** especificar un directorio de confianza si quiere usar scripts Python. Por tanto, asegúrese de guardar en este directorio todos los scripts Python que desea usar.

Aunque todos los resultados generados por el servidor para solicitudes de trabajo HTTP se escriben en el directorio de salida de trabajos (que es un subdirectorio de `output-root-directory`), esta limitación no afecta a los scripts Python, que pueden escribir en cualquier ubicación. El administrador del servidor debería revisar los scripts Python del directorio de confianza para evitar problemas de seguridad.

Altova RaptorXML+XBRL Server 2018

Interfaz de la línea de comandos (ILC)

3 Interfaz de la línea de comandos (ILC)

El ejecutable de RaptorXML+XBRL Server que sirve para trabajar con la interfaz de la línea de comandos (ILC) se instala por defecto en esta carpeta:

Windows `<CarpetaArchivosPrograma>\Altova\RaptorXMLXBRLServer2018\bin\`**`RaptorXMLXBRL.exe`**

Linux `/opt/Altova/RaptorXMLXBRLServer2018/bin/`**`raptorxmlxbrl`**

macOS `/usr/local/Altova/RaptorXMLXBRLServer2018/bin/`**`raptorxmlxbrl`**

Uso de mayúsculas/minúsculas y de barras diagonales en la línea de comandos

`RaptorXMLXBRL` *en Windows*
`raptorxmlxbrl` *en Unix (Linux, Mac)*

* Las minúsculas (`raptorxmlxbrl`) funcionan en todas las plataformas (Windows, Linux y Mac), mientras que la mezcla de mayúsculas y minúsculas (`RaptorXMLXBRL`) sólo funciona en Windows y Mac.
* En Linux y Mac utilice barras diagonales. En Windows utilice barras diagonales inversas.

Uso

Esta es la sintaxis de la línea de comandos:

Windows **`RaptorXMLXBRL --h | --help | --version | <comando> [opciones] [argumentos]`**

Linux **`raptorxmlxbrl --h | --help | --version | <comando> [opciones] [argumentos]`**

Mac **`raptorxmlxbrl --h | --help | --version | <comando> [opciones] [argumentos]`**

`RaptorXMLXBRL`	Llama a la aplicación (en plataformas Windows)
`raptorxmlxbrl`	Llama a la aplicación (en plataformas Unix: Linux y Mac)
`--h | --help`	Muestra el texto de ayuda
`--version`	Muestra el número de versión de la aplicación
`<comando>`	Comando que se debe ejecutar (*ver la lista que aparece más abajo*). Cada comando (con sus argumentos y opciones respectivos) se describe y explica en los apartados de esta sección.
`[opciones]`	Opciones de un comando. Cada apartado de esta sección describe un comando y sus opciones. Además, el apartado Opciones enumera todas

 © 2018 Altova GmbH

	las opciones de la línea de comandos.
`[argumentos]`	Argumentos de un comando. Cada apartado de esta sección describe un comando y sus argumentos.

Comandos de la ILC

A continuación enumeramos todos los comandos de la ILC de RaptorXML ordenados según su función (algunos aparecen en más de una categoría). Estos comandos se describen uno a uno en los apartados de esta sección, junto con sus opciones y argumentos.

Todos los comandos de validación

`valdtd | dtd`	Valida un documento DTD.
`valxsd | xsd`	Valida un documento de W3C XML Schema.
`valxml-withdtd | xml`	Valida un documento XML con una DTD.
`valxml-withxsd | xsi`	Valida un documento XML con un esquema XML.
`valxslt`	Valida un documento XSLT.
`valxquery`	Valida un documento XQuery.
`valxbrl | xbrl`	Valida un documento de instancia XBRL (extensión `.xbrl`).
`valinlinexbrl | ixbrl`	Valida un documento Inline XBRL (iXBRL).
`valxbrltaxonomy | dts`	Valida un documento de taxonomía (esquema) XBRL (extensión `.xsd`).
`valtaxonomypackage`	Valida un paquete de taxonomías XBRL.
`valjson`	Valida un documento JSON.
`valjsonschema`	Valida un documento de esquema JSON con la especificación de esquema.
`valavroschema`	Valida un esquema Avro con la especificación Avro schema.
`valavrojson`	Valida un archivo de datos JSON con un esquema Avro.
`valavro`	Valida los datos de un binario Avro con su esquema Avro.
`valany`	Valida cualquier tipo de documento compatible. El tipo de documento se detecta automáticamente.

Comandos para comprobar el formato XML

`wfjson`	Comprueba si un documento JSON tiene un formato correcto.
`wfxml`	Comprueba si un documento XML tiene un formato correcto.
`wfdtd`	Comprueba si un documento DTD tiene un formato correcto.
`wfany`	Comprueba si el documento (ya sea XML o DTD) tiene un formato correcto.

Comandos de validación XBRL

valxbrl | xbrl	Valida un documento de instancia XBRL (extensión `.xbrl`).
valinlinexbrl | ixbrl	Valida un documento Inline XBRL (iXBRL).
valxbrltaxonomy | dts	Valida un documento de taxonomía (esquema) XBRL (extensión `.xsd`).
valtaxonomypackage	Valida un paquete de taxonomías XBRL.
valxbrlver	Valida informes de versionado XBRL.

Comandos XSLT

xslt	Realiza una transformación con un archivo XSLT.
valxslt	Valida un documento XSLT.

Comandos XQuery

xquery	Ejecuta un XQuery con un archivo XQuery dado.
valxquery	Valida un documento XQuery.

Comandos XML Signature

xmlsignature-sign	Crea un documento de salida de firma XML a partir de un documento de entrada.
xmlsignature-verify	Verifica un documento de firma XML.
xmlsignature-update	Actualiza la firma de un documento XML (modificado).
xmlsignature-remove	Quita la firma de un documento XML.

Comandos JSON/Avro

avroextractschema	Extrae el esquema Avro de un archivo binario Avro.
valavro	Valida los datos de archivos binarios Avro con el esquema Avro correspondiente a cada binario.
valavrojson	Valida archivos de datos JSON con un esquema Avro.
valavroschema	Valida un esquema Avro con la especificación Avro schema.
valjsonschema	Comprueba la validez de documentos de esquema JSON.

 © 2018 Altova GmbH

valjson	Comprueba la validez de documentos JSON.
wfjson	Comprueba si un documento JSON tiene un formato correcto.

3.1 Comandos para validar XML, DTD, XSD

Los comandos de validación XML sirven para validar este tipo de documentos:

- *XML:* los documentos de instancia XML se validan con una DTD (`valxml-withdtd | xml`) o con un esquema XML 1.0/1.1 (`valxml-withxsd | xsi`).
- *DTD:* los DTD se revisan para ver si su formato correcto y confirmar que no tienen errores (`valdtd | dtd`).
- *XSD:* los esquemas XML del W3C (XSD) se validan según las reglas de la especificación XML Schema (`valxsd | xsd`).

`valxml-withdtd	xml`	Valida un documento de instancia XML con una DTD.
`valxml-withxsd	xsi`	Valida un documento de instancia XML con un esquema XML.
`valdtd	dtd`	Valida un documento DTD.
`valxsd	xsd`	Valida un documento de esquema XML del W3C (XSD).

Nota: también se pueden validar instancias XBRL, taxonomías XBRL, documentos XSLT, documentos XQuery, documentos JSON y documentos Avro. Estos comandos de validación se describen en los apartados correspondientes: Comandos de validación XBRL, Comandos XSLT, Comandos XQuery y Comandos JSON/Avro.

3.1.1 valxml-withdtd (xml)

El comando **`valxml-withdtd | xml`** valida uno o varios documentos XML de instancia con una DTD.

Windows **`RaptorXMLXBRL valxml-withdtd | xml [opciones] ArchivoEntrada`**

Linux **`raptorxmlxbrl valxml-withdtd | xml [opciones] ArchivoEntrada`**

Mac **`raptorxmlxbrl valxml-withdtd | xml [opciones] ArchivoEntrada`**

El argumento *`ArchivoEntrada`* es el documento XML que debe validarse. Si existe una referencia a una DTD en el documento, no es necesario usar la opción `--dtd`.

Si desea validar varios documentos, tiene dos opciones: (i) enumerar los archivos que se deben validar en la línea de comandos, separados por espacios o (ii) enumerar los archivos que se deben validar en un archivo de texto (`.txt`) donde aparece un nombre de archivo por línea y dar este archivo de texto como argumento *`ArchivoEntrada`* junto con la opción `--listfile` con valor `true` (*ver lista de opciones más abajo*).

Ejemplos

- **raptorxmlxbrl** `valxml-withdtd --dtd=c:\MiDTD.dtd c:\Test.xml`
- **raptorxmlxbrl** `xml c:\Test.xml`
- **raptorxmlxbrl** `xml --verbose=true c:\Test.xml`
- **raptorxmlxbrl** `xml --listfile=true c:\ListaArchivos.txt`

Uso de mayúsculas/minúsculas y de barras diagonales en la línea de comandos

RaptorXMLXBRL *en Windows*
raptorxmlxbrl *en Unix (Linux, Mac)*

* Las minúsculas (`raptorxmlxbrl`) funcionan en todas las plataformas (Windows, Linux y Mac), mientras que la mezcla de mayúsculas y minúsculas (`RaptorXMLXBRL`) sólo funciona en Windows y Mac.
* En Linux y Mac utilice barras diagonales. En Windows utilice barras diagonales inversas.

Opciones

Las opciones del comando aparecen más abajo divididas en grupos. Los valores se pueden dar sin comillas excepto en estos dos casos: (i) cuando la cadena de valor contiene espacios y (ii) cuando en la descripción de la opción se indique explícitamente que es necesario el uso de comillas.

Validación y procesamiento

dtd

--dtd = *ARCHIVO*
Especifica el documento DTD externo que debe utilizarse para la validación. Si en el documento XML hay una referencia a una DTD externa, esta opción de la ILC reemplaza a la referencia externa.

listfile

--listfile = `true|false`
Si el valor es `true`, el argumento *ArchivoEntrada* del comando se entiende como un archivo de texto que contiene un nombre de archivo por línea. Otra opción es enumerar los archivos en la ILC, separados por un espacio. No obstante, recuerde que las ILC tienen un límite de caracteres. Además, no olvide que la opción `--listfile` solamente afecta a los argumentos y no a las opciones.
Valor predeterminado: `false`.
Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

namespaces

--namespaces = `true|false`
Habilita el procesamiento preparado para espacios de nombres. Esta opción es muy útil si quiere buscar en la instancia XML errores resultantes de espacios de nombres erróneos.
Valor predeterminado: `false`.
Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- recurse

  --recurse = true|false

  Esta opción se utiliza para seleccionar ficheros dentro de subdirectorios (incluso en archivos ZIP). Si el valor es true, el argumento *ArchivoEntrada* del comando seleccionará el fichero seleccionado también en los subdirectorios. Por ejemplo: test.zip|zip\test.xml seleccionará los ficheros llamados test.xml en todos los subdirectorios de la carpeta ZIP. Si quiere puede usar los caracteres comodín * y ?. Por ejemplo: *.xml seleccionaría todos los ficheros de la carpeta ZIP que tengan la extensión .xml. Valor predeterminado: false

  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en true.

- streaming

  --streaming = true|false

  Habilita la transmisión por secuencias. En el modo de transmisión por secuencias, el almacenamiento de datos en memoria se reduce al mínimo y el procesamiento es más rápido. El inconveniente es que puede que no esté disponible cierta información que podría necesitar más adelante, como el modelo de datos del documento XML, por ejemplo. Si quiere evitar esto, debería deshabilitar el modo de transmisión por secuencias (dándole el valor false a la opción --streaming). Cuando use la opción --script con el comando valxml-withxsd, aconsejamos deshabilitar la transmisión por secuencias. Recuerde que la opción --streaming se ignora, si el valor de --parallel-assessment es true.

  Valor predeterminado: true.

  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en true.

- Catálogos y recursos globales

  - catalog

    --catalog = *ARCHIVO*

    Especifica la ruta de acceso absoluta a un archivo de catálogo que no está en el archivo de catálogo raíz instalado. El valor predeterminado es la ruta de acceso absoluta del archivo de catálogo raíz instalado (<carpeta-instalación>\Altova\RaptorXMLXBRLServer2018\etc\RootCatalog.xml). Consulte el apartado Catálogos XML para obtener más información.

  - user-catalog

    --user-catalog = *ARCHIVO*

    Especifica la ruta de acceso absoluta a un catálogo XML que debe utilizarse junto con el catálogo raíz. Consulte el apartado Catálogos XML para obtener más información.

  - enable-globalresources

    --enable-globalresources = true|false

    Habilita la función de recursos globales. Valor predeterminado: false.

    Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en true.

 © 2018 Altova GmbH

- globalresourceconfig [gc]

  --gc | --globalresourceconfig = *VALOR*
  Especifica la configuración activa del recurso global (y habilita los recursos globales).

- globalresourcefile [gr]

  --gr | --globalresourcefile = *ARCHIVO*
  Especifica el archivo de recursos globales (y habilita los recursos globales).

- Mensajes, errores, ayuda, tiempo de espera, versión
  - error-format

    --error-format = `text|shortxml|longxml`
    Especifica el formato de la salida de error. Los valores posibles son formatos de texto, XML y XML detallado (`longxml`). Valor predeterminado: `text`.

  - error-limit

    --error-limit = *N*
    Especifica el límite de errores con un valor comprendido entre `1` y `999`. El valor predeterminado es `100`. Cuando se alcanza el límite de error, se detiene la validación. Esta opción es muy práctica a la hora de limitar el uso del procesador durante la validación o transformación.

  - help

    --help
    Muestra el texto de ayuda para el comando. Por ejemplo `valany --h`. (Otra opción es usar el comando `help` con un argumento. Por ejemplo: `help valany`).

  - log-output

    --log-output = *ARCHIVO*
    Escribe el registro de salida en la URL de archivo indicada. Compruebe que la ILC tiene permiso de escritura en la ubicación de destino.

  - network-timeout

    --network-timeout = *VALOR*
    Indica el tiempo de espera en segundos para operaciones remotas de entrada y salida. Valor predeterminado: `40`.

  - verbose

    --verbose = `true|false`
    Si el valor es `true`, se genera información adicional durante la validación. Valor predeterminado es `false`.
    Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

  - verbose-output

    --verbose-output = *ARCHIVO*

Escribe el resultado detallado en el *ARCHIVO* indicado.

- version

  --version
  Muestra el número de versión de RaptorXML+XBRL Server. Si se utiliza con un comando, escriba la opción `--version` antes del comando.

- warning-limit

  --warning-limit = *VALUE*
  Especifica el límite de advertencia en el rango `1-65535`. El procesamiento continua si se alcanza el límite pero no se registrarán más advertencias. El valor por defecto es `100`.

3.1.2 valxml-withxsd (xsi)

El comando **valxml-withxsd | xsi** valida uno o varios documentos XML de instancia con las especificaciones XML Schema Definition Language (XSD) 1.0 y 1.1 del W3C.

Windows	**RaptorXMLXBRL valxml-withxsd | xsi [opciones] *ArchivoEntrada***
Linux	**raptorxmlxbrl valxml-withxsd | xsi [opciones] *ArchivoEntrada***
Mac	**raptorxmlxbrl valxml-withxsd | xsi [opciones] *ArchivoEntrada***

El argumento **ArchivoEntrada** suministra el documento XML que debe validarse. La opción `--schemalocation-hints=true|false` indica si la referencia XSD del documento XML debe utilizarse o no, siendo `true` su valor predeterminado (es decir, se utiliza). La opción `--xsd=ARCHIVO` especifica qué esquema se utiliza.

Si desea validar varios documentos, tiene dos opciones: (i) enumerar los archivos que se deben validar en la línea de comandos, separados por espacios o (ii) enumerar los archivos que se deben validar en un archivo de texto (`.txt`) donde aparece un nombre de archivo por línea y dar este archivo de texto como argumento *ArchivoEntrada* junto con la opción `--listfile` con valor `true` (*ver lista de opciones más abajo*).

Nota: si usa la opción `--script` para ejecutar scripts Python, no olvide especificar la opción `--streaming=false`.

Ejemplos

- **raptorxmlxbrl** `valxml-withxsd --schemalocation-hints=false --xsd=c:\MiXSD.xsd c:\SinRefXSD.xml`
- **raptorxmlxbrl** `xsi c:\SinRefXSD.xml`
- **raptorxmlxbrl** `xsi --xsd-version=1.1 --listfile=true c:\ListaArchivos.txt`

 © 2018 Altova GmbH

- Uso de mayúsculas/minúsculas y de barras diagonales en la línea de comandos

  `RaptorXMLXBRL` *en Windows*
  `raptorxmlxbrl` *en Unix (Linux, Mac)*

  * Las minúsculas (`raptorxmlxbrl`) funcionan en todas las plataformas (Windows, Linux y Mac), mientras que la mezcla de mayúsculas y minúsculas (`RaptorXMLXBRL`) sólo funciona en Windows y Mac.
  * En Linux y Mac utilice barras diagonales. En Windows utilice barras diagonales inversas.

Opciones

Las opciones del comando aparecen más abajo divididas en grupos. Los valores se pueden dar sin comillas excepto en estos dos casos: (i) cuando la cadena de valor contiene espacios y (ii) cuando en la descripción de la opción se indique explícitamente que es necesario el uso de comillas.

- Validación y procesamiento
  - assessment-mode

    `--assessment-mode =` `lax|strict`
    Especifica el modo de evaluación de la validez del esquema, según se define en las especificaciones XSD. El documento XML de instancia se validará en función del modo especificado en esta opción. Valor predeterminado: `strict`.

  - ct-restrict-mode

    `--ct-restrict-mode =` `1.0|1.1|default`
    Especifica cómo comprobar restricciones de tipo complejo. Un valor de **`1.0`** comprueba restricciones de tipo complejo conforme a lo definido en la especificación XSD 1.0 (incluso estando en modo de validación XSD 1.1). Un valor de **`1.1`** comprueba restricciones de tipo complejo conforme a los definido en la especificación XSD 1.1 (incluso estando en modo de validación XSD 1.0). Un valor de **`default`** comprueba restricciones de tipo complejo conforme a lo definido en la especificación XSD del modo de validación habilitado en ese momento (1.0 o 1.1). El valor predeterminado es **`default`**.

  - listfile

    `--listfile =` `true|false`
    Si el valor es `true`, el argumento *`ArchivoEntrada`* del comando se entiende como un archivo de texto que contiene un nombre de archivo por línea. Otra opción es enumerar los archivos en la ILC, separados por un espacio. No obstante, recuerde que las ILC tienen un límite de caracteres. Además, no olvide que la opción `--listfile` solamente afecta a los argumentos y no a las opciones.
    Valor predeterminado: `false`.
    Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

  - parallel-assessment [pa]

    `--pa | --parallel-assessment =` `true|false`
    Si el valor es `true`, la evaluación de la validez de esquemas se realiza en paralelo. Esto significa que si hay más de 128 elementos en cualquiera de los niveles, estos

elementos se procesan en paralelo utilizando varios subprocesos. Por tanto, los archivos XML de gran tamaño se pueden procesar más rápido si se habilita esta opción. La evaluación en paralelo se lleva cabo nivel por nivel, pero puede ocurrir en varios niveles de un mismo conjunto de información. Recuerde que la evaluación en paralelo no funciona en modo de transmisión por secuencias. Por este motivo la opción `--streaming` se pasa por alto si el valor de la opción `--parallel-assessment` es `true`. Además, se usa más memoria cuando se utiliza la opción `--parallel-assessment`. El valor predeterminado de esta opción es false y su forma abreviada es `--pa`.

Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

recurse

`--recurse =` `true|false`

Esta opción se utiliza para seleccionar ficheros dentro de subdirectorios (incluso en archivos ZIP). Si el valor es `true`, el argumento *`ArchivoEntrada`* del comando seleccionará el fichero seleccionado también en los subdirectorios. Por ejemplo: `test.zip|zip\test.xml` seleccionará los ficheros llamados `test.xml` en todos los subdirectorios de la carpeta ZIP. Si quiere puede usar los caracteres comodín `*` y `?`. Por ejemplo: `*.xml` seleccionaría todos los ficheros de la carpeta ZIP que tengan la extensión `.xml`. Valor predeterminado: `false`

Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

schema-imports

`--schema-imports =` `load-by-schemalocation | load-preferring-schemalocation | load-by-namespace | load-combining-both | license-namespace-only`

Esta opción indica el comportamiento de los elementos `xs:import`. Cada uno de estos elementos tiene un atributo opcional `namespace` y un atributo opcional `schemaLocation`: `<import namespace="unEspacioNombres" schemaLocation="unaURL">`. La opción indica si se debe cargar un documento de esquema o solo autorizar a un espacio de nombres. Si la opción indica que se debe cargar un documento de esquema, entonces indica también qué información debe utilizarse para encontrar el documento de esquema. Valor predeterminado: `load-preferring-schemalocation`.

- `load-by-schemalocation`: el valor del atributo `schemaLocation` se utiliza para buscar el esquema, teniendo en cuenta las asignaciones de catálogo. Si está presente el atributo `namespace`, se importa el espacio de nombres (con licencia).
- `load-preferring-schemalocation`: si está presente, se utiliza el atributo `schemaLocation` teniendo en cuenta las asignaciones de catálogo. Si no está presente el atributo `schemaLocation`, entonces se usa el valor del atributo `namespace` a través de las asignaciones de catálogo. Este es el **valor predeterminado**.
- `load-by-namespace`: el valor del atributo `namespace` se utiliza para buscar el esquema por medio de una asignación de catálogo.
- `load-combining-both`: si el atributo `namespace` o `schemaLocation` tiene una asignación de catálogo, entonces se usa la asignación. Si ambos atributos tienen asignaciones de catálogo, entonces es el valor de la opción `--schema-mapping` (opción XBRL y opción XML/XSD) decide qué asignación se utiliza. Si no hay

ninguna asignación de catálogo, entonces se usa el atributo `schemaLocation`.
- `license-namespace-only`: se importa el espacio de nombres. No se importa el documento de esquema.

schemalocation-hints

`--schemalocation-hints =` `load-by-schemalocation | load-by-namespace | load-combining-both | ignore`
Determina el comportamiento predeterminado de los atributos `xsi:schemaLocation` y `xsi:noNamespaceSchemaLocation`. Indica si se debe cargar un documento de esquema y, si así es, indica qué información debe utilizarse para encontrarlo. Valor predeterminado: `load-by-schemalocation`.

- **Valor predeterminado**: `load-by-schemalocation`. Este valor toma la URL de la ubicación del esquema de los atributos `xsi:schemaLocation` y `xsi:noNamespaceSchemaLocation` de los documentos de instancia XML o XBRL.

- El valor `load-by-namespace` toma la parte de espacio de nombres del atributo `xsi:schemaLocation` y una cadena vacía en el caso del atributo `xsi:noNamespaceSchemaLocation` y encuentra el esquema por medio de una asignación de catálogo.

- Si usa el valor `load-combining-both` y el espacio de nombres o la URL tienen una asignación de catálogo, se usa dicha asignación. Si ambos tienen asignaciones de catálogo, el valor de la opción `schema-mapping` (opción XBRL y opción XML/XSD) decide qué asignación se utiliza. Si ni el espacio de nombres ni la URL tiene una asignación de catálogo, se usa la URL.

- El valor `ignore` ignora los atributos `xsi:schemaLocation` y `xsi:noNamespaceSchemaLocation`.

schema-mapping

`--schema-mapping =` `prefer-schemalocation | prefer-namespace`
Si se usa la ubicación y el espacio de nombres para buscar el documento de esquema, esta opción indica cuál de ellos debe ser la opción preferida durante la búsqueda en el catálogo.
Si la opción `--schemalocation-hints` o la opción `--schema-imports` tiene el valor `load-combining-both` y si las partes de espacio de nombres y URL pertinentes tienen asignaciones de catálogo, entonces el valor de la opción especifica cuál de las dos asignaciones se utiliza (la asignación del espacio de nombres o de la URL: el valor `prefer-schemalocation` se refiere a la asignación de la URL). Valor predeterminado: `prefer-schemalocation`.

script

`--script =` *`ARCHIVO`*
Una vez finalizada la validación, ejecuta el script Python. Para indicar más de un script basta con agregar la opción varias veces.

script-api-version

`--api, --script-api-version =` `1|2|2.1|2.2|2.3|2.4|2.4.1|2.5`
Especifica la versión API Python que ha de usarse para el script. El valor por defecto es

la última versión, actualmente **2.45**. En vez de de los valores 1 y 2, también puede usar losa valores 1.0 y 2.0 respectivamente.

- script-output

  --script-output = *FILE*
  Escribe el resultado estandard del script en el archivo nombres como *FILE*.

- script-param

  --script-param = *CLAVE:VALOR*
  Parámetros definidos por el usuario a los que se puede acceder durante la ejecución de scripts Python.

- streaming

  --streaming = true|false
  Habilita la transmisión por secuencias. En el modo de transmisión por secuencias, el almacenamiento de datos en memoria se reduce al mínimo y el procesamiento es más rápido. El inconveniente es que puede que no esté disponible cierta información que podría necesitar más adelante, como el modelo de datos del documento XML, por ejemplo. Si quiere evitar esto, debería deshabilitar el modo de transmisión por secuencias (dándole el valor false a la opción --streaming). Cuando use la opción --script con el comando valxml-withxsd, aconsejamos deshabilitar la transmisión por secuencias. Recuerde que la opción --streaming se ignora, si el valor de --parallel-assessment es true.
  Valor predeterminado: true.
  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en true.

- xinclude

  --xinclude = true|false
  Habilita la compatibilidad con inclusiones XML (XInclude). Si el valor es false, los elementos XInclude include se ignoran. Valor predeterminado: false.
  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en true.

- xml-mode

  --xml-mode = wf|id|valid
  Especifica el modo de procesamiento XML que debe utilizarse para el documento de instancia XML: wf=comprobación de formato; id=comprobación de formato con ID/IDREF; valid=validación. Valor predeterminado: wf. Recuerde que el valor valid exige que cada documento de instancia que se cargue durante el procesamiento haga referencia a una DTD. Si no existe ninguna DTD, se generará un error.

- xml-mode-for-schemas

  --xml-mode-for-schemas = wf|id|valid
  Especifica el modo de procesamiento XML que debe utilizarse para el documento de instancia XML: wf=comprobación de formato; id=comprobación de formato con ID/IDREF; valid=validación. Valor predeterminado: wf. Recuerde que el valor valid exige que cada documento de esquema que se cargue durante el procesamiento haga

 © 2018 Altova GmbH

referencia a una DTD. Si no existe ninguna DTD, se generará un error.

- xsd

  --xsd = *ARCHIVO*

  Especifica qué esquemas XML deben utilizarse para la validación de documentos XML. Si quiere especificar más de un esquema, añada la opción varias veces.

- xsd-version

  --xsd-version = `1.0|1.1|detect`

  Especifica qué versión de la especificación Schema Definition Language (XSD) del W3C se debe usar. Valor predeterminado: `1.0`.

  Esta opción también puede ser útil si quiere ver en qué aspectos no es compatible un esquema 1.0 con la especificación 1.1. El valor `detect` es una característica de Altova. Permite detectar la versión del esquema XML (`1.0` o `1.1`) leyendo el valor del atributo `vc:minVersion` del elemento `<xs:schema>` del documento. Si el valor del atributo `@vc:minVersion` es `1.1`, se entiende que la versión del esquema es `1.1`. Si el atributo tiene otro valor que no sea `1.1` (o si no está presente el atributo `@vc:minVersion`), se entiende que la versión del esquema es `1.0`.

- Catálogos y recursos globales
  - catalog

    --catalog = *ARCHIVO*

    Especifica la ruta de acceso absoluta a un archivo de catálogo que no está en el archivo de catálogo raíz instalado. El valor predeterminado es la ruta de acceso absoluta del archivo de catálogo raíz instalado (`<carpeta-instalación>\Altova\RaptorXMLXBRLServer2018\etc\RootCatalog.xml`). Consulte el apartado Catálogos XML para obtener más información.

  - user-catalog

    --user-catalog = *ARCHIVO*

    Especifica la ruta de acceso absoluta a un catálogo XML que debe utilizarse junto con el catálogo raíz. Consulte el apartado Catálogos XML para obtener más información.

  - enable-globalresources

    --enable-globalresources = `true|false`

    Habilita la función de recursos globales. Valor predeterminado: `false`.

    Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

  - globalresourceconfig [gc]

    --gc | --globalresourceconfig = *VALOR*

    Especifica la configuración activa del recurso global (y habilita los recursos globales).

  - globalresourcefile [gr]

    --gr | --globalresourcefile = *ARCHIVO*

Especifica el archivo de recursos globales (y habilita los recursos globales).

- Mensajes, errores, ayuda, tiempo de espera, versión
  - error-format

    --error-format = `text|shortxml|longxml`
    Especifica el formato de la salida de error. Los valores posibles son formatos de texto, XML y XML detallado (`longxml`). Valor predeterminado: `text`.

  - error-limit

    --error-limit = *N*
    Especifica el límite de errores con un valor comprendido entre `1` y `999`. El valor predeterminado es `100`. Cuando se alcanza el límite de error, se detiene la validación. Esta opción es muy práctica a la hora de limitar el uso del procesador durante la validación o transformación.

  - help

    --help
    Muestra el texto de ayuda para el comando. Por ejemplo `valany --h`. (Otra opción es usar el comando `help` con un argumento. Por ejemplo: `help valany`).

  - log-output

    --log-output = *ARCHIVO*
    Escribe el registro de salida en la URL de archivo indicada. Compruebe que la ILC tiene permiso de escritura en la ubicación de destino.

  - network-timeout

    --network-timeout = *VALOR*
    Indica el tiempo de espera en segundos para operaciones remotas de entrada y salida. Valor predeterminado: `40`.

  - verbose

    --verbose = `true|false`
    Si el valor es `true`, se genera información adicional durante la validación. Valor predeterminado es `false`.
    Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

  - verbose-output

    --verbose-output = *ARCHIVO*
    Escribe el resultado detallado en el *ARCHIVO* indicado.

  - version

    --version
    Muestra el número de versión de RaptorXML+XBRL Server. Si se utiliza con un comando, escriba la opción `--version` antes del comando.

 © 2018 Altova GmbH

- warning-limit

  --warning-limit = *VALUE*

  Especifica el límite de advertencia en el rango `1-65535`. El procesamiento continua si se alcanza el límite pero no se registrarán más advertencias. El valor por defecto es `100`.

3.1.3 valdtd (dtd)

El comando **valdtd | dtd** valida uno o varios documentos DTD con la especificación XML 1.0 o XML 1.1.

Windows	**RaptorXMLXBRL valdtd | dtd [opciones]** ***ArchivoEntrada***
Linux	**raptorxmlxbrl valdtd | dtd [opciones]** ***ArchivoEntrada***
Mac	**raptorxmlxbrl valdtd | dtd [opciones]** ***ArchivoEntrada***

El argumento *ArchivoEntrada* es el documento DTD que debe validarse.

Si desea validar varios documentos, tiene dos opciones: (i) enumerar los archivos que se deben validar en la línea de comandos, separados por espacios o (ii) enumerar los archivos que se deben validar en un archivo de texto (`.txt`) donde aparece un nombre de archivo por línea y dar este archivo de texto como argumento *ArchivoEntrada* junto con la opción `--listfile` con valor `true` (*ver lista de opciones más abajo*).

Ejemplos

- **raptorxmlxbrl** `valdtd c:\Test.dtd`
- **raptorxmlxbrl** `dtd --verbose=true c:\Test.dtd`
- **raptorxmlxbrl** `dtd --listfile=true c:\ListaArchivos.txt`

- Uso de mayúsculas/minúsculas y de barras diagonales en la línea de comandos

  RaptorXMLXBRL *en Windows*
  raptorxmlxbrl *en Unix (Linux, Mac)*

  * Las minúsculas (`raptorxmlxbrl`) funcionan en todas las plataformas (Windows, Linux y Mac), mientras que la mezcla de mayúsculas y minúsculas (`RaptorXMLXBRL`) sólo funciona en Windows y Mac.
  * En Linux y Mac utilice barras diagonales. En Windows utilice barras diagonales inversas.

Opciones

Las opciones del comando aparecen más abajo divididas en grupos. Los valores se pueden dar

sin comillas excepto en estos dos casos: (i) cuando la cadena de valor contiene espacios y (ii) cuando en la descripción de la opción se indique explícitamente que es necesario el uso de comillas.

- Validación y procesamiento
  - listfile

    --listfile = true|false
    Si el valor es true, el argumento *ArchivoEntrada* del comando se entiende como un archivo de texto que contiene un nombre de archivo por línea. Otra opción es enumerar los archivos en la ILC, separados por un espacio. No obstante, recuerde que las ILC tienen un límite de caracteres. Además, no olvide que la opción --listfile solamente afecta a los argumentos y no a las opciones.
    Valor predeterminado: false.
    Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en true.

  - recurse

    --recurse = true|false
    Esta opción se utiliza para seleccionar ficheros dentro de subdirectorios (incluso en archivos ZIP). Si el valor es true, el argumento *ArchivoEntrada* del comando seleccionará el fichero seleccionado también en los subdirectorios. Por ejemplo: test.zip|zip\test.xml seleccionará los ficheros llamados test.xml en todos los subdirectorios de la carpeta ZIP. Si quiere puede usar los caracteres comodín * y ?. Por ejemplo: *.xml seleccionaría todos los ficheros de la carpeta ZIP que tengan la extensión .xml. Valor predeterminado: false
    Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en true.

  - script

    --script = *ARCHIVO*
    Una vez finalizada la validación, ejecuta el script Python. Para indicar más de un script basta con agregar la opción varias veces.

  - script-api-version

    --api, --script-api-version = 1|2|2.1|2.2|2.3|2.4|2.4.1|2.5
    Especifica la versión API Python que ha de usarse para el script. El valor por defecto es la última versión, actualmente **2.45**. En vez de de los valores 1 y 2, también puede usar losa valores 1.0 y 2.0 respectivamente.

  - script-output

    --script-output = *FILE*
    Escribe el resultado estandard del script en el archivo nombres como *FILE*.

  - script-param

    --script-param = *CLAVE:VALOR*
    Parámetros definidos por el usuario a los que se puede acceder durante la ejecución de scripts Python.

 © 2018 Altova GmbH

- Catálogos y recursos globales
  - catalog

    --catalog = *ARCHIVO*
    Especifica la ruta de acceso absoluta a un archivo de catálogo que no está en el archivo de catálogo raíz instalado. El valor predeterminado es la ruta de acceso absoluta del archivo de catálogo raíz instalado (`<carpeta-instalación>\Altova\RaptorXMLXBRLServer2018\etc\RootCatalog.xml`). Consulte el apartado Catálogos XML para obtener más información.

  - user-catalog

    --user-catalog = *ARCHIVO*
    Especifica la ruta de acceso absoluta a un catálogo XML que debe utilizarse junto con el catálogo raíz. Consulte el apartado Catálogos XML para obtener más información.

  - enable-globalresources

    --enable-globalresources = `true|false`
    Habilita la función de recursos globales. Valor predeterminado: `false`.
    Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

  - globalresourceconfig [gc]

    --gc | --globalresourceconfig = *VALOR*
    Especifica la configuración activa del recurso global (y habilita los recursos globales).

  - globalresourcefile [gr]

    --gr | --globalresourcefile = *ARCHIVO*
    Especifica el archivo de recursos globales (y habilita los recursos globales).

- Mensajes, errores, ayuda, tiempo de espera, versión
  - error-format

    --error-format = `text|shortxml|longxml`
    Especifica el formato de la salida de error. Los valores posibles son formatos de texto, XML y XML detallado (`longxml`). Valor predeterminado: `text`.

  - error-limit

    --error-limit = *N*
    Especifica el límite de errores con un valor comprendido entre `1` y `999`. El valor predeterminado es `100`. Cuando se alcanza el límite de error, se detiene la validación. Esta opción es muy práctica a la hora de limitar el uso del procesador durante la validación o transformación.

  - help

    --help
    Muestra el texto de ayuda para el comando. Por ejemplo `valany --h`. (Otra opción es

usar el comando `help` con un argumento. Por ejemplo: `help valany`).

- log-output

  `--log-output =` *`ARCHIVO`*
  Escribe el registro de salida en la URL de archivo indicada. Compruebe que la ILC tiene permiso de escritura en la ubicación de destino.

- network-timeout

  `--network-timeout =` *`VALOR`*
  Indica el tiempo de espera en segundos para operaciones remotas de entrada y salida. Valor predeterminado: `40`.

- verbose

  `--verbose =` `true|false`
  Si el valor es `true`, se genera información adicional durante la validación. Valor predeterminado es `false`.
  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- verbose-output

  `--verbose-output =` *`ARCHIVO`*
  Escribe el resultado detallado en el *`ARCHIVO`* indicado.

- version

  `--version`
  Muestra el número de versión de RaptorXML+XBRL Server. Si se utiliza con un comando, escriba la opción `--version` antes del comando.

- warning-limit

  `--warning-limit =` *`VALUE`*
  Especifica el límite de advertencia en el rango `1-65535`. El procesamiento continua si se alcanza el límite pero no se registrarán más advertencias. El valor por defecto es `100`.

3.1.4 valxsd (xsd)

El comando **`valxsd | xsd`** valida uno o varios documentos de esquema XML (documentos XSD) con la especificación XML Schema Definition Language (XSD) 1.0 o 1.1 del W3C. Tenga en cuenta que lo que se valida con la especificación XML Schema es un esquema XML y no un documento XML de instancia con un esquema XML.

Windows **`RaptorXMLXBRL valxsd | xsd [opciones] ArchivoEntrada`**

Linux **`raptorxmlxbrl valxsd | xsd [opciones] ArchivoEntrada`**

 © 2018 Altova GmbH

Mac **raptorxmlxbrl valxsd | xsd [opciones] *ArchivoEntrada***

El argumento *ArchivoEntrada* suministra el documento de esquema XML que debe validarse. La opción --xsd-version=1.0|1.1|detect indica la versión XSD con la que debe validarse, siendo 1.0 su valor predeterminado.

Si desea validar varios documentos, tiene dos opciones: (i) enumerar los archivos que se deben validar en la línea de comandos, separados por espacios o (ii) enumerar los archivos que se deben validar en un archivo de texto (.txt) donde aparece un nombre de archivo por línea y dar este archivo de texto como argumento *ArchivoEntrada* junto con la opción --listfile con valor true (*ver lista de opciones más abajo*).

Ejemplos

- **raptorxmlxbrl** valxsd c:\Test.xsd
- **raptorxmlxbrl** xsd --verbose=true c:\Test.xsd
- **raptorxmlxbrl** xsd --listfile=true c:\ListaArchivos.txt

Uso de mayúsculas/minúsculas y de barras diagonales en la línea de comandos

RaptorXMLXBRL *en Windows*
raptorxmlxbrl *en Unix (Linux, Mac)*

* Las minúsculas (raptorxmlxbrl) funcionan en todas las plataformas (Windows, Linux y Mac), mientras que la mezcla de mayúsculas y minúsculas (RaptorXMLXBRL) sólo funciona en Windows y Mac.
* En Linux y Mac utilice barras diagonales. En Windows utilice barras diagonales inversas.

Opciones

Las opciones del comando aparecen más abajo divididas en grupos. Los valores se pueden dar sin comillas excepto en estos dos casos: (i) cuando la cadena de valor contiene espacios y (ii) cuando en la descripción de la opción se indique explícitamente que es necesario el uso de comillas.

Validación y procesamiento

ct-restrict-mode

--ct-restrict-mode = 1.0|1.1|default

Especifica cómo comprobar restricciones de tipo complejo. Un valor de **1.0** comprueba restricciones de tipo complejo conforme a lo definido en la especificación XSD 1.0 (incluso estando en modo de validación XSD 1.1). Un valor de **1.1** comprueba restricciones de tipo complejo conforme a los definido en la especificación XSD 1.1 (incluso estando en modo de validación XSD 1.0). Un valor de **default** comprueba restricciones de tipo complejo conforme a lo definido en la especificación XSD del modo de validación habilitado en ese momento (1.0 o 1.1). El valor predeterminado es **default**.

listfile

--listfile = true|false
Si el valor es true, el argumento *ArchivoEntrada* del comando se entiende como un archivo de texto que contiene un nombre de archivo por línea. Otra opción es enumerar los archivos en la ILC, separados por un espacio. No obstante, recuerde que las ILC tienen un límite de caracteres. Además, no olvide que la opción --listfile solamente afecta a los argumentos y no a las opciones.
Valor predeterminado: false.
Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en true.

- recurse

--recurse = true|false
Esta opción se utiliza para seleccionar ficheros dentro de subdirectorios (incluso en archivos ZIP). Si el valor es true, el argumento *ArchivoEntrada* del comando seleccionará el fichero seleccionado también en los subdirectorios. Por ejemplo: test.zip|zip\test.xml seleccionará los ficheros llamados test.xml en todos los subdirectorios de la carpeta ZIP. Si quiere puede usar los caracteres comodín * y ?. Por ejemplo: *.xml seleccionaría todos los ficheros de la carpeta ZIP que tengan la extensión .xml. Valor predeterminado: false
Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en true.

- schema-imports

--schema-imports = load-by-schemalocation | load-preferring-schemalocation | load-by-namespace | load-combining-both | license-namespace-only
Esta opción indica el comportamiento de los elementos xs:import. Cada uno de estos elementos tiene un atributo opcional namespace y un atributo opcional schemaLocation: <import namespace="unEspacioNombres" schemaLocation="unaURL">. La opción indica si se debe cargar un documento de esquema o solo autorizar a un espacio de nombres. Si la opción indica que se debe cargar un documento de esquema, entonces indica también qué información debe utilizarse para encontrar el documento de esquema. Valor predeterminado: load-preferring-schemalocation.

- load-by-schemalocation: el valor del atributo schemaLocation se utiliza para buscar el esquema, teniendo en cuenta las asignaciones de catálogo. Si está presente el atributo namespace, se importa el espacio de nombres (con licencia).
- load-preferring-schemalocation: si está presente, se utiliza el atributo schemaLocation teniendo en cuenta las asignaciones de catálogo. Si no está presente el atributo schemaLocation, entonces se usa el valor del atributo namespace a través de las asignaciones de catálogo. Este es el **valor predeterminado**.
- load-by-namespace: el valor del atributo namespace se utiliza para buscar el esquema por medio de una asignación de catálogo.
- load-combining-both: si el atributo namespace o schemaLocation tiene una asignación de catálogo, entonces se usa la asignación. Si ambos atributos tienen asignaciones de catálogo, entonces es el valor de la opción --schema-mapping (opción XBRL y opción XML/XSD) decide qué asignación se utiliza. Si no hay ninguna asignación de catálogo, entonces se usa el atributo schemaLocation.
- license-namespace-only: se importa el espacio de nombres. No se importa el

documento de esquema.

- schemalocation-hints

  --schemalocation-hints = `load-by-schemalocation | load-by-namespace | load-combining-both | ignore`
  Determina el comportamiento predeterminado de los atributos `xsi:schemaLocation` y `xsi:noNamespaceSchemaLocation`. Indica si se debe cargar un documento de esquema y, si así es, indica qué información debe utilizarse para encontrarlo. Valor predeterminado: `load-by-schemalocation`.

  - **Valor predeterminado**: `load-by-schemalocation`. Este valor toma la URL de la ubicación del esquema de los atributos `xsi:schemaLocation` y `xsi:noNamespaceSchemaLocation` de los documentos de instancia XML o XBRL.
  - El valor `load-by-namespace` toma la parte de espacio de nombres del atributo `xsi:schemaLocation` y una cadena vacía en el caso del atributo `xsi:noNamespaceSchemaLocation` y encuentra el esquema por medio de una asignación de catálogo.
  - Si usa el valor `load-combining-both` y el espacio de nombres o la URL tienen una asignación de catálogo, se usa dicha asignación. Si ambos tienen asignaciones de catálogo, el valor de la opción `schema-mapping` (opción XBRL y opción XML/XSD) decide qué asignación se utiliza. Si ni el espacio de nombres ni la URL tiene una asignación de catálogo, se usa la URL.
  - El valor `ignore` ignora los atributos `xsi:schemaLocation` y `xsi:noNamespaceSchemaLocation`.

- schema-mapping

  --schema-mapping = `prefer-schemalocation | prefer-namespace`
  Si se usa la ubicación y el espacio de nombres para buscar el documento de esquema, esta opción indica cuál de ellos debe ser la opción preferida durante la búsqueda en el catálogo.
  Si la opción `--schemalocation-hints` o la opción `--schema-imports` tiene el valor `load-combining-both` y si las partes de espacio de nombres y URL pertinentes tienen asignaciones de catálogo, entonces el valor de la opción especifica cuál de las dos asignaciones se utiliza (la asignación del espacio de nombres o de la URL: el valor `prefer-schemalocation` se refiere a la asignación de la URL). Valor predeterminado: `prefer-schemalocation`.

- script

  --script = *ARCHIVO*
  Una vez finalizada la validación, ejecuta el script Python. Para indicar más de un script basta con agregar la opción varias veces.

- script-api-version

  --api, --script-api-version = `1|2|2.1|2.2|2.3|2.4|2.4.1|2.5`
  Especifica la versión API Python que ha de usarse para el script. El valor por defecto es la última versión, actualmente **`2.45`**. En vez de de los valores `1` y `2`, también puede usar losa valores 1.0 y 2.0 respectivamente.

- script-output

  --script-output = *FILE*
  Escribe el resultado estandard del script en el archivo nombres como *FILE*.

- script-param

  --script-param = *CLAVE:VALOR*
  Parámetros definidos por el usuario a los que se puede acceder durante la ejecución de scripts Python.

- xinclude

  --xinclude = `true|false`
  Habilita la compatibilidad con inclusiones XML (XInclude). Si el valor es `false`, los elementos XInclude `include` se ignoran. Valor predeterminado: `false`.
  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- xml-mode

  --xml-mode = `wf|id|valid`
  Especifica el modo de procesamiento XML que debe utilizarse para el documento de instancia XML: `wf`=comprobación de formato; `id`=comprobación de formato con ID/IDREF; `valid`=validación. Valor predeterminado: `wf`. Recuerde que el valor `valid` exige que cada documento de instancia que se cargue durante el procesamiento haga referencia a una DTD. Si no existe ninguna DTD, se generará un error.

- xml-mode-for-schemas

  --xml-mode-for-schemas = `wf|id|valid`
  Especifica el modo de procesamiento XML que debe utilizarse para el documento de instancia XML: `wf`=comprobación de formato; `id`=comprobación de formato con ID/IDREF; `valid`=validación. Valor predeterminado: `wf`. Recuerde que el valor `valid` exige que cada documento de esquema que se cargue durante el procesamiento haga referencia a una DTD. Si no existe ninguna DTD, se generará un error.

- xsd-version

  --xsd-version = `1.0|1.1|detect`

  Especifica qué versión de la especificación Schema Definition Language (XSD) del W3C se debe usar. Valor predeterminado: `1.0`.

  Esta opción también puede ser útil si quiere ver en qué aspectos no es compatible un esquema 1.0 con la especificación 1.1. El valor `detect` es una característica de Altova. Permite detectar la versión del esquema XML (`1.0` o `1.1`) leyendo el valor del atributo `vc:minVersion` del elemento `<xs:schema>` del documento. Si el valor del atributo `@vc:minVersion` es `1.1`, se entiende que la versión del esquema es `1.1`. Si el atributo tiene otro valor que no sea `1.1` (o si no está presente el atributo `@vc:minVersion`), se entiende que la versión del esquema es `1.0`.

- Catálogos y recursos globales

- catalog

  --catalog = *ARCHIVO*
  Especifica la ruta de acceso absoluta a un archivo de catálogo que no está en el archivo de catálogo raíz instalado. El valor predeterminado es la ruta de acceso absoluta del archivo de catálogo raíz instalado (`<carpeta-instalación>\Altova\RaptorXMLXBRLServer2018\etc\RootCatalog.xml`). Consulte el apartado Catálogos XML para obtener más información.

- user-catalog

  --user-catalog = *ARCHIVO*
  Especifica la ruta de acceso absoluta a un catálogo XML que debe utilizarse junto con el catálogo raíz. Consulte el apartado Catálogos XML para obtener más información.

- enable-globalresources

  --enable-globalresources = `true|false`
  Habilita la función de recursos globales. Valor predeterminado: `false`.
  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- globalresourceconfig [gc]

  --gc | --globalresourceconfig = *VALOR*
  Especifica la configuración activa del recurso global (y habilita los recursos globales).

- globalresourcefile [gr]

  --gr | --globalresourcefile = *ARCHIVO*
  Especifica el archivo de recursos globales (y habilita los recursos globales).

- Mensajes, errores, ayuda, tiempo de espera, versión
  - error-format

    --error-format = `text|shortxml|longxml`
    Especifica el formato de la salida de error. Los valores posibles son formatos de texto, XML y XML detallado (`longxml`). Valor predeterminado: `text`.

  - error-limit

    --error-limit = *N*
    Especifica el límite de errores con un valor comprendido entre `1` y `999`. El valor predeterminado es `100`. Cuando se alcanza el límite de error, se detiene la validación. Esta opción es muy práctica a la hora de limitar el uso del procesador durante la validación o transformación.

  - help

    --help
    Muestra el texto de ayuda para el comando. Por ejemplo `valany --h`. (Otra opción es usar el comando `help` con un argumento. Por ejemplo: `help valany`).

  - log-output

--log-output = *ARCHIVO*
Escribe el registro de salida en la URL de archivo indicada. Compruebe que la ILC tiene permiso de escritura en la ubicación de destino.

- network-timeout

--network-timeout = *VALOR*
Indica el tiempo de espera en segundos para operaciones remotas de entrada y salida. Valor predeterminado: `40`.

- verbose

--verbose = `true|false`
Si el valor es `true`, se genera información adicional durante la validación. Valor predeterminado es `false`.
Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- verbose-output

--verbose-output = *ARCHIVO*
Escribe el resultado detallado en el *ARCHIVO* indicado.

- version

--version
Muestra el número de versión de RaptorXML+XBRL Server. Si se utiliza con un comando, escriba la opción `--version` antes del comando.

- warning-limit

--warning-limit = *VALUE*
Especifica el límite de advertencia en el rango `1-65535`. El procesamiento continua si se alcanza el límite pero no se registrarán más advertencias. El valor por defecto es `100`.

 © 2018 Altova GmbH

3.2 Comandos para comprobar el formato

Los comandos de comprobación de formato sirven para comprobar si el formato de documentos XML y DTD es correcto.

wfxml	Comprueba si el documento XML tiene un formato correcto.
wfdtd	Comprueba si el documento DTD tiene un formato correcto.
wfany	Comprueba si el documento XML o DTD tiene un formato correcto. El tipo se detecta automáticamente.

3.2.1 wfxml

El comando **wfxml** revisa uno o varios documentos XML y comprueba si su formato es correcto según la especificación XML 1.0 o XML 1.1.

Windows **RaptorXMLXBRL wfxml [opciones] *ArchivoEntrada***

Linux **raptorxmlxbrl wfxml [opciones] *ArchivoEntrada***

Mac **raptorxmlxbrl wfxml [opciones] *ArchivoEntrada***

El argumento *ArchivoEntrada* es el documento XML cuyo formato debe comprobarse.

Si desea comprobar el formato de varios documentos, tiene dos opciones: (i) enumerar los archivos que se deben revisar en la línea de comandos, separados por espacios o (ii) enumerar los archivos que se deben revisar en un archivo de texto (`.txt`) donde aparece un nombre de archivo por línea y dar este archivo de texto como argumento *ArchivoEntrada* junto con la opción `--listfile` con valor `true` (*ver lista de opciones más abajo*).

Ejemplos

- **raptorxmlxbrl** `wfxml c:\Test.xml`
- **raptorxmlxbrl** `wfxml --verbose=true c:\Test.xml`
- **raptorxmlxbrl** `wfxml --listfile=true c:\ListaArchivos.txt`

Uso de mayúsculas/minúsculas y de barras diagonales en la línea de comandos

RaptorXMLXBRL *en Windows*
raptorxmlxbrl *en Unix (Linux, Mac)*

* Las minúsculas (`raptorxmlxbrl`) funcionan en todas las plataformas (Windows, Linux y Mac), mientras que la mezcla de mayúsculas y minúsculas (`RaptorXMLXBRL`) sólo funciona en Windows y Mac.

* En Linux y Mac utilice barras diagonales. En Windows utilice barras diagonales inversas.

Opciones

Las opciones del comando aparecen más abajo divididas en grupos. Los valores se pueden dar sin comillas excepto en estos dos casos: (i) cuando la cadena de valor contiene espacios y (ii) cuando en la descripción de la opción se indique explícitamente que es necesario el uso de comillas.

- Validación y procesamiento
  - dtd

    `--dtd =` *`ARCHIVO`*

    Especifica el documento DTD externo que debe utilizarse para la validación. Si en el documento XML hay una referencia a una DTD externa, esta opción de la ILC reemplaza a la referencia externa.

  - listfile

    `--listfile =` `true|false`

    Si el valor es `true`, el argumento *`ArchivoEntrada`* del comando se entiende como un archivo de texto que contiene un nombre de archivo por línea. Otra opción es enumerar los archivos en la ILC, separados por un espacio. No obstante, recuerde que las ILC tienen un límite de caracteres. Además, no olvide que la opción `--listfile` solamente afecta a los argumentos y no a las opciones.
    Valor predeterminado: `false`.
    Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

  - namespaces

    `--namespaces =` `true|false`

    Habilita el procesamiento preparado para espacios de nombres. Esta opción es muy útil si quiere buscar en la instancia XML errores resultantes de espacios de nombres erróneos.
    Valor predeterminado: `false`.
    Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

  - recurse

    `--recurse =` `true|false`

    Esta opción se utiliza para seleccionar ficheros dentro de subdirectorios (incluso en archivos ZIP). Si el valor es `true`, el argumento *`ArchivoEntrada`* del comando seleccionará el fichero seleccionado también en los subdirectorios. Por ejemplo: `test.zip|zip\test.xml` seleccionará los ficheros llamados `test.xml` en todos los subdirectorios de la carpeta ZIP. Si quiere puede usar los caracteres comodín `*` y `?`. Por ejemplo: `*.xml` seleccionaría todos los ficheros de la carpeta ZIP que tengan la extensión `.xml`. Valor predeterminado: `false`
    Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

 © 2018 Altova GmbH

- streaming

  --streaming = `true|false`
  Habilita la transmisión por secuencias. En el modo de transmisión por secuencias, el almacenamiento de datos en memoria se reduce al mínimo y el procesamiento es más rápido. El inconveniente es que puede que no esté disponible cierta información que podría necesitar más adelante, como el modelo de datos del documento XML, por ejemplo. Si quiere evitar esto, debería deshabilitar el modo de transmisión por secuencias (dándole el valor `false` a la opción `--streaming`). Cuando use la opción `--script` con el comando `valxml-withxsd`, aconsejamos deshabilitar la transmisión por secuencias. Recuerde que la opción `--streaming` se ignora, si el valor de `--parallel-assessment` es `true`.
  Valor predeterminado: `true`.
  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- Catálogos y recursos globales
  - catalog

    --catalog = *ARCHIVO*
    Especifica la ruta de acceso absoluta a un archivo de catálogo que no está en el archivo de catálogo raíz instalado. El valor predeterminado es la ruta de acceso absoluta del archivo de catálogo raíz instalado (`<carpeta-instalación>\Altova\RaptorXMLXBRLServer2018\etc\RootCatalog.xml`). Consulte el apartado Catálogos XML para obtener más información.

  - user-catalog

    --user-catalog = *ARCHIVO*
    Especifica la ruta de acceso absoluta a un catálogo XML que debe utilizarse junto con el catálogo raíz. Consulte el apartado Catálogos XML para obtener más información.

  - enable-globalresources

    --enable-globalresources = `true|false`
    Habilita la función de recursos globales. Valor predeterminado: `false`.
    Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

  - globalresourceconfig [gc]

    --gc | --globalresourceconfig = *VALOR*
    Especifica la configuración activa del recurso global (y habilita los recursos globales).

  - globalresourcefile [gr]

    --gr | --globalresourcefile = *ARCHIVO*
    Especifica el archivo de recursos globales (y habilita los recursos globales).

- Mensajes, errores, ayuda, tiempo de espera, versión
  - error-format

--error-format = `text|shortxml|longxml`
Especifica el formato de la salida de error. Los valores posibles son formatos de texto, XML y XML detallado (`longxml`). Valor predeterminado: `text`.

- error-limit

--error-limit = *N*
Especifica el límite de errores con un valor comprendido entre `1` y `999`. El valor predeterminado es `100`. Cuando se alcanza el límite de error, se detiene la validación. Esta opción es muy práctica a la hora de limitar el uso del procesador durante la validación o transformación.

- help

--help
Muestra el texto de ayuda para el comando. Por ejemplo `valany --h`. (Otra opción es usar el comando `help` con un argumento. Por ejemplo: `help valany`).

- log-output

--log-output = *ARCHIVO*
Escribe el registro de salida en la URL de archivo indicada. Compruebe que la ILC tiene permiso de escritura en la ubicación de destino.

- network-timeout

--network-timeout = *VALOR*
Indica el tiempo de espera en segundos para operaciones remotas de entrada y salida. Valor predeterminado: `40`.

- verbose

--verbose = `true|false`
Si el valor es `true`, se genera información adicional durante la validación. Valor predeterminado es `false`.
Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- verbose-output

--verbose-output = *ARCHIVO*
Escribe el resultado detallado en el *ARCHIVO* indicado.

- version

--version
Muestra el número de versión de RaptorXML+XBRL Server. Si se utiliza con un comando, escriba la opción `--version` antes del comando.

- warning-limit

--warning-limit = *VALUE*
Especifica el límite de advertencia en el rango `1-65535`. El procesamiento continua si se alcanza el límite pero no se registrarán más advertencias. El valor por defecto es `100`.

 © 2018 Altova GmbH

3.2.2 wfdtd

El comando **wfdtd** revisa uno o varios documentos DTD y comprueba si su formato es correcto según la especificación XML 1.0 o XML 1.1.

Windows	**RaptorXMLXBRL wfdtd [opciones] *ArchivoEntrada***
Linux	**raptorxmlxbrl wfdtd [opciones] *ArchivoEntrada***
Mac	**raptorxmlxbrl wfdtd [opciones] *ArchivoEntrada***

El argumento *ArchivoEntrada* es el documento DTD cuyo formato debe comprobarse.

Si desea comprobar el formato de varios documentos, tiene dos opciones: (i) enumerar los archivos que se deben revisar en la línea de comandos, separados por espacios o (ii) enumerar los archivos que se deben revisar en un archivo de texto (`.txt`) donde aparece un nombre de archivo por línea y dar este archivo de texto como argumento *ArchivoEntrada* junto con la opción `--listfile` con valor `true` (*ver lista de opciones más abajo*).

Ejemplos

- **raptorxmlxbrl** `wfdtd c:\Test.dtd`
- **raptorxmlxbrl** `wfdtd --verbose=true c:\Test.dtd`
- **raptorxmlxbrl** `wfdtd --listfile=true c:\ListaArchivos.txt`

Uso de mayúsculas/minúsculas y de barras diagonales en la línea de comandos

RaptorXMLXBRL *en Windows*
raptorxmlxbrl *en Unix (Linux, Mac)*

* Las minúsculas (`raptorxmlxbrl`) funcionan en todas las plataformas (Windows, Linux y Mac), mientras que la mezcla de mayúsculas y minúsculas (`RaptorXMLXBRL`) sólo funciona en Windows y Mac.
* En Linux y Mac utilice barras diagonales. En Windows utilice barras diagonales inversas.

Opciones

Las opciones del comando aparecen más abajo divididas en grupos. Los valores se pueden dar sin comillas excepto en estos dos casos: (i) cuando la cadena de valor contiene espacios y (ii) cuando en la descripción de la opción se indique explícitamente que es necesario el uso de comillas.

Validación y procesamiento

listfile

--listfile = true|false
Si el valor es true, el argumento *ArchivoEntrada* del comando se entiende como un archivo de texto que contiene un nombre de archivo por línea. Otra opción es enumerar los archivos en la ILC, separados por un espacio. No obstante, recuerde que las ILC tienen un límite de caracteres. Además, no olvide que la opción --listfile solamente afecta a los argumentos y no a las opciones.
Valor predeterminado: false.
Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en true.

- recurse

--recurse = true|false
Esta opción se utiliza para seleccionar ficheros dentro de subdirectorios (incluso en archivos ZIP). Si el valor es true, el argumento *ArchivoEntrada* del comando seleccionará el fichero seleccionado también en los subdirectorios. Por ejemplo: test.zip|zip\test.xml seleccionará los ficheros llamados test.xml en todos los subdirectorios de la carpeta ZIP. Si quiere puede usar los caracteres comodín * y ?. Por ejemplo: *.xml seleccionaría todos los ficheros de la carpeta ZIP que tengan la extensión .xml. Valor predeterminado: false
Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en true.

- Catálogos y recursos globales
  - catalog

--catalog = *ARCHIVO*
Especifica la ruta de acceso absoluta a un archivo de catálogo que no está en el archivo de catálogo raíz instalado. El valor predeterminado es la ruta de acceso absoluta del archivo de catálogo raíz instalado (<carpeta-instalación>\Altova\RaptorXMLXBRLServer2018\etc\RootCatalog.xml). Consulte el apartado Catálogos XML para obtener más información.

  - user-catalog

--user-catalog = *ARCHIVO*
Especifica la ruta de acceso absoluta a un catálogo XML que debe utilizarse junto con el catálogo raíz. Consulte el apartado Catálogos XML para obtener más información.

  - enable-globalresources

--enable-globalresources = true|false
Habilita la función de recursos globales. Valor predeterminado: false.
Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en true.

  - globalresourceconfig [gc]

--gc | --globalresourceconfig = *VALOR*
Especifica la configuración activa del recurso global (y habilita los recursos globales).

  - globalresourcefile [gr]

--gr | --globalresourcefile = *ARCHIVO*
Especifica el archivo de recursos globales (y habilita los recursos globales).

- Mensajes, errores, ayuda, tiempo de espera, versión
  - error-format

    --error-format = `text|shortxml|longxml`
    Especifica el formato de la salida de error. Los valores posibles son formatos de texto, XML y XML detallado (`longxml`). Valor predeterminado: `text`.

  - error-limit

    --error-limit = *N*
    Especifica el límite de errores con un valor comprendido entre `1` y `999`. El valor predeterminado es `100`. Cuando se alcanza el límite de error, se detiene la validación. Esta opción es muy práctica a la hora de limitar el uso del procesador durante la validación o transformación.

  - help

    --help
    Muestra el texto de ayuda para el comando. Por ejemplo `valany --h`. (Otra opción es usar el comando `help` con un argumento. Por ejemplo: `help valany`).

  - log-output

    --log-output = *ARCHIVO*
    Escribe el registro de salida en la URL de archivo indicada. Compruebe que la ILC tiene permiso de escritura en la ubicación de destino.

  - network-timeout

    --network-timeout = *VALOR*
    Indica el tiempo de espera en segundos para operaciones remotas de entrada y salida. Valor predeterminado: `40`.

  - verbose

    --verbose = `true|false`
    Si el valor es `true`, se genera información adicional durante la validación. Valor predeterminado es `false`.
    Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

  - verbose-output

    --verbose-output = *ARCHIVO*
    Escribe el resultado detallado en el *ARCHIVO* indicado.

  - version

    --version
    Muestra el número de versión de RaptorXML+XBRL Server. Si se utiliza con un

comando, escriba la opción `--version` antes del comando.

- warning-limit

  `--warning-limit =` *`VALUE`*
  Especifica el límite de advertencia en el rango `1-65535`. El procesamiento continua si se alcanza el límite pero no se registrarán más advertencias. El valor por defecto es `100`.

3.2.3 wfany

El comando **`wfany`** comprueba si el formato de un documento XML o DTD es correcto según la especificación correspondiente. El tipo de documento se detecta automáticamente.

Windows	**`RaptorXMLXBRL wfany [opciones]`** ***`ArchivoEntrada`***
Linux	**`raptorxmlxbrl wfany [opciones]`** ***`ArchivoEntrada`***
Mac	**`raptorxmlxbrl wfany [opciones]`** ***`ArchivoEntrada`***

El argumento *`ArchivoEntrada`* es el documento cuyo formato debe comprobarse. Recuerde que como argumento del comando se puede suministrar solamente un documento. El tipo de documento se detecta automáticamente.

Ejemplos

- **`raptorxmlxbrl`** `wfany c:\Test.xml`
- **`raptorxmlxbrl`** `wfany --errorformat=text c:\Test.xml`

- Uso de mayúsculas/minúsculas y de barras diagonales en la línea de comandos

  `RaptorXMLXBRL` *en Windows*
  `raptorxmlxbrl` *en Unix (Linux, Mac)*

  * Las minúsculas (`raptorxmlxbrl`) funcionan en todas las plataformas (Windows, Linux y Mac), mientras que la mezcla de mayúsculas y minúsculas (`RaptorXMLXBRL`) sólo funciona en Windows y Mac.
  * En Linux y Mac utilice barras diagonales. En Windows utilice barras diagonales inversas.

Opciones

Las opciones del comando aparecen más abajo divididas en grupos. Los valores se pueden dar sin comillas excepto en estos dos casos: (i) cuando la cadena de valor contiene espacios y (ii) cuando en la descripción de la opción se indique explícitamente que es necesario el uso de comillas.

- Catálogos y recursos globales

 © 2018 Altova GmbH

- catalog

  --catalog = *ARCHIVO*
  Especifica la ruta de acceso absoluta a un archivo de catálogo que no está en el archivo de catálogo raíz instalado. El valor predeterminado es la ruta de acceso absoluta del archivo de catálogo raíz instalado (`<carpeta-instalación>\Altova\RaptorXMLXBRLServer2018\etc\RootCatalog.xml`). Consulte el apartado Catálogos XML para obtener más información.

- user-catalog

  --user-catalog = *ARCHIVO*
  Especifica la ruta de acceso absoluta a un catálogo XML que debe utilizarse junto con el catálogo raíz. Consulte el apartado Catálogos XML para obtener más información.

- enable-globalresources

  --enable-globalresources = `true|false`
  Habilita la función de recursos globales. Valor predeterminado: `false`.
  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- globalresourceconfig [gc]

  --gc | --globalresourceconfig = *VALOR*
  Especifica la configuración activa del recurso global (y habilita los recursos globales).

- globalresourcefile [gr]

  --gr | --globalresourcefile = *ARCHIVO*
  Especifica el archivo de recursos globales (y habilita los recursos globales).

- Mensajes, errores, ayuda, tiempo de espera, versión
  - error-format

    --error-format = `text|shortxml|longxml`
    Especifica el formato de la salida de error. Los valores posibles son formatos de texto, XML y XML detallado (`longxml`). Valor predeterminado: `text`.

  - error-limit

    --error-limit = *N*
    Especifica el límite de errores con un valor comprendido entre `1` y `999`. El valor predeterminado es `100`. Cuando se alcanza el límite de error, se detiene la validación. Esta opción es muy práctica a la hora de limitar el uso del procesador durante la validación o transformación.

  - help

    --help
    Muestra el texto de ayuda para el comando. Por ejemplo `valany --h`. (Otra opción es usar el comando `help` con un argumento. Por ejemplo: `help valany`).

  - log-output

--log-output = *ARCHIVO*
Escribe el registro de salida en la URL de archivo indicada. Compruebe que la ILC tiene permiso de escritura en la ubicación de destino.

- network-timeout

--network-timeout = *VALOR*
Indica el tiempo de espera en segundos para operaciones remotas de entrada y salida. Valor predeterminado: `40`.

- verbose

--verbose = `true|false`
Si el valor es `true`, se genera información adicional durante la validación. Valor predeterminado es `false`.
Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- verbose-output

--verbose-output = *ARCHIVO*
Escribe el resultado detallado en el *ARCHIVO* indicado.

- version

--version
Muestra el número de versión de RaptorXML+XBRL Server. Si se utiliza con un comando, escriba la opción `--version` antes del comando.

- warning-limit

--warning-limit = *VALUE*
Especifica el límite de advertencia en el rango `1-65535`. El procesamiento continua si se alcanza el límite pero no se registrarán más advertencias. El valor por defecto es `100`.

 © 2018 Altova GmbH

3.3 Comandos de validación XBRL

Los comandos de validación XBRL sirven para validar documentos de instancia y taxonomías XBRL con las especificaciones XBRL 2.1, Dimensions 1.0 y Formula 1.0.

valxbrl | xbrl	Valida un documento de instancia XBRL (extensión de archivo `.xbrl`).
valinlinexbrl (ixbrl)	Valida documentos Inline XBRL (iXBRL) de acuerdo con lo estipulado por la especificación Inline XBRL 1.0 o Inline XBRL 1.1.
valxbrltaxonomy | dts	Valida un documento de taxonomía (esquema) XBRL (extensión de archivo `.xsd`).
valtaxonomypackage | taxpkg	Valida paquetes de taxonomías XBRL de acuerdo con lo estipulado por la especificación Taxonomy 1.0 Packages.
valxbrlver	Valida informes de versionado XBRL de acuerdo con lo estipulado por la especificación XBRL Versioning 1.0.

3.3.1 valxbrl (xbrl)

El comando **`valxbrl | xbrl`** valida uno o varios documentos de instancia XBRL con las especificaciones XBRL 2.1, Dimensions 1.0 y Formula 1.0.

Windows **`RaptorXMLXBRL valxbrl | xbrl [opciones] ArchivoEntrada`**

Linux **`raptorxmlxbrl valxbrl | xbrl [opciones] ArchivoEntrada`**

Mac **`raptorxmlxbrl valxbrl | xbrl [opciones] ArchivoEntrada`**

El argumento *`ArchivoEntrada`* es el documento de instancia XBRL que debe validarse.

Si desea validar varios documentos, tiene dos opciones: (i) enumerar los archivos que se deben validar en la línea de comandos, separados por espacios o (ii) enumerar los archivos que se deben validar en un archivo de texto (`.txt`) donde aparece un nombre de archivo por línea y dar este archivo de texto como argumento *`ArchivoEntrada`* junto con la opción `--listfile` con valor `true` (*ver lista de opciones más abajo*).

Nota: el documento de instancia XBRL no puede estar anidado en otro documento XML y su elemento raíz debe ser el elemento `xbrl`.

```
<xbrl xmlns="http://www.xbrl.org/2003/instance"> ... </xbrl>
```

Validación EDGAR

El sistema EDGAR de recopilación, análisis y recuperación de datos electrónicos es un sistema que recopila, valida e indexa automáticamente los informes financieros presentados por empresas a la Comisión de bolsa y valores estadounidense (SEC). raptorxmlxbrl ofrece funciones de validación con EDGAR a través de la API de Python. Para ejecutar una validación EDGAR en un archivo de instancia XBRL, utilice la opción `--script` para ejecutar el script Python de validación EDGAR. Este script (**`efm-validation.py`**) está ubicado en la carpeta `etc\scripts\sec-edgar-tools` de la carpeta de la aplicación:

```
valxbrl --script="C:\Archivos de programa\Altova\RaptorXMLXBRLServer2018\etc
\scripts\sec-edgar-tools\efm-validation.py" myinstance.xbrl
```

Ejemplos

- **raptorxmlxbrl** `valxbrl c:\Test.xbrl`
- **raptorxmlxbrl** `xbrl --formula-execution=true --formula-output=c:\FormulaSalida.xml c:\Test.xbrl`
- **raptorxmlxbrl** `xbrl --formula-execution --assertions-output=c:\AssertionsSalida.xml c:\Test.xbrl`
- **raptorxmlxbrl** `xbrl --formula-execution --formula-output=c:\FormulaSalida.xml --assertions-output=c:\AssertionsSalida.xml c:\Test.xbrl`

Uso de mayúsculas/minúsculas y de barras diagonales en la línea de comandos

`RaptorXMLXBRL` *en Windows*
`raptorxmlxbrl` *en Unix (Linux, Mac)*

* Las minúsculas (`raptorxmlxbrl`) funcionan en todas las plataformas (Windows, Linux y Mac), mientras que la mezcla de mayúsculas y minúsculas (`RaptorXMLXBRL`) sólo funciona en Windows y Mac.
* En Linux y Mac utilice barras diagonales. En Windows utilice barras diagonales inversas.

Opciones

Las opciones del comando aparecen más abajo divididas en grupos. Los valores se pueden dar sin comillas excepto en estos dos casos: (i) cuando la cadena de valor contiene espacios y (ii) cuando en la descripción de la opción se indique explícitamente que es necesario el uso de comillas.

Validación y procesamiento XBRL

additional-dts-entry-point

`--additional-dts-entry-point =` *`FILE`*
Fuerza la inclusión del archivo de punto de entrada en el proceso de detección de DTS. Añada la opción varias veces para especificar más de un punto de entrada.

ct-restrict-mode

`--ct-restrict-mode =` `1.0|1.1|default`

 © 2018 Altova GmbH

Especifica cómo comprobar restricciones de tipo complejo. Un valor de `1.0` comprueba restricciones de tipo complejo conforme a lo definido en la especificación XSD 1.0 (incluso estando en modo de validación XSD 1.1). Un valor de `1.1` comprueba restricciones de tipo complejo conforme a los definido en la especificación XSD 1.1 (incluso estando en modo de validación XSD 1.0). Un valor de **`default`** comprueba restricciones de tipo complejo conforme a lo definido en la especificación XSD del modo de validación habilitado en ese momento (1.0 o 1.1). El valor predeterminado es **`default`**.

- dimensions

  `--dimensions =` `true|false`
  Habilita las extensiones XBRL Dimension 1.0. Valor predeterminado: `true`.
  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- dts

  `--dts =` *`FILE`*
  Recarga el punto de entrada DTS determinado en *`ARCHIVO`* y lo usa para la validación de archivos de instancia. Agregue la opción de varias veces para especificar más de un punto de entrada. Los puntos de entrada en los archivos de instancia se ignorarán. El comando se usa para validar por lotes varios archivos de instancia que hacen referencia a la misma taxonomía. La opción `--dts` da la instrucción al motor Raptor para que cargar el DTS en un sólo momento - y no para cada instancia en la lista de archivo. (Se genera un archivo o bien en el CLI o en un archivo de texto que se indica como lista de archivo configurando la opción `--listfile` en `true`.). Si un archivo de instancia en la lista de archivo hace referencia a una taxonomía diferente, entonces se genera un aviso. Este comando puede acelerar enormemente el tiempo de validación cuando se validan varios archivos de instancia de pequeño tamaño cada uno de los cuales posea un DTS de gran compatibilidad: no proporciona ventaja cuando se valida un archivo de instancia único.

- essence-alias-checks

  `--essence-alias-checks =` `true|false`
  Habilita la creación de informes sobre incoherencias relacionadas con essence-alias. El valor predeterminado es `true`.
  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- extensible-enumerations

  `--extensible-enumerations =` `true|false`
  Si es `true`, habilita la extensión Enumeraciones extensibles XBRL 1.0. Por defecto es: `true`.

- inconsistencies-limit

  `--inconsistencies-limit =` *`VALUE`*
  Especifica el límite de las inconsistencias XBRL con valores que se encuentran en el rango `1-65535`. El procesamiento continua si se alcanza el límite, pero no se registran el resto de las incoherencias. El valor por defecto es `100`.

- listfile

  --listfile = true|false

  Si el valor es true, el argumento *ArchivoEntrada* del comando se entiende como un archivo de texto que contiene un nombre de archivo por línea. Otra opción es enumerar los archivos en la ILC, separados por un espacio. No obstante, recuerde que las ILC tienen un límite de caracteres. Además, no olvide que la opción --listfile solamente afecta a los argumentos y no a las opciones.
  Valor predeterminado: false.
  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en true.

- parallel-assessment [pa]

  --pa | --parallel-assessment = true|false

  Si el valor es true, la evaluación de la validez de esquemas se realiza en paralelo. Esto significa que si hay más de 128 elementos en cualquiera de los niveles, estos elementos se procesan en paralelo utilizando varios subprocesos. Por tanto, los archivos XML de gran tamaño se pueden procesar más rápido si se habilita esta opción. La evaluación en paralelo se lleva cabo nivel por nivel, pero puede ocurrir en varios niveles de un mismo conjunto de información. Recuerde que la evaluación en paralelo no funciona en modo de transmisión por secuencias. Por este motivo la opción --streaming se pasa por alto si el valor de la opción --parallel-assessment es true. Además, se usa más memoria cuando se utiliza la opción --parallel-assessment. El valor predeterminado de esta opción es false y su forma abreviada es --pa.
  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en true.

- preload-xbrl-schemas

  --preload-xbrl-schemas = true|false

  Carga previa de los esquemas de la especificación XBRL 2.1. Valor predeterminado: true.
  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en true.

- recurse

  --recurse = true|false

  Esta opción se utiliza para seleccionar ficheros dentro de subdirectorios (incluso en archivos ZIP). Si el valor es true, el argumento *ArchivoEntrada* del comando seleccionará el fichero seleccionado también en los subdirectorios. Por ejemplo: test.zip|zip\test.xml seleccionará los ficheros llamados test.xml en todos los subdirectorios de la carpeta ZIP. Si quiere puede usar los caracteres comodín * y ?. Por ejemplo: *.xml seleccionaría todos los ficheros de la carpeta ZIP que tengan la extensión .xml. Valor predeterminado: false
  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en true.

- requires-element-checks

  --requires-element-checks = true|false

  Habilita la creación de informes sobre incoherencias relacionadas con requires-element.

El valor predeterminado es `true`.
Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

schema-imports

`--schema-imports =` `load-by-schemalocation | load-preferring-schemalocation | load-by-namespace | load-combining-both | license-namespace-only`

Esta opción indica el comportamiento de los elementos `xs:import`. Cada uno de estos elementos tiene un atributo opcional `namespace` y un atributo opcional `schemaLocation`: `<import namespace="unEspacioNombres" schemaLocation="unaURL">`. La opción indica si se debe cargar un documento de esquema o solo autorizar a un espacio de nombres. Si la opción indica que se debe cargar un documento de esquema, entonces indica también qué información debe utilizarse para encontrar el documento de esquema. Valor predeterminado: `load-preferring-schemalocation`.

- `load-by-schemalocation`: el valor del atributo `schemaLocation` se utiliza para buscar el esquema, teniendo en cuenta las asignaciones de catálogo. Si está presente el atributo `namespace`, se importa el espacio de nombres (con licencia).
- `load-preferring-schemalocation`: si está presente, se utiliza el atributo `schemaLocation` teniendo en cuenta las asignaciones de catálogo. Si no está presente el atributo `schemaLocation`, entonces se usa el valor del atributo `namespace` a través de las asignaciones de catálogo. Este es el **valor predeterminado**.
- `load-by-namespace`: el valor del atributo `namespace` se utiliza para buscar el esquema por medio de una asignación de catálogo.
- `load-combining-both`: si el atributo `namespace` o `schemaLocation` tiene una asignación de catálogo, entonces se usa la asignación. Si ambos atributos tienen asignaciones de catálogo, entonces es el valor de la opción `--schema-mapping` (opción XBRL y opción XML/XSD) decide qué asignación se utiliza. Si no hay ninguna asignación de catálogo, entonces se usa el atributo `schemaLocation`.
- `license-namespace-only`: se importa el espacio de nombres. No se importa el documento de esquema.

schema-mapping

`--schema-mapping =` `prefer-schemalocation | prefer-namespace`

Si se usa la ubicación y el espacio de nombres para buscar el documento de esquema, esta opción indica cuál de ellos debe ser la opción preferida durante la búsqueda en el catálogo.
Si la opción `--schemalocation-hints` o la opción `--schema-imports` tiene el valor `load-combining-both` y si las partes de espacio de nombres y URL pertinentes tienen asignaciones de catálogo, entonces el valor de la opción especifica cuál de las dos asignaciones se utiliza (la asignación del espacio de nombres o de la URL: el valor `prefer-schemalocation` se refiere a la asignación de la URL). Valor predeterminado: `prefer-schemalocation`.

schemalocation-hints

`--schemalocation-hints =` `load-by-schemalocation | load-by-namespace | load-combining-both | ignore`

Determina el comportamiento predeterminado de los atributos `xsi:schemaLocation` y `xsi:noNamespaceSchemaLocation`. Indica si se debe cargar un documento de esquema y, si así es, indica qué información debe utilizarse para encontrarlo. Valor predeterminado: `load-by-schemalocation`.

- **Valor predeterminado**: `load-by-schemalocation`. Este valor toma la URL de la ubicación del esquema de los atributos `xsi:schemaLocation` y `xsi:noNamespaceSchemaLocation` de los documentos de instancia XML o XBRL.
- El valor `load-by-namespace` toma la parte de espacio de nombres del atributo `xsi:schemaLocation` y una cadena vacía en el caso del atributo `xsi:noNamespaceSchemaLocation` y encuentra el esquema por medio de una asignación de catálogo.
- Si usa el valor `load-combining-both` y el espacio de nombres o la URL tienen una asignación de catálogo, se usa dicha asignación. Si ambos tienen asignaciones de catálogo, el valor de la opción `schema-mapping` (opción XBRL y opción XML/XSD) decide qué asignación se utiliza. Si ni el espacio de nombres ni la URL tiene una asignación de catálogo, se usa la URL.
- El valor `ignore` ignora los atributos `xsi:schemaLocation` y `xsi:noNamespaceSchemaLocation`.

script

`--script =` *`ARCHIVO`*

Una vez finalizada la validación, ejecuta el script Python. Para indicar más de un script basta con agregar la opción varias veces.

script-api-version

`--api, --script-api-version =` `1|2|2.1|2.2|2.3|2.4|2.4.1|2.5`

Especifica la versión API Python que ha de usarse para el script. El valor por defecto es la última versión, actualmente **`2.45`**. En vez de de los valores `1` y `2`, también puede usar losa valores 1.0 y 2.0 respectivamente.

script-output

`--script-output =` *`FILE`*

Escribe el resultado estandard del script en el archivo nombres como *`FILE`*.

script-param

`--script-param =` *`CLAVE:VALOR`*

Parámetros definidos por el usuario a los que se puede acceder durante la ejecución de scripts Python.

summation-item-checks

`--summation-item-checks =` `true|false`

Habilita la creación de informes sobre incoherencias relacionadas con el ítem de suma. El valor predeterminado es `true`.

Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

 © 2018 Altova GmbH

- taxonomy-package

  --taxonomy-package = *FILE*

  Especifica la ruta absoluta al paquete de taxonomía tradicional tal y como se describe en los barrados de trabajo del Paquete de taxonomía 1.0 y los Paquetes de taxonomía 1.0. El valor del ARCHIVO devuelve la ubicación del paquete de la taxonomía. Agregue la opción *varias veces* para especificar más de un paquete de taxonomía.

- taxonomy-packages-config-file

  --taxonomy-packages-config-file = *ARCHIVO*

  Especifica la ruta de acceso del archivo **TaxonomyPackagesConfig.json**, que se utiliza pra cargar los paquetes de taxonomías XBRL. Es recomendable tener este archivo porque sirve como catálogo de los paquetes de taxonomías. La estructura del archivo JSON es como la del ejemplo que aparece más abajo. El valor de la clave **uri** indica la ubicación del paquete. La clave **active** (de un conjunto de paquetes) cambia cuando se usan dichos paquetes.

```
{"taxonomies":
  {"EIOPA Solvency II XBRL Taxonomy 2.1.0":
    {"packages":[
      {"uri":"C:\\test\\XBRL\
\EIOPA_SolvencyII_XBRL_Taxonomy_2.1.0.zip"},
      {"uri":"C:\\test\\XBRL\\AdditionalTestPkg.zip"}
      ],"active":true
    },"Test Taxonomy":
      {"packages":[{"uri":"C:\\test\\XBRL\\test.zip"}],"active":true
      }
  }
}
```

- treat-inconsistencies-as-errors

  --treat-inconsistencies-as-errors = `true|false`

  La validación XBRL falla si el archivo contiene incoherencias según lo definido por la especificación XBRL 2.1. El valor predeterminado es `false`.

  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- utr

  --utr = `true|false`

  Si es `true`, habilita la extensión Registro de unidad XBRL 1.0. Por defecto es: `false`.

- utr-file

  --utr-file = *FILE*

  Especifica la ruta de acceso del archivo del registro de unidades. El valor predeterminado es: **http://www.xbrl.org/utr/utr.xml**.

- utr-status

  --utr-status = `PWD|CR|REC|NIE|PROPOSED|IWD|ACK|RR`

  Habilita la compatibilidad con definiciones de unidad con el estado dado en el registro de unidades. Añada la opción varias veces para indicar más de un estado de unidad. El

valor predeterminado es **REC**.

- validate-dts-only

  --validate-dts-only = `true|false`
  La DTS se detecta iniciando desde el documento de instancia XBRL. Todos los esquemas de taxonomía y bases de enlace referenciados se detectan y validan. Los demás documentos de instancia XBRL se ignoran. Valor predeterminado: `false`.
  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- xinclude

  --xinclude = `true|false`
  Habilita la compatibilidad con inclusiones XML (XInclude). Si el valor es `false`, los elementos XInclude `include` se ignoran. Valor predeterminado: `false`.
  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- Fórmulas y aserciones XBRL
  - assertion-severity

    --assertion-severity = `true|false`
    Activa la extensión Assertion Severity 1.0. El valor por defecto es `true`.
    Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

  - assertions-output

    --assertions-output = *ARCHIVO*
    Escribe el resultado de la evaluación de la aserción en el *ARCHIVO* indicado. Si se define, especifica automáticamente `--formula-execution=true`.

  - assertions-output-format

    --assertions-output-format = `json|xml`
    Especifica el formato de salida de la evaluación de la aserción. Valor predeterminado: `json`.

  - evaluate-referenced-parameters-only

    --evaluate-referenced-parameters-only = `true|false`
    Si tiene valor `false`, obliga a evaluar todos los parámetros incluso si no se les hace referencia en ninguna fórmula, aserción o tabla. Valor predeterminado: `true`.

  - formula

    --formula = `true|false`
    Habilita la extensión XBRL Formula 1.0. Valor predeterminado: `true`.
    Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

  - formula-assertion-set [[OBSOLETA]]

--formula-assertion-set = *VALOR*
Limita la ejecución de fórmulas al conjunto de aserciones dado. Añada la opción varias veces para especificar más de un grupo de aserciones. Su forma abreviada es `--as`. El *VALOR* es el valor del atributo `@id` o un URI con un fragmento XPointer que identifica el recurso. También puede usar los valores especiales `##none` y `##all`.

- formula-execution

--formula-execution = `true|false`
Habilita la evaluación de fórmulas XBRL. Valor predeterminado: `true`. Si es `true`, especifica automáticamente `--formula=true`.
Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- formula-output

--formula-output = *ARCHIVO*
Escribe el resultado de la evaluación de la fórmula en el *ARCHIVO* indicado. Si se define, especifica automáticamente `--formula-execution=true`.

- formula-parameters

--formula-parameters = *MATRIZ-JSON*
Especifica parámetros para la evaluación de fórmulas XBRL como matriz de mapas JSON en la interfaz de la línea de comandos directamente. Consulte el apartado *Parámetros de XBRL Formula*.

- formula-parameters-file

--formula-parameters-file = *ARCHIVO*
Especifica un *ARCHIVO* que contiene los parámetros para la evaluación de fórmulas XBRL. Puede tratarse de un archivo XML o de un archivo JSON. Consulte el apartado *Parámetros de XBRL Formula* para obtener más información.

- ignore-assertion

--ignore-assertion = *VALOR*
Excluye la aserción dada de la ejecución. Añada esta opción varias veces para indicar más de una aserción.

- ignore-assertions-file

--ignore-assertions-file = *ARCHIVO*
Especifica la ruta de acceso de un archivo que contiene una lista de identificadores/XPointers que se deben excluir del procesamiento. En el archivo debe haber una aserción por línea.

- ignore-formula

--ignore-formula = *VALOR*
Excluye la fórmula dada de la ejecución. Añada esta opción varias veces para especificar más de una fórmula.

- ignore-formulas-file

--ignore-formulas-file = *ARCHIVO*

Especifica la ruta de acceso de un archivo que contiene una lista de identificadores/ XPointers de fórmula que se deben excluir del procesamiento. En el archivo debe haber una fórmula por línea.

- message-lang

  --message-lang = *VALUE*
  Especifica el idioma que se va a usar cuando se muestren los mensajes de validación. Por defecto es: `en`.
  El resto de valores aceptados son `de`, `es`, `fr`, `ja`, para alemán, español, francés y japonés, respectivamente.

- message-role

  --message-role = *VALUE*
  Especifica el papel del mensaje predefinido que se usa cuando se muestran los mensajes de validación. Por defecto es: `http://www.xbrl.org/2010/role/message`

- preload-formula-schemas

  --preload-formula-schemas = `true|false`
  Carga previa de los esquemas de la especificación XBRL Formula 1.0. Valor predeterminado: `false`.
  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- process-assertion [a]

  --a | --process-assertion = *VALOR*
  Limita la ejecución de fórmulas a la aserción dada. Añada la opción varias veces para especificar más de una aserción. Su forma abreviada es `--a`. El *VALOR* es el valor del atributo `@id` o un URI con un fragmento XPointer que identifica el recurso. También se pueden usar los valores especiales `##none` y `##all`.

- process-assertion-set [as]

  --as | --process-assertion-set = *VALOR*
  Limita la ejecución de fórmulas al conjunto de aserciones dado. Añada la opción varias veces para especificar más de un conjunto de aserciones. Su forma abreviada es `--as`. El ***VALOR*** es el valor del atributo `@id` o un URI con un fragmento XPointer que identifica el recurso. También puede usar los valores especiales `##none` y `##all`.

- process-assertions-file

  --process-assertions-file = *ARCHIVO*
  Especifica la ruta de acceso de un archivo que contiene una lista de identificadores/ XPointers de aserciones que se deben ejecutar. En el archivo debe haber una aserción por línea.

- process-formula [f]

  --f | --process-formula = *VALOR*
  Limita la ejecución de fórmulas a la fórmula dada. Añada la opción varias veces para especificar más de una fórmula. Su forma abreviada es `--as`. El ***VALOR*** es el valor del

atributo `@id` o un URI con un fragmento XPointer que identifica el recurso. También puede usar los valores especiales `##none` y `##all`.

- process-formulas-file

  `--process-formulas-file =` *`ARCHIVO`*
  Especifica la ruta de acceso de un archivo que contiene una lista de identificadores/ XPointers de fórmulas que se deben ejecutar. En el archivo debe haber una fórmula por línea.

- report-unsatisfied-assertion-evaluations

  `--report-unsatisfied-assertion-evaluations =` `true|false`
  Informa sobre la evaluación de una aserción insatisfactoria como errores o advertencias que dependen de nivel de severidad de la asercación. El valor predeterminado es `false`.
  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- validate-formula-output

  `--validate-formula-output =` `true|false`
  Habilita la validación XBRL de instancias de archivos de salida de XBRL Formula. El valor predeterminado es `false`.
  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- variableset-execution-timeout

  `--variableset-execution-timeout =` *`VALOR`*
  Se aplica cuando se ejecutan fórmulas (**`--formula-execution=true`**). Especifica el tiempo de espera máximo permitido para ejecutar un conjunto de variables único (una fórmula o un valor o una aserción de existencia o de coherencia). El tiempo de espera se indica en minutos y debe ser un número positivo. El valor predeterminado es 30 minutos. Si un conjunto de variables determinado no termina de ejecutarse antes de cumplirse el tiempo de espera, la ejecución se anula. Aparece un mensaje de error (que se introduce en el registro detallado). No obstante, debe tener en cuenta que la comprobación de tiempo de espera solamente se realiza después de evaluarse cada conjunto de variables (y no durante la ejecución de las expresiones XPath). Por tanto, si una expresión XPath tarda mucho en ejecutarse, puede que se cumpla el tiempo de espera. La ejecución del conjunto de variables se anula solamente cuando se ejecute la evaluación completa del conjunto de variables.

- Tablas XBRL
  - concept-label-linkrole

    `--concept-label-linkrole =` *`VALOR`*
    Especifica la función de vínculo preferida a la hora de representar etiquetas de concepto.

  - concept-label-role

    `--cconcept-label-role =` *`VALOR`*

Especifica la función de etiqueta preferida a la hora de representar etiquetas de concepto. La función predeterminada es `http://www.xbrl.org/2003/role/label`.

- generic-label-linkrole

  `--generic-label-linkrole =` *`VALOR`*
  Especifica la función de vínculo extendido preferida a la hora de representar etiquetas genéricas.

- generic-label-role

  `--generic-label-role =` *`VALOR`*
  Especifica la función de etiqueta preferida a la hora de representar etiquetas genéricas. La función predeterminada es `http://www.xbrl.org/2003/role/label`

- label-lang

  `--label-lang =` *`VALOR`*
  Especifica el idioma preferido a la hora de representar etiquetas. Valor predeterminado: `en`

- preload-table-schemas

  `--preload-table-schemas =` `true|false`
  Carga previa de los esquemas de la especificación XBRL Table 1.0. Valor predeterminado: `false`.
  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- process-table [t]

  `--t | --process-table =` *`VALOR`*
  Limita la ejecución de fórmulas a la taba dada. Añada la opción varias veces para especificar más de una tabla. Su forma abreviada es `--t`. El ***VALOR*** es el valor del atributo `@id` o un URI con un fragmento XPointer que identifica el recurso. También puede usar los valores especiales `##none` y `##all`.

- table

  `--table =` `true|false`
  Habilita la extensión XBRL Table 1.0. Si es `true`, especifica automáticamente `--formula=true` y `--dimensions=true`. Valor predeterminado: `true`.
  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- table-elimination

  `--table-elimination =` `true|false`
  Habilita la eliminación de columnas/filas de tabla vacías en el formato de salida HTML. Valor predeterminado: `true`.
  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- table-execution

--table-execution = true|false
Habilita la evaluación de tablas XBRL. Valor predeterminado: false. Si se especifica la opción --table-output, a esta opción se asigna true. Si tiene el valor true, se especifica automáticamente --table=true.
Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en true.

- table-linkbase-namespace

--table-linkbase-namespace =
##detect |
http://xbrl.org/PWD/2013-05-17/table |
http://xbrl.org/PWD/2013-08-28/table |
http://xbrl.org/CR/2013-11-13/table |
http://xbrl.org/PR/2013-12-18/table |
http://xbrl.org/2014/table

Habilita la carga de bases de enlaces de tabla escritas con una especificación borrador previa. La validación, la resolución y el diseño de bases de enlaces de tabla se realiza, sin embargo, siguiendo la recomendación Table Linkbase 1.0 del 18 de marzo de 2014. Para habilitar la detección automática, utilice ##detect.

- table-output

--table-output = *ARCHIVO*
Escribe la tabla de salida en el ***ARCHIVO*** indicado. Si se define, especifica automáticamente --table-execution=true.

- table-output-format

--table-output-format = xml|html
Especifica el formato de la tabla de salida. Valor predeterminado: xml.

- Catálogos y recursos globales
  - catalog

--catalog = *ARCHIVO*
Especifica la ruta de acceso absoluta a un archivo de catálogo que no está en el archivo de catálogo raíz instalado. El valor predeterminado es la ruta de acceso absoluta del archivo de catálogo raíz instalado (<carpeta-instalación>\Altova\RaptorXMLXBRLServer2018\etc\RootCatalog.xml). Consulte el apartado Catálogos XML para obtener más información.

  - user-catalog

--user-catalog = *ARCHIVO*
Especifica la ruta de acceso absoluta a un catálogo XML que debe utilizarse junto con el catálogo raíz. Consulte el apartado Catálogos XML para obtener más información.

  - enable-globalresources

--enable-globalresources = true|false

Habilita la función de recursos globales. Valor predeterminado: `false`.
Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- globalresourceconfig [gc]

  `--gc | --globalresourceconfig =` *`VALOR`*
  Especifica la configuración activa del recurso global (y habilita los recursos globales).

- globalresourcefile [gr]

  `--gr | --globalresourcefile =` *`ARCHIVO`*
  Especifica el archivo de recursos globales (y habilita los recursos globales).

- Mensajes, errores, ayuda, tiempo de espera, versión
  - error-format

    `--error-format =` `text|shortxml|longxml`
    Especifica el formato de la salida de error. Los valores posibles son formatos de texto, XML y XML detallado (`longxml`). Valor predeterminado: `text`.

  - error-limit

    `--error-limit =` *`N`*
    Especifica el límite de errores con un valor comprendido entre `1` y `999`. El valor predeterminado es `100`. Cuando se alcanza el límite de error, se detiene la validación. Esta opción es muy práctica a la hora de limitar el uso del procesador durante la validación o transformación.

  - help

    `--help`
    Muestra el texto de ayuda para el comando. Por ejemplo `valany --h`. (Otra opción es usar el comando `help` con un argumento. Por ejemplo: `help valany`).

  - log-output

    `--log-output =` *`ARCHIVO`*
    Escribe el registro de salida en la URL de archivo indicada. Compruebe que la ILC tiene permiso de escritura en la ubicación de destino.

  - network-timeout

    `--network-timeout =` *`VALOR`*
    Indica el tiempo de espera en segundos para operaciones remotas de entrada y salida. Valor predeterminado: `40`.

  - verbose

    `--verbose =` `true|false`
    Si el valor es `true`, se genera información adicional durante la validación. Valor predeterminado es `false`.
    Nota: si no se especifica un valor para la opción, el valor booleano de la opción se

establece en `true`.

- verbose-output

  `--verbose-output =` *`ARCHIVO`*
  Escribe el resultado detallado en el *`ARCHIVO`* indicado.

- version

  `--version`
  Muestra el número de versión de RaptorXML+XBRL Server. Si se utiliza con un comando, escriba la opción `--version` antes del comando.

- warning-limit

  `--warning-limit =` *`VALUE`*
  Especifica el límite de advertencia en el rango `1-65535`. El procesamiento continua si se alcanza el límite pero no se registrarán más advertencias. El valor por defecto es `100`.

3.3.2 valinlinexbrl (ixbrl)

El comando **`valinlinexbrl | ixbrl`** valida documentos Inline XBRL (iXBRL) de acuerdo con lo estipulado por la especificación Inline XBRL 1.0 o Inline XBRL 1.1. Con Inline XBRL se genera un archivo XBRL que se puede guardar.

Windows	**`RaptorXMLXBRL valxbrl	xbrl [opciones]`** ***`ArchivoEntrada`***
Linux	**`raptorxmlxbrl valxbrl	xbrl [opciones]`** ***`ArchivoEntrada`***
Mac	**`raptorxmlxbrl valxbrl	xbrl [opciones]`** ***`ArchivoEntrada`***

El argumento *`ArchivoEntrada`* especifica qué documento Inline XBRL se debe validar (por lo general se trata de un documento XHTML). En el primer paso del proceso de validación, la ejecución del comando transforma el documento XHTML en XBRL extrayendo los datos Inline XBRL. Si no se detectan errores en este primer paso y si la opción `--validate-xbrl` se configuró con el valor `true`, entonces el XBRL generado se valida con la taxonomía XBRL especificada por los datos Inline XBRL.

Si desea validar varios documentos, tiene dos opciones: (i) enumerar los archivos que se deben validar en la línea de comandos, separados por espacios o (ii) enumerar los archivos que se deben validar en un archivo de texto (`.txt`) donde aparece un nombre de archivo por línea y dar este archivo de texto como argumento *`ArchivoEntrada`* junto con la opción `--listfile` con valor `true` (*ver lista de opciones más abajo*). Recuerde que, aunque se procesen varios archivos de entrada Inline XBRL, solamente se genera un documento XBRL.

Ejemplos

- **raptorxmlxbrl** `valinlinexbrl c:\MiIXBRL.xhtml`
- **raptorxmlxbrl** `valinlinexbrl --ixbrl-version=1.1 c:\MiIXBRL.xhtml`
- **raptorxmlxbrl** `ixbrl --validate-xbrl=true --xbrl-output=C:\MiIXBRLsalida.xbrl C:\MiIXBRL.xhtml`
- **raptorxmlxbrl** `ixbrl --validate-xbrl=false --xbrl-output=C:\MiIXBRLsalida.xbrl C:\MiIXBRL.xhtml`
- **raptorxmlxbrl** `ixbrl --xbrl-output=C:\MiIXBRLsalida.xbrl --document-set=true --listfile=true C:\MiListaArchivos.txt`

Uso de mayúsculas/minúsculas y de barras diagonales en la línea de comandos

RaptorXMLXBRL *en Windows*
raptorxmlxbrl *en Unix (Linux, Mac)*

* Las minúsculas (`raptorxmlxbrl`) funcionan en todas las plataformas (Windows, Linux y Mac), mientras que la mezcla de mayúsculas y minúsculas (`RaptorXMLXBRL`) sólo funciona en Windows y Mac.
* En Linux y Mac utilice barras diagonales. En Windows utilice barras diagonales inversas.

Opciones

Las opciones del comando aparecen más abajo divididas en grupos. Los valores se pueden dar sin comillas excepto en estos dos casos: (i) cuando la cadena de valor contiene espacios y (ii) cuando en la descripción de la opción se indique explícitamente que es necesario el uso de comillas.

Validación y procesamiento XBRL

additional-dts-entry-point

--additional-dts-entry-point = *FILE*
Fuerza la inclusión del archivo de punto de entrada en el proceso de detección de DTS. Añada la opción varias veces para especificar más de un punto de entrada.

ct-restrict-mode

--ct-restrict-mode = `1.0|1.1|default`
Especifica cómo comprobar restricciones de tipo complejo. Un valor de **1.0** comprueba restricciones de tipo complejo conforme a lo definido en la especificación XSD 1.0 (incluso estando en modo de validación XSD 1.1). Un valor de **1.1** comprueba restricciones de tipo complejo conforme a los definido en la especificación XSD 1.1 (incluso estando en modo de validación XSD 1.0). Un valor de **default** comprueba restricciones de tipo complejo conforme a lo definido en la especificación XSD del modo de validación habilitado en ese momento (1.0 o 1.1). El valor predeterminado es **default**.

dimensions

--dimensions = `true|false`
Habilita las extensiones XBRL Dimension 1.0. Valor predeterminado: `true`.
Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

 © 2018 Altova GmbH

- dts

  --dts = *FILE*

  Recarga el punto de entrada DTS determinado en *ARCHIVO* y lo usa para la validación de archivos de instancia. Agregue la opción de varias veces para especificar más de un punto de entrada. Los puntos de entrada en los archivos de instancia se ignorarán. El comando se usa para validar por lotes varios archivos de instancia que hacen referencia a la misma taxonomía. La opción `--dts` da la instrucción al motor Raptor para que cargar el DTS en un sólo momento - y no para cada instancia en la lista de archivo. (Se genera un archivo o bien en el CLI o en un archivo de texto que se indica como lista de archivo configurando la opción `--listfile` en `true`.). Si un archivo de instancia en la lista de archivo hace referencia a una taxonomía diferente, entonces se genera un aviso. Este comando puede acelerar enormemente el tiempo de validación cuando se validan varios archivos de instancia de pequeño tamaño cada uno de los cuales posea un DTS de gran compatibilidad: no proporciona ventaja cuando se valida un archivo de instancia único.

- essence-alias-checks

  --essence-alias-checks = `true|false`

  Habilita la creación de informes sobre incoherencias relacionadas con essence-alias. El valor predeterminado es `true`.

  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- extensible-enumerations

  --extensible-enumerations = `true|false`

  Si es `true`, habilita la extensión Enumeraciones extensibles XBRL 1.0. Por defecto es: `true`.

- inconsistencies-limit

  --inconsistencies-limit = *VALUE*

  Especifica el límite de las inconsistencias XBRL con valores que se encuentran en el rango `1-65535`. El procesamiento continua si se alcanza el límite, pero no se registran el resto de las incoherencias. El valor por defecto es `100`.

- listfile

  --listfile = `true|false`

  Si el valor es `true`, el argumento *ArchivoEntrada* del comando se entiende como un archivo de texto que contiene un nombre de archivo por línea. Otra opción es enumerar los archivos en la ILC, separados por un espacio. No obstante, recuerde que las ILC tienen un límite de caracteres. Además, no olvide que la opción `--listfile` solamente afecta a los argumentos y no a las opciones.

  Valor predeterminado: `false`.

  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- parallel-assessment [pa]

  --pa | --parallel-assessment = `true|false`

Si el valor es `true`, la evaluación de la validez de esquemas se realiza en paralelo. Esto significa que si hay más de 128 elementos en cualquiera de los niveles, estos elementos se procesan en paralelo utilizando varios subprocesos. Por tanto, los archivos XML de gran tamaño se pueden procesar más rápido si se habilita esta opción. La evaluación en paralelo se lleva cabo nivel por nivel, pero puede ocurrir en varios niveles de un mismo conjunto de información. Recuerde que la evaluación en paralelo no funciona en modo de transmisión por secuencias. Por este motivo la opción `--streaming` se pasa por alto si el valor de la opción `--parallel-assessment` es `true`. Además, se usa más memoria cuando se utiliza la opción `--parallel-assessment`. El valor predeterminado de esta opción es false y su forma abreviada es `--pa`.

Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- preload-xbrl-schemas

`--preload-xbrl-schemas =` `true|false`

Carga previa de los esquemas de la especificación XBRL 2.1. Valor predeterminado: `true`.

Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- recurse

`--recurse =` `true|false`

Esta opción se utiliza para seleccionar ficheros dentro de subdirectorios (incluso en archivos ZIP). Si el valor es `true`, el argumento *`ArchivoEntrada`* del comando seleccionará el fichero seleccionado también en los subdirectorios. Por ejemplo: `test.zip|zip\test.xml` seleccionará los ficheros llamados `test.xml` en todos los subdirectorios de la carpeta ZIP. Si quiere puede usar los caracteres comodín `*` y `?`. Por ejemplo: `*.xml` seleccionaría todos los ficheros de la carpeta ZIP que tengan la extensión `.xml`. Valor predeterminado: `false`

Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- requires-element-checks

`--requires-element-checks =` `true|false`

Habilita la creación de informes sobre incoherencias relacionadas con requires-element. El valor predeterminado es `true`.

Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- schema-imports

`--schema-imports =` `load-by-schemalocation | load-preferring-schemalocation | load-by-namespace | load-combining-both | license-namespace-only`

Esta opción indica el comportamiento de los elementos `xs:import`. Cada uno de estos elementos tiene un atributo opcional `namespace` y un atributo opcional `schemaLocation`: `<import namespace="unEspacioNombres" schemaLocation="unaURL">`. La opción indica si se debe cargar un documento de esquema o solo autorizar a un espacio de nombres. Si la opción indica que se debe

cargar un documento de esquema, entonces indica también qué información debe utilizarse para encontrar el documento de esquema. Valor predeterminado: `load-preferring-schemalocation`.

- `load-by-schemalocation`: el valor del atributo `schemaLocation` se utiliza para buscar el esquema, teniendo en cuenta las asignaciones de catálogo. Si está presente el atributo `namespace`, se importa el espacio de nombres (con licencia).
- `load-preferring-schemalocation`: si está presente, se utiliza el atributo `schemaLocation` teniendo en cuenta las asignaciones de catálogo. Si no está presente el atributo `schemaLocation`, entonces se usa el valor del atributo `namespace` a través de las asignaciones de catálogo. Este es el **valor predeterminado**.
- `load-by-namespace`: el valor del atributo `namespace` se utiliza para buscar el esquema por medio de una asignación de catálogo.
- `load-combining-both`: si el atributo `namespace` o `schemaLocation` tiene una asignación de catálogo, entonces se usa la asignación. Si ambos atributos tienen asignaciones de catálogo, entonces es el valor de la opción `--schema-mapping` (opción XBRL y opción XML/XSD) decide qué asignación se utiliza. Si no hay ninguna asignación de catálogo, entonces se usa el atributo `schemaLocation`.
- `license-namespace-only`: se importa el espacio de nombres. No se importa el documento de esquema.

schema-mapping

`--schema-mapping =` `prefer-schemalocation | prefer-namespace`

Si se usa la ubicación y el espacio de nombres para buscar el documento de esquema, esta opción indica cuál de ellos debe ser la opción preferida durante la búsqueda en el catálogo.
Si la opción `--schemalocation-hints` o la opción `--schema-imports` tiene el valor `load-combining-both` y si las partes de espacio de nombres y URL pertinentes tienen asignaciones de catálogo, entonces el valor de la opción especifica cuál de las dos asignaciones se utiliza (la asignación del espacio de nombres o de la URL: el valor `prefer-schemalocation` se refiere a la asignación de la URL). Valor predeterminado: `prefer-schemalocation`.

schemalocation-hints

`--schemalocation-hints =` `load-by-schemalocation | load-by-namespace | load-combining-both | ignore`

Determina el comportamiento predeterminado de los atributos `xsi:schemaLocation` y `xsi:noNamespaceSchemaLocation`. Indica si se debe cargar un documento de esquema y, si así es, indica qué información debe utilizarse para encontrarlo. Valor predeterminado: `load-by-schemalocation`.

- **Valor predeterminado**: `load-by-schemalocation`. Este valor toma la URL de la ubicación del esquema de los atributos `xsi:schemaLocation` y `xsi:noNamespaceSchemaLocation` de los documentos de instancia XML o XBRL.

- El valor `load-by-namespace` toma la parte de espacio de nombres del atributo `xsi:schemaLocation` y una cadena vacía en el caso del atributo `xsi:noNamespaceSchemaLocation` y encuentra el esquema por medio de una asignación de catálogo.

- Si usa el valor `load-combining-both` y el espacio de nombres o la URL tienen una asignación de catálogo, se usa dicha asignación. Si ambos tienen asignaciones de catálogo, el valor de la opción `schema-mapping` (opción XBRL y opción XML/XSD) decide qué asignación se utiliza. Si ni el espacio de nombres ni la URL tiene una asignación de catálogo, se usa la URL.

- El valor `ignore` ignora los atributos `xsi:schemaLocation` y `xsi:noNamespaceSchemaLocation`.

script

`--script =` *`ARCHIVO`*
Una vez finalizada la validación, ejecuta el script Python. Para indicar más de un script basta con agregar la opción varias veces.

script-api-version

`--api, --script-api-version =` `1|2|2.1|2.2|2.3|2.4|2.4.1|2.5`
Especifica la versión API Python que ha de usarse para el script. El valor por defecto es la última versión, actualmente **`2.45`**. En vez de de los valores `1` y `2`, también puede usar losa valores 1.0 y 2.0 respectivamente.

script-output

`--script-output =` *`FILE`*
Escribe el resultado estandard del script en el archivo nombres como *`FILE`*.

script-param

`--script-param =` *`CLAVE:VALOR`*
Parámetros definidos por el usuario a los que se puede acceder durante la ejecución de scripts Python.

summation-item-checks

`--summation-item-checks =` `true|false`
Habilita la creación de informes sobre incoherencias relacionadas con el ítem de suma. El valor predeterminado es `true`.
Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

taxonomy-package

`--taxonomy-package =` *`FILE`*
Especifica la ruta absoluta al paquete de taxonomía tradicional tal y como se describe en los barrados de trabajo del Paquete de taxonomía 1.0 y los Paquetes de taxonomía 1.0. El valor del ARCHIVO devuelve la ubicación del paquete de la taxonomía. Agregue la opción *varias veces* para especificar más de un paquete de taxonomía.

taxonomy-packages-config-file

`--taxonomy-packages-config-file =` *`ARCHIVO`*
Especifica la ruta de acceso del archivo **`TaxonomyPackagesConfig.json`**, que se utiliza pra cargar los paquetes de taxonomías XBRL. Es recomendable tener este archivo porque sirve como catálogo de los paquetes de taxonomías. La estructura del archivo

JSON es como la del ejemplo que aparece más abajo. El valor de la clave **uri** indica la ubicación del paquete. La clave **active** (de un conjunto de paquetes) cambia cuando se usan dichos paquetes.

```
{"taxonomies":
  {"EIOPA Solvency II XBRL Taxonomy 2.1.0":
    {"packages":[
      {"uri":"C:\\test\\XBRL\
\EIOPA_SolvencyII_XBRL_Taxonomy_2.1.0.zip"},
      {"uri":"C:\\test\\XBRL\\AdditionalTestPkg.zip"}
      ],"active":true
    },"Test Taxonomy":
      {"packages":[{"uri":"C:\\test\\XBRL\\test.zip"}],"active":true
      }
  }
}
```

- treat-inconsistencies-as-errors

  --treat-inconsistencies-as-errors = `true|false`

  La validación XBRL falla si el archivo contiene incoherencias según lo definido por la especificación XBRL 2.1. El valor predeterminado es `false`.

  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- utr

  --utr = `true|false`

  Si es `true`, habilita la extensión Registro de unidad XBRL 1.0. Por defecto es: `false`.

- utr-file

  --utr-file = *FILE*

  Especifica la ruta de acceso del archivo del registro de unidades. El valor predeterminado es: **http://www.xbrl.org/utr/utr.xml**.

- utr-status

  --utr-status = `PWD|CR|REC|NIE|PROPOSED|IWD|ACK|RR`

  Habilita la compatibilidad con definiciones de unidad con el estado dado en el registro de unidades. Añada la opción varias veces para indicar más de un estado de unidad. El valor predeterminado es **REC**.

- validate-dts-only

  --validate-dts-only = `true|false`

  La DTS se detecta iniciando desde el documento de instancia XBRL. Todos los esquemas de taxonomía y bases de enlace referenciados se detectan y validan. Los demás documentos de instancia XBRL se ignoran. Valor predeterminado: `false`.

  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- xinclude

  --xinclude = `true|false`

  Habilita la compatibilidad con inclusiones XML (XInclude). Si el valor es `false`, los

elementos XInclude `include` se ignoran. Valor predeterminado: `false`.
Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- Inline XBRL
  - document-set

    `--document-set =` `true|false`
    Si su valor es `true`, todos los archivos suministrados (los documentos Inline XBRL) se tratarán como un conjunto de documentos Inline XBRL. Valor predeterminado: `false`.
    Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

  - ixbrl-version

    `--ixbrl-version =` `1.0|1.1|detect`
    Especifica qué versión de la especificación Inline XBRL debe utilizarse para la validación. Valor predeterminado: `detect`.

  - transformation-registry

    `--transformation-registry =`
    `#all |`
    `http://www.xbrl.org/2008/inlineXBRL/transformation |`
    `http://www.xbrl.org/inlineXBRL/transformation/2010-04-20 |`
    `http://www.xbrl.org/inlineXBRL/transformation/2011-07-31 |`
    `http://www.xbrl.org/inlineXBRL/transformation/2015-02-26`

    En Inline XBRL los tipos numéricos y de fecha se pueden representar con diferentes formatos léxicos, pero en XBRL estos mismos tipos tienen formatos específicos. Por ejemplo, una fecha en Inline XBRL puede darse como `01 January 2017`, pero en XBRL el formato de esta fecha sería `2017-01-01`. La recomendación Inline XBRL Transformation Registries especifica las reglas de conversión que se deben seguir para convertir datos numéricos y de fecha de Inline XBRL a XBRL. La opción `transformation-registry` sirve para especificar qué registro se utiliza. Para especificar más de un registro de transformación basta con añadir la opción tantas veces como sea necesario. El valor predeterminado es `#all`, que selecciona todos los registros de transformación disponibles.

    Nota: para ver una lista de los registros de transformación disponibles en su versión de RaptorXML+XBRL Server, consulte la descripción del comando en la interfaz de la línea de comandos.

  - uri-transformation

    `--uri-transformation =` `none|make-absolute|make-relative|keep-relative`
    Especifica cómo se deben escribir los URI en el documento XBRL que se genera.

    - `none`: copiar el URI textualmente en el documento de destino.
    - `make-absolute`: convertir los URI relativos en absolutos resolviéndolos con el URI base dentro del ámbito del elemento correspondiente en el documento de entrada. Por ejemplo: si el archivo de entrada es `c:\test\inlinexbrl.xhtml`

y contiene una referencia relativa a un esquema llamado `schemas\schema.xsd`, entonces la referencia relativa se resolvería como `c:\test\schemas\schema.xsd`. Recuerde que los atributos `xml:base` del documento de entrada pueden cambiar el URI base.

- `make-relative`: si se puede, convertir los URI absolutos y relativos en relativos al documento de salida (y si no se puede, escribir el URI absoluto resuelto).
- `keep-relative`: si se puede, convertir solamente los URI relativos en relativos al documento de salida (y copiar los URI que sean absolutos).

▼ validate-xbrl

`--validate-xbrl =` `true|false`

Si su valor es `true`, habilita la validación del documento XBRL que se genera a partir del documento Inline XBRL. Por el contrario, si su valor es `false`, la validación se detiene después del primer paso que genere el XBRL. Valor predeterminado: `true`.

Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

▼ xbrl-output

`--xbrl-output =` *`ARCHIVO`*

Especifica la ubicación donde se debe escribir el archivo XBRL de salida que se genera. Recuerde que, aunque se procesen varios archivos Inline XBRL de entrada, solamente se genera un documento XBRL.

▼ Fórmulas y aserciones XBRL

▼ assertion-severity

`--assertion-severity =` `true|false`

Activa la extensión Assertion Severity 1.0. El valor por defecto es `true`.

Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

▼ assertions-output

`--assertions-output =` *`ARCHIVO`*

Escribe el resultado de la evaluación de la aserción en el *`ARCHIVO`* indicado. Si se define, especifica automáticamente `--formula-execution=true`.

▼ assertions-output-format

`--assertions-output-format =` `json|xml`

Especifica el formato de salida de la evaluación de la aserción. Valor predeterminado: `json`.

▼ evaluate-referenced-parameters-only

`--evaluate-referenced-parameters-only =` `true|false`

Si tiene valor `false`, obliga a evaluar todos los parámetros incluso si no se les hace referencia en ninguna fórmula, aserción o tabla. Valor predeterminado: `true`.

- formula

  --formula = true|false
  Habilita la extensión XBRL Formula 1.0. Valor predeterminado: true.
  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en true.

- formula-assertion-set [[OBSOLETA]]

  --formula-assertion-set = *VALOR*
  Limita la ejecución de fórmulas al conjunto de aserciones dado. Añada la opción varias veces para especificar más de un grupo de aserciones. Su forma abreviada es --as. El *VALOR* es el valor del atributo @id o un URI con un fragmento XPointer que identifica el recurso. También puede usar los valores especiales ##none y ##all.

- formula-execution

  --formula-execution = true|false
  Habilita la evaluación de fórmulas XBRL. Valor predeterminado: true. Si es true, especifica automáticamente --formula=true.
  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en true.

- formula-output

  --formula-output = *ARCHIVO*
  Escribe el resultado de la evaluación de la fórmula en el *ARCHIVO* indicado. Si se define, especifica automáticamente --formula-execution=true.

- formula-parameters

  --formula-parameters = *MATRIZ-JSON*
  Especifica parámetros para la evaluación de fórmulas XBRL como matriz de mapas JSON en la interfaz de la línea de comandos directamente. Consulte el apartado *Parámetros de XBRL Formula*.

- formula-parameters-file

  --formula-parameters-file = *ARCHIVO*
  Especifica un *ARCHIVO* que contiene los parámetros para la evaluación de fórmulas XBRL. Puede tratarse de un archivo XML o de un archivo JSON. Consulte el apartado *Parámetros de XBRL Formula* para obtener más información.

- ignore-assertion

  --ignore-assertion = *VALOR*
  Excluye la aserción dada de la ejecución. Añada esta opción varias veces para indicar más de una aserción.

- ignore-assertions-file

  --ignore-assertions-file = *ARCHIVO*
  Especifica la ruta de acceso de un archivo que contiene una lista de identificadores/XPointers que se deben excluir del procesamiento. En el archivo debe haber una aserción por línea.

 © 2018 Altova GmbH

- ignore-formula

  `--ignore-formula =` *`VALOR`*
  Excluye la fórmula dada de la ejecución. Añada esta opción varias veces para especificar más de una fórmula.

- ignore-formulas-file

  `--ignore-formulas-file =` *`ARCHIVO`*
  Especifica la ruta de acceso de un archivo que contiene una lista de identificadores/XPointers de fórmula que se deben excluir del procesamiento. En el archivo debe haber una fórmula por línea.

- message-lang

  `--message-lang =` *`VALUE`*
  Especifica el idioma que se va a usar cuando se muestren los mensajes de validación. Por defecto es: `en`.
  El resto de valores aceptados son `de`, `es`, `fr`, `ja`, para alemán, español, francés y japonés, respectivamente.

- message-role

  `--message-role =` *`VALUE`*
  Especifica el papel del mensaje predefinido que se usa cuando se muestran los mensajes de validación. Por defecto es: `http://www.xbrl.org/2010/role/message`

- preload-formula-schemas

  `--preload-formula-schemas =` `true|false`
  Carga previa de los esquemas de la especificación XBRL Formula 1.0. Valor predeterminado: `false`.
  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- process-assertion [a]

  `--a | --process-assertion =` *`VALOR`*
  Limita la ejecución de fórmulas a la aserción dada. Añada la opción varias veces para especificar más de una aserción. Su forma abreviada es `--a`. El *`VALOR`* es el valor del atributo `@id` o un URI con un fragmento XPointer que identifica el recurso. También se pueden usar los valores especiales `##none` y `##all`.

- process-assertion-set [as]

  `--as | --process-assertion-set =` *`VALOR`*
  Limita la ejecución de fórmulas al conjunto de aserciones dado. Añada la opción varias veces para especificar más de un conjunto de aserciones. Su forma abreviada es `--as`. El ***`VALOR`*** es el valor del atributo `@id` o un URI con un fragmento XPointer que identifica el recurso. También puede usar los valores especiales `##none` y `##all`.

- process-assertions-file

  `--process-assertions-file =` *`ARCHIVO`*
  Especifica la ruta de acceso de un archivo que contiene una lista de identificadores/

XPointers de aserciones que se deben ejecutar. En el archivo debe haber una aserción por línea.

- process-formula [f]

  --f | --process-formula = *VALOR*
  Limita la ejecución de fórmulas a la fórmula dada. Añada la opción varias veces para especificar más de una fórmula. Su forma abreviada es `--as`. El ***VALOR*** es el valor del atributo `@id` o un URI con un fragmento XPointer que identifica el recurso. También puede usar los valores especiales `##none` y `##all`.

- process-formulas-file

  --process-formulas-file = *ARCHIVO*
  Especifica la ruta de acceso de un archivo que contiene una lista de identificadores/XPointers de fórmulas que se deben ejecutar. En el archivo debe haber una fórmula por línea.

- report-unsatisfied-assertion-evaluations

  --report-unsatisfied-assertion-evaluations = `true|false`
  Informa sobre la evaluación de una aserción insatisfactoria como errores o advertencias que dependen de nivel de severidad de la asercación. El valor predeterminado es `false`.
  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- variableset-execution-timeout

  --variableset-execution-timeout = *VALOR*
  Se aplica cuando se ejecutan fórmulas (**`--formula-execution=true`**). Especifica el tiempo de espera máximo permitido para ejecutar un conjunto de variables único (una fórmula o un valor o una aserción de existencia o de coherencia). El tiempo de espera se indica en minutos y debe ser un número positivo. El valor predeterminado es 30 minutos. Si un conjunto de variables determinado no termina de ejecutarse antes de cumplirse el tiempo de espera, la ejecución se anula. Aparece un mensaje de error (que se introduce en el registro detallado). No obstante, debe tener en cuenta que la comprobación de tiempo de espera solamente se realiza después de evaluarse cada conjunto de variables (y no durante la ejecución de las expresiones XPath). Por tanto, si una expresión XPath tarda mucho en ejecutarse, puede que se cumpla el tiempo de espera. La ejecución del conjunto de variables se anula solamente cuando se ejecute la evaluación completa del conjunto de variables.

- Tablas XBRL
  - aspect-node-order

    --aspect-node-order = `lexical|instance`
    Especifica el orden de la presentación de un nodoAspecto abierto. El valor predeterminado es **`lexical`**.

  - concept-label-linkrole

    --concept-label-linkrole = *VALOR*

Especifica la función de vínculo preferida a la hora de representar etiquetas de concepto.

- concept-label-role

  --cconcept-label-role = *VALOR*
  Especifica la función de etiqueta preferida a la hora de representar etiquetas de concepto. La función predeterminada es `http://www.xbrl.org/2003/role/label`.

- generic-label-linkrole

  --generic-label-linkrole = *VALOR*
  Especifica la función de vínculo extendido preferida a la hora de representar etiquetas genéricas.

- generic-label-role

  --generic-label-role = *VALOR*
  Especifica la función de etiqueta preferida a la hora de representar etiquetas genéricas. La función predeterminada es `http://www.xbrl.org/2003/role/label`

- label-lang

  --label-lang = *VALOR*
  Especifica el idioma preferido a la hora de representar etiquetas. Valor predeterminado: `en`

- preload-table-schemas

  --preload-table-schemas = `true|false`
  Carga previa de los esquemas de la especificación XBRL Table 1.0. Valor predeterminado: `false`.
  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- process-table [t]

  --t | --process-table = *VALOR*
  Limita la ejecución de fórmulas a la taba dada. Añada la opción varias veces para especificar más de una tabla. Su forma abreviada es `--t`. El ***VALOR*** es el valor del atributo `@id` o un URI con un fragmento XPointer que identifica el recurso. También puede usar los valores especiales `##none` y `##all`.

- table

  --table = `true|false`
  Habilita la extensión XBRL Table 1.0. Si es `true`, especifica automáticamente `--formula=true` y `--dimensions=true`. Valor predeterminado: `true`.
  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- table-elimination

  --table-elimination = `true|false`
  Habilita la eliminación de columnas/filas de tabla vacías en el formato de salida HTML.

Valor predeterminado: `true`.
Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- table-execution

`--table-execution =` `true|false`
Habilita la evaluación de tablas XBRL. Valor predeterminado: `false`. Si se especifica la opción `--table-output`, a esta opción se asigna `true`. Si tiene el valor `true`, se especifica automáticamente `--table=true`.
Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- table-linkbase-namespace

```
--table-linkbase-namespace =
  ##detect |
  http://xbrl.org/PWD/2013-05-17/table |
  http://xbrl.org/PWD/2013-08-28/table |
  http://xbrl.org/CR/2013-11-13/table |
  http://xbrl.org/PR/2013-12-18/table |
  http://xbrl.org/2014/table
```

Habilita la carga de bases de enlaces de tabla escritas con una especificación borrador previa. La validación, la resolución y el diseño de bases de enlaces de tabla se realiza, sin embargo, siguiendo la recomendación Table Linkbase 1.0 del 18 de marzo de 2014. Para habilitar la detección automática, utilice `##detect`.

- table-output

`--table-output =` *`ARCHIVO`*
Escribe la tabla de salida en el ***ARCHIVO*** indicado. Si se define, especifica automáticamente `--table-execution=true`.

- table-output-format

`--table-output-format =` `xml|html`
Especifica el formato de la tabla de salida. Valor predeterminado: `xml`.

- Catálogos y recursos globales
  - catalog

`--catalog =` *`ARCHIVO`*
Especifica la ruta de acceso absoluta a un archivo de catálogo que no está en el archivo de catálogo raíz instalado. El valor predeterminado es la ruta de acceso absoluta del archivo de catálogo raíz instalado (`<carpeta-instalación>\Altova\RaptorXMLXBRLServer2018\etc\RootCatalog.xml`). Consulte el apartado Catálogos XML para obtener más información.

  - user-catalog

`--user-catalog =` *`ARCHIVO`*

Especifica la ruta de acceso absoluta a un catálogo XML que debe utilizarse junto con el catálogo raíz. Consulte el apartado Catálogos XML para obtener más información.

- enable-globalresources

  --enable-globalresources = `true|false`
  Habilita la función de recursos globales. Valor predeterminado: `false`.
  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- globalresourceconfig [gc]

  --gc | --globalresourceconfig = *VALOR*
  Especifica la configuración activa del recurso global (y habilita los recursos globales).

- globalresourcefile [gr]

  --gr | --globalresourcefile = *ARCHIVO*
  Especifica el archivo de recursos globales (y habilita los recursos globales).

- Mensajes, errores, ayuda, tiempo de espera, versión
  - error-format

    --error-format = `text|shortxml|longxml`
    Especifica el formato de la salida de error. Los valores posibles son formatos de texto, XML y XML detallado (`longxml`). Valor predeterminado: `text`.

  - error-limit

    --error-limit = *N*
    Especifica el límite de errores con un valor comprendido entre `1` y `999`. El valor predeterminado es `100`. Cuando se alcanza el límite de error, se detiene la validación. Esta opción es muy práctica a la hora de limitar el uso del procesador durante la validación o transformación.

  - help

    --help
    Muestra el texto de ayuda para el comando. Por ejemplo `valany --h`. (Otra opción es usar el comando `help` con un argumento. Por ejemplo: `help valany`).

  - log-output

    --log-output = *ARCHIVO*
    Escribe el registro de salida en la URL de archivo indicada. Compruebe que la ILC tiene permiso de escritura en la ubicación de destino.

  - network-timeout

    --network-timeout = *VALOR*
    Indica el tiempo de espera en segundos para operaciones remotas de entrada y salida. Valor predeterminado: `40`.

- verbose

  --verbose = true|false
  Si el valor es true, se genera información adicional durante la validación. Valor predeterminado es false.
  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en true.

- verbose-output

  --verbose-output = *ARCHIVO*
  Escribe el resultado detallado en el *ARCHIVO* indicado.

- version

  --version
  Muestra el número de versión de RaptorXML+XBRL Server. Si se utiliza con un comando, escriba la opción --version antes del comando.

- warning-limit

  --warning-limit = *VALUE*
  Especifica el límite de advertencia en el rango 1-65535. El procesamiento continua si se alcanza el límite pero no se registrarán más advertencias. El valor por defecto es 100.

3.3.3 valxbrltaxonomy (dts)

El comando **valxbrltaxonomy | dts** valida una o varias taxonomías (esquemas) XBRL con las especificaciones XBRL 2.1, Dimensions 1.0 y Formula 1.0.

Windows	**RaptorXMLXBRL valxbrltaxonomy | dts [opciones] *ArchivoEntrada***
Linux	**raptorxmlxbrl valxbrltaxonomy | dts [opciones] *ArchivoEntrada***
Mac	**raptorxmlxbrl valxbrltaxonomy | dts [opciones] *ArchivoEntrada***

El argumento *ArchivoEntrada* es la taxonomía XBRL que debe validarse.

Si desea validar varios documentos, tiene dos opciones: (i) enumerar los archivos que se deben validar en la línea de comandos, separados por espacios o (ii) enumerar los archivos que se deben validar en un archivo de texto (.txt) donde aparece un nombre de archivo por línea y dar este archivo de texto como argumento *ArchivoEntrada* junto con la opción --listfile con valor true (*ver lista de opciones más abajo*).

Ejemplos

- **raptorxmlxbrl** valxbrltaxonomy c:\Test.xsd
- **raptorxmlxbrl** dts --listfile c:\ListaArchivos.txt

Uso de mayúsculas/minúsculas y de barras diagonales en la línea de comandos

RaptorXMLXBRL *en Windows*
raptorxmlxbrl *en Unix (Linux, Mac)*

* Las minúsculas (raptorxmlxbrl) funcionan en todas las plataformas (Windows, Linux y Mac), mientras que la mezcla de mayúsculas y minúsculas (RaptorXMLXBRL) sólo funciona en Windows y Mac.
* En Linux y Mac utilice barras diagonales. En Windows utilice barras diagonales inversas.

Opciones

Las opciones del comando aparecen más abajo divididas en grupos. Los valores se pueden dar sin comillas excepto en estos dos casos: (i) cuando la cadena de valor contiene espacios y (ii) cuando en la descripción de la opción se indique explícitamente que es necesario el uso de comillas.

Validación y procesamiento XBRL

ct-restrict-mode

--ct-restrict-mode = 1.0|1.1|default
Especifica cómo comprobar restricciones de tipo complejo. Un valor de **1.0** comprueba restricciones de tipo complejo conforme a lo definido en la especificación XSD 1.0 (incluso estando en modo de validación XSD 1.1). Un valor de **1.1** comprueba restricciones de tipo complejo conforme a los definido en la especificación XSD 1.1 (incluso estando en modo de validación XSD 1.0). Un valor de **default** comprueba restricciones de tipo complejo conforme a lo definido en la especificación XSD del modo de validación habilitado en ese momento (1.0 o 1.1). El valor predeterminado es **default**.

assertion-severity

--assertion-severity = true|false
Activa la extensión Assertion Severity 1.0. El valor por defecto es true.
Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en true.

dimensions

--dimensions = true|false
Habilita las extensiones XBRL Dimension 1.0. Valor predeterminado: true.
Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en true.

evaluate-referenced-parameters-only

--evaluate-referenced-parameters-only = true|false
Si tiene valor false, obliga a evaluar todos los parámetros incluso si no se les hace referencia en ninguna fórmula, aserción o tabla. Valor predeterminado: true.

extensible-enumerations

--extensible-enumerations = true|false
Si es true, habilita la extensión Enumeraciones extensibles XBRL 1.0. Por defecto es: true.

- inconsistencies-limit

--inconsistencies-limit = *VALUE*
Especifica el límite de las inconsistencias XBRL con valores que se encuentran en el rango 1-65535. El procesamiento continua si se alcanza el límite, pero no se registran el resto de las incoherencias. El valor por defecto es 100.

- formula

--formula = true|false
Habilita la extensión XBRL Formula 1.0. Valor predeterminado: true.
Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en true.

- formula-parameters

--formula-parameters = *MATRIZ-JSON*
Especifica parámetros para la evaluación de fórmulas XBRL como matriz de mapas JSON en la interfaz de la línea de comandos directamente. Consulte el apartado *Parámetros de XBRL Formula*.

- formula-parameters-file

--formula-parameters-file = *ARCHIVO*
Especifica un *ARCHIVO* que contiene los parámetros para la evaluación de fórmulas XBRL. Puede tratarse de un archivo XML o de un archivo JSON. Consulte el apartado *Parámetros de XBRL Formula* para obtener más información.

- listfile

--listfile = true|false
Si el valor es true, el argumento *ArchivoEntrada* del comando se entiende como un archivo de texto que contiene un nombre de archivo por línea. Otra opción es enumerar los archivos en la ILC, separados por un espacio. No obstante, recuerde que las ILC tienen un límite de caracteres. Además, no olvide que la opción --listfile solamente afecta a los argumentos y no a las opciones.
Valor predeterminado: false.
Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en true.

- preload-formula-schemas

--preload-formula-schemas = true|false
Carga previa de los esquemas de la especificación XBRL Formula 1.0. Valor predeterminado: false.
Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en true.

- preload-xbrl-schemas

--preload-xbrl-schemas = true|false

Carga previa de los esquemas de la especificación XBRL 2.1. Valor predeterminado: `true`.
Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- recurse

`--recurse =` `true|false`
Esta opción se utiliza para seleccionar ficheros dentro de subdirectorios (incluso en archivos ZIP). Si el valor es `true`, el argumento *`ArchivoEntrada`* del comando seleccionará el fichero seleccionado también en los subdirectorios. Por ejemplo: `test.zip|zip\test.xml` seleccionará los ficheros llamados `test.xml` en todos los subdirectorios de la carpeta ZIP. Si quiere puede usar los caracteres comodín `*` y `?`. Por ejemplo: `*.xml` seleccionaría todos los ficheros de la carpeta ZIP que tengan la extensión `.xml`. Valor predeterminado: `false`
Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- schema-imports

`--schema-imports =` `load-by-schemalocation | load-preferring-schemalocation | load-by-namespace | load-combining-both | license-namespace-only`
Esta opción indica el comportamiento de los elementos `xs:import`. Cada uno de estos elementos tiene un atributo opcional `namespace` y un atributo opcional `schemaLocation`: `<import namespace="unEspacioNombres" schemaLocation="unaURL">`. La opción indica si se debe cargar un documento de esquema o solo autorizar a un espacio de nombres. Si la opción indica que se debe cargar un documento de esquema, entonces indica también qué información debe utilizarse para encontrar el documento de esquema. Valor predeterminado: `load-preferring-schemalocation`.

- `load-by-schemalocation`: el valor del atributo `schemaLocation` se utiliza para buscar el esquema, teniendo en cuenta las asignaciones de catálogo. Si está presente el atributo `namespace`, se importa el espacio de nombres (con licencia).
- `load-preferring-schemalocation`: si está presente, se utiliza el atributo `schemaLocation` teniendo en cuenta las asignaciones de catálogo. Si no está presente el atributo `schemaLocation`, entonces se usa el valor del atributo `namespace` a través de las asignaciones de catálogo. Este es el **valor predeterminado**.
- `load-by-namespace`: el valor del atributo `namespace` se utiliza para buscar el esquema por medio de una asignación de catálogo.
- `load-combining-both`: si el atributo `namespace` o `schemaLocation` tiene una asignación de catálogo, entonces se usa la asignación. Si ambos atributos tienen asignaciones de catálogo, entonces es el valor de la opción `--schema-mapping` (opción XBRL y opción XML/XSD) decide qué asignación se utiliza. Si no hay ninguna asignación de catálogo, entonces se usa el atributo `schemaLocation`.
- `license-namespace-only`: se importa el espacio de nombres. No se importa el documento de esquema.

- schema-mapping

`--schema-mapping =` `prefer-schemalocation | prefer-namespace`

Si se usa la ubicación y el espacio de nombres para buscar el documento de esquema, esta opción indica cuál de ellos debe ser la opción preferida durante la búsqueda en el catálogo.
Si la opción `--schemalocation-hints` o la opción `--schema-imports` tiene el valor `load-combining-both` y si las partes de espacio de nombres y URL pertinentes tienen asignaciones de catálogo, entonces el valor de la opción especifica cuál de las dos asignaciones se utiliza (la asignación del espacio de nombres o de la URL: el valor `prefer-schemalocation` se refiere a la asignación de la URL). Valor predeterminado: `prefer-schemalocation`.

schemalocation-hints

`--schemalocation-hints =` `load-by-schemalocation | load-by-namespace | load-combining-both | ignore`
Determina el comportamiento predeterminado de los atributos `xsi:schemaLocation` y `xsi:noNamespaceSchemaLocation`. Indica si se debe cargar un documento de esquema y, si así es, indica qué información debe utilizarse para encontrarlo. Valor predeterminado: `load-by-schemalocation`.

- **Valor predeterminado**: `load-by-schemalocation`. Este valor toma la URL de la ubicación del esquema de los atributos `xsi:schemaLocation` y `xsi:noNamespaceSchemaLocation` de los documentos de instancia XML o XBRL.
- El valor `load-by-namespace` toma la parte de espacio de nombres del atributo `xsi:schemaLocation` y una cadena vacía en el caso del atributo `xsi:noNamespaceSchemaLocation` y encuentra el esquema por medio de una asignación de catálogo.
- Si usa el valor `load-combining-both` y el espacio de nombres o la URL tienen una asignación de catálogo, se usa dicha asignación. Si ambos tienen asignaciones de catálogo, el valor de la opción `schema-mapping` (opción XBRL y opción XML/XSD) decide qué asignación se utiliza. Si ni el espacio de nombres ni la URL tiene una asignación de catálogo, se usa la URL.
- El valor `ignore` ignora los atributos `xsi:schemaLocation` y `xsi:noNamespaceSchemaLocation`.

script

`--script =` *`ARCHIVO`*
Una vez finalizada la validación, ejecuta el script Python. Para indicar más de un script basta con agregar la opción varias veces.

script-api-version

`--api, --script-api-version =` `1|2|2.1|2.2|2.3|2.4|2.4.1|2.5`
Especifica la versión API Python que ha de usarse para el script. El valor por defecto es la última versión, actualmente **`2.45`**. En vez de de los valores `1` y `2`, también puede usar losa valores 1.0 y 2.0 respectivamente.

script-output

`--script-output =` *`FILE`*
Escribe el resultado estandard del script en el archivo nombres como *`FILE`*.

 © 2018 Altova GmbH

- script-param

  --script-param = *CLAVE:VALOR*

  Parámetros definidos por el usuario a los que se puede acceder durante la ejecución de scripts Python.

- taxonomy-package

  --taxonomy-package = *FILE*

  Especifica la ruta absoluta al paquete de taxonomía tradicional tal y como se describe en los barrados de trabajo del Paquete de taxonomía 1.0 y los Paquetes de taxonomía 1.0. El valor del ARCHIVO devuelve la ubicación del paquete de la taxonomía. Agregue la opción *varias veces* para especificar más de un paquete de taxonomía.

- taxonomy-packages-config-file

  --taxonomy-packages-config-file = *ARCHIVO*

  Especifica la ruta de acceso del archivo **TaxonomyPackagesConfig.json**, que se utiliza pra cargar los paquetes de taxonomías XBRL. Es recomendable tener este archivo porque sirve como catálogo de los paquetes de taxonomías. La estructura del archivo JSON es como la del ejemplo que aparece más abajo. El valor de la clave **uri** indica la ubicación del paquete. La clave **active** (de un conjunto de paquetes) cambia cuando se usan dichos paquetes.

```
{"taxonomies":
  {"EIOPA Solvency II XBRL Taxonomy 2.1.0":
    {"packages":[
      {"uri":"C:\\test\\XBRL\
\EIOPA_SolvencyII_XBRL_Taxonomy_2.1.0.zip"},
      {"uri":"C:\\test\\XBRL\\AdditionalTestPkg.zip"}
      ],"active":true
    },"Test Taxonomy":
      {"packages":[{"uri":"C:\\test\\XBRL\\test.zip"}],"active":true
      }
  }
}
```

- treat-inconsistencies-as-errors

  --treat-inconsistencies-as-errors = `true|false`

  La validación XBRL falla si el archivo contiene incoherencias según lo definido por la especificación XBRL 2.1. El valor predeterminado es `false`.

  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- xinclude

  --xinclude = `true|false`

  Habilita la compatibilidad con inclusiones XML (XInclude). Si el valor es `false`, los elementos XInclude `include` se ignoran. Valor predeterminado: `false`.

  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- Tablas XBRL

- aspect-node-order

  --aspect-node-order = `lexical|instance`
  Especifica el orden de la presentación de un nodoAspecto abierto. El valor predeterminado es **`lexical`**.

- concept-label-linkrole

  --concept-label-linkrole = *VALOR*
  Especifica la función de vínculo preferida a la hora de representar etiquetas de concepto.

- concept-label-role

  --cconcept-label-role = *VALOR*
  Especifica la función de etiqueta preferida a la hora de representar etiquetas de concepto. La función predeterminada es `http://www.xbrl.org/2003/role/label`.

- evaluate-referenced-parameters-only

  --evaluate-referenced-parameters-only = `true|false`
  Si tiene valor `false`, obliga a evaluar todos los parámetros incluso si no se les hace referencia en ninguna fórmula, aserción o tabla. Valor predeterminado: `true`.

- generic-label-linkrole

  --generic-label-linkrole = *VALOR*
  Especifica la función de vínculo extendido preferida a la hora de representar etiquetas genéricas.

- generic-label-role

  --generic-label-role = *VALOR*
  Especifica la función de etiqueta preferida a la hora de representar etiquetas genéricas. La función predeterminada es `http://www.xbrl.org/2003/role/label`

- label-lang

  --label-lang = *VALOR*
  Especifica el idioma preferido a la hora de representar etiquetas. Valor predeterminado: `en`

- preload-table-schemas

  --preload-table-schemas = `true|false`
  Carga previa de los esquemas de la especificación XBRL Table 1.0. Valor predeterminado: `false`.
  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- process-table [t]

  --t | --process-table = *VALOR*
  Limita la ejecución de fórmulas a la taba dada. Añada la opción varias veces para especificar más de una tabla. Su forma abreviada es `--t`. El ***VALOR*** es el valor del atributo `@id` o un URI con un fragmento XPointer que identifica el recurso. También

puede usar los valores especiales `##none` y `##all`.

- table

  `--table =` `true|false`
  Habilita la extensión XBRL Table 1.0. Si es `true`, especifica automáticamente `--formula=true` y `--dimensions=true`. Valor predeterminado: `true`.
  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- table-execution

  `--table-execution =` `true|false`
  Habilita la evaluación de tablas XBRL. Valor predeterminado: `false`. Si se especifica la opción `--table-output`, a esta opción se asigna `true`. Si tiene el valor `true`, se especifica automáticamente `--table=true`.
  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- table-linkbase-namespace

  `--table-linkbase-namespace =`
  `##detect` **|**
  `http://xbrl.org/PWD/2013-05-17/table` **|**
  `http://xbrl.org/PWD/2013-08-28/table` **|**
  `http://xbrl.org/CR/2013-11-13/table` **|**
  `http://xbrl.org/PR/2013-12-18/table` **|**
  `http://xbrl.org/2014/table`

  Habilita la carga de bases de enlaces de tabla escritas con una especificación borrador previa. La validación, la resolución y el diseño de bases de enlaces de tabla se realiza, sin embargo, siguiendo la recomendación Table Linkbase 1.0 del 18 de marzo de 2014. Para habilitar la detección automática, utilice `##detect`.

- table-output

  `--table-output =` *`ARCHIVO`*
  Escribe la tabla de salida en el ***ARCHIVO*** indicado. Si se define, especifica automáticamente `--table-execution=true`.

- table-output-format

  `--table-output-format =` `xml|html`
  Especifica el formato de la tabla de salida. Valor predeterminado: `xml`.

- Catálogos y recursos globales
  - catalog

    `--catalog =` *`ARCHIVO`*
    Especifica la ruta de acceso absoluta a un archivo de catálogo que no está en el archivo de catálogo raíz instalado. El valor predeterminado es la ruta de acceso absoluta del archivo de catálogo raíz instalado (`<carpeta-instalación>\Altova`

`\RaptorXMLXBRLServer2018\etc\RootCatalog.xml`). Consulte el apartado Catálogos XML para obtener más información.

- user-catalog

  `--user-catalog =` *`ARCHIVO`*
  Especifica la ruta de acceso absoluta a un catálogo XML que debe utilizarse junto con el catálogo raíz. Consulte el apartado Catálogos XML para obtener más información.

- enable-globalresources

  `--enable-globalresources =` `true|false`
  Habilita la función de recursos globales. Valor predeterminado: `false`.
  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- globalresourceconfig [gc]

  `--gc | --globalresourceconfig =` *`VALOR`*
  Especifica la configuración activa del recurso global (y habilita los recursos globales).

- globalresourcefile [gr]

  `--gr | --globalresourcefile =` *`ARCHIVO`*
  Especifica el archivo de recursos globales (y habilita los recursos globales).

- Mensajes, errores, ayuda, tiempo de espera, versión
  - error-format

    `--error-format =` `text|shortxml|longxml`
    Especifica el formato de la salida de error. Los valores posibles son formatos de texto, XML y XML detallado (`longxml`). Valor predeterminado: `text`.

  - error-limit

    `--error-limit =` *`N`*
    Especifica el límite de errores con un valor comprendido entre `1` y `999`. El valor predeterminado es `100`. Cuando se alcanza el límite de error, se detiene la validación. Esta opción es muy práctica a la hora de limitar el uso del procesador durante la validación o transformación.

  - help

    `--help`
    Muestra el texto de ayuda para el comando. Por ejemplo `valany --h`. (Otra opción es usar el comando `help` con un argumento. Por ejemplo: `help valany`).

  - log-output

    `--log-output =` *`ARCHIVO`*
    Escribe el registro de salida en la URL de archivo indicada. Compruebe que la ILC tiene permiso de escritura en la ubicación de destino.

 © 2018 Altova GmbH

- network-timeout

  `--network-timeout =` *`VALOR`*
  Indica el tiempo de espera en segundos para operaciones remotas de entrada y salida. Valor predeterminado: `40`.

- verbose

  `--verbose =` `true|false`
  Si el valor es `true`, se genera información adicional durante la validación. Valor predeterminado es `false`.
  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- verbose-output

  `--verbose-output =` *`ARCHIVO`*
  Escribe el resultado detallado en el *`ARCHIVO`* indicado.

- version

  `--version`
  Muestra el número de versión de RaptorXML+XBRL Server. Si se utiliza con un comando, escriba la opción `--version` antes del comando.

- warning-limit

  `--warning-limit =` *`VALUE`*
  Especifica el límite de advertencia en el rango `1-65535`. El procesamiento continua si se alcanza el límite pero no se registrarán más advertencias. El valor por defecto es `100`.

3.3.4 valtaxonomypackage (taxpkg)

El comando **`valtaxonomypackage | taxpkg`** valida paquetes de taxonomías XBRL de acuerdo con lo estipulado por la especificación Taxonomy 1.0 Packages.

Windows	**`RaptorXMLXBRL valtaxonomypackage	taxpkg [opciones]`** ***`PaqueteTaxonomías`***
Linux	**`raptorxmlxbrl valtaxonomypackage	taxpkg [opciones]`** ***`PaqueteTaxonomías`***
Mac	**`raptorxmlxbrl valtaxonomypackage	taxpkg [opciones]`** ***`PaqueteTaxonomías`***

El argumento *`PaqueteTaxonomías`* es el paquete de taxonomías XBRL que se debe validar, que suele ser un archivo comprimido ZIP.

Si desea validar varios paquetes, tiene dos opciones: (i) enumerar los paquetes que se deben validar en la línea de comandos, separados por espacios o (ii) enumerar los paquetes que se deben validar en un archivo de texto (`.txt`) donde aparece un nombre de paquete por línea y dar este archivo de texto como argumento *`PaqueteTaxonomías`* junto con la opción `--listfile` con valor `true` (*ver lista de opciones más abajo*).

Ejemplos

- **`raptorxmlxbrl`** `valxtaxonomypackage c:\Test.zip`
- **`raptorxmlxbrl`** `taxpkg --listfile=true c:\ListaPaquetes.txt`
- **`raptorxmlxbrl`** `taxpkg --listfile c:\ListaPaquetes.txt`

Uso de mayúsculas/minúsculas y de barras diagonales en la línea de comandos

`RaptorXMLXBRL` *en Windows*
`raptorxmlxbrl` *en Unix (Linux, Mac)*

* Las minúsculas (`raptorxmlxbrl`) funcionan en todas las plataformas (Windows, Linux y Mac), mientras que la mezcla de mayúsculas y minúsculas (`RaptorXMLXBRL`) sólo funciona en Windows y Mac.
* En Linux y Mac utilice barras diagonales. En Windows utilice barras diagonales inversas.

Opciones

Las opciones del comando aparecen más abajo divididas en grupos. Los valores se pueden dar sin comillas excepto en estos dos casos: (i) cuando la cadena de valor contiene espacios y (ii) cuando en la descripción de la opción se indique explícitamente que es necesario el uso de comillas.

Procesamiento

listfile

`--listfile =` `true|false`
Si el valor es `true`, el argumento *`ArchivoEntrada`* del comando se entiende como un archivo de texto que contiene un nombre de archivo por línea. Otra opción es enumerar los archivos en la ILC, separados por un espacio. No obstante, recuerde que las ILC tienen un límite de caracteres. Además, no olvide que la opción `--listfile` solamente afecta a los argumentos y no a las opciones.
Valor predeterminado: `false`.
Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

recurse

`--recurse =` `true|false`
Esta opción se utiliza para seleccionar ficheros dentro de subdirectorios (incluso en archivos ZIP). Si el valor es `true`, el argumento *`ArchivoEntrada`* del comando seleccionará el fichero seleccionado también en los subdirectorios. Por ejemplo: `test.zip|zip\test.xml` seleccionará los ficheros llamados `test.xml` en todos los subdirectorios de la carpeta ZIP. Si quiere puede usar los caracteres comodín `*` y `?`.

 © 2018 Altova GmbH

Por ejemplo: `*.xml` seleccionaría todos los ficheros de la carpeta ZIP que tengan la extensión `.xml`. Valor predeterminado: `false`
Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

schema-imports

`--schema-imports =` `load-by-schemalocation | load-preferring-schemalocation | load-by-namespace | load-combining-both | license-namespace-only`
Esta opción indica el comportamiento de los elementos `xs:import`. Cada uno de estos elementos tiene un atributo opcional `namespace` y un atributo opcional `schemaLocation`: `<import namespace="unEspacioNombres" schemaLocation="unaURL">`. La opción indica si se debe cargar un documento de esquema o solo autorizar a un espacio de nombres. Si la opción indica que se debe cargar un documento de esquema, entonces indica también qué información debe utilizarse para encontrar el documento de esquema. Valor predeterminado: `load-preferring-schemalocation`.

- `load-by-schemalocation`: el valor del atributo `schemaLocation` se utiliza para buscar el esquema, teniendo en cuenta las asignaciones de catálogo. Si está presente el atributo `namespace`, se importa el espacio de nombres (con licencia).
- `load-preferring-schemalocation`: si está presente, se utiliza el atributo `schemaLocation` teniendo en cuenta las asignaciones de catálogo. Si no está presente el atributo `schemaLocation`, entonces se usa el valor del atributo `namespace` a través de las asignaciones de catálogo. Este es el **valor predeterminado**.
- `load-by-namespace`: el valor del atributo `namespace` se utiliza para buscar el esquema por medio de una asignación de catálogo.
- `load-combining-both`: si el atributo `namespace` o `schemaLocation` tiene una asignación de catálogo, entonces se usa la asignación. Si ambos atributos tienen asignaciones de catálogo, entonces es el valor de la opción `--schema-mapping` (opción XBRL y opción XML/XSD) decide qué asignación se utiliza. Si no hay ninguna asignación de catálogo, entonces se usa el atributo `schemaLocation`.
- `license-namespace-only`: se importa el espacio de nombres. No se importa el documento de esquema.

schema-mapping

`--schema-mapping =` `prefer-schemalocation | prefer-namespace`
Si se usa la ubicación y el espacio de nombres para buscar el documento de esquema, esta opción indica cuál de ellos debe ser la opción preferida durante la búsqueda en el catálogo.
Si la opción `--schemalocation-hints` o la opción `--schema-imports` tiene el valor `load-combining-both` y si las partes de espacio de nombres y URL pertinentes tienen asignaciones de catálogo, entonces el valor de la opción especifica cuál de las dos asignaciones se utiliza (la asignación del espacio de nombres o de la URL: el valor `prefer-schemalocation` se refiere a la asignación de la URL). Valor predeterminado: `prefer-schemalocation`.

schemalocation-hints

`--schemalocation-hints =` `load-by-schemalocation | load-by-namespace |`

`load-combining-both | ignore`
Determina el comportamiento predeterminado de los atributos `xsi:schemaLocation` y `xsi:noNamespaceSchemaLocation`. Indica si se debe cargar un documento de esquema y, si así es, indica qué información debe utilizarse para encontrarlo. Valor predeterminado: `load-by-schemalocation`.

- **Valor predeterminado**: `load-by-schemalocation`. Este valor toma la URL de la ubicación del esquema de los atributos `xsi:schemaLocation` y `xsi:noNamespaceSchemaLocation` de los documentos de instancia XML o XBRL.
- El valor `load-by-namespace` toma la parte de espacio de nombres del atributo `xsi:schemaLocation` y una cadena vacía en el caso del atributo `xsi:noNamespaceSchemaLocation` y encuentra el esquema por medio de una asignación de catálogo.
- Si usa el valor `load-combining-both` y el espacio de nombres o la URL tienen una asignación de catálogo, se usa dicha asignación. Si ambos tienen asignaciones de catálogo, el valor de la opción `schema-mapping` (opción XBRL y opción XML/XSD) decide qué asignación se utiliza. Si ni el espacio de nombres ni la URL tiene una asignación de catálogo, se usa la URL.
- El valor `ignore` ignora los atributos `xsi:schemaLocation` y `xsi:noNamespaceSchemaLocation`.

script

`--script =` *`ARCHIVO`*
Una vez finalizada la validación, ejecuta el script Python. Para indicar más de un script basta con agregar la opción varias veces.

script-api-version

`--api, --script-api-version =` `1|2|2.1|2.2|2.3|2.4|2.4.1|2.5`
Especifica la versión API Python que ha de usarse para el script. El valor por defecto es la última versión, actualmente **`2.45`**. En vez de de los valores `1` y `2`, también puede usar losa valores 1.0 y 2.0 respectivamente.

script-output

`--script-output =` *`FILE`*
Escribe el resultado estandard del script en el archivo nombres como *`FILE`*.

script-param

`--script-param =` *`CLAVE:VALOR`*
Parámetros definidos por el usuario a los que se puede acceder durante la ejecución de scripts Python.

Catálogos y recursos globales

catalog

`--catalog =` *`ARCHIVO`*
Especifica la ruta de acceso absoluta a un archivo de catálogo que no está en el

archivo de catálogo raíz instalado. El valor predeterminado es la ruta de acceso absoluta del archivo de catálogo raíz instalado (`<carpeta-instalación>\Altova\RaptorXMLXBRLServer2018\etc\RootCatalog.xml`). Consulte el apartado Catálogos XML para obtener más información.

- user-catalog

  `--user-catalog =` *`ARCHIVO`*
  Especifica la ruta de acceso absoluta a un catálogo XML que debe utilizarse junto con el catálogo raíz. Consulte el apartado Catálogos XML para obtener más información.

- enable-globalresources

  `--enable-globalresources =` `true|false`
  Habilita la función de recursos globales. Valor predeterminado: `false`.
  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- globalresourceconfig [gc]

  `--gc | --globalresourceconfig =` *`VALOR`*
  Especifica la configuración activa del recurso global (y habilita los recursos globales).

- globalresourcefile [gr]

  `--gr | --globalresourcefile =` *`ARCHIVO`*
  Especifica el archivo de recursos globales (y habilita los recursos globales).

- Mensajes, errores, ayuda, tiempo de espera, versión
  - error-format

    `--error-format =` `text|shortxml|longxml`
    Especifica el formato de la salida de error. Los valores posibles son formatos de texto, XML y XML detallado (`longxml`). Valor predeterminado: `text`.

  - error-limit

    `--error-limit =` *`N`*
    Especifica el límite de errores con un valor comprendido entre `1` y `999`. El valor predeterminado es `100`. Cuando se alcanza el límite de error, se detiene la validación. Esta opción es muy práctica a la hora de limitar el uso del procesador durante la validación o transformación.

  - help

    `--help`
    Muestra el texto de ayuda para el comando. Por ejemplo `valany --h`. (Otra opción es usar el comando `help` con un argumento. Por ejemplo: `help valany`).

  - log-output

    `--log-output =` *`ARCHIVO`*
    Escribe el registro de salida en la URL de archivo indicada. Compruebe que la ILC tiene

permiso de escritura en la ubicación de destino.

- network-timeout

  `--network-timeout =` *`VALOR`*
  Indica el tiempo de espera en segundos para operaciones remotas de entrada y salida. Valor predeterminado: `40`.

- verbose

  `--verbose =` `true|false`
  Si el valor es `true`, se genera información adicional durante la validación. Valor predeterminado es `false`.
  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- verbose-output

  `--verbose-output =` *`ARCHIVO`*
  Escribe el resultado detallado en el *`ARCHIVO`* indicado.

- version

  `--version`
  Muestra el número de versión de RaptorXML+XBRL Server. Si se utiliza con un comando, escriba la opción `--version` antes del comando.

- warning-limit

  `--warning-limit =` *`VALUE`*
  Especifica el límite de advertencia en el rango `1-65535`. El procesamiento continua si se alcanza el límite pero no se registrarán más advertencias. El valor por defecto es `100`.

3.3.5 valxbrlver

El comando **`valxbrlver`** valida informes de versionado XBRL de acuerdo con lo estipulado en la especificación XBRL Versioning 1.0.

Windows	**`RaptorXMLXBRL valxbrlver [opciones]`** ***`ArchivoEntrada`***
Linux	**`raptorxmlxbrl valxbrlver [opciones]`** ***`ArchivoEntrada`***
Mac	**`raptorxmlxbrl valxbrlver [opciones]`** ***`ArchivoEntrada`***

El argumento *`ArchivoEntrada`* es el informe de versionado XBRL, que es un documento XML (con la extensión `.xml`). Si desea validar varios documentos, tiene dos opciones: (i) enumerar los archivos que se deben validar en la línea de comandos, separados por espacios o (ii) enumerar los archivos que se deben validar en un archivo de texto (`.txt`) donde aparece un nombre de

archivo por línea y dar este archivo de texto como argumento *ArchivoEntrada* junto con la opción `--listfile` con valor `true` (*ver lista de opciones más abajo*).

Ejemplos

- **raptorxmlxbrl** valxbrlver c:\MiInformeVersionadoXBRL.xml
- **raptorxmlxbrl** valxbrlver c:\MisInformesVersionadoXBRL.txt

Uso de mayúsculas/minúsculas y de barras diagonales en la línea de comandos

RaptorXMLXBRL *en Windows*
raptorxmlxbrl *en Unix (Linux, Mac)*

* Las minúsculas (`raptorxmlxbrl`) funcionan en todas las plataformas (Windows, Linux y Mac), mientras que la mezcla de mayúsculas y minúsculas (`RaptorXMLXBRL`) sólo funciona en Windows y Mac.
* En Linux y Mac utilice barras diagonales. En Windows utilice barras diagonales inversas.

Opciones

Las opciones del comando aparecen más abajo divididas en grupos. Los valores se pueden dar sin comillas excepto en estos dos casos: (i) cuando la cadena de valor contiene espacios y (ii) cuando en la descripción de la opción se indique explícitamente que es necesario el uso de comillas.

Mensajes, errores, ayuda, tiempo de espera, versión

error-format

--error-format = `text|shortxml|longxml`
Especifica el formato de la salida de error. Los valores posibles son formatos de texto, XML y XML detallado (`longxml`). Valor predeterminado: `text`.

error-limit

--error-limit = *N*
Especifica el límite de errores con un valor comprendido entre `1` y `999`. El valor predeterminado es `100`. Cuando se alcanza el límite de error, se detiene la validación. Esta opción es muy práctica a la hora de limitar el uso del procesador durante la validación o transformación.

help

--help
Muestra el texto de ayuda para el comando. Por ejemplo `valany --h`. (Otra opción es usar el comando `help` con un argumento. Por ejemplo: `help valany`).

log-output

--log-output = *ARCHIVO*
Escribe el registro de salida en la URL de archivo indicada. Compruebe que la ILC tiene permiso de escritura en la ubicación de destino.

- network-timeout

  --network-timeout = *VALOR*
  Indica el tiempo de espera en segundos para operaciones remotas de entrada y salida. Valor predeterminado: `40`.

- verbose

  --verbose = `true|false`
  Si el valor es `true`, se genera información adicional durante la validación. Valor predeterminado es `false`.
  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- verbose-output

  --verbose-output = *ARCHIVO*
  Escribe el resultado detallado en el *ARCHIVO* indicado.

- version

  --version
  Muestra el número de versión de RaptorXML+XBRL Server. Si se utiliza con un comando, escriba la opción `--version` antes del comando.

- warning-limit

  --warning-limit = *VALUE*
  Especifica el límite de advertencia en el rango `1-65535`. El procesamiento continua si se alcanza el límite pero no se registrarán más advertencias. El valor por defecto es `100`.

- Catálogos y recursos globales
  - catalog

    --catalog = *ARCHIVO*
    Especifica la ruta de acceso absoluta a un archivo de catálogo que no está en el archivo de catálogo raíz instalado. El valor predeterminado es la ruta de acceso absoluta del archivo de catálogo raíz instalado (`<carpeta-instalación>\Altova\RaptorXMLXBRLServer2018\etc\RootCatalog.xml`). Consulte el apartado Catálogos XML para obtener más información.

  - user-catalog

    --user-catalog = *ARCHIVO*
    Especifica la ruta de acceso absoluta a un catálogo XML que debe utilizarse junto con el catálogo raíz. Consulte el apartado Catálogos XML para obtener más información.

  - enable-globalresources

    --enable-globalresources = `true|false`
    Habilita la función de recursos globales. Valor predeterminado: `false`.
    Nota: si no se especifica un valor para la opción, el valor booleano de la opción se

 © 2018 Altova GmbH

establece en `true`.

- globalresourceconfig [gc]

  `--gc | --globalresourceconfig =` *`VALOR`*
  Especifica la configuración activa del recurso global (y habilita los recursos globales).

- globalresourcefile [gr]

  `--gr | --globalresourcefile =` *`ARCHIVO`*
  Especifica el archivo de recursos globales (y habilita los recursos globales).

- XML Schema
  - schema-imports

    `--schema-imports =` `load-by-schemalocation | load-preferring-schemalocation | load-by-namespace | load-combining-both | license-namespace-only`
    Esta opción indica el comportamiento de los elementos `xs:import`. Cada uno de estos elementos tiene un atributo opcional `namespace` y un atributo opcional `schemaLocation`: `<import namespace="unEspacioNombres" schemaLocation="unaURL">`. La opción indica si se debe cargar un documento de esquema o solo autorizar a un espacio de nombres. Si la opción indica que se debe cargar un documento de esquema, entonces indica también qué información debe utilizarse para encontrar el documento de esquema. Valor predeterminado: `load-preferring-schemalocation`.

    - `load-by-schemalocation`: el valor del atributo `schemaLocation` se utiliza para buscar el esquema, teniendo en cuenta las asignaciones de catálogo. Si está presente el atributo `namespace`, se importa el espacio de nombres (con licencia).
    - `load-preferring-schemalocation`: si está presente, se utiliza el atributo `schemaLocation` teniendo en cuenta las asignaciones de catálogo. Si no está presente el atributo `schemaLocation`, entonces se usa el valor del atributo `namespace` a través de las asignaciones de catálogo. Este es el **valor predeterminado**.
    - `load-by-namespace`: el valor del atributo `namespace` se utiliza para buscar el esquema por medio de una asignación de catálogo.
    - `load-combining-both`: si el atributo `namespace` o `schemaLocation` tiene una asignación de catálogo, entonces se usa la asignación. Si ambos atributos tienen asignaciones de catálogo, entonces es el valor de la opción `--schema-mapping` (opción XBRL y opción XML/XSD) decide qué asignación se utiliza. Si no hay ninguna asignación de catálogo, entonces se usa el atributo `schemaLocation`.
    - `license-namespace-only`: se importa el espacio de nombres. No se importa el documento de esquema.

  - schema-mapping

    `--schema-mapping =` `prefer-schemalocation | prefer-namespace`
    Si se usa la ubicación y el espacio de nombres para buscar el documento de esquema, esta opción indica cuál de ellos debe ser la opción preferida durante la búsqueda en el catálogo.
    Si la opción `--schemalocation-hints` o la opción `--schema-imports` tiene el valor

`load-combining-both` y si las partes de espacio de nombres y URL pertinentes tienen asignaciones de catálogo, entonces el valor de la opción especifica cuál de las dos asignaciones se utiliza (la asignación del espacio de nombres o de la URL: el valor `prefer-schemalocation` se refiere a la asignación de la URL). Valor predeterminado: `prefer-schemalocation`.

- schemalocation-hints

`--schemalocation-hints =` `load-by-schemalocation | load-by-namespace | load-combining-both | ignore`
Determina el comportamiento predeterminado de los atributos `xsi:schemaLocation` y `xsi:noNamespaceSchemaLocation`. Indica si se debe cargar un documento de esquema y, si así es, indica qué información debe utilizarse para encontrarlo. Valor predeterminado: `load-by-schemalocation`.

- **Valor predeterminado**: `load-by-schemalocation`. Este valor toma la URL de la ubicación del esquema de los atributos `xsi:schemaLocation` y `xsi:noNamespaceSchemaLocation` de los documentos de instancia XML o XBRL.
- El valor `load-by-namespace` toma la parte de espacio de nombres del atributo `xsi:schemaLocation` y una cadena vacía en el caso del atributo `xsi:noNamespaceSchemaLocation` y encuentra el esquema por medio de una asignación de catálogo.
- Si usa el valor `load-combining-both` y el espacio de nombres o la URL tienen una asignación de catálogo, se usa dicha asignación. Si ambos tienen asignaciones de catálogo, el valor de la opción `schema-mapping` (opción XBRL y opción XML/XSD) decide qué asignación se utiliza. Si ni el espacio de nombres ni la URL tiene una asignación de catálogo, se usa la URL.
- El valor `ignore` ignora los atributos `xsi:schemaLocation` y `xsi:noNamespaceSchemaLocation`.

- Opciones varias
  - listfile

`--listfile =` `true|false`
Si el valor es `true`, el argumento *`ArchivoEntrada`* del comando se entiende como un archivo de texto que contiene un nombre de archivo por línea. Otra opción es enumerar los archivos en la ILC, separados por un espacio. No obstante, recuerde que las ILC tienen un límite de caracteres. Además, no olvide que la opción `--listfile` solamente afecta a los argumentos y no a las opciones.
Valor predeterminado: `false`.
Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

  - xinclude

`--xinclude =` `true|false`
Habilita la compatibilidad con inclusiones XML (XInclude). Si el valor es `false`, los elementos XInclude `include` se ignoran. Valor predeterminado: `false`.
Nota: si no se especifica un valor para la opción, el valor booleano de la opción se

establece en `true`.

3.4 Comandos XSLT

Estos son los comandos XSLT:

- xslt: para transformar documentos XML con un documento XSLT
- valxslt: para validar documentos XSLT

Los argumentos y opciones de cada comando se describen en los siguientes apartados.

3.4.1 xslt

El comando **xslt** toma un archivo XSLT como único argumento y lo utiliza para transformar un archivo XML de entrada y generar un archivo de salida. Los archivos de entrada y salida se especifican como opciones.

Windows	**RaptorXMLXBRL xslt [opciones]** ***Archivo-XSLT***
Linux	**raptorxmlxbrl xslt [opciones]** ***Archivo-XSLT***
Mac	**raptorxmlxbrl xslt [opciones]** ***Archivo-XSLT***

El argumento *Archivo-XSLT* es la ruta de acceso y el nombre del archivo XSLT que se debe usar para la transformación. Se necesita un archivo XML de entrada (--input) o el punto de entrada de una plantilla con nombre (--template-entry-point). Si no especifica la opción del documento de salida --output, se genera un resultado estándar. Puede usar XSLT 1.0, 2.0 o 3.0. La versión predeterminada es XSLT 3.0.

Ejemplos

- **raptorxmlxbrl** xslt --input=c:\Test.xml --output=c:\Salida.xml c:\Test.xslt
- **raptorxmlxbrl** xslt --template-entry-point=PlantillaInicial --output=c:\Salida.xml c:\Test.xslt
- **raptorxmlxbrl** xslt --input=c:\Test.xml --output=c:\Salida.xml --param=date://node[1]/@att1 --p=title:'cadenasinespacios' --param=title:"'cadena con espacios'" --p=amount:456 c:\Test.xslt

Uso de mayúsculas/minúsculas y de barras diagonales en la línea de comandos

RaptorXMLXBRL *en Windows*
raptorxmlxbrl *en Unix (Linux, Mac)*

* Las minúsculas (raptorxmlxbrl) funcionan en todas las plataformas (Windows, Linux y Mac), mientras que la mezcla de mayúsculas y minúsculas (RaptorXMLXBRL) sólo funciona en Windows y Mac.
* En Linux y Mac utilice barras diagonales. En Windows utilice barras diagonales inversas.

 © 2018 Altova GmbH

Opciones

Las opciones del comando aparecen más abajo divididas en grupos. Los valores se pueden dar sin comillas excepto en estos dos casos: (i) cuando la cadena de valor contiene espacios y (ii) cuando en la descripción de la opción se indique explícitamente que es necesario el uso de comillas.

- Procesamiento XSLT
  - indent-characters

    --indent-characters = *VALOR*

    Especifica la cadena de caracteres que debe usarse como sangría.

  - function-param

    --function-param = *VALUE*

    Especifica las funciones que se transmiten a la función inicial. Para especificar más de una función, use la opción varias veces, pero recuerde que el orden es importante.

  - global-context-item

    --global-context-item = *VALUE*

    Especifica el componente de contexto que se debe usar para evaluar las variables globales.

  - initial-function

    --initial-function = *VALUE*

    El nombre de una función que se ejecuta como punto de entrada de la transformación.

  - initial-match-selection

    --initial-match-selection = *VALUE*

    Especifica el valor (secuencia) de la selección de coincidencia inicial.

  - initial-mode, template-mode

    --initial-mode, --template-mode = *VALOR*

    Especifica el modo de plantilla que debe usarse para la transformación.

  - initial-template, template-entry-point

    --initial-template, --template-entry-point = *VALOR*

    Indica el nombre de una plantilla con nombre de la hoja de estilos XSLT que sirve de punto de entrada de la transformación.

  - input

    --input = *ARCHIVO*

    La URL del archivo XML que se debe transformar.

  - output, xsltoutput

    output = *ARCHIVO*, **xsltoutput =** *ARCHIVO*

La URL del archivo de salida principal. Por ejemplo, en caso de tener varios archivos HTML de salida, el archivo de salida principal será la ubicación del archivo HTML del punto de entrada. Los demás archivos de salida (como archivos de imagen generados) se indican con `xslt-additional-output-files`. Si no se especifica la opción `--output` ni la opción `--xsloutput`, se genera un resultado estándar.

param [p]

`--p | --param =` *`CLAVE:VALOR`*

XQuery

Especifica el valor de un parámetro externo. En el documento XQuery los parámetros externos se declaran con la declaración `declare variable` seguida de un nombre de variable y después la palabra clave `external` seguida del punto y coma final. Por ejemplo: `declare variable $foo as xs:string external;`

Al usar la palabra clave `external`, `$foo` se convierte en parámetro externo y su valor se pasa en tiempo de ejecución desde una fuente externa. El parámetro externo recibe un valor con el comando de la ILC. Por ejemplo: `--param=foo:'MiNombre'`

En la descripción anterior, *`CLAVE`* es el nombre de parámetro externo y *`VALOR`* es su valor, dado como expresión XPath. Los nombres de parámetro utilizados en la ILC deben declararse en el documento XQuery. Si se pasan valores a varios parámetros externos en la ILC, cada parámetro debe llevar una opción `--param` distinta. Si la expresión XPath contiene espacios, entonces debe estar entre comillas dobles.

XSLT

Especifica un parámetro global de la hoja de estilos. *`CLAVE`* es el nombre del parámetro y *`VALOR`* es una expresión XPath que da un valor al parámetro. Los nombres de parámetro utilizados en la ILC deben declararse en la hoja de estilos. Si usa más de un parámetro, debe usar el modificador `--param` antes de cada parámetro. Si la expresión XPath incluye espacios, entonces debe ir entre comillas dobles, tanto si el espacio está en la expresión propiamente dicha o en un literal de cadena de la expresión. Por ejemplo:

```
raptorxmlxbrl xslt --input=c:\Test.xml --output=c:\Salida.xml --
param=date://node[1]/@att1 --p=title:cadenasinespacios --
param=title:"'cadena con espacios'" --p=amount:456 c:\Test.xslt
```

streaming-serialization-enabled

`--streaming-serialization-enabled =` `true|false`

Habilita la serialización de secuencias de datos. Valor predeterminado: `true`.

Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

template-param

`--template-param =` *`KEY:VALUE`*

Especifica los parámetros que pasan únicamente a la plantilla inicial (y no a llamadas

descendientes de la plantilla). Para especificar varios parámetros, use esta opción una vez por cada uno de ellos.

- tunnel-param

  `--tunnel-param =` *`KEY:VALUE`*
  Especifica parámetros que pasan a la plantilla inicial y a llamadas descendientes de la plantilla. Para especificar varios parámetros, use esta opción una vez por cada uno de ellos.

- xslt-version

  `--xslt-version =` `1|1.0|2|2.0|3|3.0|3.1`
  Especifica si el procesador XSLT debe usar XSLT 1.0, XSLT 2.0 o XSLT 3.0. Valor predeterminado: `3`

- Instancias y esquemas XML
  - load-xml-with-psvi

    `--load-xml-with-psvi =` `true|false`
    Habilita la validación de archivos XML de entrada y genera información posterior a la validación. Valor predeterminado: `true`.

  - schema-imports

    `--schema-imports =` `load-by-schemalocation | load-preferring-schemalocation | load-by-namespace | load-combining-both | license-namespace-only`
    Esta opción indica el comportamiento de los elementos `xs:import`. Cada uno de estos elementos tiene un atributo opcional `namespace` y un atributo opcional `schemaLocation`: `<import namespace="unEspacioNombres" schemaLocation="unaURL">`. La opción indica si se debe cargar un documento de esquema o solo autorizar a un espacio de nombres. Si la opción indica que se debe cargar un documento de esquema, entonces indica también qué información debe utilizarse para encontrar el documento de esquema. Valor predeterminado: `load-preferring-schemalocation`.

    - `load-by-schemalocation`: el valor del atributo `schemaLocation` se utiliza para buscar el esquema, teniendo en cuenta las asignaciones de catálogo. Si está presente el atributo `namespace`, se importa el espacio de nombres (con licencia).
    - `load-preferring-schemalocation`: si está presente, se utiliza el atributo `schemaLocation` teniendo en cuenta las asignaciones de catálogo. Si no está presente el atributo `schemaLocation`, entonces se usa el valor del atributo `namespace` a través de las asignaciones de catálogo. Este es el **valor predeterminado**.
    - `load-by-namespace`: el valor del atributo `namespace` se utiliza para buscar el esquema por medio de una asignación de catálogo.
    - `load-combining-both`: si el atributo `namespace` o `schemaLocation` tiene una asignación de catálogo, entonces se usa la asignación. Si ambos atributos tienen asignaciones de catálogo, entonces es el valor de la opción `--schema-mapping` (opción XBRL y opción XML/XSD) decide qué asignación se utiliza. Si no hay ninguna asignación de catálogo, entonces se usa el atributo `schemaLocation`.

- `license-namespace-only`: se importa el espacio de nombres. No se importa el documento de esquema.

schemalocation-hints

`--schemalocation-hints =` `load-by-schemalocation | load-by-namespace | load-combining-both | ignore`
Determina el comportamiento predeterminado de los atributos `xsi:schemaLocation` y `xsi:noNamespaceSchemaLocation`. Indica si se debe cargar un documento de esquema y, si así es, indica qué información debe utilizarse para encontrarlo. Valor predeterminado: `load-by-schemalocation`.

- **Valor predeterminado**: `load-by-schemalocation`. Este valor toma la URL de la ubicación del esquema de los atributos `xsi:schemaLocation` y `xsi:noNamespaceSchemaLocation` de los documentos de instancia XML o XBRL.

- El valor `load-by-namespace` toma la parte de espacio de nombres del atributo `xsi:schemaLocation` y una cadena vacía en el caso del atributo `xsi:noNamespaceSchemaLocation` y encuentra el esquema por medio de una asignación de catálogo.

- Si usa el valor `load-combining-both` y el espacio de nombres o la URL tienen una asignación de catálogo, se usa dicha asignación. Si ambos tienen asignaciones de catálogo, el valor de la opción `schema-mapping` (opción XBRL y opción XML/XSD) decide qué asignación se utiliza. Si ni el espacio de nombres ni la URL tiene una asignación de catálogo, se usa la URL.

- El valor `ignore` ignora los atributos `xsi:schemaLocation` y `xsi:noNamespaceSchemaLocation`.

schema-mapping

`--schema-mapping =` `prefer-schemalocation | prefer-namespace`
Si se usa la ubicación y el espacio de nombres para buscar el documento de esquema, esta opción indica cuál de ellos debe ser la opción preferida durante la búsqueda en el catálogo.
Si la opción `--schemalocation-hints` o la opción `--schema-imports` tiene el valor `load-combining-both` y si las partes de espacio de nombres y URL pertinentes tienen asignaciones de catálogo, entonces el valor de la opción especifica cuál de las dos asignaciones se utiliza (la asignación del espacio de nombres o de la URL: el valor `prefer-schemalocation` se refiere a la asignación de la URL). Valor predeterminado: `prefer-schemalocation`.

xinclude

`--xinclude =` `true|false`
Habilita la compatibilidad con inclusiones XML (XInclude). Si el valor es `false`, los elementos XInclude `include` se ignoran. Valor predeterminado: `false`.
Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

xml-mode

`--xml-mode =` `wf|id|valid`

 © 2018 Altova GmbH

Especifica el modo de procesamiento XML que debe utilizarse para el documento de instancia XML: `wf`=comprobación de formato; `id`=comprobación de formato con ID/IDREF; `valid`=validación. Valor predeterminado: `wf`. Recuerde que el valor `valid` exige que cada documento de instancia que se cargue durante el procesamiento haga referencia a una DTD. Si no existe ninguna DTD, se generará un error.

- xml-validation-error-as-warning

  `--xml-validation-error-as-warning =` `true|false`
  Si es `true`, tratar los errores de validación como advertencias. Si los errores se tratan como advertencias, el procesamiento adicional como la transformación XSLT, proseguirá independientemente de los errores. Por defecto es `false`.

- xsd-version

  `--xsd-version =` `1.0|1.1|detect`

  Especifica qué versión de la especificación Schema Definition Language (XSD) del W3C se debe usar. Valor predeterminado: `1.0`.

  Esta opción también puede ser útil si quiere ver en qué aspectos no es compatible un esquema 1.0 con la especificación 1.1. El valor `detect` es una característica de Altova. Permite detectar la versión del esquema XML (`1.0` o `1.1`) leyendo el valor del atributo `vc:minVersion` del elemento `<xs:schema>` del documento. Si el valor del atributo `@vc:minVersion` es `1.1`, se entiende que la versión del esquema es `1.1`. Si el atributo tiene otro valor que no sea `1.1` (o si no está presente el atributo `@vc:minVersion`), se entiende que la versión del esquema es `1.0`.

- Catálogos y recursos globales
  - catalog

    `--catalog =` *`ARCHIVO`*
    Especifica la ruta de acceso absoluta a un archivo de catálogo que no está en el archivo de catálogo raíz instalado. El valor predeterminado es la ruta de acceso absoluta del archivo de catálogo raíz instalado (`<carpeta-instalación>\Altova\RaptorXMLXBRLServer2018\etc\RootCatalog.xml`). Consulte el apartado Catálogos XML para obtener más información.

  - user-catalog

    `--user-catalog =` *`ARCHIVO`*
    Especifica la ruta de acceso absoluta a un catálogo XML que debe utilizarse junto con el catálogo raíz. Consulte el apartado Catálogos XML para obtener más información.

  - enable-globalresources

    `--enable-globalresources =` `true|false`
    Habilita la función de recursos globales. Valor predeterminado: `false`.
    Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

  - globalresourceconfig [gc]

--gc | --globalresourceconfig = *VALOR*
Especifica la configuración activa del recurso global (y habilita los recursos globales).

- globalresourcefile [gr]

--gr | --globalresourcefile = *ARCHIVO*
Especifica el archivo de recursos globales (y habilita los recursos globales).

- Extensiones

Estas opciones definen cómo se gestionan las funciones de extensión especiales disponibles en la edición Enterprise Edition de varios productos de Altova (como XMLSpy Enterprise Edition). Su uso se describe detalladamente en el manual del usuario de cada producto.

- chartext-disable

--chartext-disable = `true|false`
Deshabilita las extensiones de gráficos. Valor predeterminado: `false`.
Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- dotnetext-disable

--dotnetext-disable = `true|false`
Deshabilita las extensiones .NET. Valor predeterminado: `false`.
Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- jvm-location

--jvm-location = *ARCHIVO*
ARCHIVO especifica la ubicación del equipo virtual Java (*DLL* en Windows, objeto *compartido (shared object)* en Linux. El JVM se necesita si usa las funciones de extensión Java en su código XSLT/XQuery. El valor predeterminado es `false`.

- javaext-barcode-location

--javaext-barcode-location = *ARCHIVO*
Especifica la ruta de acceso de la carpeta que contiene el archivo de extensión de código de barras `AltovaBarcodeExtension.jar`. La ruta de acceso debe darse en uno de estos formatos:

- Un URI de archivo (ejemplo: `--javaext-barcode-location="file:///C:/Archivos de programa/Altova/RaptorXMLXBRLServer2018/etc/jar/"`)
- Una ruta de acceso Windows con caracteres de escape para las barras diagonales inversas (ejemplo: `--javaext-barcode-location="C:\\Archivos de programa\\Altova\\RaptorXMLXBRLServer2018\\etc\\jar\\"`)

- javaext-disable

--javaext-disable = `true|false`
Deshabilita las extensiones Java. Valor predeterminado: `false`.
Nota: si no se especifica un valor para la opción, el valor booleano de la opción se

 © 2018 Altova GmbH

establece en `true`.

- Mensajes, errores, ayuda, tiempo de espera, versión
  - error-format

    `--error-format =` `text|shortxml|longxml`
    Especifica el formato de la salida de error. Los valores posibles son formatos de texto, XML y XML detallado (`longxml`). Valor predeterminado: `text`.

  - error-limit

    `--error-limit =` *`N`*
    Especifica el límite de errores con un valor comprendido entre `1` y `999`. El valor predeterminado es `100`. Cuando se alcanza el límite de error, se detiene la validación. Esta opción es muy práctica a la hora de limitar el uso del procesador durante la validación o transformación.

  - help

    `--help`
    Muestra el texto de ayuda para el comando. Por ejemplo `valany --h`. (Otra opción es usar el comando `help` con un argumento. Por ejemplo: `help valany`).

  - log-output

    `--log-output =` *`ARCHIVO`*
    Escribe el registro de salida en la URL de archivo indicada. Compruebe que la ILC tiene permiso de escritura en la ubicación de destino.

  - network-timeout

    `--network-timeout =` *`VALOR`*
    Indica el tiempo de espera en segundos para operaciones remotas de entrada y salida. Valor predeterminado: `40`.

  - verbose

    `--verbose =` `true|false`
    Si el valor es `true`, se genera información adicional durante la validación. Valor predeterminado es `false`.
    Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

  - verbose-output

    `--verbose-output =` *`ARCHIVO`*
    Escribe el resultado detallado en el *`ARCHIVO`* indicado.

  - version

    `--version`
    Muestra el número de versión de RaptorXML+XBRL Server. Si se utiliza con un comando, escriba la opción `--version` antes del comando.

- warning-limit

  --warning-limit = *VALUE*
  Especifica el límite de advertencia en el rango `1-65535`. El procesamiento continua si se alcanza el límite pero no se registrarán más advertencias. El valor por defecto es `100`.

3.4.2 valxslt

El comando **valxslt** toma un archivo XSLT como único argumento y lo valida.

Windows	**RaptorXMLXBRL valxslt [opciones]** ***Archivo-XSLT***
Linux	**raptorxmlxbrl valxslt [opciones]** ***Archivo-XSLT***
Mac	**raptorxmlxbrl valxslt [opciones]** ***Archivo-XSLT***

El argumento *Archivo-XSLT* es la ruta de acceso y el nombre del archivo XSLT que debe validarse. El archivo se valida con la especificación XSLT 1.0, 2.0 o 3.0. La versión predeterminada es XSLT 3.0.

Ejemplos

- **raptorxmlxbrl** `valxslt c:\Test.xslt`
- **raptorxmlxbrl** `valxslt --xslt-version=2 c:\Test.xslt`

- Uso de mayúsculas/minúsculas y de barras diagonales en la línea de comandos

  RaptorXMLXBRL *en Windows*
  raptorxmlxbrl *en Unix (Linux, Mac)*

  * Las minúsculas (`raptorxmlxbrl`) funcionan en todas las plataformas (Windows, Linux y Mac), mientras que la mezcla de mayúsculas y minúsculas (`RaptorXMLXBRL`) sólo funciona en Windows y Mac.
  * En Linux y Mac utilice barras diagonales. En Windows utilice barras diagonales inversas.

Opciones

Las opciones del comando aparecen más abajo divididas en grupos. Los valores se pueden dar sin comillas excepto en estos dos casos: (i) cuando la cadena de valor contiene espacios y (ii) cuando en la descripción de la opción se indique explícitamente que es necesario el uso de comillas.

- Procesamiento XSLT

 © 2018 Altova GmbH

- initial-template, template-entry-point

  --initial-template, --template-entry-point = *VALOR*
  Indica el nombre de una plantilla con nombre de la hoja de estilos XSLT que sirve de punto de entrada de la transformación.

- initial-mode, template-mode

  --initial-mode, --template-mode = *VALOR*
  Especifica el modo de plantilla que debe usarse para la transformación.

- xslt-version

  --xslt-version = `1|1.0|2|2.0|3|3.0|3.1`
  Especifica si el procesador XSLT debe usar XSLT 1.0, XSLT 2.0 o XSLT 3.0. Valor predeterminado: `3`

- Instancias y esquemas XML
  - load-xml-with-psvi

    --load-xml-with-psvi = `true|false`
    Habilita la validación de archivos XML de entrada y genera información posterior a la validación. Valor predeterminado: `true`.

  - schema-imports

    --schema-imports = `load-by-schemalocation | load-preferring-schemalocation | load-by-namespace | load-combining-both | license-namespace-only`
    Esta opción indica el comportamiento de los elementos `xs:import`. Cada uno de estos elementos tiene un atributo opcional `namespace` y un atributo opcional `schemaLocation`: `<import namespace="unEspacioNombres" schemaLocation="unaURL">`. La opción indica si se debe cargar un documento de esquema o solo autorizar a un espacio de nombres. Si la opción indica que se debe cargar un documento de esquema, entonces indica también qué información debe utilizarse para encontrar el documento de esquema. Valor predeterminado: `load-preferring-schemalocation`.

    - `load-by-schemalocation`: el valor del atributo `schemaLocation` se utiliza para buscar el esquema, teniendo en cuenta las asignaciones de catálogo. Si está presente el atributo `namespace`, se importa el espacio de nombres (con licencia).
    - `load-preferring-schemalocation`: si está presente, se utiliza el atributo `schemaLocation` teniendo en cuenta las asignaciones de catálogo. Si no está presente el atributo `schemaLocation`, entonces se usa el valor del atributo `namespace` a través de las asignaciones de catálogo. Este es el **valor predeterminado**.
    - `load-by-namespace`: el valor del atributo `namespace` se utiliza para buscar el esquema por medio de una asignación de catálogo.
    - `load-combining-both`: si el atributo `namespace` o `schemaLocation` tiene una asignación de catálogo, entonces se usa la asignación. Si ambos atributos tienen asignaciones de catálogo, entonces es el valor de la opción `--schema-mapping` (opción XBRL y opción XML/XSD) decide qué asignación se utiliza. Si no hay ninguna asignación de catálogo, entonces se usa el atributo `schemaLocation`.

- `license-namespace-only`: se importa el espacio de nombres. No se importa el documento de esquema.

schemalocation-hints

`--schemalocation-hints =` `load-by-schemalocation | load-by-namespace | load-combining-both | ignore`
Determina el comportamiento predeterminado de los atributos `xsi:schemaLocation` y `xsi:noNamespaceSchemaLocation`. Indica si se debe cargar un documento de esquema y, si así es, indica qué información debe utilizarse para encontrarlo. Valor predeterminado: `load-by-schemalocation`.

- **Valor predeterminado**: `load-by-schemalocation`. Este valor toma la URL de la ubicación del esquema de los atributos `xsi:schemaLocation` y `xsi:noNamespaceSchemaLocation` de los documentos de instancia XML o XBRL.

- El valor `load-by-namespace` toma la parte de espacio de nombres del atributo `xsi:schemaLocation` y una cadena vacía en el caso del atributo `xsi:noNamespaceSchemaLocation` y encuentra el esquema por medio de una asignación de catálogo.

- Si usa el valor `load-combining-both` y el espacio de nombres o la URL tienen una asignación de catálogo, se usa dicha asignación. Si ambos tienen asignaciones de catálogo, el valor de la opción `schema-mapping` (opción XBRL y opción XML/XSD) decide qué asignación se utiliza. Si ni el espacio de nombres ni la URL tiene una asignación de catálogo, se usa la URL.

- El valor `ignore` ignora los atributos `xsi:schemaLocation` y `xsi:noNamespaceSchemaLocation`.

schema-mapping

`--schema-mapping =` `prefer-schemalocation | prefer-namespace`
Si se usa la ubicación y el espacio de nombres para buscar el documento de esquema, esta opción indica cuál de ellos debe ser la opción preferida durante la búsqueda en el catálogo.
Si la opción `--schemalocation-hints` o la opción `--schema-imports` tiene el valor `load-combining-both` y si las partes de espacio de nombres y URL pertinentes tienen asignaciones de catálogo, entonces el valor de la opción especifica cuál de las dos asignaciones se utiliza (la asignación del espacio de nombres o de la URL: el valor `prefer-schemalocation` se refiere a la asignación de la URL). Valor predeterminado: `prefer-schemalocation`.

xinclude

`--xinclude =` `true|false`
Habilita la compatibilidad con inclusiones XML (XInclude). Si el valor es `false`, los elementos XInclude `include` se ignoran. Valor predeterminado: `false`.
Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

xml-mode

`--xml-mode =` `wf|id|valid`

 © 2018 Altova GmbH

Especifica el modo de procesamiento XML que debe utilizarse para el documento de instancia XML: `wf`=comprobación de formato; `id`=comprobación de formato con ID/IDREF; `valid`=validación. Valor predeterminado: `wf`. Recuerde que el valor `valid` exige que cada documento de instancia que se cargue durante el procesamiento haga referencia a una DTD. Si no existe ninguna DTD, se generará un error.

- xml-mode-for-schemas

  `--xml-mode-for-schemas =` `wf|id|valid`

  Especifica el modo de procesamiento XML que debe utilizarse para el documento de instancia XML: `wf`=comprobación de formato; `id`=comprobación de formato con ID/IDREF; `valid`=validación. Valor predeterminado: `wf`. Recuerde que el valor `valid` exige que cada documento de esquema que se cargue durante el procesamiento haga referencia a una DTD. Si no existe ninguna DTD, se generará un error.

- xsd-version

  `--xsd-version =` `1.0|1.1|detect`

  Especifica qué versión de la especificación Schema Definition Language (XSD) del W3C se debe usar. Valor predeterminado: `1.0`.

  Esta opción también puede ser útil si quiere ver en qué aspectos no es compatible un esquema 1.0 con la especificación 1.1. El valor `detect` es una característica de Altova. Permite detectar la versión del esquema XML (`1.0` o `1.1`) leyendo el valor del atributo `vc:minVersion` del elemento `<xs:schema>` del documento. Si el valor del atributo `@vc:minVersion` es `1.1`, se entiende que la versión del esquema es `1.1`. Si el atributo tiene otro valor que no sea `1.1` (o si no está presente el atributo `@vc:minVersion`), se entiende que la versión del esquema es `1.0`.

- Catálogos y recursos globales
  - catalog

    `--catalog =` *`ARCHIVO`*

    Especifica la ruta de acceso absoluta a un archivo de catálogo que no está en el archivo de catálogo raíz instalado. El valor predeterminado es la ruta de acceso absoluta del archivo de catálogo raíz instalado (`<carpeta-instalación>\Altova\RaptorXMLXBRLServer2018\etc\RootCatalog.xml`). Consulte el apartado Catálogos XML para obtener más información.

  - user-catalog

    `--user-catalog =` *`ARCHIVO`*

    Especifica la ruta de acceso absoluta a un catálogo XML que debe utilizarse junto con el catálogo raíz. Consulte el apartado Catálogos XML para obtener más información.

  - enable-globalresources

    `--enable-globalresources =` `true|false`

    Habilita la función de recursos globales. Valor predeterminado: `false`.

    Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- globalresourceconfig [gc]

  --gc | --globalresourceconfig = *VALOR*
  Especifica la configuración activa del recurso global (y habilita los recursos globales).

- globalresourcefile [gr]

  --gr | --globalresourcefile = *ARCHIVO*
  Especifica el archivo de recursos globales (y habilita los recursos globales).

- Extensiones

  Estas opciones definen cómo se gestionan las funciones de extensión especiales disponibles en la edición Enterprise Edition de varios productos de Altova (como XMLSpy Enterprise Edition). Su uso se describe detalladamente en el manual del usuario de cada producto.

  - chartext-disable

    --chartext-disable = `true|false`
    Deshabilita las extensiones de gráficos. Valor predeterminado: `false`.
    Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

  - dotnetext-disable

    --dotnetext-disable = `true|false`
    Deshabilita las extensiones .NET. Valor predeterminado: `false`.
    Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

  - jvm-location

    --jvm-location = *ARCHIVO*
    ARCHIVO especifica la ubicación del equipo virtual Java (*DLL* en Windows, objeto *compartido (shared object)* en Linux. El JVM se necesita si usa las funciones de extensión Java en su código XSLT/XQuery. El valor predeterminado es `false`.

  - javaext-barcode-location

    --javaext-barcode-location = *ARCHIVO*
    Especifica la ruta de acceso de la carpeta que contiene el archivo de extensión de código de barras `AltovaBarcodeExtension.jar`. La ruta de acceso debe darse en uno de estos formatos:

    - Un URI de archivo (ejemplo: `--javaext-barcode-location="file:///C:/Archivos de programa/Altova/RaptorXMLXBRLServer2018/etc/jar/"`)
    - Una ruta de acceso Windows con caracteres de escape para las barras diagonales inversas (ejemplo: `--javaext-barcode-location="C:\\Archivos de programa\\Altova\\RaptorXMLXBRLServer2018\\etc\\jar\\"`)

  - javaext-disable

    --javaext-disable = `true|false`
    Deshabilita las extensiones Java. Valor predeterminado: `false`.

 © 2018 Altova GmbH

Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- Mensajes, errores, ayuda, tiempo de espera, versión
  - error-format

    `--error-format =` `text|shortxml|longxml`
    Especifica el formato de la salida de error. Los valores posibles son formatos de texto, XML y XML detallado (`longxml`). Valor predeterminado: `text`.

  - error-limit

    `--error-limit =` *`N`*
    Especifica el límite de errores con un valor comprendido entre `1` y `999`. El valor predeterminado es `100`. Cuando se alcanza el límite de error, se detiene la validación. Esta opción es muy práctica a la hora de limitar el uso del procesador durante la validación o transformación.

  - help

    `--help`
    Muestra el texto de ayuda para el comando. Por ejemplo `valany --h`. (Otra opción es usar el comando `help` con un argumento. Por ejemplo: `help valany`).

  - log-output

    `--log-output =` *`ARCHIVO`*
    Escribe el registro de salida en la URL de archivo indicada. Compruebe que la ILC tiene permiso de escritura en la ubicación de destino.

  - network-timeout

    `--network-timeout =` *`VALOR`*
    Indica el tiempo de espera en segundos para operaciones remotas de entrada y salida. Valor predeterminado: `40`.

  - verbose

    `--verbose =` `true|false`
    Si el valor es `true`, se genera información adicional durante la validación. Valor predeterminado es `false`.
    Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

  - verbose-output

    `--verbose-output =` *`ARCHIVO`*
    Escribe el resultado detallado en el *`ARCHIVO`* indicado.

  - version

    `--version`
    Muestra el número de versión de RaptorXML+XBRL Server. Si se utiliza con un

comando, escriba la opción `--version` antes del comando.

- warning-limit

  `--warning-limit =` *`VALUE`*

  Especifica el límite de advertencia en el rango `1-65535`. El procesamiento continua si se alcanza el límite pero no se registrarán más advertencias. El valor por defecto es `100`.

© 2018 Altova GmbH

3.5 Comandos XQuery

Estos son los comandos XQuery:

- xquery: para ejecutar documentos XQuery y, si se quiere, con un documento de entrada
- xqueryupdate: para ejecutar una actualización XQuery con un documento XQuery y el documento XML de entrada que se debe actualizar
- valxquery: para validar documentos XQuery
- valxqueryupdate: para validar documentos XQuery de actualización

Los argumentos y opciones de cada comando se describen en los siguientes apartados.

3.5.1 xquery

El comando **xquery** toma un archivo XQuery como único argumento y lo ejecuta con un archivo de entrada opcional para generar un archivo de salida. Los archivos de entrada y salida se especifican como opciones.

Windows	**RaptorXMLXBRL xquery [opciones] *Archivo-XQuery***
Linux	**raptorxmlxbrl xquery [opciones] *Archivo-XQuery***
Mac	**raptorxmlxbrl xquery [opciones] *Archivo-XQuery***

El argumento *Archivo-XQuery* es la ruta de acceso y el nombre del archivo XQuery que debe ejecutarse. Puede usar tanto XQuery 1.0 como 3.0, pero la versión predeterminada es 3.0.

Ejemplos

- **raptorxmlxbrl** xquery --output=c:\Salida.xml c:\TestConsulta.xq
- **raptorxmlxbrl** xquery --input=c:\Entrada.xml --output=c:\Salida.xml --param=company:"Altova" --p=date:"2006-01-01" c:\TestConsulta.xq
- **raptorxmlxbrl** xquery --input=c:\Entrada.xml --output=c:\Salida.xml --param=source:" doc('c:\test\books.xml')//book "
- **raptorxmlxbrl** xquery --output=c:\Salida.xml --omit-xml-declaration=false --output-encoding=ASCII c:\TestConsulta.xq

Uso de mayúsculas/minúsculas y de barras diagonales en la línea de comandos

RaptorXMLXBRL *en Windows*
raptorxmlxbrl *en Unix (Linux, Mac)*

* Las minúsculas (raptorxmlxbrl) funcionan en todas las plataformas (Windows, Linux y Mac), mientras que la mezcla de mayúsculas y minúsculas (RaptorXMLXBRL) sólo funciona en Windows y Mac.
* En Linux y Mac utilice barras diagonales. En Windows utilice barras diagonales inversas.

Opciones

Las opciones del comando aparecen más abajo divididas en grupos. Los valores se pueden dar sin comillas excepto en estos dos casos: (i) cuando la cadena de valor contiene espacios y (ii) cuando en la descripción de la opción se indique explícitamente que es necesario el uso de comillas.

- Procesamiento XQuery
  - indent-characters

    `--indent-characters =` *`VALOR`*

    Especifica la cadena de caracteres que debe usarse como sangría.

  - input

    `--input =` *`ARCHIVO`*

    La URL del archivo XML que se debe transformar.

  - omit-xml-declaration

    `--omit-xml-declaration =` `true|false`

    Opción de serialización que especifica si la declaración XML se omite en el resultado o no. Si el valor es `true`, el documento de salida no tendrá una declaración XML. Si el valor es `false`, se incluye una declaración XML en el documento de salida. Valor predeterminado: `false`.

    Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

  - output, xsltoutput

    `output =` *`ARCHIVO`*, **`xsltoutput =`** *`ARCHIVO`*

    La URL del archivo de salida principal. Por ejemplo, en caso de tener varios archivos HTML de salida, el archivo de salida principal será la ubicación del archivo HTML del punto de entrada. Los demás archivos de salida (como archivos de imagen generados) se indican con `xslt-additional-output-files`. Si no se especifica la opción `--output` ni la opción `--xsloutput`, se genera un resultado estándar.

  - output-encoding

    `--output-encoding =` *`VALOR`*

    El valor del atributo `encoding` del documento de salida. Son valores válidos todos los nombres del registro de juego de caracteres IANA. Valor predeterminado: `UTF-8`.

  - output-indent

    `--output-indent =` `true|false`

    Si el valor es `true`, la sangría del documento de salida seguirá su estructura jerárquica. Si el valor es `false`, el documento de salida no tendrá sangría jerárquica. Valor predeterminado: `false`.

    Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

  - output-method

 © 2018 Altova GmbH

--output-method = `xml|html|xhtml|text`
Especifica el formato de salida. Valor predeterminado: `xml`.

- param [p]

--p | --param = *CLAVE:VALOR*

XQuery

Especifica el valor de un parámetro externo. En el documento XQuery los parámetros externos se declaran con la declaración `declare variable` seguida de un nombre de variable y después la palabra clave `external` seguida del punto y coma final. Por ejemplo: `declare variable $foo as xs:string external;`

Al usar la palabra clave `external`, `$foo` se convierte en parámetro externo y su valor se pasa en tiempo de ejecución desde una fuente externa. El parámetro externo recibe un valor con el comando de la ILC. Por ejemplo: `--param=foo:'MiNombre'`

En la descripción anterior, *CLAVE* es el nombre de parámetro externo y *VALOR* es su valor, dado como expresión XPath. Los nombres de parámetro utilizados en la ILC deben declararse en el documento XQuery. Si se pasan valores a varios parámetros externos en la ILC, cada parámetro debe llevar una opción `--param` distinta. Si la expresión XPath contiene espacios, entonces debe estar entre comillas dobles.

XSLT

Especifica un parámetro global de la hoja de estilos. *CLAVE* es el nombre del parámetro y *VALOR* es una expresión XPath que da un valor al parámetro. Los nombres de parámetro utilizados en la ILC deben declararse en la hoja de estilos. Si usa más de un parámetro, debe usar el modificador `--param` antes de cada parámetro. Si la expresión XPath incluye espacios, entonces debe ir entre comillas dobles, tanto si el espacio está en la expresión propiamente dicha o en un literal de cadena de la expresión. Por ejemplo:

```
raptorxmlxbrl xslt --input=c:\Test.xml --output=c:\Salida.xml --
param=date://node[1]/@att1 --p=title:cadenasinespacios --
param=title:"'cadena con espacios'" --p=amount:456 c:\Test.xslt
```

- Instancias y esquemas XML
  - load-xml-with-psvi

--load-xml-with-psvi = `true|false`
Habilita la validación de archivos XML de entrada y genera información posterior a la validación. Valor predeterminado: `true`.

  - schema-imports

--schema-imports = `load-by-schemalocation | load-preferring-schemalocation | load-by-namespace | load-combining-both | license-namespace-only`
Esta opción indica el comportamiento de los elementos `xs:import`. Cada uno de estos

elementos tiene un atributo opcional `namespace` y un atributo opcional `schemaLocation`: `<import namespace="unEspacioNombres" schemaLocation="unaURL">`. La opción indica si se debe cargar un documento de esquema o solo autorizar a un espacio de nombres. Si la opción indica que se debe cargar un documento de esquema, entonces indica también qué información debe utilizarse para encontrar el documento de esquema. Valor predeterminado: `load-preferring-schemalocation`.

- `load-by-schemalocation`: el valor del atributo `schemaLocation` se utiliza para buscar el esquema, teniendo en cuenta las asignaciones de catálogo. Si está presente el atributo `namespace`, se importa el espacio de nombres (con licencia).
- `load-preferring-schemalocation`: si está presente, se utiliza el atributo `schemaLocation` teniendo en cuenta las asignaciones de catálogo. Si no está presente el atributo `schemaLocation`, entonces se usa el valor del atributo `namespace` a través de las asignaciones de catálogo. Este es el **valor predeterminado**.
- `load-by-namespace`: el valor del atributo `namespace` se utiliza para buscar el esquema por medio de una asignación de catálogo.
- `load-combining-both`: si el atributo `namespace` o `schemaLocation` tiene una asignación de catálogo, entonces se usa la asignación. Si ambos atributos tienen asignaciones de catálogo, entonces es el valor de la opción `--schema-mapping` (opción XBRL y opción XML/XSD) decide qué asignación se utiliza. Si no hay ninguna asignación de catálogo, entonces se usa el atributo `schemaLocation`.
- `license-namespace-only`: se importa el espacio de nombres. No se importa el documento de esquema.

schemalocation-hints

`--schemalocation-hints =` `load-by-schemalocation | load-by-namespace | load-combining-both | ignore`

Determina el comportamiento predeterminado de los atributos `xsi:schemaLocation` y `xsi:noNamespaceSchemaLocation`. Indica si se debe cargar un documento de esquema y, si así es, indica qué información debe utilizarse para encontrarlo. Valor predeterminado: `load-by-schemalocation`.

- **Valor predeterminado**: `load-by-schemalocation`. Este valor toma la URL de la ubicación del esquema de los atributos `xsi:schemaLocation` y `xsi:noNamespaceSchemaLocation` de los documentos de instancia XML o XBRL.

- El valor `load-by-namespace` toma la parte de espacio de nombres del atributo `xsi:schemaLocation` y una cadena vacía en el caso del atributo `xsi:noNamespaceSchemaLocation` y encuentra el esquema por medio de una asignación de catálogo.

- Si usa el valor `load-combining-both` y el espacio de nombres o la URL tienen una asignación de catálogo, se usa dicha asignación. Si ambos tienen asignaciones de catálogo, el valor de la opción `schema-mapping` (opción XBRL y opción XML/XSD) decide qué asignación se utiliza. Si ni el espacio de nombres ni la URL tiene una asignación de catálogo, se usa la URL.

- El valor `ignore` ignora los atributos `xsi:schemaLocation` y `xsi:noNamespaceSchemaLocation`.

 © 2018 Altova GmbH

- schema-mapping

  --schema-mapping = prefer-schemalocation | prefer-namespace
  Si se usa la ubicación y el espacio de nombres para buscar el documento de esquema, esta opción indica cuál de ellos debe ser la opción preferida durante la búsqueda en el catálogo.
  Si la opción --schemalocation-hints o la opción --schema-imports tiene el valor load-combining-both y si las partes de espacio de nombres y URL pertinentes tienen asignaciones de catálogo, entonces el valor de la opción especifica cuál de las dos asignaciones se utiliza (la asignación del espacio de nombres o de la URL: el valor prefer-schemalocation se refiere a la asignación de la URL). Valor predeterminado: prefer-schemalocation.

- xinclude

  --xinclude = true|false
  Habilita la compatibilidad con inclusiones XML (XInclude). Si el valor es false, los elementos XInclude include se ignoran. Valor predeterminado: false.
  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en true.

- xml-mode

  --xml-mode = wf|id|valid
  Especifica el modo de procesamiento XML que debe utilizarse para el documento de instancia XML: wf=comprobación de formato; id=comprobación de formato con ID/IDREF; valid=validación. Valor predeterminado: wf. Recuerde que el valor valid exige que cada documento de instancia que se cargue durante el procesamiento haga referencia a una DTD. Si no existe ninguna DTD, se generará un error.

- xml-mode-for-schemas

  --xml-mode-for-schemas = wf|id|valid
  Especifica el modo de procesamiento XML que debe utilizarse para el documento de instancia XML: wf=comprobación de formato; id=comprobación de formato con ID/IDREF; valid=validación. Valor predeterminado: wf. Recuerde que el valor valid exige que cada documento de esquema que se cargue durante el procesamiento haga referencia a una DTD. Si no existe ninguna DTD, se generará un error.

- xsd-version

  --xsd-version = 1.0|1.1|detect

  Especifica qué versión de la especificación Schema Definition Language (XSD) del W3C se debe usar. Valor predeterminado: 1.0.

  Esta opción también puede ser útil si quiere ver en qué aspectos no es compatible un esquema 1.0 con la especificación 1.1. El valor detect es una característica de Altova. Permite detectar la versión del esquema XML (1.0 o 1.1) leyendo el valor del atributo vc:minVersion del elemento <xs:schema> del documento. Si el valor del atributo @vc:minVersion es 1.1, se entiende que la versión del esquema es 1.1. Si el atributo tiene otro valor que no sea 1.1 (o si no está presente el atributo @vc:minVersion), se entiende que la versión del esquema es 1.0.

- Catálogos y recursos globales
  - catalog

    --catalog = *ARCHIVO*
    Especifica la ruta de acceso absoluta a un archivo de catálogo que no está en el archivo de catálogo raíz instalado. El valor predeterminado es la ruta de acceso absoluta del archivo de catálogo raíz instalado (`<carpeta-instalación>\Altova\RaptorXMLXBRLServer2018\etc\RootCatalog.xml`). Consulte el apartado Catálogos XML para obtener más información.

  - user-catalog

    --user-catalog = *ARCHIVO*
    Especifica la ruta de acceso absoluta a un catálogo XML que debe utilizarse junto con el catálogo raíz. Consulte el apartado Catálogos XML para obtener más información.

  - enable-globalresources

    --enable-globalresources = `true|false`
    Habilita la función de recursos globales. Valor predeterminado: `false`.
    Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

  - globalresourceconfig [gc]

    --gc | --globalresourceconfig = *VALOR*
    Especifica la configuración activa del recurso global (y habilita los recursos globales).

  - globalresourcefile [gr]

    --gr | --globalresourcefile = *ARCHIVO*
    Especifica el archivo de recursos globales (y habilita los recursos globales).

- Extensiones

  Estas opciones definen cómo se gestionan las funciones de extensión especiales disponibles en la edición Enterprise Edition de varios productos de Altova (como XMLSpy Enterprise Edition). Su uso se describe detalladamente en el manual del usuario de cada producto.

  - chartext-disable

    --chartext-disable = `true|false`
    Deshabilita las extensiones de gráficos. Valor predeterminado: `false`.
    Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

  - dotnetext-disable

    --dotnetext-disable = `true|false`
    Deshabilita las extensiones .NET. Valor predeterminado: `false`.
    Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

  - jvm-location

 © 2018 Altova GmbH

--jvm-location = *ARCHIVO*
ARCHIVO especifica la ubicación del equipo virtual Java (*DLL* en Windows, objeto *compartido (shared object)* en Linux. El JVM se necesita si usa las funciones de extensión Java en su código XSLT/XQuery. El valor predeterminado es `false`.

- javaext-barcode-location

  --javaext-barcode-location = *ARCHIVO*
  Especifica la ruta de acceso de la carpeta que contiene el archivo de extensión de código de barras `AltovaBarcodeExtension.jar`. La ruta de acceso debe darse en uno de estos formatos:

  - Un URI de archivo (ejemplo: `--javaext-barcode-location="file:///C:/Archivos de programa/Altova/RaptorXMLXBRLServer2018/etc/jar/"`)
  - Una ruta de acceso Windows con caracteres de escape para las barras diagonales inversas (ejemplo: `--javaext-barcode-location="C:\\Archivos de programa\\Altova\\RaptorXMLXBRLServer2018\\etc\\jar\\"`)

- javaext-disable

  --javaext-disable = `true|false`
  Deshabilita las extensiones Java. Valor predeterminado: `false`.
  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- Mensajes, errores, ayuda, tiempo de espera, versión
  - error-format

    --error-format = `text|shortxml|longxml`
    Especifica el formato de la salida de error. Los valores posibles son formatos de texto, XML y XML detallado (`longxml`). Valor predeterminado: `text`.

  - error-limit

    --error-limit = *N*
    Especifica el límite de errores con un valor comprendido entre `1` y `999`. El valor predeterminado es `100`. Cuando se alcanza el límite de error, se detiene la validación. Esta opción es muy práctica a la hora de limitar el uso del procesador durante la validación o transformación.

  - help

    --help
    Muestra el texto de ayuda para el comando. Por ejemplo `valany --h`. (Otra opción es usar el comando `help` con un argumento. Por ejemplo: `help valany`).

  - log-output

    --log-output = *ARCHIVO*
    Escribe el registro de salida en la URL de archivo indicada. Compruebe que la ILC tiene permiso de escritura en la ubicación de destino.

- network-timeout

  --network-timeout = *VALOR*
  Indica el tiempo de espera en segundos para operaciones remotas de entrada y salida. Valor predeterminado: `40`.

- verbose

  --verbose = `true|false`
  Si el valor es `true`, se genera información adicional durante la validación. Valor predeterminado es `false`.
  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- verbose-output

  --verbose-output = *ARCHIVO*
  Escribe el resultado detallado en el *ARCHIVO* indicado.

- version

  --version
  Muestra el número de versión de RaptorXML+XBRL Server. Si se utiliza con un comando, escriba la opción `--version` antes del comando.

- warning-limit

  --warning-limit = *VALUE*
  Especifica el límite de advertencia en el rango `1-65535`. El procesamiento continua si se alcanza el límite pero no se registrarán más advertencias. El valor por defecto es `100`.

3.5.2 xqueryupdate

El comando **xqueryupdate** toma un archivo XQuery como único argumento y lo ejecuta con un archivo de entrada opcional con el objetivo de producir un archivo de salida actualizado. Los archivos de entrada y salida se especifican en las opciones del comando.

Windows	**RaptorXMLXBRL xqueryupdate [opciones] *Archivo-XQuery***
Linux	**raptorxmlxbrl xqueryupdate [opciones] *Archivo-XQuery***
Mac	**raptorxmlxbrl xqueryupdate [opciones] *Archivo-XQuery***

El argumento ***Archivo-XQuery*** es la ruta de acceso y el nombre del archivo XQuery que se debe ejecutar. También puede especificar si se utiliza XQuery Update 1.0 o 3.0 (la opción predeterminada XQuery Update 3.0).

 © 2018 Altova GmbH

Ejemplos

- **raptorxmlxbrl** `xqueryupdate --output=c:\Salida.xml c:\TestQuery.xq`
- **raptorxmlxbrl** `xqueryupdate --input=c:\Entrada.xml --output=c:\Salida.xml --param=company:"Altova" --p=date:"2006-01-01" c:\TestQuery.xq`
- **raptorxmlxbrl** `xqueryupdate --input=c:\Entrada.xml --output=c:\Salida.xml --param=source:" doc( 'c:\test\books.xml' )//book "`
- **raptorxmlxbrl** `xqueryupdate --output=c:\Salida.xml --omit-xml-declaration=false --output-encoding=ASCII c:\TestQuery.xq`

Uso de mayúsculas/minúsculas y de barras diagonales en la línea de comandos

RaptorXMLXBRL *en Windows*
raptorxmlxbrl *en Unix (Linux, Mac)*

* Las minúsculas (`raptorxmlxbrl`) funcionan en todas las plataformas (Windows, Linux y Mac), mientras que la mezcla de mayúsculas y minúsculas (`RaptorXMLXBRL`) sólo funciona en Windows y Mac.
* En Linux y Mac utilice barras diagonales. En Windows utilice barras diagonales inversas.

Opciones

Las opciones del comando aparecen más abajo divididas en grupos. Los valores se pueden dar sin comillas excepto en estos dos casos: (i) cuando la cadena de valor contiene espacios y (ii) cuando en la descripción de la opción se indique explícitamente que es necesario el uso de comillas.

Procesamiento XQuery Update

Procesamiento XQuery

indent-characters

--indent-characters = *VALOR*
Especifica la cadena de caracteres que debe usarse como sangría.

input

--input = *ARCHIVO*
La URL del archivo XML que se debe transformar.

omit-xml-declaration

--omit-xml-declaration = `true|false`
Opción de serialización que especifica si la declaración XML se omite en el resultado o no. Si el valor es `true`, el documento de salida no tendrá una declaración XML. Si el valor es `false`, se incluye una declaración XML en el documento de salida. Valor predeterminado: `false`.
Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

output, xsltoutput

output = *ARCHIVO*, xsltoutput = *ARCHIVO*
La URL del archivo de salida principal. Por ejemplo, en caso de tener varios archivos HTML de salida, el archivo de salida principal será la ubicación del archivo HTML del punto de entrada. Los demás archivos de salida (como archivos de imagen generados) se indican con `xslt-additional-output-files`. Si no se especifica la opción `--output` ni la opción `--xsloutput`, se genera un resultado estándar.

- output-encoding

--output-encoding = *VALOR*
El valor del atributo `encoding` del documento de salida. Son valores válidos todos los nombres del registro de juego de caracteres IANA. Valor predeterminado: `UTF-8`.

- output-indent

--output-indent = `true|false`
Si el valor es `true`, la sangría del documento de salida seguirá su estructura jerárquica. Si el valor es `false`, el documento de salida no tendrá sangría jerárquica. Valor predeterminado: `false`.
Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- output-method

--output-method = `xml|html|xhtml|text`
Especifica el formato de salida. Valor predeterminado: `xml`.

- param [p]

--p | --param = *CLAVE:VALOR*

XQuery

Especifica el valor de un parámetro externo. En el documento XQuery los parámetros externos se declaran con la declaración `declare variable` seguida de un nombre de variable y después la palabra clave `external` seguida del punto y coma final. Por ejemplo: `declare variable $foo as xs:string external;`

Al usar la palabra clave `external`, `$foo` se convierte en parámetro externo y su valor se pasa en tiempo de ejecución desde una fuente externa. El parámetro externo recibe un valor con el comando de la ILC. Por ejemplo: `--param=foo:'MiNombre'`

En la descripción anterior, *CLAVE* es el nombre de parámetro externo y *VALOR* es su valor, dado como expresión XPath. Los nombres de parámetro utilizados en la ILC deben declararse en el documento XQuery. Si se pasan valores a varios parámetros externos en la ILC, cada parámetro debe llevar una opción `--param` distinta. Si la expresión XPath contiene espacios, entonces debe estar entre comillas dobles.

XSLT

Especifica un parámetro global de la hoja de estilos. *CLAVE* es el nombre del parámetro y *VALOR* es una expresión XPath que da un valor al parámetro. Los nombres de parámetro utilizados en la ILC deben declararse en la hoja de estilos. Si usa más de un parámetro, debe usar el modificador `--param` antes de cada parámetro. Si la expresión XPath incluye espacios, entonces debe ir entre comillas dobles, tanto si el espacio está en la expresión propiamente dicha o en un literal de cadena de la expresión. Por ejemplo:

```
raptorxmlxbrl xslt --input=c:\Test.xml --output=c:\Salida.xml
--param=date://node[1]/@att1 --p=title:cadenasinespacios --
param=title:"'cadena con espacios'" --p=amount:456 c:\Test.xslt
```

- xquery-update-version

  `--xquery-update-version =` `1|1.0|3|3.0|`
  Indica si el procesador XQuery debería usar XQuery Update Facility 1.0 o XQuery Update Facility 3.0. Valor predeterminado: `3`.

- keep-formatting

  `--keep-formatting =` `true|false`
  Conserva en la medida de lo posible el formato del documento de destino. Valor predeterminado: `true`.

- updated-xml

  `--updated-xml =` `discard|writeback|asmainresult`
  Indica qué se debe hacer con el archivo XML actualizado. Las actualizaciones se pueden:

  - descartar (los cambios no se escriben en el archivo). Valor `discard`
  - escribir en el archivo XML de entrada que se especificó con la opción `--input`. Valor `writeback`
  - guardar en el resultado estándar o en la ubicación que se especificó con la opción `--output` (si se usó esta opción)

  Valor predeterminado: `discard`.

- Instancias y esquemas XML
  - load-xml-with-psvi

    `--load-xml-with-psvi =` `true|false`
    Habilita la validación de archivos XML de entrada y genera información posterior a la validación. Valor predeterminado: `true`.

  - xinclude

    `--xinclude =` `true|false`
    Habilita la compatibilidad con inclusiones XML (XInclude). Si el valor es `false`, los elementos XInclude `include` se ignoran. Valor predeterminado: `false`.

Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

schema-imports

`--schema-imports =` `load-by-schemalocation | load-preferring-schemalocation | load-by-namespace | load-combining-both | license-namespace-only`

Esta opción indica el comportamiento de los elementos `xs:import`. Cada uno de estos elementos tiene un atributo opcional `namespace` y un atributo opcional `schemaLocation`: `<import namespace="unEspacioNombres" schemaLocation="unaURL">`. La opción indica si se debe cargar un documento de esquema o solo autorizar a un espacio de nombres. Si la opción indica que se debe cargar un documento de esquema, entonces indica también qué información debe utilizarse para encontrar el documento de esquema. Valor predeterminado: `load-preferring-schemalocation`.

- `load-by-schemalocation`: el valor del atributo `schemaLocation` se utiliza para buscar el esquema, teniendo en cuenta las asignaciones de catálogo. Si está presente el atributo `namespace`, se importa el espacio de nombres (con licencia).
- `load-preferring-schemalocation`: si está presente, se utiliza el atributo `schemaLocation` teniendo en cuenta las asignaciones de catálogo. Si no está presente el atributo `schemaLocation`, entonces se usa el valor del atributo `namespace` a través de las asignaciones de catálogo. Este es el **valor predeterminado**.
- `load-by-namespace`: el valor del atributo `namespace` se utiliza para buscar el esquema por medio de una asignación de catálogo.
- `load-combining-both`: si el atributo `namespace` o `schemaLocation` tiene una asignación de catálogo, entonces se usa la asignación. Si ambos atributos tienen asignaciones de catálogo, entonces es el valor de la opción `--schema-mapping` (opción XBRL y opción XML/XSD) decide qué asignación se utiliza. Si no hay ninguna asignación de catálogo, entonces se usa el atributo `schemaLocation`.
- `license-namespace-only`: se importa el espacio de nombres. No se importa el documento de esquema.

schemalocation-hints

`--schemalocation-hints =` `load-by-schemalocation | load-by-namespace | load-combining-both | ignore`

Determina el comportamiento predeterminado de los atributos `xsi:schemaLocation` y `xsi:noNamespaceSchemaLocation`. Indica si se debe cargar un documento de esquema y, si así es, indica qué información debe utilizarse para encontrarlo. Valor predeterminado: `load-by-schemalocation`.

- **Valor predeterminado**: `load-by-schemalocation`. Este valor toma la URL de la ubicación del esquema de los atributos `xsi:schemaLocation` y `xsi:noNamespaceSchemaLocation` de los documentos de instancia XML o XBRL.

- El valor `load-by-namespace` toma la parte de espacio de nombres del atributo `xsi:schemaLocation` y una cadena vacía en el caso del atributo `xsi:noNamespaceSchemaLocation` y encuentra el esquema por medio de una asignación de catálogo.

 © 2018 Altova GmbH

- Si usa el valor `load-combining-both` y el espacio de nombres o la URL tienen una asignación de catálogo, se usa dicha asignación. Si ambos tienen asignaciones de catálogo, el valor de la opción `schema-mapping` (opción XBRL y opción XML/XSD) decide qué asignación se utiliza. Si ni el espacio de nombres ni la URL tiene una asignación de catálogo, se usa la URL.

- El valor `ignore` ignora los atributos `xsi:schemaLocation` y `xsi:noNamespaceSchemaLocation`.

schema-mapping

`--schema-mapping =` `prefer-schemalocation | prefer-namespace`
Si se usa la ubicación y el espacio de nombres para buscar el documento de esquema, esta opción indica cuál de ellos debe ser la opción preferida durante la búsqueda en el catálogo.
Si la opción `--schemalocation-hints` o la opción `--schema-imports` tiene el valor `load-combining-both` y si las partes de espacio de nombres y URL pertinentes tienen asignaciones de catálogo, entonces el valor de la opción especifica cuál de las dos asignaciones se utiliza (la asignación del espacio de nombres o de la URL: el valor `prefer-schemalocation` se refiere a la asignación de la URL). Valor predeterminado: `prefer-schemalocation`.

xinclude

`--xinclude =` `true|false`
Habilita la compatibilidad con inclusiones XML (XInclude). Si el valor es `false`, los elementos XInclude `include` se ignoran. Valor predeterminado: `false`.
Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

xml-mode

`--xml-mode =` `wf|id|valid`
Especifica el modo de procesamiento XML que debe utilizarse para el documento de instancia XML: `wf`=comprobación de formato; `id`=comprobación de formato con ID/IDREF; `valid`=validación. Valor predeterminado: `wf`. Recuerde que el valor `valid` exige que cada documento de instancia que se cargue durante el procesamiento haga referencia a una DTD. Si no existe ninguna DTD, se generará un error.

xml-mode-for-schemas

`--xml-mode-for-schemas =` `wf|id|valid`
Especifica el modo de procesamiento XML que debe utilizarse para el documento de instancia XML: `wf`=comprobación de formato; `id`=comprobación de formato con ID/IDREF; `valid`=validación. Valor predeterminado: `wf`. Recuerde que el valor `valid` exige que cada documento de esquema que se cargue durante el procesamiento haga referencia a una DTD. Si no existe ninguna DTD, se generará un error.

xml-validation-error-as-warning

`--xml-validation-error-as-warning =` `true|false`
Si es `true`, tratar los errores de validación como advertencias. Si los errores se tratan como advertencias, el procesamiento adicional como la transformación XSLT, proseguirá independientemente de los errores. Por defecto es `false`.

xsd

--xsd = *ARCHIVO*
Especifica qué esquemas XML deben utilizarse para la validación de documentos XML. Si quiere especificar más de un esquema, añada la opción varias veces.

- xsd-version

--xsd-version = `1.0|1.1|detect`

Especifica qué versión de la especificación Schema Definition Language (XSD) del W3C se debe usar. Valor predeterminado: `1.0`.

Esta opción también puede ser útil si quiere ver en qué aspectos no es compatible un esquema 1.0 con la especificación 1.1. El valor `detect` es una característica de Altova. Permite detectar la versión del esquema XML (`1.0` o `1.1`) leyendo el valor del atributo `vc:minVersion` del elemento `<xs:schema>` del documento. Si el valor del atributo `@vc:minVersion` es `1.1`, se entiende que la versión del esquema es `1.1`. Si el atributo tiene otro valor que no sea `1.1` (o si no está presente el atributo `@vc:minVersion`), se entiende que la versión del esquema es `1.0`.

- Catálogos y recursos globales
  - catalog

--catalog = *ARCHIVO*
Especifica la ruta de acceso absoluta a un archivo de catálogo que no está en el archivo de catálogo raíz instalado. El valor predeterminado es la ruta de acceso absoluta del archivo de catálogo raíz instalado (`<carpeta-instalación>\Altova\RaptorXMLXBRLServer2018\etc\RootCatalog.xml`). Consulte el apartado Catálogos XML para obtener más información.

  - user-catalog

--user-catalog = *ARCHIVO*
Especifica la ruta de acceso absoluta a un catálogo XML que debe utilizarse junto con el catálogo raíz. Consulte el apartado Catálogos XML para obtener más información.

  - enable-globalresources

--enable-globalresources = `true|false`
Habilita la función de recursos globales. Valor predeterminado: `false`.
Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

  - globalresourceconfig [gc]

--gc | --globalresourceconfig = *VALOR*
Especifica la configuración activa del recurso global (y habilita los recursos globales).

  - globalresourcefile [gr]

--gr | --globalresourcefile = *ARCHIVO*
Especifica el archivo de recursos globales (y habilita los recursos globales).

 © 2018 Altova GmbH

- Extensiones

  Estas opciones definen cómo se gestionan las funciones de extensión especiales disponibles en la edición Enterprise Edition de varios productos de Altova (como XMLSpy Enterprise Edition). Su uso se describe detalladamente en el manual del usuario de cada producto.

  - chartext-disable

    `--chartext-disable =` `true|false`
    Deshabilita las extensiones de gráficos. Valor predeterminado: `false`.
    Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

  - dotnetext-disable

    `--dotnetext-disable =` `true|false`
    Deshabilita las extensiones .NET. Valor predeterminado: `false`.
    Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

  - jvm-location

    `--jvm-location =` *`ARCHIVO`*
    `ARCHIVO` especifica la ubicación del equipo virtual Java (*DLL* en Windows, objeto *compartido (shared object)* en Linux. El JVM se necesita si usa las funciones de extensión Java en su código XSLT/XQuery. El valor predeterminado es `false`.

  - javaext-barcode-location

    `--javaext-barcode-location =` *`ARCHIVO`*
    Especifica la ruta de acceso de la carpeta que contiene el archivo de extensión de código de barras `AltovaBarcodeExtension.jar`. La ruta de acceso debe darse en uno de estos formatos:

    - Un URI de archivo (ejemplo: `--javaext-barcode-location="file:///C:/Archivos de programa/Altova/RaptorXMLXBRLServer2018/etc/jar/"`)
    - Una ruta de acceso Windows con caracteres de escape para las barras diagonales inversas (ejemplo: `--javaext-barcode-location="C:\\Archivos de programa\\Altova\\RaptorXMLXBRLServer2018\\etc\\jar\\"`)

  - javaext-disable

    `--javaext-disable =` `true|false`
    Deshabilita las extensiones Java. Valor predeterminado: `false`.
    Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- Mensajes, errores, ayuda, tiempo de espera, versión

  - error-format

    `--error-format =` `text|shortxml|longxml`
    Especifica el formato de la salida de error. Los valores posibles son formatos de texto,

XML y XML detallado (`longxml`). Valor predeterminado: `text`.

- error-limit

  `--error-limit =` *`N`*
  Especifica el límite de errores con un valor comprendido entre `1` y `999`. El valor predeterminado es `100`. Cuando se alcanza el límite de error, se detiene la validación. Esta opción es muy práctica a la hora de limitar el uso del procesador durante la validación o transformación.

- help

  `--help`
  Muestra el texto de ayuda para el comando. Por ejemplo `valany --h`. (Otra opción es usar el comando `help` con un argumento. Por ejemplo: `help valany`).

- log-output

  `--log-output =` *`ARCHIVO`*
  Escribe el registro de salida en la URL de archivo indicada. Compruebe que la ILC tiene permiso de escritura en la ubicación de destino.

- network-timeout

  `--network-timeout =` *`VALOR`*
  Indica el tiempo de espera en segundos para operaciones remotas de entrada y salida. Valor predeterminado: `40`.

- verbose

  `--verbose =` `true|false`
  Si el valor es `true`, se genera información adicional durante la validación. Valor predeterminado es `false`.
  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- verbose-output

  `--verbose-output =` *`ARCHIVO`*
  Escribe el resultado detallado en el *`ARCHIVO`* indicado.

- version

  `--version`
  Muestra el número de versión de RaptorXML+XBRL Server. Si se utiliza con un comando, escriba la opción `--version` antes del comando.

- warning-limit

  `--warning-limit =` *`VALUE`*
  Especifica el límite de advertencia en el rango `1-65535`. El procesamiento continua si se alcanza el límite pero no se registrarán más advertencias. El valor por defecto es `100`.

 © 2018 Altova GmbH

3.5.3 valxquery

El comando **valxquery** toma un archivo XQuery como único argumento y lo valida.

Windows	**RaptorXMLXBRL valxquery [opciones] *Archivo-XQuery***
Linux	**raptorxmlxbrl valxquery [opciones] *Archivo-XQuery***
Mac	**raptorxmlxbrl valxquery [opciones] *Archivo-XQuery***

El argumento *Archivo-XQuery* es la ruta de acceso y el nombre del archivo XQuery que debe validarse.

Ejemplos

- **raptorxmlxbrl** valxquery c:\Test.xquery
- **raptorxmlxbrl** valxquery --xquery-version=1 c:\Test.xquery

Uso de mayúsculas/minúsculas y de barras diagonales en la línea de comandos

RaptorXMLXBRL *en Windows*
raptorxmlxbrl *en Unix (Linux, Mac)*

* Las minúsculas (raptorxmlxbrl) funcionan en todas las plataformas (Windows, Linux y Mac), mientras que la mezcla de mayúsculas y minúsculas (RaptorXMLXBRL) sólo funciona en Windows y Mac.
* En Linux y Mac utilice barras diagonales. En Windows utilice barras diagonales inversas.

Opciones

Las opciones del comando aparecen más abajo divididas en grupos. Los valores se pueden dar sin comillas excepto en estos dos casos: (i) cuando la cadena de valor contiene espacios y (ii) cuando en la descripción de la opción se indique explícitamente que es necesario el uso de comillas.

Procesamiento XQuery

omit-xml-declaration

--omit-xml-declaration = true|false
Opción de serialización que especifica si la declaración XML se omite en el resultado o no. Si el valor es true, el documento de salida no tendrá una declaración XML. Si el valor es false, se incluye una declaración XML en el documento de salida. Valor predeterminado: false.
Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en true.

xquery-version

--xquery-version = 1|1.0|3|3.0|3.1
Indica si el procesador XQuery debe usar XQuery 1.0 o 3.0. Valor predeterminado: 3.1

- Instancias y esquemas XML
  - load-xml-with-psvi

    --load-xml-with-psvi = true|false
    Habilita la validación de archivos XML de entrada y genera información posterior a la validación. Valor predeterminado: true.

  - schema-imports

    --schema-imports = load-by-schemalocation | load-preferring-schemalocation | load-by-namespace | load-combining-both | license-namespace-only
    Esta opción indica el comportamiento de los elementos xs:import. Cada uno de estos elementos tiene un atributo opcional namespace y un atributo opcional schemaLocation: <import namespace="unEspacioNombres" schemaLocation="unaURL">. La opción indica si se debe cargar un documento de esquema o solo autorizar a un espacio de nombres. Si la opción indica que se debe cargar un documento de esquema, entonces indica también qué información debe utilizarse para encontrar el documento de esquema. Valor predeterminado: load-preferring-schemalocation.

    - load-by-schemalocation: el valor del atributo schemaLocation se utiliza para buscar el esquema, teniendo en cuenta las asignaciones de catálogo. Si está presente el atributo namespace, se importa el espacio de nombres (con licencia).
    - load-preferring-schemalocation: si está presente, se utiliza el atributo schemaLocation teniendo en cuenta las asignaciones de catálogo. Si no está presente el atributo schemaLocation, entonces se usa el valor del atributo namespace a través de las asignaciones de catálogo. Este es el **valor predeterminado**.
    - load-by-namespace: el valor del atributo namespace se utiliza para buscar el esquema por medio de una asignación de catálogo.
    - load-combining-both: si el atributo namespace o schemaLocation tiene una asignación de catálogo, entonces se usa la asignación. Si ambos atributos tienen asignaciones de catálogo, entonces es el valor de la opción --schema-mapping (opción XBRL y opción XML/XSD) decide qué asignación se utiliza. Si no hay ninguna asignación de catálogo, entonces se usa el atributo schemaLocation.
    - license-namespace-only: se importa el espacio de nombres. No se importa el documento de esquema.

  - schemalocation-hints

    --schemalocation-hints = load-by-schemalocation | load-by-namespace | load-combining-both | ignore
    Determina el comportamiento predeterminado de los atributos xsi:schemaLocation y xsi:noNamespaceSchemaLocation. Indica si se debe cargar un documento de esquema y, si así es, indica qué información debe utilizarse para encontrarlo. Valor predeterminado: load-by-schemalocation.

 © 2018 Altova GmbH

- **Valor predeterminado**: `load-by-schemalocation`. Este valor toma la URL de la ubicación del esquema de los atributos `xsi:schemaLocation` y `xsi:noNamespaceSchemaLocation` de los documentos de instancia XML o XBRL.

- El valor `load-by-namespace` toma la parte de espacio de nombres del atributo `xsi:schemaLocation` y una cadena vacía en el caso del atributo `xsi:noNamespaceSchemaLocation` y encuentra el esquema por medio de una asignación de catálogo.

- Si usa el valor `load-combining-both` y el espacio de nombres o la URL tienen una asignación de catálogo, se usa dicha asignación. Si ambos tienen asignaciones de catálogo, el valor de la opción `schema-mapping` (opción XBRL y opción XML/XSD) decide qué asignación se utiliza. Si ni el espacio de nombres ni la URL tiene una asignación de catálogo, se usa la URL.

- El valor `ignore` ignora los atributos `xsi:schemaLocation` y `xsi:noNamespaceSchemaLocation`.

schema-mapping

`--schema-mapping =` `prefer-schemalocation | prefer-namespace`
Si se usa la ubicación y el espacio de nombres para buscar el documento de esquema, esta opción indica cuál de ellos debe ser la opción preferida durante la búsqueda en el catálogo.
Si la opción `--schemalocation-hints` o la opción `--schema-imports` tiene el valor `load-combining-both` y si las partes de espacio de nombres y URL pertinentes tienen asignaciones de catálogo, entonces el valor de la opción especifica cuál de las dos asignaciones se utiliza (la asignación del espacio de nombres o de la URL: el valor `prefer-schemalocation` se refiere a la asignación de la URL). Valor predeterminado: `prefer-schemalocation`.

xinclude

`--xinclude =` `true|false`
Habilita la compatibilidad con inclusiones XML (XInclude). Si el valor es `false`, los elementos XInclude `include` se ignoran. Valor predeterminado: `false`.
Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

xml-mode

`--xml-mode =` `wf|id|valid`
Especifica el modo de procesamiento XML que debe utilizarse para el documento de instancia XML: `wf`=comprobación de formato; `id`=comprobación de formato con ID/IDREF; `valid`=validación. Valor predeterminado: `wf`. Recuerde que el valor `valid` exige que cada documento de instancia que se cargue durante el procesamiento haga referencia a una DTD. Si no existe ninguna DTD, se generará un error.

xml-mode-for-schemas

`--xml-mode-for-schemas =` `wf|id|valid`
Especifica el modo de procesamiento XML que debe utilizarse para el documento de instancia XML: `wf`=comprobación de formato; `id`=comprobación de formato con ID/IDREF; `valid`=validación. Valor predeterminado: `wf`. Recuerde que el valor `valid` exige

que cada documento de esquema que se cargue durante el procesamiento haga referencia a una DTD. Si no existe ninguna DTD, se generará un error.

- xsd-version

  --xsd-version = 1.0|1.1|detect

  Especifica qué versión de la especificación Schema Definition Language (XSD) del W3C se debe usar. Valor predeterminado: 1.0.

  Esta opción también puede ser útil si quiere ver en qué aspectos no es compatible un esquema 1.0 con la especificación 1.1. El valor detect es una característica de Altova. Permite detectar la versión del esquema XML (1.0 o 1.1) leyendo el valor del atributo vc:minVersion del elemento <xs:schema> del documento. Si el valor del atributo @vc:minVersion es 1.1, se entiende que la versión del esquema es 1.1. Si el atributo tiene otro valor que no sea 1.1 (o si no está presente el atributo @vc:minVersion), se entiende que la versión del esquema es 1.0.

- Catálogos y recursos globales
  - catalog

    --catalog = *ARCHIVO*
    Especifica la ruta de acceso absoluta a un archivo de catálogo que no está en el archivo de catálogo raíz instalado. El valor predeterminado es la ruta de acceso absoluta del archivo de catálogo raíz instalado (<carpeta-instalación>\Altova\RaptorXMLXBRLServer2018\etc\RootCatalog.xml). Consulte el apartado Catálogos XML para obtener más información.

  - user-catalog

    --user-catalog = *ARCHIVO*
    Especifica la ruta de acceso absoluta a un catálogo XML que debe utilizarse junto con el catálogo raíz. Consulte el apartado Catálogos XML para obtener más información.

  - enable-globalresources

    --enable-globalresources = true|false
    Habilita la función de recursos globales. Valor predeterminado: false.
    Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en true.

  - globalresourceconfig [gc]

    --gc | --globalresourceconfig = *VALOR*
    Especifica la configuración activa del recurso global (y habilita los recursos globales).

  - globalresourcefile [gr]

    --gr | --globalresourcefile = *ARCHIVO*
    Especifica el archivo de recursos globales (y habilita los recursos globales).

- Extensiones

Estas opciones definen cómo se gestionan las funciones de extensión especiales disponibles en la edición Enterprise Edition de varios productos de Altova (como XMLSpy Enterprise Edition). Su uso se describe detalladamente en el manual del usuario de cada producto.

- chartext-disable

  `--chartext-disable =` `true|false`
  Deshabilita las extensiones de gráficos. Valor predeterminado: `false`.
  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- dotnetext-disable

  `--dotnetext-disable =` `true|false`
  Deshabilita las extensiones .NET. Valor predeterminado: `false`.
  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- jvm-location

  `--jvm-location =` *`ARCHIVO`*
  `ARCHIVO` especifica la ubicación del equipo virtual Java (*DLL* en Windows, objeto *compartido (shared object)* en Linux. El JVM se necesita si usa las funciones de extensión Java en su código XSLT/XQuery. El valor predeterminado es `false`.

- javaext-barcode-location

  `--javaext-barcode-location =` *`ARCHIVO`*
  Especifica la ruta de acceso de la carpeta que contiene el archivo de extensión de código de barras `AltovaBarcodeExtension.jar`. La ruta de acceso debe darse en uno de estos formatos:

  - Un URI de archivo (ejemplo: `--javaext-barcode-location="file:///C:/Archivos de programa/Altova/RaptorXMLXBRLServer2018/etc/jar/"`)
  - Una ruta de acceso Windows con caracteres de escape para las barras diagonales inversas (ejemplo: `--javaext-barcode-location="C:\\Archivos de programa\\Altova\\RaptorXMLXBRLServer2018\\etc\\jar\\"`)

- javaext-disable

  `--javaext-disable =` `true|false`
  Deshabilita las extensiones Java. Valor predeterminado: `false`.
  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

Mensajes, errores, ayuda, tiempo de espera, versión

- error-format

  `--error-format =` `text|shortxml|longxml`
  Especifica el formato de la salida de error. Los valores posibles son formatos de texto, XML y XML detallado (`longxml`). Valor predeterminado: `text`.

- error-limit

--error-limit = *N*
Especifica el límite de errores con un valor comprendido entre `1` y `999`. El valor predeterminado es `100`. Cuando se alcanza el límite de error, se detiene la validación. Esta opción es muy práctica a la hora de limitar el uso del procesador durante la validación o transformación.

- help

--help
Muestra el texto de ayuda para el comando. Por ejemplo `valany --h`. (Otra opción es usar el comando `help` con un argumento. Por ejemplo: `help valany`).

- log-output

--log-output = *ARCHIVO*
Escribe el registro de salida en la URL de archivo indicada. Compruebe que la ILC tiene permiso de escritura en la ubicación de destino.

- network-timeout

--network-timeout = *VALOR*
Indica el tiempo de espera en segundos para operaciones remotas de entrada y salida. Valor predeterminado: `40`.

- verbose

--verbose = `true|false`
Si el valor es `true`, se genera información adicional durante la validación. Valor predeterminado es `false`.
Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- verbose-output

--verbose-output = *ARCHIVO*
Escribe el resultado detallado en el *ARCHIVO* indicado.

- version

--version
Muestra el número de versión de RaptorXML+XBRL Server. Si se utiliza con un comando, escriba la opción `--version` antes del comando.

- warning-limit

--warning-limit = *VALUE*
Especifica el límite de advertencia en el rango `1-65535`. El procesamiento continua si se alcanza el límite pero no se registrarán más advertencias. El valor por defecto es `100`.

 © 2018 Altova GmbH

3.5.4 valxqueryupdate

El comando **valxqueryupdate** toma un archivo XQuery como único argumento y lo valida.

Windows	**RaptorXMLXBRL valxqueryupdate [opciones] *Archivo-XQuery***
Linux	**raptorxmlxbrl valxqueryupdate [opciones] *Archivo-XQuery***
Mac	**raptorxmlxbrl valxqueryupdate [opciones] *Archivo-XQuery***

El argumento ***Archivo-XQuery*** es la ruta de acceso y el nombre del archivo XQuery que se debe validar.

Ejemplos

- **raptorxmlxbrl** `valxqueryupdae c:\Test.xqu`
- **raptorxmlxbrl** `valxqueryupdate --xquery-version=1 c:\Test.xqu`

Uso de mayúsculas/minúsculas y de barras diagonales en la línea de comandos

RaptorXMLXBRL *en Windows*
raptorxmlxbrl *en Unix (Linux, Mac)*

* Las minúsculas (`raptorxmlxbrl`) funcionan en todas las plataformas (Windows, Linux y Mac), mientras que la mezcla de mayúsculas y minúsculas (`RaptorXMLXBRL`) sólo funciona en Windows y Mac.
* En Linux y Mac utilice barras diagonales. En Windows utilice barras diagonales inversas.

Opciones

Las opciones del comando aparecen más abajo divididas en grupos. Los valores se pueden dar sin comillas excepto en estos dos casos: (i) cuando la cadena de valor contiene espacios y (ii) cuando en la descripción de la opción se indique explícitamente que es necesario el uso de comillas.

Procesamiento XQuery

omit-xml-declaration

--omit-xml-declaration = `true|false`
Opción de serialización que especifica si la declaración XML se omite en el resultado o no. Si el valor es `true`, el documento de salida no tendrá una declaración XML. Si el valor es `false`, se incluye una declaración XML en el documento de salida. Valor predeterminado: `false`.
Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

xquery-update-version

--xquery-update-version = 1|1.0|3|3.0|
Indica si el procesador XQuery debería usar XQuery Update Facility 1.0 o XQuery Update Facility 3.0. Valor predeterminado: 3.

- Instancias y esquemas XML
  - load-xml-with-psvi

    --load-xml-with-psvi = true|false
    Habilita la validación de archivos XML de entrada y genera información posterior a la validación. Valor predeterminado: true.

  - schema-imports

    --schema-imports = load-by-schemalocation | load-preferring-schemalocation | load-by-namespace | load-combining-both | license-namespace-only
    Esta opción indica el comportamiento de los elementos xs:import. Cada uno de estos elementos tiene un atributo opcional namespace y un atributo opcional schemaLocation: <import namespace="unEspacioNombres" schemaLocation="unaURL">. La opción indica si se debe cargar un documento de esquema o solo autorizar a un espacio de nombres. Si la opción indica que se debe cargar un documento de esquema, entonces indica también qué información debe utilizarse para encontrar el documento de esquema. Valor predeterminado: load-preferring-schemalocation.

    - load-by-schemalocation: el valor del atributo schemaLocation se utiliza para buscar el esquema, teniendo en cuenta las asignaciones de catálogo. Si está presente el atributo namespace, se importa el espacio de nombres (con licencia).
    - load-preferring-schemalocation: si está presente, se utiliza el atributo schemaLocation teniendo en cuenta las asignaciones de catálogo. Si no está presente el atributo schemaLocation, entonces se usa el valor del atributo namespace a través de las asignaciones de catálogo. Este es el **valor predeterminado**.
    - load-by-namespace: el valor del atributo namespace se utiliza para buscar el esquema por medio de una asignación de catálogo.
    - load-combining-both: si el atributo namespace o schemaLocation tiene una asignación de catálogo, entonces se usa la asignación. Si ambos atributos tienen asignaciones de catálogo, entonces es el valor de la opción --schema-mapping (opción XBRL y opción XML/XSD) decide qué asignación se utiliza. Si no hay ninguna asignación de catálogo, entonces se usa el atributo schemaLocation.
    - license-namespace-only: se importa el espacio de nombres. No se importa el documento de esquema.

  - schemalocation-hints

    --schemalocation-hints = load-by-schemalocation | load-by-namespace | load-combining-both | ignore
    Determina el comportamiento predeterminado de los atributos xsi:schemaLocation y xsi:noNamespaceSchemaLocation. Indica si se debe cargar un documento de esquema y, si así es, indica qué información debe utilizarse para encontrarlo. Valor predeterminado: load-by-schemalocation.

 © 2018 Altova GmbH

- **Valor predeterminado**: `load-by-schemalocation`. Este valor toma la URL de la ubicación del esquema de los atributos `xsi:schemaLocation` y `xsi:noNamespaceSchemaLocation` de los documentos de instancia XML o XBRL.

- El valor `load-by-namespace` toma la parte de espacio de nombres del atributo `xsi:schemaLocation` y una cadena vacía en el caso del atributo `xsi:noNamespaceSchemaLocation` y encuentra el esquema por medio de una asignación de catálogo.

- Si usa el valor `load-combining-both` y el espacio de nombres o la URL tienen una asignación de catálogo, se usa dicha asignación. Si ambos tienen asignaciones de catálogo, el valor de la opción `schema-mapping` (opción XBRL y opción XML/XSD) decide qué asignación se utiliza. Si ni el espacio de nombres ni la URL tiene una asignación de catálogo, se usa la URL.

- El valor `ignore` ignora los atributos `xsi:schemaLocation` y `xsi:noNamespaceSchemaLocation`.

schema-mapping

`--schema-mapping =` `prefer-schemalocation | prefer-namespace`
Si se usa la ubicación y el espacio de nombres para buscar el documento de esquema, esta opción indica cuál de ellos debe ser la opción preferida durante la búsqueda en el catálogo.
Si la opción `--schemalocation-hints` o la opción `--schema-imports` tiene el valor `load-combining-both` y si las partes de espacio de nombres y URL pertinentes tienen asignaciones de catálogo, entonces el valor de la opción especifica cuál de las dos asignaciones se utiliza (la asignación del espacio de nombres o de la URL: el valor `prefer-schemalocation` se refiere a la asignación de la URL). Valor predeterminado: `prefer-schemalocation`.

xinclude

`--xinclude =` `true|false`
Habilita la compatibilidad con inclusiones XML (XInclude). Si el valor es `false`, los elementos XInclude `include` se ignoran. Valor predeterminado: `false`.
Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

xml-mode

`--xml-mode =` `wf|id|valid`
Especifica el modo de procesamiento XML que debe utilizarse para el documento de instancia XML: `wf`=comprobación de formato; `id`=comprobación de formato con ID/IDREF; `valid`=validación. Valor predeterminado: `wf`. Recuerde que el valor `valid` exige que cada documento de instancia que se cargue durante el procesamiento haga referencia a una DTD. Si no existe ninguna DTD, se generará un error.

xml-mode-for-schemas

`--xml-mode-for-schemas =` `wf|id|valid`
Especifica el modo de procesamiento XML que debe utilizarse para el documento de instancia XML: `wf`=comprobación de formato; `id`=comprobación de formato con ID/

IDREF; `valid`=validación. Valor predeterminado: `wf`. Recuerde que el valor `valid` exige que cada documento de esquema que se cargue durante el procesamiento haga referencia a una DTD. Si no existe ninguna DTD, se generará un error.

- xsd-version

  `--xsd-version =` `1.0|1.1|detect`

  Especifica qué versión de la especificación Schema Definition Language (XSD) del W3C se debe usar. Valor predeterminado: `1.0`.

  Esta opción también puede ser útil si quiere ver en qué aspectos no es compatible un esquema 1.0 con la especificación 1.1. El valor `detect` es una característica de Altova. Permite detectar la versión del esquema XML (`1.0` o `1.1`) leyendo el valor del atributo `vc:minVersion` del elemento `<xs:schema>` del documento. Si el valor del atributo `@vc:minVersion` es `1.1`, se entiende que la versión del esquema es `1.1`. Si el atributo tiene otro valor que no sea `1.1` (o si no está presente el atributo `@vc:minVersion`), se entiende que la versión del esquema es `1.0`.

- Catálogos y recursos globales
  - catalog

    `--catalog =` *`ARCHIVO`*
    Especifica la ruta de acceso absoluta a un archivo de catálogo que no está en el archivo de catálogo raíz instalado. El valor predeterminado es la ruta de acceso absoluta del archivo de catálogo raíz instalado (`<carpeta-instalación>\Altova\RaptorXMLXBRLServer2018\etc\RootCatalog.xml`). Consulte el apartado Catálogos XML para obtener más información.

  - user-catalog

    `--user-catalog =` *`ARCHIVO`*
    Especifica la ruta de acceso absoluta a un catálogo XML que debe utilizarse junto con el catálogo raíz. Consulte el apartado Catálogos XML para obtener más información.

  - enable-globalresources

    `--enable-globalresources =` `true|false`
    Habilita la función de recursos globales. Valor predeterminado: `false`.
    Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

  - globalresourceconfig [gc]

    `--gc | --globalresourceconfig =` *`VALOR`*
    Especifica la configuración activa del recurso global (y habilita los recursos globales).

  - globalresourcefile [gr]

    `--gr | --globalresourcefile =` *`ARCHIVO`*
    Especifica el archivo de recursos globales (y habilita los recursos globales).

 © 2018 Altova GmbH

- Extensiones

  Estas opciones definen cómo se gestionan las funciones de extensión especiales disponibles en la edición Enterprise Edition de varios productos de Altova (como XMLSpy Enterprise Edition). Su uso se describe detalladamente en el manual del usuario de cada producto.

  - chartext-disable

    `--chartext-disable =` `true|false`
    Deshabilita las extensiones de gráficos. Valor predeterminado: `false`.
    Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

  - dotnetext-disable

    `--dotnetext-disable =` `true|false`
    Deshabilita las extensiones .NET. Valor predeterminado: `false`.
    Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

  - jvm-location

    `--jvm-location =` *`ARCHIVO`*
    `ARCHIVO` especifica la ubicación del equipo virtual Java (*DLL* en Windows, objeto *compartido (shared object)* en Linux. El JVM se necesita si usa las funciones de extensión Java en su código XSLT/XQuery. El valor predeterminado es `false`.

  - javaext-barcode-location

    `--javaext-barcode-location =` *`ARCHIVO`*
    Especifica la ruta de acceso de la carpeta que contiene el archivo de extensión de código de barras `AltovaBarcodeExtension.jar`. La ruta de acceso debe darse en uno de estos formatos:

    - Un URI de archivo (ejemplo: `--javaext-barcode-location="file:///C:/Archivos de programa/Altova/RaptorXMLXBRLServer2018/etc/jar/"`)
    - Una ruta de acceso Windows con caracteres de escape para las barras diagonales inversas (ejemplo: `--javaext-barcode-location="C:\\Archivos de programa\\Altova\\RaptorXMLXBRLServer2018\\etc\\jar\\"`)

  - javaext-disable

    `--javaext-disable =` `true|false`
    Deshabilita las extensiones Java. Valor predeterminado: `false`.
    Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- Mensajes, errores, ayuda, tiempo de espera, versión

  - error-format

    `--error-format =` `text|shortxml|longxml`
    Especifica el formato de la salida de error. Los valores posibles son formatos de texto, XML y XML detallado (`longxml`). Valor predeterminado: `text`.

- error-limit

  --error-limit = *N*
  Especifica el límite de errores con un valor comprendido entre `1` y `999`. El valor predeterminado es `100`. Cuando se alcanza el límite de error, se detiene la validación. Esta opción es muy práctica a la hora de limitar el uso del procesador durante la validación o transformación.

- help

  --help
  Muestra el texto de ayuda para el comando. Por ejemplo `valany --h`. (Otra opción es usar el comando `help` con un argumento. Por ejemplo: `help valany`).

- log-output

  --log-output = *ARCHIVO*
  Escribe el registro de salida en la URL de archivo indicada. Compruebe que la ILC tiene permiso de escritura en la ubicación de destino.

- network-timeout

  --network-timeout = *VALOR*
  Indica el tiempo de espera en segundos para operaciones remotas de entrada y salida. Valor predeterminado: `40`.

- verbose

  --verbose = `true|false`
  Si el valor es `true`, se genera información adicional durante la validación. Valor predeterminado es `false`.
  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- verbose-output

  --verbose-output = *ARCHIVO*
  Escribe el resultado detallado en el *ARCHIVO* indicado.

- version

  --version
  Muestra el número de versión de RaptorXML+XBRL Server. Si se utiliza con un comando, escriba la opción `--version` antes del comando.

- warning-limit

  --warning-limit = *VALUE*
  Especifica el límite de advertencia en el rango `1-65535`. El procesamiento continua si se alcanza el límite pero no se registrarán más advertencias. El valor por defecto es `100`.

 © 2018 Altova GmbH

3.6 Comandos XML Signature

Los comandos XML Signature sirven para firmar documentos XML y verificar documentos firmados. Encontrará una descripción detallada de los comandos XML Signature en los apartados de esta sección:

xmlsignature-sign	Crea un documento de salida de firma XML a partir de un documento de entrada.
xmlsignature-verify	Verifica un documento de firma XML.
xmlsignature-update	Actualiza la firma de un documento XML (modificado).
xmlsignature-remove	Quita la firma de un documento XML.

3.6.1 xmlsignature-sign

El comando **xmlsignature-sign | xsign** toma como entrada un documento XML y crea un documento con firma XML de salida usando las opciones de firma especificadas.

Windows
```
RaptorXMLXBRL xmlsignature-sign [opciones] --output=Archivo --
signature-type=Valor
--signature-canonicalization-method=Valor ArchivoEntrada
```

Linux
```
raptorxmlxbrl xmlsignature-sign [opciones] --output=Archivo --
signature-type=Valor
--signature-canonicalization-method=Valor ArchivoEntrada
```

Mac
```
raptorxmlxbrl xmlsignature-sign [opciones] --output=Archivo --
signature-type=Valor
--signature-canonicalization-method=Valor ArchivoEntrada
```

El argumento *ArchivoEntrada* es el documento XML que se debe firmar. La opción **--output** especifica la ubicación del documento que contiene la firma XML.

Ejemplos

- **raptorxmlxbrl** `xsign --output=c:\ArchivoFirmado.xml --signature-type=enveloped --signature-canonicalization-method=xml-c14n11 c:\ArchivoSinFirmar.xml`

▼ Uso de mayúsculas/minúsculas y de barras diagonales en la línea de comandos

RaptorXMLXBRL *en Windows*
raptorxmlxbrl *en Unix (Linux, Mac)*

* Las minúsculas (`raptorxmlxbrl`) funcionan en todas las plataformas (Windows, Linux y

Mac), mientras que la mezcla de mayúsculas y minúsculas (`RaptorXMLXBRL`) sólo funciona en Windows y Mac.
* En Linux y Mac utilice barras diagonales. En Windows utilice barras diagonales inversas.

Opciones

Las opciones del comando aparecen más abajo divididas en grupos. Los valores se pueden dar sin comillas excepto en estos dos casos: (i) cuando la cadena de valor contiene espacios y (ii) cuando en la descripción de la opción se indique explícitamente que es necesario el uso de comillas.

- Opciones comunes
  - output

    `output =` *`ARCHIVO`*
    La URL del documento de salida que se crea con la nueva firma XML.

  - verbose

    `--verbose =` `true|false`
    Si el valor es `true`, se genera información adicional durante la validación. Valor predeterminado es `false`.
    Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- Opciones de firma XML
  - absolute-reference-uri

    `--absolute-reference-uri =` `true|false`
    Especifica si el URI del documento firmado debe leerse como absoluto (`true`) o relativo (`false`). El valor predeterminado es `false`.
    Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

  - certname, certificate-name

    `--certname, --certificate-name =` *`VALOR`*
    El nombre del certificado utilizado para firmar.

    Windows
    Se trata del nombre `Subject` de un certificado del almacén de certificados (**`--certificate-store`**) seleccionado.

    Ejemplo para enumerar los certificados (bajo PowerShell)

```
% ls cert://UsuarioActual/My
PSParentPath: Microsoft.PowerShell.Security
\Certificate::UsuarioActual\My
Thumbprint Subject
```

 © 2018 Altova GmbH

```
---------- -------
C9DF64BB0AAF5FA73474D78B7CCFFC37C95BFC6C CN=certificate1
... CN=...
```

Ejemplo: **--certificate-name==certificado1**

Linux/macOS

--certname especifica el nombre de archivo de un certificado X.509v3 con cifrado PEM con la clave privada. Estos archivos suelen tener la extensión **.pem**.

Ejemplo: **--certificate-name==/ruta/de/certificado1.pem**

certstore, certificate-store

--certstore, --certificate-store = *VALOR*

La ubicación donde está almacenado el certificado especificado con **--certificate-name**.

Windows

Nombre de un almacén de certificados situado dentro de **cert://UsuarioActual**. Los almacenes de certificados disponibles se pueden enumerar (bajo PowerShell) usando **% ls cert://UsuarioActual/**. Los certificados se enumerarán de la siguiente manera:

```
Nombre : TrustedPublisher
Nombre : ClientAuthIssuer
Nombre : Root
Nombre : UserDS
Nombre : CA
Nombre : ACRS
Nombre : REQUEST
Nombre : AuthRoot
Nombre : MSIEHistoryJournal
Nombre : TrustedPeople
Nombre : MyCertStore
Nombre : Local NonRemovable Certificates
Nombre : SmartCardRoot
Nombre : Trust
Nombre : Disallowed
```

Ejemplo: **--certificate-store==MiAlmacénCertificados**

Linux/MacOS

La opción **--certstore** no es compatible por ahora.

- digest, digest-method

  --digest, --digest-method = sha1|sha256|sha384|sha512|base64
  El algoritmo que se utiliza para calcular el valor implícito con el archivo XML de entrada.

- hmackey, hmac-secret-key

  --hmackey, --hmac-secret-key = *VALOR*
  La clave secreta compartida HMAC. Debe tener una longitud mínima de seis caracteres.

  Ejemplo: **--hmackey=contraseñaSecreta**

- hmaclen, hmac-output-length

  --hmaclen, --hmac-output-length = *LONGITUD*

  Trunca el resultado del algoritmo HMAC hasta el número de bits indicado por LONGITUD. Si se especifica, este valor debe ser:
  - múltiplo de 8,
  - mayor que 80 y
  - mayor que la mitad de la longitud de salida del algoritmo hash subyacente.

- keyinfo, append-keyinfo

  --keyinfo, --append-keyinfo = true|false
  Especifica si se debe incluir el elemento **KeyInfo** en la firma o no. El valor predeterminado es **false**.

- sigc14nmeth, signature-canonicalization-method

  --sigc14nmeth, --signature-canonicalization-method = *VALOR*
  Especifica el algoritmo de canonización que se debe aplicar al elemento **SignedInfo**. El valor debe ser uno de estos tres valores:

  - REC-xml-c14n-20010315
  - xml-c14n11
  - xml-exc-c14n#

- sigmeth, signature-method

  --sigmeth, --signature-method = *VALOR*
  Especifica el algoritmo que se debe usar para generar la firma.

  Cuando se usa un certificado
  Si se especificó un certificado, entonces --signature-method es opcional y el valor para este parámetro se deriva del certificado. Debe coincidir con el algoritmo utilizado por el certificado.

 © 2018 Altova GmbH

Ejemplo: **`--signature-method=rsa-sha256`**

Cuando se usa --hmac-secret-key
Si se usa la opción `--hmac-secret-key`, entonces esta opción es obligatoria. El valor debe ser uno de los algoritmos HMAC compatibles:

- `hmac-sha256`
- `hmac-sha386`
- `hmac-sha512`
- `hmac-sha1` (no recomendable según la especificación)

Ejemplo: **`--signature-method=hmac-sha256`**

- sigtype, signature-type

`--sigtype, --signature-type =` `detached | enveloping | enveloped`
Especifica el tipo de firma que se debe generar (separada, envolvente o envuelta).

- transforms

`--transforms =` *`VALOR`*
Especifica las transformaciones XML Signature que se deben aplicar al documento de salida. Esta opción se puede especificar más de una vez. Si se especifica varias veces, el orden en que se especifica es importante. La primera transformación especificada recibe el documento de entrada. La última se utiliza inmediatamente antes de calcular el valor implícito.

Valores compatibles:

- `REC-xml-c14n-20010315` para Canonical XML 1.0 (omitir comentarios)
- `xml-c14n11` para Canonical XML 1.1 (omitir comentarios)
- `xml-exc-c14n#` para Exclusive XML Canonicalization 1.0 (omitir comentarios)
- `REC-xml-c14n-20010315#WithComments` para Canonical XML 1.0 (con comentarios)
- `xml-c14n11#WithComments` para Canonical XML 1.1 (con comentarios)
- `xml-exc-c14n#WithComments` para Exclusive XML Canonicalization 1.0 (con comentarios)
- `base64`
- extensión de Altova `strip-whitespaces`

Ejemplo: **`--transforms=xml-c14n11`**

- write-default-attributes

`--write-default-attributes =` `true|false`
Especifica si se deben incluir los valores de atributo predeterminados de la DTD en el documento firmado.

- Opciones de ayuda y versión
  - help

--help
Muestra el texto de ayuda para el comando. Por ejemplo `valany --h`. (Otra opción es usar el comando `help` con un argumento. Por ejemplo: `help valany`).

- version

--version
Muestra el número de versión de RaptorXML+XBRL Server. Si se utiliza con un comando, escriba la opción `--version` antes del comando.

3.6.2 xmlsignature-verify

El comando **xmlsignature-verify | xverify** verifica la firma XML del archivo de entrada. Si la firma se verifica correctamente, aparece un mensaje **resultado="OK"**. En caso contrario aparece el mensaje **resultado="Con errores"**.

Windows	**RaptorXMLXBRL xmlsignature-verify [opciones] *ArchivoEntrada***
Linux	**raptorxmlxbrl xmlsignature-verify [opciones] *ArchivoEntrada***
Mac	**raptorxmlxbrl xmlsignature-verify [opciones] *ArchivoEntrada***

El argumento *ArchivoEntrada* es el documento XML firmado que se debe verificar.

Ejemplos

- **raptorxmlxbrl** `xverify c:\ArchivoFirmado.xml`

- Uso de mayúsculas/minúsculas y de barras diagonales en la línea de comandos

RaptorXMLXBRL *en Windows*
raptorxmlxbrl *en Unix (Linux, Mac)*

* Las minúsculas (`raptorxmlxbrl`) funcionan en todas las plataformas (Windows, Linux y Mac), mientras que la mezcla de mayúsculas y minúsculas (`RaptorXMLXBRL`) sólo funciona en Windows y Mac.
* En Linux y Mac utilice barras diagonales. En Windows utilice barras diagonales inversas.

Opciones

Las opciones del comando aparecen más abajo divididas en grupos. Los valores se pueden dar sin comillas excepto en estos dos casos: (i) cuando la cadena de valor contiene espacios y (ii) cuando en la descripción de la opción se indique explícitamente que es necesario el uso de comillas.

- Opciones comunes
  - verbose

    --verbose = `true|false`
    Si el valor es `true`, se genera información adicional durante la validación. Valor predeterminado es `false`.
    Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- Opciones de firma XML
  - certname, certificate-name

    --certname, --certificate-name = *VALOR*
    El nombre del certificado utilizado para firmar.

    Windows
    Se trata del nombre `Subject` de un certificado del almacén de certificados (**--certificate-store**) seleccionado.

    Ejemplo para enumerar los certificados (bajo PowerShell)

```
% ls cert://UsuarioActual/My
PSParentPath: Microsoft.PowerShell.Security\Certificate::UsuarioActual
\My
Thumbprint Subject
---------- -------
C9DF64BB0AAF5FA73474D78B7CCFFC37C95BFC6C CN=certificate1
... CN=...
```

    Ejemplo: **--certificate-name==certificado1**

    Linux/macOS
    --certname especifica el nombre de archivo de un certificado X.509v3 con cifrado PEM con la clave privada. Estos archivos suelen tener la extensión **.pem**.

    Ejemplo: **--certificate-name==/ruta/de/certificado1.pem**

  - certstore, certificate-store

    --certstore, --certificate-store = *VALOR*
    La ubicación donde está almacenado el certificado especificado con **--certificate-name**.

    Windows
    Nombre de un almacén de certificados situado dentro de **cert://UsuarioActual**. Los

almacenes de certificados disponibles se pueden enumerar (bajo PowerShell) usando `% ls cert://UsuarioActual/`. Los certificados se enumerarán de la siguiente manera:

```
Nombre : TrustedPublisher
Nombre : ClientAuthIssuer
Nombre : Root
Nombre : UserDS
Nombre : CA
Nombre : ACRS
Nombre : REQUEST
Nombre : AuthRoot
Nombre : MSIEHistoryJournal
Nombre : TrustedPeople
Nombre : MyCertStore
Nombre : Local NonRemovable Certificates
Nombre : SmartCardRoot
Nombre : Trust
Nombre : Disallowed
```

Ejemplo: **`--certificate-store==MiAlmacénCertificados`**

Linux/MacOS
La opción **`--certstore`** no es compatible por ahora.

- hmackey, hmac-secret-key

  `--hmackey, --hmac-secret-key =` *VALOR*
  La clave secreta compartida HMAC. Debe tener una longitud mínima de seis caracteres.

  Ejemplo: **`--hmackey=contraseñaSecreta`**

- Opciones de ayuda y versión
  - help

    `--help`
    Muestra el texto de ayuda para el comando. Por ejemplo `valany --h`. (Otra opción es usar el comando `help` con un argumento. Por ejemplo: `help valany`).

  - version

    `--version`
    Muestra el número de versión de RaptorXML+XBRL Server. Si se utiliza con un comando, escriba la opción `--version` antes del comando.

 © 2018 Altova GmbH

3.6.3 xmlsignature-update

El comando **xmlsignature-update | xupdate** actualiza la firma XML en el archivo de entrada firmado. Si el documento sufrió cambios, la firma XML será diferente. En caso contrario, la firma actualizada será la misma que la anterior.

Windows **RaptorXMLXBRL xmlsignature-update [opciones] --output=Archivo *ArchivoFirmado***

Linux **raptorxmlxbrl xmlsignature-update [opciones] --output=Archivo *ArchivoFirmado***

Mac **raptorxmlxbrl xmlsignature-update [opciones] --output=Archivo *ArchivoFirmado***

El argumento *ArchivoFirmado* es el documento XML firmado que se debe actualizar. Es obligatorio especificar (i) la opción `hmac-secret-key` o (ii) la opción `certificate-name`. Si se especifican las opciones `certificate-name` y `certificate-store` , entonces deben coincidir con las utilizadas previamente para firmar el documento XML. (Recuerde que la opción `certificate-store` no se puede usar en Linux ni macOS.)

Ejemplos

- **raptorxmlxbrl** `xupdate --output=c:\ActualizarArchivoFirmado.xml --certname=certificado1 --certstore=MiAlmacénCert c:\UnArchivoFirmado.xml`
- **raptorxmlxbrl** `xupdate --output=c:\ActualizarArchivoFirmado.xml --hmackey=ContraseñaSecreta c:\UnArchivoFirmado.xml`

Uso de mayúsculas/minúsculas y de barras diagonales en la línea de comandos

RaptorXMLXBRL *en Windows*
raptorxmlxbrl *en Unix (Linux, Mac)*

* Las minúsculas (`raptorxmlxbrl`) funcionan en todas las plataformas (Windows, Linux y Mac), mientras que la mezcla de mayúsculas y minúsculas (`RaptorXMLXBRL`) sólo funciona en Windows y Mac.
* En Linux y Mac utilice barras diagonales. En Windows utilice barras diagonales inversas.

Opciones

Las opciones del comando aparecen más abajo divididas en grupos. Los valores se pueden dar sin comillas excepto en estos dos casos: (i) cuando la cadena de valor contiene espacios y (ii) cuando en la descripción de la opción se indique explícitamente que es necesario el uso de comillas.

Opciones comunes

output

output = *ARCHIVO*
La URL del documento de salida que se crea con la nueva firma XML.

- verbose

--verbose = `true|false`
Si el valor es `true`, se genera información adicional durante la validación. Valor predeterminado es `false`.
Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- Opciones de firma XML
  - certname, certificate-name

--certname, --certificate-name = *VALOR*
El nombre del certificado utilizado para firmar.

Windows
Se trata del nombre `Subject` de un certificado del almacén de certificados (**--certificate-store**) seleccionado.

Ejemplo para enumerar los certificados (bajo PowerShell)

```
% ls cert://UsuarioActual/My
PSParentPath: Microsoft.PowerShell.Security\Certificate::UsuarioActual
\My
Thumbprint Subject
---------- -------
C9DF64BB0AAF5FA73474D78B7CCFFC37C95BFC6C CN=certificate1
... CN=...
```

Ejemplo: **--certificate-name==certificado1**

Linux/macOS
--certname especifica el nombre de archivo de un certificado X.509v3 con cifrado PEM con la clave privada. Estos archivos suelen tener la extensión **.pem**.

Ejemplo: **--certificate-name==/ruta/de/certificado1.pem**

- certstore, certificate-store

--certstore, --certificate-store = *VALOR*
La ubicación donde está almacenado el certificado especificado con **--certificate-name**.

 © 2018 Altova GmbH

Windows
Nombre de un almacén de certificados situado dentro de **`cert://UsuarioActual`**. Los almacenes de certificados disponibles se pueden enumerar (bajo PowerShell) usando **`% ls cert://UsuarioActual/`**. Los certificados se enumerarán de la siguiente manera:

```
Nombre : TrustedPublisher
Nombre : ClientAuthIssuer
Nombre : Root
Nombre : UserDS
Nombre : CA
Nombre : ACRS
Nombre : REQUEST
Nombre : AuthRoot
Nombre : MSIEHistoryJournal
Nombre : TrustedPeople
Nombre : MyCertStore
Nombre : Local NonRemovable Certificates
Nombre : SmartCardRoot
Nombre : Trust
Nombre : Disallowed
```

Ejemplo: **`--certificate-store==MiAlmacénCertificados`**

Linux/MacOS
La opción **`--certstore`** no es compatible por ahora.

- hmackey, hmac-secret-key

  `--hmackey, --hmac-secret-key =` *VALOR*
  La clave secreta compartida HMAC. Debe tener una longitud mínima de seis caracteres.

  Ejemplo: **`--hmackey=contraseñaSecreta`**

- Opciones de ayuda y versión
  - help

    `--help`
    Muestra el texto de ayuda para el comando. Por ejemplo `valany --h`. (Otra opción es usar el comando `help` con un argumento. Por ejemplo: `help valany`).

  - version

    `--version`
    Muestra el número de versión de RaptorXML+XBRL Server. Si se utiliza con un comando, escriba la opción `--version` antes del comando.

3.6.4 xmlsignature-remove

El comando **xmlsignature-remove | xremove** quita la firma XML del archivo de entrada firmado y guarda el documento de salida sin firmar en la ubicación de destino especificada.

Windows **RaptorXMLXBRL xmlsignature-remove [opciones] --output=Archivo *ArchivoFirmado***

Linux **raptorxmlxbrl xmlsignature-remove [opciones] --output=Archivo *ArchivoFirmado***

Mac **raptorxmlxbrl xmlsignature-remove [opciones] --output=Archivo *ArchivoFirmado***

El argumento *ArchivoFirmado* es el documento XML firmado del que desea quitar la firma XML. La opción `--output` especifica la ubicación del documento XML sin firmar que se genera.

Ejemplos

- **raptorxmlxbrl** `xremove --output=c:\ArchivoSinFirmar.xml c:\ArchivoFirmado.xml`

Uso de mayúsculas/minúsculas y de barras diagonales en la línea de comandos

RaptorXMLXBRL *en Windows*
raptorxmlxbrl *en Unix (Linux, Mac)*

* Las minúsculas (`raptorxmlxbrl`) funcionan en todas las plataformas (Windows, Linux y Mac), mientras que la mezcla de mayúsculas y minúsculas (`RaptorXMLXBRL`) sólo funciona en Windows y Mac.
* En Linux y Mac utilice barras diagonales. En Windows utilice barras diagonales inversas.

Opciones

Las opciones del comando aparecen más abajo divididas en grupos. Los valores se pueden dar sin comillas excepto en estos dos casos: (i) cuando la cadena de valor contiene espacios y (ii) cuando en la descripción de la opción se indique explícitamente que es necesario el uso de comillas.

Opciones comunes

output

output = *ARCHIVO*
La URL del archivo de salida que se crea tras eliminar la firma XML.

verbose

--verbose = `true|false`

Si el valor es `true`, se genera información adicional durante la validación. Valor predeterminado es `false`.
Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- Opciones de ayuda y versión
  - help

    `--help`
    Muestra el texto de ayuda para el comando. Por ejemplo `valany --h`. (Otra opción es usar el comando `help` con un argumento. Por ejemplo: `help valany`).

  - version

    `--version`
    Muestra el número de versión de RaptorXML+XBRL Server. Si se utiliza con un comando, escriba la opción `--version` antes del comando.

3.7 Comandos JSON/Avro

Los comandos JSON sirven para comprobar la validez de esquemas y documentos de instancia JSON y comprobar si su formato es correcto. Para más información sobre cada comando consulte los apartados de esta sección:

avroextractschema	Extrae el esquema Avro de un archivo binario Avro.
valavro	Valida los datos de archivos binarios Avro con el esquema Avro correspondiente de cada binario.
valavrojson	Valida archivos de datos JSON con un esquema Avro.
valavroschema	Valida un esquema Avro con la especificación Avro.
valjsonschema	Comprueba la validez de documentos de esquema JSON.
valjson	Comprueba la validez de documentos JSON.
wfjson	Comprueba si el formato de los documentos JSON es correcto.

3.7.1 avroextractschema

Un archivo binario Avro contiene un bloque de datos Avro precedido por un esquema Avro que define la estructura de dicho bloque de datos. El comando **avroextractschema** extrae el esquema Avro del binario Avro y lo serializa como JSON.

Windows
```
RaptorXMLXBRL avroextractschema [opciones] --
avrooutput=ArchivoEsquemaAvro ArchivoBinarioAvro
```

Linux
```
raptorxmlxbrl avroextractschema [opciones] --
avrooutput=ArchivoEsquemaAvro ArchivoBinarioAvro
```

Mac
```
raptorxmlxbrl avroextractschema [opciones] --
avrooutput=ArchivoEsquemaAvro ArchivoBinarioAvro
```

El argumento *ArchivoBinarioAvro* especifica de qué archivo binario Avro se debe extraer el esquema Avro. La opción `--avrooutput` especifica la ubicación del esquema Avro que se extrae.

Ejemplos

```
raptorxmlxbrl avroextractschema --avrooutput=c:\MiEsquemaAvro.avsc c:
   \MiBinarioAvro.avro
```

- Uso de mayúsculas/minúsculas y de barras diagonales en la línea de comandos

RaptorXMLXBRL *en Windows*
raptorxmlxbrl *en Unix (Linux, Mac)*

* Las minúsculas (raptorxmlxbrl) funcionan en todas las plataformas (Windows, Linux y Mac), mientras que la mezcla de mayúsculas y minúsculas (RaptorXMLXBRL) sólo funciona en Windows y Mac.
* En Linux y Mac utilice barras diagonales. En Windows utilice barras diagonales inversas.

Opciones

Las opciones del comando aparecen más abajo divididas en grupos. Los valores se pueden dar sin comillas excepto en estos dos casos: (i) cuando la cadena de valor contiene espacios y (ii) cuando en la descripción de la opción se indique explícitamente que es necesario el uso de comillas.

- Procesamiento
  - output, **avrooutput**

    --output = *ARCHIVO*, **--avrooutput =** *ARCHIVO*
    Establece la ubicación del archivo Avro de salida.

  - recurse

    --recurse = true|false
    Esta opción se utiliza para seleccionar ficheros dentro de subdirectorios (incluso en archivos ZIP). Si el valor es true, el argumento *ArchivoEntrada* del comando seleccionará el fichero seleccionado también en los subdirectorios. Por ejemplo: test.zip|zip\test.xml seleccionará los ficheros llamados test.xml en todos los subdirectorios de la carpeta ZIP. Si quiere puede usar los caracteres comodín * y ?. Por ejemplo: *.xml seleccionaría todos los ficheros de la carpeta ZIP que tengan la extensión .xml. Valor predeterminado: false
    Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en true.

- Catálogos y recursos globales
  - catalog

    --catalog = *ARCHIVO*
    Especifica la ruta de acceso absoluta a un archivo de catálogo que no está en el archivo de catálogo raíz instalado. El valor predeterminado es la ruta de acceso absoluta del archivo de catálogo raíz instalado (<carpeta-instalación>\Altova\RaptorXMLXBRLServer2018\etc\RootCatalog.xml). Consulte el apartado Catálogos XML para obtener más información.

  - user-catalog

    --user-catalog = *ARCHIVO*
    Especifica la ruta de acceso absoluta a un catálogo XML que debe utilizarse junto con el catálogo raíz. Consulte el apartado Catálogos XML para obtener más información.

- enable-globalresources

  --enable-globalresources = `true|false`
  Habilita la función de recursos globales. Valor predeterminado: `false`.
  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- globalresourceconfig [gc]

  --gc | --globalresourceconfig = *VALOR*
  Especifica la configuración activa del recurso global (y habilita los recursos globales).

- globalresourcefile [gr]

  --gr | --globalresourcefile = *ARCHIVO*
  Especifica el archivo de recursos globales (y habilita los recursos globales).

- Mensajes, errores, ayuda, tiempo de espera, versión
  - error-format

    --error-format = `text|shortxml|longxml`
    Especifica el formato de la salida de error. Los valores posibles son formatos de texto, XML y XML detallado (`longxml`). Valor predeterminado: `text`.

  - error-limit

    --error-limit = *N*
    Especifica el límite de errores con un valor comprendido entre `1` y `999`. El valor predeterminado es `100`. Cuando se alcanza el límite de error, se detiene la validación. Esta opción es muy práctica a la hora de limitar el uso del procesador durante la validación o transformación.

  - help

    --help
    Muestra el texto de ayuda para el comando. Por ejemplo `valany --h`. (Otra opción es usar el comando `help` con un argumento. Por ejemplo: `help valany`).

  - log-output

    --log-output = *ARCHIVO*
    Escribe el registro de salida en la URL de archivo indicada. Compruebe que la ILC tiene permiso de escritura en la ubicación de destino.

  - network-timeout

    --network-timeout = *VALOR*
    Indica el tiempo de espera en segundos para operaciones remotas de entrada y salida. Valor predeterminado: `40`.

  - verbose

    --verbose = `true|false`
    Si el valor es `true`, se genera información adicional durante la validación. Valor

predeterminado es `false`.
Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- verbose-output

  `--verbose-output =` *`ARCHIVO`*
  Escribe el resultado detallado en el *`ARCHIVO`* indicado.

- version

  `--version`
  Muestra el número de versión de RaptorXML+XBRL Server. Si se utiliza con un comando, escriba la opción `--version` antes del comando.

- warning-limit

  `--warning-limit =` *`VALUE`*
  Especifica el límite de advertencia en el rango `1-65535`. El procesamiento continua si se alcanza el límite pero no se registrarán más advertencias. El valor por defecto es `100`.

3.7.2 valavro (avro)

El comando **`valavro | avro`** valida los bloques de datos de archivos binarios con los esquemas Avro correspondientes de cada archivo binario.

Windows	**`RaptorXMLXBRL valavro	avro [opciones]`** ***`ArchivoBinarioAvro`***
Linux	**`raptorxmlxbrl valavro	avro [opciones]`** ***`ArchivoBinarioAvro`***
Mac	**`raptorxmlxbrl valavro	avro [opciones]`** ***`ArchivoBinarioAvro`***

El argumento *`ArchivoBinarioAvro`* especifica qué archivos binarios Avro se deben validar. Concretamente, el bloque de datos de cada archivo binario Avro se valida con el esquema Avro del archivo correspondiente.

Si desea validar varios binarios Avro, tiene dos opciones: (i) enumerar los archivos que se deben validar en la línea de comandos, separados por espacios o (ii) enumerar los archivos que se deben validar en un archivo de texto (`.txt`) donde aparece un nombre de archivo por línea y dar este archivo de texto como argumento *`ArchivoBinarioAvro`* junto con la opción `--listfile` con valor `true` (*ver lista de opciones más abajo*).

Ejemplos

- **`raptorxmlxbrl`** `valavro c:\MiBinarioAvro.avro`

- **raptorxmlxbrl** `valavro c:\MiBinarioAvro01.avro c:\MiBinarioAvro02.avro`
- **raptorxmlxbrl** `avro--listfile=true c:\MiListaArchivos.txt`

Uso de mayúsculas/minúsculas y de barras diagonales en la línea de comandos

RaptorXMLXBRL *en Windows*
raptorxmlxbrl *en Unix (Linux, Mac)*

* Las minúsculas (`raptorxmlxbrl`) funcionan en todas las plataformas (Windows, Linux y Mac), mientras que la mezcla de mayúsculas y minúsculas (`RaptorXMLXBRL`) sólo funciona en Windows y Mac.
* En Linux y Mac utilice barras diagonales. En Windows utilice barras diagonales inversas.

Opciones

Las opciones del comando aparecen más abajo divididas en grupos. Los valores se pueden dar sin comillas excepto en estos dos casos: (i) cuando la cadena de valor contiene espacios y (ii) cuando en la descripción de la opción se indique explícitamente que es necesario el uso de comillas.

Procesamiento

listfile

--listfile = `true|false`

Si el valor es `true`, el argumento *`ArchivoEntrada`* del comando se entiende como un archivo de texto que contiene un nombre de archivo por línea. Otra opción es enumerar los archivos en la ILC, separados por un espacio. No obstante, recuerde que las ILC tienen un límite de caracteres. Además, no olvide que la opción `--listfile` solamente afecta a los argumentos y no a las opciones.
Valor predeterminado: `false`.
Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

recurse

--recurse = `true|false`

Esta opción se utiliza para seleccionar ficheros dentro de subdirectorios (incluso en archivos ZIP). Si el valor es `true`, el argumento *`ArchivoEntrada`* del comando seleccionará el fichero seleccionado también en los subdirectorios. Por ejemplo: `test.zip|zip\test.xml` seleccionará los ficheros llamados `test.xml` en todos los subdirectorios de la carpeta ZIP. Si quiere puede usar los caracteres comodín `*` y `?`. Por ejemplo: `*.xml` seleccionaría todos los ficheros de la carpeta ZIP que tengan la extensión `.xml`. Valor predeterminado: `false`
Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

Catálogos y recursos globales

catalog

--catalog = *`ARCHIVO`*

Especifica la ruta de acceso absoluta a un archivo de catálogo que no está en el

archivo de catálogo raíz instalado. El valor predeterminado es la ruta de acceso absoluta del archivo de catálogo raíz instalado (`<carpeta-instalación>\Altova\RaptorXMLXBRLServer2018\etc\RootCatalog.xml`). Consulte el apartado Catálogos XML para obtener más información.

- user-catalog

  `--user-catalog =` *`ARCHIVO`*
  Especifica la ruta de acceso absoluta a un catálogo XML que debe utilizarse junto con el catálogo raíz. Consulte el apartado Catálogos XML para obtener más información.

- enable-globalresources

  `--enable-globalresources =` `true|false`
  Habilita la función de recursos globales. Valor predeterminado: `false`.
  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- globalresourceconfig [gc]

  `--gc | --globalresourceconfig =` *`VALOR`*
  Especifica la configuración activa del recurso global (y habilita los recursos globales).

- globalresourcefile [gr]

  `--gr | --globalresourcefile =` *`ARCHIVO`*
  Especifica el archivo de recursos globales (y habilita los recursos globales).

- Mensajes, errores, ayuda, tiempo de espera, versión
  - error-format

    `--error-format =` `text|shortxml|longxml`
    Especifica el formato de la salida de error. Los valores posibles son formatos de texto, XML y XML detallado (`longxml`). Valor predeterminado: `text`.

  - error-limit

    `--error-limit =` *`N`*
    Especifica el límite de errores con un valor comprendido entre `1` y `999`. El valor predeterminado es `100`. Cuando se alcanza el límite de error, se detiene la validación. Esta opción es muy práctica a la hora de limitar el uso del procesador durante la validación o transformación.

  - help

    `--help`
    Muestra el texto de ayuda para el comando. Por ejemplo `valany --h`. (Otra opción es usar el comando `help` con un argumento. Por ejemplo: `help valany`).

  - log-output

    `--log-output =` *`ARCHIVO`*
    Escribe el registro de salida en la URL de archivo indicada. Compruebe que la ILC tiene

permiso de escritura en la ubicación de destino.

- network-timeout

  `--network-timeout =` *`VALOR`*
  Indica el tiempo de espera en segundos para operaciones remotas de entrada y salida. Valor predeterminado: `40`.

- verbose

  `--verbose =` `true|false`
  Si el valor es `true`, se genera información adicional durante la validación. Valor predeterminado es `false`.
  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- verbose-output

  `--verbose-output =` *`ARCHIVO`*
  Escribe el resultado detallado en el *`ARCHIVO`* indicado.

- version

  `--version`
  Muestra el número de versión de RaptorXML+XBRL Server. Si se utiliza con un comando, escriba la opción `--version` antes del comando.

- warning-limit

  `--warning-limit =` *`VALUE`*
  Especifica el límite de advertencia en el rango `1-65535`. El procesamiento continua si se alcanza el límite pero no se registrarán más advertencias. El valor por defecto es `100`.

3.7.3 valavrojson (avrojson)

El comando **`valavrojson | avrojson`** valida un documento JSON con un esquema Avro.

Windows	**`RaptorXMLXBRL valavrojson | avrojson [opciones] --avroschema=`*`EsquemaAvro ArchivoJSON`***
Linux	**`raptorxmlxbrl valavrojson | avrojson [opciones] --avroschema=`*`EsquemaAvro ArchivoJSON`***
Mac	**`raptorxmlxbrl valavrojson | avrojson [opciones] --avroschema=`*`EsquemaAvro ArchivoJSON`***

El argumento *`ArchivoJSON`* especifica qué documento JSON se debe validar. La opción `--avroschema` especifica qué esquema Avro se debe usar para validar el documento JSON.

Si desea validar varios archivos JSON, tiene dos opciones: (i) enumerar los archivos que se deben validar en la línea de comandos, separados por espacios o (ii) enumerar los archivos que se deben validar en un archivo de texto (`.txt`) donde aparece un nombre de archivo por línea y dar este archivo de texto como argumento *`ArchivoJSON`* junto con la opción `--listfile` con valor `true` (*ver lista de opciones más abajo*).

Ejemplos

- **`raptorxmlxbrl`** `valavrojson --avroschema=c:\MiEsquemaAvro.avsc c:\MiArchivoJSON.json`
- **`raptorxmlxbrl`** `avrojson --avroschema=c:\MiEsquemaAvro.avsc c:\MiArchivoJSON.json`

Uso de mayúsculas/minúsculas y de barras diagonales en la línea de comandos

`RaptorXMLXBRL` *en Windows*
`raptorxmlxbrl` *en Unix (Linux, Mac)*

* Las minúsculas (`raptorxmlxbrl`) funcionan en todas las plataformas (Windows, Linux y Mac), mientras que la mezcla de mayúsculas y minúsculas (`RaptorXMLXBRL`) sólo funciona en Windows y Mac.
* En Linux y Mac utilice barras diagonales. En Windows utilice barras diagonales inversas.

Opciones

Las opciones del comando aparecen más abajo divididas en grupos. Los valores se pueden dar sin comillas excepto en estos dos casos: (i) cuando la cadena de valor contiene espacios y (ii) cuando en la descripción de la opción se indique explícitamente que es necesario el uso de comillas.

Procesamiento

listfile

`--listfile =` `true|false`
Si el valor es `true`, el argumento *`ArchivoEntrada`* del comando se entiende como un archivo de texto que contiene un nombre de archivo por línea. Otra opción es enumerar los archivos en la ILC, separados por un espacio. No obstante, recuerde que las ILC tienen un límite de caracteres. Además, no olvide que la opción `--listfile` solamente afecta a los argumentos y no a las opciones.
Valor predeterminado: `false`.
Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

recurse

`--recurse =` `true|false`
Esta opción se utiliza para seleccionar ficheros dentro de subdirectorios (incluso en archivos ZIP). Si el valor es `true`, el argumento *`ArchivoEntrada`* del comando seleccionará el fichero seleccionado también en los subdirectorios. Por ejemplo:

`test.zip|zip\test.xml` seleccionará los ficheros llamados `test.xml` en todos los subdirectorios de la carpeta ZIP. Si quiere puede usar los caracteres comodín `*` y `?`. Por ejemplo: `*.xml` seleccionaría todos los ficheros de la carpeta ZIP que tengan la extensión `.xml`. Valor predeterminado: `false`
Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- Catálogos y recursos globales
  - catalog

    `--catalog =` *`ARCHIVO`*
    Especifica la ruta de acceso absoluta a un archivo de catálogo que no está en el archivo de catálogo raíz instalado. El valor predeterminado es la ruta de acceso absoluta del archivo de catálogo raíz instalado (`<carpeta-instalación>\Altova\RaptorXMLXBRLServer2018\etc\RootCatalog.xml`). Consulte el apartado Catálogos XML para obtener más información.

  - user-catalog

    `--user-catalog =` *`ARCHIVO`*
    Especifica la ruta de acceso absoluta a un catálogo XML que debe utilizarse junto con el catálogo raíz. Consulte el apartado Catálogos XML para obtener más información.

  - enable-globalresources

    `--enable-globalresources =` `true|false`
    Habilita la función de recursos globales. Valor predeterminado: `false`.
    Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

  - globalresourceconfig [gc]

    `--gc | --globalresourceconfig =` *`VALOR`*
    Especifica la configuración activa del recurso global (y habilita los recursos globales).

  - globalresourcefile [gr]

    `--gr | --globalresourcefile =` *`ARCHIVO`*
    Especifica el archivo de recursos globales (y habilita los recursos globales).

- Mensajes, errores, ayuda, tiempo de espera, versión
  - error-format

    `--error-format =` `text|shortxml|longxml`
    Especifica el formato de la salida de error. Los valores posibles son formatos de texto, XML y XML detallado (`longxml`). Valor predeterminado: `text`.

  - error-limit

    `--error-limit =` *`N`*
    Especifica el límite de errores con un valor comprendido entre `1` y `999`. El valor

predeterminado es `100`. Cuando se alcanza el límite de error, se detiene la validación. Esta opción es muy práctica a la hora de limitar el uso del procesador durante la validación o transformación.

- help

  `--help`
  Muestra el texto de ayuda para el comando. Por ejemplo `valany --h`. (Otra opción es usar el comando `help` con un argumento. Por ejemplo: `help valany`).

- log-output

  `--log-output =` *`ARCHIVO`*
  Escribe el registro de salida en la URL de archivo indicada. Compruebe que la ILC tiene permiso de escritura en la ubicación de destino.

- network-timeout

  `--network-timeout =` *`VALOR`*
  Indica el tiempo de espera en segundos para operaciones remotas de entrada y salida. Valor predeterminado: `40`.

- verbose

  `--verbose =` `true|false`
  Si el valor es `true`, se genera información adicional durante la validación. Valor predeterminado es `false`.
  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- verbose-output

  `--verbose-output =` *`ARCHIVO`*
  Escribe el resultado detallado en el *`ARCHIVO`* indicado.

- version

  `--version`
  Muestra el número de versión de RaptorXML+XBRL Server. Si se utiliza con un comando, escriba la opción `--version` antes del comando.

- warning-limit

  `--warning-limit =` *`VALUE`*
  Especifica el límite de advertencia en el rango `1-65535`. El procesamiento continua si se alcanza el límite pero no se registrarán más advertencias. El valor por defecto es `100`.

3.7.4 valavroschema (avroschema)

El comando **valavroschema | avroschema** valida documentos de esquema Avro con la especificación Avro schema.

Windows	**RaptorXMLXBRL valavroschema | avroschema [opciones] *EsquemaAvro***
Linux	**raptorxmlxbrl valavroschema | avroschema [opciones] *EsquemaAvro***
Mac	**raptorxmlxbrl valavroschema | avroschema [opciones] *EsquemaAvro***

El argumento *EsquemaAvro* es el documento de esquema Avro que se debe validar.

Si desea validar varios documentos de esquema Avro, tiene dos opciones: (i) enumerar los archivos de esquema que se deben validar en la línea de comandos, separados por espacios o (ii) enumerar los archivos que se deben validar en un archivo de texto (`.txt`) donde aparece un nombre de archivo por línea y dar este archivo de texto como argumento *EsquemaAvro* junto con la opción `--listfile` con valor `true` (*ver lista de opciones más abajo*).

Ejemplos

- **raptorxmlxbrl** `valavroschema c:\MiEsquemaAvro.avsc`
- **raptorxmlxbrl** `valavroschema c:\MiEsquemaAvro01.avsc c:\MiEsquemaAvro02.avsc`
- **raptorxmlxbrl** `avroschema--listfile=true c:\MiListaArchivos.txt`

Uso de mayúsculas/minúsculas y de barras diagonales en la línea de comandos

RaptorXMLXBRL *en Windows*
raptorxmlxbrl *en Unix (Linux, Mac)*

* Las minúsculas (`raptorxmlxbrl`) funcionan en todas las plataformas (Windows, Linux y Mac), mientras que la mezcla de mayúsculas y minúsculas (`RaptorXMLXBRL`) sólo funciona en Windows y Mac.
* En Linux y Mac utilice barras diagonales. En Windows utilice barras diagonales inversas.

Opciones

Las opciones del comando aparecen más abajo divididas en grupos. Los valores se pueden dar sin comillas excepto en estos dos casos: (i) cuando la cadena de valor contiene espacios y (ii) cuando en la descripción de la opción se indique explícitamente que es necesario el uso de comillas.

Procesamiento

listfile

--listfile = `true|false`

Si el valor es `true`, el argumento *`ArchivoEntrada`* del comando se entiende como un archivo de texto que contiene un nombre de archivo por línea. Otra opción es enumerar los archivos en la ILC, separados por un espacio. No obstante, recuerde que las ILC tienen un límite de caracteres. Además, no olvide que la opción `--listfile` solamente afecta a los argumentos y no a las opciones.
Valor predeterminado: `false`.
Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- recurse

`--recurse =` `true|false`
Esta opción se utiliza para seleccionar ficheros dentro de subdirectorios (incluso en archivos ZIP). Si el valor es `true`, el argumento *`ArchivoEntrada`* del comando seleccionará el fichero seleccionado también en los subdirectorios. Por ejemplo: `test.zip|zip\test.xml` seleccionará los ficheros llamados `test.xml` en todos los subdirectorios de la carpeta ZIP. Si quiere puede usar los caracteres comodín `*` y `?`. Por ejemplo: `*.xml` seleccionaría todos los ficheros de la carpeta ZIP que tengan la extensión `.xml`. Valor predeterminado: `false`
Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- Catálogos y recursos globales
  - catalog

`--catalog =` *`ARCHIVO`*
Especifica la ruta de acceso absoluta a un archivo de catálogo que no está en el archivo de catálogo raíz instalado. El valor predeterminado es la ruta de acceso absoluta del archivo de catálogo raíz instalado (`<carpeta-instalación>\Altova\RaptorXMLXBRLServer2018\etc\RootCatalog.xml`). Consulte el apartado Catálogos XML para obtener más información.

  - user-catalog

`--user-catalog =` *`ARCHIVO`*
Especifica la ruta de acceso absoluta a un catálogo XML que debe utilizarse junto con el catálogo raíz. Consulte el apartado Catálogos XML para obtener más información.

  - enable-globalresources

`--enable-globalresources =` `true|false`
Habilita la función de recursos globales. Valor predeterminado: `false`.
Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

  - globalresourceconfig [gc]

`--gc | --globalresourceconfig =` *`VALOR`*
Especifica la configuración activa del recurso global (y habilita los recursos globales).

  - globalresourcefile [gr]

--gr | --globalresourcefile = *ARCHIVO*
Especifica el archivo de recursos globales (y habilita los recursos globales).

- Mensajes, errores, ayuda, tiempo de espera, versión
  - error-format

    --error-format = `text|shortxml|longxml`
    Especifica el formato de la salida de error. Los valores posibles son formatos de texto, XML y XML detallado (`longxml`). Valor predeterminado: `text`.

  - error-limit

    --error-limit = *N*
    Especifica el límite de errores con un valor comprendido entre `1` y `999`. El valor predeterminado es `100`. Cuando se alcanza el límite de error, se detiene la validación. Esta opción es muy práctica a la hora de limitar el uso del procesador durante la validación o transformación.

  - help

    --help
    Muestra el texto de ayuda para el comando. Por ejemplo `valany --h`. (Otra opción es usar el comando `help` con un argumento. Por ejemplo: `help valany`).

  - log-output

    --log-output = *ARCHIVO*
    Escribe el registro de salida en la URL de archivo indicada. Compruebe que la ILC tiene permiso de escritura en la ubicación de destino.

  - network-timeout

    --network-timeout = *VALOR*
    Indica el tiempo de espera en segundos para operaciones remotas de entrada y salida. Valor predeterminado: `40`.

  - verbose

    --verbose = `true|false`
    Si el valor es `true`, se genera información adicional durante la validación. Valor predeterminado es `false`.
    Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

  - verbose-output

    --verbose-output = *ARCHIVO*
    Escribe el resultado detallado en el *ARCHIVO* indicado.

  - version

    --version
    Muestra el número de versión de RaptorXML+XBRL Server. Si se utiliza con un

 © 2018 Altova GmbH

comando, escriba la opción `--version` antes del comando.

- warning-limit

  `--warning-limit =` *`VALUE`*
  Especifica el límite de advertencia en el rango `1-65535`. El procesamiento continua si se alcanza el límite pero no se registrarán más advertencias. El valor por defecto es `100`.

3.7.5 valjsonschema (jsonschema)

El comando **`valjsonschema | jsonschema`** valida documentos de esquema JSON con las especificaciones JSON Schema Draft 4.

Windows	**`RaptorXMLXBRL valjsonschema	jsonschema [opciones]`** ***`ArchivoEntrada`***
Linux	**`raptorxmlxbrl valjsonschema	jsonschema [opciones]`** ***`ArchivoEntrada`***
Mac	**`raptorxmlxbrl valjsonschema	jsonschema [opciones]`** ***`ArchivoEntrada`***

El argumento *`ArchivoEntrada`* es el documento de esquema JSON que se debe validar.

Si desea validar varios documentos, tiene dos opciones: (i) enumerar los archivos que se deben validar en la línea de comandos, separados por espacios o (ii) enumerar los archivos que se deben validar en un archivo de texto (`.txt`) donde aparece un nombre de archivo por línea y dar este archivo de texto como argumento *`ArchivoEntrada`* junto con la opción `--listfile` con valor `true` (*ver lista de opciones más abajo*).

Ejemplos

- **`raptorxmlxbrl`** `valjsonschema c:\MiEsquemaJSON.json`
- **`raptorxmlxbrl`** `jsonschema c:\MiEsquemaJSON-01.json c:\MiEsquemaJSON-02.json`
- **`raptorxmlxbrl`** `jsonschema --listfile=true c:\ListaArchivos.txt`

- Uso de mayúsculas/minúsculas y de barras diagonales en la línea de comandos

  `RaptorXMLXBRL` *en Windows*
  `raptorxmlxbrl` *en Unix (Linux, Mac)*

  * Las minúsculas (`raptorxmlxbrl`) funcionan en todas las plataformas (Windows, Linux y Mac), mientras que la mezcla de mayúsculas y minúsculas (`RaptorXMLXBRL`) sólo funciona en Windows y Mac.
  * En Linux y Mac utilice barras diagonales. En Windows utilice barras diagonales inversas.

Opciones

Las opciones del comando aparecen más abajo divididas en grupos. Los valores se pueden dar sin comillas excepto en estos dos casos: (i) cuando la cadena de valor contiene espacios y (ii) cuando en la descripción de la opción se indique explícitamente que es necesario el uso de comillas.

- Validación y procesamiento
  - listfile

    `--listfile =` `true|false`

    Si el valor es `true`, el argumento *`ArchivoEntrada`* del comando se entiende como un archivo de texto que contiene un nombre de archivo por línea. Otra opción es enumerar los archivos en la ILC, separados por un espacio. No obstante, recuerde que las ILC tienen un límite de caracteres. Además, no olvide que la opción `--listfile` solamente afecta a los argumentos y no a las opciones.
    Valor predeterminado: `false`.
    Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

  - recurse

    `--recurse =` `true|false`

    Esta opción se utiliza para seleccionar ficheros dentro de subdirectorios (incluso en archivos ZIP). Si el valor es `true`, el argumento *`ArchivoEntrada`* del comando seleccionará el fichero seleccionado también en los subdirectorios. Por ejemplo: `test.zip|zip\test.xml` seleccionará los ficheros llamados `test.xml` en todos los subdirectorios de la carpeta ZIP. Si quiere puede usar los caracteres comodín `*` y `?`. Por ejemplo: `*.xml` seleccionaría todos los ficheros de la carpeta ZIP que tengan la extensión `.xml`. Valor predeterminado: `false`
    Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- Catálogos y recursos globales
  - catalog

    `--catalog =` *`ARCHIVO`*

    Especifica la ruta de acceso absoluta a un archivo de catálogo que no está en el archivo de catálogo raíz instalado. El valor predeterminado es la ruta de acceso absoluta del archivo de catálogo raíz instalado (`<carpeta-instalación>\Altova\RaptorXMLXBRLServer2018\etc\RootCatalog.xml`). Consulte el apartado Catálogos XML para obtener más información.

  - user-catalog

    `--user-catalog =` *`ARCHIVO`*

    Especifica la ruta de acceso absoluta a un catálogo XML que debe utilizarse junto con el catálogo raíz. Consulte el apartado Catálogos XML para obtener más información.

  - enable-globalresources

 © 2018 Altova GmbH

--enable-globalresources = `true|false`
Habilita la función de recursos globales. Valor predeterminado: `false`.
Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- globalresourceconfig [gc]

  --gc | --globalresourceconfig = *VALOR*
  Especifica la configuración activa del recurso global (y habilita los recursos globales).

- globalresourcefile [gr]

  --gr | --globalresourcefile = *ARCHIVO*
  Especifica el archivo de recursos globales (y habilita los recursos globales).

- Mensajes, errores, ayuda, tiempo de espera, versión
  - error-format

    --error-format = `text|shortxml|longxml`
    Especifica el formato de la salida de error. Los valores posibles son formatos de texto, XML y XML detallado (`longxml`). Valor predeterminado: `text`.

  - error-limit

    --error-limit = *N*
    Especifica el límite de errores con un valor comprendido entre `1` y `999`. El valor predeterminado es `100`. Cuando se alcanza el límite de error, se detiene la validación. Esta opción es muy práctica a la hora de limitar el uso del procesador durante la validación o transformación.

  - help

    --help
    Muestra el texto de ayuda para el comando. Por ejemplo `valany --h`. (Otra opción es usar el comando `help` con un argumento. Por ejemplo: `help valany`).

  - log-output

    --log-output = *ARCHIVO*
    Escribe el registro de salida en la URL de archivo indicada. Compruebe que la ILC tiene permiso de escritura en la ubicación de destino.

  - network-timeout

    --network-timeout = *VALOR*
    Indica el tiempo de espera en segundos para operaciones remotas de entrada y salida. Valor predeterminado: `40`.

  - verbose

    --verbose = `true|false`
    Si el valor es `true`, se genera información adicional durante la validación. Valor predeterminado es `false`.

Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- verbose-output

  `--verbose-output =` *`ARCHIVO`*
  Escribe el resultado detallado en el *`ARCHIVO`* indicado.

- version

  `--version`
  Muestra el número de versión de RaptorXML+XBRL Server. Si se utiliza con un comando, escriba la opción `--version` antes del comando.

- warning-limit

  `--warning-limit =` *`VALUE`*
  Especifica el límite de advertencia en el rango `1-65535`. El procesamiento continua si se alcanza el límite pero no se registrarán más advertencias. El valor por defecto es `100`.

3.7.6 valjson (json)

El comando **`valjson | json`** valida documentos de instancia JSON con el esquema JSON dado con la opción `--schema`.

Windows **`RaptorXMLXBRL valjson | json [opciones] --jsonschema=`*`Archivo ArchivoEntrada`***

Linux **`raptorxmlxbrl valjson | json [opciones] --jsonschema=`*`Archivo ArchivoEntrada`***

Mac **`raptorxmlxbrl valjson | json [opciones] --jsonschema=`*`Archivo ArchivoEntrada`***

El argumento *`ArchivoEntrada`* es el documento de instancia que se debe validar.

Si desea validar varios documentos, tiene dos opciones: (i) enumerar los archivos que se deben validar en la línea de comandos, separados por espacios o (ii) enumerar los archivos que se deben validar en un archivo de texto (`.txt`) donde aparece un nombre de archivo por línea y dar este archivo de texto como argumento *`ArchivoEntrada`* junto con la opción `--listfile` con valor `true` (*ver lista de opciones más abajo*).

Ejemplos

- **`raptorxmlxbrl`** `valjson --jsonschema=c:\MiEsquemaJSON.json c:`

\MiInstanciaJSON.json

- **raptorxmlxbrl** json --jsonschema=c:\MiEsquemaJSON.json c:\MiInstanciaJSON-01.json c:\MiInstanciaJSON-02.json
- **raptorxmlxbrl** json --jsonschema=c:\MiEsquemaJSON.json --listfile=true c:\ListaArchivos.txt

Uso de mayúsculas/minúsculas y de barras diagonales en la línea de comandos

RaptorXMLXBRL *en Windows*
raptorxmlxbrl *en Unix (Linux, Mac)*

* Las minúsculas (raptorxmlxbrl) funcionan en todas las plataformas (Windows, Linux y Mac), mientras que la mezcla de mayúsculas y minúsculas (RaptorXMLXBRL) sólo funciona en Windows y Mac.
* En Linux y Mac utilice barras diagonales. En Windows utilice barras diagonales inversas.

Opciones

Las opciones del comando aparecen más abajo divididas en grupos. Los valores se pueden dar sin comillas excepto en estos dos casos: (i) cuando la cadena de valor contiene espacios y (ii) cuando en la descripción de la opción se indique explícitamente que es necesario el uso de comillas.

Validación y procesamiento

schema, **jsonschema**

--schema = *ARCHIVO*, **--jsonschema =** *ARCHIVO*
Especifica el documento del esquema JSON que se va usar para la validación de documentos de instancia JSON.

listfile

--listfile = true|false
Si el valor es true, el argumento *ArchivoEntrada* del comando se entiende como un archivo de texto que contiene un nombre de archivo por línea. Otra opción es enumerar los archivos en la ILC, separados por un espacio. No obstante, recuerde que las ILC tienen un límite de caracteres. Además, no olvide que la opción --listfile solamente afecta a los argumentos y no a las opciones.
Valor predeterminado: false.
Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en true.

recurse

--recurse = true|false
Esta opción se utiliza para seleccionar ficheros dentro de subdirectorios (incluso en archivos ZIP). Si el valor es true, el argumento *ArchivoEntrada* del comando seleccionará el fichero seleccionado también en los subdirectorios. Por ejemplo: test.zip|zip\test.xml seleccionará los ficheros llamados test.xml en todos los subdirectorios de la carpeta ZIP. Si quiere puede usar los caracteres comodín * y ?. Por ejemplo: *.xml seleccionaría todos los ficheros de la carpeta ZIP que tengan la extensión .xml. Valor predeterminado: false

Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- json5

  `--json5 =` `true|false`
  Habilita la compatibilidad con JSON5. Valor predeterminado: `false`.
  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- Catálogos y recursos globales
  - catalog

    `--catalog =` *`ARCHIVO`*
    Especifica la ruta de acceso absoluta a un archivo de catálogo que no está en el archivo de catálogo raíz instalado. El valor predeterminado es la ruta de acceso absoluta del archivo de catálogo raíz instalado (`<carpeta-instalación>\Altova\RaptorXMLXBRLServer2018\etc\RootCatalog.xml`). Consulte el apartado Catálogos XML para obtener más información.

  - user-catalog

    `--user-catalog =` *`ARCHIVO`*
    Especifica la ruta de acceso absoluta a un catálogo XML que debe utilizarse junto con el catálogo raíz. Consulte el apartado Catálogos XML para obtener más información.

  - enable-globalresources

    `--enable-globalresources =` `true|false`
    Habilita la función de recursos globales. Valor predeterminado: `false`.
    Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

  - globalresourceconfig [gc]

    `--gc | --globalresourceconfig =` *`VALOR`*
    Especifica la configuración activa del recurso global (y habilita los recursos globales).

  - globalresourcefile [gr]

    `--gr | --globalresourcefile =` *`ARCHIVO`*
    Especifica el archivo de recursos globales (y habilita los recursos globales).

- Mensajes, errores, ayuda, tiempo de espera, versión
  - error-format

    `--error-format =` `text|shortxml|longxml`
    Especifica el formato de la salida de error. Los valores posibles son formatos de texto, XML y XML detallado (`longxml`). Valor predeterminado: `text`.

  - error-limit

    `--error-limit =` *`N`*

Especifica el límite de errores con un valor comprendido entre `1` y `999`. El valor predeterminado es `100`. Cuando se alcanza el límite de error, se detiene la validación. Esta opción es muy práctica a la hora de limitar el uso del procesador durante la validación o transformación.

- help

  `--help`
  Muestra el texto de ayuda para el comando. Por ejemplo `valany --h`. (Otra opción es usar el comando `help` con un argumento. Por ejemplo: `help valany`).

- log-output

  `--log-output =` *`ARCHIVO`*
  Escribe el registro de salida en la URL de archivo indicada. Compruebe que la ILC tiene permiso de escritura en la ubicación de destino.

- network-timeout

  `--network-timeout =` *`VALOR`*
  Indica el tiempo de espera en segundos para operaciones remotas de entrada y salida. Valor predeterminado: `40`.

- verbose

  `--verbose =` `true|false`
  Si el valor es `true`, se genera información adicional durante la validación. Valor predeterminado es `false`.
  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- verbose-output

  `--verbose-output =` *`ARCHIVO`*
  Escribe el resultado detallado en el *`ARCHIVO`* indicado.

- version

  `--version`
  Muestra el número de versión de RaptorXML+XBRL Server. Si se utiliza con un comando, escriba la opción `--version` antes del comando.

- warning-limit

  `--warning-limit =` *`VALUE`*
  Especifica el límite de advertencia en el rango `1-65535`. El procesamiento continua si se alcanza el límite pero no se registrarán más advertencias. El valor por defecto es `100`.

3.7.7 wfjson

El comando **wfjson** comprueba si el formato de los documentos JSON se ajusta a las normas de la especificación ECMA-404.

Windows	**RaptorXMLXBRL wfjson [opciones] *ArchivoEntrada***
Linux	**raptorxmlxbrl wfjson [opciones] *ArchivoEntrada***
Mac	**raptorxmlxbrl wfjson [opciones] *ArchivoEntrada***

El argumento *ArchivoEntrada* es el documento JSON (esquema o instancia) cuyo formato debe revisarse.

Si desea revisar el formato de varios documentos, tiene dos opciones: (i) enumerar los archivos que se deben revisar en la línea de comandos, separados por espacios o (ii) enumerar los archivos que se deben revisar en un archivo de texto (`.txt`) donde aparece un nombre de archivo por línea y dar este archivo de texto como argumento *ArchivoEntrada* junto con la opción `--listfile` con valor `true` (*ver lista de opciones más abajo*).

Ejemplos

- **raptorxmlxbrl** `wfjson c:\MiArchivoJSON.json`
- **raptorxmlxbrl** `wfjson c:\MiArchivoJSON-01.json c:\MiArchivoJSON-02.json`
- **raptorxmlxbrl** `wfjson --listfile=true c:\ListaArchivos.txt`

Uso de mayúsculas/minúsculas y de barras diagonales en la línea de comandos

RaptorXMLXBRL *en Windows*
raptorxmlxbrl *en Unix (Linux, Mac)*

* Las minúsculas (`raptorxmlxbrl`) funcionan en todas las plataformas (Windows, Linux y Mac), mientras que la mezcla de mayúsculas y minúsculas (`RaptorXMLXBRL`) sólo funciona en Windows y Mac.
* En Linux y Mac utilice barras diagonales. En Windows utilice barras diagonales inversas.

Opciones

Las opciones del comando aparecen más abajo divididas en grupos. Los valores se pueden dar sin comillas excepto en estos dos casos: (i) cuando la cadena de valor contiene espacios y (ii) cuando en la descripción de la opción se indique explícitamente que es necesario el uso de comillas.

Validación y procesamiento

json5

--json5 = `true|false`

Habilita la compatibilidad con JSON5. Valor predeterminado: `false`.
Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- listfile

  `--listfile =` `true|false`
  Si el valor es `true`, el argumento *`ArchivoEntrada`* del comando se entiende como un archivo de texto que contiene un nombre de archivo por línea. Otra opción es enumerar los archivos en la ILC, separados por un espacio. No obstante, recuerde que las ILC tienen un límite de caracteres. Además, no olvide que la opción `--listfile` solamente afecta a los argumentos y no a las opciones.
  Valor predeterminado: `false`.
  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- recurse

  `--recurse =` `true|false`
  Esta opción se utiliza para seleccionar ficheros dentro de subdirectorios (incluso en archivos ZIP). Si el valor es `true`, el argumento *`ArchivoEntrada`* del comando seleccionará el fichero seleccionado también en los subdirectorios. Por ejemplo: `test.zip|zip\test.xml` seleccionará los ficheros llamados `test.xml` en todos los subdirectorios de la carpeta ZIP. Si quiere puede usar los caracteres comodín `*` y `?`. Por ejemplo: `*.xml` seleccionaría todos los ficheros de la carpeta ZIP que tengan la extensión `.xml`. Valor predeterminado: `false`
  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- Catálogos y recursos globales
  - catalog

    `--catalog =` *`ARCHIVO`*
    Especifica la ruta de acceso absoluta a un archivo de catálogo que no está en el archivo de catálogo raíz instalado. El valor predeterminado es la ruta de acceso absoluta del archivo de catálogo raíz instalado (`<carpeta-instalación>\Altova\RaptorXMLXBRLServer2018\etc\RootCatalog.xml`). Consulte el apartado Catálogos XML para obtener más información.

  - user-catalog

    `--user-catalog =` *`ARCHIVO`*
    Especifica la ruta de acceso absoluta a un catálogo XML que debe utilizarse junto con el catálogo raíz. Consulte el apartado Catálogos XML para obtener más información.

  - enable-globalresources

    `--enable-globalresources =` `true|false`
    Habilita la función de recursos globales. Valor predeterminado: `false`.
    Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- globalresourceconfig [gc]

  --gc | --globalresourceconfig = *VALOR*
  Especifica la configuración activa del recurso global (y habilita los recursos globales).

- globalresourcefile [gr]

  --gr | --globalresourcefile = *ARCHIVO*
  Especifica el archivo de recursos globales (y habilita los recursos globales).

- Mensajes, errores, ayuda, tiempo de espera, versión
  - error-format

    --error-format = `text|shortxml|longxml`
    Especifica el formato de la salida de error. Los valores posibles son formatos de texto, XML y XML detallado (`longxml`). Valor predeterminado: `text`.

  - error-limit

    --error-limit = *N*
    Especifica el límite de errores con un valor comprendido entre `1` y `999`. El valor predeterminado es `100`. Cuando se alcanza el límite de error, se detiene la validación. Esta opción es muy práctica a la hora de limitar el uso del procesador durante la validación o transformación.

  - help

    --help
    Muestra el texto de ayuda para el comando. Por ejemplo `valany --h`. (Otra opción es usar el comando `help` con un argumento. Por ejemplo: `help valany`).

  - log-output

    --log-output = *ARCHIVO*
    Escribe el registro de salida en la URL de archivo indicada. Compruebe que la ILC tiene permiso de escritura en la ubicación de destino.

  - network-timeout

    --network-timeout = *VALOR*
    Indica el tiempo de espera en segundos para operaciones remotas de entrada y salida. Valor predeterminado: `40`.

  - verbose

    --verbose = `true|false`
    Si el valor es `true`, se genera información adicional durante la validación. Valor predeterminado es `false`.
    Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

  - verbose-output

    --verbose-output = *ARCHIVO*

 © 2018 Altova GmbH

Escribe el resultado detallado en el *ARCHIVO* indicado.

- version

  --version
  Muestra el número de versión de RaptorXML+XBRL Server. Si se utiliza con un comando, escriba la opción `--version` antes del comando.

- warning-limit

  --warning-limit = *VALUE*
  Especifica el límite de advertencia en el rango `1-65535`. El procesamiento continua si se alcanza el límite pero no se registrarán más advertencias. El valor por defecto es `100`.

3.8 Comando valany

El comando **valany** es un comando de uso general que valida un documento en función de su tipo. El tipo de documento de entrada se detecta automáticamente y la correspondiente validación se lleva a cabo según lo dispuesto por la especificación pertinente. El argumento *ArchivoEntrada* es el documento que se debe validar. Recuerde que como puede dar un documento como máximo como argumento del comando.

Windows **RaptorXMLXBRL valany [opciones] *ArchivoEntrada***

Linux **raptorxmlxbrl valany [opciones] *ArchivoEntrada***

Mac **raptorxmlxbrl valany [opciones] *ArchivoEntrada***

El comando **valany** se encarga de los tipos de validación que se enumeran a continuación. Sus opciones son todas las opciones disponibles para cada uno de los comandos de validación correspondientes. Consulte la descripción de cada comando de validación para ver su lista de opciones.

- valdtd (dtd)
- valxsd (xsd)
- valxml-withdtd (xml)
- valxml-withxsd (xsi)
- valxslt
- valxquery
- valxbrl (xbrl)
- valinlinexbrl (ixbrl)
- valxbrltaxonomy (dts)
- valavrojson (avrojson)

Ejemplos

- **raptorxmlxbrl** valany c:\Test.xsd
- **raptorxmlxbrl** valany --error-format=text c:\Test.xbrl

Uso de mayúsculas/minúsculas y de barras diagonales en la línea de comandos

RaptorXMLXBRL *en Windows*
raptorxmlxbrl *en Unix (Linux, Mac)*

* Las minúsculas (raptorxmlxbrl) funcionan en todas las plataformas (Windows, Linux y Mac), mientras que la mezcla de mayúsculas y minúsculas (RaptorXMLXBRL) sólo funciona en Windows y Mac.
* En Linux y Mac utilice barras diagonales. En Windows utilice barras diagonales inversas.

 © 2018 Altova GmbH

Opciones

Consulte la descripción de cada comando de validación para ver su lista de opciones. Es importante recordar que, aunque la mayoría de los comandos de validación aceptan varios documentos de entrada, el comando **`valany`** acepta un documento de entrada como máximo. Por tanto, opciones como `--listfile` no son pertinentes en el caso del comando **`valany`**.

3.9 Comando script

El comando **`script`** ejecuta un script de Python 3 que utiliza la API de Python de RaptorXML.

Windows	**`RaptorXMLXBRL script [opciones] Archivo`**
Linux	**`raptorxmlxbrl script [opciones] Archivo`**
Mac	**`raptorxmlxbrl script [opciones] Archivo`**

El argumento *`Archivo`* es la ruta de acceso del script de Python que se debe ejecutar. Hay varias opciones más, que puede consultar con este comando:

Windows	**`RaptorXMLXBRL script [-h | --help]`**
Linux	**`raptorxmlxbrl script [-h | --help]`**
Mac	**`raptorxmlxbrl script [-h | --help]`**

Ejemplos

- **`raptorxmlxbrl`** `script c:\MiScriptPython.py`
- **`raptorxmlxbrl`** `script -h`
- **`raptorxmlxbrl`** `script` # *Sin un archivo de script se inicia una shell de Python interactiva*
- **`raptorxmlxbrl`** `script -m pip` # *Carga y ejecuta el módulo* `pip`*. Ver apartado "Opciones" más abajo*

Uso de mayúsculas/minúsculas y de barras diagonales en la línea de comandos

`RaptorXMLXBRL` *en Windows*
`raptorxmlxbrl` *en Unix (Linux, Mac)*

* Las minúsculas (`raptorxmlxbrl`) funcionan en todas las plataformas (Windows, Linux y Mac), mientras que la mezcla de mayúsculas y minúsculas (`RaptorXMLXBRL`) sólo funciona en Windows y Mac.
* En Linux y Mac utilice barras diagonales. En Windows utilice barras diagonales inversas.

Opciones

Las opciones y argumentos dados tras el comando `script` se reenvian al intérprete de Python directamente. Consulte la página de la documentación de Python para ver una lista de todas las opciones disponibles (https://docs.python.org/3/using/cmdline.html).

 © 2018 Altova GmbH

3.10 Comandos de ayuda y licencias

3.10.1 help

El comando **help** toma un solo argumento: el nombre del comando para el que desea obtener información adicional. Muestra la sintaxis del comando y otra información importante para poder ejecutar el comando correctamente.

Windows	**RaptorXMLXBRL help *Comando***
Linux	**raptorxmlxbrl help *Comando***
Mac	**raptorxmlxbrl help *Comando***

Nota: si ejecuta el comando **help** sin ningún argumento, aparece la ayuda sobre todos los comandos de la ILC, cada uno con una breve descripción.

Ejemplo

```
raptorxmlxbrl help valany
```

Este comando contiene un argumento, el comando `valany`. Cuando se ejecuta este comando, aparece información de ayuda sobre el comando `valany`.

Uso de mayúsculas/minúsculas y de barras diagonales en la línea de comandos

RaptorXMLXBRL *en Windows*
raptorxmlxbrl *en Unix (Linux, Mac)*

* Las minúsculas (`raptorxmlxbrl`) funcionan en todas las plataformas (Windows, Linux y Mac), mientras que la mezcla de mayúsculas y minúsculas (`RaptorXMLXBRL`) sólo funciona en Windows y Mac.
* En Linux y Mac utilice barras diagonales. En Windows utilice barras diagonales inversas.

La opción `--help`

También puede ver la información de ayuda de un comando si usa la opción `--help` con dicho comando. Por ejemplo, si usamos la opción `--help` con el comando `valany`:

```
raptorxmlxbrl valany --help
```

conseguimos el mismo resultado que cuando usamos el comando `help` con el argumento `valany`:

```
raptorxmlxbrl help valany
```

En ambos casos aparece información de ayuda sobre el comando `valany`.

3.10.2 licenseserver

El comando `licenseserver` registra RaptorXML+XBRL Server con el servidor de licencias Altova LicenseServer. Este comando toma como argumento el nombre o la dirección IP del servidor en el que se ejecuta LicenseServer.

Windows	**RaptorXMLXBRL licenseserver [opciones]** ***Servidor-O-Dirección-IP***
Linux	**raptorxmlxbrl licenseserver [opciones]** ***Servidor-O-Dirección-IP***
Mac	**raptorxmlxbrl licenseserver [opciones]** ***Servidor-O-Dirección-IP***

Tras registrar RaptorXML+XBRL Server con LicenseServer, se devuelve la URL de la interfaz web de LicenseServer. Escriba la URL en una ventana del explorador para acceder a la interfaz web y después siga los pasos descritos en la documentación de LicenseServer para asignar una licencia a RaptorXML+XBRL Server.

Ejemplo

```
raptorxmlxbrl licenseserver DOC.altova.com
```

El comando especifica que el equipo `DOC.altova.com` es el equipo donde se ejecuta el servidor de licencias LicenseServer.

Uso de mayúsculas/minúsculas y de barras diagonales en la línea de comandos

`RaptorXMLXBRL` *en Windows*
`raptorxmlxbrl` *en Unix (Linux, Mac)*

* Las minúsculas (`raptorxmlxbrl`) funcionan en todas las plataformas (Windows, Linux y Mac), mientras que la mezcla de mayúsculas y minúsculas (`RaptorXMLXBRL`) sólo funciona en Windows y Mac.
* En Linux y Mac utilice barras diagonales. En Windows utilice barras diagonales inversas.

Opciones

Las opciones de este comando son:

`--j|json=true|false`
Imprime el resultado del intento de registro en formato de objeto JSON analizable por computador.

`--h|help`

 © 2018 Altova GmbH

Muestra el texto de ayuda del comando.

--version
Muestra el número de versión de RaptorXML+XBRL Server. Esta opción debe escribirse antes del comando: **raptorxmlxbrl** `--version licenseserver`.

3.10.3 assignlicense

El comando `assignlicense` solamente está disponible en Windows. Carga un archivo de licencias al servidor Altova LicenseServer donde está registrado RaptorXML+XBRL Server y asigna la licencia a RaptorXML+XBRL Server. Toma la URL del archivo de licencia como argumento. Este comando también sirve para comprobar la validez de las licencias.

Windows	**RaptorXMLXBRL assignlicense [opciones]** ***ARCHIVO-LICENCIAS***
Linux	***no está disponible***
Mac	***no está disponible***

Ejemplo

- **raptorxmlxbrl** `assignlicense C:\licensepool\miclave.lic`
- **raptorxmlxbrl** `assignlicense --test-only=true C:\licensepool\miclave.lic`

Uso de mayúsculas/minúsculas y de barras diagonales en la línea de comandos

RaptorXMLXBRL *en Windows*
raptorxmlxbrl *en Unix (Linux, Mac)*

* Las minúsculas (`raptorxmlxbrl`) funcionan en todas las plataformas (Windows, Linux y Mac), mientras que la mezcla de mayúsculas y minúsculas (`RaptorXMLXBRL`) sólo funciona en Windows y Mac.
* En Linux y Mac utilice barras diagonales. En Windows utilice barras diagonales inversas.

Opciones

Estas son las opciones disponibles:

--t|test-only=true|false
Si el valor es `true`, entonces la licencia se carga a LicenseServer y se valida, pero no se asigna al producto.

Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

--h|help
Muestra el texto de ayuda del comando.

--version
Muestra el número de versión de RaptorXML+XBRL Server. Esta opción debe escribirse antes del comando: **raptorxmlxbrl** `--version licenseserver`.

3.10.4 verifylicense

El comando `verifylicense` solamente está disponible en Windows y sirve para comprobar si RaptorXML+XBRL Server tiene asignada una licencia e incluso comprobar si determinada clave de licencia ya está asignada a RaptorXML+XBRL Server. Toma como argumento la clave de licencia. El comando devuelve una declaración sobre el estado de la licencia o sobre la validez de la clave de licencia dada.

Windows	**RaptorXMLXBRL verifylicense [opciones]**
Linux	***no está disponible***
Mac	***no está disponible***

Ejemplo

- **raptorxmlxbrl** `verifylicense`
- **raptorxmlxbrl** `verifylicense --license-key=a-39-character-long-license-code-(7x5, plus 4 hyphens)`
- **raptorxmlxbrl** `verifylicense --l=a-39-character-long-license-code-(7x5, plus 4 hyphens)`

Las declaraciones que se devuelven son de este tipo:

- `El producto tiene una licencia válida`
- `El producto no tiene una licencia válida`
- `La clave de licencia key AAAAAAA-BBBBBBB-CCCCCCC-DDDDDDD-EEEEEEE está asignada a este producto`
- `La clave de licencia AAAAAAA-BBBBBBB-CCCCCCC-DDDDDDD-EEEEEEE no está asignada a este producto`

Uso de mayúsculas/minúsculas y de barras diagonales en la línea de comandos

RaptorXMLXBRL *en Windows*
raptorxmlxbrl *en Unix (Linux, Mac)*

* Las minúsculas (`raptorxmlxbrl`) funcionan en todas las plataformas (Windows, Linux y Mac), mientras que la mezcla de mayúsculas y minúsculas (`RaptorXMLXBRL`) sólo funciona en Windows y Mac.
* En Linux y Mac utilice barras diagonales. En Windows utilice barras diagonales inversas.

 © 2018 Altova GmbH

Opciones

Estas son las opciones disponibles:

--l|license-key=*VALOR*

`VALOR` es la clave de licencia de 39 caracteres sin delimitadores. La clave está compuesta por cinco bloques de 7 caracteres cada uno separados por guiones.

--h|help

Muestra el texto de ayuda del comando.

--version

Muestra el número de versión de RaptorXML+XBRL Server. Esta opción debe escribirse antes del comando: **raptorxmlxbrl** `--version licenseserver`.

3.11 Comandos de localización

Si quiere puede localizar (traducir) la aplicación RaptorXML en cualquier idioma. RaptorXML ya está disponible en cinco idiomas (español, francés, inglés, alemán y japonés) en la carpeta `<CarpetaArchivosPrograma>\Altova\RaptorXMLXBRLServer2018\bin\`.

Para localizar RaptorXML en otros idiomas:

1. Genere un archivo XML con las cadenas de recursos usando el comando exportresourcestrings. Las cadenas de recursos del archivo XML generado estarán en uno de los cinco idiomas compatibles: inglés (`en`), español (`es`), francés (`fr`), alemán (`de`) o japonés (`ja`), dependiendo del argumento utilizado con el comando.
2. Traduzca las cadenas de recursos del XML generado al idioma de destino. Las cadenas de recursos son el contenido de los elementos `<string>` del archivo XML. No traduzca las variables que aparecen entre llaves, como `{option}` o `{product}`, por ejemplo.
3. Póngase en contacto con el equipo de soporte técnico de Altova para generar un archivo DLL localizado de RaptorXML a partir de su archivo XML traducido.
4. Cuando reciba el archivo DLL localizado del equipo de soporte técnico de Altova, guárdelo en la carpeta `<CarpetaArchivosPrograma>\Altova\RaptorXMLXBRLServer2018\bin\`. El DLL tendrá un nombre similar a este `RaptorXMLXBRLServer_ci.dll`. La parte `_ci` del nombre contiene el código del idioma. Por ejemplo, en `RaptorXMLXBRLServer_de.dll`, la parte `de` es el código del idioma alemán (`Deutsch`).
5. Ejecute el comando setdeflang para establecer el archivo DLL localizado como aplicación predeterminada. Use el código de idioma del nombre del archivo DLL como argumento del comando setdeflang.

Nota: Altova RaptorXML+XBRL Server está disponible en cinco idiomas (español, francés, inglés, alemán y japonés) y, por tanto, no es necesario traducirlo a estos idiomas. Para establecer uno de estos idiomas como idioma predeterminado use el comando setdeflang.

3.11.1 exportresourcestrings

El comando **`exportresourcestrings`** genera un archivo XML que contiene las cadenas de recursos de RaptorXML. Este comando toma dos argumentos: (i) el idioma de las cadenas de recursos del archivo XML de salida y (ii) la ruta de acceso y el nombre del archivo XML de salida. Los idiomas de exportación compatibles (y sus códigos) son: inglés (`en`), alemán, (`de`), español (`es`), francés (`fr`) y japonés (`ja`).

Windows **`RaptorXMLXBRL exportresourcestrings`** ***`CódigoIdioma ArchivoXMLSalida`***

Linux **`raptorxmlxbrl exportresourcestrings`** ***`CódigoIdioma ArchivoXMLSalida`***

Mac **`raptorxmlxbrl exportresourcestrings`** ***`CódigoIdioma ArchivoXMLSalida`***

Argumentos

El comando `exportresourcestrings` toma dos argumentos:

`CódigoIdioma`	idioma de las cadenas de recursos del archivo XML de salida exportado. Idiomas permitidos: `en`, `de`, `es`, `fr`, `ja`
`ArchivoXMLSalida`	ubicación y nombre del archivo XML de salida exportado.

Ejemplo

Este comando crea un archivo llamado `Strings.xml` en `c:\` que contiene todas las cadenas de recursos de la aplicación RaptorXML traducida al alemán.

```
raptorxmlxbrl exportresourcestrings de c:\Strings.xml
```

▼ Uso de mayúsculas/minúsculas y de barras diagonales en la línea de comandos

`RaptorXMLXBRL` *en Windows*
`raptorxmlxbrl` *en Unix (Linux, Mac)*

* Las minúsculas (`raptorxmlxbrl`) funcionan en todas las plataformas (Windows, Linux y Mac), mientras que la mezcla de mayúsculas y minúsculas (`RaptorXMLXBRL`) sólo funciona en Windows y Mac.
* En Linux y Mac utilice barras diagonales. En Windows utilice barras diagonales inversas.

3.11.2 setdeflang

El comando **`setdeflang`** (formato abreviado **`sdl`**) establece el idioma predeterminado de RaptorXML. Su argumento obligatorio es `CódigoIdioma`.

Windows	**`RaptorXMLXBRL setdeflang | sdl`** ***`CódigoIdioma`***
Linux	**`raptorxmlxbrl setdeflang | sdl`** ***`CódigoIdioma`***
Mac	**`raptorxmlxbrl setdeflang | sdl`** ***`CódigoIdioma`***

Ejemplo

Este comando define el idioma español como idioma predeterminado de los mensajes de la aplicación.

```
raptorxmlxbrl setdeflang es
```

▼ Uso de mayúsculas/minúsculas y de barras diagonales en la línea de comandos

`RaptorXMLXBRL` *en Windows*
`raptorxmlxbrl` *en Unix (Linux, Mac)*

* Las minúsculas (`raptorxmlxbrl`) funcionan en todas las plataformas (Windows, Linux y Mac), mientras que la mezcla de mayúsculas y minúsculas (`RaptorXMLXBRL`) sólo funciona en Windows y Mac.
* En Linux y Mac utilice barras diagonales. En Windows utilice barras diagonales inversas.

Idiomas compatibles

Los idiomas disponibles son:

EN	inglés
DE	alemán
ES	español
FR	francés
JA	japonés

 © 2018 Altova GmbH

3.12 Opciones

En esta sección describimos todas las opciones de la ILC, organizadas por funciones. Para ver qué opción debe usar con cada comando, consulte la descripción del comando correspondiente.

- Catálogos, recursos globales y archivos ZIP
- Mensajes, errores y ayuda
- Procesamiento
- XBRL
- XML
- XSD
- XQuery
- XSLT

3.12.1 Catálogos, recursos globales, archivos ZIP

catalog

--catalog = *ARCHIVO*
Especifica la ruta de acceso absoluta a un archivo de catálogo que no está en el archivo de catálogo raíz instalado. El valor predeterminado es la ruta de acceso absoluta del archivo de catálogo raíz instalado (`<carpeta-instalación>\Altova\RaptorXMLXBRLServer2018\etc\RootCatalog.xml`). Consulte el apartado Catálogos XML para obtener más información.

user-catalog

--user-catalog = *ARCHIVO*
Especifica la ruta de acceso absoluta a un catálogo XML que debe utilizarse junto con el catálogo raíz. Consulte el apartado Catálogos XML para obtener más información.

enable-globalresources

--enable-globalresources = `true|false`
Habilita la función de recursos globales. Valor predeterminado: `false`.
Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

globalresourceconfig [gc]

--gc | --globalresourceconfig = *VALOR*
Especifica la configuración activa del recurso global (y habilita los recursos globales).

globalresourcefile [gr]

--gr | --globalresourcefile = *ARCHIVO*
Especifica el archivo de recursos globales (y habilita los recursos globales).

recurse

--recurse = `true|false`
Esta opción se utiliza para seleccionar ficheros dentro de subdirectorios (incluso en archivos

ZIP). Si el valor es `true`, el argumento *`ArchivoEntrada`* del comando seleccionará el fichero seleccionado también en los subdirectorios. Por ejemplo: `test.zip|zip\test.xml` seleccionará los ficheros llamados `test.xml` en todos los subdirectorios de la carpeta ZIP. Si quiere puede usar los caracteres comodín `*` y `?`. Por ejemplo: `*.xml` seleccionaría todos los ficheros de la carpeta ZIP que tengan la extensión `.xml`. Valor predeterminado: `false`
Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

3.12.2 Mensajes, errores, ayuda, tiempo de espera y versión

- error-format

  `--error-format =` `text|shortxml|longxml`
  Especifica el formato de la salida de error. Los valores posibles son formatos de texto, XML y XML detallado (`longxml`). Valor predeterminado: `text`.

- error-limit

  `--error-limit =` *`N`*
  Especifica el límite de errores con un valor comprendido entre `1` y `999`. El valor predeterminado es `100`. Cuando se alcanza el límite de error, se detiene la validación. Esta opción es muy práctica a la hora de limitar el uso del procesador durante la validación o transformación.

- help

  `--help`
  Muestra el texto de ayuda para el comando. Por ejemplo `valany --h`. (Otra opción es usar el comando `help` con un argumento. Por ejemplo: `help valany`).

- log-output

  `--log-output =` *`ARCHIVO`*
  Escribe el registro de salida en la URL de archivo indicada. Compruebe que la ILC tiene permiso de escritura en la ubicación de destino.

- network-timeout

  `--network-timeout =` *`VALOR`*
  Indica el tiempo de espera en segundos para operaciones remotas de entrada y salida. Valor predeterminado: `40`.

- verbose

  `--verbose =` `true|false`
  Si el valor es `true`, se genera información adicional durante la validación. Valor predeterminado es `false`.
  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- verbose-output

 © 2018 Altova GmbH

--verbose-output = *ARCHIVO*
Escribe el resultado detallado en el *ARCHIVO* indicado.

- version

--version
Muestra el número de versión de RaptorXML+XBRL Server. Si se utiliza con un comando, escriba la opción `--version` antes del comando.

- warning-limit

--warning-limit = *VALUE*
Especifica el límite de advertencia en el rango `1-65535`. El procesamiento continua si se alcanza el límite pero no se registrarán más advertencias. El valor por defecto es `100`.

3.12.3 Procesamiento

- listfile

--listfile = `true|false`
Si el valor es `true`, el argumento *ArchivoEntrada* del comando se entiende como un archivo de texto que contiene un nombre de archivo por línea. Otra opción es enumerar los archivos en la ILC, separados por un espacio. No obstante, recuerde que las ILC tienen un límite de caracteres. Además, no olvide que la opción `--listfile` solamente afecta a los argumentos y no a las opciones.
Valor predeterminado: `false`.
Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- parallel-assessment [pa]

--pa | --parallel-assessment = `true|false`
Si el valor es `true`, la evaluación de la validez de esquemas se realiza en paralelo. Esto significa que si hay más de 128 elementos en cualquiera de los niveles, estos elementos se procesan en paralelo utilizando varios subprocesos. Por tanto, los archivos XML de gran tamaño se pueden procesar más rápido si se habilita esta opción. La evaluación en paralelo se lleva cabo nivel por nivel, pero puede ocurrir en varios niveles de un mismo conjunto de información. Recuerde que la evaluación en paralelo no funciona en modo de transmisión por secuencias. Por este motivo la opción `--streaming` se pasa por alto si el valor de la opción `--parallel-assessment` es `true`. Además, se usa más memoria cuando se utiliza la opción `--parallel-assessment`. El valor predeterminado de esta opción es false y su forma abreviada es `--pa`.
Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- script

--script = *ARCHIVO*
Una vez finalizada la validación, ejecuta el script Python. Para indicar más de un script basta con agregar la opción varias veces.

- script-api-version

  `--api, --script-api-version =` `1|2|2.1|2.2|2.3|2.4|2.4.1|2.5`
  Especifica la versión API Python que ha de usarse para el script. El valor por defecto es la última versión, actualmente **`2.45`**. En vez de de los valores `1` y `2`, también puede usar losa valores 1.0 y 2.0 respectivamente.

- script-param

  `--script-param =` *`CLAVE:VALOR`*
  Parámetros definidos por el usuario a los que se puede acceder durante la ejecución de scripts Python.

- streaming

  `--streaming =` `true|false`
  Habilita la transmisión por secuencias. En el modo de transmisión por secuencias, el almacenamiento de datos en memoria se reduce al mínimo y el procesamiento es más rápido. El inconveniente es que puede que no esté disponible cierta información que podría necesitar más adelante, como el modelo de datos del documento XML, por ejemplo. Si quiere evitar esto, debería deshabilitar el modo de transmisión por secuencias (dándole el valor `false` a la opción `--streaming`). Cuando use la opción `--script` con el comando `valxml-withxsd`, aconsejamos deshabilitar la transmisión por secuencias. Recuerde que la opción `--streaming` se ignora, si el valor de `--parallel-assessment` es `true`.
  Valor predeterminado: `true`.
  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- xml-validation-error-as-warning

  `--xml-validation-error-as-warning =` `true|false`
  Si es `true`, tratar los errores de validación como advertencias. Si los errores se tratan como advertencias, el procesamiento adicional como la transformación XSLT, proseguirá independientemente de los errores. Por defecto es `false`.

3.12.4 XBRL

- Opciones de validación y procesamiento XBRL
  - dimensions

    `--dimensions =` `true|false`
    Habilita las extensiones XBRL Dimension 1.0. Valor predeterminado: `true`.
    Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

  - evaluate-referenced-parameters-only

    `--evaluate-referenced-parameters-only =` `true|false`
    Si tiene valor `false`, obliga a evaluar todos los parámetros incluso si no se les hace referencia en ninguna fórmula, aserción o tabla. Valor predeterminado: `true`.

 © 2018 Altova GmbH

- extensible-enumerations

  `--extensible-enumerations =` `true|false`
  Si es `true`, habilita la extensión Enumeraciones extensibles XBRL 1.0. Por defecto es: `true`.

- inconsistencies-limit

  `--inconsistencies-limit =` *`VALUE`*
  Especifica el límite de las inconsistencias XBRL con valores que se encuentran en el rango `1-65535`. El procesamiento continua si se alcanza el límite, pero no se registran el resto de las incoherencias. El valor por defecto es `100`.

- listfile

  `--listfile =` `true|false`
  Si el valor es `true`, el argumento *`ArchivoEntrada`* del comando se entiende como un archivo de texto que contiene un nombre de archivo por línea. Otra opción es enumerar los archivos en la ILC, separados por un espacio. No obstante, recuerde que las ILC tienen un límite de caracteres. Además, no olvide que la opción `--listfile` solamente afecta a los argumentos y no a las opciones.
  Valor predeterminado: `false`.
  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- parallel-assessment [pa]

  `--pa | --parallel-assessment =` `true|false`
  Si el valor es `true`, la evaluación de la validez de esquemas se realiza en paralelo. Esto significa que si hay más de 128 elementos en cualquiera de los niveles, estos elementos se procesan en paralelo utilizando varios subprocesos. Por tanto, los archivos XML de gran tamaño se pueden procesar más rápido si se habilita esta opción. La evaluación en paralelo se lleva cabo nivel por nivel, pero puede ocurrir en varios niveles de un mismo conjunto de información. Recuerde que la evaluación en paralelo no funciona en modo de transmisión por secuencias. Por este motivo la opción `--streaming` se pasa por alto si el valor de la opción `--parallel-assessment` es `true`. Además, se usa más memoria cuando se utiliza la opción `--parallel-assessment`. El valor predeterminado de esta opción es false y su forma abreviada es `--pa`.
  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- preload-xbrl-schemas

  `--preload-xbrl-schemas =` `true|false`
  Carga previa de los esquemas de la especificación XBRL 2.1. Valor predeterminado: `true`.
  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- recurse

  `--recurse =` `true|false`
  Esta opción se utiliza para seleccionar ficheros dentro de subdirectorios (incluso en

archivos ZIP). Si el valor es `true`, el argumento *`ArchivoEntrada`* del comando seleccionará el fichero seleccionado también en los subdirectorios. Por ejemplo: `test.zip|zip\test.xml` seleccionará los ficheros llamados `test.xml` en todos los subdirectorios de la carpeta ZIP. Si quiere puede usar los caracteres comodín `*` y `?`. Por ejemplo: `*.xml` seleccionaría todos los ficheros de la carpeta ZIP que tengan la extensión `.xml`. Valor predeterminado: `false`

Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

schema-imports

`--schema-imports =` `load-by-schemalocation | load-preferring-schemalocation | load-by-namespace | load-combining-both | license-namespace-only`

Esta opción indica el comportamiento de los elementos `xs:import`. Cada uno de estos elementos tiene un atributo opcional `namespace` y un atributo opcional `schemaLocation`: `<import namespace="unEspacioNombres" schemaLocation="unaURL">`. La opción indica si se debe cargar un documento de esquema o solo autorizar a un espacio de nombres. Si la opción indica que se debe cargar un documento de esquema, entonces indica también qué información debe utilizarse para encontrar el documento de esquema. Valor predeterminado: `load-preferring-schemalocation`.

- `load-by-schemalocation`: el valor del atributo `schemaLocation` se utiliza para buscar el esquema, teniendo en cuenta las asignaciones de catálogo. Si está presente el atributo `namespace`, se importa el espacio de nombres (con licencia).
- `load-preferring-schemalocation`: si está presente, se utiliza el atributo `schemaLocation` teniendo en cuenta las asignaciones de catálogo. Si no está presente el atributo `schemaLocation`, entonces se usa el valor del atributo `namespace` a través de las asignaciones de catálogo. Este es el **valor predeterminado**.
- `load-by-namespace`: el valor del atributo `namespace` se utiliza para buscar el esquema por medio de una asignación de catálogo.
- `load-combining-both`: si el atributo `namespace` o `schemaLocation` tiene una asignación de catálogo, entonces se usa la asignación. Si ambos atributos tienen asignaciones de catálogo, entonces es el valor de la opción `--schema-mapping` (opción XBRL y opción XML/XSD) decide qué asignación se utiliza. Si no hay ninguna asignación de catálogo, entonces se usa el atributo `schemaLocation`.
- `license-namespace-only`: se importa el espacio de nombres. No se importa el documento de esquema.

schema-mapping

`--schema-mapping =` `prefer-schemalocation | prefer-namespace`

Si se usa la ubicación y el espacio de nombres para buscar el documento de esquema, esta opción indica cuál de ellos debe ser la opción preferida durante la búsqueda en el catálogo.

Si la opción `--schemalocation-hints` o la opción `--schema-imports` tiene el valor `load-combining-both` y si las partes de espacio de nombres y URL pertinentes tienen asignaciones de catálogo, entonces el valor de la opción especifica cuál de las dos asignaciones se utiliza (la asignación del espacio de nombres o de la URL: el valor `prefer-schemalocation` se refiere a la asignación de la URL). Valor predeterminado: `prefer-schemalocation`.

 © 2018 Altova GmbH

schemalocation-hints

--schemalocation-hints = `load-by-schemalocation | load-by-namespace | load-combining-both | ignore`
Determina el comportamiento predeterminado de los atributos `xsi:schemaLocation` y `xsi:noNamespaceSchemaLocation`. Indica si se debe cargar un documento de esquema y, si así es, indica qué información debe utilizarse para encontrarlo. Valor predeterminado: `load-by-schemalocation`.

- **Valor predeterminado**: `load-by-schemalocation`. Este valor toma la URL de la ubicación del esquema de los atributos `xsi:schemaLocation` y `xsi:noNamespaceSchemaLocation` de los documentos de instancia XML o XBRL.
- El valor `load-by-namespace` toma la parte de espacio de nombres del atributo `xsi:schemaLocation` y una cadena vacía en el caso del atributo `xsi:noNamespaceSchemaLocation` y encuentra el esquema por medio de una asignación de catálogo.
- Si usa el valor `load-combining-both` y el espacio de nombres o la URL tienen una asignación de catálogo, se usa dicha asignación. Si ambos tienen asignaciones de catálogo, el valor de la opción `schema-mapping` (opción XBRL y opción XML/XSD) decide qué asignación se utiliza. Si ni el espacio de nombres ni la URL tiene una asignación de catálogo, se usa la URL.
- El valor `ignore` ignora los atributos `xsi:schemaLocation` y `xsi:noNamespaceSchemaLocation`.

script

--script = *ARCHIVO*
Una vez finalizada la validación, ejecuta el script Python. Para indicar más de un script basta con agregar la opción varias veces.

script-api-version

--api, --script-api-version = `1|2|2.1|2.2|2.3|2.4|2.4.1|2.5`
Especifica la versión API Python que ha de usarse para el script. El valor por defecto es la última versión, actualmente **`2.45`**. En vez de de los valores `1` y `2`, también puede usar losa valores 1.0 y 2.0 respectivamente.

script-param

--script-param = *CLAVE:VALOR*
Parámetros definidos por el usuario a los que se puede acceder durante la ejecución de scripts Python.

treat-inconsistencies-as-errors

--treat-inconsistencies-as-errors = `true|false`
La validación XBRL falla si el archivo contiene incoherencias según lo definido por la especificación XBRL 2.1. El valor predeterminado es `false`.
Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- utr

  --utr = true|false
  Si es true, habilita la extensión Registro de unidad XBRL 1.0. Por defecto es: false.

- validate-dts-only

  --validate-dts-only = true|false
  La DTS se detecta iniciando desde el documento de instancia XBRL. Todos los esquemas de taxonomía y bases de enlace referenciados se detectan y validan. Los demás documentos de instancia XBRL se ignoran. Valor predeterminado: false.
  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en true.

- xinclude

  --xinclude = true|false
  Habilita la compatibilidad con inclusiones XML (XInclude). Si el valor es false, los elementos XInclude include se ignoran. Valor predeterminado: false.
  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en true.

- Opciones para fórmulas y aserciones XBRL
  - assertion-severity

    --assertion-severity = true|false
    Activa la extensión Assertion Severity 1.0. El valor por defecto es true.
    Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en true.

  - assertions-output

    --assertions-output = *ARCHIVO*
    Escribe el resultado de la evaluación de la aserción en el *ARCHIVO* indicado. Si se define, especifica automáticamente --formula-execution=true.

  - assertions-output-format

    --assertions-output-format = json|xml
    Especifica el formato de salida de la evaluación de la aserción. Valor predeterminado: json.

  - formula

    --formula = true|false
    Habilita la extensión XBRL Formula 1.0. Valor predeterminado: true.
    Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en true.

  - formula-assertion-set [[OBSOLETA]]

    --formula-assertion-set = *VALOR*
    Limita la ejecución de fórmulas al conjunto de aserciones dado. Añada la opción varias veces para especificar más de un grupo de aserciones. Su forma abreviada es --as. El

VALOR es el valor del atributo `@id` o un URI con un fragmento XPointer que identifica el recurso. También puede usar los valores especiales `##none` y `##all`.

- formula-execution

  `--formula-execution =` `true|false`
  Habilita la evaluación de fórmulas XBRL. Valor predeterminado: `true`. Si es `true`, especifica automáticamente `--formula=true`.
  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- formula-output

  `--formula-output =` *ARCHIVO*
  Escribe el resultado de la evaluación de la fórmula en el *ARCHIVO* indicado. Si se define, especifica automáticamente `--formula-execution=true`.

- formula-parameters

  `--formula-parameters =` *MATRIZ-JSON*
  Especifica parámetros para la evaluación de fórmulas XBRL como matriz de mapas JSON en la interfaz de la línea de comandos directamente. Consulte el apartado *Parámetros de XBRL Formula*.

- formula-parameters-file

  `--formula-parameters-file =` *ARCHIVO*
  Especifica un *ARCHIVO* que contiene los parámetros para la evaluación de fórmulas XBRL. Puede tratarse de un archivo XML o de un archivo JSON. Consulte el apartado *Parámetros de XBRL Formula* para obtener más información.

- message-lang

  `--message-lang =` *VALUE*
  Especifica el idioma que se va a usar cuando se muestren los mensajes de validación. Por defecto es: `en`.
  El resto de valores aceptados son `de`, `es`, `fr`, `ja`, para alemán, español, francés y japonés, respectivamente.

- message-role

  `--message-role =` *VALUE*
  Especifica el papel del mensaje predefinido que se usa cuando se muestran los mensajes de validación. Por defecto es: `http://www.xbrl.org/2010/role/message`

- preload-formula-schemas

  `--preload-formula-schemas =` `true|false`
  Carga previa de los esquemas de la especificación XBRL Formula 1.0. Valor predeterminado: `false`.
  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- process-assertion [a]

  `--a | --process-assertion =` *VALOR*

Limita la ejecución de fórmulas a la aserción dada. Añada la opción varias veces para especificar más de una aserción. Su forma abreviada es `--a`. El *VALOR* es el valor del atributo `@id` o un URI con un fragmento XPointer que identifica el recurso. También se pueden usar los valores especiales `##none` y `##all`.

- process-assertion-set [as]

  `--as | --process-assertion-set =` *VALOR*
  Limita la ejecución de fórmulas al conjunto de aserciones dado. Añada la opción varias veces para especificar más de un conjunto de aserciones. Su forma abreviada es `--as`. El ***VALOR*** es el valor del atributo `@id` o un URI con un fragmento XPointer que identifica el recurso. También puede usar los valores especiales `##none` y `##all`.

- process-formula [f]

  `--f | --process-formula =` *VALOR*
  Limita la ejecución de fórmulas a la fórmula dada. Añada la opción varias veces para especificar más de una fórmula. Su forma abreviada es `--as`. El ***VALOR*** es el valor del atributo `@id` o un URI con un fragmento XPointer que identifica el recurso. También puede usar los valores especiales `##none` y `##all`.

- report-unsatisfied-assertion-evaluations

  `--report-unsatisfied-assertion-evaluations =` `true|false`
  Informa sobre la evaluación de una aserción insatisfactoria como errores o advertencias que dependen de nivel de severidad de la asercación. El valor predeterminado es `false`.
  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- variableset-execution-timeout

  `--variableset-execution-timeout =` *VALOR*
  Se aplica cuando se ejecutan fórmulas (**`--formula-execution=true`**). Especifica el tiempo de espera máximo permitido para ejecutar un conjunto de variables único (una fórmula o un valor o una aserción de existencia o de coherencia). El tiempo de espera se indica en minutos y debe ser un número positivo. El valor predeterminado es 30 minutos. Si un conjunto de variables determinado no termina de ejecutarse antes de cumplirse el tiempo de espera, la ejecución se anula. Aparece un mensaje de error (que se introduce en el registro detallado). No obstante, debe tener en cuenta que la comprobación de tiempo de espera solamente se realiza después de evaluarse cada conjunto de variables (y no durante la ejecución de las expresiones XPath). Por tanto, si una expresión XPath tarda mucho en ejecutarse, puede que se cumpla el tiempo de espera. La ejecución del conjunto de variables se anula solamente cuando se ejecute la evaluación completa del conjunto de variables.

- Opciones para tablas XBRL
  - concept-label-linkrole

    `--concept-label-linkrole =` *VALOR*
    Especifica la función de vínculo preferida a la hora de representar etiquetas de concepto.

- concept-label-role

  --cconcept-label-role = *VALOR*
  Especifica la función de etiqueta preferida a la hora de representar etiquetas de concepto. La función predeterminada es `http://www.xbrl.org/2003/role/label`.

- evaluate-referenced-parameters-only

  --evaluate-referenced-parameters-only = `true|false`
  Si tiene valor `false`, obliga a evaluar todos los parámetros incluso si no se les hace referencia en ninguna fórmula, aserción o tabla. Valor predeterminado: `true`.

- generic-label-linkrole

  --generic-label-linkrole = *VALOR*
  Especifica la función de vínculo extendido preferida a la hora de representar etiquetas genéricas.

- generic-label-role

  --generic-label-role = *VALOR*
  Especifica la función de etiqueta preferida a la hora de representar etiquetas genéricas. La función predeterminada es `http://www.xbrl.org/2003/role/label`

- label-lang

  --label-lang = *VALOR*
  Especifica el idioma preferido a la hora de representar etiquetas. Valor predeterminado: `en`

- preload-table-schemas

  --preload-table-schemas = `true|false`
  Carga previa de los esquemas de la especificación XBRL Table 1.0. Valor predeterminado: `false`.
  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- process-table [t]

  --t | --process-table = *VALOR*
  Limita la ejecución de fórmulas a la taba dada. Añada la opción varias veces para especificar más de una tabla. Su forma abreviada es `--t`. El ***VALOR*** es el valor del atributo `@id` o un URI con un fragmento XPointer que identifica el recurso. También puede usar los valores especiales `##none` y `##all`.

- table

  --table = `true|false`
  Habilita la extensión XBRL Table 1.0. Si es `true`, especifica automáticamente `--formula=true` y `--dimensions=true`. Valor predeterminado: `true`.
  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- table-elimination

--table-elimination = true|false
Habilita la eliminación de columnas/filas de tabla vacías en el formato de salida HTML. Valor predeterminado: true.
Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en true.

- table-execution

--table-execution = true|false
Habilita la evaluación de tablas XBRL. Valor predeterminado: false. Si se especifica la opción --table-output, a esta opción se asigna true. Si tiene el valor true, se especifica automáticamente --table=true.
Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en true.

- table-linkbase-namespace

--table-linkbase-namespace =
##detect |
http://xbrl.org/PWD/2013-05-17/table |
http://xbrl.org/PWD/2013-08-28/table |
http://xbrl.org/CR/2013-11-13/table |
http://xbrl.org/PR/2013-12-18/table |
http://xbrl.org/2014/table

Habilita la carga de bases de enlaces de tabla escritas con una especificación borrador previa. La validación, la resolución y el diseño de bases de enlaces de tabla se realiza, sin embargo, siguiendo la recomendación Table Linkbase 1.0 del 18 de marzo de 2014. Para habilitar la detección automática, utilice ##detect.

- table-output

--table-output = *ARCHIVO*
Escribe la tabla de salida en el ***ARCHIVO*** indicado. Si se define, especifica automáticamente --table-execution=true.

- table-output-format

--table-output-format = xml|html
Especifica el formato de la tabla de salida. Valor predeterminado: xml.

3.12.5 XML

- assessment-mode

--assessment-mode = lax|strict
Especifica el modo de evaluación de la validez del esquema, según se define en las especificaciones XSD. El documento XML de instancia se validará en función del modo especificado en esta opción. Valor predeterminado: strict.

- dtd

  --dtd = *ARCHIVO*
  Especifica el documento DTD externo que debe utilizarse para la validación. Si en el documento XML hay una referencia a una DTD externa, esta opción de la ILC reemplaza a la referencia externa.

- load-xml-with-psvi

  --load-xml-with-psvi = `true|false`
  Habilita la validación de archivos XML de entrada y genera información posterior a la validación. Valor predeterminado: `true`.

- namespaces

  --namespaces = `true|false`
  Habilita el procesamiento preparado para espacios de nombres. Esta opción es muy útil si quiere buscar en la instancia XML errores resultantes de espacios de nombres erróneos. Valor predeterminado: `false`.
  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- xinclude

  --xinclude = `true|false`
  Habilita la compatibilidad con inclusiones XML (XInclude). Si el valor es `false`, los elementos XInclude `include` se ignoran. Valor predeterminado: `false`.
  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- xml-mode

  --xml-mode = `wf|id|valid`
  Especifica el modo de procesamiento XML que debe utilizarse para el documento de instancia XML: `wf`=comprobación de formato; `id`=comprobación de formato con ID/IDREF; `valid`=validación. Valor predeterminado: `wf`. Recuerde que el valor `valid` exige que cada documento de instancia que se cargue durante el procesamiento haga referencia a una DTD. Si no existe ninguna DTD, se generará un error.

- xml-validation-error-as-warning

  --xml-validation-error-as-warning = `true|false`
  Si es `true`, tratar los errores de validación como advertencias. Si los errores se tratan como advertencias, el procesamiento adicional como la transformación XSLT, proseguirá independientemente de los errores. Por defecto es `false`.

- xsd

  --xsd = *ARCHIVO*
  Especifica qué esquemas XML deben utilizarse para la validación de documentos XML. Si quiere especificar más de un esquema, añada la opción varias veces.

3.12.6 XSD

- assessment-mode

  --assessment-mode = lax|strict
  Especifica el modo de evaluación de la validez del esquema, según se define en las especificaciones XSD. El documento XML de instancia se validará en función del modo especificado en esta opción. Valor predeterminado: strict.

- namespaces

  --namespaces = true|false
  Habilita el procesamiento preparado para espacios de nombres. Esta opción es muy útil si quiere buscar en la instancia XML errores resultantes de espacios de nombres erróneos. Valor predeterminado: false.
  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en true.

- schema-imports

  --schema-imports = load-by-schemalocation | load-preferring-schemalocation | load-by-namespace | load-combining-both | license-namespace-only
  Esta opción indica el comportamiento de los elementos xs:import. Cada uno de estos elementos tiene un atributo opcional namespace y un atributo opcional schemaLocation: <import namespace="unEspacioNombres" schemaLocation="unaURL">. La opción indica si se debe cargar un documento de esquema o solo autorizar a un espacio de nombres. Si la opción indica que se debe cargar un documento de esquema, entonces indica también qué información debe utilizarse para encontrar el documento de esquema. Valor predeterminado: load-preferring-schemalocation.

  - load-by-schemalocation: el valor del atributo schemaLocation se utiliza para buscar el esquema, teniendo en cuenta las asignaciones de catálogo. Si está presente el atributo namespace, se importa el espacio de nombres (con licencia).
  - load-preferring-schemalocation: si está presente, se utiliza el atributo schemaLocation teniendo en cuenta las asignaciones de catálogo. Si no está presente el atributo schemaLocation, entonces se usa el valor del atributo namespace a través de las asignaciones de catálogo. Este es el **valor predeterminado**.
  - load-by-namespace: el valor del atributo namespace se utiliza para buscar el esquema por medio de una asignación de catálogo.
  - load-combining-both: si el atributo namespace o schemaLocation tiene una asignación de catálogo, entonces se usa la asignación. Si ambos atributos tienen asignaciones de catálogo, entonces es el valor de la opción --schema-mapping (opción XBRL y opción XML/XSD) decide qué asignación se utiliza. Si no hay ninguna asignación de catálogo, entonces se usa el atributo schemaLocation.
  - license-namespace-only: se importa el espacio de nombres. No se importa el documento de esquema.

- schemalocation-hints

  --schemalocation-hints = load-by-schemalocation | load-by-namespace | load-combining-both | ignore
  Determina el comportamiento predeterminado de los atributos xsi:schemaLocation y xsi:noNamespaceSchemaLocation. Indica si se debe cargar un documento de esquema y,

 © 2018 Altova GmbH

si así es, indica qué información debe utilizarse para encontrarlo. Valor predeterminado: `load-by-schemalocation`.

- **Valor predeterminado**: `load-by-schemalocation`. Este valor toma la URL de la ubicación del esquema de los atributos `xsi:schemaLocation` y `xsi:noNamespaceSchemaLocation` de los documentos de instancia XML o XBRL.

- El valor `load-by-namespace` toma la parte de espacio de nombres del atributo `xsi:schemaLocation` y una cadena vacía en el caso del atributo `xsi:noNamespaceSchemaLocation` y encuentra el esquema por medio de una asignación de catálogo.

- Si usa el valor `load-combining-both` y el espacio de nombres o la URL tienen una asignación de catálogo, se usa dicha asignación. Si ambos tienen asignaciones de catálogo, el valor de la opción `schema-mapping` (opción XBRL y opción XML/XSD) decide qué asignación se utiliza. Si ni el espacio de nombres ni la URL tiene una asignación de catálogo, se usa la URL.

- El valor `ignore` ignora los atributos `xsi:schemaLocation` y `xsi:noNamespaceSchemaLocation`.

schema-mapping

`--schema-mapping =` `prefer-schemalocation | prefer-namespace`

Si se usa la ubicación y el espacio de nombres para buscar el documento de esquema, esta opción indica cuál de ellos debe ser la opción preferida durante la búsqueda en el catálogo. Si la opción `--schemalocation-hints` o la opción `--schema-imports` tiene el valor `load-combining-both` y si las partes de espacio de nombres y URL pertinentes tienen asignaciones de catálogo, entonces el valor de la opción especifica cuál de las dos asignaciones se utiliza (la asignación del espacio de nombres o de la URL: el valor `prefer-schemalocation` se refiere a la asignación de la URL). Valor predeterminado: `prefer-schemalocation`.

xsd-version

`--xsd-version =` `1.0|1.1|detect`

Especifica qué versión de la especificación Schema Definition Language (XSD) del W3C se debe usar. Valor predeterminado: `1.0`.

Esta opción también puede ser útil si quiere ver en qué aspectos no es compatible un esquema 1.0 con la especificación 1.1. El valor `detect` es una característica de Altova. Permite detectar la versión del esquema XML (`1.0` o `1.1`) leyendo el valor del atributo `vc:minVersion` del elemento `<xs:schema>` del documento. Si el valor del atributo `@vc:minVersion` es `1.1`, se entiende que la versión del esquema es `1.1`. Si el atributo tiene otro valor que no sea `1.1` (o si no está presente el atributo `@vc:minVersion`), se entiende que la versión del esquema es `1.0`.

3.12.7 XQuery

- indent-characters

  --indent-characters = *VALOR*
  Especifica la cadena de caracteres que debe usarse como sangría.

- input

  --input = *ARCHIVO*
  La URL del archivo XML que se debe transformar.

- keep-formatting

  --keep-formatting = `true|false`
  Conserva en la medida de lo posible el formato del documento de destino. Valor predeterminado: `true`.

- omit-xml-declaration

  --omit-xml-declaration = `true|false`
  Opción de serialización que especifica si la declaración XML se omite en el resultado o no. Si el valor es `true`, el documento de salida no tendrá una declaración XML. Si el valor es `false`, se incluye una declaración XML en el documento de salida. Valor predeterminado: `false`.
  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- output, xsltoutput

  output = *ARCHIVO,* **xsltoutput =** *ARCHIVO*
  La URL del archivo de salida principal. Por ejemplo, en caso de tener varios archivos HTML de salida, el archivo de salida principal será la ubicación del archivo HTML del punto de entrada. Los demás archivos de salida (como archivos de imagen generados) se indican con `xslt-additional-output-files`. Si no se especifica la opción `--output` ni la opción `--xsloutput`, se genera un resultado estándar.

- output-encoding

  --output-encoding = *VALOR*
  El valor del atributo `encoding` del documento de salida. Son valores válidos todos los nombres del registro de juego de caracteres IANA. Valor predeterminado: `UTF-8`.

- output-indent

  --output-indent = `true|false`
  Si el valor es `true`, la sangría del documento de salida seguirá su estructura jerárquica. Si el valor es `false`, el documento de salida no tendrá sangría jerárquica. Valor predeterminado: `false`.
  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- output-method

  --output-method = `xml|html|xhtml|text`

 © 2018 Altova GmbH

Especifica el formato de salida. Valor predeterminado: `xml`.

- param [p]

`--p | --param =` *`CLAVE:VALOR`*

XQuery

Especifica el valor de un parámetro externo. En el documento XQuery los parámetros externos se declaran con la declaración `declare variable` seguida de un nombre de variable y después la palabra clave `external` seguida del punto y coma final. Por ejemplo: `declare variable $foo as xs:string external;`

Al usar la palabra clave `external`, `$foo` se convierte en parámetro externo y su valor se pasa en tiempo de ejecución desde una fuente externa. El parámetro externo recibe un valor con el comando de la ILC. Por ejemplo: `--param=foo:'MiNombre'`

En la descripción anterior, *`CLAVE`* es el nombre de parámetro externo y *`VALOR`* es su valor, dado como expresión XPath. Los nombres de parámetro utilizados en la ILC deben declararse en el documento XQuery. Si se pasan valores a varios parámetros externos en la ILC, cada parámetro debe llevar una opción `--param` distinta. Si la expresión XPath contiene espacios, entonces debe estar entre comillas dobles.

XSLT

Especifica un parámetro global de la hoja de estilos. *`CLAVE`* es el nombre del parámetro y *`VALOR`* es una expresión XPath que da un valor al parámetro. Los nombres de parámetro utilizados en la ILC deben declararse en la hoja de estilos. Si usa más de un parámetro, debe usar el modificador `--param` antes de cada parámetro. Si la expresión XPath incluye espacios, entonces debe ir entre comillas dobles, tanto si el espacio está en la expresión propiamente dicha o en un literal de cadena de la expresión. Por ejemplo:

```
raptorxmlxbrl xslt --input=c:\Test.xml --output=c:\Salida.xml --
param=date://node[1]/@att1 --p=title:cadenasinespacios --
param=title:"'cadena con espacios'" --p=amount:456 c:\Test.xslt
```

- updated-xml

`--updated-xml =` `discard|writeback|asmainresult`

Indica qué se debe hacer con el archivo XML actualizado. Las actualizaciones se pueden:

- descartar (los cambios no se escriben en el archivo). Valor `discard`
- escribir en el archivo XML de entrada que se especificó con la opción `--input`. Valor `writeback`
- guardar en el resultado estándar o en la ubicación que se especificó con la opción `--output` (si se usó esta opción)

Valor predeterminado: `discard`.

- xquery-update-version

`--xquery-update-version =` `1|1.0|3|3.0|`

Indica si el procesador XQuery debería usar XQuery Update Facility 1.0 o XQuery Update Facility 3.0. Valor predeterminado: `3`.

- xquery-version

  --xquery-version = `1|1.0|3|3.0|3.1`
  Indica si el procesador XQuery debe usar XQuery 1.0 o 3.0. Valor predeterminado: `3.1`

3.12.8 XSLT

- chartext-disable

  --chartext-disable = `true|false`
  Deshabilita las extensiones de gráficos. Valor predeterminado: `false`.
  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- dotnetext-disable

  --dotnetext-disable = `true|false`
  Deshabilita las extensiones .NET. Valor predeterminado: `false`.
  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- indent-characters

  --indent-characters = *VALOR*
  Especifica la cadena de caracteres que debe usarse como sangría.

- input

  --input = *ARCHIVO*
  La URL del archivo XML que se debe transformar.

- javaext-barcode-location

  --javaext-barcode-location = *ARCHIVO*
  Especifica la ruta de acceso de la carpeta que contiene el archivo de extensión de código de barras `AltovaBarcodeExtension.jar`. La ruta de acceso debe darse en uno de estos formatos:

  - Un URI de archivo (ejemplo: `--javaext-barcode-location="file:///C:/Archivos de programa/Altova/RaptorXMLXBRLServer2018/etc/jar/"`)
  - Una ruta de acceso Windows con caracteres de escape para las barras diagonales inversas (ejemplo: `--javaext-barcode-location="C:\\Archivos de programa\\Altova\\RaptorXMLXBRLServer2018\\etc\\jar\\"`)

- javaext-disable

  --javaext-disable = `true|false`
  Deshabilita las extensiones Java. Valor predeterminado: `false`.
  Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

 © 2018 Altova GmbH

- output, xsltoutput

output = *ARCHIVO,* **xsltoutput =** *ARCHIVO*

La URL del archivo de salida principal. Por ejemplo, en caso de tener varios archivos HTML de salida, el archivo de salida principal será la ubicación del archivo HTML del punto de entrada. Los demás archivos de salida (como archivos de imagen generados) se indican con `xslt-additional-output-files`. Si no se especifica la opción `--output` ni la opción `--xsloutput`, se genera un resultado estándar.

- param [p]

--p | --param = *CLAVE:VALOR*

XQuery

Especifica el valor de un parámetro externo. En el documento XQuery los parámetros externos se declaran con la declaración `declare variable` seguida de un nombre de variable y después la palabra clave `external` seguida del punto y coma final. Por ejemplo: `declare variable $foo as xs:string external;`

Al usar la palabra clave `external`, `$foo` se convierte en parámetro externo y su valor se pasa en tiempo de ejecución desde una fuente externa. El parámetro externo recibe un valor con el comando de la ILC. Por ejemplo: `--param=foo:'MiNombre'`

En la descripción anterior, *CLAVE* es el nombre de parámetro externo y *VALOR* es su valor, dado como expresión XPath. Los nombres de parámetro utilizados en la ILC deben declararse en el documento XQuery. Si se pasan valores a varios parámetros externos en la ILC, cada parámetro debe llevar una opción `--param` distinta. Si la expresión XPath contiene espacios, entonces debe estar entre comillas dobles.

XSLT

Especifica un parámetro global de la hoja de estilos. *CLAVE* es el nombre del parámetro y *VALOR* es una expresión XPath que da un valor al parámetro. Los nombres de parámetro utilizados en la ILC deben declararse en la hoja de estilos. Si usa más de un parámetro, debe usar el modificador `--param` antes de cada parámetro. Si la expresión XPath incluye espacios, entonces debe ir entre comillas dobles, tanto si el espacio está en la expresión propiamente dicha o en un literal de cadena de la expresión. Por ejemplo:

```
raptorxmlxbrl xslt --input=c:\Test.xml --output=c:\Salida.xml --
param=date://node[1]/@att1 --p=title:cadenasinespacios --
param=title:"'cadena con espacios'" --p=amount:456 c:\Test.xslt
```

- streaming

--streaming = `true|false`

Habilita la transmisión por secuencias. En el modo de transmisión por secuencias, el almacenamiento de datos en memoria se reduce al mínimo y el procesamiento es más rápido. El inconveniente es que puede que no esté disponible cierta información que podría necesitar más adelante, como el modelo de datos del documento XML, por ejemplo. Si quiere evitar esto, debería deshabilitar el modo de transmisión por secuencias (dándole el valor `false` a la opción `--streaming`). Cuando use la opción `--script` con el comando `valxml-withxsd`, aconsejamos deshabilitar la transmisión por secuencias. Recuerde que la opción `--streaming` se ignora, si el valor de `--parallel-assessment` es `true`.
Valor predeterminado: `true`.

Nota: si no se especifica un valor para la opción, el valor booleano de la opción se establece en `true`.

- initial-template, template-entry-point

  `--initial-template, --template-entry-point =` *`VALOR`*
  Indica el nombre de una plantilla con nombre de la hoja de estilos XSLT que sirve de punto de entrada de la transformación.

- initial-mode, template-mode

  `--initial-mode, --template-mode =` *`VALOR`*
  Especifica el modo de plantilla que debe usarse para la transformación.

- xslt-version

  `--xslt-version =` `1|1.0|2|2.0|3|3.0|3.1`
  Especifica si el procesador XSLT debe usar XSLT 1.0, XSLT 2.0 o XSLT 3.0. Valor predeterminado: `3`

3.12.9 JSON/Avro

- schema, **`jsonschema`**

  `--schema =` *`ARCHIVO`*, **`--jsonschema =`** *`ARCHIVO`*
  Especifica el documento del esquema JSON que se va usar para la validación de documentos de instancia JSON.

- codec

  `--codec =` `null|deflate`
  Especifica qué códec de comprensión Avro se debe utilizar. Valor predeterminado: `null`.

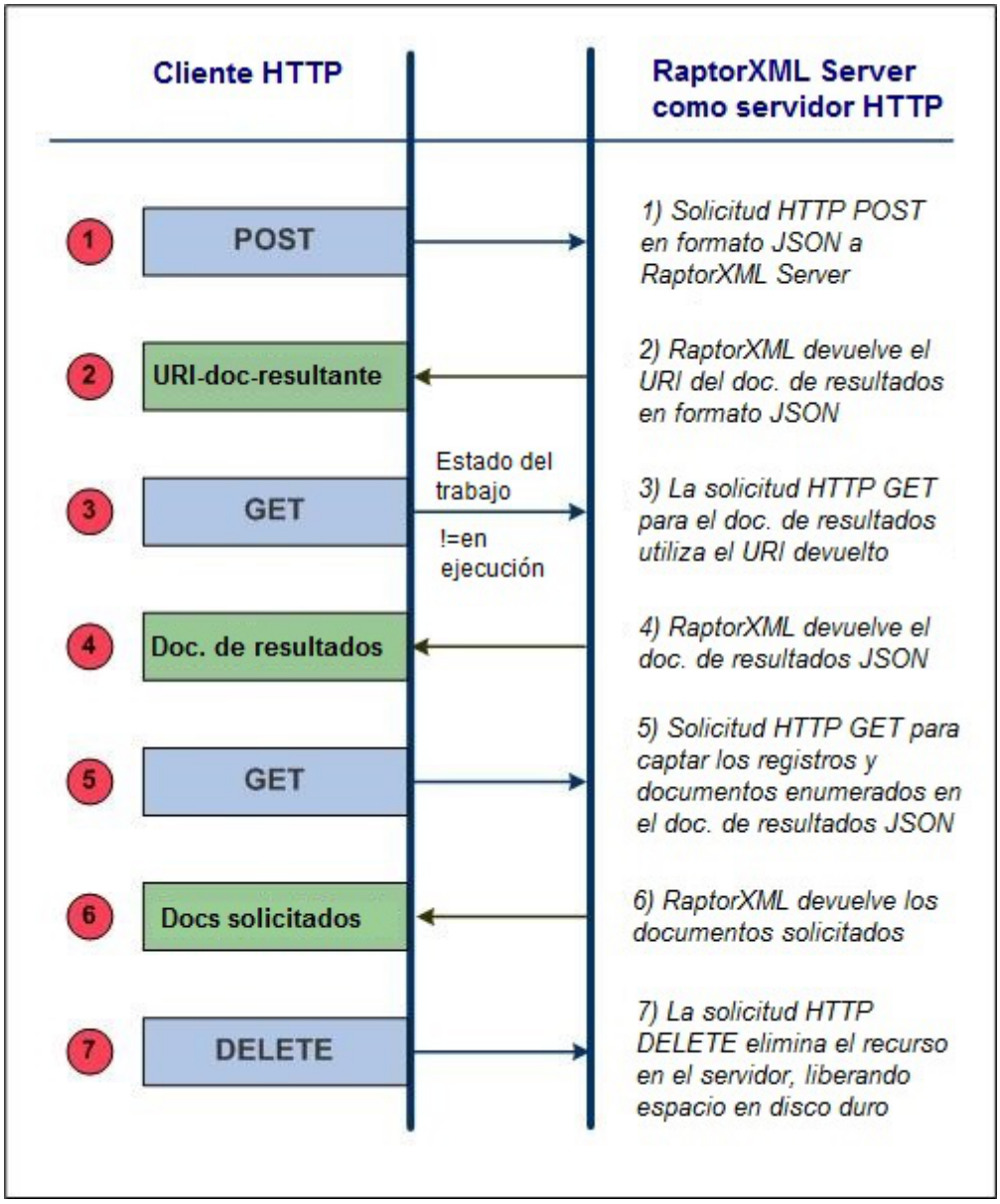
 © 2018 Altova GmbH

Altova RaptorXML+XBRL Server 2018

Interfaz HTTP

4 Interfaz HTTP

RaptorXML+XBRL Server acepta trabajos de validación suministrados por HTTP (o HTTPS). La descripción del trabajo así como los resultados se intercambian en formato JSON. El diagrama que aparece a continuación muestra el flujo de trabajo.



 © 2018 Altova GmbH

Problemas de seguridad relacionados con la interfaz HTTP
La interfaz HTTP permite por defecto escribir documentos de resultados en cualquier ubicación indicada por el cliente (y a la que se pueda acceder con el protocolo HTTP). Por tanto, es importante tener en cuenta este aspecto de seguridad cuando instale y configure RaptorXML+XBRL Server.

Si le preocupa que esto pueda comprometer la seguridad de su sistema o que la interfaz se utilice de forma incorrecta, puede configurar el servidor para que escriba los documentos de resultados en un directorio de salida específico del servidor mismo. Esto se consigue estableciendo el valor `false` para la opción `server.unrestricted-filesystem-access` del archivo de configuración del servidor. Si se limita así el acceso, el cliente puede descargar los documentos de resultados del directorio de salida específico mediante solicitudes `GET`. Otra opción es que el administrador copie/cargue los documentos de resultados del servidor en la ubicación de destino.

Secciones de este tema

Antes de enviar una solicitud cliente, RaptorXML+XBRL Server debe iniciarse y configurarse correctamente. Esto se explica en la sección Preparar el servidor, mientras que la sección Solicitudes cliente describe cómo se envían solicitudes clientes al servidor.

4.1 Preparar el servidor

Siga las instrucciones que aparecen a continuación para preparar correctamente RaptorXML +XBRL Server. Recuerde que antes debe instalarlo y asignarle licencias.

1. Para poder acceder a RaptorXML+XBRL Server por HTTP o HTTPS antes debe iniciarlo como servicio o como aplicación. Hay varias formas de hacerlo, dependiendo del sistema operativo: en Windows, en Linux, en macOS.
2. Utilice la configuración inicial del servidor para probar la conexión con el servidor. (La configuración inicial del servidor es la configuración predeterminada que viene con la instalación). Puede usar una simple solicitud HTTP `GET` como `http://localhost:8087/v1/version` para probar la conexión. (La solicitud también se puede escribir directamente en la barra de dirección del explorador). Si el servicio está en ejecución, debería recibir una respuesta a la solicitud HTTP de prueba.
3. Revise el archivo de configuración del servidor, `server_config.xml`. Si quiere cambiar alguna opción de configuración, edite el archivo y guarde los cambios. La opción HTTPS está deshabilitada por defecto y debe habilitarse en el archivo de configuración.
4. Si editó el archivo de configuración del servidor, reinicie RaptorXML+XBRL Server como servicio para que se aplique la nueva configuración. Pruebe la conexión otra vez para asegurarse de que el servicio está en ejecución y se puede acceder a él.

Nota: los errores de inicio del servidor, el archivo de configuración utilizado y los errores de licencia se registran en el registro del sistema. Por tanto, si tiene problemas con el servidor, consulte el registro del sistema.

Para obtener más información sobre HTTPS consulte el apartado Configuración HTTPS.

4.1.1 Iniciar el servidor

Temas de este apartado:

- Ubicación del ejecutable del servidor
- Iniciar RaptorXML como servicio (Windows)
- Iniciar Raptor XML como servicio (Linux)
- Iniciar RaptorXML como servicio (macOS)

Ubicación del ejecutable del servidor

El ejecutable de RaptorXML+XBRL Server se instala por defecto en esta carpeta:

```
<CarpetaArchivosProgramas>\Altova\RaptorXMLXBRLServer2018\bin
\RaptorXMLXBRL.exe
```

Puede usar este ejecutable para iniciar RaptorXML+XBRL Server como servicio.

 © 2018 Altova GmbH

Iniciar RaptorXML como servicio en Windows

Durante el proceso de instalación se registra RaptorXML+XBRL Server como servicio en Windows, pero después debe **iniciarlo** como servicio. Tiene varias opciones:

- Con Altova ServiceController, disponible en la bandeja del sistema. Si el icono de Altova ServiceController no aparece en la bandeja del sistema, haga clic en el menú **Inicio** y seleccione **Todos los programas | Altova | Altova LicenseServer | Altova ServiceController**.
- Con la consola de gestión de servicios de Windows: **Panel de control | Todos los elementos de Panel de control | Herramientas administrativas | Servicios**.
- Iniciando el símbolo del sistema con derechos de administrador y usando este comando en cualquier directorio: `net start "Altova RaptorXML+XBRL Server"`
- Con el ejecutable de RaptorXML+XBRL Server en una ventana del símbolo del sistema: `RaptorXMLXBRLServer.exe debug`. Esto inicia el servidor y la información de actividad del servidor aparece directamente en la ventana del símbolo del sistema. Para ver/ocultar esta información, modifique la opción `http.log-screen` del archivo de configuración del servidor. Para detener el servidor, pulse **Ctrl+Interrumpir** (o **Ctrl+Pausa**). Cuando el servidor se inicia de esta forma, el servidor se detiene cuando se cierra la consola de la línea de comandos o cuando el usuario cierra sesión.

Iniciar RaptorXML como servicio en Linux

Utilice este comando para iniciar RaptorXML+XBRL Server como servicio:

[< Debian 8]	`sudo /etc/init.d/raptorxmlxbrlserver start`
[≥ Debian 8]	`sudo systemctl start raptorxmlxbrlserver`
[< CentOS 7]	`sudo initctl start raptorxmlxbrlserver`
[≥ CentOS 7]	`sudo systemctl start raptorxmlxbrlserver`
[< Ubuntu 15]	`sudo initctl start raptorxmlxbrlserver`
[≥ Ubuntu 15]	`sudo systemctl start raptorxmlxbrlserver`
[RedHat]	`sudo initctl start raptorxmlxbrlserver`

Si necesita detener RaptorXML+XBRL Server, utilice este otro comando:

[< Debian 8]	`sudo /etc/init.d/raptorxmlxbrlserver stop`
[≥ Debian 8]	`sudo systemctl stop raptorxmlxbrlserver`
[< CentOS 7]	`sudo initctl stop raptorxmlxbrlserver`
[≥ CentOS 7]	`sudo systemctl stop raptorxmlxbrlserver`
[< Ubuntu 15]	`sudo initctl stop raptorxmlxbrlserver`
[≥ Ubuntu 15]	`sudo systemctl stop raptorxmlxbrlserver`
[RedHat]	`sudo initctl stop raptorxmlxbrlserver`

Iniciar RaptorXML como servicio en macOS

Utilice este comando para iniciar RaptorXML+XBRL Server como servicio:

```
sudo launchctl load /Library/LaunchDaemons/
com.altova.RaptorXMLXBRLServer2018.plist
```

Si necesita detener RaptorXML+XBRL Server, utilice este otro comando:

```
sudo launchctl unload /Library/LaunchDaemons/
com.altova.RaptorXMLXBRLServer2018.plist
```

4.1.2 Probar la conexión

Temas de este apartado:

- Solicitud GET para probar la conexión
- Respuesta del servidor y ejemplo de datos JSON

Solicitud GET para probar la conexión

Tras iniciar RaptorXML+XBRL Server puede probar la conexión con ayuda de una solicitud `GET`. También puede escribir la solicitud en la barra de dirección del explorador.

```
http://localhost:8087/v1/version
```

Nota: la interfaz y el número de puerto de RaptorXML+XBRL Server se especifican en el archivo de configuración del servidor, `server_config.xml`, que se describe en el apartado Configurar el servidor.

Respuesta del servidor y ejemplo de datos JSON

Si el servicio está en ejecución y el servidor está configurado correctamente, la solicitud no debería dar ningún error. RaptorXML+XBRL Server devuelve información sobre su número de versión en forma de datos JSON:

```
{
  "copyright": "Copyright (c) 1998-2013 Altova GmbH. ...",
  "name": "Altova RaptorXML+XBRL Server 2013 rel. 2 sp1",
  "eula": "http://www.altova.com/server_software_license_agreement.html"
}
```

Nota: si modifica la configuración del servidor (al editar el archivo de configuración del servidor), recomendamos que pruebe la conexión una vez más.

 © 2018 Altova GmbH

4.1.3 Configurar el servidor

Temas de este apartado:

- Archivo de configuración del servidor: configuración inicial
- Archivo de configuración del servidor: modificar configuración inicial, volver a la configuración inicial
- Archivo de configuración del servidor: ejemplo y opciones de configuración
- Archivo de configuración del servidor: descripción de las opciones de configuración
- Configurar la dirección del servidor

Archivo de configuración del servidor: configuración inicial

RaptorXML+XBRL Server se configura por medio de un archivo de configuración llamado `server_config.xml`, que se coloca por defecto en esta carpeta:

```
C:\Program Files (x86)\Altova\RaptorXMLXBRLServer2018\etc\server_config.xml
```

La configuración inicial para RaptorXML+XBRL Server define:

- El número de puerto `8087` como puerto del servidor
- Que el servidor solamente escucha a conexiones locales (`localhost`).
- Que el servidor escribe los resultados en la carpeta `C:\ProgramData\Altova\RaptorXMLXBRLServer2018\Output\`.

Las demás opciones de configuración predeterminadas aparecen más abajo en el fragmento del archivo de configuración `server_config.xml`.

Archivo de configuración del servidor: modificar o volver a la configuración inicial

Si quiere cambiar la configuración inicial, edite el archivo de configuración del servidor `server_config.xml` (ver más abajo), guárdelo y después reinicie RaptorXML+XBRL Server como servicio.

Si quiere recrear el archivo de configuración original (para volver a la configuración inicial), ejecute el comando `createconfig`:

```
RaptorXMLXBRL.exe createconfig
```

Al ejecutar este comando, el archivo de configuración inicial se recrea y sobrescribe el archivo `server_config.xml`. Por tanto, el comando `createconfig` sirve para devolver al servidor a su configuración inicial de fábrica.

Archivo de configuración: opciones y ejemplo

A continuación puede ver el archivo de configuración del servidor `server_config.xml` en su versión original. Las opciones de configuración se explican más abajo.

server_config.xml

```
<config xmlns="http://www.altova.com/schemas/altova/raptorxml/config"
  xsi:schemaLocation="http://www.altova.com/schemas/altova/raptorxml/config
  http://www.altova.com/schemas/altova/raptorxml/config.xsd"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:xs="http://www.w3.org/2001/XMLSchema">

  <language>en</language>
  <server.unrestricted-filesystem-access>true</server.unrestricted-filesystem-
access>
  <server.output-root-dir>C:\ProgramData\Altova\RaptorXMLXBRLServer2018\output
\</server.output-root-dir>
  <server.script-root-dir>C:\Program Files\Altova\RaptorXMLXBRLServer2018\etc
\scripts\</server.script-root-dir>
  <!--<server.default-script-api-version>2</server.default-script-api-version>--
>
  <!--<server.catalog-file>catalog.xml</server.catalog-file>-->
  <!--<server.log-file>C:\ProgramData\Altova\RaptorXMLXBRLServer2018\Log
\server.log</server.log-file>-->

  <http.enable>true</http.enable>
  <http.environment>production</http.environment>
  <http.socket-host>127.0.0.1</http.socket-host>
  <http.socket-port>8087</http.socket-port>
  <http.log-screen>true</http.log-screen>
  <http.access-file>C:\ProgramData\Altova\RaptorXMLXBRLServer2018\Log
  \access.log</http.access-file>
  <http.error-file>C:\ProgramData\Altova\RaptorXMLXBRLServer2018\Log\error.log</
  http.error-file>

  <https.enable>false</https.enable>
  <https.socket-host>127.0.0.1</https.socket-host>
  <https.socket-port>443</https.socket-port>
  <https.private-key>C:\Program Files\Altova\RaptorXMLXBRLServer2018\etc\cert
\key.pem</https.private-key>
  <https.certificate>C:\Program Files\Altova\RaptorXMLXBRLServer2018\etc\cert
\cert.pem</https.certificate>
  <!--<https.certificate-chain>/path/to/chain.pem</https.certificate-chain>-->

</config>
```

Opciones de configuración

language

Define el idioma de los mensajes del servidor, en el elemento opcional `language`. El valor predeterminado es `es` (español) y los valores permitidos son `en|de|es|fr|ja` (inglés, alemán, español, francés y japonés respectivamente). Para aprender a localizar (=traducir) RaptorXML, consulte el apartado Comandos de localización.

server.unrestricted-filesystem-access

- Si el valor es `true` (valor predeterminado), los archivos de salida se escriben directamente en la ubicación especificada por el cliente y en los scripts Python (y posiblemente sobrescriben los archivos ya existentes del mismo nombre). Sin embargo, no se pueden utilizar rutas de acceso de archivo locales para acceder a archivos desde un equipo remoto por HTTP. Por tanto, si RaptorXML se está ejecutando en un equipo remoto, utilice el valor **`false`** para esta opción. El valor **`true`** solo se puede usar si el cliente y el servidor están en el mismo equipo y desea escribir los archivos de salida en un directorio de ese equipo.
- Si el valor es `false`, los archivos se escriben en el directorio del trabajo del directorio de salida y los URI de estos archivos se incluye en el documento de resultados. Por tanto, el valor `false` aporta mayor seguridad, porque los archivos solo se pueden escribir en el disco en un directorio del trabajo especializado y conocido. Los archivos de salida del trabajo se pueden copiar después en otras ubicaciones.

`server.output-root-dir`
Directorio en el que se guardan los documentos de salida de todos los trabajos.

`server.script-root-dir`
Directorio en el que se deben guardar los scripts Python de confianza. Cuando se usa con la interfaz HTTP, la opción `script` solamente funciona si se utilizan scripts del directorio de confianza. Si especifica un script Python de otro directorio, se produce un error. Para más información consulte el apartado *Trabajar con scripts Python seguros*.

`server.default-script-api-version`
Versión predeterminada de la API de Python para ejecutar scripts de Python. Se usa por defecto la versión más reciente de la API. Los valores permitidos actualmente son 1 y 2.

`server.catalog-file`
La URL del archivo de catálogo XML que se debe usar. El archivo de catálogo `RootCatalog.xml` está por defecto en la carpeta `<`*`CarpetaArchivosProgramas`*`>\Altova\RaptorXMLXBRLServer2018\etc`. Solamente debe utilizar la opción `server.catalog-file` si quiere cambiar de archivo catálogo.

`server.log-file`
Nombre y ubicación del archivo de registro del servidor. Los eventos del servidor, como *Se inició/ Se detuvo el servidor,* se registran continuamente en el registro de eventos del servidor y aparecen en un visor de eventos del sistema, como el visor de eventos de Windows. Además, los mensajes del registro se pueden escribir en el archivo indicado por medio de la opción `server.log-file`. El archivo de registro del servidor, por tanto, contiene información sobre todas las actividades del servidor, incluidos los errores de inicio del servidor, el archivo de configuración utilizado y los errores de licencia.

`http.enable`
Un valor booleano para habilitar o deshabilitar HTTP: `true | false`. HTTP puede habilitarse/ deshabilitarse independientemente de HTTPS y ambos pueden estar activos simultáneamente.

http.environment
Entornos internos de raptorxml: `production | development`. El entorno de desarrollo `development` está enfocado a los requisitos de los desarrolladores y permite realizar tareas de depuración con mayor facilidad que el entorno de producción `production`.

http.socket-host
Interfaz por la que se accede a RaptorXML+XBRL Server. Si quiere que RaptorXML+XBRL Server acepte conexiones desde equipos remotos, quite la marca de comentario de este elemento y defina su contenido como `0.0.0.0`, es decir: `<http.socket-host>0.0.0.0</http.socket-host>`. Esto almacena el servicio en cada interfaz direccionable del equipo servidor. En este caso, compruebe que las opciones del servidor de seguridad están configuradas correctamente. Las excepciones de entrada del servidor de seguridad para los productos de Altova deben registrarse de la siguiente manera:

- Altova LicenseServer: puerto `8088`
- Altova RaptorXML+XBRL Server: puerto `8087`
- Altova FlowForce Server: puerto `8082`

http.socket-port
El puerto por el que se accede al servicio. El puerto debe ser fijo y conocido para que las solicitudes HTTP se puedan direccionar correctamente al servicio.

http.log-screen
Si inició RaptorXML+XBRL Server con el comando `RaptorXMLXBRLServer.exe debug`, (*ver Iniciar el servidor*) y el valor de `http.log-screen` es `true`,
la actividad del servidor aparece en la consola de la línea de comandos. De lo contrario, la actividad del servidor no se muestra en la consola. Además de aparecer en pantalla, la actividad se registra en archivos de registro.

http.access-file
Nombre y ubicación del archivo de acceso HTTP. El archivo de acceso contiene información sobre la actividad relacionada con el acceso que puede ayudarle a resolver posibles problemas de conexión.

http.error-file
Nombre y ubicación del archivo de errores HTTP. El archivo de errores contiene errores relacionados con el tráfico entrante y saliente del servidor. Si hay problemas de conexión, este archivo puede ayudarle a resolverlos.

http.max_request_body_size
Esta opción especifica el tamaño máximo del cuerpo de la solicitud que RaptorXML+XBRL Server debe aceptar. El valor predeterminado es 100MB. Si el sitio de un cuerpo de solicitud es mayor que el valor indicado en esta opción, el servidor responderá con el error HTTP 413: `Entidad de solicitud demasiado larga`. El valor de la opción debe superior o igual a cero. El límite se puede deshabilitar con `http.max_request_body_size=0`.

 © 2018 Altova GmbH

https.enable
Un valor booleano para habilitar o deshabilitar HTTPS: `true | false`. HTTPS puede habilitarse/deshabilitarse independientemente de HTTP y ambos pueden estar activos simultáneamente. La opción HTTPS está deshabilitada por defecto y debe habilitarse cambiando el valor de esta opción por **`true`**.

https.socket-host
Toma un valor de cadena que es la dirección de host donde se aceptan conexiones HTTP. Para aceptar conexiones del host local solamente, defina **`localhost`** o **`127.0.0.1`**. Si desea que RaptorXML+XBRL Server acepte conexiones de todos los equipos remotos, defina el valor `0.0.0.0`, de la siguiente manera: `<http.socket-host>0.0.0.0</http.socket-host>`. Esto hospeda el servicio en todas las interfaces direccionables del equipo servidor. En este caso se debe comprobar que el firewall está configurado correctamente. Las excepciones de entrada del firewall para productos de Altova deben registrarse de la siguiente manera: Altova LicenseServer: puerto `8088`; Altova RaptorXML+XBRL Server: puerto `8087`; Altova FlowForce Server: puerto `8082`. También puede usar direcciones IPv6 como: `'::'`.

https.socket-port
Un valor entero que es el puerto donde se acepta HTTPS. El puerto debe ser fijo y conocido para que las solicitudes HTTP se puedan dirigir correctamente al servicio.

https.private-key, https.certificate
Identificadores URI que son rutas de acceso de la clave privada del servidor y de los archivos de certificado del servidor respectivamente. Ambos son obligatorios. Consulte los apartados Configuración HTTPS y Configurar el cifrado SSL para obtener más información. En equipos Windows también puede usar rutas de acceso Windows.

https.certificate-chain
Opción opcional. Se trata de un URI que encuentra el archivo de certificado intermedio. Si tiene dos certificados intermedios (principal y secundario), combínelos en un solo archivo siguiendo las instrucciones del paso nº7 del apartado Configurar el cifrado SSL. Consulte los apartados Configuración HTTPS y Configurar el cifrado SSL para obtener más información

La dirección de RaptorXML+XBRL Server
La dirección HTTP del servidor está formada por el host y el puerto del socket:

```
http://{socket-host}:{socket-puerto}/
```

La dirección de la configuración inicial es:

```
http://localhost:8087/
```

Para cambiar de dirección, cambie las opciones `http.socket-host` y `http.socket-port` del archivo de configuración del servidor `server_config.xml`. Por ejemplo, si el equipo servidor tiene la dirección IP `123.12.123.1` y se cambiaron las opciones de configuración

por:
```
<http.socket-host>0.0.0.0</http.socket-host>
<http.socket-port>8087</http.socket-port>
```

Puede dirigirse a RaptorXML+XBRL Server con:
```
http://123.12.123.1:8087/
```

Nota: tras modificar el archivo de configuración del servidor `server_config.xml`, es necesario reiniciar RaptorXML+XBRL Server para que los cambios se apliquen.

Nota: si tiene problemas para conectarse con RaptorXML+XBRL Server, puede que los archivos `http.access-file` y `http.error-file` le ayuden a resolver el problema.

Nota: los mensajes enviados a RaptorXML+XBRL Server deben incluir nombres de ruta de acceso válidos en el equipo servidor. A los documentos del equipo servidor se puede acceder de forma local o remota (en el último caso mediante URI HTTP, por ejemplo).

4.1.4 Configuración HTTPS

RaptorXML+XBRL Server permite el inicio como servidor HTTP y también como servidor HTTPS. Ambos tipos de conexión pueden estar activos de forma simultánea.

Habilitar HTTPS

La opción HTTPS está deshabilitada por defecto. Para habilitar HTTPS en el archivo de configuración del servidor, `server_config.xml`, defina el valor **`true`** para la opción **`https.enable`**. Modifique las distintas opciones HTTPS del archivo de configuración en función de los requisitos de servidor.

Clave privada y certificado

Hay dos maneras de obtener una clave privada y archivos de certificado:

- De una entidad de certificación siguiendo los pasos descritos en el apartado Configurar el cifrado SSL.
- Creando un certificado autofirmado con este comando OpenSSL (modificado según el entorno que se utilice):

```
openssl req -x509 -newkey rsa:4096 -nodes -keyout key.pem -out cert.pem -
days 365 -subj "/C=AT/ST=vienna/L=vienna/O=Altova Gmbh/OU=dev/
CN=www.altova.com"
```

 © 2018 Altova GmbH

Probar la conexión

Puede probar la conexión con ayuda de la herramienta de la línea de comandos curl que sirve para transferir datos con direcciones URL. Puede usar este comando:

```
curl.exe https://localhost:443/v1/version
```

Si el certificado no es de confianza, use la opción **-k**:

```
curl.exe -k https://localhost:443/v1/version
```

Este comando ejecuta el ejemplo HTTP Python que se distribuye con RaptorXML+XBRL Server:

```
python3.exe examples\ServerAPI\python\RunRaptorXML.py --host localhost -p 443
-s
```

4.1.5 Configurar el cifrado SSL

Si desea cifrar la comunicación entre RaptorXML+XBRL Server con el protocolo SSL, será necesario:

- Generar una clave privada SSL y crear un archivo de certificado de clave pública SSL
- Configurar RaptorXML+XBRL Server para la comunicación con cifrado SSL.

Más abajo encontrará instrucciones para hacerlo.

Este método utiliza el kit de herramientas OpenSSL de código abierto para gestionar el cifrado SSL. Por tanto, los pasos que se describen en las instrucciones solo funcionarán en equipos con OpenSSL. El kit de herramientas OpenSSL suele estar instalado por defecto en la mayoría de las distribuciones de Linux y en equipos macOS, pero también se puede instalar en equipos Windows. En la wiki de OpenSSL encontrará enlaces para descargar a los binarios de instalación.

1. Generar una clave privada

   SSL requiere tener instalada una **clave privada** en RaptorXML+XBRL Server. Esta clave privada se utilizará para cifrar todos los datos de RaptorXML+XBRL Server. Para crear la clave privada utilice este comando de OpenSSL:

   ```
   openssl genrsa -out private.key 2048
   ```

   Esto crea un archivo llamado **private.key**, que contiene la clave privada. Recuerde dónde guarda el archivo porque lo necesitará para (i) generar la solicitud de firma de certificado (CSR) y (ii) instalarlo en RaptorXML+XBRL Server.

2. Solicitudes de firma de certificado (CSR)

   La solicitud de firma de certificado (CSR) se envía a una entidad de certificación (como VeriSign o Thawte) para solicitar un certificado de clave pública. La CSR se basa en la clave privada y contiene información sobre su compañía. Cree una CSR con este

comando de OpenSSL:

```
openssl req -new -nodes -key private.key -out my.csr
```

Este comando aporta el archivo de clave privada `private.key` creado en el paso nº1.

Durante la generación de la CSR deberá indicar datos sobre su compañía. La entidad de certificación utilizará estos datos para verificar su identidad:

- *País*
- *Localidad (ciudad donde está situada su compañía)*
- *Organización (nombre de su compañía).* No utilice caracteres especiales porque el certificado no será válido.
- *Nombre común (nombre DNS de su servidor).* Debe ser idéntico al nombre oficial de su servidor (es decir, debe ser el nombre DNS que utilizarán las aplicaciones cliente para conectarse al servidor).
- *Contraseña de comprobación.* Deje este campo vacío.

3. Comprar un certificado SSL

   Compre un certificado SSL de una entidad de certificación reconocida, como VeriSign o Thawte. En adelante utilizamos el procedimiento de VeriSign, pero es similar al de otras entidades de certificación:

   - Visite el sitio web de VeriSign.
   - Haga clic en **Buy SSL Certificates**.
   - Hay varios tipos de certificados SSL a la venta. Para RaptorXML+XBRL Server es suficiente un certificado Secure Site o Secure Site Pro. Como no existe una barra de dirección verde no será necesaria una comprobación extendida (EV).
   - Siga los pasos y rellene el formulario de compra con sus datos.
   - Cuando se le solicite la CSR (creada en el paso nº2), copie y pegue el contenido del archivo **`my.csr`** en el formulario.
   - Efectúe el pago con una tarjeta de crédito válida.

   Tiempo de espera para obtener el certificado
   El certificado de una entidad de certificación SSL suele tardar **dos o tres días laborales**. Tenga esto en cuenta a la hora de configurar RaptorXML+XBRL Server.

4. Recibir la clave pública de la entidad de certificación

   La autoridad de certificación elegida terminará el proceso de registro en dos o tres días laborales. Entre tanto es posible que reciba algún correo electrónico o llamada telefónica para comprobar si tiene autorización para solicitar un certificado SSL para su dominio DNS.

   Una vez completado el proceso de registro y autorización, recibirá un correo electrónico con la clave pública de su certificado SSL. Esta clave pública estará en texto sin formato o será un archivo **`.cer`**.

 © 2018 Altova GmbH

5. Guardar la clave pública en un archivo

   Para poder usarla con RaptorXML+XBRL Server la clave pública debe estar guardada en un archivo **`.cer`**. Si recibió la clave pública como texto sin formato, copie y pegue todas las líneas de la clave, desde `--BEGIN CERTIFICATE--` hasta `--END CERTIFICATE--` en un archivo de texto que llamaremos **`miCertificado.cer`**.

6. Guardar los certificados intermedios de la autoridad de certificación en un archivo

   Para completar el certificado SSL necesitará otros dos certificados: el **certificado intermedio principal** y el **certificado intermedio secundario**. En el sitio web de su autoridad de certificación encontrará el contenido de los certificados intermedios:

   - Certificados intermedios de Verisign: https://knowledge.verisign.com/support/ssl-certificates-support/index?page=content&id=AR657&actp=LIST&viewlocale=en_US
   - Certificados intermedios de Verisign para su producto Secure Site: https://knowledge.verisign.com/support/ssl-certificates-support/index?page=content&id=AR1735

   Copie y pegue los dos certificados intermedios en sendos archivos de texto y guárdelos en el equipo.

7. (Opcional) Combinar los certificados en un solo archivo de certificado de clave pública

   Ahora cuenta con tres archivos de certificado:

   - La clave pública (`miCertificado.cer`)
   - El certificado intermedio secundario
   - El certificado intermedio principal

   Si lo desea, puede integrar los certificados intermedios en el certificado de clave pública. Esto se explica a continuación. Si lo prefiere, puede usar la opción del archivo de configuración **`https.certificate-chain`** para especificar la ubicación de los certificados intermedios.

   Todos contienen bloques de texto entre líneas similares a estas:

   ```
   --BEGIN CERTIFICATE--
     ...
   --END CERTIFICATE--
   ```

   Ahora copie y pegue los tres certificados en un solo archivo, uno detrás del otro. El orden de aparición es importante: (i) primero la clave pública, (ii) después el certificado intermedio secundario y (iii) por último el certificado intermedio principal. Compruebe que no hay líneas vacías entre un certificado y el siguiente.

   `--BEGIN CERTIFICATE--`
   ***clave pública** de* `miCertificado.cer` *(paso nº5)*
   `--END CERTIFICATE--`
   `--BEGIN CERTIFICATE--`
   ***certificado intermedio secundario** (paso nº6)*

```
--END CERTIFICATE--
--BEGIN CERTIFICATE--
```
certificado intermedio principal *(paso nº6)*
```
--END CERTIFICATE--
```

Guarde el texto resultante en un archivo llamado **`publickey.cer`**, que es ya el certificado de clave pública de su certificado SSL. Incluye el certificado de clave pública y la cadena de confianza (es decir, los certificados intermedios utilizados por la entidad de certificación para firmar el certificado.

 © 2018 Altova GmbH

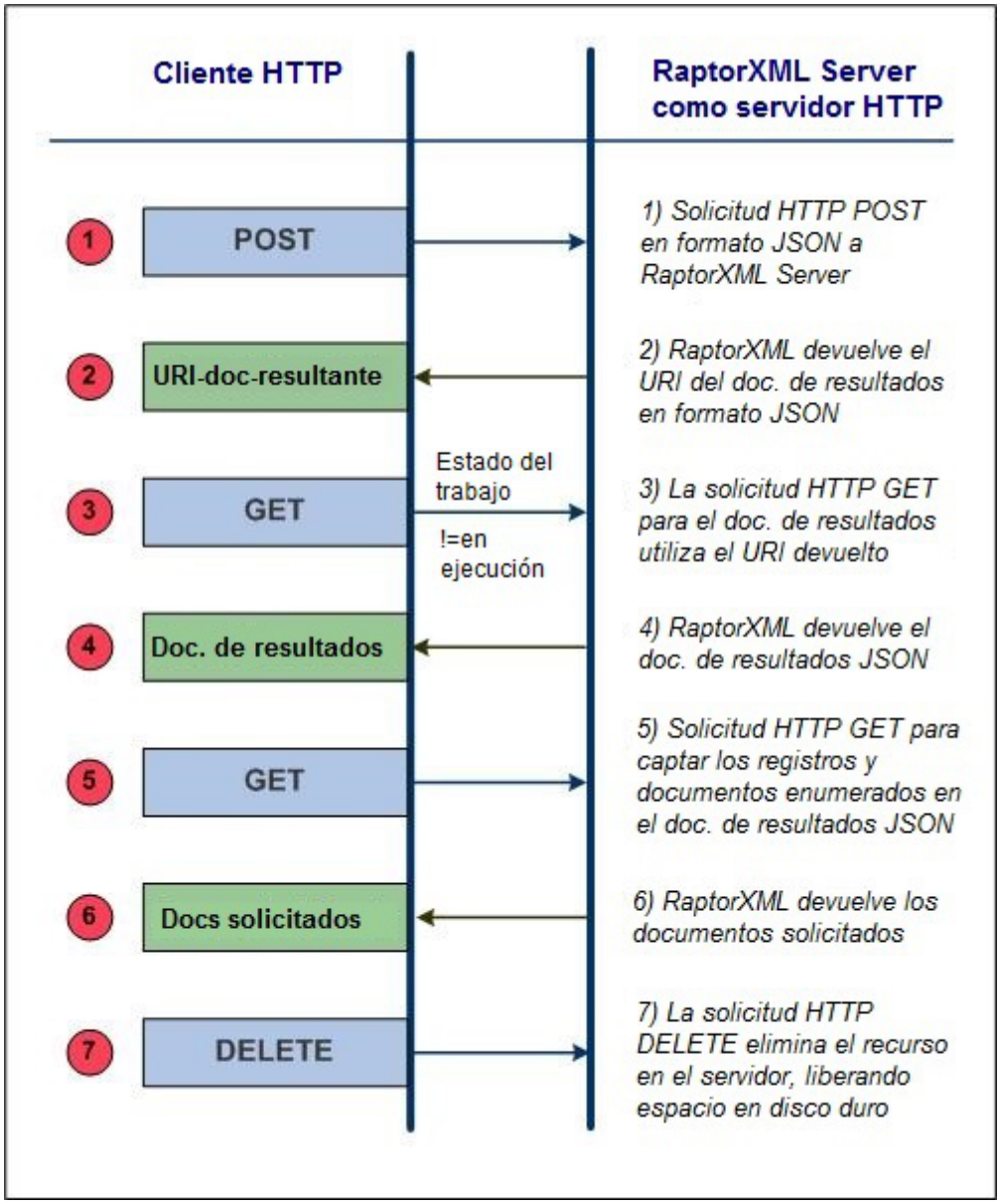
4.2 Solicitudes cliente

Tras iniciar RaptorXML+XBRL Server como servicio, podrá acceder a sus funciones cualquier cliente HTTP que pueda:

- usar los métodos HTTP `GET`, `PUT`, `POST` y `DELETE`
- establecer el campo de encabezado `Content-Type`

Un sencillo cliente HTTP
Hay varios clientes web que puede descargar de Internet, pero el cliente web RESTClient de Mozilla es sencillo y fiable y se puede añadir como complemento de Firefox. Es fácil de instalar, es compatible con los métodos HTTP necesarios para trabajar con RaptorXML y ofrece funciones de color de sintaxis JSON. Si no tiene experiencia con clientes HTTP, recomendamos trabajar con RESTClient. No obstante, la instalación y el uso del cliente RESTClient corre por su cuenta y riesgo.

Una solicitud cliente normal suele estar formada por varios pasos, como muestra el diagrama siguiente.



A continuación destacamos los aspectos más importantes de cada paso, señalando en **negrita** los términos más importantes.

1. Los métodos HTTP `POST` se utilizan para realizar una solicitud y el cuerpo de la solicitud está en **formato JSON**. La solicitud puede ser para cualquier función de RaptorXML +XBRL Server. Por ejemplo, la solicitud puede ser para una validación o una transformación XSLT. Los comandos, los argumentos y las opciones utilizados en la solicitud son los mismos que los de la línea de comandos. La solicitud se envía a **`http://localhost:8087/v1/queue`**, si suponemos que **`localhost:8087`** es la dirección de RaptorXML+XBRL Server (la dirección inicial del servidor). Dicha solicitud recibe el nombre de **trabajo de RaptorXML+XBRL Server**.

2. Si RaptorXML+XBRL Server recibe y acepta procesar la solicitud, después de procesar el

trabajo se genera un **documento de resultados** que contiene los resultados de la acción del servidor. El **URI de este documento de resultados** (ver *URI-doc-resultante* en el diagrama anterior), se devuelve al cliente. Recuerde que el URI se devolverá inmediatamente después de que se acepte procesar el trabajo (después de que se ponga en la cola) e incluso si el procesamiento no ha terminado.

3. El cliente envía al servidor una solicitud para el documento de resultados (usando el URI del documento de resultados) en un método `GET`. Si todavía no empezó el procesamiento del trabajo o si, cuando se recibe la solicitud, el procesamiento continúa, el servidor devuelve el estado *En ejecución*. La solicitud `GET` debe repetirse hasta que el procesamiento del trabajo haya finalizado y se haya creado el documento de resultados.

4. RaptorXML+XBRL Server devuelve el documento de resultados en formato JSON. El documento de resultados puede contener los **URI de los documentos de errores o de salida** generados por RaptorXML+XBRL Server al procesar la solicitud inicial. Por ejemplo, si una validación devuelve errores, se devuelve un registro de errores. Los documentos de salida principales, como el resultado de la transformación XSLT, se devuelven si el trabajo generador de resultados finaliza correctamente.

5. El cliente envía al servidor los URI de los documentos de salida recibidos en el paso nº4 a través de un método HTTP `GET`. Cada solicitud se envía en un método `GET` distinto.

6. RaptorXML+XBRL Server devuelve los documentos solicitados en respuesta a las solicitudes `GET` realizadas en el paso nº5.

7. El cliente puede eliminar del servidor documentos no deseados generados como resultado de una solicitud de trabajo. Esto se hace enviando en un método HTTP `DELETE` el URI del documento de resultados en cuestión. Se eliminan del disco todos los archivos relacionados con dicho trabajo. Esto incluye el documento de resultados, los archivos temporales, los documentos de errores y los documentos de salida. Este paso es muy práctico si quiere liberar espacio en el disco duro del servidor.

Todos estos pasos se describen con más detalle en los apartados de esta sección.

4.2.1 Iniciar trabajos con POST

Temas de este apartado:

- Enviar la solicitud
- Sintaxis JSON para solicitudes POST
- Cargar archivos con la solicitud POST
- Cargar archivos ZIP

Enviar la solicitud

Los trabajos de RaptorXML+XBRL Server se inician con el método HTTP `POST`.

Método	URI	Campo de	Cuerpo

HTTP		encabezado Content-Type	
POST	http://localhost:8087/v1/queue/	application/json	JSON

Observe que:

- El URI de la tabla tiene una dirección de servidor que utiliza las opciones de la configuración inicial.
- El URI tiene una ruta de acceso /v1/queue/, que debe estar presente en el URI. Puede entenderse como una carpeta abstracta en memoria y en ella se coloca el trabajo.
- El número de versión /vN es el que devuelve el servidor (no necesariamente el que aparece en esta documentación). El número que devuelve el servidor es el número de versión de la interfaz HTTP actual. Los números de versión anteriores indican versiones antiguas de la interfaz HTTP y se admiten para garantizar la compatibilidad con versiones anteriores.
- El encabezado debe incluir el campo: Content-Type: application/json. Sin embargo, si quiere cargar archivos dentro del cuerpo de la solicitud POST, el encabezado del mensaje debe tener el tipo de contenido multipart/form-data (es decir, Content-Type: multipart/form-data). Para más información consulte el apartado Cargar archivos con la solicitud POST.
- El cuerpo de la solicitud debe estar en formato JSON.
- Los archivos que desea procesar deben estar en el servidor. Por tanto, debe copiar los archivos al servidor antes de realizar la solicitud o cargarlos junto con la solicitud POST. En este último caso, el encabezado del mensaje debe tener el tipo de contenido multipart/form-data. Para más información consulte el apartado Cargar archivos con la solicitud POST.

Esta sería la solicitud en formato JSON para comprobar si un archivo XML tiene un formato correcto:

```
{
   "command": "wfxml", "args": [ "file:///c:/Test/Report.xml" ]
}
```

Los comandos, sus argumentos y sus opciones se documentan en la sección Interfaz de la línea de comandos.

Sintaxis JSON para solicitudes HTTP POST

```
{
  "command": "Nombre-Comando",
  "options": {"opción1": "opción1-valor", "opción2": "opción2-valor"},
  "args"   : ["file:///c:/nombreArchivo1", "file:///c:/nombreArchivo2"]
}
```

 © 2018 Altova GmbH

- El texto de color negro es obligatorio y debe incluirse. Esto incluye todas las llaves, comillas, dos puntos, comas y corchetes. Los espacios en blanco se pueden normalizar.
- El texto de color azul en cursiva son marcadores de posición para nombres de comandos, opciones, valores de opciones y valores de argumentos. Para más información consulte la sección Interfaz de la línea de comandos.

- Las claves `command` y `args` son obligatorias. La opción `options` es opcional. Algunas claves `options` tienen valores predeterminados. Por tanto, de todas estas opciones, solamente debe especificar aquellas cuyos valores predeterminados desee cambiar.

- Todas las cadenas de texto deben ir entre comillas dobles. Los valores booleanos y los números no necesitan ir entre comillas. El uso correcto sería `{"error-limit": "unlimited"}` y `{"error-limit": 1}`.

- Es recomendable usar los URI de archivo y no sus rutas de acceso. Recuerde que los URI usan barras diagonales, mientras que las rutas de acceso usan barras diagonales inversas. Además, en Windows es necesario añadir caracteres de escape a las barras diagonales inversas de las rutas de acceso (es decir, `"c:\\dir\\nombreArchivo"`). Por último, recuerde que los URI y las rutas de acceso son cadenas y, por tanto, deben ir entre comillas.

A continuación puede ver un ejemplo con opciones. Observe que algunas opciones (como `input` o `xslt-version`) toman un valor directo, mientras que otros (como `param`) toman un par clave-valor y, por tanto, necesitan otra sintaxis.

```
{
   "command": "xslt",
   "args": [
           "file:///C:/Work/Prueba.xslt"
           ],
   "options": {
               "input": "file:///C:/Work/Prueba.xml",
               "xslt-version": 1,
               "param": {
                        "key": "miPruebaParam",
                        "value": "ValorDeParám"
                        },
               "output": "file:///C:/temp/salida2.xml"
               }
}
```

El ejemplo que aparece a continuación muestra otro tipo de opción más: una matriz de valores (como en la opción `xsd` del ejemplo). En casos así, debe usar la sintaxis de una matriz JSON.

```
{
   "command": "xsi",
   "args": [
           "file:///C:/Work/Prueba.xml"
           ],
   "options": {
              "xsd" : ["file:///C:/Work/File1.xsd", "file:///C:/Work/
Archivo2.xsd"]
              }
}
```

Cargar archivos con la solicitud POST

Puede cargar los archivos que deben procesarse dentro del cuerpo de la solicitud `POST`. En este caso, la solicitud `POST` debe crearse como se muestra a continuación.

Encabezado de la solicitud

En el encabezado de la solicitud, defina `Content-Type: multipart/form-data` y especifique una cadena de texto cualquiera como frontera. Por ejemplo:

```
Content-Type: multipart/form-data; boundary=---FronteraParcial
```

El objetivo de la frontera es delimitar las diferentes partes `form-data` del cuerpo de la solicitud (*ver más abajo*).

Cuerpo de la solicitud: parte del mensaje

El cuerpo de la solicitud está formado por varias partes `form-data`, separadas por la cadena de frontera especificada en el encabezado (*ver más arriba*):

- *Partes `form-data` obligatorias:* la parte **`msg`**, que especifica la acción de procesamiento solicitada, y la parte **`args`**, que contiene los archivos que deben cargarse como argumentos del comando especificado en la parte **`msg`**. (*Ver ejemplo que aparece más abajo*).
- *Parte `form-data` opcional:* la parte **`additional-files`** contiene los archivos a los que se hace referencia desde las partes `form-data` obligatorias **`msg`** y **`args`**. Además, las partes `form-data` con nombre de opción de comando también pueden incluir archivos para cargar.

Nota: todos los archivos cargados se crean en el mismo directorio virtual.

Consulte el Ejemplo nº1 (con llamadas): validar XML para ver un ejemplo de código y el Ejemplo nº2: usar un catálogo para buscar el esquema.

Pruebas con CURL

Puede usar aplicaciones externas de transferencia de datos como CURL (http://curl.haxx.se/) para probar la solicitud POST. CURL ofrece una opción de seguimiento muy práctica que genera y enumera las fronteras parciales de las solicitudes. Esto evita tener que crear a mano las fronteras parciales. Para más información consulte el apartado Pruebas con CURL.

Cargar archivos ZIP

También puede cargar archivos ZIP y hacer referencia a los ficheros del archivo ZIP con el esquema `additional-files`. Por ejemplo:

```
additional-files:///migranarchivo.zip%7Czip/instanciagrande.xml
```

Nota: en la parte `|zip/` el URI tiene que estar entre caracteres de escape (`%7Czip/`) para ajustarse al RFC del URI porque el símbolo `|` no está permitido. El uso de patrones glob (`*` y `?`) también está permitido. Por tanto, para validar todos los archivos XML del archivo ZIP puede usar algo así:

```
{"command": "xsi", "args": ["additional-files:///migranarchivo.zip%7Czip/
*.xml"], "options": {...}}
```

Para ver otro ejemplo de código consulte el Ejemplo nº3: usar archivos ZIP.

4.2.1.1 *Ejemplo nº1 (con llamadas): validar XML*

A continuación puede ver un ejemplo en el que aparece el cuerpo de una solicitud `POST` seguida de una tabla donde se explican todas sus partes (que aparecen numeradas). El comando de la ILC equivalente al comando enviado con esta solicitud sería:

```
raptorxmlxbrl xsi Primero.xml Segundo.xml --xsd=Demo.xsd
```

La solicitud pide validar dos archivos XML con un esquema. El cuerpo de la solicitud tendría este aspecto, si imaginamos que en el encabezado se especificó la cadena de frontera `---PartBoundary` (*ver Encabezado de solicitud*).

```
-----PartBoundary                                                    1
Content-Disposition: form-data; name="msg"
Content-Type: application/json
```

```
{"command": "xsi", "options": {} , "args": []}                          2

-----PartBoundary                                                       3
Content-Disposition: attachment; filename="Primero.xml"; name="args"
Content-Type: application/octet-stream

  <?xml version="1.0" encoding="UTF-8"?>                                4
  <test xsi:noNamespaceSchemaLocation="Demo.xsd" xmlns:xsi="http://
  www.w3.org/2001/XMLSchema-instance">42</test>

-----PartBoundary                                                       5
Content-Disposition: attachment; filename="Segundo.xml"; name="args"
Content-Type: application/octet-stream

  <?xml version="1.0" encoding="UTF-8"?>                                6
  <test xsi:noNamespaceSchemaLocation="Demo.xsd" xmlns:xsi="http://
  www.w3.org/2001/XMLSchema-instance">35</test>

-----PartBoundary                                                       7
Content-Disposition: attachment; filename="Demo.xsd";
name="additional-files"
Content-Type: application/octet-stream

  <?xml version="1.0" encoding="UTF-8"?>                                8
  <xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
  elementFormDefault="qualified" attributeFormDefault="unqualified">
     <xs:element name="test" type="xs:int"/>
  </xs:schema>

-----PartBoundary--                                                     9
```

1. El nombre de las fronteras de las principales partes `form-data` se declaran en el encabezado de la solicitud. El separador de frontera de parte debe ser una cadena única que no aparezca en ninguna otra parte de los documentos incrustados. Se le añade un prefijo de dos guiones y se utiliza para separar sus partes. La primera parte `form-data` (de este ejemplo) es la parte **`msg`**. Observe que su tipo de contenido es `application/json`.
2. Esta es la sintaxis estándar para solicitudes HTTP `POST`. Si `args` contiene una referencia a un archivo y se cargaron más archivos, ambos grupos de archivos se pasan al servidor.
3. El primer miembro de la matriz `args` es un archivo adjunto llamado `Primero.xml`.
4. El texto del archivo `Primero.xml`. Contiene una referencia a un esquema llamado `Demo.xsd`, que también se cargará (en la parte `form-data` **`additional_files`**).
5. El segundo miembro de la matriz **`args`** es un archivo adjunto llamado `Segundo.xml`.
6. El texto del archivo `Segundo.xml`. También contiene una referencia al esquema `Demo.xsd` (*ver el punto 7*).

7 El final del último miembro de la matriz **`args`** (y por tanto de la matriz propiamente dicha) se indica añadiendo el sufijo `'--'` a la cadena de frontera de **`args`**.

8 El texto del archivo `Demo.xsd`.

9 El final de la parte **`additional_files`**. Observe que el último separador de frontera de parte tiene tanto un prefijo como un sufijo formados por dos guiones.

4.2.1.2 Ejemplo nº2: usar un catálogo para buscar el esquema

En este ejemplo se utiliza un archivo de catálogo para buscar el esquema XML al que hacen referencia los archivos XML que se deben validar.

```
-----PartBoundary
Content-Disposition: form-data; name="msg"
Content-Type: application/json

{"command": "xsi", "args": ["additional-files:///Primero.xml", "additional-
files:///Segundo.xml"], "options": {"user-catalog": "additional-files:///
catalog.xml"}}

-----PartBoundary
Content-Disposition: attachment; filename="Primero.xml"; name="additional-files"
Content-Type: application/octet-stream

<?xml version="1.0" encoding="UTF-8"?>
<test xsi:noNamespaceSchemaLocation="http://ejemplo.com/Demo.xsd"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">42</test>

-----PartBoundary
Content-Disposition: attachment; filename="Segundo.xml"; name="additional-files"
Content-Type: application/octet-stream

<?xml version="1.0" encoding="UTF-8"?>
<test xsi:noNamespaceSchemaLocation="http://ejemplo.com/Demo.xsd"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">35</test>

-----PartBoundary
Content-Disposition: attachment; filename="Demo.xsd"; name="additional-files"
Content-Type: application/octet-stream

<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified" attributeFormDefault="unqualified">
   <xs:element name="test" type="xs:int"/>
</xs:schema>

-----PartBoundary
Content-Disposition: attachment; filename="catalog.xml"; name="additional-files"
Content-Type: application/octet-stream
```

```
<?xml version='1.0' encoding='UTF-8'?>
<catalog xmlns='urn:oasis:names:tc:entity:xmlns:xml:catalog' xmlns:xsi='http://
www.w3.org/2001/XMLSchema-instance'
xsi:schemaLocation='urn:oasis:names:tc:entity:xmlns:xml:catalog Catalog.xsd'>
                    <uri name="http://ejemplo.com/Demo.xsd" uri="additional-
files:///Demo.xsd"/>
</catalog>

-----PartBoundary--
```

4.2.1.3 *Ejemplo nº3: usar archivos ZIP*

También puede cargar archivos ZIP y hacer referencia a los ficheros del archivo ZIP con el esquema **`additional-files`**. Por ejemplo:

```
additional-files:///migranarchivo.zip%7Czip/instanciagrande.xml
```

Nota: en la parte **`|zip/`** el URI tiene que estar entre caracteres de escape (**`%7Czip/`**) para ajustarse al RFC del URI porque el símbolo **`|`** no está permitido. El uso de patrones glob (**`*`** y **`?`**) también está permitido. Por tanto, para validar todos los archivos XML del archivo ZIP puede usar algo así:

```
{"command": "xsi", "args": ["additional-files:///migranarchivo.zip%7Czip/
*.xml"], "options": {...}}
```

Nota: `'Content-Disposition: form-data'` también es válido, junto con `'Content-Disposition: attachment'`. Como varias herramientas generan `form-data` como `content-disposition`, también se acepta el valor `form-data`.

Ejemplo: validar todos los archivos XML del archivo ZIP

En este ejemplo, se supone que todas las referencias de esquema son rutas de acceso relativas y que todos los esquemas están dentro del archivo ZIP.

```
-----PartBoundary
Content-Disposition: form-data; name="msg"
Content-Type: application/json

{"command": "xsi", "args": ["additional-files:///Demo.zip%7Czip/*.xml"],
"options": {}}

-----PartBoundary
Content-Disposition: attachment; filename="Demo.zip"; name="additional-files"
Content-Type: application/octet-stream

Contenido binario del archivo Demo.zip

-----PartBoundary--
```

Ejemplo: validar archivos XML del archivo ZIP con referencias a esquemas externos

En este ejemplo los archivos XML del archivo ZIP se validan por medio de referencias a un esquema externo, que está en otro archivo ZIP.

```
-----PartBoundary
Content-Disposition: form-data; name="msg"
Content-Type: application/json

{"command": "xsi", "args": ["additional-files:///Instancias.zip%7Czip/*.xml"],
"options": {"user-catalog": "additional-files:///Schemas.zip%7Czip/
catalog.xml"}}

-----PartBoundary
Content-Disposition: attachment; filename="Instancias.zip"; name="additional-
files"
Content-Type: application/octet-stream

Contenido binario del archivo Instancias.zip

-----PartBoundary
Content-Disposition: attachment; filename="Esquemas.zip"; name="additional-
files"
Content-Type: application/octet-stream

Contenido binario del archivo Esquemas.zip

-----PartBoundary--
```

4.2.1.4 Pruebas con CURL

La aplicación CURL (http://curl.haxx.se/) es una utilidad de la línea de comandos que puede utilizarse para probar la solicitud POST. CURL ofrece una opción de seguimiento muy práctica que genera y enumera las fronteras de parte de las solicitudes. Puede utilizar estas fronteras en sus solicitudes directamente o utilizarlas como referencia.

Más abajo puede ver un ejemplo donde se valida un archivo XML con un esquema XML.

- En este ejemplo se supone que los comandos se ejecutan desde la carpeta donde residen los archivos que se deben validar (esto nos permite escribir rutas de acceso relativas sencillas para estos archivos). Si instaló la aplicación Altova XMLSpy, encontrará los archivos utilizados en el ejemplo en la carpeta `Examples` de la aplicación, cuya ubicación predeterminada es: `C:\Users\<nombreUsuario>\Documents\Altova\XMLSpy2018\Examples`.
- En este ejemplo se supone que RaptorXML+XBRL Server se ejecuta localmente en el puerto 8087.

Para más información sobre las opciones de la línea de comandos CURL consulte la

documentación de esta aplicación.

Llamar a CURL con el comando de validación en Windows

```
[input: powershell]
  \path\to\curl.exe -F 'msg={\"command\": \"xsi\", \"args\":[\"additional-
  files:///PurchaseOrder.zip%7Czip/ipo.xml\"], \"options\":{}};type=application/
  json' -F "additional-files=@PurchaseOrder.zip;type=application/octet-stream"
  http://localhost:8087/v1/queue
```

Nota: en powershell, si utiliza comillas dentro de otras comillas, éstas deben ser de dos tipos diferentes (sencillas y dobles).

```
[input: cmd]
  \path\to\curl.exe -F "msg={\"command\": \"xsi\", \"args\":[\"additional-
  files:///PurchaseOrder.zip%7Czip/ipo.xml\"], \"options\":{}};type=application/
  json" -F "additional-files=@PurchaseOrder.zip;type=application/octet-stream"
  http://localhost:8087/v1/queue

[output]
  {"jobid": "058F9E97-CB95-43EF-AC0A-496CD3AC43A3", "result": "/v1/
  results/058F9E97-CB95-43EF-AC0A-496CD3AC43A3"}
```

Usar la URL de "result" para recuperar el resultado

```
[input]
  \path\to\curl.exe http://localhost:8087/v1/results/058F9E97-CB95-43EF-AC0A-
  496CD3AC43A3

[output]
  {"jobid":"058F9E97-CB95-43EF-AC0A-496CD3AC43A3","state":"OK","error":
  {},"jobs":[{"file":"additional-files:///PurchaseOrder.zip%7Czip/
  ipo.xml","jobid":"D4B91CB0-CF03-4D29-B563-B6506E123A06","output":
  {},"state":"OK","error":{}}]}
```

Opción de seguimiento de CURL

CURL ofrece la opción de seguimiento (`--trace-ascii`), que sigue el tráfico HTTP entrante y saliente del servidor. La opción es muy práctica porque enumera las fronteras de parte que son necesarias para iniciar trabajos con POST. Puede usar la información de seguimiento directamente o como referencia para crear fronteras de partes. En el fragmento de código que aparece a continuación puede ver la información de seguimiento que se obtiene al ejecutar el comando.

- *Fragmento de código de seguimiento*

```
== Info:   Trying ::1...
== Info: Connected to localhost (::1) port 8087 (#0)
=> Send header, 217 bytes (0xd9)
0000: POST /v1/queue HTTP/1.1
0019: Host: localhost:8087
002f: User-Agent: curl/7.42.1
0048: Accept: */*
0055: Content-Length: 2939
006b: Expect: 100-continue
0081: Content-Type: multipart/form-data; boundary=--------------------
00c1: ----d887ed58324015c3
00d7:
<= Recv header, 23 bytes (0x17)
0000: HTTP/1.1 100 Continue
=> Send data, 393 bytes (0x189)
0000: --------------------------d887ed58324015c3
002c: Content-Disposition: form-data; name="msg"
0058: Content-Type: application/json
0078:
007a: {"command": "xsi", "args":["additional-files:///PurchaseOrder.zi
00ba: p%7Czip/ipo.xml"], "options":{}}
00dc: --------------------------d887ed58324015c3
0108: Content-Disposition: form-data; name="additional-files"; filenam
0148: e="PurchaseOrder.zip"
015f: Content-Type: application/octet-stream
0187:
=> Send data, 2498 bytes (0x9c2)
0000: PK........"..6}.c.....M.......ipo.xsd.T.N.@.}N....O 5v.}..S....(
0040:  .JU/...$Y..5{.E.▯.....I*...g...Y...\....Z..~......P.A.ct....▯y.
...
0940: ..........."..6]g......l............. .......address.xsdPK.......
0980: ..."..6I..v................. .......ipo.xmlPK.................
09c0: ..
=> Send data, 48 bytes (0x30)
0000:
0002: --------------------------d887ed58324015c3--
<= Recv header, 22 bytes (0x16)
0000: HTTP/1.1 201 Created
<= Recv header, 13 bytes (0xd)
0000: Allow: POST
<= Recv header, 32 bytes (0x20)
0000: Content-Type: application/json
<= Recv header, 37 bytes (0x25)
0000: Date: Fri, 24 Jul 2015 16:58:08 GMT
<= Recv header, 24 bytes (0x18)
0000: Server: CherryPy/3.6.0
<= Recv header, 21 bytes (0x15)
0000: Content-Length: 111
<= Recv header, 2 bytes (0x2)
0000:
<= Recv data, 111 bytes (0x6f)
0000: {"jobid": "058F9E97-CB95-43EF-AC0A-496CD3AC43A3", "result": "/v1
0040: /results/058F9E97-CB95-43EF-AC0A-496CD3AC43A3"}
== Info: Connection #0 to host localhost left intact
```

Nota: observe que también es válido usar `'Content-Disposition: form-data'` junto con `'Content-Disposition: attachment'`.

Llamar a CURL con el comando well-formed-check en Linux

```
/path/to/curl -F 'msg={"command": "wfxml", "args":[]};type=application/json' -
F "args=@ipo.xml;type=application/octet-stream" http://localhost:8087/v1/queue

/path/to/curl -F 'msg={"command": "wfxml", "args":["additional-files:///
ipo.zip%7Czip/ipo.xml"]};type=application/json' -F "additional-
files=@ipo.zip;type=application/octet-stream" http://localhost:8087/v1/queue
```

4.2.2 Respuesta del servidor a solicitudes POST

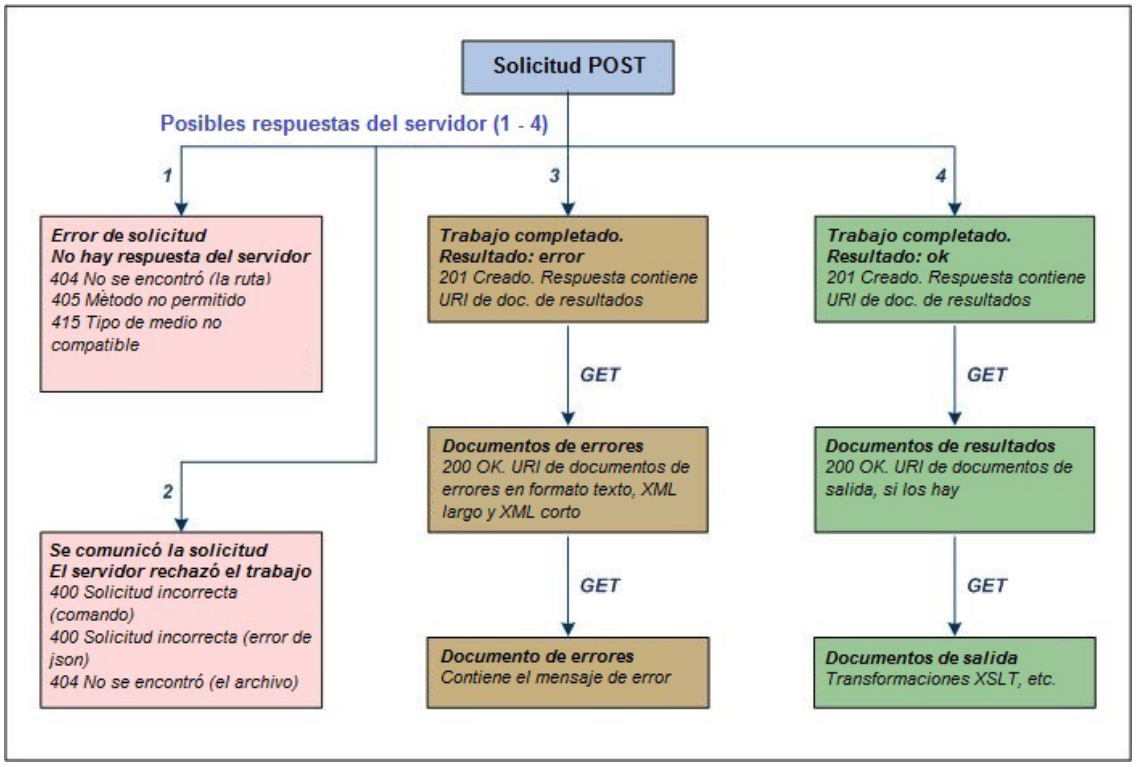
Temas de este apartado:

- Posibles respuestas del servidor
- Respuesta: Error de solicitud, sin respuesta del servidor
- Respuesta: Se comunicó la solicitud, pero el servidor rechazó el trabajo
- Respuesta: Se ejecutó el trabajo (con resultados positivos o negativos)

Cuando la solicitud POST se realiza correctamente, el trabajo se coloca en la cola de trabajos del servidor y se devuelve un mensaje `201 Creado` y el URI del documento de resultados. El trabajo se procesará lo antes posible. Mientras tato, si se solicita el documento de resultados, se devuelve el mensaje `"estado": "En ejecución"` (si el trabajo está sin completar) o el mensaje `"estado": "Distribuido"` (si el trabajo está en la cola del servidor pero todavía no se inició).

El resultado del trabajo (por ejemplo, de un trabajo de validación) puede ser negativo ("***Failed***") o positivo ("***OK***"). En ambos casos se devuelve el mensaje `201 Creado` y se genera el documento de resultados. También puede ocurrir que la solicitud POST no se comunique al servidor (*Error de solicitud*) o que se comunique al servidor pero este la rechace (Se *comunicó la solicitud, pero el servidor rechazó el trabajo*). Todas las posibles respuestas del servidor se resumen en este diagrama:

 © 2018 Altova GmbH



Las posibles respuestas a la solicitud `POST` del cliente son:

Error de solicitud, no hay respuesta del servidor

Cuando la solicitud se envía al servidor, los errores más comunes son:

Mensaje	Motivo
404 No se encontró (*la ruta*)	La ruta de acceso correcta es `http://localhost:8087/v1/queue/`
405 Método no permitido	El método especificado para este recurso no es válido. Utilice el método `POST`.
415 Tipo de medio no compatible	El encabezado del mensaje debería ser `Content-Type:application/json`.

Se comunicó la solicitud, pero el servidor rechazó el trabajo

Cuando la solicitud se envía al servidor, este puede rechazarla por varios motivos:

Mensaje	Motivo
400 Solicitud incorrecta (*comando malo*)	El comando de RaptorXML no es correcto.

400 Solicitud incorrecta (*error de JSON*)	Hay un error de sintaxis JSON en el cuerpo de la solicitud.
404 No se encontró (*el archivo*)	Revise la sintaxis del URI (o ruta) de archivo de todos los archivos nombrados en el comando.

El trabajo se ejecutó (con resultados negativos o positivos)

Cuando se ejecuta un trabajo (por ejemplo, un trabajo de validación) su resultado puede ser positivo (***OK***) o negativo (***Failed***). Por ejemplo, el resultado de un trabajo de validación es positivo (***OK***) cuando el documento es válido y negativo (***Failed***) cuando el documento no es válido.

En ambos casos se ejecuta el trabajo, pero los resultados obtenidos son diferentes. En ambos casos se devuelve el mensaje `201 Creado` en cuanto el trabajo se coloca en la cola de trabajos del servidor. Además, en ambos casos se devuelve un URI de documento de resultados al cliente HTTP que hizo la solicitud. (Puede que el documento de resultados propiamente dicho no exista aún si el procesamiento del trabajo no comenzó/finalizó). Una vez creado el documento de resultados, puede recuperarlo con una solicitud HTTP `GET`. Además del documento de resultados se generan algunos documentos más:

- *Trabajo ejecutado. Resultado 'Failed':* se crea un registro de errores en tres formatos, texto, XML largo y XML corto. Los URI de estos tres documentos se envían en el documento de resultados (que está en formato JSON). Los URI se pueden usar en una solicitud HTTP `GET` para recuperar los documentos de errores.
- *Trabajo ejecutado. Resultado 'OK':* el trabajo se procesa correctamente y se crean documentos de salida (por ejemplo, los de una transformación XSLT). Si se generan archivos de salida, sus URI se incluyen en el documento de resultados. Estos URI se pueden usar en una solicitud HTTP `GET` para recuperar los documentos. Recuerde que no todos los trabajos tienen archivos de salida (por ejemplo, los trabajos de validación). Además, un trabajo puede terminar con un resultado positivo pero generar advertencias y otros mensajes en los archivos de errores. En este caso, los URI de los archivos de errores también se envían en el documento de resultados.

Para más información sobre estos documentos y cómo acceder a ellos consulte los apartados Obtener el documento de resultados y Obtener documentos de errores y de salida.

4.2.3 Obtener el documento de resultados

Temas de este apartado:

- El URI del documento de resultados
- Recuperar el documento de resultados
  - Documento de resultados con los URI de los documentos de errores
  - Documento de resultados con los URI de los documentos de salida
  - Documento de resultados sin URI
- Acceder a los documentos de salida y de errores enumerados en el documento de resultados

 © 2018 Altova GmbH

El URI del documento de resultados

Cada vez que se crea un trabajo se genera un documento de resultados, independientemente de si el resultado del trabajo (p. ej. un trabajo de validación) es positivo (el documento es válido) o negativo (el documento no es válido). En ambos casos se devuelve un mensaje `201 Creado`. Este mensaje está en formato JSON y contiene el URI relativo del documento de resultados. Por ejemplo:

```
{
  "result": "/v1/results/E6C4262D-8ADB-49CB-8693-990DF79EABEB",
  "jobid": "E6C4262D-8ADB-49CB-8693-990DF79EABEB"
}
```

El objeto `result` contiene el URI relativo del documento de resultados. El URI es relativo a la dirección del servidor. Por ejemplo, si la dirección del servidor es `http://localhost:8087/` (la dirección según la configuración inicial), el URI ampliado del documento de resultados anterior sería:

```
http://localhost:8087/v1/results/E6C4262D-8ADB-49CB-8693-990DF79EABEB
```

Nota: el número de versión `/vN` es el que devuelve el servidor (no necesariamente el de esta documentación). El número que devuelve el servidor es el número de versión de la interfaz HTTP actual. Los números de versión antiguos indican versiones previas de la interfaz HTTP, que se admiten para garantizar la compatibilidad con versiones previas.

Recuperar el documento de resultados

Para obtener el documento de resultados puede enviar el URI ampliado del documento (*ver más arriba*) en una solicitud HTTP `GET`. Se devuelve el documento de resultados, que puede ser de tres tipos (*ver más abajo*).

Nota: si el trabajo se coloca en la cola del servidor, este devuelve el URI del documento de resultados. Si el cliente solicita el resultado antes de que comience el trabajo (si todavía está en la cola), se devuelve el mensaje `"estado": "Distribuido"`. Si el trabajo se inició pero no terminó todavía (porque es un trabajo grande, por ejemplo), se devuelve el mensaje `"estado": "En ejecución"`. En ambos casos el cliente debe esperar un poco para volver a solicitar el documento de resultados.

Nota: en los ejemplos que aparecen a continuación se entiende que hay acceso de cliente restringido. Es decir, se entiende que los documentos de errores, mensajes y de salida se guardan en el directorio correspondiente del servidor. Los URI de estos documentos en el documento de resultados son, por tanto, URI relativos. Ninguno de ellos es un URI de archivo (como el que se generaría con acceso de cliente no restringido). Para más información consulte el siguiente apartado de esta sección.

Documento de resultados con los URI de los documentos de errores

Si el trabajo solicitado termina con el estado **`Failed`**, el trabajo devuelve un resultado negativo.

Por ejemplo, un trabajo de validación devuelve un resultado de documento no válido. Los errores detectados durante la ejecución del trabajo se almacenan en registros de errores, creados en tres formatos de archivo: (i) texto, (ii) XML largo (registro detallado) y (iii) XML corto (registro menos detallado).


```
{
  "jobid": "6B4EE31B-FAC9-4834-B50A-582FABF47B58",
  "state": "Failed",
  "error":
  {
    "text": "/v1/results/6B4EE31B-FAC9-4834-B50A-582FABF47B58/error/error.txt",
    "longxml": "/v1/results/6B4EE31B-FAC9-4834-B50A-582FABF47B58/error/
  long.xml",
    "shortxml": "/v1/results/6B4EE31B-FAC9-4834-B50A-582FABF47B58/error/
  short.xml"
  },
  "jobs":
  [
    {
      "file": "file:///c:/Test/ExpReport.xml",
      "jobid": "20008201-219F-4790-BB59-C091C276FED2",
      "output":
      {
      },
      "state": "Failed",
      "error":
      {
        "text": "/v1/results/20008201-219F-4790-BB59-C091C276FED2/error/
  error.txt",
        "longxml": "/v1/results/20008201-219F-4790-BB59-C091C276FED2/error/
  long.xml",
        "shortxml": "/v1/results/20008201-219F-4790-BB59-C091C276FED2/error/
  short.xml"
      }
    }
  ]
}
```


Es importante destacar que:

- Los trabajos tienen trabajos subordinados.
- Los errores de los trabajos subordinados se propagan al trabajo de nivel superior. Es decir, el estado del trabajo solamente puede ser OK si el de los trabajos subordinados también lo es.
- Cada trabajo y trabajo subordinado tiene su registro de errores.
- Los registros de errores incluyen advertencias. Es decir, incluso si el trabajo tiene el estado OK, puede haber archivos de errores.
- Los URI de los archivos de errores son relativos a la dirección del servidor (*ver más arriba*).

Documento de resultados con los URI de los documentos de salida

Si el trabajo solicitado termina con el estado OK, el trabajo devuelve un resultado positivo. Por ejemplo, un trabajo de validación devuelve un resultado de documento válido. Si el trabajo generó un documento de salida (por ejemplo, como resultado de una transformación XSLT), se devuelve el URI del documento de salida.

 © 2018 Altova GmbH

```
{
  "jobid": "5E47A3E9-D229-42F9-83B4-CC11F8366466",
  "state": "OK",
  "error":
  {
  },
  "jobs":
  [
    {
      "file": "file:///c:/Test/SimpleExample.xml",
      "jobid": "D34B5684-C6FF-4A7A-BF35-EBB9A8A8C2C8",
      "output":
      {
        "xslt-output-file":
        [
          "/v1/results/D34B5684-C6FF-4A7A-BF35-EBB9A8A8C2C8/output/1"
        ]
      },
      "state": "OK",
      "output-mapping":
      {
        "/v1/results/D34B5684-C6FF-4A7A-BF35-EBB9A8A8C2C8/output/1": "file:///
        c:/temp/test.html"
      },
      "error":
      {
      }
    }
  ]
}
```


Es importante destacar que:

- El archivo de salida se crea en la carpeta `output` del trabajo. Puede usar su URI relativo para acceder al archivo.
- Los URI de los archivos de salida son relativos a la dirección del servidor (*ver más arriba*).
- El elemento `output-mapping` asigna el documento de salida del directorio del trabajo del servidor a la ubicación de archivo especificada por el cliente en la solicitud. Recuerde que solo tienen una asignación los documentos de salida especificados por el cliente en la solicitud. Los archivos relacionados con el trabajo generados por el servidor (como los archivos de errores) no tienen ninguna asignación.
- Otra opción es recuperar todos los documentos de resultados generados para un trabajo específico en un archivo ZIP por medio de la URL `"/v1/results/JOBID/output/zip"`. Esta característica no está disponible en el modo unrestricted filesystem. Recuerde que el archivo ZIP incluirá nombres de archivo alterados, que deberán asignarse a los nombres reales con ayuda del objeto `output-mapping`.

Documento de resultados sin URI

Si el trabajo solicitado terminó con el estado `OK`, significa que el trabajo devolvió un resultado positivo. Por ejemplo, un trabajo de validación devuelve un resultado de documento válido. Algunos trabajos, como los de validación o de comprobación de formato, no generan documentos de salida. Si un trabajo de este tipo termina con el estado `OK`, el documento de resultados generado no tendrá ni el URI de documento de salida ni el URI del registro de errores.


```
{
```

```
  "jobid": "3FC8B90E-A2E5-427B-B9E9-27CB7BB6B405",
  "state": "OK",
  "error":
  {
  },
  "jobs":
  [
    {
      "file": "file:///c:/Test/SimpleExample.xml",
      "jobid": "532F14A9-F9F8-4FED-BCDA-16A17A848FEA",
      "output":
      {
      },
      "state": "OK",
      "error":
      {
      }
    }
  ]
}
```


Observe que:

- Tanto los componentes de salida y de error del trabajo subordinado están vacíos en el ejemplo anterior.
- Un trabajo puede terminar con un estado OK, pero con advertencias y otro tipo de mensajes, que se registran en los archivos de errores. En este caso, el documento de resultados incluirá el URI de los archivos de errores, aunque el trabajo terminara con el estado OK.

Acceder a los documentos de errores y de salida enumerados en el documento de resultados

Puede acceder a los documentos de errores y de salida con solicitudes HTTP `GET`. Para más información consulte el siguiente apartado de esta sección.

4.2.4 Obtener los documentos de salida/errores

Un documento de resultados puede contener los URI de archivo o los URI relativos de los documentos de errores, documentos de mensajes (como los registros) y documentos de salida. (En algunos casos el documento de resultados no contiene ningún URI). Los diferentes tipos de URI se describen más abajo.

Para acceder a estos documentos por HTTP:

1. Amplíe el URI relativo del archivo del documento de resultados hasta su URI absoluto.
2. Use el URI ampliado en una solicitud HTTP `GET` para acceder al archivo.

URI (en el documento de resultados) de los documentos de errores, de mensajes y de salida

El documento de resultados contiene los URI de los documentos de errores, de mensajes y de salida. Los documentos de errores y de mensajes están relacionados con el trabajo y los genera

el servidor. Siempre se guardan en el servidor, en el directorio del trabajo. Los documentos de salida (como los resultantes de transformaciones XSLT) se pueden guardar en una de estas dos ubicaciones:

- En cualquier ubicación de archivo a la que pueda acceder el servidor. Para guardar los archivos en cualquier ubicación, debe configurar el servidor para permitir acceso no restringido al cliente (configuración predeterminada).
- En el directorio de trabajo del servidor. El servidor se configura para restringir el acceso al cliente.

Si un cliente especifica que se debe crear un archivo de salida, la ubicación en la que se guarda el archivo depende de la opción `server.unrestricted-filesystem-access` del archivo de configuración del servidor.

- Si el acceso no está restringido, el archivo se guardará en la ubicación especificada por el ciente y el URI devuelto para el documento será un URI de archivo.
- Si el acceso está restringido, el archivo se guardará en el directorio del trabajo y su URI será un URI relativo. Además, se creará una asignación entre este URI relativo y la URL de archivo especificada por el cliente (ver el ejemplo del apartado *Documento de resultados con los URI de los documentos de salida*)

Para resumir, hay 3 tipos de URI:

URI de archivo de los documentos de errores / mensajes
Estos documentos se guardan en el servidor en el directorio del trabajo. Los URI de archivo tienen este formato:

```
file:///<dir-raíz-de-salida>/JOBID/mensaje.doc
```

URI de archivo de los documentos de salida
Estos documentos se guardan en cualquier ubicación. Los URI de archivo tienen este formato:

```
file:///<ruta-del-archivo>/salida.doc
```

URI HTTP de los documentos de errores/mensajes/salida
Estos documentos se guardan en el servidor en el directorio del trabajo. Los URI son relativos a la dirección del servidor y deben ampliarse a un URI HTTP. Este es su formato:

`/vN/results/JOBID/error/error.txt`	para documentos de errores
`/vN/results/JOBID/output/verbose.log`	para documentos de mensajes
`/vN/results/JOBID/output/1`	para documentos de salida

En el caso de los documentos de salida, se dan las asignaciones de salida (*ver fragmento de ejemplo*). Estas asignaciones unen cada URI de documento de salida del documento de resultados con el documento correspondiente de la solicitud cliente.

Ampliar el URI relativo

Amplíe el URI relativo del documento de resultados a un URI HTTP absoluto añadiendo la dirección del servidor como prefijo al URI relativo. Por ejemplo, si la dirección del servidor es:

`http://localhost:8087/` (la configuración inicial)

y el URI relativo de un archivo de errores del documento de resultados es:

```
/v1/results/20008201-219F-4790-BB59-C091C276FED2/error/error.txt
```

entonces la dirección absoluta ampliada será

```
http://localhost:8087/v1/results/20008201-219F-4790-BB59-C091C276FED2/error/
error.txt
```

Para más información consulte los apartados Configurar el servidor y Obtener el documento de resultados.

Utilice una solicitud HTTP GET para acceder a los archivos

Utilice el URI ampliado en una solicitud HTTP GET para obtener el archivo que necesita. RaptorXML+XBRL Server devuelve el documento solicitado.

4.2.5 Liberar espacio tras el procesamiento

RaptorXML+XBRL Server guarda en el disco duro el documento de resultados, archivos temporales, documentos de errores y documentos de salida relacionados con un trabajo. Hay dos maneras de eliminar estos archivos:

- Suministrando el URI del documento de resultados con un método HTTP DELETE. Esto elimina todos los archivos relacionados con el trabajo indicado con el URI del documento de resultados, incluidos los documentos de salida y de errores.
- Eliminando a mano los archivos del servidor (se necesitan derechos de administrador).

La estructura del URI que debe usar con el método HTTP DELETE aparece a continuación. Observe que el URI está formado por la dirección del servidor más el URI relativo del documento de resultados.

Método HTTP	URI
DELETE	http://localhost:8087/v1/result/D405A84A-AB96-482A-96E7-4399885FAB0F

Para encontrar el directorio de salida de un trabajo en el disco, construya el URI de esta forma:

[<server.output-root-dir> *ver archivo de configuración del servidor*] + [IDtrabajo]

Nota: es posible que se generen muchos documentos de salida y de errores, por lo que aconsejamos supervisar el uso del disco duro y programar eliminaciones frecuentes, en función de su entorno y requisitos.

 © 2018 Altova GmbH

Altova RaptorXML+XBRL Server 2018

Interfaces API de Python y .NET

5 Interfaces API de Python y .NET

RaptorXML+XBRL Server viene con estas interfaces API que permiten acceder a las funciones y características de RaptorXML:

- API de Python
- API de .NET Framework

Registrar el equipo cliente con Altova LicenseServer

Para poder ejecutar el paquete de la API en un equipo cliente, deberá asignarle una licencia como si se tratase de un cliente RaptorXML+XBRL Server. La asignación de licencias es un proceso de dos pasos:

1. Primero debe registrar el equipo como cliente RaptorXML+XBRL Server con el servidor de licencias Altova LicenseServer.
2. Después debe asignar al equipo una licencia de RaptorXML+XBRL Server desde Altova LicenseServer.

Si tiene pensado ejecutar el paquete de la API desde un equipo concreto, tiene dos opciones:

- Si el equipo cliente ya ejecuta una versión con licencia de RaptorXML+XBRL Server, entonces podrá ejecutar el paquete de la API sin necesidad de configurar nada. El motivo es que el equipo ya tiene asignada una licencia que le permite ejecutar RaptorXML+XBRL Server. Por tanto, el uso del paquete de la API en dicho equipo está habilitado por medio de la licencia que se asignó a RaptorXML+XBRL Server en dicho equipo.
- Si RaptorXML+XBRL Server no está instalado en el equipo y prefiere no instalarlo por el motivo que sea, deberá registrar el equipo como cliente RaptorXML+XBRL Server y asignarle una licencia de RaptorXML+XBRL Server (ver instrucciones más abajo).

Para registrar un equipo (que no tenga RaptorXML+XBRL Server instalado) como cliente RaptorXML+XBRL Server basta con usar la aplicación de la línea de comandos **`registerlicense.exe`**, disponible en la carpeta **`bin`** de la aplicación.

Windows	`Archivos de programa\Altova\RaptorXMLXBRLServer2018\bin`
Linux	`/opt/Altova/RaptorXMLXBRLServer2018/bin`
Mac	`/usr/local/Altova/RaptorXMLXBRLServer2018/bin`

Después, ejecute el comando:

```
registerlicense <LicenseServer>
```

donde **`<LicenseServer>`** será la dirección IP o nombre de host del equipo donde se ejecuta el servidor de licencias LicenseServer.

Este comando registrará el equipo como cliente RaptorXML+XBRL Server con Altova LicenseServer. Para aprender a asignar licencias al equipo y a comprender el proceso de

 © 2018 Altova GmbH

asignación de licencias consulte la documentación de Altova LicenseServer.

Implementación en Linux

Para implementar la aplicación **registerlicense** con el paquete wheel de Python, las bibliotecas compartidas que aparecen a continuación deben estar presentes en un directorio **lib** del mismo nivel. Las bibliotecas compartidas se pueden copiar desde la carpeta de instalación de RaptorXML:

/opt/Altova/RaptorXMLServer2017/lib

- libcrypto.so.1.0.0
- libssl.so.1.0.0
- libstdc++.so.6
- libtbb.so.2

5.1 API de Python

La API de Python de RaptorXML+XBRL Server permite acceder y recuperar datos de documentos XML, XSD, instancias XBRL y taxonomías XBRL a través de diferentes API de Python (para XML, XSD y XBRL). Los datos de los documentos de origen que se deben procesar y el método de procesamiento se especifican en un script Python que se envía a RaptorXML+XBRL Server.

Interfaces API de Python

Las API de Python (para XML, XSD y XBRL) ofrecen acceso a metadatos, información estructural y datos incluidos en documentos XML, XSD, instancias XBRL y taxonomías XBRL. Como resultado, puede crear scripts Python que utilicen las API para acceder y procesar los datos de estos documentos. Por ejemplo, se puede enviar un script Python a RaptorXML+XBRL Server que escriba datos desde un documento XML o de instancia XBRL en una base de datos o en un archivo CSV.

Puede encontrar ejemplos de scripts para las API de Python de RaptorXML en https://github.com/altova

También puede consultar la referencia de las API de Python en el sitio web de Altova:

- Referencia de la API de Python (versión 1)
- Referencia de la API de Python (versión 2)

Nota: La **versión 1 de la API de Python de Raptor está obsoleta**. Use en su lugar la versión 2 de la API de Python.

Paquete de RaptorXML+XBRL Server para Python

En la instalación de RaptorXML+XBRL Server también encontrará un paquete Python en formato wheel. Puede usar el comando `pip` de Python para instalar este paquete como módulo de la instalación Python. Tras instalar el módulo RaptorXMLXBRL, podrá usar sus funciones dentro del código. De este modo, las características y funciones de RaptorXMLXBRL se pueden usar fácilmente en cualquier programa Python junto con bibliotecas Python de otros autores, como bibliotecas gráficas, por ejemplo.

Para más información sobre cómo utilizar el paquete de RaptorXML+XBRL Server para Python consulte la sección RaptorXML+XBRL Server como paquete Python.

Scripts Python

Los scripts Python creados por el usuario se pueden enviar a RaptorXML con el parámetro `--script` de estos comandos:

- `valxml-withxsd (xsi)`
- `valxsd (xsd)`
- `valxbrltaxonomy (dts)`
- `valxbrl (xbrl)`

 © 2018 Altova GmbH

Estos comandos que invocan a scripts Python se pueden usar tanto en la interfaz de la línea de comandos (ILC) como en la interfaz HTTP. El uso de scripts Python dentro de las API de Python de RaptorXML+XBRL Server se documenta en https://github.com/altova.

Trabajar con scripts Python seguros

Cuando se especifica un script Python por HTTP para RaptorXML+XBRL Server, el script solo funciona si está ubicado en el directorio de confianza. El script se ejecuta desde el directorio de confianza. Si especifica un script de cualquier otro directorio, se produce un error. El directorio de confianza se define en la opción `server.script-root-dir` del archivo de configuración del servidor y **es obligatorio** especificar un directorio de confianza si quiere usar scripts Python. Por tanto, asegúrese de guardar en este directorio todos los scripts Python que desea usar.

Aunque todos los resultados generados por el servidor para solicitudes de trabajo HTTP se escriben en el directorio de salida de trabajos (que es un subdirectorio de `output-root-directory`), esta limitación no afecta a los scripts Python, que pueden escribir en cualquier ubicación. El administrador del servidor debería revisar los scripts Python del directorio de confianza para evitar problemas de seguridad.

5.1.1 Versiones de las API de Python

RaptorXML+XBRL Server ofrece varias versiones de la API de Python y todas las versiones antiguas de la API de Python son compatibles con la versión actual de RaptorXML. La versión de la API de Python se selecciona con la marca de la línea de comandos **`--script-api-version=MAJOR_VERSION`**. El valor predeterminado del argumento `MAJOR_VERSION` siempre es la versión actual. Cada vez que se introduzcan mejoras o cambios incompatibles, estará disponible un `MAJOR_VERSION` nuevo para la API de Python de RaptorXML+XBRL Server. El usuario de la API no tiene que actualizar sus scripts cuando se publiquen versiones nuevas.

Sin embargo, recomendamos:

- Utilizar la marca `--script-api-version=MAJOR_VERSION` para invocar scripts de otras utilidades desde la línea de comandos de RaptorXML+XBRL Server (o desde la API web). Así se garantiza que los scripts sigan funcionando tras la instalación de actualizaciones de RaptorXML+XBRL Server (incluso si se publica una versión nueva de `MAJOR_VERSION`).
- Utilizar la versión más reciente de la API para proyectos nuevos, aunque las versiones nuevas de RaptorXML+XBRL Server sean compatibles con versiones antiguas de la API.

Las versiones de la API que aparecen más abajo son las que están disponibles actualmente. La documentación de estas API se puede consultar en el sitio web de Altova (ver enlaces más abajo).

Archivos de ejemplo

Puede encontrar ejemplos de scripts para las API de Python de RaptorXML en https://github.com/altova.

API de Python (versión 1)

Se introdujo con RaptorXML+XBRL Server v2014.

Marca de la línea de comandos:	**`--script-api-version=1`**
Documentación:	Referencia de la API de Python (versión 1)

Se trata de la API de Python de RaptorXML+XBRL Server original. Ofrece funciones para acceder al modelo interno de RaptorXML+XBRL Server para:

- XML 1.0 y XML 1.1 (módulo API **`xml`**)
- XMLSchema 1.0 y XMLSchema 1.1 (módulo API **`xsd`**)
- XBRL 2.1 (módulo API **`xbrl`**)

La API se puede utilizar a través de varias funciones de devolución de llamada que se implementan en un archivo de script Python.

- `on_xsi_valid`
- `on_xsd_valid`
- `on_dts_valid`
- `on_xbrl_valid`

El script se especifica con la opción `--script` en la línea de comandos. Las funciones de devolución de llamada se invocan solamente si la validación finaliza correctamente. Para obtener más información sobre las funciones de devolución de llamada y la API, consulte la Referencia de la API de Python (versión 1) en el sitio web de Altova.

Nota: La **versión 1 de la API de Python de Raptor está obsoleta**. Use en su lugar la versión 2 de la API de Python.

API de Python (versión 2)

Se introdujo con RaptorXML+XBRL Server v2015r3. La versión más reciente de la API es la versión **`2.5.0`**.

Marca de la línea de comandos:	**`--script-api-version=2`**
	`--script-api-version=2.1`
	`--script-api-version=2.2`
	`--script-api-version=2.3`
	`--script-api-version=2.4`
	`--script-api-version=2.4.1`
	`--script-api-version=2.5.0`
		`v 2015r3`
		`v 2015r4`
		`v 2016`
		`v 2016r2`
		`v 2017`
		`v 2018`
		`v 2018r2`

Documentación:	Referencia de la API de Python (versión 2)	

 © 2018 Altova GmbH

Esta versión de la API introduce más de 300 clases nuevas y reorganiza los módulos de la versión 1 de tal modo que la información que se utiliza con más frecuencia (como los datos PSVI, por ejemplo) sea más accesible. Además las API relacionadas se agrupan de forma lógica (por ejemplo, `xbrl.taxonomy`, `xbrl.formula`, `xbrl.table`). En esta versión las funciones de devolución de llamada no solamente se invocan cuando la validación finaliza correctamente, sino también cuando se produce un error durante la validación. Por ello se ha modificado el nombre de las funciones de devolución de llamada:

- `on_xsi_finished`
- `on_xsd_finished`
- `on_dts_finished`
- `on_xbrl_finished`

Además, RaptorXML+XBRL Server ofrece varias opciones `--script`. Las devoluciones de llamada implementadas en los scripts Python se ejecutan en el orden indicado en la línea de comandos.

5.1.2 RaptorXML+XBRL Server como paquete Python

A partir de la versión 2017 de RaptorXML+XBRL Server, la API de Python está disponible como paquete wheel nativo de Python para **Python 3.5**. El paquete wheel de Python puede instalarse como módulo de extensión en la distribución de Python 3.5 que usted prefiere (p. ej. la distribución de python.org). Algunas distribuciones de Python 3 (p. ej. las de jupyter.org, anaconda.org y SciPy.org) incluyen una amplia gama de módulos de extensión para datos masivos, matemáticas, ciencia, ingeniería y gráficos. Estos módulos ya pueden estar a disposición de RaptorXML+XBRL Server sin necesidad de generarlos para RaptorXML+XBRL Server específicamente. Por lo demás, el paquete wheel funciona igual que la aplicación **`RaptorXMLXBRL-python.exe`** que viene con RaptorXML+XBRL Server.

Nota: el paquete wheel de Python es un módulo de extensión de Python y debe coincidir con la versión de Python que tenga instalada.

En este apartado encontrará toda la información necesaria para instalar el paquete RaptorXML +XBRL Server correctamente:

- Nombre del archivo wheel
- Ubicación del archivo wheel
- Instalar un archivo wheel con pip
- El archivo de catálogo raíz
- El archivo de configuración JSON

Para más información sobre cómo usar la API de Python de RaptorXML+XBRL Server consulte la referencia de la API de Python y los ejemplos. También encontrará ejemplos de scripts que usan la API de Python en https://github.com/altova.

Nombre del archivo wheel

El nombre de los archivos wheel siguen este patrón:

```
raptorxmlserver-{versión}(-{marca compilación})?-{marca python}-{marca abi}-
{marca plataforma}.whl
```

Ejemplos:
raptorxmlserver-2.4.0-cp35-cp35m-win_amd64.whl
raptorxmlserver-2.4.1-cp35-cp35m-win_amd64.whl

Ubicación del archivo wheel

La instalación de RaptorXML+XBRL Server viene con un archivo wheel, ubicado en la carpeta **bin** de la aplicación:

Windows	Archivos de programa\Altova\RaptorXMLXBRLServer2018\bin
Linux	/opt/Altova/RaptorXMLXBRLServer2018/bin
Mac	/usr/local/Altova/RaptorXMLXBRLServer2018/bin

Instalar un archivo wheel con pip

Utilice el comando **pip** para instalar el paquete RaptorXML+XBRL Server como módulo de Python:

```
pip install <archivo-wheel>.whl
python -m pip install <archivo-wheel>.whl
```

Si instaló Python 3.5 o superior desde python.org, entonces también tendrá instalado pip. Si no es así, primero deberá instalar **pip**. Visite https://docs.python.org/3/installing/ para obtener más información.

El archivo de catálogo raíz

El módulo RaptorXMLXBRL para Python debe ser capaz de encontrar **RootCatalog.xml**, el archivo de catálogo raíz que está en la carpeta de instalación de RaptorXML+XBRL Server. El módulo RaptorXMLXBRL necesita el catálogo para encontrar los diferentes recursos (como esquemas y especificaciones) a los que hace referencia el módulo para llevar a cabo diferentes funciones, como validaciones y transformaciones, por ejemplo. El módulo RaptorXMLXBRL encontrará **RootCatalog.xml** automáticamente si la ubicación del catálogo no cambió desde que se instaló RaptorXML+XBRL Server.

En caso de que se modificara el entorno de RaptorXML+XBRL Server o de que se moviera **RootCatalog.xml** a otra ubicación, podrá especificar la ubicación del catálogo mediante variables de entorno y desde el archivo de configuración JSON del módulo RaptorXMLXBRL. En la lista que aparece a continuación puede ver varios métodos diferentes. El módulo RaptorXMLXBRL determina la ubicación de **RootCatalog.xml** buscando estos recursos en el orden especificado en la tabla.

1	Variable de entorno ALTOVA_RAPTORXML_PYTHON_CATALOGPATH	Crear con un valor que sea la ruta de acceso de **RootCatalog.xml**

2	Registro HKLM: `SOFTWARE\Altova\RaptorXMLXBRLServer\Installation_v2018_x64\Setup\CatalogPath`	La clave de registro la añade el programa de instalación de RaptorXML+XBRL Server. Su valor es la ruta de acceso de **`RootCatalog.xml`**. (*Sólo para Windows.*)
3	Ubicación: `/opt/Altova/RaptorXMLXBRLServer2018/etc/RootCatalog.xml`	*Sólo para Linux.*
4	Ubicación: `/usr/local/Altova/RaptorXMLXBRLServer2018/etc/RootCatalog.xml`	*Sólo para Mac.*
5	Variable de entorno `ALTOVA_RAPTORXML_PYTHON_CONFIG`	Crear con un valor que sea la ruta de acceso del archivo de configuración JSON.
6	Ubicación: `.altova/`**`raptorxml-python.config`**	El archivo de configuración JSON que está en el directorio de trabajo actual.
7	Ubicación: `~/.config/altova/`**`raptorxml-python.config`**	El archivo de configuración JSON que está en el directorio de inicio del usuario.
8	Ubicación: `/etc/altova/altova/`**`raptorxml-python.config`**	El archivo de configuración JSON. (*Sólo para Linux y Mac.*)

El archivo de configuración JSON

Puede crear una archivo de configuración JSON para el módulo RaptorXMLXBRLServer. Este archivo se usará en las opciones 5, 6, 7 y 8 de la tabla anterior para buscar el archivo de catálogo raíz. El archivo de configuración JSON debe contener un mapa con una clave `"CatalogPath"` cuyo valor sea la ruta de acceso del archivo de catálogo raíz.

Fragmento de un archivo de configuración JSON

```
{
    "CatalogPath": "/path/to/RootCatalog.xml"
}
```

5.2 API de .NET Framework

La **API de .NET Framework** de RaptorXML+XBRL Server permite integrar RaptorXML+XBRL Server en aplicaciones escritas en C# y en otros lenguajes .NET.

Se implementa como ensamblado .NET y coloca RaptorXML+XBRL Server dentro de una aplicación directamente o de un mecanismo de extensión basado en .NET Framework como VSTO (Visual Studio Tools for Office). La API ofrece acceso a funciones de validación de documentos y consulta del modelo de datos desde RaptorXML+XBRL Server. El mecanismo utilizado es similar al que se utiliza para la API de Python de RaptorXML+XBRL Server.

Referencia y recursos

- *Documentación de la API:* la versión más reciente de la documentación de la API de .NET Framework de RaptorXML+XBRL Server puede consultarse aquí: http://manual.altova.com/es/RaptorXML/dotnetapiv2/html/index.html.
- *Ejemplo de código:* puede consultar un ejemplo en https://github.com/altova/RaptorXML-Examples.

 © 2018 Altova GmbH

Altova RaptorXML+XBRL Server 2018

Interfaz Java

6 Interfaz Java

A la API de RaptorXML también se puede acceder desde código Java. Para ello es necesario que las bibliotecas que aparecen más abajo residan en el parámetro classpath. Esta bibliotecas se instalan en la carpeta `bin` de la carpeta de instalación.

- `RaptorXMLServer.jar`: es la biblioteca que se comunica con el servidor de RaptorXML mediante solicitudes HTTP.
- `RaptorXMLServer_JavaDoc.zip`: es un archivo Javadoc que contiene la documentación de ayuda para la API de Java.

Nota: para poder usar la API de Java, el archivo Jar debe estar en el parámetro classpath. Si quiere, puede copiar el archivo Jar en la ubicación que mejor le venga a su proyecto en lugar de hacerle referencia desde la carpeta de instalación.

Resumen sobre la interfaz Java

La API de Java viene empaquetada en el paquete `com.altova.raptorxml`. La clase RaptorXML ofrece un método de punto de entrada llamado `getFactory()`, que devuelve objetos `RaptorXMLFactory`. De modo que se puede crear una instancia de `RaptorXMLFactory` con la llamada `RaptorXML.getFactory()`.

La interfaz `RaptorXMLFactory` ofrece métodos para obtener objetos del motor para realizar tareas de validación y procesamiento (como transformaciones XSLT).

Nota: el método `getFactory` devuelve el objeto de fábrica correspondiente dependiendo de la edición de RaptorXML que esté instalada.

La interfaz pública de `RaptorXMLFactory` se describe en este fragmento:

```
public interface RaptorXMLFactory
{
   public XMLValidator getXMLValidator();
   public XBRL getXBRL();
   public XQuery getXQuery();
   public XSLT getXSLT();
   public void setServerName(String name) throws RaptorXMLException;
   public void setServerFile(String file) throws RaptorXMLException;
   public void setServerPort(int port) throws RaptorXMLException;
   public void setGlobalCatalog(String catalog);
   public void setUserCatalog(String catalog);
   public void setGlobalResourcesFile(String file);
   public void setGlobalResourceConfig(String config);
   public void setErrorFormat(ENUMErrorFormat format);
   public void setErrorLimit(int limit);
   public void setReportOptionalWarnings(boolean report);
}
```

 © 2018 Altova GmbH

Para más información consulte la descripción de `RaptorXMLFactory` y de la correspondiente interfaz Java. También puede consultar un ejemplo de proyecto Java.

6.1 Ejemplo de proyecto Java

El fragmento de código Java que aparece más adelante muestra cómo acceder a las funciones básicas. El código tiene varias partes:

- Busca la carpeta de ejemplos y crea una instancia de objeto COM de RaptorXML
- Valida un archivo XML
- Realiza una transformación XSLT y devuelve el resultado en forma de cadena de texto
- Procesa un documento XQuery y devuelve el resultado en forma de cadena de texto
- Ejecuta el proyecto

Estas funciones básicas se incluyen en los archivos de la carpeta `examples/API` de la carpeta de aplicación de RaptorXML+XBRL Server.

```
public class RunRaptorXML
{
    // Buscar muestras instaladas con el producto
    // (está dos niveles más arriba de examples/API/Java)
    // NOTA: quizás sea necesario cambiar esta ruta de acceso
       static final String strExamplesFolder = System.getProperty("user.dir") +
"/../../" ;

       static com.altova.raptorxml.RaptorXMLFactory rxml;





       static void ValidateXML() throws com.altova.raptorxml.RaptorXMLException
       {
           com.altova.raptorxml.XMLValidator xmlValidator =
rxml.getXMLValidator();
           System.out.println("RaptorXML Java - XML validation");
           xmlValidator.setInputFromText( "<!DOCTYPE root [ <!ELEMENT root
(#PCDATA)> ]> <root>simple input document</root>" );
              if( xmlValidator.isWellFormed() )
                  System.out.println( "The input string is well-formed" );
              else
                  System.out.println( "Input string is not well-formed: " +
xmlValidator.getLastErrorMessage() );

              if( xmlValidator.isValid() )
                  System.out.println( "The input string is valid" );
              else
                  System.out.println( "Input string is not valid: " +
xmlValidator.getLastErrorMessage() );
       }
```

```
        static void RunXSLT() throws com.altova.raptorxml.RaptorXMLException
        {
            System.out.println("RaptorXML Java - XSL Transformation");
            com.altova.raptorxml.XSLT xsltEngine = rxml.getXSLT();
            xsltEngine.setInputXMLFileName( strExamplesFolder + "simple.xml" );
            xsltEngine.setXSLFileName( strExamplesFolder + "transform.xsl" );
            String result = xsltEngine.executeAndGetResultAsString();
            if( result == null )
                System.out.println( "Transformation failed: " +
xsltEngine.getLastErrorMessage() );
            else
                System.out.println( "Result is " + result );
        }




        static void RunXQuery() throws com.altova.raptorxml.RaptorXMLException
        {
            System.out.println("RaptorXML Java - XQuery execution");
            com.altova.raptorxml.XQuery xqEngine = rxml.getXQuery();
            xqEngine.setInputXMLFileName( strExamplesFolder + "simple.xml" );
            xqEngine.setXQueryFileName( strExamplesFolder + "CopyInput.xq" );
            System result = xqEngine.executeAndGetResultAsString();
            if( result == null )
                System.out.println( "Execution failed: " +
xqEngine.getLastErrorMessage() );
            else
                System.out.println( "Result is " + result );
        }




        public static void main(String[] args)
        {
            try
            {
                rxml = com.altova.raptorxml.RaptorXML.getFactory();
                rxml.setErrorLimit( 3 );

                ValidateXML();
                RunXSLT();
                RunXQuery();
            }

            catch( com.altova.raptorxml.RaptorXMLException e )
            {
                e.printStackTrace();
            }

        }
```

```
}
```


© 2018 Altova GmbH

6.2 Interfaces de RaptorXML para Java

A continuación resumimos las interfaces Java de la API de RaptorXML. Para más información consulte los apartados correspondientes.

- RaptorXMLFactory
  Crea una instancia de objeto COM de RaptorXML nueva por medio de una llamada nativa y ofrece acceso a los motores de RaptorXML.

- XMLValidator
  Interfaz para el motor de validación XML.

- XSLT
  Interfaz para los motores XSLT.

- XQuery
  Interfaz para los motores XQuery.

- XBRL
  Interfaz para el motor XBRL.

- RaptorXMLException
  Interfaz para el método `RaptorXMLException`.

6.2.1 RaptorXMLFactory

`public interface` **`RaptorXMLFactory`**

Utilice `RaptorXMLFactory()` para crear una instancia de objeto COM de RaptorXML nueva. Esto ofrece acceso a los motores RaptorXML. La relación entre `RaptorXMLFactory` y el objeto COM de RaptorXML es una relación entre iguales. Esto significa que las siguientes llamadas a la función `get<ENGINENAME>()` devolverán interfaces para la misma instancia del motor. A continuación describimos los métodos de esta interfaz, agrupados según su función. Las enumeraciones de la interfaz **`RaptorXMLFactory`** se describen en otro apartado.

Motores

Métodos para llamar a los correspondientes motores.

- getXBRL

  `public XBRL getXBRL`
  Recupera el motor XBRL y devuelve una instancia XBRL nueva de esta interfaz `RaptorXMLFactory`.

- getXMLValidator

public XMLValidator getXMLValidator
Recupera el motor XMLValidator y devuelve una instancia `XMLValidator` nueva de esta interfaz `RaptorXMLFactory`.

- getXQuery

public XQuery getXQuery
Recupera el motor XQuery y devuelve una instancia `XQuery` nueva de esta interfaz `RaptorXMLFactory`.

- getXSLT

public XSLT getXSLT
Recupera el motor XSLT y devuelve una instancia `XSLT` nueva de esta interfaz `RaptorXMLFactory`.

Errores y advertencias

Métodos para establecer parámetros para errores y advertencias.

- setErrorFormat

public void setErrorFormat(*ENUMErrorFormat* format)
Establece el formato de error de RaptorXML en uno de los literales de `ENUMErrorFormat` (Text, ShortXML, LongXML).

- setErrorLimit

public void setErrorLimit(*int* limit)
Establece el límite error de validación de RaptorXML. El parámetro `limit` es de tipo `int` (Java), `uint` (COM/.NET). Especifica el número de errores que se deben notificar antes de detener la ejecución. Use el valor `-1` para que `limit` sea ilimitado (es decir, que se notifiquen todos los errores). El valor predeterminado es `100`.

- setReportOptionalWarnings

public void setReportOptionalWarnings(*boolean* report)
Habilita o deshabilita la notificación de advertencias. El valor `true` habilita las advertencias y `false` las deshabilita.

Catálogos y recursos globales

Métodos que ofrecen la ubicación de los archivos de catálogo que se deben usar.

- setGlobalCatalog

public void setGlobalCatalog(*String* catalog)

 © 2018 Altova GmbH

Establece la ubicación, en forma de URL, del archivo de catálogo principal (de punto de entrada). La cadena dada debe ser una URL absoluta que indique la ubicación exacta del archivo de catálogo principal que se debe usar.

- setUserCatalog

```
public void setUserCatalog(String catalog)
```

Establece la ubicación, en forma de URL, del archivo de catálogo del usuario personal. La cadena dada debe ser una URL absoluta que indique la ubicación exacta del archivo de catálogo personal que se debe usar.

- setGlobalResourceConfig

```
public void setGlobalResourceConfig(String config)
```

Establece la configuración activa del recurso global. El parámetro `config` es de tipo `String` y especifica el nombre de la configuración utilizada por el recurso global activo.

- setGlobalResourceFile

```
public void setGlobalResourceFile(String file)
```

Establece la ubicación, en forma de URL, del archivo XML de recursos globales. La cadena dada debe ser una URL absoluta que indique la ubicación exacta del archivo XML de recursos globales.

Opciones de configuración de servidores HTTP

Métodos para establecer el nombre y puerto del servidor HTTP y especificar el archivo de configuración del servidor.

- setServerFile

```
public void setServerFile(String file)
```

Establece la ubicación del archivo de configuración del servidor HTTP en relación a la dirección del servidor HTTP. Si se produce un error, se emite una excepción **`RaptorXMLException`** (*Java*). El parámetro de entrada es una cadena que indica la dirección del archivo de configuración del servidor HTTP en relación a la dirección del servidor.

- setServerName

```
public void setServerName(String name)
```

Establece el nombre del servidor HTTP. Si se produce un error, se emite una excepción **`RaptorXMLException`** (*Java*). El parámetro de entrada es una cadena que indica el nombre del servidor HTTP.

- setServerPort

```
public void setServerPort(int port)
```

Establece el puerto del servidor HTTP por el que se accede al servicio. El puerto debe ser fijo y conocido para que las solicitudes HTTP se puedan dirigir correctamente al servicio. Si se produce un error, se emite una excepción **RaptorXMLException** (*Java*). El parámetro de entrada es un entero que especifica el puerto de acceso del servidor HTTP.

Información sobre el producto

Métodos para recuperar información sobre el producto instalado.

getProductName

```
public string getProductName()
```

Devuelve el nombre del producto en forma de cadena. *Ejemplo:* para `Altova RaptorXML +XBRL Server 2018r2sp1(x64)`, devolvería `Altova RaptorXML+XBRL Server`. Si se produce un error, se emite la excepción **RaptorXMLException** (*Java*).

getProductNameAndVersion

```
public string getProductNameAndVersion()
```

Devuelve la versión de service pack del producto en forma de entero. *Ejemplo:* para `Altova RaptorXML+XBRL Server 2018r2sp1(x64)`, devolvería `Altova RaptorXML+XBRL Server 2018r2sp1(x64)`. Si se produce un error, se emite la excepción **RaptorXMLException** (*Java*).

getMajorVersion

```
public int getMajorVersion()
```

Devuelve la versión principal del producto en forma de entero. *Ejemplo:* para `Altova RaptorXML+XBRL Server 2014r2sp1(x64)`, devolvería `16` (la diferencia entre la versión principal (`2014`) y el año inicial `1998`). Si se produce un error, se emite la excepción **RaptorXMLException** (*Java*).

getMinorVersion

```
public int getMinorVersion()
```

Devuelve la versión secundaria del producto en forma de entero. *Ejemplo:* para `Altova RaptorXML+XBRL Server 2018r2sp1(x64)`, devolvería `2` (por el número de versión secundaria `r2`). Si se produce un error, se emite la excepción **RaptorXMLException** (*Java*).

getServicePackVersion

```
public int getServicePackVersion()
```

Devuelve la versión de service pack del producto como entero. *Ejemplo*: para `RaptorXML +XBRL Server 2018r2sp1(x64)`, devolvería `1` (por el número de versión de service pack `sp1`). Si se produce un error, se emite una excepción **RaptorXMLException** (*Java*).

 © 2018 Altova GmbH

- is64Bit

  public boolean is64Bit()
  Comprueba si la aplicación es un archivo ejecutable de 64 bits. Devuelve el valor `true` si se trata de una aplicación de 64 bits y el valor `false` si no lo es. *Ejemplo:* para `Altova RaptorXML+XBRL Server 2018r2sp1(x64)`, devolverá `true`. Si se produce un error, se emite una excepción **`RaptorXMLException`** (*detalles de la interfaz Java*).

- getAPIMajorVersion

  public int getAPIMajorVersion()
  Devuelve la versión principal de la API en forma de entero. La versión principal de la API puede diferir de la versión principal del producto si la API está conectada a otro servidor.

- getAPIMinorVersion

  public int getAPIMinorVersion()
  Devuelve la versión secundaria de la API en forma de entero. La versión secundaria de la API puede diferir de la versión secundaria del producto si la API está conectada a otro servidor.

- getAPIServicePackVersion

  public int getAPIServicePackVersion()
  Devuelve la versión de service pack de la API en forma de entero. La versión de service pack de la API puede diferir de la versión de service pack del producto si la API está conectada a otro servidor.

6.2.2 XMLValidator

`public interface` **`XMLValidator`**

Valida el documento XML, esquema XML o DTD suministrados. La validación de documentos XML se puede hacer con DTD o esquemas XML internos o externos. También comprueba si los documentos tienen un formato XML correcto. A continuación describimos los métodos de esta interfaz, agrupados según su función.

Procesamiento

Métodos para especificar los parámetros de una validación y para recuperar información sobre la validación.

- extractAvroSchema

  public boolean extractAvroSchema(*String* **outputPath)**
  Extrae un esquema Avro de un archivo binario. El parámetro `outputPath` es una URL absoluta que especifica la ubicación de salida. El resultado es `true` si la extracción finaliza correctamente. De lo contrario, el resultado es `false`. Si se produce un error, se emite una

excepción **RaptorXMLException** (*detalle de la interfaz Java*). Use getLastErrorMessage (***en Java***) o LastErrorMessage (***en COM/.NET***) para obtener más información.

isValid(ENUM type)

```
public boolean isValid(ENUMValidationType type)
```

Devuelve el resultado de la validación del documento XML, documento de esquema o documento DTD. El tipo de documento que se debe validar viene dado por el parámetro type, que toma como valor un literal de la enumeración ENUMValidationType (Java, COM/.NET). Si el documento es válido, el resultado es true. Si no lo es, el resultado es false. Si se produce un error, se emite una excepción **RaptorXMLException** (*detalles de la interfaz Java*). Use getLastErrorMessage (***en Java***) o LastErrorMessage (***en COM/.NET***) para obtener más información.

isValid

```
public boolean isValid()
```

Devuelve el resultado de la validación del documento dado. Si el documento es correcto, devuelve true. Si no lo es, devuelve false.

isWellFormed(ENUM type)

```
public boolean isWellFormed(ENUMWellformedCheckType type)
```

Devuelve el resultado de la comprobación de formato del documento XML o DTD. El tipo de documento que se debe revisar se especifica con el parámetro type, que toma como valor un literal de la enumeración ENUMWellformedCheckType (Java, COM/.NET). Si el documento tiene el formato correcto, el resultado es true. De lo contrario devuelve false. Si se produce un error, se emite la excepción **RaptorXMLException** (*detalles de la interfaz Java*). Use getLastErrorMessage (***en Java***) o LastErrorMessage (***en COM/.NET***) para obtener más información.

isWellFormed

```
public boolean isWellFormed()
```

Devuelve el resultado de la comprobación de formato del documento XML o DTD. Si el documento tiene el formato correcto, el resultado es true. De lo contrario devuelve false.

getLastErrorMessage

```
public String getLastErrorMessage()
```

Recupera una cadena que es el último mensaje de error del motor de RaptorXML.

setAssessmentMode

```
public void setAssessmentMode(ENUMAssessmentMode mode)
```

Establece el modo de evaluación de la validación XML (Strict/Lax), que viene dado por un literal de ENUMAssessmentMode (Java, COM/.NET).

 © 2018 Altova GmbH

setPythonScriptFile

```
public void setPythonScriptFile(String file)
```

Establece la ubicación del archivo de script Python. La cadena dada debe ser una URL absoluta que indique la ubicación exacta del archivo Python.

setStreaming

```
public void setStreaming(boolean support)
```

Habilita la validación por transmisión por secuencias. En el modo de transmisión por secuencias, los datos que están almacenados en memoria se minimizan y el procesamiento es más rápido. El valor `true` habilita la transmisión por secuencias y `false` la deshabilita. El valor predeterminado es `true`.

Archivos de entrada

Métodos para especificar los archivos de entrada de un comando de validación (XML, XML Schema y DTD).

setAvroSchemaFileName

```
public void setAvroSchemaFileName(String filePath)
```

Establece la ubicación, en forma de URL, del esquema Avro externo que se debe usar. La cadena dada debe ser una URL absoluta que indique la ubicación exacta del archivo de esquema Avro.

setAvroSchemaFromText

```
public void setAvroSchemaFromText(String schemaText)
```

Aporta el contenido del documento de esquema Avro que se debe usar. La cadena dada es el contenido del documento de esquema Avro que se debe usar.

setInputFileName

```
public void setInputFileName(String filePath)
```

Establece la ubicación, en forma de URL, de los datos XML de entrada que se deben procesar. La cadena dada debe ser una URL absoluta que indique la ubicación exacta del archivo de entrada.

setInputFileCollection

```
public void setInputFileCollection(Collection<?> fileCollection)
```

Aporta la colección de archivos XML que se deben usar como datos de entrada. Los archivos se identifican por medio de direcciones URL. Se aporta una colección de cadenas y cada cadena es la URL absoluta de un archivo XML de entrada.

setInputFromText

```
public void setInputFromText(String inputXMLText)
```
Aporta el contenido del documento XML que se debe procesar. La cadena dada es el contenido del documento XML que se debe procesar.

- setInputTextCollection

```
public void setInputTextCollection(Collection<?> stringCollection)
```
Aporta el contenido de varios archivos XML que se usarán como datos de entrada. Una colección de cadenas, cada una de las cuales es el contenido de un archivo XML de entrada.

- setInputXMLFileCollection (DEPRECATED. Use setInputFileCollection instead.)

```
public void setInputXMLFileCollection(Collection<?> fileCollection)
```
Aporta la colección de archivos XML que se usarán como datos de entrada. Los archivos se identifican por medio de una dirección URL. Una cadena de cadenas, cada una de las cuales es la URL absoluta de un archivo XML de entrada.

- setInputXMLFileName (DEPRECATED. Use setInputFileName instead.)

```
public void setInputXMLFileName(String xmlFile)
```
Establece la ubicación, en forma de URL, del documento XML que se debe procesar. La cadena dada debe ser una URL absoluta que indique la ubicación exacta del archivo XML.

- setInputXMLFromText (DEPRECATED. Use setInputFromText instead.)

```
public void setInputXMLFromText(String inputXMLText)
```
Aporta el contenido del documento XML que se debe procesar. La cadena dada es el contenido del documento XML que se debe procesar.

- setInputXMLTextCollection(DEPRECATED. Use setInputTextCollection instead.)

```
public void setInputXMLTextCollection(Collection<?> stringCollection)
```
Aporta el contenido de varios archivos XML que se usarán como datos de entrada. Una colección de cadenas, cada una de las cuales es el contenido de un archivo XML de entrada.

- setSchemaFileCollection

```
public void setSchemaFileCollection(Collection<?> fileCollection)
```
Aporta la colección de archivos de esquema XML que se usarán como esquemas XML externos. Los archivos se identifican por medio de su dirección URL. La entrada es una colección de cadenas, cada una de las cuales es la URL absoluta de un archivo de esquema XML.

- setSchemaFileName

```
public void setSchemaFileName(String filePath)
```

Establece la ubicación, en forma de URL, del documento de esquema XML que se debe validar. La cadena dada debe ser una URL absoluta que indique la ubicación exacta del archivo de esquema XML.

- setSchemaFromText

```
public void setSchemaFromText(String schemaText)
```

Aporta el contenido del documento de esquema XML que se debe usar. La cadena dada es el contenido del documento de esquema XML que se debe usar.

- setSchemaTextCollection

```
public void setSchemaTextCollection(Collection<?> stringCollection)
```

Aporta el contenido de varios archivos de esquema XML. La entrada es una colección de cadenas, cada una de las cuales es el contenido del documento de esquema XML.

- setDTDFileName

```
public void setDTDFileName(String filePath)
```

Establece la ubicación, en forma de URL, del documento DTD que se debe usar para la validación. La cadena dada debe ser una URL absoluta que indique la ubicación exacta del documento DTD.

- setDTDFromText

```
public void setDTDFromText(String dtdText)
```

Aporta el contenido del documento DTD que se debe usar para la validación. La cadena dada es el contenido del documento DTD que se debe usar.

XML Schema

Métodos para establecer las opciones del esquema XML que se debe utilizar para la validación.

- setSchemaImports

```
public void setSchemaImports(ENUMSchemaImports opt)
```

Especifica cómo se deben realizar las importaciones de esquema, en función de los valores de atributo de los elementos `xs:import`. El tipo de importación se especifica con el literal de `ENUMSchemaImports` (Java, COM/.NET) que se aporta.

- setSchemalocationHints

```
public void setSchemalocationHints(ENUMLoadSchemalocation opt)
```

Especifica el mecanismo que se debe utilizar para encontrar el esquema. El mecanismo se especifica por medio del literal de `ENUMLoadSchemalocation` (Java, COM/.NET) que se seleccione.

- setSchemaMapping

  public void setSchemaMapping(*ENUMSchemaMapping* opt)
  Establece qué asignación se debe usar para encontrar el esquema. La asignación se especifica por medio del literal de `ENUMSchemaMapping` (Java, COM/.NET) que se seleccione.

- setXSDVersion

  public void setXSDVersion(*ENUMXSDVersion* version)
  Establece la versión de XML Schema con la que se debe validar el documento XML. Es un literal de la enumeración `ENUMXSDVersion` (Java, COM/.NET).

XML

Métodos para especificar opciones relacionadas con los datos XML de entrada.

- setEnableNamespaces

  public void setEnableNamespaces(*boolean* enable)
  Habilita el procesamiento preparado para espacios de nombres. Esto ayuda en la revisión de instancias XML para buscar errores debidos a espacios de nombres incorrectos. El valor `true` habilita un procesamiento preparado para espacios de nombres. El valor `false` lo deshabilita. El valor predeterminado es `false`.

- setXincludeSupport

  public void setXincludeSupport(*boolean* support)
  Habilita o deshabilita el uso de los elementos `XInclude`. El valor `true` habilita la compatibilidad con XInclude y `false` la deshabilita. El valor predeterminado es `false`.

- setXMLValidationMode

  public void setXMLValidationMode(*ENUMXMLValidationMode* mode)
  Establece el modo de validación XML, que es un literal de la enumeración `ENUMXMLValidationMode` (Java, COM/.NET) que determina si se debe comprobar la validez del documento o revisar su formato.

6.2.3 XSLT

`public interface` **XSLT**

Transforma datos XML usando el documento XSLT 1.0, 2.0 o 3.0 dado. Los documentos XML y XSLT se pueden aportar en forma de archivo (con una URL) o de cadena de texto. El resultado se

devuelve en forma de archivo (en la ubicación de destino dada) o como cadena de texto. Puede indicar parámetros XSLT y habilitar funciones de extensión de Altova para un procesamiento avanzado (p. ej. generación de gráficos). El documento XSLT también se puede validar. Cuando las cadenas de texto de entrada se deben interpretar como URL, deben utilizarse rutas de acceso absolutas.

A continuación describimos los métodos de esta interfaz, agrupados según su función.

Procesamiento

Métodos para especificar los parámetros de la transformación XSLT.

- isValid

  public boolean isValid()
  Devuelve el resultado de validar el documento XSLT de acuerdo con la especificación XSLT indicada en ENUMXSLTVersion (véase el método setVersion). El resultado es `true` si el documento es válido y `false` si no lo es. Si ocurre un error, se emite una excepción RaptorXMLException. Use el método getLastErrorMessage para obtener más información.

- execute

  public boolean execute(*String* outputFile)
  Ejecuta la transformación XSLT de acuerdo con la especificación XSLT indicada en ENUMXSLTVersion (véase el método setVersion) y guarda el resultado en el archivo de salida indicado en el parámetro `outputFile`. Si ocurre un error, se emite una excepción RaptorXMLException. Use el método getLastErrorMessage para obtener más información.

- executeAndGetResultAsString

  public String executeAndGetResultAsString()
  Ejecuta la transformación XSLT de acuerdo con la especificación XSLT indicada en ENUMXSLTVersion (véase el método setVersion) y devuelve el resultado en forma de cadena de texto. Este método no produce archivos de resultados adicionales como gráficos ni resultados secundarios. Tampoco almacena resultados binarios como archivos OOXML. Si necesita más archivos de salida, utilice el método execute. Si ocurre un error, se emite una excepción RaptorXMLException. Use el método getLastErrorMessage para obtener más información.

- executeAndGetResultAsStringWithBaseOutputURI

  public String executeAndGetResultAsStringWithBaseOutputURI(*String* baseURI)
  Ejecuta la transformación XSLT de acuerdo con la especificación XSLT indicada en ENUMXSLTVersion (véase el método setVersion) y devuelve el resultado en forma de cadena de texto en la ubicación definida por el URI base. El parámetro **baseURI** es una cadena que aporta un URI. Este método no produce archivos de resultados adicionales como gráficos ni resultados secundarios. Tampoco almacena resultados binarios como archivos OOXML.Si necesita más archivos de salida, utilice el método execute. Si ocurre un error, se emite una excepción RaptorXMLException. Use el método getLastErrorMessage para obtener más información.

- getMainOutput

```
public String getMainOutput()
```

Devuelve el resultado principal del último trabajo que se ejecutó como cadena de texto.

- getAdditionalOutputs

```
public String getAdditionalOutputs()
```

Devuelve resultados adicionales del último trabajo que se ejecutó.

- getLastErrorMessage

```
public String getLastErrorMessage()
```

Recupera una cadena que es el último mensaje de error del motor de RaptorXML.

- setIndentCharacters

```
public void setIndentCharacters(String chars)
```

Aporta la cadena de caracteres que se usará para aplicar sangría en el documento de salida.

- setStreamingSerialization

```
public void setStreamingSerialization(boolean support)
```

Habilita la serialización de secuencias de datos. En el modo de transmisión por secuencias, los datos almacenados en memoria se minimizan y el procesamiento es más rápido. El valor `true` habilita la serialización de secuencias de datos; `false` la deshabilita.

XSLT

Métodos para especificar opciones relacionadas con la hoja de estilos XSLT.

- setVersion

```
public void setVersion(ENUMXSLTVersion version)
```

Establece la versión XSLT que se debe usar para el procesamiento (validación o transformación XSLT).
Parámetros: `version`: almacena un literal de `EnumXSLTVersion`: `eVersion10`, `eVersion20` o `eVersion30`.

- setXSLFileName

```
public void setXSLFileName(String xslFile)
```

Establece la ubicación en forma de URL del documento XSLT que se debe usar para la transformación.
Parámetros: `xslFile`: la cadena indicada debe ser una URL absoluta que aporta la ubicación exacta del archivo XSLT.

 © 2018 Altova GmbH

- setXSLFromText

```
public void setXSLFromText(String xslText)
```

Aporta el contenido del documento XSLT como texto.
Parámetros: `xslText`: la cadena indicada es el documento XSLT que se debe usar para la transformación.

- addExternalParameter

```
public void addExternalParameter(String name, String value)
```

Añade el nombre y valor de un parámetro externo nuevo. Cada parámetro externo y su valor debe especificarse en una llamada distinta al método. Los parámetros deben declararse en el documento XML. Como los valores de parámetros son expresiones XPath, los valores de parámetros que sean cadenas deben ir entre comillas simples.
Parámetros: `name`: almacena el nombre del parámetro, que es un QName, como cadena de texto. `value`: almacena el valor del parámetro como cadena de texto.

- clearExternalParameterList

```
public void clearExternalVariableList()
```

Borra la lista de parámetros externos creada con el método `AddExternalParameter`.

- setInitialTemplateMode

```
public void setInitialTemplateMode(String mode)
```

Establece el nombre del modo de plantilla inicial. El procesamiento comenzará a partir de las plantillas que tengan este valor de modo. La transformación debe iniciarse después de asignar los documentos XML y XSLT.
Parámetros: `mode`: el nombre del modo de plantilla inicial, en forma de cadena de texto.

- setNamedTemplateEntryPoint

```
public void setNamedTemplateEntryPoint(boolean template)
```

Aporta el nombre de la plantilla con nombre con la que se debe iniciar el procesamiento.
Parámetros: `template`: el nombre de la plantilla con nombre, en forma de cadena de texto.

Esquema XML

Métodos para establecer opciones para el esquema XML que se debe usar para la validación.

- setSchemaImports

```
public void setSchemaImports(ENUMSchemaImports opt)
```

Especifica cómo se deben realizar las importaciones de esquema, en función de los valores de atributo de los elementos `xs:import`. El tipo de importación se especifica con el literal de `ENUMSchemaImports` (Java, COM/.NET) que se aporta.

- setSchemalocationHints

  public void setSchemalocationHints(*ENUMLoadSchemalocation* opt)
  Especifica el mecanismo que se debe utilizar para encontrar el esquema. El mecanismo se especifica por medio del literal de `ENUMLoadSchemalocation` (Java, COM/.NET) que se seleccione.

- setSchemaMapping

  public void setSchemaMapping(*ENUMSchemaMapping* opt)
  Establece qué asignación se debe usar para encontrar el esquema. La asignación se especifica por medio del literal de `ENUMSchemaMapping` (Java, COM/.NET) que se seleccione.

- setXSDVersion

  public void setXSDVersion(*ENUMXSDVersion* version)
  Establece la versión de XML Schema con la que se debe validar el documento XML. Es un literal de la enumeración `ENUMXSDVersion` (Java, COM/.NET).

XML

Métodos para especificar parámetros relacionados con los datos XML que se deben procesar.

- setInputXMLFileName

  public void setInputXMLFileName(*String* xmlFile)
  Establece la ubicación, en forma de URL, del documento XML que se debe procesar. La cadena dada debe ser una URL absoluta que indique la ubicación exacta del archivo XML.

- setInputXMLFromText

  public void setInputXMLFromText(*String* inputXMLText)
  Aporta el contenido del documento XML que se debe procesar. La cadena dada es el contenido del documento XML que se debe procesar.

- setLoadXMLWithPSVI

  public void setLoadXMLWithPSVI(*boolean* psvi)
  Habilita la validación de archivos XML de entrada y genera información posterior a la validación con esquema sobre dichos archivos. El valor `true` habilita la validación XML y genera información posterior a la validación con esquema para los archivos XML. El valor `false` deshabilita la validación. El valor predeterminado es `true`.

- setXincludeSupport

  public void setXincludeSupport(*boolean* support)

 © 2018 Altova GmbH

Habilita o deshabilita el uso de los elementos `XInclude`. El valor `true` habilita la compatibilidad con XInclude y `false` la deshabilita. El valor predeterminado es `false`.

- setXMLValidationErrorAsWarning

  `public void setXMLValidationErrorAsWarning(boolean enable)`
  Establece el modo de validación XML, que es un literal de la enumeración `ENUMXMLValidationMode` (Java, COM/.NET) que determina si se debe comprobar la validez del documento o revisar su formato.

- setXMLValidationMode

  `public void setXMLValidationMode(ENUMXMLValidationMode mode)`
  Establece el modo de validación XML, que es un literal de la enumeración `ENUMXMLValidationMode` (Java, COM/.NET) que determina si se debe comprobar la validez del documento o revisar su formato.

Extensiones

Métodos para especificar si se deben habilitar o no las funciones de extensión de Altova para tareas de procesamiento especiales (p. ej. para trabajar con gráficos).

- setChartExtensionsEnabled

  `public void setChartExtensionsEnabled(boolean enable)`
  Habilita o deshabilita las funciones de extensión de Altova para gráficos. El valor `true` habilita las extensiones para gráficos. El valor `false` las deshabilita. El valor predeterminado es `true`.

- setDotNetExtensionsEnabled

  `public void setDotNetExtensionsEnabled(boolean enable)`
  Habilita o deshabilita las funciones de extensión .NET. El valor `true` habilita las extensiones .NET. El valor `false` las deshabilita. El valor predeterminado `true`.

- setJavaExtensionsEnabled

  `public void setJaveExtensionsEnabled(boolean enable)`
  Habilita o deshabilita las funciones de extensión Java. El valor `true` habilita las extensiones Java y `false` las deshabilita. El valor predeterminado es `true`.

- setJavaBarcodeExtensionLocation

  `public void setJavaBarcodeExtensionLocation(String path)`
  Especifica la ruta de acceso de la carpeta que contiene el archivo de extensión para códigos de barras `AltovaBarcodeExtension.jar`. Consulte el apartado Funciones de extensión de Altova: códigos de barras para obtener más información. La ruta de acceso debe darse con

uno de estos dos formatos:

- Con un URI de archivo (p. ej. `--javaext-barcode-location="file:///C:/Archivos de programa/Altova/RaptorXMLServer2015/etc/jar/"`)
- Con una ruta de acceso Windows con las barras diagonales inversas entre caracteres de escape (p. ej. `--javaext-barcode-location="C:\\Archivos de programa\\Altova\\RaptorXMLServer2015\\etc\\jar\\"`)

Parámetros:
`path`: la cadena dada debe ser una URL absoluta que indique la ubicación base del archivo que se debe utilizar.

6.2.4 XQuery

```
public interface XQuery
```

Ejecuta documentos XQuery 1.0 y 3.0 usando el motor de RaptorXML. Los documentos XQuery y XML pueden darse como archivos (por medio de una URL) o como cadenas de texto. El resultado se devuelve en forma de archivo (en la ubicación de destino indicada) o como cadena de texto. Puede indicar variables XQuery externas y habilitar diferentes opciones de serialización. El documento XQuery también se puede validar. Cuando las cadenas de texto de entrada se deben interpretar como URL, deben utilizarse rutas de acceso absolutas.

A continuación describimos los métodos de esta interfaz, agrupados según su función.

Procesamiento

Métodos para especificar parámetros de la ejecución XQuery.

isValid

```
public boolean isValid()
```

Devuelve el resultado de validar el documento XQuery de acuerdo con la especificación XQuery indicada en `ENUMXQueryVersion` (véase el método `setVersion`). El resultado es `true` si el documento es válido y `false` si no lo es. Si ocurre un error, se emite una excepción `RaptorXMLException`. Use el método `getLastErrorMessage` para obtener más información.

isValidUpdate

```
public boolean isValidUpdate()
```

Devuelve el resultado de validar el documento XQuery Update de acuerdo con la especificación XQuery Update indicada en `ENUMXQueryVersion` (véase el método `setVersion`). El resultado es `true` si el documento es válido y `false` si no lo es. Si ocurre un error, se emite una excepción `RaptorXMLException`. Use el método `getLastErrorMessage` para obtener más información.

 © 2018 Altova GmbH

- execute

  public boolean execute(*String* outputFile)
  Ejecuta la transformación XQuery según la versión XQuery nombrada en ENUMXQueryVersion (véase el método setVersion) y guarda el resultado en el archivo de salida nombrado en el parámetro outputFile. El parámetro es una cadena que aporta la ubicación (ruta de acceso y nombre de archivo) del archivo de salida. El resultado es true si la transformación se ejecuta correctamente y false si no es así. Si se producen errores, se emite la excepción RaptorXMLException. Use el método getLastErrorMessage para obtener más información.

- executeAndGetResultAsString

  public String executeAndGetResultAsString()
  Ejecuta la transformación XSLT de acuerdo con la especificación XSLT nombrada en ENUMXSLTVersion (véase el método setVersion) y devuelve el resultado en forma de cadena de texto. Este método no produce archivos de resultados adicionales como gráficos ni resultados secundarios. Tampoco almacena resultados binarios como archivos OOXML. Si necesita más archivos de salida, utilice el método execute.

- executeUpdate

  public boolean executeUpdate(*String* outputFile)
  Ejecuta la actualización XQuery de acuerdo con la especificación XQuery Update nombrada en ENUMXQueryVersion (véase el método setVersion) y guarda el resultado en el archivo de salida nombrado en el parámetro outputFile. El parámetro es una cadena que aporta la ubicación (ruta de acceso y nombre de archivo) del archivo de salida. El resultado es true si la actualización se ejecuta correctamente y false si no es así. Si se producen errores, se emite la excepción RaptorXMLException. Use el método getLastErrorMessage para obtener más información.

- executeUpdateAndGetResultAsString

  public String executeUpdateAndGetResultAsString()
  Ejecuta la actualización XQuery de acuerdo con la especificación XQuery Update nombrada en ENUMXQueryVersion (véase el método setVersion)y devuelve el resultado en forma de cadena de texto. Este método no produce archivos de resultados adicionales como gráficos ni resultados secundarios. Tampoco almacena resultados binarios como archivos OOXML. Si necesita más archivos de salida, utilice el método execute.

- getLastErrorMessage

  public String getLastErrorMessage()
  Recupera una cadena que es el último mensaje de error del motor de RaptorXML.

- setUpdatedXMLWriteMode

  public void setUpdatedXMLWriteMode(*ENUMXQueryUpdatedXML* mode)
  Establece el modo que se debe usar para la actualización. Toma un literal de la enumeración ENUMXQueryUpdatedXML: eUpdatedDiscard, eUpdatedWriteback o

eUpdatedAsMainResult

XML

Métodos para especificar parámetros relacionados con los datos XML que se deben procesar.

setInputXMLFileName

```
public void setInputXMLFileName(String xmlFile)
```

Establece la ubicación, en forma de URL, del documento XML que se debe procesar. La cadena dada debe ser una URL absoluta que indique la ubicación exacta del archivo XML.

setInputXMLFromText

```
public void setInputXMLFromText(String inputXMLText)
```

Aporta el contenido del documento XML que se debe procesar. La cadena dada es el contenido del documento XML que se debe procesar.

setLoadXMLWithPSVI

```
public void setLoadXMLWithPSVI(boolean psvi)
```

Habilita la validación de archivos XML de entrada y genera información posterior a la validación con esquema sobre dichos archivos. El valor `true` habilita la validación XML y genera información posterior a la validación con esquema para los archivos XML. El valor `false` deshabilita la validación. El valor predeterminado es `true`.

setXincludeSupport

```
public void setXincludeSupport(boolean support)
```

Habilita o deshabilita el uso de los elementos `XInclude`. El valor `true` habilita la compatibilidad con XInclude y `false` la deshabilita. El valor predeterminado es `false`.

setXMLValidationErrorAsWarning

```
public void setXMLValidationErrorAsWarning(boolean enable)
```

Establece el modo de validación XML, que es un literal de la enumeración `ENUMXMLValidationMode` (Java, COM/.NET) que determina si se debe comprobar la validez del documento o revisar su formato.

setXMLValidationMode

```
public void setXMLValidationMode(ENUMXMLValidationMode mode)
```

Establece el modo de validación XML, que es un literal de la enumeración `ENUMXMLValidationMode` (Java, COM/.NET) que determina si se debe comprobar la validez del documento o revisar su formato.

 © 2018 Altova GmbH

- setXSDVersion

  public void setXSDVersion(*ENUMXSDVersion* version)
  Establece la versión de XML Schema con la que se debe validar el documento XML. Es un literal de la enumeración `ENUMXSDVersion` (Java, COM/.NET).

XQuery

Métodos para especificar opciones relacionadas con el documento XQuery.

- setVersion

  public void setVersion(*ENUMXQueryVersion* version)
  Establece la versión de XQuery que se debe usar para el procesamiento (validación o ejecución de XQuery). Toma un literal de la enumeración `EnumXQueryVersion`: `eVersion10` o `eVersion30`. El valor predeterminado es `eVersion30ml`.

- setXQueryFileName

  public void setXQueryFileName(*String* queryFile)
  Establece la ubicación en forma de URL del archivo XQuery que debe ejecutarse. La cadena indicada debe ser una URL absoluta que dé la ubicación exacta del archivo XML que se debe usar.

- setXQueryFromText

  public void setXQueryFromText(*String* queryText)
  Aporta el contenido del documento XQuery en forma de texto. La cadena dada es el documento XQuery que se debe procesar.

- addExternalVariable

  public void addExternalVariable(*String* name, *String* value)
  Añade el nombre y valor de una variable externa nueva. Cada variable externa y su valor se debe especificar en una llama distinta al método. Las variables se deben declarar en el documento XQuery (y la declaración de tipo es opcional). Si el valor de la variable es una cadena de texto, ponga el valor entre comillas simples. El parámetro **`name`** almacena el nombre de la variable, que es un QName, como cadena. El parámetro **`value`** almacena el valor de la variable como cadena.

- clearExternalVariableList

  public void clearExternalVariableList()
  Borra la lista de variables externas creadas con el método `AddExternalVariable`.

Opciones de serialización

Métodos para especificar propiedades del resultado del procesamiento.

- setIndentCharacters

  public void setIndentCharacters(*String* chars)

  Aporta la cadena de caracteres que se usará para aplicar sangría en el documento de salida.

- setKeepFormatting

  public void setKeepFormatting(*boolean* keep)

  Habilita o deshabilita la opción de conservar el formato original de los archivos que se actualizarán con `executeUpdate`. El parámetro **keep** toma el valor booleano `true` o `false`.

- setOutputEncoding

  public void setOutputEncoding(*String* encoding)

  Establece la codificación del documento de salida. Use un nombre de codificación IANA oficial como `UTF-8`, `UTF-16`, `US-ASCII`, `ISO-8859-1` como cadena de texto.

- setOutputIndent

  public void setOutputIndent(*boolean* indent)

  Habilita o deshabilita la aplicación de sangría en el documento de salida. El valor `true` habilita la sangría; `false` la deshabilita.

- setOutputMethod

  public void setOutputMethod(*String* outputMethod)

  Especifica la serialización del documento de salida. Los valores válidos son `xml | xhtml | html | text`, dados como cadena de texto. El valor predeterminado es `xml`.

- setOutputOmitXMLDeclaration

  public void setOutputOmitXMLDeclaration(*boolean* omit)

  Habilita o deshabilita la inclusión de la declaración XML en el documento de salida. El valor `true` omite la declaración; `false` la incluye. El valor predeterminado es `false`.

Extensiones

Métodos para especificar si se deben habilitar o no las funciones de extensión de Altova para tareas de procesamiento especiales (p. ej. para trabajar con gráficos).

- setChartExtensionsEnabled

 © 2018 Altova GmbH

public void setChartExtensionsEnabled(*boolean* enable)
Habilita o deshabilita las funciones de extensión de Altova para gráficos. El valor `true` habilita las extensiones para gráficos. El valor `false` las deshabilita. El valor predeterminado es `true`.

- setDotNetExtensionsEnabled

public void setDotNetExtensionsEnabled(*boolean* enable)
Habilita o deshabilita las funciones de extensión .NET. El valor `true` habilita las extensiones .NET. El valor `false` las deshabilita. El valor predeterminado `true`.

- setJavaExtensionsEnabled

public void setJaveExtensionsEnabled(*boolean* enable)
Habilita o deshabilita las funciones de extensión Java. El valor `true` habilita las extensiones Java y `false` las deshabilita. El valor predeterminado es `true`.

6.2.5 XBRL

`public interface` **XBRL**

Valida el documento de instancia XBRL o de taxonomía XBRL dado. A continuación describimos los métodos de esta interfaz. Sus enumeraciones se describen en otro apartado.

Clase de utilidad

Se define una clase de utilidad para `FormulaParam`. Almacena dos miembros y un constructor.

- ParamValuePair

```
public class ParamValuePair
public class ParamValuePair
    {
        public String paramType;
        public String paramValue;
        public ParamValuePair( String type, String value )
        {
                        paramType = type;
                        paramValue = value;
        }
    };
```

Procesamiento

Métodos para especificar los parámetros de una validación y recuperar información sobre la validación.

- isValid(ENUM type)

  `public boolean isValid(ENUMValidationType type)`
  Devuelve el resultado de validar el documento de instancia XBRL o la taxonomía XBRL. El tipo de documento que se debe validar viene dado por el parámetro `type`, que toma como valor un literal de `ENUMValidationType`. El resultado es `true` si el documento es válido y `false` si no es válido. Si ocurre un error, se emite una excepción `RaptorXMLException`. Use el método `getLastErrorMessage` para obtener más información.

- isValid

  `public boolean isValid()`
  Devuelve el resultado de validar el documento XBRL indicado. El resultado es `true` si el documento es válido y `false` si no es válido.

- getLastErrorMessage

  `public String getLastErrorMessage()`
  Recupera una cadena que es el último mensaje de error del motor de RaptorXML.

- setEvaluateReferencedParametersOnly

  `public void setEvaluateReferencedParametersOnly(boolean bEnable)`
  Si se le da el valor `false`, obliga a evaluar todos los parámetros incluso si no se les hace referencia desde ninguna fórmula, aserción o tabla. Valor predeterminado: `true`.

- setParallelAssessment

  `public void setParallelAssessment(boolean support)`
  Habilita o deshabilita el uso de la evaluación en paralelo. El valor `true` habilita la evaluación en paralelo; `false` la deshabilita. El valor predeterminado: `false`.

- setPythonScriptFile

  `public void setPythonScriptFile(String file)`
  Establece la ubicación en forma de URL del archivo de script Python. La cadena indicada debe ser una URL absoluta que dé la ubicación exacta del archivo Python.

Archivos de entrada

Métodos para especificar la entrada de los comandos de validación.

- setInputFileName

  `public void setInputFileName(String filePath)`

 © 2018 Altova GmbH

Establece la ubicación en forma de URL del documento XBRL que se debe validar. La cadena indicada debe ser una URL absoluta que indica la ubicación exacta del archivo XBRL.

- setInputFileCollection

```
public void setInputFileCollection(Collection<?> fileCollection)
```

Suministra la colección de archivos XBRL que se usarán como datos de entrada. Los archivos se identifican con sus URL. El parámetro `fileCollection` es una colección de cadenas de texto, cada una de las cuales es la URL absoluta de un archivo XBRL de entrada.

- setInputFromText

```
public void setInputFromText(String schemaText)
```

Suministra el contenido del documento XBRL en forma de texto. La cadena indicada es el contenido del documento XBRL que se debe validar.

- setInputTextCollection

```
public void setInputTextCollection(Collection<?> stringCollection)
```

Suministra el contenido de varios archivos XBRL que se usará como datos de entrada. El parámetro `stringCollection` es una colección de cadenas de texto, cada una de las cuales es el contenido de un documento XBRL de entrada.

Fórmulas y aserciones

Métodos para especificar opciones para fórmulas y aserciones XBRL.

- addAssertionForProcessing

```
public void addAssertionForProcessing(String assertion)
```

Limita la ejecución de aserciones a la aserción dada. Llame varias veces a este método para especificar más de una aserción. La cadena dada almacena el nombre de la aserción. Use `##none` para no procesar ninguna aserción y `##all` para procesarlas todas.

- addAssertionSetForProcessing

```
public void addAssertionSetForProcessing(String assertionSet)
```

Limita la ejecución al conjunto de aserciones dado. Llame varias veces a este método para especificar más de un conjunto de aserciones. La cadena dada almacena el nombre del conjunto de aserciones. Use `##none` para no procesar ningún conjunto de aserciones y `##all` para procesarlos todos.

- addFormulaArrayParameter

```
public void addFormulaArrayParameter(String type, String name, Object[]
```

```
values)
```

Añade un parámetro matriz utilizado en el proceso de evaluación de fórmulas.
Parámetros:
`type`: una cadena que da el tipo de datos predeterminado de los valores de no pares dentro de los valores de matriz. Tipo predeterminado: `xs:string`.
`name`: una cadena que da el nombre del parámetro.
`values`: una matriz de valores y pares valor/tipo de datos.
Para más información y ejemplos de código consulte el apartado Parámetros de fórmulas XBRL.

addFormulaForProcessing

```
public void addFormulaForProcessing(String formula)
```

Limita la ejecución a la fórmula dada. Llame varias veces a este método para especificar más de una fórmula. La cadena dada almacena el nombre de la fórmula. Use `##none` para no procesar ninguna fórmula y `##all` para procesarlas todas.

addFormulaParameter (with NS)

```
public void addFormulaParameter(String type, String name, String value,
String namespace)
```

Añade un parámetro utilizado en el proceso de evaluación de fórmulas.
Parámetros:
`type`: una cadena que indica el tipo de datos del parámetro.
`name`: una cadena que indica el nombre del parámetro.
`value`: una cadena que indica el valor del parámetro.
`namespace`: una cadena que da el espacio de nombres del parámetro.
Nota: si usa este método, el espacio de nombres se pasa a `addFormulaParameterNamespace`.

addFormulaParameter

```
public void addFormulaParameter(String type, String name, String value)
```

Añade un parámetro utilizado en el proceso de evaluación de fórmulas.
Parámetros:
`type`: una cadena que indica el tipo de datos del parámetro.
`name`: una cadena que indica el nombre del parámetro.
`value`: una cadena que indica el valor del parámetro.

addFormulaParameterNamespace

```
public void addFormulaParameterNamespace(String prefix, String URI)
```

Define un espacio de nombres utilizado en los QName de nombres, tipos o valores de parámetro.
Parámetros:
`prefix`: el prefijo de espacio de nombres de los valores pasados a `addFormulaArrayParameter`.
`URI`: el URI de espacio de nombres.

 © 2018 Altova GmbH

clearFormulaParameterList

```
public void clearFormulaParameterList()
```

Borra la lista de parámetros de fórmula creados con el método `addFormulaParameter`.

readFormulaAssertions

```
public String readFormulaAssertions()
```

Recupera las aserciones de fórmula del archivo especificado. Devuelve una cadena que contiene las aserciones de fórmula.

readFormulaOutput

```
public String readFormulaOutput()
```

Evalúa las aserciones de fórmula del archivo especificado y devuelve el resultado de la evaluación. Devuelve una cadena que es la evaluación de las aserciones de fórmula.

setFormulaAssertionsAsXML

```
public void setFormulaAssertionsAsXML(boolean enable)
```

Habilita el formato XML del archivo de aserciones cuando RaptorXML+XBRL se ejecuta con las aserciones habilitadas. El valor `true` habilita el formato de salida XML; `false` genera resultados en formato JSON. El valor predeterminado es `false`.

setFormulaAssertionsOutput

```
public void setFormulaAssertionsOutput(String outputFile)
```

Establece la ubicación del archivo que contiene las aserciones de fórmulas recuperadas. La cadena dada almacena la ruta de acceso completa del archivo de salida.

setFormulaOutput

```
public void setFormulaOutput(String outputFile)
```

Establece la ubicación del archivo que contiene el resultado de la evaluación de fórmulas. La cadena dada almacena la ruta de acceso completa del archivo de salida.

evaluateFormula

```
public boolean evaluateFormula()
```

Devuelve el resultado de evaluar las fórmulas XBRL de un archivo de instancia XBRL. El resultado es `true` si la evaluación finaliza correctamente y `false` si no. Si ocurre un error, se emite una excepción `RaptorXMLException`. Use el método `getLastErrorMessage` para obtener más información.

setFormulaExtensionEnabled

```
public void setFormulaExtensionEnabled(boolean enable)
```

Habilita la validación con la extensión XBRL Formula. El valor `true` habilita esta opción; `false` la deshabilita. El valor predeterminado es `true`.

- setFormulaPreloadSchemas

  `public void setFormulaPreloadSchemas(boolean enable)`
  Determina si los esquemas de XBRL Formula se cargan previamente o no. El valor `true` defina la carga previa de estos esquemas; `false` no carga los esquemas. El valor predeterminado es `false`.

Tablas

Métodos para especificar opciones para tablas XBRL.

- addTableForProcessing

  `public void addTableForProcessing(String table)`
  Limita la generación de tablas a la tabla dada. Llame varias veces a este método para especificar más de una tabla. La cadena dada almacena el nombre de la tabla. Use `##none` para no procesar ninguna tabla y `##all` para procesarlas todas.

- generateTables

  `public boolean generateTables()`
  Evalúa las tablas XBRL de un archivo de instancia. El resultado es `true` si la evaluación es correcta y `false` si no lo es. Si ocurre un error, se genera `RaptorXMLException`. Use el método `getLastErrorMessage` para obtener más información.

- setTableEliminateEmptyRows

  `public void setTableEliminateEmptyRows(boolean enable)`
  Habilita la eliminación de columnas/filas vacías de tablas en el formato de salida HTML. El valor `true` la habilita; `false` la deshabilita.

- setTableExtensionEnabled

  `public void setTableExtensionEnabled(boolean enable)`
  Habilita las extensiones XBRL Table 1.0 para la validación. El valor `true` las habilita; `false` las deshabilita.

- setTableLinkbaseNamespace

  `public void setTableLinkbaseNamespace(String namespace)`
  Habilita la carga de bases de enlaces de tabla escritas con una especificación borrador previa. El parámetro `namespace` especifica la base de enlaces de tabla. La validación, la resolución y el diseño de la base de enlaces de tabla, sin embargo, se realiza siempre según la recomendación Table Linkbase 1.0 del 18 de marzo de 2014. Use `##detect` para habilitar la detección automática.
  Parámetros: `namespace`: se reconocen estos valores:
  `##detect`

 © 2018 Altova GmbH

```
http://xbrl.org/PWD/2013-05-17/table
http://xbrl.org/PWD/2013-08-28/table
http://xbrl.org/CR/2013-11-13/table
http://xbrl.org/PR/2013-12-18/table
http://xbrl.org/2014/table
```

- setTableOutput

  `public void setTableOutput(String outputFile)`
  Establece la ubicación del archivo que contiene el resultado de la generación de tablas. La cadena dada almacena la ruta de acceso completa del archivo de salida.

- setTableOutputFormat

  `public void setTableOutputAsXML(ENUMTableOutputFormat format)`
  Define el formato del archivo de tablas de salida. El formato será el valor de `ENUMTableOutputFormat`.

- setTablePreloadSchemas

  `public void setTablePreloadSchemas(boolean enable)`
  Habilita la carga previa de esquemas de la especificación XBRL Table 1.0. El valor `true` lo habilita; `false` lo deshabilita. Valor predeterminado: `false`.

XML y XML Schema

Métodos para establecer el nivel de compatibilidad con XInclude y opciones para XML Schema.

- setXincludeSupport

  `public void setXincludeSupport(boolean support)`
  Habilita o deshabilita el uso de los elementos `XInclude`. El valor `true` habilita la compatibilidad con XInclude y `false` la deshabilita. El valor predeterminado es `false`.

- setSchemaImports

  `public void setSchemaImports(ENUMSchemaImports opt)`
  Especifica cómo se deben realizar las importaciones de esquema, en función de los valores de atributo de los elementos `xs:import`. El tipo de importación se especifica con el literal de `ENUMSchemaImports` (Java, COM/.NET) que se aporta.

- setSchemalocationHints

  `public void setSchemalocationHints(ENUMLoadSchemalocation opt)`
  Especifica el mecanismo que se debe utilizar para encontrar el esquema. El mecanismo se especifica por medio del literal de `ENUMLoadSchemalocation` (Java, COM/.NET) que se seleccione.

- setSchemaMapping

  `public void setSchemaMapping(ENUMSchemaMapping opt)`
  Establece qué asignación se debe usar para encontrar el esquema. La asignación se especifica por medio del literal de `ENUMSchemaMapping` (Java, COM/.NET) que se seleccione.

Métodos generales

Métodos para especificar opciones para etiquetas y parámetros generales XBRL.

- addIXBRLTransformationRegistryLimit

  `public void addIXBRLTransformationRegistryLimit(String limit)`
  Limita los registros de transformación Inline XBRL disponibles a la versión especificada. El parámetro `limit` es la versión a la que se deben limitar los registros Inline XBRL disponibles.

- clearIXBRLTransformationRegistryLimit

  `public void clearIXBRLTransformationRegistryLimit()`
  Elimina cualquier límite especificado previamente para los registros de transformación Inline XBRL.

- enableIXBRLValidateTarget

  `public void enableIXBRLValidateTarget(boolean enable)`
  Habilita la validación XBRL del documento XBRL de salida que se genera. Toma el valor booleano `true` o `false`. Valor predeterminado: `true`.

- setConceptLabelLinkrole

  `public void setConceptLabelLinkrole(String labelLinkrole)`
  Especifica la función de vínculo extendido preferida a la hora de representar etiquetas de concepto. La cadena dada almacena la función de vínculo preferida.

- setConceptLabelRole

  `public void setConceptLabelRole(String labelRole)`
  Especifica la función de etiqueta preferida a la hora de representar etiquetas de concepto. La cadena dada almacena la función de etiqueta preferida. Valor predeterminado: `http://www.xbrl.org/2008/role/label`.

- setDimensionsExtensionEnabled

  `public void setDimensionExtensionEnabled(boolean bEnable)`
  Habilita la validación con la extensión XBRL Dimension. El valor `true` habilita esta opción;

 © 2018 Altova GmbH

`false` la deshabilita. El valor predeterminado es `true`.

setGenericLabelRole

```
public void setGenericLabelRole(String labelRole)
```

Especifica la función de etiqueta preferida a la hora de representar etiquetas genéricas. La cadena dada almacena la función de etiqueta preferida. Valor predeterminado: `http://www.xbrl.org/2008/role/label`.

setIXBRLOutput

```
public void setIXBRLOutput(String outputFile)
```

Si se especifica, el documento XBRL de salida que se genera se escribirá en esta ruta de acceso. La cadena dada almacena la URL absoluta de la ubicación de salida.

setIXBRLTreatAsDocumentSet

```
public void setIXBRLTreatAsDocumentSet(boolean enable)
```

Trata todas las entradas como un solo conjunto de documentos Inline XBRL. Toma el valor booleano `true` o `false`. Valor predeterminado: `false`.

setIXBRLUriTransformationStrategy

```
public void setIXBRLUriTransformationStrategy(ENUMIXBRLUriStrategy strategy)
```

Especifica cómo se deben transformar identificadores URI Inline XBRL en el documento XBRL que se genera. La enumeración seleccionada especifica qué estrategia se debe seguir. La enumeración predeterminada es: `eStrategyKeepRelative`.

setIXBRLVersion

```
public void setIXBRLVersion(ENUMIXBRLVersion version)
```

Especifica qué versión Inline XBRL se debe utilizar. El literal de enumeración seleccionado especifica qué versión se debe usar. El literal de enumeración predeterminado es `eVersionDetect`.

setLabelLang

```
public void setLabelLang(String labelLang)
```

Especifica el idioma de etiqueta preferido a la hora de representar etiquetas. La cadena dada almacena el idioma de etiqueta preferido. Predeterminado: `en`.

setPreloadSchemas

```
public void setPreloadSchemas(boolean preload)
```

Determina si los esquemas de XBRL 2.1 se cargan previamente o no. Toma el valor booleano `true` o `false`. El valor `true` defina la carga previa de estos esquemas; `false` no carga los esquemas. El valor predeterminado es `true`.

▼ setTreatXBRLInconsistenciesAsErrors

```
public void setTreatXBRLInconsistenciesAsErrors(boolean treat)
```

Si toma el valor `true`, la validación XBRL fallará si el archivo contiene incoherencias tal y como se definen en la especificación XBRL 2.1. Toma el valor booleano `true` o `false`. Valor predeterminado: `false`. Si toma el valor `false`, las incoherencias XBRL tal y como se definen en la especificación XBRL 2.1 no se tratan como errores.

6.2.6 RaptorXMLException

```
public interface RaptorXMLException
```

Tiene un solo método que genera la excepción.

RaptorXMLException

```
public void RaptorXMLException(String message)
```

Genera una excepción que contiene información sobre un error que ocurre durante el procesamiento.

Parámetros:

`message`: una cadena con información sobre el error.

 © 2018 Altova GmbH

6.3 Enumeraciones de RaptorXML para Java

Estas son las enumeraciones que están definidas. En esta sección se agrupan según su función:

RaptorXMLFactory
XML Validation
XSLT and XQuery
XBRL

6.3.1 RaptorXMLFactory

La interfaz **RaptorXMLFactory** define estas enumeraciones.

ENUMErrorFormat

```
public enum ENUMErrorFormat {
        eFormatText
        eFormatShortXML
        eFormatLongXML }
```

ENUMErrorFormat puede tomar uno de estos literales de enumeración: eFormatText, eFormatShortXML, eFormatLongXML. Estos literales establecen el formato de los mensajes de error, siendo eLongXML el formato más detallado. El literal predeterminado es eFormatText.

Utilizado por (Interfaz::Método):
RaptorXMLFactory setErrorFormat

6.3.2 XML Validation

La interfaz XMLValidator define estas enumeraciones.

ENUMAssessmentMode

```
public enum ENUMAssessmentMode {
        eAssessmentModeLax
        eAssessmentModeStrict   }
```

ENUMAssessmentMode toma uno de estos literales de enumeración: eAssessmentModeLax, eAssessmentModeStrict. Establecen si la validación debe ser laxa o estricta.

Utilizado por (Interfaz::Método):
XMLValidator setAssessmentMode

ENUMLoadSchemaLocation

```
public enum ENUMLoadSchemalocation {
```

```
        eLoadBySchemalocation
        eLoadByNamespace
        eLoadCombiningBoth
        eLoadIgnore    }
```

`ENUMLoadSchemalocation` contiene el literal de enumeración que especifica el mecanismo para ubicar el esquema. La selección se basa en el atributo de ubicación del esquema del documento de instancia XML o XBRL. Este atributo puede ser `xsi:schemaLocation` o `xsi:noNamespaceSchemaLocation`.

- `eLoadBySchemalocation` usa la URL del atributo de ubicación del esquema del documento de instancia XML o XBRL. Este literal de enumeración es el **valor predeterminado**.
- `eLoadByNamespace` usa la parte del espacio de nombres de `xsi:schemaLocation` y una cadena vacía en el caso de `xsi:noNamespaceSchemaLocation` para encontrar el esquema a través de una asignación de catálogo.
- `eLoadCombiningBoth`: si la URL del espacio de nombres o la URL de la ubicación del esquema incluye una asignación de catálogo, entonces se usa esta última. Si ambas URL tienen asignaciones de catálogo, entonces el valor de `ENUMSchemaMapping` decide qué asignación se usa. Si ninguna de ellas tiene una asignación de catálogo, se usa la URL de ubicación del esquema.
- `eLoadCombiningBoth`: se omite tanto el atributo `xsi:schemaLocation` como `xsi:noNamespaceSchemaLocation`.

Utilizado por (Interfaz::Método):

XMLValidator	setSchemalocationHints
XSLT	setSchemalocationHints
XBRL	setSchemalocationHints

ENUMSchemaImports

```
public enum ENUMSchemaImports {
        eSILoadBySchemalocation
        eSILoadPreferringSchemalocation
        eSILoadByNamespace
        eSILoadCombiningBoth
        eSILicenseNamespaceOnly    }
```

`ENUMSchemaImports` contiene el literal de enumeración que define el comportamiento de los elementos `xs:import` del esquema. Cada uno de estos elementos tiene un atributo opcional `namespace` y un atributo opcional `schemaLocation`.

- `eSILoadBySchemalocation` usa el valor del atributo `schemaLocation` para encontrar el esquema, teniendo en cuenta las asignaciones de catálogo. Si está presente el atributo `namespace`, se importa el espacio de nombres.
- `eSILoadPreferringSchemalocation`: si está presente el atributo `schemaLocation`, entonces se utiliza teniendo en cuenta las asignaciones de catálogo. Si no hay ningún atributo `schemaLocation`, entonces se usa el valor del atributo `namespace` a través de una asignación de catálogo. Este literal de enumeración es el **valor predeterminado**.
- `eSILoadByNamespace` usa el valor del atributo `namespace` para encontrar el esquema a través de una asignación de catálogo.

- eSILoadCombiningBoth: si tanto la URL de namespace como la URL de schemaLocation incluyen una asignación de catálogo, entonces se usa la asignación de catálogo. Si ambas URL tienen asignaciones de catálogo, entonces el valor de ENUMSchemaMapping decide qué asignación se utiliza. Si ninguna URL tiene una asignación de catálogo, entonces se usa la URL de schemaLocation.
- eSILicenseNamespaceOnly: el espacio de nombres se importa. No se importa ningún documento de esquema.

Utilizado por (Interfaz::Método):

XMLValidator	setSchemaImports
XSLT	setSchemaImports
XBRL	setSchemaImports

ENUMSchemaMapping

```
public enum ENUMSchemaMapping {
       eSMPreferSchemalocation
       eSMPreferNamespace    }
```

ENUMSchemaMapping contiene el literal de enumeración que especifica si se selecciona el espacio de nombres o la ubicación del esquema.

- eSMPreferNamespace: selecciona el espacio de nombres.
- eSMPreferSchemalocation: selecciona la ubicación del esquema. Este es el valor predeterminado.

Utilizado por (Interfaz::Método):

XMLValidator	setSchemaMapping
XSLT	setSchemaMapping
XBRL	setSchemaMapping

ENUMXMLValidationMode

```
public enum ENUMXMLValidationMode {
       eProcessingModeValid
       eProcessingModeWF   }
```

ENUMXMLValidationMode contiene el literal de enumeración que especifica el tipo de validación XML que se debe llevar a cabo (validación o comprobación de formato).

- eProcessingModeValid: establece el modo de procesamiento XML validation.
- eProcessingModeWF: establece el modo de procesamiento XML wellformed. Este es el valor predeterminado

Utilizado por (Interfaz::Método):

XMLValidator	setXMLValidationMode
XSLT	setXMLValidationMode
XQuery	setXMLValidationMode

ENUMXMLValidationType

```
public enum ENUMValidationType {
        eValidateAny
        eValidateXMLWithDTD
        eValidateXMLWithXSD
        eValidateDTD
        eValidateXSD
        eValidateJSON
        eValidateJSONSchema
        eValidateAvro
        eValidateAvroSchema
        eValidateAvroJSON }
```

`ENUMValidationType` contiene el literal de enumeración que especifica qué validación se debe llevar a cabo y, en caso de documentos XML, si se usa una DTD o un XSD para la validación.

- `eValidateAny`: el tipo de documento se selecciona automáticamente.
- `eValidateXMLWithDTD`: valida un documento XML con una DTD.
- `eValidateXMLWithXSD`: valida un documento XML con un esquema XML (XSD).
- `eValidateDTD`: valida un documento DTD.
- `eValidateXSD`: valida un documento XSD.
- `eValidateJSON`: valida un documento de instancia JSON.
- `eValidateJSONSchema`: valida un documento de esquema JSON.
- `eValidateAvro`: valida un archivo binario Avro. Los datos Avro del archivo binario se validan con el esquema Avro que contiene el archivo binario.
- `eValidateAvroSchema`: valida un esquema Avro con la especificación Avro schema.
- `eValidateAvroJSON`: valida un archivo de datos Avro serializados en JSON con un esquema Avro.

Utilizado por (Interfaz::Método):
`XMLValidator` `isValid`

ENUMWellFormedCheckType

```
public enum ENUMWellformedCheckType {
        eWellformedAny
        eWellformedXML
        eWellformedDTD
        eWellformedJSON    }
```

`ENUMWellformedCheckType` contiene el literal de enumeración que especifica qué tipo de comprobación de formato se lleva a cabo (para documentos XML o DTD).

- `eWellformedAny`: el tipo de documento se detecta automáticamente.
- `eWellformedXML`: comprueba si un documento XML tiene el formato correcto.
- `eWellformedDTD`: comprueba si un documento DTD tiene el formato correcto.
- `eWellformedJSON`: comprueba si un documento JSON tiene el formato correcto.

Utilizado por (Interfaz::Método):
`XMLValidator` `isWellformed`

 © 2018 Altova GmbH

▼ ENUMXSDVersion

```
public enum ENUMXSDVersion {
        eXSDVersionAuto
        eXSDVersion10
        eXSDVersion11    }
```

`ENUMXSDVersion` contiene el literal de enumeración que especifica la versión de XML Schema.

- `eXSDVersionAuto`: la versión de XML Schema se detecta automáticamente a partir del atributo `vc:minVersion` del documento XSD. Si este atributo tiene el valor `1.1`, se considera que el documento es XSD 1.1. Si el atributo tiene cualquier otro valor o el atributo no existe, se considera que el documento es XSD 1.0.
- `eXSDVersion10`: establece la versión XML Schema 1.0 para la validación.
- `eXSDVersion11`: establece la versión XML Schema 1.1 para la validación.

Utilizado por (Interfaz::Método):

XMLValidator	setXSDVersion
XSLT	setXSDVersion
XQuery	setXSDVersion

6.3.3 XSLT y XQuery

La interfaz XSLT define estas enumeraciones.

▼ ENUMXSLTVersion

```
public enum ENUMXSLTVersion {
        eVersion10
        eVersion20
        eVersion30 }
```

`ENUMXSLTVersion` toma uno de estos literales de enumeración: `eVersion10`, `eVersion20`, `eVersion30`. Estos literales establecen qué versión de XSLT se usa para el procesamiento (para la validación o transformación XSLT).

Utilizado por (Interfaz::Método):

XSLT	setVersion

La interfaz XQuery define estas enumeraciones.

▼ ENUMXQueryVersion

```
public enum ENUMXQueryVersion {
        eVersion10
        eVersion30
```

```
        eVersion31 }
```

ENUMXQueryVersion toma uno de estos literales de enumeración: eVersion10, eVersion30, eVersion31. Estos literales establecen qué versión de XQuery se usa para el procesamiento (para la ejecución o validación).

Utilizado por (Interfaz::Método):
XQuery setVersion

▼ ENUMXQueryUpdatedXML

```
public enum ENUMXQueryUpdatedXML {
        eUpdatedDiscard
        eUpdatedWriteback
        eeUpdatedAsMainResult }
```

ENUMXQueryVersion toma uno de estos literales de enumeración:

- eUpdatedDiscard: las actualizaciones se descartan y no se escriben en el archivo.
- eUpdatedWriteback: las actualizaciones se escriben en el archivo XML de entrada que se especifica con setInputXMLFileName.
- eUpdatedAsMainResult: las actualizaciones se escriben en la ubicación que especifica el parámetro outputFile de ExecuteUpdate.

Utilizado por (Interfaz::Método):
XQuery setUpdatedXMLWriteMode

6.3.4 XBRL

La interfaz XBRL define estas enumeraciones.

▼ ENUMValidationType

```
public enum ENUMValidationType {
        eValidateAny
        eValidateInstance
        eValidateTaxonomy
        eValidateInline
        eValidateTaxonomyPackage }
```

ENUMValidationType contiene el literal de enumeración que especifica qué tipo de validación se debe llevar a cabo y, en el caso de los documentos XML, si se usa una DTD o un XSD para la validación.

- eValidateAny: el tipo de documento se detecta automáticamente.
- eValidateInstance: valida un documento de instancia XBRL (extensión de archivo .xbrl).
- eValidateTaxonomy: valida una taxonomía XBRL (extensión de archivo .xsd).

- eValidateInline: valida un documento Inline XBRL (iXBRL) con la especificación Inline XBRL 1.0 o Inline XBRL 1.1.
- eValidateTaxonomyPackage: valida un paquete de taxonomías XBRL con la especificación Taxonomy 1.0 Packages.

Utilizado por (Interfaz::Método):

XBRL	isValid

ENUMTableOutputFormat

```
public enum ENUMTableOutputFormat {
        eFormatXML
        eFormatHTML }
```

ENUMTableOutputFormat contiene el literal de enumeración que especifica el formato de salida del documento que contiene las tablas que se generan.

- eFormatXML: el documento de salida con las tablas generadas está en formato XML.
- eFormatHTML: el documento de salida con las tablas generadas está en formato HTML.

Utilizado por (Interfaz::Método):

XBRL	setTableOutputFormat

ENUMIXBRLVersion

```
public enum ENUMIXBRLVersion {
        eVersion10
        eVersion11
        eVersionDetect }
```

ENUMIXBRLVersion contiene el literal de enumeración que especifica la versión de la especificación Inline XBRL que se debe usar: Inline XBRL 1.0 o Inline XBRL 1.1. El literal eVersionDetect habilita la detección automática de la versión.

Utilizado por (Interfaz::Método):

XBRL	isValid

ENUMIXBRLUriStrategy

```
public enum ENUMIXBRLUriStrategy {
        eStrategyNone
        eStrategyMakeAbsolute
        eStrategyMakeRelative
        eStrategyKeepRelative }
```

ENUMIXBRLUriStrategy especifica cómo se deben escribir los URI en el documento XBRL que se genera.

- eStrategyNone: copia el URL literalmente en el documento de destino.

- `eStrategyMakeAbsolute`: convierte los URI relativos en absolutos resolviéndolos con el URI base dentro del ámbito del elemento correspondiente del documento de entrada.
- `eStrategyMakeRelative`: si se puede, convierte los URI absolutos y relativos en relativos al documento de salida (y si no se puede, escribe el URI absoluto resuelto).
- `eStrategyKeepRelative`: si se puede, convierte solamente los URI relativos en relativos al documento de salida (y copia los URI que sean absolutos).

Utilizado por (Interfaz::Método):

`XBRL` `setIXBRLUriTransformationStrategy`

 © 2018 Altova GmbH

Altova RaptorXML+XBRL Server 2018

Interfaces COM y .NET

7 Interfaces COM y .NET

Dos interfaces, una sola API

Las interfaces COM y .NET de RaptorXML+XBRL Server utilizan una sola API: la API de COM/.NET API para RaptorXML+XBRL Server. La interfaz .NET está construida como un contenedor alrededor de la interfaz COM.

Puede usar RaptorXML con:

- Lenguajes de scripting como JavaScript, a través de la interfaz COM
- Lenguajes de programación como C#, a través de la interfaz .NET

Apartados de esta sección

Esta sección se divide en varios apartados:

- Notas sobre la interfaz COM: funcionamiento general de la interfaz COM y pasos necesarios para trabajar con ella.
- Notas sobre la interfaz .NET: cómo preparar el entorno de desarrollo para trabajar con la interfaz .NET.
- Lenguajes de programación: fragmentos de código en los lenguajes de programación más frecuentes, que muestran cómo se llama a las funciones de RaptorXML.
- Referencia de API: documentación del modelo de objetos, de los objetos y de las propiedades de la API.

 © 2018 Altova GmbH

7.1 Notas sobre la interfaz COM

Durante la instalación, RaptorXML+XBRL Server se registra automáticamente como objeto de servidor COM. Esto permite invocar a RaptorXML+XBRL Server desde otras aplicaciones y lenguajes de scripting compatibles con el uso de llamadas COM. Si quiere cambiar la ubicación del paquete de instalación de RaptorXML+XBRL Server, es mejor desinstalar RaptorXML+XBRL Server y volver a instalarlo en su nueva ubicación. De este modo el programa de instalación se encarga del proceso de registro.

Comprobar si RaptorXML+XBRL Server se registró correctamente
Si RaptorXML+XBRL Server se registró correctamente, el registro incluye las clases `RaptorXML.Server`. Estas clases suelen estar en `HKEY_LOCAL_MACHINE\SOFTWARE\Classes`.

Código de ejemplo

El ejemplo de código VBScript muestra cómo se puede usar la API de RaptorXML a través de su interfaz COM. Para este fragmento de código también ofrecemos un archivo de ejemplo en la carpeta `examples/API` de la carpeta de aplicación de RaptorXML.

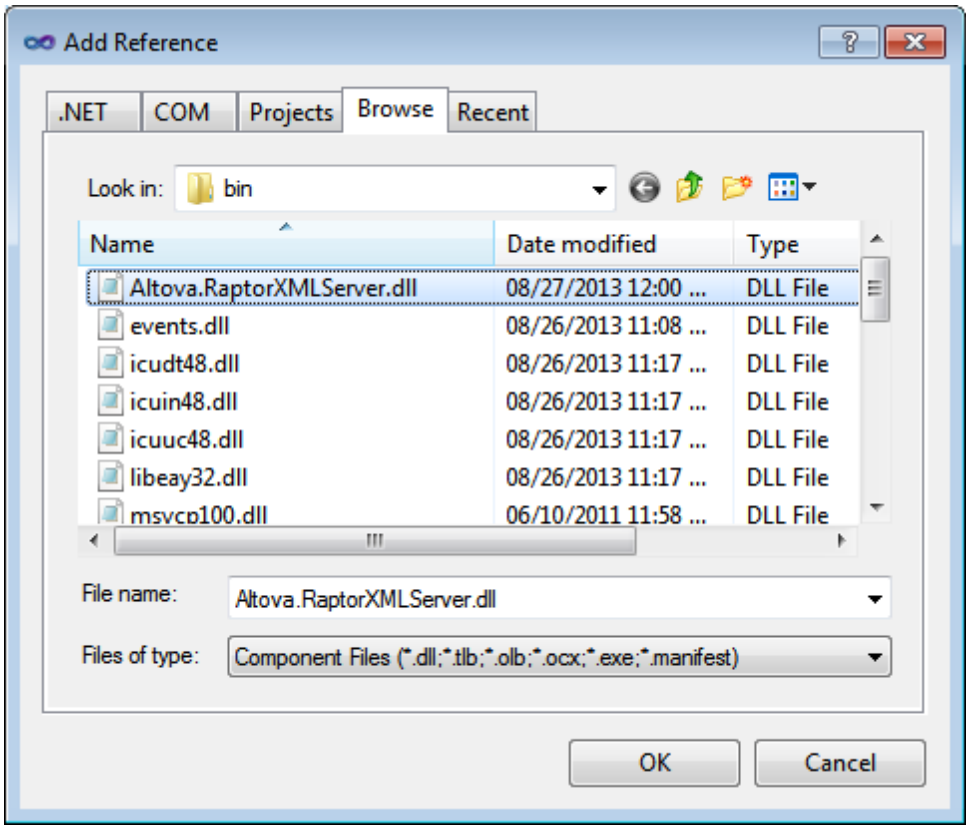
7.2 Notas sobre la interfaz NET

La interfaz .NET está construida como contenedor alrededor de la interfaz COM de RaptorXML. Se ofrece como ensamblado de interoperabilidad principal firmado por Altova y usa el espacio de nombres `Altova.RaptorXMLServer`.

Añadir al proyecto Visual Studio .NET una referencia al DLL de RaptorXML
Para poder usar RaptorXML en su proyecto .NET debe añadir una referencia al DLL de RaptorXML (`Altova.RaptorXMLServer.dll`) en el proyecto. El paquete de instalación de RaptorXML+XBRL Server contiene un archivo DLL firmado y llamado `Altova.RaptorXMLServer.dll`, que se añadirá automáticamente al caché global de ensamblados cuando se instale RaptorXML usando el programa de instalación de RaptorXML. (Su ubicación suele ser la carpeta `C:\WINDOWS\assembly`.)

Para añadir una referencia a este DLL en un proyecto .NET:

1. Abra el proyecto .NET y haga clic en **Proyecto | Añadir referencia**. Aparece el cuadro de diálogo "Añadir referencia" (*imagen siguiente*).



2. En la pestaña *Browse* vaya a la carpeta: `<RaptorXML application folder>/bin`, seleccione el DLL de RaptorXML `Altova.RaptorXMLServer.dll` y haga clic en **Aceptar**.
3. Seleccione el comando **View | Object Browser** para ver los objetos de la API de RaptorXML.

 © 2018 Altova GmbH

Cuando `Altova.RaptorXMLServer.dll` esté a disposición de la interfaz .NET y RaptorXML esté registrado como objeto servidor COM, las funciones de RaptorXML estarán disponibles en el proyecto .NET.

Nota: RaptorXML se registra automáticamente como objeto servidor COM durante la instalación. No es necesario registrarlo a mano.

Nota: si recibe un error de acceso, compruebe que tiene los permisos necesarios. Vaya a Servicios de componentes y otorge permisos a la misma cuenta que ejecuta el grupo de aplicaciones que contiene RaptorXML.

Código de ejemplo

El ejemplo de código C# y de código Visual Basic .NET muestran cómo se puede usar la API de RaptorXML a través de su interfaz .NET. Para estos ejemplos de código también ofrecemos archivos de muestra en la carpeta `examples/API` de la carpeta de aplicación de RaptorXML.

7.3 Lenguajes de programación

No todos los lenguajes de programación ofrecen la misma compatibilidad con COM y .NET. A continuación puede ver algunos ejemplos para los lenguajes de programación más frecuentes. Los fragmentos de código que aparecen en este apartado explican cómo acceder a funciones básicas. Estas funciones básicas están incluidas en los archivos de la carpeta `examples/API` de la carpeta de aplicación RaptorXML+XBRL Server.

VBScript

VBScript se puede usar para acceder a la API de COM de RaptorXML+XBRL Server. El ejemplo de código VBScript demuestra estas funciones básicas:

- Conectarse a la API de COM de RaptorXML+XBRL Server
- Validar un archivo XML
- Realizar una transformación XSLT
- Ejecutar un XQuery

C#

C# se puede usar para acceder a la API de .NET de RaptorXML+XBRL Server. El ejemplo de código C# demuestra cómo acceder a la API para utilizar estas funciones básicas:

- Conectarse a la API de .NET de RaptorXML+XBRL Server
- Validar un archivo XML
- Realizar una transformación XSLT
- Ejecutar un XQuery

Visual Basic .NET

Visual Basic.NET solo difiere de C# en la sintaxis. El acceso a la API de .NET funciona exactamente igual. El ejemplo de código Visual Basic demuestra estas funciones básicas:

- Conectarse a la API de .NET de RaptorXML+XBRL Server
- Validar un archivo XML
- Realizar una transformación XSLT
- Ejecutar un XQuery

Este apartado contiene estos ejemplos:

Para la interfaz COM

- Un ejemplo de código VBScript

 © 2018 Altova GmbH

Para la interfaz .NET

- Un ejemplo de código C#
- Un ejemplo de código Visual Basic

7.3.1 Ejemplo de COM: VBScript

Este ejemplo de código VBScript se divide en varias partes:

- Preparación e inicialización del objeto COM de RaptorXML
- Validación de un archivo XML
- Transformación XSLT y devolución del resultado en forma de cadena de texto
- Procesamiento de un documento XQuery y devolución del resultado en forma de archivo
- Preparación de la secuencia de ejecución del script y de su punto de entrada

```
' El objeto COM de RaptorXML
dim objRaptor

' Inicializar el objeto COM de RaptorXML
sub Init
	objRaptor = Null
	On Error Resume Next
	' Intentar cargar el objeto COM de 32 bits. No emitir excepciones si no
se encuentra el objeto
	Set objRaptor = WScript.GetObject( "", "RaptorXML.Server" )
	On Error Goto 0
	if ( IsNull( objRaptor ) ) then
		' Intentar cargar el objeto COM de 64 bits (se emite una excepción
si no se encuentra el objeto)
		Set objRaptor = WScript.GetObject( "", "RaptorXML_x64.Server" )
	end if
	' Configurar el servidor: notificación de errores, nombre y puerto del
servidor HTTP (IPv6 localhost en este	ejemplo)
	objRaptor.ErrorLimit = 1
	objRaptor.ReportOptionalWarnings = true
	objRaptor.ServerName = "::1"
	objRaptor.ServerPort = 8087
end sub



' Validar un archivo
sub ValidateXML
	' Obtener una instancia del validador del objeto servidor
	dim objXMLValidator
	Set objXMLValidator = objRaptor.GetXMLValidator()

	' Configurar los datos de entrada
```

```
        objXMLValidator.InputFileName = "MiArchivoXML.xml"

        ' Validar; en caso de que el archivo no sea válido, notificar el problema
devuelto por RaptorXML
        if ( objXMLValidator.IsValid() ) then
                MsgBox( "La cadena de entrada es válida" )
        else
                MsgBox( objXMLValidator.LastErrorMessage )
        end if
end sub



' Realizar una transformación; devolver el resultado en forma de cadena de texto
sub RunXSLT
        ' Obtener una instancia del motor XSLT del objeto servidor
        dim objXSLT
        set objXSLT = objRaptor.GetXSLT

        ' Configurar los datos de entrada
        objXSLT.InputXMLFileName = "MiArchivoXML.xml"
        objXSLT.XSLFileName = "MiTransformación.xsl"

        ' Ejecutar la transformación; si finaliza correctamente, el resultado se
devuelve. Si no, el motor devuelve  una lista de errores
        MsgBox( objXSLT.ExecuteAndGetResultAsString() )
end sub



' Ejecutar un XQuery; guardar el resultado en un archivo
sub RunXQuery
        ' Obtener una instancia del motor XQuery del objeto servidor
        dim objXQ
        set objXQ = objRaptor.GetXQuery()

        ' Configurar los datos de entrada
        objXQ.InputXMLFileName = "MiArchivoXML.xml"
        objXQ.XQueryFileName = "MiXQuery.xq"

        ' Configurar la serialización (opcional si quiere perfeccionar el formato
del archivo de salida)
        objXQ.OutputEncoding = "UTF8"
        objXQ.OutputIndent = true
        objXQ.OutputMethod = "xml"
        objXQ.OutputOmitXMLDeclaration = false

        ' Ejecutar la consulta; el resultado se serializa en la ruta de acceso
indicada
        call objXQ.Execute( "MyQueryResult.xml" )
end sub
```

 © 2018 Altova GmbH

```
' Realizar todas las funciones de muestra
sub main
       Init
       ValidateXML
       RunXSLT
       RunXQuery
end sub

' Punto de entrada del script; ejecutar la función principal
main
```

7.3.2 Ejemplo de .NET: C#

Este ejemplo de código C# se divide en varias partes:

- Preparación e inicialización del objeto .NET de RaptorXML
- Validación de un archivo XML
- Transformación XSLT y devolución del resultado en forma de cadena de texto
- Procesamiento de un documento XQuery y devolución del resultado en forma de archivo
- Preparar la secuencia de ejecución del código y su punto de entrada

```
using System;
using System.Text;
using Altova.RaptorXMLServer;

namespace RaptorXMLRunner
{
    class Program
    {
    // El objeto .NET de RaptorXML
       static ServerClass objRaptorXMLServer;

    // Inicializar el objeto .NET de RaptorXML
       static void Init()
       {
       // Asignar un objeto de RaptorXML
          objRaptorXMLServer = new ServerClass();

       // Configurar el servidor: notificación de errores, nombre y puerto del
servidor HTTP
       // (IPv6 localhost en este ejemplo)
          objRaptorXMLServer.ErrorLimit = 1;
          objRaptorXMLServer.ReportOptionalWarnings = true;
          objRaptorXMLServer.ServerName = "::1"
          objRaptorXMLServer.ServerPort = 8087
       }
```

```
        // Validar un archivo
           static void ValidateXML()
           {
           // Obtener una instancia del motor de validación del objeto servidor
              XMLValidator objXMLValidator =
objRaptorXMLServer.GetXMLValidator();

           // Configurar los datos de entrada
              objXMLValidator.InputFileName = "MyXMLFile.xml";

           // Validar; si el archivo no es válido,
              notificar el problema devuelto por RaptorXML
              if ( objXMLValidator.IsValid() )
                 Console.WriteLine( "La cadena de entrada no es válida" );
              else
                 Console.WriteLine( objXMLValidator.LastErrorMessage );
           }



        // Realizar una transformación XSLT y
        // devolver el resultado en forma de cadena de texto
           static void RunXSLT()
           {
           // Obtener una instancia del motor XSLT del objeto servidor
              XSLT objXSLT = objRaptorXMLServer.GetXSLT();

              // Configurar los datos de entrada
                 objXSLT.InputXMLFileName = "MiArchivoXML.xml";
                 objXSLT.XSLFileName = "MiTransformación.xsl";

             // Ejecutar la transformación.
             // Si se ejecuta correctamente, el resultado se devuelve.
             // Si hay errores, se emite una lista de errores
                Console.WriteLine( objXSLT.ExecuteAndGetResultAsString() );
           }



        // Ejecutar un XQuery y guardar el resultado en un archivo
           static void RunXQuery()
           {
           // Obtener una instancia del motor XQuery del objeto servidor
              XQuery objXQuery = objRaptorXMLServer.GetXQuery();
```

```
            // Configurar los datos de entrada
               objXQuery.InputXMLFileName = exampleFolder + "simple.xml";
               objXQuery.XQueryFileName = exampleFolder + "CopyInput.xq";

           // Configurar la serialización (opcional si quiere mejorar el formato
de salida)
              objXQuery.OutputEncoding = "UTF8"
              objXQuery.OutputIndent = true
              objXQuery.OutputMethod = "xml"
              objXQuery.OutputOmitXMLDeclaration = false

           // Ejecutar la consulta. El resultado se serializa en la ruta de
acceso indicada
              objXQuery.Execute( "MyQueryResult.xml" );
          }





          static void Main(string[] args)
          {
            try
            {
            // Punto de entrada. Realizar todas las funciones
               Init();
               ValidateXML();
               RunXSLT();
               RunXQuery();
            }
            catch (System.Exception ex)
            {
                Console.WriteLine( ex.Message );
                Console.WriteLine( ex.ToString() );
            }
        }
    }
}
```

7.3.3 Ejemplo de .NET: Visual Basic .NET

Este ejemplo de código Visual Basic se divide en varias partes:

- Preparar e inicializar el objeto .NET de RaptorXML
- Validar un archivo XML
- Transformación XSLT y devolución del resultado en forma de cadena de texto
- Procesamiento de un documento XQuery y devolución del resultado en forma de archivo
- Preparación de la secuencia de ejecución del código y de su punto de entrada

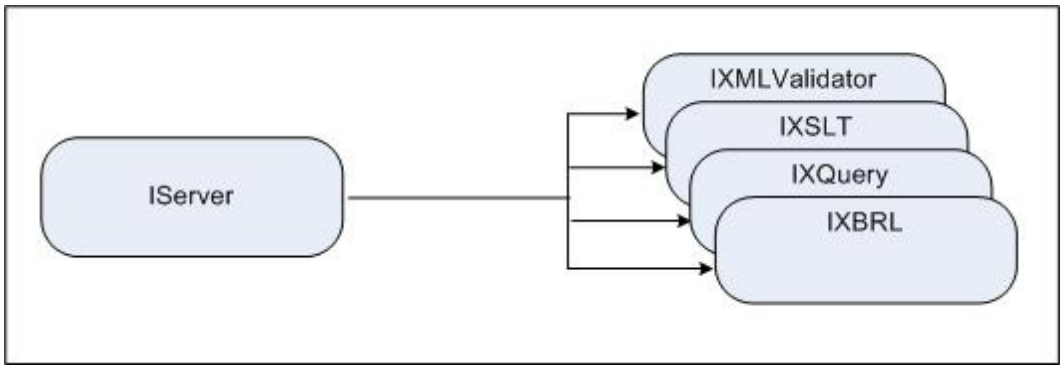
```
Option Explicit On
Imports Altova.RaptorXMLServer

Module RaptorXMLRunner

   ' El objeto .NET de RaptorXML
     Dim objRaptor As Server

   ' Inicializar el objeto .NET de RaptorXML
     Sub Init()

       ' Asignar un objeto RaptorXML
         objRaptor = New Server()

       ' Configurar el servidor: notificación de errores, nombre y puerto del
servidor HTTP (IPv6 localhost en este ejemplo)
         objRaptor.ErrorLimit = 1
         objRaptor.ReportOptionalWarnings = True
         objRaptor.ServerName = "::1"
         objRaptor.ServerPort = 8087
     End Sub



   ' Validar un archivo
     Sub ValidateXML()

       ' Obtener una instancia del validador del objeto RaptorXML
         Dim objXMLValidator As XMLValidator
         objXMLValidator = objRaptor.GetXMLValidator()

       ' Configurar los datos de entrada
         objXMLValidator.InputFileName = "MiArchivoXML.xml"

       ' Validar; si el archivo no es válido, notificar el problema devuelto por
RaptorXML
         If (objXMLValidator.IsValid()) Then
            Console.WriteLine("La cadena de entrada es válida")
         Else
            Console.WriteLine(objXMLValidator.LastErrorMessage)
         End If
     End Sub



   ' Realizar una transformación; devolver el resultado en forma de cadena de
texto
     Sub RunXSLT()

       ' Obtener una instancia del motor XSLT del objeto servidor
```

```
        Dim objXSLT As XSLT
        objXSLT = objRaptor.GetXSLT()

      ' Configurar los datos de entrada
        objXSLT.InputXMLFileName = "MiArchivoXML.xml"
        objXSLT.XSLFileName = "MiTransformación.xsl"

      ' Ejecutar la transformación; si se ejecuta correctamente, se devuelve el
resultado; si hay errores, devolver  una lista de errores
        Console.WriteLine(objXSLT.ExecuteAndGetResultAsString())
    End Sub




   ' Ejecutar un XQuery; guardar el resultado en un archivo
    Sub RunXQuery()

      ' Obtener una instancia del motor XQuery del objeto servidor
        Dim objXQ As XQuery
        objXQ = objRaptor.GetXQuery()

      ' Configurar los datos de entrada
        objXQ.InputXMLFileName = "MyXMLFile.xml"
        objXQ.XQueryFileName = "MyQuery.xq"

      ' Configurar la serialización (opcional para mejorar el formato del
archivo de salida)
        objXQ.OutputEncoding = "UTF8"
        objXQ.OutputIndent = true
        objXQ.OutputMethod = "xml"
        objXQ.OutputOmitXMLDeclaration = false

      ' Ejecutar la consulta; el resultado se serializa en la ruta de acceso
indicada
        objXQ.Execute( "MyQueryResult.xml" )
    End Sub




    Sub Main()
      ' Punto de entrada; realizar todas las funciones de muestra
        Init()
        ValidateXML()
        RunXSLT()
        RunXQuery()
    End Sub

End Module
```

7.4 Referencia de la API

Esta sección describe la especificación de la API: su modelo de objetos y los detalles de sus interfaces y enumeraciones.

El punto de partida para usar las funciones de RaptorXML es la interfaz `IServer`. Este objeto contiene los objetos que ofrecen las funciones de RaptorXML: validación XML, validación XBRL, transformaciones XSLT y procesamiento de documentos XQuery. El modelo de objetos de la API de RaptorXML aparece en este esquema:



La jerarquía del modelo de objetos aparece más abajo y las interfaces se describen una por una en los apartados siguientes. Los métodos y propiedades de cada interfaz también se describen en detalle.

```
-- IServer
  |-- IXMLValidator
  |-- IXSLT
  |-- IXQuery
  |-- IXBRL
```

7.4.1 Interfaces

Las interfaces definidas son estas:

IServer
IXMLValidator
IXSLT
IXQuery
IXBRL

 © 2018 Altova GmbH

7.4.1.1 IServer

La interfaz **IServer** ofrece métodos para devolver las interfaces del motor de RaptorXML correspondiente: del validador XML, XBRL, XSLT y XQuery. Las propiedades definen los parámetros de la interfaz.

Métodos

Los métodos de la **IServer** devuelven interfaces del correspondiente motor de RaptorXML: del validador XML, XBRL, XSLT y XQuery.

- GetXMLValidator

  IXMLValidator **GetXMLValidator()**
  Devuelve una instancia del motor validador XML.

- GetXBRL

  IXBRL **GetXBRL()**
  Devuelve una instancia del motor XBRL.

- GetXSLT

  IXSLT **GetXSLT()**
  Devuelve una instancia del motor XSLT.

- GetXQuery

  IXQuery **GetXQuery()**
  Devuelve una instancia del motor XQuery.

Propiedades

Las propiedades de la interfaz **IServer** se describen a continuación por orden alfabético. Observe que las entradas de cadena que se deben interpretar como direcciones URL deben darse como rutas de acceso absolutas. Si se usa una ruta de acceso relativa, en el módulo de llamada debería definirse un mecanismo para resolver la ruta de acceso relativa.

- APIMajorVersion

  int **APIMajorVersion**
  Devuelve la versión principal de la API en forma de entero. La versión principal de la API puede diferir de la versión principal del producto si la API está conectada a otro servidor.

- APIMinorVersion

`int` **APIMinorVersion**
Devuelve la versión secundaria de la API en forma de entero. La versión secundaria de la API puede diferir de la versión secundaria del producto si la API está conectada a otro servidor.

- APIServicePackVersion

`int` **APIServicePackVersion**
Devuelve la versión de service pack de la API en forma de entero. La versión de service pack de la API puede diferir de la versión de service pack del producto si la API está conectada a otro servidor.

- ErrorFormat

ENUMErrorFormat **ErrorFormat**
Establece el formato de error de RaptorXML en uno de los literales de `ENUMErrorFormat` (Text, ShortXML, LongXML).

- ErrorLimit

`int` **ErrorLimit**
Establece el límite error de validación de RaptorXML. El parámetro `limit` es de tipo `int` (Java), `uint` (COM/.NET). Especifica el número de errores que se deben notificar antes de detener la ejecución. Use el valor `-1` para que `limit` sea ilimitado (es decir, que se notifiquen todos los errores). El valor predeterminado es `100`.

- GlobalCatalog

`string` **GlobalCatalog**
Establece la ubicación, en forma de URL, del archivo de catálogo principal (de punto de entrada). La cadena dada debe ser una URL absoluta que indique la ubicación exacta del archivo de catálogo principal que se debe usar.

- GlobalResourceConfig

`string` **GlobalResourceConfig**
Establece la configuración activa del recurso global. El parámetro `config` es de tipo `String` y especifica el nombre de la configuración utilizada por el recurso global activo.

- GlobalResourceFile

`string` **GlobalResourceFile**
Establece la ubicación, en forma de URL, del archivo XML de recursos globales. La cadena dada debe ser una URL absoluta que indique la ubicación exacta del archivo XML de recursos globales.

- MajorVersion

`int` **MajorVersion**
Devuelve la versión principal del producto en forma de entero. *Ejemplo:* para `Altova RaptorXML+XBRL Server 2014r2sp1(x64)`, devolvería `16` (la diferencia entre la versión principal (`2014`) y el año inicial `1998`). Si se produce un error, se emite la excepción **`RaptorXMLException`** (*Java*).

- MinorVersion

`int` **MinorVersion**

Devuelve la versión secundaria del producto en forma de entero. *Ejemplo:* para `Altova RaptorXML+XBRL Server 2018r2sp1(x64)`, devolvería `2` (por el número de versión secundaria `r2`). Si se produce un error, se emite la excepción **`RaptorXMLException`** (*Java*).

- ProductName

  `string` **`ProductName`**
  Devuelve el nombre del producto en forma de cadena. *Ejemplo:* para `Altova RaptorXML +XBRL Server 2018r2sp1(x64)`, devolvería `Altova RaptorXML+XBRL Server`. Si se produce un error, se emite la excepción **`RaptorXMLException`** (*Java*).

- ProductNameAndVersion

  `string` **`ProductNameAndVersion`**
  Devuelve la versión de service pack del producto en forma de entero. *Ejemplo:* para `Altova RaptorXML+XBRL Server 2018r2sp1(x64)`, devolvería `Altova RaptorXML+XBRL Server 2018r2sp1(x64)`. Si se produce un error, se emite la excepción **`RaptorXMLException`** (*Java*).

- ReportOptionalWarnings

  `bool` **`ReportOptionalWarnings`**
  Habilita o deshabilita la notificación de advertencias. El valor `true` habilita las advertencias y `false` las deshabilita.

- ServerName

  `string` **`ServerName`**
  Establece el nombre del servidor HTTP. Si se produce un error, se emite una excepción **`RaptorXMLException`** (*Java*). El parámetro de entrada es una cadena que indica el nombre del servidor HTTP.

- ServerPath

  `string` **`ServerPath`**
  Especifica, en forma de URL, la ruta de acceso del servidor HTTP. Si se produce un error, se emite una excepción `RaptorXMLException`.

- ServerPort

  `int` **`ServerPort`**
  Especifica el puerto servidor del servidor HTTP. El tipo es `ushort`. Si se produce un error, se emite una excepción `RaptorXMLException`.

- ServicePackVersion

  `int` **`ServicePackVersion`**
  Devuelve la versión de service pack del producto como entero. *Ejemplo*: para `RaptorXML +XBRL Server 2018r2sp1(x64)`, devolvería `1` (por el número de versión de service pack `sp1`). Si se produce un error, se emite una excepción **`RaptorXMLException`** (*Java*).

- UserCatalog

  `string` **`UserCatalog`**
  Establece la ubicación, en forma de URL, del archivo de catálogo del usuario personal. La

cadena dada debe ser una URL absoluta que indique la ubicación exacta del archivo de catálogo personal que se debe usar.

7.4.1.2 IXMLValidator

Métodos

La interfaz **IXMLValidator** ofrece estos métodos. Todos ellos devuelven el valor booleano `True` o `False`, dependiendo de si la operación finalizó correctamente o no.

- ExtractAvroSchema

  bool ExtractAvroSchema(string **outputPath**)

  Extrae un esquema Avro de un archivo binario. El parámetro `outputPath` es una URL absoluta que especifica la ubicación de salida. El resultado es `true` si la extracción finaliza correctamente. De lo contrario, el resultado es `false`. Si se produce un error, se emite una excepción **RaptorXMLException** (*detalle de la interfaz Java*). Use `getLastErrorMessage` (***en Java***) o `LastErrorMessage` (***en COM/.NET***) para obtener más información.

- IsValid

  bool IsValid(ENUMValidationType **type**)

  Devuelve el resultado de la validación del documento XML, documento de esquema o documento DTD. El tipo de documento que se debe validar viene dado por el parámetro `type`, que toma como valor un literal de la enumeración `ENUMValidationType` (Java, COM/.NET). Si el documento es válido, el resultado es `true`. Si no lo es, el resultado es `false`. Si se produce un error, se emite una excepción **RaptorXMLException** (*detalles de la interfaz Java*). Use `getLastErrorMessage` (***en Java***) o `LastErrorMessage` (***en COM/.NET***) para obtener más información.

- IsWellFormed

  bool IsWellFormed(ENUMWellformedCheckType **type**)

  Devuelve el resultado de la comprobación de formato del documento XML o DTD. El tipo de documento que se debe revisar se especifica con el parámetro `type`, que toma como valor un literal de la enumeración `ENUMWellformedCheckType` (Java, COM/.NET). Si el documento tiene el formato correcto, el resultado es `true`. De lo contrario devuelve `false`. Si se produce un error, se emite la excepción **RaptorXMLException** (*detalles de la interfaz Java*). Use `getLastErrorMessage` (***en Java***) o `LastErrorMessage` (***en COM/.NET***) para obtener más información.

Propiedades

Las propiedades de la interfaz **IXMLValidator** se describen a continuación por orden alfabético. Observe que las cadenas de entrada que se deben interpretar como direcciones URL deben aportar rutas de acceso absolutas. Si se usa una ruta de acceso relativa, en el módulo de

 © 2018 Altova GmbH

llamada se debe definir un mecanismo para resolver la ruta de acceso relativa.

- AssessmentMode

  ENUMAssessmentMode **AssessmentMode**
  Establece el modo de evaluación de la validación XML (`Strict/Lax`), que viene dado por un literal de `ENUMAssessmentMode` (Java, COM/.NET).

- AvroSchemaFileName

  string AvroSchemaFileName
  Establece la ubicación, en forma de URL, del esquema Avro externo que se debe usar. La cadena dada debe ser una URL absoluta que indique la ubicación exacta del archivo de esquema Avro.

- AvroSchemaFromText

  string AvroSchemaFromText
  Aporta el contenido del documento de esquema Avro que se debe usar. La cadena dada es el contenido del documento de esquema Avro que se debe usar.

- DTDFileName

  string DTDFileName
  Establece la ubicación, en forma de URL, del documento DTD que se debe usar para la validación. La cadena dada debe ser una URL absoluta que indique la ubicación exacta del documento DTD.

- DTDFromText

  string DTDFromText
  Aporta el contenido del documento DTD que se debe usar para la validación. La cadena dada es el contenido del documento DTD que se debe usar.

- EnableNamespaces

  bool EnableNamespaces
  Habilita el procesamiento preparado para espacios de nombres. Esto ayuda en la revisión de instancias XML para buscar errores debidos a espacios de nombres incorrectos. El valor `true` habilita un procesamiento preparado para espacios de nombres. El valor `false` lo deshabilita. El valor predeterminado es `false`.

- InputFileArray

  object InputFileArray
  Aporta una matriz de direcciones URL de archivos XML que se deben usar como datos de entrada. La propiedad aporta un objeto que contiene, como cadenas, las URL absolutas de cada archivo XML.

- InputFileCollection

  string InputFileCollection
  Aporta la colección de archivos XML que se deben usar como datos de entrada. Los archivos se identifican por medio de direcciones URL. Se aporta una colección de cadenas y cada cadena es la URL absoluta de un archivo XML de entrada.

- InputFileName

  string InputFileName
  Establece la ubicación, en forma de URL, de los datos XML de entrada que se deben procesar. La cadena dada debe ser una URL absoluta que indique la ubicación exacta del archivo de entrada.

- InputFromText

  string InputFromText
  Aporta el contenido del documento XML que se debe procesar. La cadena dada es el contenido del documento XML que se debe procesar.

- InputTextArray

  object InputTextArray
  Aporta una matriz de direcciones URL de los archivos de texto que se deben usar como datos de entrada. La propiedad aporta un objeto que contiene, como cadenas, las direcciones URL absolutas de cada uno de los archivos de texto.

- InputTextCollection

  string InputTextCollection
  Aporta el contenido de varios archivos XML que se usarán como datos de entrada. Una colección de cadenas, cada una de las cuales es el contenido de un archivo XML de entrada.

- InputXMLFileCollection (OBSOLETO. Usar InputFileCollection en su lugar)

  string InputXMLFileCollection
  Aporta la colección de archivos XML que se usarán como datos de entrada. Los archivos se identifican por medio de una dirección URL. Una cadena de cadenas, cada una de las cuales es la URL absoluta de un archivo XML de entrada.

- InputXMLFileName (OBSOLETO. Usar InputFileCollection en su lugar)

  string InputXMLFileName
  Establece la ubicación, en forma de URL, del documento XML que se debe procesar. La cadena dada debe ser una URL absoluta que indique la ubicación exacta del archivo XML.

- InputXMLFromText (OBSOLETO. Usar InputFileCollection en su lugar)

 © 2018 Altova GmbH

string InputXMLFromText
Aporta el contenido del documento XML que se debe procesar. La cadena dada es el contenido del documento XML que se debe procesar.

- InputXMLTextCollection (OBSOLETO. Usar InputFileCollection en su lugar)

string InputXMLTextCollection
Aporta el contenido de varios archivos XML que se usarán como datos de entrada. Una colección de cadenas, cada una de las cuales es el contenido de un archivo XML de entrada.

- LastErrorMessage

string LastErrorMessage
Recupera una cadena que es el último mensaje de error del motor de RaptorXML.

- ParallelAssessment

bool ParallelAssessment
Habilita o deshabilita la evaluación de la validez de esquemas en paralelo.

- PythonScriptFile

string PythonScriptFile
Establece la ubicación del archivo de script Python. La cadena dada debe ser una URL absoluta que indique la ubicación exacta del archivo Python.

- SchemaFileArray

object **SchemaFileArray**
Aporta la colección de archivos de esquema XML que se usarán como esquemas XML externos. Los archivos se identifican por medio de su dirección URL. La entrada es una colección de cadenas, cada una de las cuales es la URL absoluta de un archivo de esquema XML.

- SchemaFileName

string SchemaFileName(*String* **filePath)**
Establece la ubicación, en forma de URL, del documento de esquema XML que se debe validar. La cadena dada debe ser una URL absoluta que indique la ubicación exacta del archivo de esquema XML.

- SchemaFromText

string SchemaFromText
Aporta el contenido del documento de esquema XML que se debe usar. La cadena dada es el contenido del documento de esquema XML que se debe usar.

- SchemaImports

  ENUMSchemaImports **SchemaImports**
  Especifica cómo se deben realizar las importaciones de esquema, en función de los valores de atributo de los elementos `xs:import`. El tipo de importación se especifica con el literal de `ENUMSchemaImports` (Java, COM/.NET) que se aporta.

- SchemalocationHints

  ENUMLoadSchemalocation **SchemalocationHints**
  Especifica el mecanismo que se debe utilizar para encontrar el esquema. El mecanismo se especifica por medio del literal de `ENUMLoadSchemalocation` (Java, COM/.NET) que se seleccione.

- SchemaMapping

  ENUMSchemaMapping **SchemaMapping**
  Establece qué asignación se debe usar para encontrar el esquema. La asignación se especifica por medio del literal de `ENUMSchemaMapping` (Java, COM/.NET) que se seleccione.

- SchemaTextArray

  object SchemaTextArray
  Aporta el contenido de varios archivos de esquema XML. La entrada es una colección de cadenas, cada una de las cuales es el contenido del documento de esquema XML.

- Streaming

  bool Streaming
  Habilita la validación por transmisión por secuencias. En el modo de transmisión por secuencias, los datos que están almacenados en memoria se minimizan y el procesamiento es más rápido. El valor `true` habilita la transmisión por secuencias y `false` la deshabilita. El valor predeterminado es `true`.

- XIncludeSupport

  bool XIncludeSupport
  Habilita o deshabilita el uso de los elementos `XInclude`. El valor `true` habilita la compatibilidad con XInclude y `false` la deshabilita. El valor predeterminado es `false`.

- XMLValidationMode

  ENUMXMLValidationMode **XMLValidationMode**
  Establece el modo de validación XML, que es un literal de la enumeración `ENUMXMLValidationMode` (Java, COM/.NET) que determina si se debe comprobar la validez del documento o revisar su formato.

 © 2018 Altova GmbH

7.4.1.3 IXSLT

La interfaz **IXSLT** ofrece métodos y propiedades para ejecutar una transformación XSLT 1.0, 2.0 o 3.0. Los resultados se pueden guardar en un archivo o devolverse como cadena de texto. La interfaz también permite pasar parámetros XSLT a la hoja de estilos XSLT. Las direcciones URL de los archivo XML y XSLT se pueden dar como cadenas de texto a través de las propiedades de la interfaz. Otra opción es construir documentos XML y XSLT dentro del código como cadenas de texto.

Nota: Cuando las cadenas de texto de entrada se deben interpretar como URL, deben utilizarse rutas de acceso absolutas. Si se usa una ruta de acceso relativa, debe definir en el módulo de llamada un mecanismo para resolver la ruta de acceso relativa.

Nota: Puede usar el motor XSLT 2.0 o 3.0 de RaptorXML en el modo compatible con versiones anteriores para procesar hojas de estilos XSLT 1.0. No obstante, el resultado puede ser diferente al que se obtendría con el motor XSLT 1.0.

Métodos

Los métodos de la **IXSLT** se describen a continuación. Recuerde que las cadenas de entrada que deben interpretarse como direcciones URL deben dar rutas de acceso absolutas. Si se usa una ruta de acceso relativa, debe definir en el módulo de llamada un mecanismo para resolver la ruta de acceso relativa.

▾ IsValid

```
bool IsValid()
```

- Devuelve el resultado de validar la hoja de estilos XSLT de acuerdo con la especificación XSLT indicada en `ENUMXSLTVersion` (ver la propiedad `EngineVersion`). El resultado es `true` si el XSLT es válido y `false` si no lo es.
- Si ocurre un error, se emite una excepción `RaptorXMLException`. Use la operación `LastErrorMessage` para obtener más información.

▾ Execute

```
bool Execute(string outputPath)
```

- Ejecuta la transformación XSLT de acuerdo con la especificación XSLT indicada en `ENUMXSLTVersion` (ver la propiedad `EngineVersion`) y guarda el resultado en un archivo de salida.
- El archivo de salida viene dado por `bstrResultFileName`, que es una cadena que indica la URL del archivo de salida.
- El resultado es `true` si la transformación finaliza correctamente y `false` si se producen errores.
- Si ocurre un error, se emite una excepción `RaptorXMLException`. Use la operación `LastErrorMessage` para obtener más información.

▾ ExecuteAndGetResultAsString

bool Execute()

- Ejecuta la transformación XSLT de acuerdo con la especificación indicada en ENUMXSLTVersion (ver la propiedad EngineVersion) y devuelve el resultado de la transformación como cadena de texto.
- Este método no produce archivos de resultados adicionales como gráficos ni resultados secundarios. Tampoco almacena resultados binarios como archivos OOXML. Si necesita archivos de salida adicionales, utilice el método Execute.
- Si ocurre un error durante la transformación, se emite una excepción RaptorXMLException. Use la operación LastErrorMessage para obtener más información.

ExecuteAndGetResultAsStringWithBaseOutputURI

bool Execute(string baseURI)

- Ejecuta la transformación XSLT de acuerdo con la especificación indicada en ENUMXSLTVersion (ver la propiedad EngineVersion) y devuelve el resultado de la transformación como cadena de texto en la ubicación definida por el URI base (la cadena bstrBaseURI).
- Este método no produce archivos de resultados adicionales como gráficos ni resultados secundarios. Tampoco almacena resultados binarios como archivos OOXML. Si necesita archivos de salida adicionales, utilice el método Execute.
- Si ocurre un error durante la transformación, se emite una excepción RaptorXMLException. Use la operación LastErrorMessage para obtener más información.

AddExternalParameter

void AddExternalParameter(string paramName, string paramValue)

- Añade el nombre y el valor de un parámetro externo: bstrName y bstrValue son cadenas.
- Cada parámetro externo y su valor se puede especificar en una llamada distinta al método. Los parámetros se deben declarar en el documento XSLT y si quiere puede añadir una declaración de tipo. Sea cual sea la declaración de tipo del documento XSLT, no es necesario ningún delimitador concreto si el valor del parámetro se indica con AddExternalParameter.

ClearExternalParameterList

void ClearExternalParameterList()

- Borra la lista de parámetros externos creada con el método AddExternalParameter.

Propiedades

Las propiedades de la interfaz **IXSLT** se describen a continuación por orden alfabético. Recuerde que las cadenas de entrada que deben interpretarse como direcciones URL deben dar rutas de acceso absolutas. Si se usa una ruta de acceso relativa, debe definir en el módulo de llamada un mecanismo para resolver la ruta de acceso relativa.

 © 2018 Altova GmbH

- AdditionalOutputs

  string AdditionalOutputs
  Devuelve los archivos de salida adicionales del último trabajo que se ejecutó.

- ChartExtensionsEnabled

  bool ChartsExtensionsEnabled
  Habilita o deshabilita las funciones de extensión de Altova para gráficos. El valor `true` habilita las extensiones para gráficos. El valor `false` las deshabilita. El valor predeterminado es `true`.

- DotNetExtensionsEnabled

  bool DotNetExtensionsEnabled
  Habilita o deshabilita las funciones de extensión .NET. El valor `true` habilita las extensiones .NET. El valor `false` las deshabilita. El valor predeterminado `true`.

- EngineVersion

  ***ENUMXSLTVersion* EngineVersion**
  Indica qué versión XSLT se debe usar (1.0, 2.0 o 3.0). El valor de la propiedad es un literal de `ENUMXSLTVersion`.

- IndentCharacters

  string IndentCharacters
  Aporta la cadena de caracteres que se usará para aplicar sangría en el documento de salida.

- InitialTemplateMode

  string InitialTemplateMode
  Establece el modo inicial para el procesamiento XSLT. Se procesarán aquellas plantillas cuyo valor de modo sea igual a la cadena indicada.

- InputXMLFileName

  string InputXMLFileName
  Establece la ubicación, en forma de URL, del documento XML que se debe procesar. La cadena dada debe ser una URL absoluta que indique la ubicación exacta del archivo XML.

- InputXMLFromText

  string InputXMLFromText
  Aporta el contenido del documento XML que se debe procesar. La cadena dada es el contenido del documento XML que se debe procesar.

JavaBarcodeExtensionLocation

`string JavaBarcodeExtensionLocation`

Especifica la ubicación del archivo de extensión de código de barras. Para obtener más información consulte el apartado dedicado a las funciones de extensión de Altova para trabajar con códigos de barras. La cadena dada debe ser una URL absoluta que indique la ubicación base del archivo que se debe usar.

JavaExtensionsEnabled

`bool JavaExtensionsEnabled`

Habilita o deshabilita las funciones de extensión Java. El valor `true` habilita las extensiones Java y `false` las deshabilita. El valor predeterminado es `true`.

LastErrorMessage

`string LastErrorMessage`

Recupera una cadena que es el último mensaje de error del motor de RaptorXML.

LoadXMLWithPSVI

`bool LoadXMLWithPSVI`

Habilita la validación de archivos XML de entrada y genera información posterior a la validación con esquema sobre dichos archivos. El valor `true` habilita la validación XML y genera información posterior a la validación con esquema para los archivos XML. El valor `false` deshabilita la validación. El valor predeterminado es `true`.

MainOutput

`string MainOutput`

Devuelve el resultado principal del último trabajo que se ejecutó.

NamedTemplateEntryPoint

`string NamedTemplateEntryPoint`

Especifica el nombre, como cadena de texto, de la plantilla con nombre que debe utilizarse como punto de entrada de la transformación.

SchemaImports

`ENUMSchemaImports` **`SchemaImports`**

Especifica cómo se deben realizar las importaciones de esquema, en función de los valores de atributo de los elementos `xs:import`. El tipo de importación se especifica con el literal de `ENUMSchemaImports` (Java, COM/.NET) que se aporta.

SchemalocationHints

`ENUMLoadSchemalocation` **`SchemalocationHints`**

 © 2018 Altova GmbH

Especifica el mecanismo que se debe utilizar para encontrar el esquema. El mecanismo se especifica por medio del literal de `ENUMLoadSchemalocation` (Java, COM/.NET) que se seleccione.

- SchemaMapping

  ENUMSchemaMapping **SchemaMapping**
  Establece qué asignación se debe usar para encontrar el esquema. La asignación se especifica por medio del literal de `ENUMSchemaMapping` (Java, COM/.NET) que se seleccione.

- StreamingSerialization

  bool StreamingSerialization
  Habilita la serialización de secuencias de datos. En el modo de transmisión por secuencias, los datos almacenados en memoria se minimizan y el procesamiento es más rápido. El valor `true` habilita la serialización de secuencias de datos; `false` la deshabilita.

- XincludeSupport

  bool XincludeSupport
  Habilita o deshabilita el uso de los elementos `XInclude`. El valor `true` habilita la compatibilidad con XInclude y `false` la deshabilita. El valor predeterminado es `false`.

- XMLValidationErrorAsWarning

  bool XMLValidationErrorAsWarning
  Define si los errores de validación XML se tratan como advertencias o no. Toma el valor booleano `true` o `false`.

- XMLValidationMode

  ENUMXMLValidationMode **XMLValidationMode**
  Establece el modo de validación XML, que es un literal de la enumeración `ENUMXMLValidationMode` (Java, COM/.NET) que determina si se debe comprobar la validez del documento o revisar su formato.

- XSDVersion

  ENUMXSDVersion **XSDVersion**
  Establece la versión de XML Schema con la que se debe validar el documento XML. Es un literal de la enumeración `ENUMXSDVersion` (Java, COM/.NET).

- XSLFileName

  string XSLFileName
  Especifica qué archivo XSLT se debe usar para la transformación. La cadena indicada debe ser una URL absoluta que dé la ubicación del archivo XSLT que se debe usar.

- XSLFromText

  `string XSLFromText`

  Ofrece, en forma de cadena de texto, el contenido del documento XSLT que se debe usar para la transformación.

7.4.1.4 IXQuery

La interfaz **IXQuery** ofrece métodos y propiedades para ejecutar un documento XQuery 1.0 o XQuery 3.0. Los resultados se pueden guardar en un archivo o devolverse como cadena de texto. La interfaz también permite pasar variables XQuery externas al documento XQuery. Las URL de los archivos XQuery y XML se pueden dar como cadenas de texto a través de las propiedades de la interfaz. Otra opción es construir los documentos XML y XQuery dentro del código como cadenas de texto.

Note: Cuando las cadenas de texto de entrada se deben interpretar como URL, deben utilizarse rutas de acceso absolutas. Si se usa una ruta de acceso relativa, debe definir en el módulo de llamada un mecanismo para resolver la ruta de acceso relativa.

Métodos

Los métodos de la **IXQuery** se describen a continuación. Recuerde que las cadenas de entrada que deben interpretarse como direcciones URL deben dar rutas de acceso absolutas. Si se usa una ruta de acceso relativa, debe definir en el módulo de llamada un mecanismo para resolver la ruta de acceso relativa.

- IsValid

  `bool IsValid()`

  - Devuelve el resultado de validar el documento XQuery de acuerdo con la especificación XQuery indicada en `ENUMXQueryVersion` (ver la propiedad `EngineVersion`). El resultado es `true` si el documento es válido y `false` si no lo es.
  - Si ocurre un error, se emite una excepción `RaptorXMLException`. Use la operación `LastErrorMessage` para obtener más información..

- IsValidUpdate

  `bool IsValidUpdate()`

  - Devuelve el resultado de validar el documento XQuery Update de acuerdo con la especificación XQuery indicada en `ENUMXQueryVersion` (ver la propiedad `EngineVersion`). El resultado es `true` si finaliza correctamente y `false` si no.
  - Si ocurre un error, se emite una excepción `RaptorXMLException`. Use la operación `LastErrorMessage` para obtener más información.

 © 2018 Altova GmbH

Execute

```
bool Execute(string outputFile)
```

- Ejecuta el documento XQuery de acuerdo con la especificación XQuery indicada en ENUMXQueryVersion (ver la propiedad EngineVersion), y guarda el resultado en un archivo de salida.
- El archivo de salida viene dado por bstrOutputFile, que es una cadena que indica la URL del archivo de salida.
- El valor binario true se devuelve si la operación finaliza correctamente y false si hay errores.
- Si ocurre un error durante la transformación, se emite una excepción RaptorXMLException. Use la operación LastErrorMessage para obtener más información.

ExecuteAndGetResultAsString

```
string ExecuteAndGetResultAsString()
```

- Ejecuta la transformación XQuery de acuerdo con la especificación XQuery indicada en ENUMXQueryVersion (ver la propiedad EngineVersion) y devuelve el resultado de la transformación como cadena de texto.
  - Este método no produce archivos de resultados adicionales como gráficos ni resultados secundarios. Tampoco almacena resultados binarios como archivos OOXML. Si necesita archivos de salida adicionales, utilice el método Execute.
- Si ocurre un error durante la transformación, se emite una excepción RaptorXMLException. Use la operación LastErrorMessage para obtener más información.

ExecuteUpdate

```
bool ExecuteUpdate(string outputFile)
```

- Ejecuta el documento XQuery Update de acuerdo con la especificación XQuery Update indicada en ENUMXQueryVersion (ver la propiedad EngineVersion), y guarda el resultado en un archivo de salida.
- El archivo de salida viene dado por bstrOutputFile, que es una cadena que indica la URL del archivo de salida.
- El valor binario true se devuelve si la operación finaliza correctamente y false si hay errores.
- Si ocurre un error durante la transformación, se emite una excepción RaptorXMLException. Use la operación LastErrorMessage para obtener más información.

ExecuteUpdateAndGetResultAsString

```
string ExecuteAndGetResultAsString()
```

- Ejecuta la transformación XQuery Update de acuerdo con la especificación XQuery Update indicada en ENUMXQueryVersion (ver la propiedad EngineVersion) y devuelve el resultado de la transformación como cadena de texto.
  - Este método no produce archivos de resultados adicionales como gráficos ni resultados secundarios. Tampoco almacena resultados binarios como archivos OOXML. Si necesita archivos de salida adicionales, utilice el método Execute.

- Si ocurre un error durante la transformación, se emite una excepción `RaptorXMLException`. Use la operación `LastErrorMessage` para obtener más información.

AddExternalVariable

```
void AddExternalVariable(string varName, string varValue)
```

- Añade el nombre y valor de una variable externa: `bstrName` y `bstrValue` son cadenas de texto.
- Cada variable externa y su valor deben indicarse en una llamada distinta al método. Las variables deben declararse en el documento XQuery y si quiere puede incluir una declaración de tipo. Si el valor de la variable es una cadena de texto, ponga el valor entre comillas simples.

ClearExternalVariableList

```
void ClearExternalVariableList()
```

- Borra la lista de variables externas creadas con el método `AddExternalVariable`.

Propiedades

Las propiedades de la interfaz **`IXQuery`** se describen a continuación por orden alfabético. Recuerde que las cadenas de entrada que deben interpretarse como direcciones URL deben dar rutas de acceso absolutas. Si se usa una ruta de acceso relativa, debe definir en el módulo de llamada un mecanismo para resolver la ruta de acceso relativa.

ChartExtensionsEnabled

```
bool ChartsExtensionsEnabled
```

Habilita o deshabilita las funciones de extensión de Altova para gráficos. El valor `true` habilita las extensiones para gráficos. El valor `false` las deshabilita. El valor predeterminado es `true`.

DotNetExtensionsEnabled

```
bool DotNetExtensionsEnabled
```

Habilita o deshabilita las funciones de extensión .NET. El valor `true` habilita las extensiones .NET. El valor `false` las deshabilita. El valor predeterminado `true`.

EngineVersion

```
ENUMXQueryVersion EngineVersion
```

Indica qué versión de XQuery se debe usar (1.0 o 3.0). El valor de la propiedad es un literal de `ENUMXQueryVersion`.

IndentCharacters

string IndentCharacters
Aporta la cadena de caracteres que se usará para aplicar sangría en el documento de salida.

- InputXMLFileName

string InputXMLFileName
Establece la ubicación, en forma de URL, del documento XML que se debe procesar. La cadena dada debe ser una URL absoluta que indique la ubicación exacta del archivo XML.

- InputXMLFromText

string InputXMLFromText
Aporta el contenido del documento XML que se debe procesar. La cadena dada es el contenido del documento XML que se debe procesar.

- JavaBarcodeExtensionLocation

string JavaBarcodeExtensionLocation
Especifica la ubicación del archivo de extensión de código de barras. Para obtener más información consulte el apartado dedicado a las funciones de extensión de Altova para trabajar con códigos de barras. La cadena dada debe ser una URL absoluta que indique la ubicación base del archivo que se debe usar.

- JavaExtensionsEnabled

bool JavaExtensionsEnabled
Habilita o deshabilita las funciones de extensión Java. El valor `true` habilita las extensiones Java y `false` las deshabilita. El valor predeterminado es `true`.

- KeepFormatting

bool KeepFormatting
Especifica si se debe conservar el formato del documento original (en la medida de lo posible) o no. El valor `true` conserva el formato; `false` no conserva el formato. El valor predeterminado es `true`.

- LastErrorMessage

string LastErrorMessage
Recupera una cadena que es el último mensaje de error del motor de RaptorXML.

- LoadXMLWithPSVI

bool LoadXMLWithPSVI
Habilita la validación de archivos XML de entrada y genera información posterior a la validación con esquema sobre dichos archivos. El valor `true` habilita la validación XML y genera información posterior a la validación con esquema para los archivos XML. El valor `false` deshabilita la validación. El valor predeterminado es `true`.

- OutputEncoding

  string OutputEncoding
  Establece la codificación del documento de salida. Use un nombre de codificación IANA oficial (p. ej. `UTF-8`, `UTF-16`, `US-ASCII`, `ISO-8859-1`) como cadena de texto.

- OutputIndent

  bool OutputIndent
  Habilita o deshabilita la sangría en el documento de salida. El valor `true` habilita la sangría; `false` la deshabilita.

- OutputMethod

  string OutputMethod
  Especifica la serialización del documento de salida. Los valores válidos son: `xml | xhtml | html | text`. El valor predeterminado es `xml`.

- OutputOmitXMLDeclaration

  bool OutputOmitXMLDeclaration
  Habilita o deshabilita la inclusión de la declaración XML en el documento de salida. El valor `true` omite la declaración; `false` la incluye. El valor predeterminado es `false`.

- UpdatedXMLWriteMode

  ENUMXQueryUpdatedXML UpdatedXMLWriteMode
  Especifica cómo se deben gestionar las actualizaciones en el archivo XML. El valor de la propiedad es un literal de `ENUMXQueryUpdatedXML`.

- XincludeSupport

  bool XincludeSupport
  Habilita o deshabilita el uso de los elementos `XInclude`. El valor `true` habilita la compatibilidad con XInclude y `false` la deshabilita. El valor predeterminado es `false`.

- XMLValidationErrorAsWarning

  bool XMLValidationErrorAsWarning
  Define si los errores de validación XML se tratan como advertencias o no. Toma el valor booleano `true` o `false`.

- XMLValidationMode

  ***ENUMXMLValidationMode* XMLValidationMode**
  Establece el modo de validación XML, que es un literal de la enumeración `ENUMXMLValidationMode` (Java, COM/.NET) que determina si se debe comprobar la validez del documento o revisar su formato.

 © 2018 Altova GmbH

- XQueryFileName

  string XQueryFileName
  Indica qué archivo XQuery se debe usar. La cadena indicada debe ser una URL absoluta que dé la ubicación del archivo XQuery que se debe usar.

- XQueryFromText

  string XQueryFromText
  Suministra, como cadena de texto, el contenido del archivo XQuery que se debe usar.

- XSDVersion

  ***ENUMXSDVersion* XSDVersion**
  Establece la versión de XML Schema con la que se debe validar el documento XML. Es un literal de la enumeración `ENUMXSDVersion` (Java, COM/.NET).

7.4.1.5 IXBRL

La interfaz **IXBRL** ofrece métodos para validar documentos de instancia XBRL y taxonomías XBRL, así como fórmulas. Los resultados son los valores booleanos `true` o `false`. La interfaz también permite pasar parámetros de fórmula para la evaluación de las mismas. Las aserciones de fórmulas y el resultado también se puede consultar y se pueden devolver como cadenas de texto. Las propiedades definen los parámetros de la interfaz.

Note: Cuando las cadenas de texto de entrada se deben interpretar como URL, deben utilizarse rutas de acceso absolutas. Si se usa una ruta de acceso relativa, debe definir en el módulo de llamada un mecanismo para resolver la ruta de acceso relativa.

Estructuras

Se define la siguiente estructura.

```
public struct XBRLParamValuePair
{
    String ParamType;
    String ParamValue;
};
```

Métodos

Los métodos de la **IXBRL** se describen a continuación. Cuando las cadenas de texto de entrada se deben interpretar como URL, deben utilizarse rutas de acceso absolutas. Si se usa una ruta de acceso relativa, debe definir en el módulo de llamada un mecanismo para resolver la ruta de

acceso relativa.

IsValid

```
bool IsValid(ENUMXBRLValidationType type)
```

- Devuelve el resultado de validar el documento de instancia XBRL o la taxonomía.
- `nType` es el valor de `ENUMXBRLValidationType`. El tipo de validación indica si lo que se valida es un documento XBRL de instancia o una taxonomía XBRL. El valor predeterminado es `eValidateXBRLAny`, que indica que RaptorXML debe determinar el tipo de documento automáticamente
- Si se produce un error durante la validación, se emite una excepción `RaptorXMLException`. Use la operación `LastErrorMessage` para obtener más información.

EvaluateFormula

```
bool EvaluateFormula()
```

- Evalúa las fórmulas XBRL de un documento XBRL de instancia. Devuelve `true` si son válidas y `false` si alguna fórmula no es válida.
- Si se produce un error durante la evaluación, se emite una excepción `RaptorXMLException`. Use la operación `LastErrorMessage` para obtener más información.

GenerateTables

```
bool GenerateTables()
```

- Genera la tabla XBRL en un documento XBRL de instancia. Devuelve `true` si la generación es correcta y `false` si no lo es.
- Si se produce un error durante la ejecución, se emite una excepción `RaptorXMLException`. Use la operación `LastErrorMessage` para obtener más información.

AddFormulaArrayParameter

```
void AddFormulaArrayParameter(string defaultType, string name, object[] values)
```

- Añade un parámetro de matriz utilizado en el proceso de evaluación de fórmulas.
- Todos los argumentos son cadenas: `sDefaultType` es el tipo de datos predeterminado de valores de no pares dentro de los valores de matriz. El tipo predeterminado es `xs:string`; `sName` es el nombre de parámetro; `variantValues` es una matriz de valor y pares valor/tipo de datos.
- Para más información y ejemplos de código consulte el apartado Parámetros de fórmulas XBRL.

AddFormulaParameter [OBSOLETO]

```
void AddFormulaParameter(string type, string name, string value, string namespace)
```

- Añade un parámetro para la evaluación de fórmulas. Es obsoleto.
- Todos los argumentos son cadenas: `sType` es el tipo de datos del parámetro; `sName` es el nombre del parámetro; `sValue` es el valor del parámetro; y `sNamespace` es el

 © 2018 Altova GmbH

espacio de nombres del parámetro.
- Cada parámetro se debe especificar en una llamada al método distinta.

AddFormulaParameter

```
void AddFormulaParameter(string type, string name, string value, string namespace="")
```

- Añade un parámetro para la evaluación de fórmulas.
- Todos los argumentos son cadenas: `sType` es el tipo de datos del parámetro; `sName` es el nombre del parámetro; `sValue` es el valor del parámetro; y `sNamespace` es el espacio de nombres del parámetro y es la cadena vacía.
- Cada parámetro se debe especificar en una llamada al método distinta.

AddFormulaParameterNamespace

```
void AddFormulaParameterNamespace(string prefix, string uri)
```

- Define un espacio de nombres utilizado en los QName de los nombres, tipos y valores de parámetro.
- Todos los argumentos son cadenas: `sPrefix` es el prefijo de espacio de nombres de los valores pasados a `AddFormulaArrayParameter`; `sURI` es el URI de espacio de nombres.
- Cada parámetro se debe especificar en una llamada al método distinta.

AddIXBRLTransformationRegistryLimit

```
void AddIXBRLTransformationRegistryLimit(string limit)
```

- Limita los registros de transformación Inline XBRL disponibles a la versión especificada.
- `sLmit` es la versión a la que se limitan los registros Inline XBRL disponibles.

ClearFormulaParameterList

```
void ClearFormulaParameterList()
```

- Borra la lista de parámetros de fórmula creados con el método AddFormulaParameter.

ClearIXBRLTransformationRegistryLimit

```
void ClearIXBRLTransformationRegistryLimit()
```

- Elimina la limitación de versiones disponibles de los registros de transformación Inline XBRL.

ReadFormulaAssertions

```
void ReadFormulaAssertions()
```

- Lee las aserciones de fórmulas del archivo que se está evaluando.

ReadFormulaOutput

`void ReadFormulaOutput()`

- Lee el resultado de las aserciones de fórmulas del archivo.

Propiedades

Las propiedades de la interfaz **IXBRL** se describen a continuación por orden alfabético. La tabla organiza las propiedades por grupos para facilitar su consulta. Recuerde que las cadenas de entrada que deben interpretarse como direcciones URL deben dar rutas de acceso absolutas. Si se usa una ruta de acceso relativa, debe definir en el módulo de llamada un mecanismo para resolver la ruta de acceso relativa.

AddAssertionForProcessing

`string AddAssertionForProcessing`

Limita la evaluación de aserciones a la aserción dada. Llámele varias veces para especificar más de una aserción. Use `##none` para no evaluar ninguna aserción y `##all` para evaluar todas.

AddAssertionSetForProcessing

`string AddAssertionSetForProcessing`

Limita la evaluación de conjuntos de aserciones al conjunto de aserciones dado. Llámele varias veces para especificar más de un conjunto de aserciones. Use `##none` para no evaluar ningún conjunto de aserciones y `##all` para evaluar todos.

AddTableForProcessing

`string AddTableForProcessing`

Limita la generación de tablas a la tabla dada. Llámele varias veces para especificar más de una tabla. Use `##none` para no generar ninguna tabla y `##all` para generar todas.

ConceptLabelLinkrole

`string ConceptLabelLinkrole`

Especifica la función de vínculo extendido preferida a la hora de representar etiquetas de concepto.

ConceptLabelRole

`string ConceptLabelRole`

Especifica la función de etiqueta preferida a la hora de representar etiquetas de concepto. Predeterminada: `http://www.xbrl.org/2008/role/label`.

DimensionExtensionEnabled

`bool DimensionExtensionEnabled`

 © 2018 Altova GmbH

Habilita o deshabilita la validación con la extensión XBRL Dimension. El valor `true` habilita la validación la extensión XBRL Dimension; `false` la deshabilita. El valor predeterminado es `true`.

- EnableIXBRLValidateTarget

  `bool EnableIXBRLValidateTarget`
  Habilita la validación XBRL del documento XBRL que se genera. El valor predeterminado es: `true`.

- EvaluateReferencedParametersOnly

  `bool EvaluateReferencedParametersOnly`
  Si tiene el valor `false`, obliga a evaluar todos los parámetros incluso si no se les hace referencia en ninguna fórmula, aserción o tabla. El valor predeterminado es `true`.

- FormulaAssertionsAsXML

  `bool FormulaAssertionsAsXML`
  Habilita el formato XML del archivo de aserciones de fórmulas cuando RaptorXML se ejecuta con las aserciones habilitadas. El valor `true` habilita el formato XML; el valor `false` genera resultados en formato JSON. El valor predeterminado es `false`.

- FormulaAssertionsOutput

  `string FormulaAssertionsOutput`
  Indica la ubicación del archivo de salida de las aserciones de fórmulas. Es obligatorio indicar una ruta de acceso completa.

- FormulaExtensionEnabled

  `bool FormulaExtensionEnabled`
  Habilita o deshabilita la validación con la extensión XBRL Formula. El valor `true` habilita la validación la extensión XBRL Formula; `false` la deshabilita. El valor predeterminado es `true`.

- FormulaOutput

  `string FormulaOutput`
  Indica la ubicación del resultado del archivo de evaluación de fórmulas XBRL. Es obligatorio indicar una ruta de acceso completa.

- FormulaParameterFile

  `string FormulaParameterFile`
  Indica la ubicación del archivo de parámetros de fórmulas. Es obligatorio indicar una ruta de acceso completa.

- FormulaPreloadSchemas

bool FormulaPreloadSchemas
Define si se cargan o no previamente los esquemas de XBRL Formula. El valor `true` habilita la carga previa de estos esquemas. El valor predeterminado es `false`, que no carga los esquemas previamente.

- GenericLabelLinkrole

string GenericLabelLinkrole
Especifica la función de vínculo extendido preferida a la hora de representar etiquetas genéricas.

- GenericLabelRole

string GenericLabelRole
Especifica la función de etiqueta preferida a la hora de representar etiquetas genéricas. Predeterminada: `http://www.xbrl.org/2008/role/label`.

- InputFileArray

object InputFileArray
Establece el conjunto de archivos XBRL que se usarán como datos de entrada/instancias. Este conjunto es un objeto que contiene las cadenas de las URL absolutas de los archivos de entrada.

- InputFileName

string InputFileName
Indica el nombre de archivo y la ubicación del archivo de instancia XBRL. La cadena dada debe ser una URL absoluta. Las rutas de acceso relativas se pueden resolver en relación a una ubicación base, dependiendo del mecanismo definido en el módulo de llamada.

- InputFromText

string InputFromText
Suministra el contenido del documento XBRL de entrada como texto.

- InputTextArray

object InputTextArray
Establece el conjunto de archivos de texto que se usarán como datos de entrada. Este conjunto es un objeto que contiene las cadenas de las URL absolutas de los archivos de entrada.

- IXBRLOutput

string IXBRLOutput
Establece la ubicación de salida del archivo XBRL de salida que se genera. La cadena dada almacena la URL absoluta de la ubicación de salida.

 © 2018 Altova GmbH

- IXBRLTreatAsDocumentSet

  bool IXBRLTreatAsDocumentSet
  Establece si todas las entradas se tratan como un solo conjunto de documentos Inline XBRL. El valor predeterminado es: `false`.

- IXBRLUriStrategy

  ENUMIXBRLUriStrategy IXBRLUriStrategy
  Especifica cómo se deben transformar los URI Inline XBRL en el documento XBRL que se genera. Toma el literal `ENUMIXBRLUriStratgey` que está seleccionado.

- IXBRLVersion

  ENUMIXBRLVersion IXBRLVersion
  Especifica qué versión de Inline XBRL se debe usar. Toma el literal `ENUMIXBRLVersion` que está seleccionado.

- LabelLang

  string LabelLang
  Especifica el idioma de etiqueta preferido a la hora de representar etiquetas. Predeterminado: `en`.

- LastErrorMessage

  string LastErrorMessage
  Recupera una cadena que es el último mensaje de error del motor de RaptorXML.

- ParallelAssessment

  bool ParallelAssessment
  Habilita/deshabilita la evaluación de la validez de esquemas en paralelo.

- PreloadSchemas

  bool PreloadSchemas
  Define si se cargan previamente los esquemas XBRL 2.1 o no. El valor `true` carga estos esquemas previamente. El valor predeterminado es `true`.

- PythonScriptFile

  string PythonScriptFile
  Indica qué archivo de script Python ofrece procesamiento adicional para el archivo XML o XSD sometido a validación. La cadena indicada debe ser una URL absoluta que dé la ubicación base del archivo de script Python.

- SchemaImports

  ENUMSchemaImports **SchemaImports**
  Especifica cómo se deben realizar las importaciones de esquema, en función de los valores de atributo de los elementos `xs:import`. El tipo de importación se especifica con el literal de `ENUMSchemaImports` (Java, COM/.NET) que se aporta.

- SchemalocationHints

  ENUMLoadSchemalocation **SchemalocationHints**
  Especifica el mecanismo que se debe utilizar para encontrar el esquema. El mecanismo se especifica por medio del literal de `ENUMLoadSchemalocation` (Java, COM/.NET) que se seleccione.

- SchemaMapping

  ENUMSchemaMapping **SchemaMapping**
  Establece qué asignación se debe usar para encontrar el esquema. La asignación se especifica por medio del literal de `ENUMSchemaMapping` (Java, COM/.NET) que se seleccione.

- TableEliminateEmptyRows

  bool TableEliminateEmptyRows
  Habilita la eliminación de filas/columnas vacías en la tabla generada en formato de salida HTML.

- TableExtensionEnabled

  bool TableExtensionEnabled
  Habilita/deshabilita la extensión XBRL Table 1.0.

- TableLinkbaseNamespace

  string TableLinkbaseNamespace
  Habilita la carga de bases de enlaces de tabla escritas con una especificación borrador previa. El valor de cadena dado especifica la base de enlaces de tabla. La validación, la resolución y el diseño de las bases de enlaces de tabla, sin embargo, se realiza siempre según la recomendación Table Linkbase 1.0 del 18 de marzo de 2014. Use `##detect` para habilitar la detección automática. Se reconocen estos valores:

  ```
  ##detect
  http://xbrl.org/PWD/2013-05-17/table
  http://xbrl.org/PWD/2013-08-28/table
  http://xbrl.org/CR/2013-11-13/table
  http://xbrl.org/PR/2013-12-18/table
  http://xbrl.org/2014/table
  ```

- TableOutput

 © 2018 Altova GmbH

string TableOutput
Especifica el nombre de archivo y la ubicación del archivo de salida de la tabla generada. La cadena dada debe ser la ruta de acceso completa del archivo de salida.

- TableOutputFormat

ENUMTableOutputFormat TableOutputFormat
Especifica el formato del archivo de salida de generación de tablas.

- TablePreloadSchemas

bool TablePreloadSchemas
Habilita/deshabilita la carga previa de los esquemas de la especificación XBRL Table 1.0.

- TreatXBRLInconsistenciesAsErrors

bool TreatXBRLInconsistenciesAsErrors
El valor `true` detiene la validación XBRL y define el archivo como no válido si se encuentran en él incoherencias de acuerdo con la especificación XBRL 2.1. El valor predeterminado es `false`: las incoherencias XBRL definidas en la especificación XBRL 2.1 no se consideran errores.

- XIncludeSupport

bool XIncludeSupport
Habilita o deshabilita el uso de los elementos `XInclude`. El valor `true` habilita la compatibilidad con XInclude y `false` la deshabilita. El valor predeterminado es `false`.

7.4.2 Enumeraciones

Las enumeraciones definidas son estas:

ENUMAssessmentMode

ENUMErrorFormat

ENUMLoadSchemalocation

ENUMQueryVersion

ENUMSchemaImports

ENUMSchemaMapping

ENUMValidationType

ENUMWellformedCheckType

ENUMXBRLValidationType

ENUMXMLValidationMode

ENUMXQueryVersion

ENUMXSDVersion

ENUMXSLTVersion

7.4.2.1 ENUMAssessmentMode

Descripción

Contiene literales de enumeración que definen el modo de evaluación del validador XML: `Strict` o `Lax`.

Utilizada por

Interfaz	*Operación*
IXMLValidator	AssessmentMode

Literales de la enumeración

```
eAssessmentModeStrict        = 0
eAssessmentModeLax           = 1
```

eAssessmentModeStrict

Establece el modo de evaluación de la validez del esquema en `Strict`. Es el valor predeterminado.

eAssessmentModeLax

Establece el modo de evaluación de la validez del esquema en `Lax`.

7.4.2.2 ENUMErrorFormat

Descripción

Contiene literales de enumeración que especifican el formato de los errores de salida.

Utilizada por

Interfaz	*Operación*

 © 2018 Altova GmbH

IServer	ErrorFormat

Literales de la enumeración

```
eFormatText              = 0
eFormatShortXML          = 1
eFormatLongXML           = 2
```

eFormatText
Establece el formato de los errores de salida en `Text`. Es el valor predeterminado.

eFormatShortXML
Establece el formato de los errores de salida en `ShortXML`. Este formato es una versión abreviada del formato `LongXML`.

eFormatLongXML
Establece el formato de los errores de salida en `LongXML`. Este formato es el que ofrece información más detallada.

7.4.2.3 ENUMLoadSchemalocation

Descripción
Contiene literales de enumeración que indican cómo determinar la ubicación del esquema.

Utilizada por

Interfaz	*Operación*
IXBRL	SchemalocationHints
IXMLValidator	SchemalocationHints
IXSLT	SchemalocationHints

Literales de la enumeración

```
eSHLoadBySchemalocation          = 0
eSHLoadByNamespace               = 1
eSHLoadCombiningBoth             = 2
eSHLoadIgnore                    = 3
```

eSHLoadBySchemalocation
Asigna a Load Schemalocation el valor `LoadBySchemalocation`. Usa la URL de schema location de los atributos `xsi:schemaLocation` y `xsi:noNamespaceSchemaLocation` de los documentos XML o XBRL de instancia. Es el **valor predeterminado**.

eSHLoadByNamespace
Asigna a Load Schemalocation el valor `LoadByNamespace`. Use la parte de espacio de nombres del atributo `xsi:schemaLocation` (en el caso de `xsi:noNamespaceSchemaLocation` es una cadena vacía) y busca el esquema a través de la asignación de catálogo.

eSHLoadCombiningBoth
Asigna a Load Schemalocation el valor `CombiningBoth`. Si el espacio de nombres o la URL tienen una asignación de catálogo, esta asignación de catálogo se utiliza. Si ambas tienen una asignación de catálogo, entonces es el valor del parámetro `ENUMSchemaMapping` lo que decide cuál de las asignaciones se utiliza. Si ninguna tiene una asignación de catálogo, se usa la URL.

eSHLoadIgnore
Asigna a Load Schemalocation el valor `LoadIgnore`. Si el valor del parámetro es `eSHLoadIgnore`, se ignoran los atributos `xsi:schemaLocation` y `xsi:noNamespaceSchemaLocation`.

7.4.2.4 ENUMQueryVersion

Descripción
Contiene literales de enumeración que especifican la versión XQuery que se debe usar: XQuery 1.0 o 3.0.

Literales de la enumeración

```
eXQVersion10        = 1
eXQVersion30        = 3
```

eXQVersion10
Establece que la versión XQuery es XQuery 1.0.

eXQVersion30
Establece que la versión XQuery es XQuery 3.0.

 © 2018 Altova GmbH

7.4.2.5 ENUMSchemaImports

Descripción
Contiene los literales de enumeración que definen el comportamiento de los elementos `xs:import`. El elemento `xs:import` tiene los atributos opcionales `namespace` y `schemaLocation`.

Utilizada por

Interfaz	*Operación*
IXBRL	SchemaImports
IXMLValidator	SchemaImports
IXSLT	SchemaImports

Literales de la enumeración

```
eSILoadBySchemalocation                    = 0
eSILoadPreferringSchemalocation            = 1
eSILoadByNamespace                         = 2
eSICombiningBoth                           = 3
eSILicenseNamespaceOnly                    = 4
```

eSILoadBySchemalocation
Asigna a Schema Import el valor `LoadBySchemalocation`. El valor del atributo `schemaLocation` se usa para buscar el esquema, teniendo en cuenta las asignaciones de catálogo. Si está presente el atributo `namespace`, se importa el espacio de nombres.

eSILoadPreferringSchemalocation
Asigna a Schema Import el valor `LoadPreferringSchemalocation`. Si está presente el atributo `schemaLocation`, éste se utiliza teniendo en cuenta las asignaciones de catálogo. Si no está presente el atributo `schemaLocation`, se usa el valor del atributo `namespace` a través de una asignación de catálogo. Este literal es el **valor predeterminado** de la enumeración.

eSILoadByNamespace
Asigna a Schema Import el valor `LoadByNamespace`. El valor del atributo `namespace` se utiliza para encontrar el esquema a través de una asignación de catálogo.

eSICombiningBoth
Asigna a Schema Import el valor `CombiningBoth`. Si el atributo `namespace` o `schemaLocation` tiene una asignación de catálogo, esta asignación de catálogo se utiliza. Si ambos tienen una asignación de catálogo, entonces es el valor del parámetro `ENUMSchemaMapping` lo que decide cuál de las asignaciones se utiliza. Si ninguna tiene una asignación de catálogo, se usa el valor

del atributo `schemaLocation` (que debería ser una URL).

eSILicenseNamespaceOnly
Asigna a Schema Import el valor `LicenseNamespaceOnly`. El espacio de nombres se importa. No se importa ningún documento de esquema.

7.4.2.6 ENUMSchemaMapping

Descripción
Contiene los literales de enumeración que definen cuál de las dos asignaciones de catálogo se prefiere: Esta enumeración sirve para eliminar ambigüedades en `ENUMLoadSchemalocation` y `ENUMSchemaImports`.

Utilizada por

Interfaz	*Operación*
`IXBRL`	`SchemaMapping`
`IXMLValidator`	`SchemaMapping`
`IXSLT`	`SchemaMapping`

Literales de la enumeración

```
eSMPreferSchemalocation          = 0
eSMPreferNamespace               = 1
```

eSMPreferSchemalocation
Establece que se debe seleccionar la URL de schema location.

eSMPreferNamespace
Establece que se debe seleccionar el espacio de nombres.

7.4.2.7 ENUMTableOutputFormat

Descripción
Contiene los literales de enumeración que definen el formato de salida del documento que contiene las tablas generadas.

 © 2018 Altova GmbH

Ustilizada por

Interfaz	*Operación*
IXBRL	TableOutputFormat

Literales de la enumeración

```
eFormatXML                  = 0
eFormatHTML                 = 1
```

eSMPreferSchemalocation
Establece la opción de asignación de esquemas para seleccionar la URL de la ubicación del esquema.

eSMPreferNamespace
Establece la opción de asignación de esquemas para seleccionar el espacio de nombres.

7.4.2.8 ENUMValidationType

Descripción
Contiene literales de enumeración que definen qué tipo de documento se debe validar.

Utilizada por

Interfaz	*Operación*
IXMLValidator	IsValid

Literales de la enumeración

```
eValidateAny                        = 0
eValidateXMLWithDTD                 = 1
eValidateXMLWithXSD                 = 2
eValidateDTD                        = 3
eValidateXSD                        = 4
eValidateJSON                       = 5
eValidateJSONSchema                 = 6
```

```
eValidateAvro                    = 7
eValidateAvroSchema              = 8
eValidateAvroJSON                = 9
```

eValidateAny
Establece que el tipo de validación es `Any`. Esto valida el documento después de detectar su tipo automáticamente.

eValidateXMLWithDTD
Establece que el tipo de validación es `XMLWithDTD`. Especifica que el documento XML debe validarse con un documento DTD.

eValidateXMLWithXSD
Establece que el tipo de validación es `XMLWithXSD`. Especifica que el documento XML debe validarse con un esquema XML.

eValidateDTD
Establece que el tipo de validación es `ValidateDTD`. Especifica que debe validarse un documento DTD.

eValidateXSD
Establece que el tipo de validación es `ValidateXSD`. Especifica que debe validarse un documento de XML Schema.

eValidateJSON
Establece que el tipo de validación es `ValidateJSON`. Especifica que debe validarse un documento de instancia JSON de acuerdo con las normas de JSON Schema v 4.

eValidateJSONSchema
Establece que el tipo de validación es `ValidateXSD`. Especifica que debe validarse un esquema JSON de acuerdo con las normas de JSON Schema v 4.

eValidateAvro
Establece que el tipo de validación es `ValidateAvro`. Especifica la validación de un archivo binario Avro.

eValidateAvroSchema
Establece que el tipo de validación es `ValidateAvroSchema`. Especifica la validación de un esquema Avro según las normas de la especificación Avro schema.

eValidateAvroJSON
Establece que el tipo de validación es `ValidateAvroJSON`. Especifica la validación de un documento de datos Avro en serialización JSON con un esquema Avro.

 © 2018 Altova GmbH

7.4.2.9 ENUMWellformedCheckType

Descripción
Contiene los literales de enumeración que definen el tipo de documento cuyo formato XML debe comprobarse: XML o DTD.

Utilizada por

Interfaz	*Operación*
IXMLValidator	IsWellFormed

Literales de la enumeración

```
eWellFormedAny          = 0
eWellFormedXML          = 1
eWellFormedDTD          = 2
```

eWellformedAny
Establece que el tipo de comprobación de formato es `Any`. Comprueba el formato XML del documento tras detectar automáticamente si se trata de un documento XML o DTD.

eWellformedXML
Establece que el tipo de comprobación de formato es `XML`. Comprueba si el formato del documento XML se ajusta a la especificación XML 1.0 o XML 1.1.

eWellformedDTD
Establece que el tipo de comprobación de formato es `DTD`. Comprueba si el formato del documento DTD es correcto.

eWellformedJSON
Establece que el tipo de comprobación de formato es `JSON`. Comprueba si el formato del documento JSON se ajusta a la especificación ECMA-404.

7.4.2.10 ENUMXBRLValidationType

Descripción
Contiene literales de enumeración que definen el tipo de documento XBRL que se debe validar: una instancia XBRL o una taxonomía XBRL.

Utilizada por

Interfaz	*Operación*
IXBRL	IsValid

Literales de la enumeración

```
eValidateXBRLAny                    = 0
eValidateXBRLInstance               = 1
eValidateXBRLTaxonomy               = 2
eValidateXBRLInline                 = 3
eValidateXBRLTaxonomyPackage        = 4
```

eValidateXBRLAny
Establece que el tipo de validación es `Any`. Valida el documento tras detectar automáticamente de qué tipo se trata.

eValidateXBRLInstance
Establece que el tipo de validación es `Instance`. Valida los documentos XBRL de instancia.

eValidateXBRLTaxonomy
Establece que el tipo de validación es `Taxonomy`. Valida los documentos de taxonomía XBRL.

eValidateXBRLInline
Establece que el tipo de validación es `Inline`. Especifica la validación de documentos Inline XBRL.

eValidateXBRLTaxonomyPackage
Establece que el tipo de validación es `TaxonomyPackage`. Especifica la validación de paquetes de taxonomías XBRL.

 © 2018 Altova GmbH

7.4.2.11 ENUMIXBRLVersion

Descripción
Contiene literales de enumeración que indican qué versión de la especificación Inline XBRL se debe usar: Inline XBRL 1.0 o Inline XBRL 1.1. El literal **eVersionDetect** habilita la detección automática.

Utilizada por

Interfaz	*Operación*
IXBRL	IXBRLVersion

Literales de la enumeración

```
eVersion10          = 0
eVersion11          = 1
eVersionDetect      = 2
```

eVersion10
Establece el uso de la especificación Inline XBRL 1.0.

eVersion11
Establece el uso de la especificación Inline XBRL 1.1.

eVersionDetect
Habilita la detección automática de la versión en función del documento Inline XBRL presentado.

7.4.2.12 ENUMIXBRLUriStrategy

Descripción
Contiene literales de enumeración que indican cómo se deben escribir los URI del Inline XBRL en el documento XBRL que se genera.

Utilizada por

Interfaz	*Operación*
IXBRL	IXBRLUriStrategy

Literales de la enumeración

```
eStrategyNone                = 0
eStrategyMakeAbsolute        = 1
eStrategyMakeRelative        = 2
eStrategyKeepRelative        = 3
```

`eStrategyNone`
Copia el URI literalmente en el documento de destino.

`eStrategyMakeAbsolute`
Convierte los URI relativos en absolutos resolviéndolos con el URI base dentro del ámbito del elemento correspondiente del documento de entrada.

`eStrategyMakeRelative`
Si se puede, convierte los URI absolutos y relativos en relativos al documento de salida. Si no se puede, escribe el URI absoluto resuelto.

`eStrategyKeepRelative`
Si se puede, convierte los URI relativos en relativos al documento de salida (y copia los URI absolutos).

7.4.2.13 ENUMXMLValidationMode

Descripción
Contiene literales de enumeración que definen el modo de procesamiento XML que se debe usar: validación o comprobación de formato.

Utilizada por

Interfaz	*Operación*
IXMLValidator	XMLValidationMode
IXQuery	XMLValidationMode
IXSLT	XMLValidationMode

Literales de la enumeración

 © 2018 Altova GmbH

```
eXMLValidationModeWF              = 0
eXMLValidationModeID              = 1
eXMLValidationModeValid           = 2
```

eXMLValidationModeWF
Establece que el modo de procesamiento XML es `Wellformed`. Es el valor predeterminado.

eXMLValidationModeID
Solo para uso interno.

eXMLValidationModeValid
Establece que el modo de procesamiento XML es `Validation`.

7.4.2.14 ENUMXQueryVersion

Descripción
Contiene literales de enumeración que especifican qué versión de XQuery se debe usar: XQuery 1.0 o 3.0.

Utilizada por

Interfaz	*Operación*
IXQuery	EngineVersion

Literales de la enumeración

```
eXQVersion10        = 1
eXQVersion30        = 3
```

eXQVersion10
Establece que la versión XQuery es XQuery 1.0.

eXQVersion30
Establece que la versión XQuery es XQuery 3.0. Es el valor predeterminado.

7.4.2.15 ENUMXQueryUpdatedXML

Descripción
Contiene los literales de enumeración necesarios para especificar cómo se gestionan las actualizaciones XQuery.

Utilizada por

Interfaz	*Operación*
IXQuery	UpdatedXMLWriteMode

Literales de la enumeración

```
eUpdatedDiscard          = 1
eUpdatedWriteback        = 2
eUpdatedAsMainResult     = 3
```

eUpdatedDiscard
Las actualizaciones se descartan y no se escriben en el archivo.

eUpdatedWriteback
Las actualizaciones se escriben en el archivo de entrada indicado con InputXMLFileName.

eUpdatedAsMainResult
Las actualizaciones se escriben en la ubicación indicada por el parámetro `outputFile` de ExecuteUpdate.

7.4.2.16 ENUMXSDVersion

Descripción
Contiene literales de enumeración que indican qué versión de XML Schema se debe usar para la validación: XSD 1.0 o 1.1.

Utilizada por

Interfaz	*Operación*
IXMLValidator	XSDVersion
IXQuery	XSDVersion
IXSLT	XSDVersion

 © 2018 Altova GmbH

Literales de la enumeración

```
eXSDVersionAuto          = 0
eXSDVersion10            = 1
eXSDVersion11            = 2
```

eXSDVersionAuto
Establece que la versión de XML Schema que se debe usar es `Auto-detect`. La versión XSD se detecta automáticamente tras analizar el documento XSD. Si el atributo `vc:minVersion` del documento XSD tiene el valor `1.1`, se considera que el documento tiene la versión XSD 1.1. Si este atributo tiene cualquier otro valor o si el atributo no está en el documento, se entiende que el documento tiene la versión XSD 1.0.

eXSDVersion10
Establece que la versión de XML Schema que se debe usar es XML Schema 1.0.

eXSDVersion11
Establece que la versión de XML Schema que se debe usar es XML-Schema 1.1.

7.4.2.17 ENUMXSLTVersion

Descripción
Contiene literales de enumeración que especifican qué versión XSLT se debe usar: XSLT 1.0, 2.0 o 3.0.

Utilizada por

Interfaz	*Operación*
IXSLT	EngineVersion

Literales de la enumeración

```
eVersion10        = 1
eVersion20        = 2
eVersion30        = 3
```

eVersion10

Establece que la versión XSLT que se debe usar es XSLT 1.0.

eVersion20
Establece que la versión XSLT que se debe usar es XSLT 2.0.

eVersion30
Establece que la versión XSLT que se debe usar es XSLT 3.0.

 © 2018 Altova GmbH

Altova RaptorXML+XBRL Server 2018

Información adicional

8 Información adicional

Esta sección contiene información adicional sobre:

- Códigos de salida
- Sugerencias sobre ubicación de esquemas
- Parámetros de XBRL Formula

 © 2018 Altova GmbH

8.1 Códigos de salida

Estos son los códigos de salida disponibles:

0	La validación finalizó correctamente
1	La validación terminó con errores / Proceso interrumpido por Ctrl+C/ Pausa/terminal cerrada / La licencia expiró durante la ejecución
11	RaptorXML no se puede iniciar. El motivo aparece en el archivo de registro
22	No se puede cargar el catálogo raíz / No se puede cargar el archivo de lista
64	Comandos / opciones no válidos
77	Error al adquirir licencia al inicio
128+**n**	RaptorXML finalizó debido al número de señal **n**. Todos los códigos de salida superiores a **128** indican que la finalización se debe a la recepción de una señal externa o a una señal que se desencadenó internamente. Por ejemplo, si el código de salida es **134**, el número de señal será **134-128=6** (el número de **SIGABRT**).

8.2 Sugerencias sobre ubicación de esquemas

Los documentos de instancia pueden usar sugerencias para indicar la ubicación del esquema. Para estas sugerencias se utilizan dos atributos:

- `xsi:schemaLocation` para documentos de esquema con espacios de nombres de destino. El valor de atributos es un par de elementos: el primero de ellos es un espacio de nombres y el segundo es una URL que encuentra el documento de esquema. El nombre del espacio de nombres debe coincidir con el espacio de nombres de destino del documento de esquema.

```
<document xmlns="http://www.altova.com/schemas/test03"
          xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
          xsi:schemaLocation="http://www.altova.com/schemas/test03
Test.xsd">
```

- `xsi:noNamespaceSchemaLocation` para documentos de esquema sin espacios de nombres de destino. El valor de atributos es la URL del documento de esquema. El documento de esquema al que se hace referencia no puede tener un espacio de nombres de destino.

```
<document xmlns="http://www.altova.com/schemas/test03"
          xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
          xsi:noNamespaceSchemaLocation="Test.xsd">
```

La opción `--schemalocation-hints` especifica cómo usar estos atributos en las sugerencias y, sobre todo, cómo utilizar la información del atributo `schemaLocation` (vers descripción de esta opción más arriba). Recuerde que RaptorXML+XBRL Server considera que la parte del espacio de nombres del valor `xsi:noNamespaceSchemaLocation` es una cadena vacía.

Las sugerencias sobre la ubicación de los esquemas también se puede dar en una instrucción de importación `import` de un documento XML Schema.

```
<import namespace="someNS" schemaLocation="someURL">
```

En la instrucción de importación las sugerencias también se pueden dar mediante un espacio de nombres que se pueda asignar a un esquema del archivo de catálogo o mediante una URL directamente (en el atributo `schemaLocation`). La opción `--schema-imports` (para XBRL y XSD/XML) indica cómo se debe seleccionar la ubicación del esquema.

 © 2018 Altova GmbH

8.3 Parámetros de XBRL Formula

Este apartado contiene varios temas:

- Formatos de los parámetros de fórmula, que ofrece ejemplos de los formatos XML y JSON de los parámetros de fórmulas XBRL.
- Usar parámetros de fórmulas contiene fragmentos de código Java, VB.NET, C#, VBScript y JScript que muestran cómo usar los parámetros de fórmulas con objetos de las bibliotecas Java y COM/.NET API.

8.3.1 Formatos de los parámetros de fórmula

Los parámetros de fórmula pueden estar en formato XML o en formato JSON.

Formato XML

El ejemplo de código que aparece a continuación muestra parámetros de fórmula en formato XML.

```
<?xml version="1.0" encoding="utf-8"?>
<options:formula-parameters
  xmlns:options="http://www.altova.com/schemas/altova/raptorxml/options"
  xmlns:p="http://xbrl.org/formula/conformance/paramstuff"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.altova.com/schemas/altova/raptorxml/options
http://www.altova.com/schemas/altova/raptorxml/options.xsd">

  <options:parameter name="p1">
    <options:value type="xs:string">hello world from new xml (without namespace)
</options:value>
  </options:parameter>
  <options:parameter name="p:p1" type="xs:string" value="hello world from new
xml"/>

</options:formula-parameters>
```

Observe estos aspectos del ejemplo:

- El atributo `@type` es opcional y toma el valor predeterminado `xs:string`.
- Se pueden indicar varios elementos secundarios `<options:value>` para asignar una secuencia XPath a un parámetro.
- No se pueden usar `@value` y `<options:value>` al mismo tiempo.

Formato JSON

El ejemplo de código que aparece a continuación muestra parámetros de fórmula en formato JSON.

```
{
  "formula-parameters": [
  {
    "name": "p1",
    "values": [
      {
        "type": "xs:string",
        "value": "hello world from json new (without namespace)"
      }
    ]
   }, {
        "name": "ns1:p1",
        "values": [
          {
            "type": "xs:string",
            "value": "hello world from json new"
          }
         ]
       }
   ],
   "namespaces": {
     "xs": "http://www.w3.org/2001/XMLSchema",
     "ns1": "http://xbrl.org/formula/conformance/paramstuff"
   }
}
```

Observe estos aspectos del ejemplo:

- La clave `type` es opcional y toma el valor predeterminado `xs:string`.
- La clave `xs` es opcional y toma el valor predeterminado `http://www.w3.org/2001/XMLSchema`.
- El tipo de la asignación de parámetros se usa si un valor se especifica como cadena JSON directamente.
- También se puede usar :

```
{
    "name": "p2",
    "type": "xs:string",
    "value": "hello world from json new (without namespace)"
 }, {
    "name": "p3",
    "type": "xs:int",
    "values": ["1", "2"]
 }, {
    "name": "p4",
    "type": "xs: int",
    "values": ["1", {"type": "xs:string", "value": "abc"}, "2"]
 }
```

8.3.2 Usar parámetros de fórmula

Los fragmentos de código que aparecen más abajo muestran cómo usar parámetros de fórmulas XBRL en varios lenguajes de programación. Para Java, consulte el apartado clase XBRL de la API

de Java. Para los demás lenguajes consulte el apartado interfaz XBRL de la API de COM/.NET.

Java

```
 RaptorXMLFactory rxml = RaptorXML.getFactory();
 XBRL xbrl = rxml.getXBRL();

 xbrl.addFormulaParameter( "ns1:string", "ns1:Param1", "ns1:theqname" );
 xbrl.addFormulaParameterNamespace( "ns1", "www.www.www" );

 // The parameter is an array of dates
 xbrl.addFormulaArrayParameter( "", "startDates", new Object[]{ new
FormulaParam( "xs:date", "2010-01-01" ), new FormulaParam( "xs:date",
"2012-01-01" ) } );

 // The parameter is an array of figs
 xbrl.addFormulaArrayParameter( "ns1:figs", "startFigs", new Object[]
{ "fig1", "fig2", "fig3" } );

 // The parameter is an array of figs, dates and raisins (rather wild
example)
 xbrl.addFormulaArrayParameter( "ns1:figs", "startDryFruit", new Object[]
{ "fig1", "fig2", new FormulaParam( "xs:date", "2010-01-01" ), new
FormulaParam( "ns1:raisin", "dried grape" ), "fig3" } );
```

VB.NET

```
Dim objRaptor As New Server()
   Dim objXBRL As XBRL
   objXBRL = objRaptor.GetXBRL()

   objXBRL.AddFormulaParameter("ns1:string", "ns1:Param1", "ns1:theqname")
   objXBRL.AddFormulaParameterNamespace("ns1", "www.www.www")

   'The parameter is an array of dates
   objXBRL.AddFormulaArrayParameter("", "startDates", {New XBRLFormulaParam
 With {.ParamType = "xs:date", .ParamValue = "2010-01-01"}, New
XBRLFormulaParam With {.ParamType = "xs:date", .ParamValue = "2012-01-
01"}})

   'The parameter is an array of figs
   objXBRL.AddFormulaArrayParameter("ns1:figs", "startFigs", {"fig1",
"fig2", "fig3"})

   'The parameter is an array of figs, dates and raisins (rather wild
example)
   objXBRL.AddFormulaArrayParameter("ns1:figs", "startDryFruit", {"fig1",
"fig2", New XBRLFormulaParam With {.ParamType = "xs:date", .ParamValue =
"2010-01-01"}, New XBRLFormulaParam With {.ParamType =
"ns1:raisin", .ParamValue = "dried grape"}, "fig3"})
```

C#

```
Server app = new Server();
   XBRL objXBRL = app.GetXBRL();

   objXBRL.AddFormulaParameter("ns1:string", "ns1:Param1", "ns1:theqname");
   objXBRL.AddFormulaParameterNamespace("ns1", "www.www.www");

   //The parameter is an array of dates
   objXBRL.AddFormulaArrayParameter("", "startDates", new object[] {new
XBRLFormulaParam { ParamType = "xs:date", ParamValue = "2010-01-01"}, new
XBRLFormulaParam {ParamType = "xs:date", ParamValue = "2012-01-01"}});

   //The parameter is an array of figs
   objXBRL.AddFormulaArrayParameter("ns1:figs", "startFigs", new object[]
{"fig1", "fig2", "fig3"});

   //The parameter is an array of figs, dates and raisins (rather wild
example)
   objXBRL.AddFormulaArrayParameter("ns1:figs", "startDryFruit", new
object[] { "fig1", "fig2", new XBRLFormulaParam { ParamType = "xs:date",
ParamValue = "2010-01-01" }, new XBRLFormulaParam { ParamType =
"ns1:raisin", ParamValue = "dried grape" }, "fig3" });
```

VBScript

Como la biblioteca de tipos de RaptorXML no se puede cargar con lenguajes de scripting y el tipo `XBRLFormulaParameters` no existe, el usuario de VBScript debe declarar una clase en su programa, en lugar de usar objetos `XBRL.FormulaParam`. La clase debe tener dos propiedades públicas: `ParamName` y `ParamValue` (al igual que `XBRL.FormulaParam`). La clase debería tener un constructor que tome el tipo y el valor (porque esto facilita su uso). De lo contrario, es necesario crear el objeto y definir sus miembros. Para más información consulte el apartado interfaz XBRL de la API de COM/.NET.

```
Class MyPair
   Public ParamType
   Public ParamValue
   Public Default Function Init( inType, inValue )
          ParamType = inType
          ParamValue = inValue
          set Init = Me
   End Function
End Class


Sub Main
   Dim objRaptor
   Set objRaptor = WScript.GetObject( "", "RaptorXML.Server" )
   Dim objXBRL
   Set objXBRL = objRaptor.GetXBRL
```

```
   Call objXBRL.AddFormulaParameter("ns1:string", "ns1:Param1",
"ns1:theqname")
   Call objXBRL.AddFormulaParameterNamespace("ns1", "www.www.www")

   'The parameter is an array of dates
   Call objXBRL.AddFormulaArrayParameter("", "startDates", Array( ( New
MyPair)( "xs:date", "2010-01-01"), ( New MyPair)( "xs:date", "2012-01-
01") ) )

   'The parameter is an array of figs
   Call objXBRL.AddFormulaArrayParameter("ns1:figs", "startFigs",
Array("fig1", "fig2", "fig3") )

   'The parameter is an array of figs, dates and raisins (rather wild
example)
   Call objXBRL.AddFormulaArrayParameter("ns1:figs", "startDryFruit",
Array("fig1", "fig2", ( New MyPair)("xs:date", "2010-01-01"(, ( New MyPair)
("ns1:raisin", "dried grape"), "fig3") )
End Sub

Call Main
```

JScript

Como la biblioteca de tipos de RaptorXML no se puede cargar con lenguajes de scripting y el tipo `XBRLFormulaParameters` no existe, el usuario de JScript debe declarar clases función en su programa que almacene el par valor/tipo, en lugar de usar objetos `XBRL.FormulaParam`. El nombre de los miembros debe ser `ParamType` y `ParamValue`. Para más información consulte el apartado interfaz XBRL de la API de COM/.NET.

```
function FormulaParam( inType, inValue)
{
   this.ParamType = inType;
   this.ParamValue = inValue;
}

function main()
{
   var objRaptor = new ActiveXObject( "RaptorXML.Server" );
   var objXBRL = objRaptor.GetXBRL();

   objXBRL.addFormulaParameter( "ns1:string", "ns1:Param1",
"ns1:theqname" );
   objXBRL.addFormulaParameter( "xs:string", "Param1", "bla",
"www.www.www" );

   // The parameter is an array of dates
   objXBRL.addFormulaArrayParameter("", "startDates", [new
FormulaParam("xs:date", "2010-01-01"), new FormulaParam("xs:date", "2012-
01-01")]);
```

```
   // The parameter is an array of figs
   objXBRL.addFormulaArrayParameter("ns1:figs", "startFigs", ["fig1",
"fig2", "fig3"]);

   // The parameter is an array of figs, dates and raisins (rather wild
example)
   objXBRL.addFormulaArrayParameter("ns1:figs", "startDryFruit", ["fig1",
"fig2", new FormulaParam("xs:date", "2010-01-01"), new
FormulaParam("ns1:raisin", "dried grape"), "fig3"]);
}

main()
```

 © 2018 Altova GmbH

Altova RaptorXML+XBRL Server 2018

Información sobre motores XSLT y XQuery

9 Información sobre motores XSLT y XQuery

Los motores XSLT y XQuery de RaptorXML+XBRL Server 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 y del predecesor de RaptorXML+XBRL Server, el procesador descatalogado AltovaXML. Por consiguiente, RaptorXML+XBRL Server 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
- XSLT 2.0
- XSLT 3.0
- XQuery 1.0
- XQuery 3.1

 © 2018 Altova GmbH

9.1 XSLT 1.0

El motor XSLT 1.0 de RaptorXML+XBRL Server cumple con la recomendación XSLT 1.0 del 16 de noviembre de 1999 y con la recomendación XPath 1.0 del 16 de noviembre de 1999, ambas del W3C.

Nota sobre la implementación

Cuando el atributo `method` de `xsl:output` tiene el valor HTML o si selecciona de forma predeterminada el formato de salida HTML, los caracteres especiales del archivo XML o XSLT se insertan en el documento HTML como referencias de caracteres HTML. Por ejemplo, el carácter U+00A0 (la referencia de carácter hexadecimal para un espacio de no separación) se inserta en el código HTML como referencia de carácter (` ` o ` `) o como referencia de entidad (` `).

9.2 XSLT 2.0

Temas de este apartado:

- Especificaciones con las que cumple el motor
- Compatibilidad con versiones antiguas
- Espacios de nombres
- Compatibilidad con esquemas
- Comportamiento propio de esta implementación

Especificaciones

El motor XSLT 2.0 de RaptorXML+XBRL Server cumple con la recomendación XSLT 2.0 del 23 de enero de 2007 y la recomendación XPath 2.0 del 14 de diciembre de 2010, ambas del W3C.

Compatibilidad con versiones antiguas

El motor XSLT 2.0 es compatible con versiones previas. Esto solamente es relevante cuando se utiliza el motor XSLT 2.0 (parámetro de la interfaz de la línea de comandos `--xslt=2`) para procesar una hoja de estilos XSLT 1.0. Tenga en cuenta que los resultados obtenidos con el motor XSLT 1.0 pueden ser diferentes a los obtenidos con el motor XSLT 2.0 en modo de compatibilidad con versiones antiguas.

Espacios de nombres

En su hoja de estilos XSLT 2.0 debe declarar estos espacios de nombres para poder usar los constructores de tipo y las funciones disponibles en XSLT 2.0. Los prefijos que aparecen a continuación son los que se suelen usar, pero puede usar otros prefijos si quiere.

Espacio de nombres	Prefijo	URI del espacio de nombres
Tipos XML Schema	`xs:`	`http://www.w3.org/2001/XMLSchema`
Funciones XPath 2.0	`fn:`	`http://www.w3.org/2005/xpath-functions`

Estos espacios de nombres se suelen declarar en el elemento `xsl:stylesheet` o en el elemento `xsl:transform`:

```
<xsl:stylesheet version="2.0"
    xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    xmlns:fn="http://www.w3.org/2005/xpath-functions"
    ...
</xsl:stylesheet>
```

Es necesario tener en cuenta que:

 © 2018 Altova GmbH

- El motor XSLT 2.0 utiliza el espacio de nombres Funciones XPath 2.0 y XQuery 1.0 como **espacio de nombres de funciones predeterminado**. Esto significa que puede usar funciones XPath 2.0 y XSLT 2.0 en su hoja de estilos sin prefijos. Si declara el espacio de nombres Funciones XPath 2.0 en su hoja de estilos con un prefijo, podrá usar el prefijo asignado en la declaración.
- Cuando se usan constructores de tipo y tipos del espacio de nombres XML Schema, el prefijo utilizado en la declaración de espacio de nombres se debe usar en la llamada al constructor de tipo (por ejemplo, `xs:date`).
- Algunas funciones XPath 2.0 se llaman igual que algunos tipos de datos de XML Schema. Por ejemplo, las funciones XPath `fn:string` y `fn:boolean` y los tipos de datos de XML Schema `xs:string` y `xs:boolean`. Por tanto, si usa la expresión `string('Hello')`, la expresión se evalúa como `fn:string('Hello')` y no como `xs:string('Hello')`.

Compatibilidad con esquemas

El motor XSLT 2.0 está preparado para esquemas de modo que puede usar tipos de esquema definidos por el usuario y la instrucción `xsl:validate`.

Comportamiento propio de esta implementación

Más abajo puede ver cómo se ocupa el motor XSLT 2.0 de algunos aspectos del comportamiento de las funciones XSLT 2.0 relacionadas con esta implementación.

`xsl:result-document`

También son compatibles estas codificaciones específicas de Altova: `x-base16tobinary` y `x-base64tobinary`.

`function-available`

Esta función mira si hay funciones del ámbito disponibles (funciones XSLT, XPath y de extensión).

`unparsed-text`

El atributo `href` acepta (i) rutas de acceso relativas para archivos que estén en la carpeta del URI base y (ii) rutas de acceso absolutas con o sin el protocolo `file://`. También son compatibles estas codificaciones específicas de Altova: `x-binarytobase16` y `x-binarytobase64`.

`unparsed-text-available`

El atributo `href` acepta (i) rutas de acceso relativas para archivos que estén en la carpeta del URI base y (ii) rutas de acceso absolutas con o sin el protocolo `file://`. También son compatibles estas codificaciones específicas de Altova: `x-binarytobase16` y `x-binarytobase64`.

Nota: estos valores de codificación estaban implementados en el ya descatalogado AltovaXML pero ya no se utilizan (son obsoletos): `base16tobinary`, `base64tobinary`, `binarytobase16` y `binarytobase64`.

9.3 XSLT 3.0

El motor XSLT 3.0 de RaptorXML+XBRL Server cumple con la propuesta de recomendación XSLT 3.0 del 8 de junio de 2017 y con la propuesta de recomendación XPath 3.1 del 21 de marzo de 2017 del consorcio W3C.

El motor XSLT 3.0 tiene las mismas características de implementación que el motor XSLT 2.0. Pero además ofrece compatibilidad con muchas de las nuevas funciones XSLT3.0, con las funciones y los operadores XPath/XQuery 3.1 y con la especificación XPath 3.1.

Nota: la característica opcional de transmisión por secuencias no es compatible por ahora. Todo el documento se cargará en memoria independientemente del valor del atributo `streamable` y se procesará si hay suficiente memoria. En las aplicaciones de 64 bits esto no debería causar problemas. Sin embargo, si se dan problemas de memoria, una solución sería añadir más memoria al sistema.

 © 2018 Altova GmbH

9.4 XQuery 1.0

Temas de este apartado:

- Especificaciones con las que cumple el motor
- Compatibilidad con esquemas
- Codificación
- Espacios de nombres
- Fuentes XML y validación
- Comprobación de tipos estática y dinámica
- Módulos biblioteca
- Funciones externas
- Intercalaciones
- Precisión de datos numéricos
- Compatibilidad con instrucciones XQuery

Especificaciones compatibles

El motor XQuery 1.0 de RaptorXML+XBRL Server 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 RaptorXML+XBRL Server.

Compatibilidad con esquemas

El motor XQuery 1.0 está preparado para esquemas.

Codificación

El motor XQuery 1.0 es compatible con las codificaciones de caracteres UTF-8 y UTF-16.

Espacios de nombres

Se predefinen estos URI de espacios de nombres y sus enlaces asociados.

Espacio de nombres	Prefijo	URI del espacio de nombres
Tipos XML Schema	`xs:`	`http://www.w3.org/2001/XMLSchema`
Schema instance	`xsi:`	`http://www.w3.org/2001/XMLSchema-instance`
Funciones integradas	`fn:`	`http://www.w3.org/2005/xpath-functions`
Funciones locales	`local:`	`http://www.w3.org/2005/xquery-local-functions`

Es importante tener en cuenta que:

- El motor XQuery 1.0 entiende que los prefijos de la tabla anterior están enlazados con los correspondientes espacios de nombres.
- Como el espacio de nombres de funciones integradas (ver tabla) es el espacio de nombres de funciones predeterminado de XQuery, no es necesario usar el prefijo `fn:` cuando se invocan funciones integradas (por ejemplo, `string("Hello")` llamará a la función `fn:string`). No obstante, el prefijo `fn:` se puede utilizar para llamar a una función integrada sin necesidad de declarar el espacio de nombres en el prólogo de la consulta (por ejemplo: `fn:string("Hello")`).
- Puede cambiar el espacio de nombres de funciones predeterminado declarando la expresión `default function namespace` en el prólogo de la consulta.
- Cuando use tipos del espacio de nombres XML Schema, puede usar el prefijo `xs:` sin necesidad de declarar los espacios de nombres de forma explícita ni enlazar estos prefijos a los espacios de nombres en el prólogo de la consulta. (Ejemplo: `xs:date` y `xs:yearMonthDuration`.) Si quiere usar otros prefijos para el espacio de nombres de XML Schema, estos se deben declarar en el prólogo de la consulta. (Ejemplo: `declare namespace alt = "http://www.w3.org/2001/XMLSchema"; alt:date("2004-10-04")`.)
- Recuerde que los tipos de datos `untypedAtomic`, `dayTimeDuration` y `yearMonthDuration` se movieron del espacio de nombres XPath Datatypes al espacio de nombres XML Schema (es decir, ahora es `xs:yearMonthDuration`.)

Si se asignaron mal los espacios de nombres para funciones, constructores de tipo, pruebas de nodo, etc., se emite un error. Sin embargo, recuerde que algunas funciones se llaman igual que los tipos de datos de esquema (p. ej. `fn:string` y `fn:boolean`.) (Se definen `xs:string` y `xs:boolean`.) El prefijo del espacio de nombres determina si se usa la función o el constructor de tipo.

Documento XML de origen y validación

Los documentos XML que se utilizan para ejecutar un documento XQuery con el motor XQuery 1.0 deben tener un formato XML correcto. Sin embargo, no es necesario que sean válidos con respecto a un esquema XML. Si el archivo no es válido, el archivo no válido se carga sin información de esquema. Si el archivo XML está asociado a un esquema externo y es válido con respecto a dicho esquema, se genera información posterior a la validación de esquema, que se utilizará para evaluar la consulta.

Comprobación de tipos estática y dinámica

En la fase de análisis estático se revisan aspectos de la consulta como la sintaxis, si existen referencias externas (p. ej. para módulos), si las funciones y variables que se invocan están definidas, etc. Si se detecta un error en la fase de análisis estático, se notifica y la ejecución se interrumpe.

La comprobación dinámica de tipos se realiza en tiempo de ejecución, cuando la consulta se ejecuta. Si un tipo no es compatible con los requisitos de una operación, se emite un error. Por ejemplo, la expresión `xs:string("1") + 1` devuelve un error porque la operación de suma no se puede llevar a cabo en un operando de tipo `xs:string`.

 © 2018 Altova GmbH

Módulos biblioteca

Los módulos biblioteca almacenan funciones y variables para poder volver a utilizarlas. El motor XQuery 1.0 es compatible con el uso de módulos almacenados en un **solo archivo XQuery externo**. Dicho archivo de módulo debe incluir una declaración `module` en su prólogo que apunte a un espacio de nombres de destino Por ejemplo:

```
module namespace libns="urn:module-library";
declare variable $libns:company := "Altova";
declare function libns:webaddress() { "http://www.altova.com" };
```

Todas las funciones y variables declaradas en el módulo pertenecen al espacio de nombres asociado al módulo. El módulo se importa en un archivo XQuery con la instrucción `import module` del prólogo de la consulta. La instrucción `import module` solamente importa funciones y variables declaradas directamente en el archivo de módulo biblioteca. Por ejemplo:

```
import module namespace modlib = "urn:module-library" at "modulefilename.xq";

if        ($modlib:company = "Altova")
then       modlib:webaddress()
else       error("No match found.")
```

Funciones externas

Las funciones externas son incompatibles con el motor XQuery 1.0, es decir, todas las expresiones que usen la palabra clave `external`. Por ejemplo:

```
declare function hoo($param as xs:integer) as xs:string external;
```

Intercalaciones

La intercalación predeterminada es la intercalación de puntos de código Unicode, que compara las cadenas de texto según sus puntos de código Unicode. Otras intercalaciones compatibles son las intercalaciones ICU que se enumeran aquí. Para usar una intercalación concreta, indique su URI tal y como aparece en la lista de intercalaciones compatibles. 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.

 © 2018 Altova GmbH

9.5 XQuery 3.1

El motor XQuery 3.1 de RaptorXML+XBRL Server cumple con la propuesta de recomendación XQuery 3.1 del 21 de marzo de 2017 del consorcio W3C y es compatible con funciones XPath y XQuery 3.1. La especificación XQuery 3.1 es un supraconjunto de la especificación 3.0. El motor XQuery 3.1, por tanto, es compatible con las características de XQuery 3.0.

Tiene las mismas características de implementación que el motor XQuery 1.0.

Altova RaptorXML+XBRL Server 2018

Funciones XSTL y XPath/XQuery

10 Funciones XSTL y XPath/XQuery

Esta sección enumera las funciones de extensión de Altova y otras funciones de extensión que se pueden utilizar con expresiones XPath y XQuery. Las funciones de extensión de Altova se pueden usar con los motores XSLT y XQuery de Altova y ofrecen algunas funciones más aparte de las que están disponibles en las bibliotecas de funciones definidas en los estándares del W3C.

Aspectos generales

Es necesario tener en cuenta estos puntos generales:

- A las funciones de las bibliotecas de funciones `core` definidas en las especificaciones W3C se les puede llamar sin un prefijo. Esto se debe a que los motores XSLT y XQuery leen funciones sin prefijo como si pertenecieran a un espacio de nombres de funciones predeterminado, que es el que se especifica en las especificaciones de las funciones XPath y XQuery`http://www.w3.org/2005/xpath-functions`. Si este espacio de nombres se declara explícitamente en un documento XSLT o XQuery, el prefijo utilizado en la declaración de espacio de nombres también se puede usar en el nombre de las funciones.
- Por lo general, si una función espera como argumento una secuencia de un elemento y se suministra una secuencia de más de un elemento, entonces se devuelve un error.
- Todas las comparaciones de cadena se realizan usando la intercalación de puntos de código Unicode.
- Los resultados que son QName se serializan de esta forma `[prefijo:]nombrelocal`.

Precisión de xs:decimal

El término *precisión* hace referencia al número de dígitos del número y la especificación exige un mínimo de 18 dígitos. Para las operaciones de división que dan un resultado de tipo `xs:decimal`, la precisión es de 19 dígitos después del punto decimal sin redondear.

Uso horario implícito

Cuando hay que comparar dos valores `date`, `time` o `dateTime`, es necesario conocer el uso horario de los valores que se deben comparar. Cuando el uso horario no se conoce de forma explícita, se usa el uso horario implícito, que se toma del reloj del sistema. Para probar cuál es su valor puede utilizar la función `fn:implicit-timezone()`.

Intercalaciones

La intercalación predeterminada es la intercalación de puntos de código Unicode, que compara las cadenas de texto según sus puntos de código Unicode. El motor usa el algoritmo de intercalación Unicode. Otras intercalaciones compatibles son las intercalaciones ICU que aparecen más abajo. Para usar una intercalación indique su URI tal y como aparece en la tabla más abajo. Las comparaciones de cadenas de texto (incluidas las que usan las funciones `fn:max` y `fn:min`) se harán según la intercalación especificada. Si no se especifica la opción de intercalación, se usa la intercalación predeterminada de puntos de código Unicode.

 © 2018 Altova GmbH

Idioma	Identificadores URI
`da`: danés	`da_DK`
`de`: alemán	`de_AT, de_BE, de_CH, de_DE, de_LI, de_LU`
`en`: inglés	`en_AS, en_AU, en_BB, en_BE, en_BM, en_BW, en_BZ, en_CA, en_GB, en_GU, en_HK, en_IE, en_IN, en_JM, en_MH, en_MP, en_MT, en_MU, en_NA, en_NZ, en_PH, en_PK, en_SG, en_TT, en_UM, en_US, en_VI, en_ZA, en_ZW`
`es`: español	`es_419, es_AR, es_BO, es_CL, es_CO, es_CR, es_DO, es_EC, es_ES, es_GQ, es_GT, es_HN, es_MX, es_NI, es_PA, es_PE, es_PR, es_PY, es_SV, es_US, es_UY, es_VE`
`fr`: francés	`fr_BE, fr_BF, fr_BI, fr_BJ, fr_BL, fr_CA, fr_CD, fr_CF, fr_CG, fr_CH, fr_CI, fr_CM, fr_DJ, fr_FR, fr_GA, fr_GN, fr_GP, fr_GQ, fr_KM, fr_LU, fr_MC, fr_MF, fr_MG, fr_ML, fr_MQ, fr_NE, fr_RE, fr_RW, fr_SN, fr_TD, fr_TG`
`it`: italiano	`it_CH, it_IT`
`ja`: jaonés	`ja_JP`
`nb`: noruego bokmål	`nb_NO`
`nl`: holandés	`nl_AW, nl_BE, nl_NL`
`nn`: noruego nynorsk	`nn_NO`
`pt`: portugués	`pt_AO, pt_BR, pt_GW, pt_MZ, pt_PT, pt_ST`
`ru`: ruso	`ru_MD, ru_RU, ru_UA`
`sv`: sueco	`sv_FI, sv_SE`

Eje del espacio de nombres

El eje del espacio de nombres se dejó de utilizar en XPath 2.0. Sin embargo, los ejes de espacio de nombres son compatibles con la aplicación. Para acceder a la información sobre el espacio de nombres con mecanismos de XPath 2.0, utilice las funciones `fn:in-scope-prefixes()`, `fn:namespace-uri()` y `fn:namespace-uri-for-prefix()`.

10.1 Funciones de extensión de Altova

Puede utilizar todas las funciones de extensión de Altova en sus expresiones XPath/XQuery. Con ellas conseguirá funciones adicionales no disponibles en la biblioteca de funciones estándar de XPath, XQuery y XSLT. Las funciones de extensión de Altova están en el **espacio de nombres** `http://www.altova.com/xslt-extensions` y en esta sección se presentan con el prefijo, que se supone estará enlazado al espacio de nombres señalado. Tenga en cuenta que en futuras versiones del producto algunas funciones pueden dejar de ser compatibles o su comportamiento puede cambiar. Por tanto, consulte siempre la documentación del producto para conocer el funcionamiento de estas funciones en cada versión del producto.

Las funciones definidas en las especificaciones XPath/XQuery Functions del W3C se pueden usar en (i) expresiones XPath en contextos XSLT y en (ii) expresiones XQuery en documentos XQuery. En esta documentación las funciones que se pueden usar en el primer contexto (XPath en XSLT) llevan el símbolo **XP** y se les llama funciones XPath. Las funciones que se pueden usar en contextos XQuery llevan el símbolo **XQ** y funcionan como funciones XQuery. Las especificaciones XSLT del W3C también definen funciones que se pueden usar en expresiones XPath en documentos XSLT. Estas funciones llevan el símbolo **XSLT** y se les denomina funciones XSLT. Por cada función se indica en qué versión de XPath/XQuery y XSLT se puede usar (ver símbolos más abajo). Las funciones de las bibliotecas de funciones XPath/XQuery y XSLT aparecen sin prefijo. Las funciones de extensión de otras bibliotecas, como las funciones de extensión de Altova, aparecen con un prefijo.

Funciones XPath (en expresiones XPath en XSLT):	**XP1** **XP2** **XP3.1**
Funciones XSLT (en expresiones XPath en XSLT):	**XSLT1** **XSLT2** **XSLT3**
Funciones XQuery (en expresiones XQuery en XQuery):	**XQ1** **XQ3.1**

Funciones XSLT

Las funciones XSLT solo se pueden utilizar en expresiones XPath en un contexto XSLT (igual que las funciones XSLT 2.0 `current-group()` o `key()`). Estas funciones no están pensadas para contextos no XSLT (p. ej. contextos XQuery) y, por tanto, no funcionarán en contextos que no sean XSLT. Recuerde que las funciones XSLT para XBRL solamente se pueden utilizar con ediciones de los productos de Altova compatibles con XBRL.

Funciones XPath/XQuery

Las funciones XPath/XQuery se pueden utilizar en expresiones XPath, en contextos XSLT y en expresiones XQuery:

- Funciones de fecha y hora
- Funciones de geoubicación
- Funciones relacionadas con imágenes

 © 2018 Altova GmbH

- Funciones numéricas
- Funciones de secuencia
- Funciones de cadena
- Funciones varias

Funciones para gráficos (solo en las ediciones Enterprise y Server Edition)

Las funciones de extensión para gráficos de Altova son compatibles con las ediciones Enterprise y Server Edition de los productos de Altova solamente. Estas funciones permiten generar gráficos a partir de datos XML.

Funciones para códigos de barras

Las funciones de extensión para códigos de barras de Altova permiten generar códigos de barras y colocarlos en los resultados generados con hojas de estilos XSLT.

10.1.1 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
Funciones XSLT (en expresiones XPath en XSLT):	XSLT1 XSLT2 XSLT3
Funciones XQuery (en expresiones XQuery en XQuery):	XQ1 XQ3.1

Funciones estándar

▾ distinct-nodes [altova:]

`altova:distinct-nodes(node()*)` como `node()*` XSLT1 XSLT2 XSLT3

Toma un conjunto de nodos como entrada y devuelve el mismo conjunto menos los nodos que tengan el mismo valor (es decir, devuelve los nodos que son únicos). La comparación se hace con la función XPath/XQuery `fn:deep-equal`.

Ejemplo

- **`altova:distinct-nodes`**`(country)` devuelve todos los nodos secundarios `country` excepto los que tengan el mismo valor.

evaluate [altova:]

`altova:evaluate(ExpresiónXPath` *`como xs:string`*`[, ValorDe$p1, ... ValorDe$pN])`
XSLT1 **XSLT2** **XSLT3**

Toma una expresión XPath, pasada como cadena, como argumento obligatorio. Devuelve el resultado de la expresión evaluada. Por ejemplo, **`evaluate('//Name[1]')`** devuelve el contenido del primer elemento `Name` del documento. Observe que para pasar la expresión `//Name[1]` como cadena basta con ponerla entre comillas simples.

La función `altova:evaluate` puede tomar más argumentos, que son los valores de las variables del ámbito que se llaman `p1`, `p2`, `p3`... `pN`. Recuerde que (i) las variables deben definirse con nombres de tipo `pX`, siendo `X` un entero; (ii) los argumentos de la función `altova:evaluate` (*ver firma más abajo*), a partir del segundo argumento, ofrecen los valores de las variables, correspondiendo la secuencia de argumentos a la secuencia numérica de variables: `p1` corresponde a `pN` y el segundo argumento será el valor de la variable `p1`, el tercer argumento al de la variable `p2`, y así sucesivamente; (iii) los valores de las variables deben ser de tipo `item*`.

Ejemplo

```
<xsl:variable name="xpath" select="'$p3, $p2, $p1'" />
<xsl:value-of select="altova:evaluate($xpath, 10, 20, 'hi')" />
```

da el resultado `"hi 20 10"`

En el ejemplo anterior puede observar que:

- El segundo argumento de la expresión `altova:evaluate` es el valor asignado a la variable `$p1`, el tercer argumento es el valor asignado a la variable `$p2` y así sucesivamente.
- Observe que el cuarto argumento de la función es un valor de cadena porque va entre comillas simples.
- El atributo `select` del elemento `xs:variable` suministra la expresión XPath. Como esta expresión debe ser de tipo `xs:string`, se pone entre comillas simples.

Más ejemplos

- ```
  <xsl:variable name="xpath" select="'$p1'" />
  <xsl:value-of select="altova:evaluate($xpath, //Name[1])" />
  ```
  El resultado es el valor del primer elemento `Name`.

- ```
  <xsl:variable name="xpath" select="'$p1'" />
  <xsl:value-of select="altova:evaluate($xpath, '//Name[1]')" />
  ```

 © 2018 Altova GmbH

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]

Funciones XBRL

Las funciones XBRL de Altova solo funcionan en las ediciones de los productos de Altova que son compatibles con XBRL.

- xbrl-footnotes [altova:]

  altova:xbrl-footnotes(*node()***)** como **node()*** XSLT2 XSLT3
  Toma un nodo como argumento de entrada y devuelve el conjunto de nodos de nota al pie XBRL al que hace referencia el nodo de entrada.

- xbrl-labels [altova:]

  altova:xbrl-labels(*xs:QName*, *xs:string***)** como **node()*** XSLT2 XSLT3
  Toma dos argumentos de entrada: un nombre de nodo y la ubicación del archivo de taxonomía en el que está el nodo. La función devuelve los nodos de etiqueta XBRL asociados al nodo de entrada.

[Subir]

10.1.2 Funciones XPath/XQuery: de fecha y hora

Las funciones de extensión de fecha y hora de Altova se pueden usar en expresiones XPath y XQuery y permiten procesar datos almacenados en tipos de datos XML Schema de fecha y hora. Estas funciones se pueden usar con los **motores XPath 3.0 y XQuery 3.0** de Altova y están disponibles en contextos XPath/XQuery.

Nota sobre el nombre de las funciones y lenguajes

Puede utilizar todas las funciones de extensión de Altova en sus expresiones XPath/XQuery. Con ellas conseguirá funciones adicionales no disponibles en la biblioteca de funciones estándar de XPath, XQuery y XSLT. Las funciones de extensión de Altova están en el **espacio de nombres** **http://www.altova.com/xslt-extensions** y en esta sección se presentan con el prefijo, que se supone estará enlazado al espacio de nombres señalado. Tenga en cuenta que en futuras versiones del producto algunas funciones pueden dejar de ser compatibles o su comportamiento puede cambiar. Por tanto, consulte siempre la documentación del producto para conocer el funcionamiento de estas funciones en cada versión del producto.

 © 2018 Altova GmbH

Funciones XPath (en expresiones XPath en XSLT):	XP1 XP2 XP3.1
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
- Agregar una duración a xs:date y devolver xs:date
- Agregar una duración a xs:time y devolver xs:time
- Recuperar duraciones y aplicarles formato
- Quitar la zona horaria de las funciones que generan la fecha/hora actual
- Devolver el día de la semana de una fecha como número entero
- Devolver el número de semana de una fecha como número entero
- Generar la fecha, la hora y el tipo de duración a partir de los componentes léxicos de cada tipo
- Construir un tipo date, dateTime o a partir de la cadena de entrada
- Funciones para calcular la edad

Funciones por orden alfabético

altova:add-days-to-date
altova:add-days-to-dateTime
altova:add-hours-to-dateTime
altova:add-hours-to-time
altova:add-minutes-to-dateTime
altova:add-minutes-to-time
altova:add-months-to-date
altova:add-months-to-dateTime
altova:add-seconds-to-dateTime
altova:add-seconds-to-time
altova:add-years-to-date
altova:add-years-to-dateTime
altova:age
altova:age-details
altova:build-date
altova:build-duration
altova:build-time
altova:current-dateTime-no-TZ
altova:current-date-no-TZ
altova:current-time-no-TZ
altova:format-duration
altova:parse-date
altova:parse-dateTime
altova:parse-duration
altova:parse-time
altova:weekday-from-date
altova:weekday-from-dateTime

altova:weeknumber-from-date
altova:weeknumber-from-dateTime

[Subir]

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+01:00 (por ejemplo).

add-years-to-dateTime [altova:]

altova:add-years-to-dateTime(FechaHora *as xs:dateTime*, **Años** *as xs:integer***)** como **xs:dateTime** XP3.1 XQ3.1

Añade una duración en años un valor de fecha y hora. El segundo argumento es el número de años que se debe añadir al valor de fecha y hora dado como primer argumento. El resultado es de tipo xs:dateTime.

Ejemplos

- **altova:add-years-to-dateTime**(xs:dateTime("2014-01-15T14:00:00"), 10) devuelve 2024-01-15T14:00:00
- **altova:add-years-to-dateTime**(xs:dateTime("2014-01-15T14:00:00"), -4) devuelve 2010-01-15T14:00:00

add-months-to-dateTime [altova:]

altova:add-months-to-dateTime(FechaHora *as xs:dateTime*, **Meses** *as xs:integer***)** como **xs:dateTime** XP3.1 XQ3.1

Añade una duración en meses a un valor de fecha y hora. El segundo argumento es el número de meses que se debe añadir al valor de fecha y hora dado como primer argumento. El resultado es de tipo xs:dateTime.

Ejemplos

- **altova:add-months-to-dateTime**(xs:dateTime("2014-01-15T14:00:00"), 10) devuelve 2014-11-15T14:00:00
- **altova:add-months-to-dateTime**(xs:dateTime("2014-01-15T14:00:00"), -2) devuelve 2013-11-15T14:00:00

add-days-to-dateTime [altova:]

altova:add-days-to-dateTime(FechaHora *as xs:dateTime*, **Días** *as xs:integer***)** como **xs:dateTime** XP3.1 XQ3.1

Añade una duración en días a un valor de fecha y hora. El segundo argumento es el número de días que se deben añadir al valor de fecha y hora dado como primer argumento. El resultado es de tipo xs:dateTime.

 © 2018 Altova GmbH

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]

Recuperar duraciones y aplicarles formato XP3.1 XQ3.1

Estas funciones añaden una duración a **xs:date** y devuelven **xs:date**. El tipo `xs:date` tiene el formato `SSAA-MM-DD`.

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]

Agregar una duración a xs:date XP3.1 XQ3.1

Estas funciones agregan una duración a **xs:date** y devuelven **xs:date**. El tipo `xs:date` tiene el formato `SSAA-MM-DD`.

add-years-to-date [altova:]

altova:add-years-to-date(Fecha *as xs:date*, **Años** *as xs:integer***)** como **xs:date** XP3.1 XQ3.1

Añade una duración en años a una fecha. El segundo parámetro es el número de años que se debe añadir a la fecha dada como primer argumento. El resultado es de tipo `xs:date`.

Ejemplos

- **altova:add-years-to-date**(xs:date("2014-01-15"), 10) devuelve 2024-01-15
- **altova:add-years-to-date**(xs:date("2014-01-15"), -4) devuelve 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]

Agregar una duración a `xs:time` XP3.1 XQ3.1

Estas funciones agregan una duración a **xs:time** y devuelven **xs:time**. El tipo `xs:time` tiene un formato léxico de este tipo `hh:mm:ss.sss`. Si quiere, puede añadir un sufijo de zona horaria. La letra `Z` indica (UTC). Las demás zonas horarias se representan con la diferencia que hay entre ellas y la zona UTC: `+hh:mm` o `-hh:mm`. Si falta el valor de zona horaria, se entiende que se desconoce (no se da por hecho que es UTC)

add-hours-to-time [altova:]

altova:add-hours-to-time(Hora *as xs:time*, **Horas** *as xs:integer*) como **xs:time**
XP3.1 **XQ3.1**
Añade una duración en horas a una hora. El segundo argumento es el número de horas que se debe añadir a la hora dada como primer argumento. El resultado es de tipo `xs:time`.

Ejemplos

- **altova:add-hours-to-time**(xs:time("11:00:00"), 10) devuelve 21:00:00
- **altova:add-hours-to-time**(xs:time("11:00:00"), -7) devuelve 04:00:00

add-minutes-to-time [altova:]

altova:add-minutes-to-time(Hora *as xs:time*, **Minutos** *as xs:integer*) como **xs:time** **XP3.1** **XQ3.1**

Añade una duración en minutos a una hora. El segundo argumento es el número de minutos que se debe añadir a la hora dada como primer argumento. El resultado es de tipo xs:time.

Ejemplos

- **altova:add-minutes-to-time**(xs:time("14:10:00"), 45) devuelve 14:55:00
- **altova:add-minutes-to-time**(xs:time("14:10:00"), -5) devuelve 14:05:00

add-seconds-to-time [altova:]

altova:add-seconds-to-time(Hora *as xs:time*, **Segundos** *as xs:integer*) como **xs:time** **XP3.1** **XQ3.1**

Añade una duración en segundos a una hora. El segundo argumento es el número de segundos que se debe añadir a la hora dada como primer argumento. El resultado es de tipo xs:time. El componente **Segundos** puede estar comprendido entre 0 y 59.999.

Ejemplos

- **altova:add-seconds-to-time**(xs:time("14:00:00"), 20) devuelve 14:00:20
- **altova:add-seconds-to-time**(xs:time("14:00:00"), 20.895) devuelve 14:00:20.895

[Subir]

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-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-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-time-no-TZ [altova:]

altova:current-time-no-TZ() as **xs: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

[Subir]

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(FechaHora *as xs:dateTime***)** como **xs:integer** XP3.1 XQ3.1
Toma una fecha con hora 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 *as xs:dateTime*, **Formato** *as xs:integer*) como **xs:integer** **XP3.1** **XQ3.1**

Toma una fecha con hora como primer 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 `Lunes=1`. 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-dateTime(xs:dateTime("2014-02-03T09:00:00"), 1)` devuelve `1`, lo cual significa "Lunes".
- `altova:weekday-from-dateTime(xs:dateTime("2014-02-03T09:00:00"), 4)` devuelve `1`, lo cual significa "Lunes".
- `altova:weekday-from-dateTime(xs:dateTime("2014-02-03T09:00:00"), 0)` devuelve `2`, lo cual significa "Lunes".

weekday-from-date [altova:]

altova:weekday-from-date(Fecha *as 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(Fecha *as xs:date*, **Formato** *as 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. Los días de la semana se numeran del 1 al 7 empezando por `Lunes=1`. 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]

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]

Generar tipos de datos de fecha, hora y duración a partir de sus componentes léxicos XP3.1 XQ3.1

Estas funciones toman los componentes léxicos de los tipos de datos xs:date, xs:time y xs:duration como argumentos de entrada y los combinan para generar el tipo de datos correspondiente.

build-date [altova:]

altova:build-date(Año *as xs:integer*, **Mes** *as xs:integer*, **Fecha** *as xs:integer***)** como **xs:date** XP3.1 XQ3.1

Los argumentos son el año, el mes y la fecha respectivamente. Se combinan para generar un valor de tipo xs:date. Los valores de los enteros deben estar en el intervalo de esa fecha en particular. Por ejemplo, el segundo argumento (para el mes) no puede ser mayor que 12.

Ejemplos

- **altova:build-date**(2014, 2, 03) devuelve 2014-02-03

build-time [altova:]

altova:build-time(Horas *as xs:integer*, **Minutos** *as xs:integer*, **Segundos** *as xs:integer***)** como **xs:time** XP3.1 XQ3.1

El primer, segundo y tercer argumentos son la hora (0 - 23), los minutos (0 - 59) y los segundos (0 - 59) respectivamente. Se combinan para generar un valor de tipo xs:time. Los valores de los enteros deben estar dentro del intervalo correcto de esa parte de tiempo concreta. Por ejemplo, el segundo argumento (Minutos) no puede ser mayor que 59. Para añadir la parte de uso horario al valor, use la firma que aparece más abajo.

Ejemplos

- **altova:build-time**(23, 4, 57) devuelve 23:04:57

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]

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]

Funciones para calcular la edad XP3.1 XQ3.1

Estas funciones devuelven la edad que se calcula obteniendo la diferencia (i) entre la fecha del argumento de entrada y la fecha actual o (ii) entre las fechas de los dos argumentos de entrada. La función **age** devuelve la edad en años, mientras que la función **age-details** devuelve la edad en forma de una secuencia de tres enteros (años, meses y días).

age [altova:]

altova:age(FechaInicio *as xs:date*) como **xs:integer** XP3.1 XQ3.1

Devuelve un entero que es la edad *en años* de algún objeto, contando a partir de la fecha de inicio dada como argumento y hasta la fecha actual (tomada del reloj del sistema). Si el argumento de entrada es un año o más después que la fecha actual, el valor devuelto será negativo.

Ejemplos

Si la fecha actual es **2014-01-15**:

- **altova:age**(xs:date("2013-01-15")) devuelve 1
- **altova:age**(xs:date("2013-01-16")) devuelve 0

- `altova:age(xs:date("2015-01-15"))` devuelve `-1`
- `altova:age(xs:date("2015-01-14"))` devuelve `0`

altova:age(FechaInicio *as xs:date*, **FechaFinal** *as xs:date*) como **xs:integer** **XP3.1** **XQ3.1**
Devuelve un entero que es la edad *en años* de algún objeto, contando a partir de la fecha de inicio dada como primer argumento y hasta la fecha dada como segundo argumento. El valor devuelto será negativo si el primer argumento es un año o más después que el segundo argumento.

Ejemplos

- `altova:age(xs:date("2000-01-15"), xs:date("2010-01-15"))` devuelve `10`
- `altova:age(xs:date("2000-01-15"), current-date())` devuelve `14` si la fecha actual es `2014-01-15`
- `altova:age(xs:date("2014-01-15"), xs:date("2010-01-15"))` devuelve `-4`

age-details [altova:]

altova:age-details(FechaEntrada *as xs:date*) como (**xs:integer**)* **XP3.1** **XQ3.1**
Devuelve tres enteros que son los años, meses y días respectivamente que hay entre la fecha dada como argumento y la fecha actual (tomada del reloj del sistema). La suma del valor devuelto nos da el tiempo total transcurrido entre ambas fechas (entre la fecha dada y la fecha actual). La fecha de entrada puede tener un valor anterior o posterior a la fecha actual, pero esto no se indica en el valor devuelto por medio de un signo negativo o positivo. El valor devuelto siempre es positivo.

Ejemplos

Si la fecha actual es **2014-01-15**:

- `altova:age-details(xs:date("2014-01-16"))` devuelve `(0 0 1)`
- `altova:age-details(xs:date("2014-01-14"))` devuelve `(0 0 1)`
- `altova:age-details(xs:date("2013-01-16"))` devuelve `(1 0 1)`
- `altova:age-details(current-date())` devuelve `(0 0 0`

altova:age-details(Fecha1 *as xs:date*, **Fecha2** *as xs:date*) como (**xs:integer**)* **XP3.1** **XQ3.1**
Devuelve tres enteros que son los años, meses y días que hay entre las dos fechas dadas por los argumentos. La suma del valor devuelto nos da el tiempo total transcurrido entre las dos fechas de entrada. Da igual cuál de las dos fechas se da como primer argumento, la más antigua o la más reciente. El valor devuelto no indica si la fecha de entrada es anterior o posterior a la fecha actual. Es decir, el valor devuelto siempre es positivo.

Ejemplos

- `altova:age-details(xs:date("2014-01-16"), xs:date("2014-01-15"))` devuelve `(0 0 1)`
- `altova:age-details(xs:date("2014-01-15"), xs:date("2014-01-16"))` devuelve `(0 0 1)`

[Subir]

10.1.3 Funciones XPath/XQuery: de geoubicación

Las funciones de extensión XPath/XQuery de geoubicación son compatibles con la versión actual de RaptorXML+XBRL Server 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
Funciones XSLT (en expresiones XPath en XSLT):	XSLT1 XSLT2 XSLT3
Funciones XQuery (en expresiones XQuery en XQuery):	XQ1 XQ3.1

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` y el atributo `@Geolocation` de los metadatos Exif se pueden utilizar para suministrar la cadena de entrada de la geoubicación (*ver ejemplos*).

Ejemplos

- **altova:parse-geolocation**("33.33 -22.22") devuelve la secuencia de dos `xs:decimals (33.33, 22.22)`
- **altova:parse-geolocation**("48°51'29.6""N 24°17'40.2""") devuelve la secuencia de dos `xs:decimals (48.8582222222222, 24.2945)`
- **altova:parse-geolocation**('48°51''29.6"N 24°17''40.2"') devuelve la secuencia de dos `xs:decimals (48.8582222222222, 24.2945)`
- **altova:parse-geolocation**(**image-exif-data**(//MisImágenes/Imagen20141130.01)**/@Geolocation**) devuelve una secuencia de dos

`xs:decimals`

Formato de las cadenas de entrada de geoubicaciones:

La cadena de entrada de la geoubicación debe contener la latitud y la longitud (en ese orden) se paradas por un espacio en blanco. Ambas pueden estar en cualquier formato de los que se indican más abajo y puede combinar formatos distintos. Es decir, la latitud puede estar en un formato y la longitud en otro. Los valores de la latitud deben estar comprendidos entre `+90` y `-90` (`N` a `S`). Los valores de longitud deben estar comprendidos entre `+180` y `-180` (`E` a `W`).

Nota: si utiliza comillas simples o dobles para delimitar el argumento de la cadena de entrada, esto dará lugar a un conflicto con las comillas simples o dobles que se utilizan, respectivamente, para indicar los valores de los minutos y los segundos. Si esto ocurre, debe añadir caracteres de escape a las comillas utilizadas para los minutos y segundos (esto se hace duplicando las comillas.

- Grados, minutos y segundos decimales + orientación como sufijo (`N/S`, `W/E`)
  `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/W`**) 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`, `W/E`)
  `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/W`**) es opcional
  `+/-D°M.MM'  +/-D°M.MM'`
  Ejemplo: **`+33°55.55'  -22°44.44'`**

- Grados decimales + orientación como sufijo (`N/S`, `W/E`)
  `D.DDN/S  D.DDW/E`
  Ejemplo: **`33.33N  22.22W`**

- Grados decimales + prefijo (`+/-`). El signo + para (**`N/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` y el atributo de metadatos Exif `@Geolocation` pueden utilizarse para suministrar las cadenas de entrada de geoubicaciones.

Ejemplos

- **altova:geolocation-distance-km**("33.33 -22.22", "48°51'29.6""N 24° 17'40.2""") devuelve el `xs:decimal` `4183.08132372392`

Formato de las cadenas de entrada de geoubicaciones:

La cadena de entrada de la geoubicación debe contener la latitud y la longitud (en ese orden) se paradas por un espacio en blanco. Ambas pueden estar en cualquier formato de los que se indican más abajo y puede combinar formatos distintos. Es decir, la latitud puede estar en un formato y la longitud en otro. Los valores de la latitud deben estar comprendidos entre `+90` y `-90` (`N` a `S`). Los valores de longitud deben estar comprendidos entre `+180` y `-180` (`E` a `W`).

Nota: si utiliza comillas simples o dobles para delimitar el argumento de la cadena de entrada, esto dará lugar a un conflicto con las comillas simples o dobles que se utilizan, respectivamente, para indicar los valores de los minutos y los segundos. Si esto ocurre, debe añadir caracteres de escape a las comillas utilizadas para los minutos y segundos (esto se hace duplicando las comillas.

- Grados, minutos y segundos decimales + orientación como sufijo (`N/S`, `W/E`)
  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/W**) 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`, `W/E`)
  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/W**) es opcional
  +/-D°M.MM' +/-D°M.MM'
  Ejemplo: **+33°55.55' -22°44.44'**

- Grados decimales + orientación como sufijo (N/S, W/E)
  D.DDN/S D.DDW/E
  Ejemplo: **33.33N 22.22W**

- Grados decimales + prefijo (+/-). El signo + para (**N/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 y el atributo de metadatos Exif @Geolocation 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.

- Grados, minutos y segundos decimales + orientación como sufijo (`N/S`, `W/E`)
  `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/W`**) 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`, `W/E`)
  `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/W`**) es opcional
  `+/-D°M.MM'  +/-D°M.MM'`
  Ejemplo: **`+33°55.55'  -22°44.44'`**

- Grados decimales + orientación como sufijo (`N/S`, `W/E`)
  `D.DDN/S  D.DDW/E`
  Ejemplo: **`33.33N  22.22W`**

- Grados decimales + prefijo (`+/-`). El signo + para (**`N/W`**) es opcional
  `+/-D.DD  +/-D.DD`
  Ejemplo: **`33.33  -22.22`**

Ejemplos de combinación de formatos:

```
33.33N  -22°44'55.25"
33.33  22°44'55.25"W
33.33  22.45
```

Atributo Exif de Altova: Geolocation

El motor XPath/XQuery de Altova genera el atributo personalizado **`Geolocation`** a partir de las etiquetas de metadatos Exif estándar. Este atributo es una concatenación de cuatro etiquetas Exif (`GPSLatitude`, `GPSLatitudeRef`, `GPSLongitude`, `GPSLongitudeRef`) seguidas de unidades:

GPSLatitude	GPSLatitudeRef	GPSLongitude	GPSLongitudeRef	Geolocation

33 51 21.91	S	151 13 11.73	E	33°51'21.91"S 151°13'11.73"E

geolocation-within-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 y el atributo de metadatos Exif @Geolocation pueden utilizarse para suministrar las cadenas de entrada de geoubicaciones.

Ejemplos

- **altova:geolocation-within-polygon("33 -22", ("58 -32", "-78 -55", "48 24", "58 -32"))** devuelve true()
- **altova:geolocation-within-polygon("33 -22", ("58 -32", "-78 -55", "48 24"))** devuelve true()
- **altova:geolocation-within-polygon("33 -22", ("58 -32", "-78 -55", "48°51'29.6""N 24°17'40.2""")))** devuelve true()

Formato de las cadenas de entrada de geoubicaciones:

La cadena de entrada de la geoubicación debe contener la latitud y la longitud (en ese orden) se paradas por un espacio en blanco. Ambas pueden estar en cualquier formato de los que se indican más abajo y puede combinar formatos distintos. Es decir, la latitud puede estar en un formato y la longitud en otro. Los valores de la latitud deben estar comprendidos entre +90 y -90 (N a S). Los valores de longitud deben estar comprendidos entre +180 y -180 (E a W).

Nota: si utiliza comillas simples o dobles para delimitar el argumento de la cadena de entrada, esto dará lugar a un conflicto con las comillas simples o dobles que se utilizan, respectivamente, para indicar los valores de los minutos y los segundos. Si esto ocurre, debe añadir caracteres de escape a las comillas utilizadas para los minutos y segundos (esto se hace duplicando las comillas.

- Grados, minutos y segundos decimales + orientación como sufijo (N/S, W/E)
  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/W**)

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, W/E)
  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/W**) es opcional
  +/-D°M.MM' +/-D°M.MM'
  Ejemplo: **+33°55.55' -22°44.44'**

- Grados decimales + orientación como sufijo (N/S, W/E)
  D.DDN/S D.DDW/E
  Ejemplo: **33.33N 22.22W**

- Grados decimales + prefijo (+/-). El signo + para (**N/W**) es opcional
  +/-D.DD +/-D.DD
  Ejemplo: **33.33 -22.22**

Ejemplos de combinación de formatos:
33.33N -22°44'55.25"
33.33 22°44'55.25"W
33.33 22.45

Atributo Exif de Altova: Geolocation

El motor XPath/XQuery de Altova genera el atributo personalizado **Geolocation** a partir de las etiquetas de metadatos Exif estándar. Este atributo es una concatenación de cuatro etiquetas Exif (GPSLatitude, GPSLatitudeRef, GPSLongitude, GPSLongitudeRef) seguidas de unidades:

GPSLatitude	GPSLatitudeRef	GPSLongitude	GPSLongitudeRef	Geolocation
33 51 21.91	S	151 13 11.73	E	33°51'21.91"S 151°13'11.73"E

geolocation-within-rectangle [altova:]

altova:geolocation-within-rectangle(Geoubicación *como xs:string*, **ÁnguloRectángulo-1** *como xs:string*, **ÁnguloRectángulo-2** *como xs:string*) como **xs:boolean** **XP3.1** **XQ3.1**

Determina si **Geoubicación** (primer argumento) está dentro del rectángulo definido por el segundo y el tercer argumento (**ÁnguloRectángulo-1** y **ÁnguloRectángulo-2**), que indican ángulos opuestos del rectángulo. Todos los argumentos de la función se dan como cadenas de entrada de geoubicación (*formatos permitidos más abajo*). Si el argumento Geoubicación está dentro del rectángulo, entonces la función devuelve true(). De lo contrario, devuelve false(). Los valores de latitud están comprendidos entre +90 y -90 (N a S). Los valores de longitud están comprendidos entre +180 y -180 (E a W).

Nota: la función `image-exif-data` y el atributo de metadatos Exif `@Geolocation` pueden utilizarse para suministrar las cadenas de entrada de geoubicaciones.

Ejemplos

- **altova:geolocation-within-rectangle**("33 -22", "58 -32", "-48 24") devuelve `true()`
- **altova:geolocation-within-rectangle**("33 -22", "58 -32", "48 24") devuelve `false()`
- **altova:geolocation-within-rectangle**("33 -22", "58 -32", "48°51'29.6""S 24°17'40.2""") devuelve `true()`

Formato de las cadenas de entrada de geoubicaciones:

La cadena de entrada de la geoubicación debe contener la latitud y la longitud (en ese orden) se paradas por un espacio en blanco. Ambas pueden estar en cualquier formato de los que se indican más abajo y puede combinar formatos distintos. Es decir, la latitud puede estar en un formato y la longitud en otro. Los valores de la latitud deben estar comprendidos entre `+90` y `-90` (`N` a `S`). Los valores de longitud deben estar comprendidos entre `+180` y -`180` (`E` a `W`).

Nota: si utiliza comillas simples o dobles para delimitar el argumento de la cadena de entrada, esto dará lugar a un conflicto con las comillas simples o dobles que se utilizan, respectivamente, para indicar los valores de los minutos y los segundos. Si esto ocurre, debe añadir caracteres de escape a las comillas utilizadas para los minutos y segundos (esto se hace duplicando las comillas.

- Grados, minutos y segundos decimales + orientación como sufijo (`N/S`, `W/E`)
  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/W**) 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`, `W/E`)
  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/W**) es opcional
  +/-D°M.MM' +/-D°M.MM'
  Ejemplo: **+33°55.55' -22°44.44'**

- Grados decimales + orientación como sufijo (`N/S`, `W/E`)
  D.DDN/S D.DDW/E
  Ejemplo: **33.33N 22.22W**

- Grados decimales + prefijo (`+/-`). El signo + para (**N/W**) es opcional
  +/-D.DD +/-D.DD
  Ejemplo: **33.33 -22.22**

 © 2018 Altova GmbH

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]

10.1.4 Funciones XPath/XQuery: relacionadas con imágenes

Las funciones de extensión XPath/XQuery para trabajar con imágenes son compatibles con la versión actual de RaptorXML+XBRL Server 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
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:

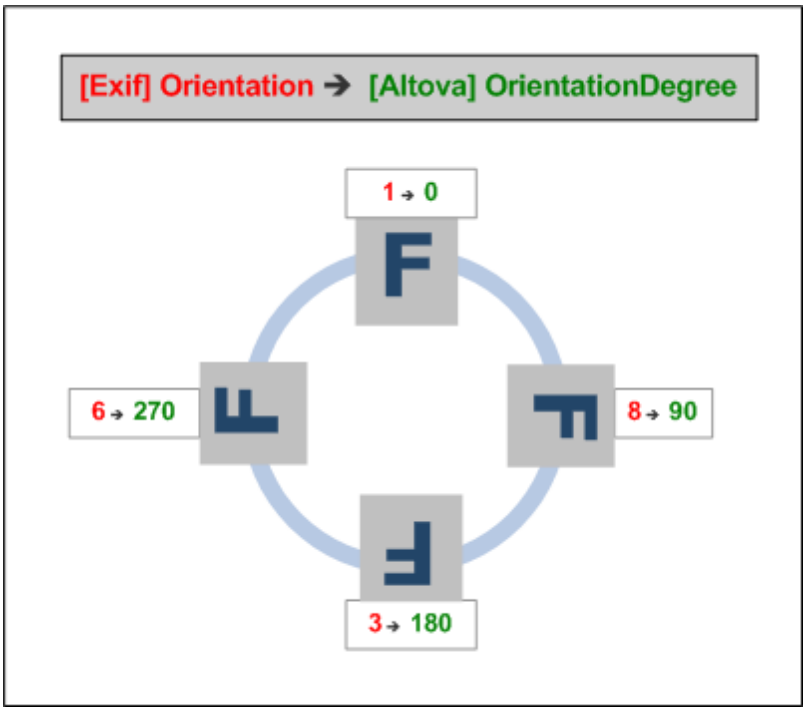
GPSLatitude	GPSLatitudeRef	GPSLongitude	GPSLongitudeRef	Geolocation
33 51 21.91	S	151 13 11.73	E	33°51'21.91"S 151°13'11.73"E

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
- 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

 © 2018 Altova GmbH

- 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
- GPSDestLongitude
- GPSDestBearingRef
- GPSDestBearing
- GPSDestDistanceRef
- GPSDestDistance
- GPSProcessingMethod
- GPSAreaInformation
- GPSDateStamp
- GPSDifferential

[Subir]

10.1.5 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
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]

Funciones numéricas

hex-string-to-integer [altova:]

`altova:hex-string-to-integer(CadenaHex` *`as xs:string`*`)` como **`xs:integer`** **XP3.1 XQ3.1**

Toma un argumento de cadena que es el equivalente Base-16 de un entero del sistema decimal (Base-10) y devuelve un entero decimal.

Ejemplos

- **`altova:hex-string-to-integer`**`('1')` devuelve `1`
- **`altova:hex-string-to-integer`**`('9')` devuelve `9`
- **`altova:hex-string-to-integer`**`('A')` devuelve `10`
- **`altova:hex-string-to-integer`**`('B')` devuelve `11`

- `altova:hex-string-to-integer('F')` devuelve `15`
- `altova:hex-string-to-integer('G')` devuelve un error
- `altova:hex-string-to-integer('10')` devuelve `16`
- `altova:hex-string-to-integer('01')` devuelve `1`
- `altova:hex-string-to-integer('20')` devuelve `32`
- `altova:hex-string-to-integer('21')` devuelve `33`
- `altova:hex-string-to-integer('5A')` devuelve `90`
- `altova:hex-string-to-integer('USA')` devuelve un error

integer-to-hex-string [altova:]

altova:integer-to-hex-string(Entero *as xs:integer*) como **xs:string** **XP3.1** **XQ3.1**
Toma el argumento **Entero** y devuelve su equivalente Base-16 en forma de cadena.

Ejemplos

- `altova:integer-to-hex-string(1)` devuelve `'1'`
- `altova:integer-to-hex-string(9)` devuelve `'9'`
- `altova:integer-to-hex-string(10)` devuelve `'A'`
- `altova:integer-to-hex-string(11)` devuelve `'B'`
- `altova:integer-to-hex-string(15)` devuelve `'F'`
- `altova:integer-to-hex-string(16)` devuelve `'10'`
- `altova:integer-to-hex-string(32)` devuelve `'20'`
- `altova:integer-to-hex-string(33)` devuelve `'21'`
- `altova:integer-to-hex-string(90)` devuelve `'5A'`

[Subir]

Funciones de formato numérico

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").

[Subir]

10.1.6 Funciones XPath/XQuery: de secuencia

Las funciones de extensión de Altova para trabajar con secuencias pueden utilizarse en expresiones XPath y XQuery y ofrecen funciones adicionales para el procesamiento de datos. Estas funciones se pueden usar con los motores **XPath 3.0** y **XQuery 3.0** de Altova. Están disponibles en contextos XPath/XQuery.

Nota sobre el nombre de las funciones y lenguajes

Puede utilizar todas las funciones de extensión de Altova en sus expresiones XPath/XQuery. Con ellas conseguirá funciones adicionales no disponibles en la biblioteca de funciones estándar de XPath, XQuery y XSLT. Las funciones de extensión de Altova están en el **espacio de nombres** **http://www.altova.com/xslt-extensions** y en esta sección se presentan con el prefijo, que se supone estará enlazado al espacio de nombres señalado. Tenga en cuenta que en futuras versiones del producto algunas funciones pueden dejar de ser compatibles o su comportamiento puede cambiar. Por tanto, consulte siempre la documentación del producto para conocer el funcionamiento de estas funciones en cada versión del producto.

Funciones XPath (en expresiones XPath en XSLT):	XP1 XP2 XP3.1
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 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 *como item()**)*, **(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)
```

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.

- 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)

10.1.7 Funciones XPath/XQuery: de cadena

Las funciones de extensión de Altova para trabajar con cadenas pueden utilizarse en expresiones XPath y XQuery y ofrecen funciones adicionales para el procesamiento de datos. Estas funciones se pueden usar con los motores **XPath 3.0** y **XQuery 3.0** de Altova. Están disponibles en contextos XPath/XQuery.

Nota sobre el nombre de las funciones y lenguajes

Puede utilizar todas las funciones de extensión de Altova en sus expresiones XPath/XQuery. Con ellas conseguirá funciones adicionales no disponibles en la biblioteca de funciones estándar de XPath, XQuery y XSLT. Las funciones de extensión de Altova están en el **espacio de nombres** **http://www.altova.com/xslt-extensions** y en esta sección se presentan con el prefijo, que se supone estará enlazado al espacio de nombres señalado. Tenga en cuenta que en futuras versiones del producto algunas funciones pueden dejar de ser compatibles o su comportamiento puede cambiar. Por tanto, consulte siempre la documentación del producto para conocer el funcionamiento de estas funciones en cada versión del producto.

Funciones XPath (en expresiones XPath en XSLT):	**XP1** **XP2** **XP3.1**
Funciones XSLT (en expresiones XPath en XSLT):	**XSLT1** **XSLT2** **XSLT3**
Funciones XQuery (en expresiones XQuery en XQuery):	**XQ1** **XQ3.1**

- camel-case [altova:]

  altova:camel-case(CadenaEntrada *como xs:string***)** como **xs:string** **XP3.1** **XQ3.1**

  Devuelve la cadena de entrada **CadenaEntrada** escrita en CamelCase. La cadena se analiza

usando la expresión regular `'\s'` (que es la forma abreviada del carácter espacio en blanco). El primer carácter que no sea un espacio en blanco situado después de un espacio en blanco o de una secuencia de espacios en blanco consecutivos se pondrá en mayúsculas. El primer carácter de la cadena de salida se pondrá en mayúsculas.

Ejemplos

- `altova:camel-case("max")` devuelve `Max`
- `altova:camel-case("max max")` devuelve `Max Max`
- `altova:camel-case("file01.xml")` devuelve `File01.xml`
- `altova:camel-case("file01.xml file02.xml")` devuelve `File01.xml File02.xml`
- `altova:camel-case("file01.xml  file02.xml")` devuelve `File01.xml File02.xml`
- `altova:camel-case("file01.xml -file02.xml")` devuelve `File01.xml -file02.xml`

altova:camel-case(CadenaEntrada *como xs:string*, **CaracteresDivisión** *como xs:string*, **EsExpReg** *como xs:boolean*) como **xs:string** **XP3.1** **XQ3.1**

Devuelve la cadena de entrada **CadenaEntrada** escrita en CamelCase usando los **CaracteresDivisón** para determinar qué caracteres desencadenan el siguiente uso de mayúsculas. El argumento **CaracteresDivisón** se usa como expresión regular **cuando EsExpReg = true()** o como caracteres planos cuando r **EsExpReg = false()**. El primer carácter de la cadensa de salida se escribe con mayúsculas.

Ejemplos

- `altova:camel-case("setname getname", "set|get", true())` devuelve `setName getName`
- `altova:camel-case("altova\documents\testcases", "\", false())` devuelve `Altova\Documents\Testcases`

char [altova:]

altova:char(Posición *as xs:integer*) como **xs:string** **XP3.1** **XQ3.1**

Devuelve una cadena que contiene el carácter que está en la posición indicada por el argumento **Posición** en la cadena que se obtiene al convertir el valor del elemento de contexto en `xs:string`. La cadena resultante estará vacía si en la posición indicada no existe ningún carácter.

Ejemplos

Si el elemento de contexto es **1234ABCD**:

- `altova:char(2)` devuelve `2`
- `altova:char(5)` devuelve `A`
- `altova:char(9)` devuelve la cadena vacía
- `altova:char(-2)` devuelve la cadena vacía

altova:char(CadenaEntrada *as xs:string*, **Posición** *as xs:integer*) como **xs:string** **XP3.1** **XQ3.1**

Devuelve una cadena que contiene el carácter que está en la posición indicada por el argumento **Posición** en la cadena dada por el argumento **CadenaEntrada**. La cadena resultante estará vacía si en la posición indicada no existe ningún carácter.

Ejemplos

- `altova:char("2014-01-15", 5)` devuelve `-`
- `altova:char("USA", 1)` devuelve `U`
- `altova:char("USA", 1)` devuelve la cadena vacía
- `altova:char("USA", -2)` devuelve la cadena vacía

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`

last-chars [altova:]

`altova:last-chars(X as xs:integer)` como `xs:string` **XP3.1** **XQ3.1**

Devuelve una cadena que contiene los X últimos caracteres de la cadena que se obtiene al convertir el valor del elemento de contexto en `xs:string`.

Ejemplos

Si el elemento de contexto es **1234ABCD**:

- `altova:last-chars(2)` devuelve `CD`
- `altova:last-chars(5)` devuelve `4ABCD`
- `altova:last-chars(9)` devuelve `1234ABCD`

`altova:last-chars(CadenaEntrada as xs:string, X as xs:integer)` como `xs:string` **XP3.1** **XQ3.1**

Devuelve una cadena que contiene los **x** últimos caracteres de la cadena dada como argumento **CadenaEntrada**.

Ejemplos

- **altova:last-chars**("2014-01-15", 5) devuelve 01-15-
- **altova:last-chars**("USA", 10) devuelve USA

pad-string-left [altova:]

altova:pad-string-left(CadenaParaRellenar *como xs:string*, **LongitudCadena** *como xs:integer*, **CarácterRelleno** *como xs:string*) como **xs:string** **XP3.1** **XQ3.1**

El argumento CarácterRelleno es un solo carácter. Se añade a la izquierda de la cadena para aumentar el número de caracteres de la CadenaParaRellenar, de modo que este número equivalga al valor entero del argumento LongitudCadena. El argumento LongitudCadena puede tener cualquier valor entero (positivo o negativo), pero el relleno solo se lleva a cabo si el valor de LongitudCadena es mayor que el número de caracteres de CadenaParaRellenar. Si CadenaParaRellenar tiene más caracteres que el valor de LongitudCadena, entonces CadenaParaRellenar se deja como está.

Ejemplos

- **altova:pad-string-left**('AP', 1, 'Z') devuelve 'AP'
- **altova:pad-string-left**('AP', 2, 'Z') devuelve 'AP'
- **altova:pad-string-left**('AP', 3, 'Z') devuelve 'ZAP'
- **altova:pad-string-left**('AP', 4, 'Z') devuelve 'ZZAP'
- **altova:pad-string-left**('AP', -3, 'Z') devuelve 'AP'
- **altova:pad-string-left**('AP', 3, 'YZ') devuelve un error indicando que el carácter de relleno es demasiado largo.

pad-string-right [altova:]

altova:pad-string-right(CadenaParaRellenar *como xs:string*, **LongitudCadena** *como xs:integer*, **CarácterRelleno** *como xs:string*) como **xs:string** **XP3.1** **XQ3.1**

El argumento CarácterRelleno es un solo carácter. Se añade a la derecha de la cadena para aumentar el número de caracteres de la CadenaParaRellenar, de modo que este número equivalga al valor entero del argumento LongitudCadena. El argumento LongitudCadena puede tener cualquier valor entero (positivo o negativo), pero el relleno solo se lleva a cabo si el valor de LongitudCadena es mayor que el número de caracteres de CadenaParaRellenar. Si CadenaParaRellenar tiene más caracteres que el valor de LongitudCadena, entonces CadenaParaRellenar se deja como está.

Ejemplos

- **altova:pad-string-right**('AP', 1, 'Z') devuelve 'AP'
- **altova:pad-string-right**('AP', 2, 'Z') devuelve 'AP'
- **altova:pad-string-right**('AP', 3, 'Z') devuelve 'APZ'
- **altova:pad-string-right**('AP', 4, 'Z') devuelve 'APZZ'
- **altova:pad-string-right**('AP', -3, 'Z') devuelve 'AP'
- **altova:pad-string-right**('AP', 3, 'YZ') devuelve un error indicando que el carácter de relleno es demasiado largo.

repeat-string [altova:]

altova:repeat-string(CadenaEntrada *as xs:string*, **Repeticiones** *as xs:integer*) como **xs:string** **XP2 XQ1 XP3.1 XQ3.1**

Genera una cadena que está compuesta por el primer argumento `CadenaEntrada` repetida tantas veces como indique el argumento `Repeticiones`.

Ejemplo

- **altova:repeat-string**("Altova #", 3)
  devuelve `Altova #Altova #Altova #"`

substring-after-last [altova:]

altova:substring-after-last(CadenaPrincipal *as xs:string*, **CadenaPrueba** *as xs:string*) como **xs:string** **XP3.1 XQ3.1**

Si **CadenaPrueba** se encuentra en **CadenaPrincipal**, la función devuelve la subcadena que aparece después de **CadenaPrueba** en **CadenaPrincipal**. Si **CadenaPrueba** no está en **CadenaPrincipal**, entonces devuelve la cadena vacía. Si **CadenaPrueba** es una cadena vacía, entonces devuelve la **CadenaPrincipal** entera. Si **CadenaPrueba** aparece varias veces en **CadenaPrincipal**, la función devuelve la subcadena que aparece después de la última **CadenaPrueba**.

Ejemplos

- **altova:substring-after-last**('ABCDEFGH', 'B') devuelve 'CDEFGH'
- **altova:substring-after-last**('ABCDEFGH', 'BC') devuelve 'DEFGH'
- **altova:substring-after-last**('ABCDEFGH', 'BD') devuelve ''
- **altova:substring-after-last**('ABCDEFGH', 'Z') devuelve ''
- **altova:substring-after-last**('ABCDEFGH', '') devuelve 'ABCDEFGH'
- **altova:substring-after-last**('ABCD-ABCD', 'B') devuelve 'CD'
- **altova:substring-after-last**('ABCD-ABCD-ABCD', 'BCD') devuelve ''

substring-before-last [altova:]

altova:substring-before-last(CadenaPrincipal *as xs:string*, **CadenaPrueba** *as xs:string*) como **xs:string** **XP3.1 XQ3.1**

Si **CadenaPrueba** se encuentra en **CadenaPrincipal**, la función devuelve la subcadena que aparece después de **CadenaPrueba** en **CadenaPrincipal**. Si **CadenaPrueba** no está en **CadenaPrincipal**, entonces devuelve la cadena vacía. Si **CadenaPrueba** es una cadena vacía, entonces devuelve la **CadenaPrincipal** entera. Si **CadenaPrueba** aparece varias veces en **CadenaPrincipal**, la función devuelve la subcadena que aparece antes de la última **CadenaPrueba**.

Ejemplos

- **altova:substring-before-last**('ABCDEFGH', 'B') devuelve 'A'
- **altova:substring-before-last**('ABCDEFGH', 'BC') devuelve 'A'
- **altova:substring-before-last**('ABCDEFGH', 'BD') devuelve ''
- **altova:substring-before-last**('ABCDEFGH', 'Z') devuelve ''
- **altova:substring-before-last**('ABCDEFGH', '') devuelve ''

 © 2018 Altova GmbH

- **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

trim-string [altova:]

altova:trim-string(**CadenaEntrada** *as xs:string*) como **xs:string** **XP3.1** **XQ3.1**

Esta función toma un argumento xs:string, quita los espacios en blanco iniciales y finales y devuelve un xs:string "recortado".

Ejemplos

- **altova:trim-string**(" Hello World ")) devuelve "Hello World"
- **altova:trim-string**("Hello World ")) devuelve "Hello World"
- **altova:trim-string**(" Hello World")) devuelve "Hello World"
- **altova:trim-string**("Hello World")) devuelve "Hello World"
- **altova:trim-string**("Hello World")) devuelve "Hello World"

trim-string-left [altova:]

altova:trim-string-left(CadenaEntrada *as xs:string*) como **xs:string** **XP3.1** **XQ3.1**

Esta función toma un argumento `xs:string`, quita los espacios en blanco iniciales y devuelve un `xs:string` recortado por la izquierda.

Ejemplos

- **altova:trim-string-left**(" Hello World ")) devuelve "Hello World "
- **altova:trim-string-left**("Hello World ")) devuelve "Hello World "
- **altova:trim-string-left**(" Hello World")) devuelve "Hello World"
- **altova:trim-string-left**("Hello World")) devuelve "Hello World"
- **altova:trim-string-left**("Hello World")) devuelve "Hello World"

trim-string-right [altova:]

altova:trim-string-right(CadenaEntrada *as xs:string*) como **xs:string** **XP3.1** **XQ3.1**

Esta función toma un argumento `xs:string`, quita los espacios en blanco finales y devuelve una cadena `xs:string` recortada por la derecha.

Ejemplos

- **altova:trim-string-right**(" Hello World ")) devuelve " Hello World"
- **altova:trim-string-right**("Hello World ")) devuelve "Hello World"
- **altova:trim-string-right**(" Hello World")) devuelve " Hello World"
- **altova:trim-string-right**("Hello World")) devuelve "Hello World"
- **altova:trim-string-right**("Hello World")) devuelve "Hello World"

10.1.8 Funciones XPath/XQuery varias

Estas funciones de extensión XPath/XQuery generales son compatibles con la versión actual de RaptorXML+XBRL Server 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**

Funciones XSLT (en expresiones XPath en XSLT):	XSLT1 XSLT2 XSLT3
Funciones XQuery (en expresiones XQuery en XQuery):	XQ1 XQ3.1

- 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() as **xs:string** XP2 XQ1 XP3.1 XQ3.1
  Genera una cadena única de la interfaz gráfica del usuario.
  - *Ejemplo*
    - **altova:generate-guid**() devuelve (por ejemplo) `85F971DA-17F3-4E4E-994E-99137873ACCD`

[Subir]

10.1.9 Funciones de extensión para gráficos

Las funciones para gráficos que aparecen a continuación sirven para crear, generar y guardar gráficos como imágenes. Esta funciones son compatibles con la versión actual de su producto de Altova. No obstante, tenga en cuenta que en futuras versiones del producto algunas funciones pueden dejar de ser compatibles o su comportamiento puede cambiar. Por tanto, consulte siempre la documentación del producto para conocer el funcionamiento de estas funciones en cada versión del producto.

Las funciones para gráficos son **funciones XPath** (no XSLT) y las hay de dos tipos:

- Funciones para generar y guardar gráficos
- Funciones para crear gráficos

Nota: las funciones para gráficos solamente son compatibles con los **productos servidor** de Altova y con las ediciones **Enterprise Edition** de las herramientas de escritorio de Altova.

Nota: en los **productos servidor** de Altova los formatos de imagen compatibles para gráficos son `jpg`, `png` y `bmp`. La opción más recomendable es `png` porque no pierde información y

es un formato comprimido. En las ediciones **Enterprise Edition** de las herramientas de escritorio de Altova, los formatos compatibles son `jpg`, `png`, `bmp` y `gif`.

Funciones para generar y guardar gráficos

Estas funciones toman el objeto del gráfico (obtenido con las funciones de creación de gráficos) y generan una imagen o guardan una imagen en un archivo.

`generate-chart-image` `($chart, $width, $height, $encoding) as atomic`

donde

- `$chart` es el componente de extensión de gráficos obtenido con la función `create-chart`
- `$width` y `$height` deben especificarse con una unidad de longitud
- `$encoding` puede ser `binarytobase64` o `binarytobase16`

La función devuelve la imagen del gráfico en la codificación elegida.

`generate-chart-image` `($chart, $width, $height, $encoding, $imagetype) as atomic`

donde

- `$chart` es el componente de extensión de gráficos obtenido con la función `create-chart`
- `$width` y `$height` deben especificarse con una unidad de longitud
- `$encoding` puede ser `base64Binary` o `hexBinary`
- `$imagetype` puede ser uno de estos formatos de imagen: `png`, `gif`, `bmp`, `jpg`, `jpeg`. Recuerde que el formato `gif` no es compatible con los productos servidor de Altova (*ver nota al principio de este apartado*)

La función devuelve la imagen del gráfico en la codificación y formato de imagen elegidos.

`save-chart-image` `($chart, $filename, $width, $height) as empty()` ***(solo en Windows)***

donde

- `$chart` es el componente de extensión de gráficos obtenido con la función `create-chart`
- `$filename` es la ruta de acceso del archivo y el nombre de archivo donde se debe guardar la imagen del gráfico
- `$width` y `$height` deben especificarse con una unidad de longitud

La función guarda la imagen del gráfico en el archivo especificado en `$filename`.

 © 2018 Altova GmbH

save-chart-image ($chart, $filename, $width, $height, $imagetype) as empty()
(solo en Windows)

donde

- $chart es el componente de extensión de gráficos obtenido con la función create-chart
- $filename es la ruta de acceso del archivo y el nombre de archivo donde se debe guardar la imagen del gráfico
- $width y $height deben especificarse con una unidad de longitud
- $imagetype puede ser uno de estos formatos de imagen: png, gif, bmp, jpg, jpeg. Recuerde que el formato gif no es compatible con los productos servidor de Altova (*ver nota al principio de este apartado*)

La función guarda la imagen del gráfico en el archivo especificado en $filename en el formato de imagen elegido.

Funciones para crear gráficos

Puede usar estas funciones para crear gráficos.

create-chart($chart-config, $chart-data-series*) como componente de extensión de gráficos

donde

- $chart-config es el componente de extensión chart-config obtenido con la función create-chart-config o con la función create-chart-config-from-xml
- $chart-data-series es el componente de extensión chart-data-series obtenido con la función create-chart-data-series o con la función create-chart-data-series-from-rows

La función devuelve un componente de extensión de gráficos, que se crea a partir de los datos suministrados con los argumentos.

create-chart-config($type-name, $title) como componente de extensión de gráficos chart-config

donde

- $type-name especifica el tipo de gráfico que se debe crear: Pie, Pie3d, BarChart, BarChart3d, BarChart3dGrouped, LineChart, ValueLineChart, RoundGauge, BarGauge
- $title es el nombre del gráfico

La función devuelve un componente de extensión chart-config que contiene los datos de

configuración del gráfico.

create-chart-config-from-xml($xml-struct) como componente de extensión de gráficos chart-config

donde

- $xml-struct es la estructura XML que contiene los datos de configuración del gráfico

La función devuelve un componente de extensión chart-config que contiene los datos de configuración del gráfico. Estos datos se suministran en un fragmento de código XML.

create-chart-data-series($series-name?, $x-values*, $y-values*) como componente de extensión de gráficos chart-data-series

donde

- $series-name especifica el nombre de la serie
- $x-values presenta la lista de valores del eje X
- $y-values presenta la lista de valores del eje Y

La función devuelve un componente de extensión chart-data-series que contiene los datos necesarios para generar el gráfico: es decir, el nombre de las series y los datos de los ejes.

create-chart-data-row(x, y1, y2, y3, ...) como componente de extensión de gráficos chart-data-x-*N*y-row

donde

- x es el valor de la columna del eje X de la fila de datos del gráfico
- yN son los valores de las columnas del eje Y

La función devuelve un componente de extensión chart-data-x-Ny-row, que contiene los datos para la columna del eje X y las columnas del eje Y de una sola serie.

create-chart-data-series-from-rows($series-names as xs:string*, $row*) como componente de extensión de gráficos chart-data-series

donde

- $series-name es el nombre de la series que se debe crear
- $row es el componente de extensión chart-data-x-Ny-row que se debe crear como serie

La función devuelve un componente de extensión chart-data-series, que contiene los datos

para el eje X y el eje Y de la serie.

create-chart-layer`($chart-config, $chart-data-series*) como componente de extensión de gráficos chart-layer`

donde

- `$chart-config` es el componente de extensión chart-config obtenido con la función `create-chart-config` o con la función `create-chart-config-from-xml`
- `$chart-data-series` es el componente de extensión chart-data-series que se obtiene con la función `create-chart-data-series` o con la función `create-chart-data-series-from-rows`

La función devuelve un componente de extensión chart-layer, que contiene los datos de la capa de gráfico.

create-multi-layer-chart`($chart-config, $chart-data-series*, $chart-layer*)`

donde

- `$chart-config` es el componente de extensión chart-config obtenido con la función `create-chart-config` o con la función `create-chart-config-from-xml`
- `$chart-data-series` es el componente de extensión chart-data-series obtenido con la función `create-chart-data-series` o con la función `create-chart-data-series-from-rows`
- `$chart-layer` es el componente de extensión de gráficos multicapas obtenido con la función `create-chart-layer`

La función devuelve un componente de extensión de gráficos multicapas.

create-multi-layer-chart`($chart-config, $chart-data-series*, $chart-layer*, xs:boolean $mergecategoryvalues)`

donde

- `$chart-config` es el componente de extensión chart-config obtenido con la función `create-chart-config` o con la función `create-chart-config-from-xml`
- `$chart-data-series` es el componente de extensión chart-data-series obtenido con la función `create-chart-data-series` o con la función `create-chart-data-series-from-rows`
- `$chart-layer` es el componente de extensión de gráficos multicapa obtenido con la función `create-chart-layer`

La función devuelve un componente de extensión de gráficos multicapa.

10.1.9.1 Estructura XML de los datos de gráficos

A continuación puede ver un fragmento de código XML con datos de gráfico, tal y como aparecería para las funciones de extensión para gráficos de Altova. Esto afecta al aspecto del gráfico. No todos los elementos se utilizan para todos los tipos de gráfico. Por ejemplo, el elemento `<Pie>` se omite en los gráficos de barras.

Nota: las funciones para gráficos son compatibles solamente con las ediciones **Enterprise** y **Server**.

```
<chart-config>
        <General
        SettingsVersion="1"   debe darse
        ChartKind="BarChart"   Pie, Pie3d, BarChart, StackedBarChart, BarChart3d,
BarChart3dGrouped, LineChart, ValueLineChart, AreaChart, StackedAreaChart, RoundGauge,
BarGauge, CandleStick
        BKColor="#ffffff"   Color
        BKColorGradientEnd="#ffffff"   Color. En caso de degradado, BKColor y
BKColorGradientEnd definen el color del degradado
        BKMode="#ffffff"   Solid, HorzGradient, VertGradient
        BKFile="Ruta+NombreArchivo"   Cadena. Si el archivo existe, su contenido se dibuja
sobre el fondo.
        BKFileMode="Stretch"   Stretch, ZoomToFit, Center, Tile
        ShowBorder="1"   Bool
        PlotBorderColor="#000000"   Color
        PlotBKColor="#ffffff"   Color
        Title=""   Cadena de texto
        ShowLegend="1"   Bool
        OutsideMargin="3.%"   PorcentajeOPíxel
        TitleToPlotMargin="3.%"   PorcentajeOPíxel
        LegendToPlotMargin="3.%"   PorcentajeOPíxel
        Orientation="vert"   Enums. Los valores posibles son: vert, horz
        >
        <TitleFont
           Color="#000000"   Color
           Name="Tahoma"   Cadena de texto
           Bold="1"   Bool
           Italic="0"   Bool
           Underline="0"   Bool
           MinFontHeight="10.pt"   TamañoFuente (solo valores pt)
           Size="8.%"   TamañoFuente />
        <LegendFont
           Color="#000000"
           Name="Tahoma"
           Bold="0"
           Italic="0"
           Underline="0"
           MinFontHeight="10.pt"
           Size="3.5%" />
        <AxisLabelFont
           Color="#000000"
```

```
            Name="Tahoma"
            Bold="1"
            Italic="0"
            Underline="0"
            MinFontHeight="10.pt"
            Size="5.%" />
        </General>

        <Line
                ConnectionShapeSize="1.%"   PorcentajeOPixel
                DrawFilledConnectionShapes="1"   Bool
                DrawOutlineConnectionShapes="0"   Bool
                DrawSlashConnectionShapes="0"   Bool
                DrawBackslashConnectionShapes="0"   Bool
        />

        <Bar
                ShowShadow="1"   Bool
                ShadowColor="#a0a0a0"   Color
                OutlineColor="#000000"   Color
                ShowOutline="1"   Bool
        />

        <Area
                Transparency="0"   UINT ( 0-255 ) 255 es totalmente transparente y 0 es
opaco
                OutlineColor="#000000"   Color
                ShowOutline="1"   Bool
        />

        <CandleStick
                FillHighClose="0"   Bool. Si es 0, el cuerpo está vacío. Si es 1,
FillColorHighClose se usa para el cuerpo de la vela
                FillColorHighClose="#ffffff"   Color. Para el cuerpo de la vela cuando el
valor de close > que el valor de open
                FillHighOpenWithSeriesColor="1"   Bool. Si es true, el color de la serie se
usa para rellenar el cuerpo de la vela cuando el valor de open > que el valor de close
                FillColorHighOpen="#000000"   Color. Para el cuerpo de la vela cuando el
valor de open > que el valor de close y FillHighOpenWithSeriesColor es false
        />

        <Colors   Combinación de colores definida por el usuario. Este elemento está vacío por
defecto, excepto el estilo, y no tiene atributos Color
                UseSubsequentColors ="1"  Booleano. Si es 0, entonces se superpone el
color. Si es 1, se usan los colores siguientes de la capa de gráfico anterior
                Style="User"   Valores posibles: "Default", "Grayscale", "Colorful", "Pastel",
"User"
                Colors="#52aca0"   Color: solamente se añade para el conjunto de colores
definido por el usuario
                Colors1="#d3c15d"   Color: solamente se añade para el conjunto de colores
definido por el usuario
                Colors2="#8971d8"   Color: solamente se añade para el conjunto de colores
definido por el usuario
                ...
```

```
            ColorsN=""   Cada conjunto de colores puede tener un máximo de diez colores,
de Colors a Colors9
      </Colors>

      <Pie
            ShowLabels="1"   Bool
            OutlineColor="#404040"   Color
            ShowOutline="1"   Bool
            StartAngle="0."   Double
            Clockwise="1"   Bool
            Draw2dHighlights="1"   Bool
            Transparency="0"   Int (De 0 a 255: 0 es opaco, 255 es totalmente
transparente)
            DropShadowColor="#c0c0c0"   Color
            DropShadowSize="5.%"   PorcentajeOPíxel
            PieHeight="10.%"   PorcentajeOPíxel. Los valores de píxel pueden ser
diferente en el resultado debido a la inclinación 3D
            Tilt="40.0"   Double (De 10 a 90: la inclinación 3D en grados de un gráfico
circular 3D)
            ShowDropShadow="1"   Bool
            ChartToLabelMargin="10.%"   PorcentajeOPíxel
            AddValueToLabel="0"   Bool
            AddPercentToLabel="0"   Bool
            AddPercentToLabels_DecimalDigits="0"  UINT ( 0 – 2 )
            >
            <LabelFont
                  Color="#000000"
                  Name="Arial"
                  Bold="0"
                  Italic="0"
                  Underline="0"
                  MinFontHeight="10.pt"
                  Size="4.%"    />
      </Pie>

      <XY>
            <XAxis   Axis
                  AutoRange="1"   Bool
                  AutoRangeIncludesZero="1"   Bool
                  RangeFrom="0."   Double: intervalo manual
                  RangeTill="1."   Double : intervalo manual
                  LabelToAxisMargin="3.%"   PorcentajeOPíxel
                  AxisLabel=""   Cadena de texto
                  AxisColor="#000000"   Color
                  AxisGridColor="#e6e6e6"   Color
                  ShowGrid="1"   Bool
                  UseAutoTick="1"   Bool
                  ManualTickInterval="1."   Double
                  AxisToChartMargin="0.px"   PorcentajeOPíxel
                  TickSize="3.px"   PorcentajeOPíxel
                  ShowTicks="1"   Bool
                  ShowValues="1"   Bool
                  AxisPosition="LeftOrBottom"   Enums: "LeftOrBottom",
"RightOrTop", "AtValue"
```

```
                AxisPositionAtValue = "0"   Double
                >
                <ValueFont
                        Color="#000000"
                        Name="Tahoma"
                        Bold="0"
                        Italic="0"
                        Underline="0"
                        MinFontHeight="10.pt"
                        Size="3.%"    />
        </XAxis>
        <YAxis   Eje (igual que XAxis)
                AutoRange="1"
                AutoRangeIncludesZero="1"
                RangeFrom="0."
                RangeTill="1."
                LabelToAxisMargin="3.%"
                AxisLabel=""
                AxisColor="#000000"
                AxisGridColor="#e6e6e6"
                ShowGrid="1"
                UseAutoTick="1"
                ManualTickInterval="1."
                AxisToChartMargin="0.px"
                TickSize="3.px"
                ShowTicks="1"   Bool
                ShowValues="1"   Bool
                AxisPosition="LeftOrBottom"   Enums: "LeftOrBottom",
"RightOrTop", "AtValue"
                AxisPositionAtValue = "0"   Double
                >
                <ValueFont
                        Color="#000000"
                        Name="Tahoma"
                        Bold="0"
                        Italic="0"
                        Underline="0"
                        MinFontHeight="10.pt"
                        Size="3.%"/>
        </YAxis>
    </XY>

    <XY3d
        AxisAutoSize="1"   Bool: Si es false, XSize y YSize definen la relación de
aspecto de los ejes x e y. Si es true, la relación de aspecto es igual a la ventana del gráfico
        XSize="100.%"   PorcentajeOPíxel. Los valores en píxel pueden ser diferentes
en el resultado debido a la inclinación 3D y a la opción de ajustar al tamaño
        YSize="100.%"   PorcentajeOPíxel. Los valores en píxel pueden ser diferentes
en el resultado debido a la inclinación 3D y a la opción de ajustar al tamaño
        SeriesMargin="30.%"   PorcentajeOPíxel. Los valores en píxel pueden ser
diferentes en el resultado debido a la inclinación 3D y a la opción de ajustar al tamaño
        Tilt="20."   Double. De -90 a +90 grados
        Rot="20."   Double. De -359 a +359 grados
        FoV="50.">   Double. Campo de visión: de 1 a 120 grados
        >
```

```
        <ZAxis
               AutoRange="1"
               AutoRangeIncludesZero="1"
               RangeFrom="0."
               RangeTill="1."
               LabelToAxisMargin="3.%"
               AxisLabel=""
               AxisColor="#000000"
               AxisGridColor="#e6e6e6"
               ShowGrid="1"
               UseAutoTick="1"
               ManualTickInterval="1."
               AxisToChartMargin="0.px"
               TickSize="3.px" >
               <ValueFont
                      Color="#000000"
                      Name="Tahoma"
                      Bold="0"
                      Italic="0"
                      Underline="0"
                      MinFontHeight="10.pt"
                      Size="3.%"/>
        </ZAxis>
</XY3d>

<Gauge
        MinVal="0."   Double
        MaxVal="100."   Double
        MinAngle="225"   UINT: -359-359
        SweepAngle="270"   UINT: 1-359
        BorderToTick="1.%"   PorcentajeOPíxel
        MajorTickWidth="3.px"   PorcentajeOPíxel
        MajorTickLength="4.%"   PorcentajeOPíxel
        MinorTickWidth="1.px"   PorcentajeOPíxel
        MinorTickLength="3.%"   PorcentajeOPíxel
        BorderColor="#a0a0a0"   Color
        FillColor="#303535"   Color
        MajorTickColor="#a0c0b0"   Color
        MinorTickColor="#a0c0b0"   Color
        BorderWidth="2.%"   PorcentajeOPíxel
        NeedleBaseWidth="1.5%"   PorcentajeOPíxel
        NeedleBaseRadius="5.%"   PorcentajeOPíxel
        NeedleColor="#f00000"   Color
        NeedleBaseColor="#141414"   Color
        TickToTickValueMargin="5.%"   PorcentajeOPíxel
        MajorTickStep="10."   Double
        MinorTickStep="5."   Double
        RoundGaugeBorderToColorRange="0.%"   PorcentajeOPíxel
        RoundGaugeColorRangeWidth ="6.%"   PorcentajeOPíxel
        BarGaugeRadius="5.%"   PorcentajeOPíxel
        BarGaugeMaxHeight="20.%"   PorcentajeOPíxel
        RoundGaugeNeedleLength="45.%"   PorcentajeOPíxel
        BarGaugeNeedleLength="3.%"   PorcentajeOPíxel
        >
        <TicksFont
```

```
                    Color="#a0c0b0"
                    Name="Tahoma"
                    Bold="0"
                    Italic="0"
                    Underline="0"
                    MinFontHeight="10.pt"
                    Size="4.%"
            />
            <ColorRanges>   Intervalos de color definidos por el usuario. Está vacío por
defecto y no tiene elementos secundarios
                    <Entry
                          From="50. "   Double
                          FillWithColor="1"   Bool
                          Color="#00ff00"   Color
                    />
                    <Entry
                          From="50.0"
                          FillWithColor="1"
                          Color="#ff0000"
                    />
                    ...
            </ColorRanges>
      </Gauge>
</chart-config>
```

10.1.9.2 Ejemplo: funciones para gráficos

A continuación puede ver un ejemplo de documento XSLT que muestra cómo usar las funciones de extensión para gráficos de Altova. Después ofrecemos un documento XML y una captura de pantalla de la imagen generada cuando el documento XML se procesa con el documento XSLT usando el motor XSLT 2.0.

Nota: las funciones para gráficos solamente son compatibles con las ediciones **Enterprise Edition** y **Server** de los productos de Altova.

Nota: para más información sobre las tablas de datos del gráfico, consulte la documentación de las herramientas XMLSpy y StyleVision de Altova.

Documento XSLT

Este documento XSLT usa las funciones de extensión para gráficos de Altova para generar un gráfico circular. Este XSLT se puede usar para procesar el documento XML que aparece más abajo.

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="2.0"
   xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
   xmlns:xs="http://www.w3.org/2001/XMLSchema"
   xmlns:altovaext="http://www.altova.com/xslt-extensions"
   exclude-result-prefixes="#all">
```

```
<xsl:output version="4.0" method="html" indent="yes" encoding="UTF-8"/>
<xsl:template match="/">
    <html>
        <head>
            <title>
                <xsl:text>HTML Page with Embedded Chart</xsl:text>
            </title>
        </head>
        <body>
            <xsl:for-each select="/Data/Region[1]">
                <xsl:variable name="extChartConfig" as="item()*">
                    <xsl:variable name="ext-chart-settings" as="item()*">
                        <chart-config>
                            <General
                                SettingsVersion="1"
                                ChartKind="Pie3d"
                                BKColor="#ffffff"
                                ShowBorder="1"
                                PlotBorderColor="#000000"
                                PlotBKColor="#ffffff"
                                Title="{@id}"
                                ShowLegend="1"
                                OutsideMargin="3.2%"
                                TitleToPlotMargin="3.%"
                                LegendToPlotMargin="6.%"
                                >
                                <TitleFont
                                    Color="#023d7d"
                                    Name="Tahoma"
                                    Bold="1"
                                    Italic="0"
                                    Underline="0"
                                    MinFontHeight="10.pt"
                                    Size="8.%" />
                            </General>
                        </chart-config>
                    </xsl:variable>
                    <xsl:sequence select="altovaext:create-chart-config-from-
xml( $ext-chart-settings )"/>
                </xsl:variable>
                <xsl:variable name="chartDataSeries" as="item()*">
                    <xsl:variable name="chartDataRows" as="item()*">
                        <xsl:for-each select="(Year)">
                            <xsl:sequence select="altovaext:create-chart-data-
row( (@id), ( .) )"/>
                        </xsl:for-each>
                    </xsl:variable>
                    <xsl:variable name="chartDataSeriesNames" as="xs:string*"
select=" ( (&quot;Series 1&quot;), &apos;&apos; )[1]"/>
                    <xsl:sequence
                        select="altovaext:create-chart-data-series-from-
rows( $chartDataSeriesNames, $chartDataRows)"/>
                </xsl:variable>
```

```
                <xsl:variable name="ChartObj" select="altovaext:create-
chart( $extChartConfig, ( $chartDataSeries), false() )"/>
                <xsl:variable name="sChartFileName" select="'mychart1.png'"/>
                <img src="{$sChartFileName, altovaext:save-chart-
image( $ChartObj, $sChartFileName, 400, 400 ) }"/>
            </xsl:for-each>
        </body>
    </html>
  </xsl:template>
</xsl:stylesheet>
```

Documento XML

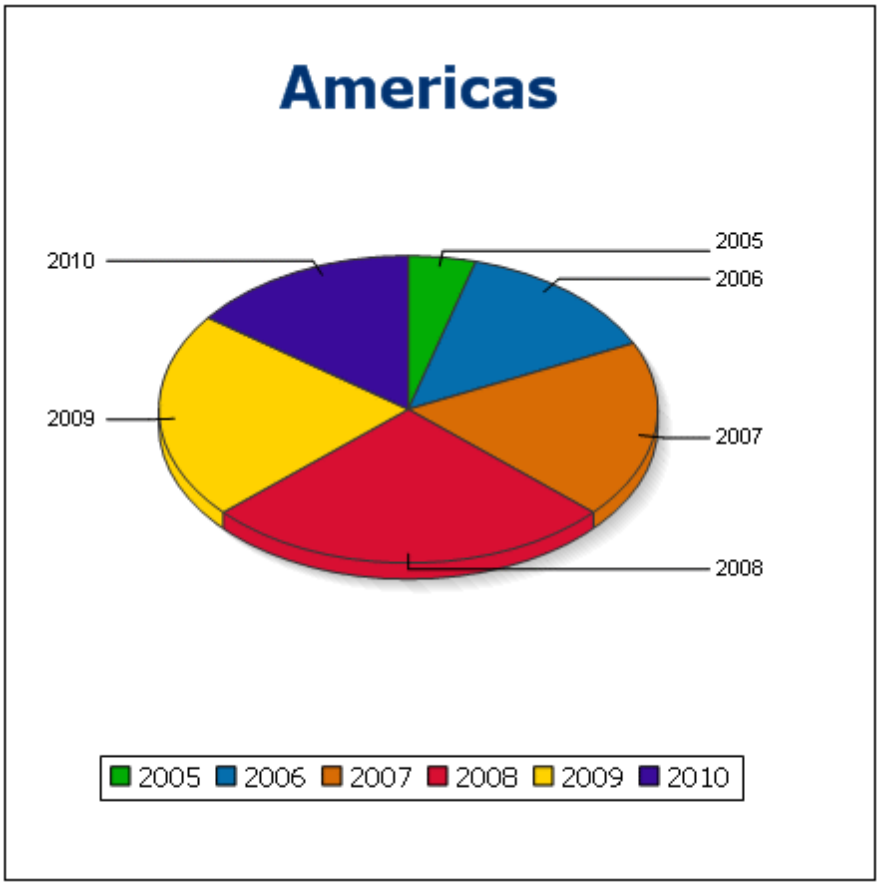
Este documento XML se puede procesar con el XSLT que aparece arriba. Los datos del documento XML se usan para generar el gráfico circular de la imagen que aparece más abajo.

```
<?xml version="1.0" encoding="UTF-8"?>
<Data xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:noNamespaceSchemaLocation="YearlySales.xsd">
   <ChartType>Pie Chart 2D</ChartType>
   <Region id="Americas">
      <Year id="2005">30000</Year>
      <Year id="2006">90000</Year>
      <Year id="2007">120000</Year>
      <Year id="2008">180000</Year>
      <Year id="2009">140000</Year>
      <Year id="2010">100000</Year>
   </Region>
   <Region id="Europe">
      <Year id="2005">50000</Year>
      <Year id="2006">60000</Year>
      <Year id="2007">80000</Year>
      <Year id="2008">100000</Year>
      <Year id="2009">95000</Year>
      <Year id="2010">80000</Year>
   </Region>
   <Region id="Asia">
      <Year id="2005">10000</Year>
      <Year id="2006">25000</Year>
      <Year id="2007">70000</Year>
      <Year id="2008">110000</Year>
      <Year id="2009">125000</Year>
      <Year id="2010">150000</Year>
   </Region>
</Data>
```

Imagen de salida

Este gráfico circular se generó al procesar el documento XML anterior con el archivo XSLT que

aparece al principio de este apartado.



10.1.10 Funciones para códigos de barras

Los motores XSLT de Altova usan bibliotecas Java de terceros para crear códigos de barras. A continuación enumeramos las clases y los métodos públicos utilizados. Las clases se empaquetan en `AltovaBarcodeExtension.jar`, que está en la carpeta `<CarpetaArchivosPrograma>\Altova\Common2018\jar`.

Las bibliotecas Java utilizadas están en las subcarpetas de la carpeta `<CarpetaArchivosPrograma>\Altova\Common2018\jar`:

- `barcode4j\barcode4j.jar` (sitio web: http://barcode4j.sourceforge.net/)
- `zxing\core.jar` (sitio web: http://code.google.com/p/zxing/)

Los archivos de licencia están también en estas carpetas.

Paquete com.altova.extensions.barcode

El paquete `com.altova.extensions.barcode` se utiliza para generar la mayoría de códigos de barras.

Se utilizan estas clases:

```
public class BarcodeWrapper
       static BarcodeWrapper newInstance( String name, String msg, int dpi, int
orientation, BarcodePropertyWrapper[] arrProperties )
       double getHeightPlusQuiet()
       double getWidthPlusQuiet()
       org.w3c.dom.Document generateBarcodeSVG()
       byte[] generateBarcodePNG()
       String generateBarcodePngAsHexString()
```

`public class` **`BarcodePropertyWrapper`** *Utilizada para almacenar las propiedades del código de barras que se establecerán más tarde de forma dinámica*

```
   BarcodePropertyWrapper( String methodName, String propertyValue )
       BarcodePropertyWrapper( String methodName, Integer propertyValue )

       BarcodePropertyWrapper( String methodName, Double propertyValue )
       BarcodePropertyWrapper( String methodName, Boolean propertyValue )
       BarcodePropertyWrapper( String methodName, Character propertyValue )
       String getMethodName()
       Object getPropertyValue()
```

`public class` **`AltovaBarcodeClassResolver`** *Registra la clase* `com.altova.extensions.barcode.proxy.zxing.QRCodeBean` *para el bean* `qrcode`*, además de las clases registradas por* `org.krysalis.barcode4j.DefaultBarcodeClassResolver`.

Paquete com.altova.extensions.barcode.proxy.zxing

El paquete `com.altova.extensions.barcode.proxy.zxing` se utiliza para generar códigos de barras de tipo QRCode.

Se utilizan estas clases:

`class` **`QRCodeBean`**

- *Extiende* `org.krysalis.barcode4j.impl.AbstractBarcodeBean`
- *Crea una interfaz* `AbstractBarcodeBean` *para* `com.google.zxing.qrcode.encoder`

```
       void generateBarcode(CanvasProvider canvasImp, String msg)
       void setQRErrorCorrectionLevel(QRCodeErrorCorrectionLevel level)
       BarcodeDimension calcDimensions(String msg)
       double getVerticalQuietZone()
       double getBarWidth()
```

`class` **`QRCodeErrorCorrectionLevel`** *Nivel de corrección de errores para* `QRCode`

```
       static QRCodeErrorCorrectionLevel byName(String name)
       "L" = ~7% correction
       "M" = ~15% correction
```

```
"H" = ~25% correction
"Q" = ~30% correction
```

Ejemplo XSLT

A continuación puede ver un ejemplo de XSLT que ilustra el uso de funciones para códigos de barras en una hoja de estilos XSLT.

```
<?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:altova="http://www.altova.com"
   xmlns:altovaext="”http://www.altova.com/xslt-extensions”
   xmlns:altovaext-barcode="java:com.altova.extensions.barcode.BarcodeWrapper"
   xmlns:altovaext-barcode-
property="java:com.altova.extensions.barcode.BarcodePropertyWrapper">
   <xsl:output method="html" encoding="UTF-8" indent="yes"/>
   <xsl:template match="/">
      <html>
         <head><title/></head>
         <body>
            <img alt="barcode" src="{altovaext:get-temp-folder()}barcode.png"/>
         </body>
      </html>
      <xsl:result-document
         href="{altovaext:get-temp-folder()}barcode.png"
         method="text" encoding="base64tobinary" >
         <xsl:variable name="barcodeObject"
            select="altovaext-
barcode:newInstance(&apos;Code39&apos;,string(&apos;some value&apos;),
            96,0, (altovaext-barcode-property:new( &apos;setModuleWidth&apos;,
25.4 div 96 * 2 ) ) )"/>
         <xsl:value-of select="xs:base64Binary(xs:hexBinary(string(altovaext-
barcode:generateBarcodePngAsHexString($barcodeObject)) ))"/>
      </xsl:result-document>
   </xsl:template>
</xsl:stylesheet>
```

 © 2018 Altova GmbH

10.2 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 y .NET, así como scripts MSXSL para XSLT. También son compatibles con funciones XBRL para XSLT.

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
- Funciones de extensión .NET
- Funciones XBRL para XSLT
- Scripts MSXSL para XSLT

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.

10.2.1 Funciones de extensión Java

Puede usar una función de extensión Java dentro de una expresión XPath o XQuery para invocar un constructor Java o llamar a un método Java (estático o de instancia).

Un campo de una clase Java se trata como un método sin argumentos. Un campo puede ser estático o de instancia. Más adelante describimos cómo se accede a los campos estáticos y de instancia.

Este apartado tiene varias partes:

- Constructores Java
- Métodos estáticos y campos estáticos
- Métodos de instancia y campos de instancia
- Tipos de datos: conversión de XPath/XQuery en Java

- Tipos de datos: conversión de Java en XPath/XQuery

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 y Archivos Jar definidos por el usuario. 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.

10.2.1.1 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*.)
- 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*.)
- El archivo de clases está en un paquete. El archivo XSLT/XQuery está en una carpeta cualquiera. (*ver ejemplo*.)
- El archivo de clases no está en un paquete. El archivo XSLT/XQuery está una carpeta cualquiera. (*ver ejemplo*.)

Imaginemos que tenemos un archivo de clases que no está en un paquete y que está en la misma carpeta que el documento XSLT/XQuery. En este caso, puesto que en la carpeta se encuentran todas las clases, no es necesario especificar la ubicación del archivo. La sintaxis que se utiliza para identificar una clase es esta:

```
java:nombreClase
```

donde

`java:` indica que se está llamando a una función definida por el usuario (por defecto se cargan las clases Java del directorio actual)

`nombreClase` es el nombre de la clase del método elegido

La clase se identifica en un URI de espacio de nombres y el espacio de nombres se usa como prefijo para la llamada al método.

El archivo de clases está en un paquete. El archivo XSLT/XQuery está en la misma carpeta que el paquete Java

El código que aparece a continuación llama al método `getVehicleType()` de la clase `Car` del paquete `com.altova.extfunc`. El paquete `com.altova.extfunc` está en la carpeta `JavaProject`. El archivo XSLT también está en la carpeta `JavaProject`.

```
<xsl:stylesheet version="2.0"
```

```
        xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
        xmlns:xs="http://www.w3.org/2001/XMLSchema"
        xmlns:fn="http://www.w3.org/2005/xpath-functions"
        xmlns:car="java:com.altova.extfunc.Car" >
<xsl:output exclude-result-prefixes="fn car xsl fo xs"/>

<xsl:template match="/">
   <a>
   <xsl:value-of select="car:getVehicleType()"/>
   </a>
</xsl:template>

</xsl:stylesheet>
```

El archivo de clases no está en un paquete. 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` del paquete `com.altova.extfunc`. El archivo de clases `Car` está en esta carpeta: `JavaProject/com/altova/extfunc`. 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` 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-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.

10.2.1.2 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.

10.2.1.3 Constructores

Una función de extensión se puede usar para llamar a un constructor Java. A todos los constructores se les llama con la pseudofunción `new()`.

Si el resultado de una llamada a un constructor Java se puede convertir de manera implícita a tipos de datos XPath/XQuery, 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*):
  ```
  <xsl:value-of select="date:toString(date:new())"
  xmlns:date="java:java.util.Date" />
  ```

10.2.1.4 Métodos estáticos y campos estáticos

La llamada a un método estático la hace directamente su nombre Java y se hace presentando los argumentos para el método. A los campos estáticos (es decir, los métodos que no toman argumentos), como los campos de valor constante `E` y `PI`, se accede sin especificar ningún argumento.

Ejemplos de código XSLT

Aquí puede ver varios ejemplos de cómo se llama a métodos y campos estáticos:

```
<xsl:value-of xmlns:jMath="java:java.lang.Math"
              select="jMath:cos(3.14)" />

<xsl:value-of xmlns:jMath="java:java.lang.Math"
              select="jMath:cos( jMath:PI() )" />

<xsl:value-of xmlns:jMath="java:java.lang.Math"
              select="jMath:E() * jMath:cos(3.14)" />
```

Observe que las funciones de extensión anteriores tienen el formato `prefijo:nombreFunción()`. En los tres ejemplos anteriores, el prefijo es `jMath:`, que está asociado al URI de espacio de nombres `java:java.lang.Math`. (El URI de espacio de nombres debe empezar por `java:`. En los ejemplos anteriores se extiende para contener el nombre de la clase (`java.lang.Math`).) La parte `nombreFunción()` de las funciones de extensión debe coincidir con el nombre de una clase pública (p. ej. `java.lang.Math`) seguido del nombre de un método estático público con sus argumentos (como `cos(3.14)`) o de un campo estático público (como `PI()`).

En los tres ejemplos anteriores, el nombre de la clase se incluyó en el URI de espacio de nombres. Si no estuviera en el URI de espacio de nombres, se incluiría en la parte `nombreFunción()` de la función de extensión. Por ejemplo:

```
<xsl:value-of xmlns:java="java:"
              select="java:java.lang.Math.cos(3.14)" />
```

Ejemplo de XQuery

Un ejemplo de XQuery similar sería:

```
<cosine xmlns:jMath="java:java.lang.Math">
   {jMath:cos(3.14)}
</cosine>
```

10.2.1.5 Métodos de instancia y campos de instancia

A un método de instancia se le pasa un objeto Java como primer argumento de la llamada a método. Dicho objeto Java suele crearse usando una función de extensión (por ejemplo, una llamada a un constructor) o un parámetro o una variable de hoja de estilos. Un ejemplo de código XSLT de este tipo sería:

```
<xsl:stylesheet version="1.0" exclude-result-prefixes="date"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
  xmlns:date="java:java.util.Date"
  xmlns:jlang="java:java.lang">
```

 © 2018 Altova GmbH

```
  <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`.

10.2.1.6 Tipos de datos: conversión de XPath/XQuery en Java

Cuando se llama a una función Java desde dentro de una expresión XPath/XQuery, el tipo de datos de los argumentos de la función es importante a la hora de determinar a cuál de las clases Java que tienen el mismo nombre se llama.

En Java se siguen estas reglas:

- Si hay más de un método Java con el mismo nombre, pero cada método tiene un número diferente de argumentos, entonces se selecciona el método Java que mejor se ajusta al número de argumentos de la llamada a función.
- Los tipos de datos de cadena, numéricos y booleanos de XPath/XQuery (*ver lista más abajo*) se convierten de forma implícita en el tipo de datos Java correspondiente. Si el tipo XPath/XQuery suministrado se puede convertir a más de un tipo Java (p. ej. `xs:integer`), entonces se selecciona el tipo Java que se declaró para el método seleccionado. Por ejemplo, si el método Java al que se llama es `fx(decimal)` y el tipo de datos XPath/XQuery suministrado es `xs:integer`, entonces `xs:integer` se convierte en el tipo de datos Java `decimal`.

La tabla que aparece a continuación enumera las conversiones implícitas de los tipos de cadena, numéricos y booleanos XPath/XQuery en tipos de datos Java.

`xs:string`	`java.lang.String`

xs:boolean	boolean (primitivo), java.lang.Boolean
xs:integer	int, long, short, byte, float, double y sus clases contenedoras, como java.lang.Integer
xs:float	float (primitivo), java.lang.Float, double (primitivo)
xs:double	double (primitivo), java.lang.Double
xs:decimal	float (primitivo), java.lang.Float, double(primitivo), java.lang.Double

Los subtipos de los tipos de datos XML Schema de la tabla anterior (que se usan en XPath y XQuery) también se convierten en los tipos Java correspondientes al tipo antecesor del subtipo.

En algunos casos quizás no sea posible seleccionar el método Java correcto usando la información dada. Por ejemplo, imagine que:

- El argumento presentado es un valor xs:untypedAtomic de 10 y está destinado al método mimétodo(float).
- Sin embargo, hay otro método en la clase que toma un argumento de otro tipo de datos: mimétodo(double).
- Puesto que los métodos tienen el mismo nombre y el tipo suministrado (xs:untypedAtomic) se puede convertir correctamente tanto en float como en double, es posible que xs:untypedAtomic se convierta en double en lugar de en float.
- Por consiguiente, el método seleccionado no será el método necesario y quizás no produzca el resultado esperado. Una solución es crear un método definido por el usuario con un nombre diferente y usar ese método.

Los tipos que no aparecen en la lista anterior (p. ej. xs:date) no se convertirán y generarán un error. No obstante, tenga en cuenta que en algunos casos, es posible crear el tipo Java necesario usando un constructor Java.

10.2.1.7 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).

10.2.2 Funciones de extensión .NET

Si trabaja en la plataforma .NET desde un equipo Windows, puede usar funciones de extensión escritas en cualquier lenguaje .NET (p. ej. C#). Una función de extensión .NET se puede usar dentro de una expresión XPath/XQuery para invocar un constructor, una propiedad o un método (estático o de instancia) de una clase .NET.

A una propiedad de una clase .NET se le llama usando la sintaxis `get_NombrePropiedad()`.

Este apartado tiene varias partes:

- Constructores
- Métodos estáticos y campos estáticos
- Métodos de instancia y campos de instancia
- Tipos de datos: conversión de XPath/XQuery en .NET
- Tipos de datos: conversión de .NET en XPath/XQuery

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:
```
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"
```

 © 2018 Altova GmbH

```
   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.

10.2.2.1 Constructores

Una función de extensión se puede usar para llamar a un constructor .NET. A todos los constructores se les llama con la pseudofunción `new()`. Si hay más de un constructor para una clase, entonces se selecciona el constructor que más se ajusta al número de argumentos suministrados. Si no se encuentra ningún constructor que coincida con los argumentos suministrados, entonces se genera el error "`No constructor found`".

Constructores que devuelven tipos de datos XPath/XQuery

Si el resultado de una llamada a un constructor .NET se puede convertir de forma implícita en tipos de datos XPath/XQuery, 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*):
```
<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(data:get_Month(date:new(2008, 4, 29)))"
xmlns:date="clitype:System.DateTime" />
```

10.2.2.2 Metodos estáticos y campos estáticos

La llamada a un método estático la hace directamente su nombre y se hace presentando los argumentos para el método. El nombre usado en la llamada debe ser el mismo que un método estático público de la clase especificada. Si el nombre del método y el número de argumentos que se dio en la llamada a función coincide con algún método de la clase, entonces los tipos de los argumentos presentados se evalúan para encontrar el resultado ideal. Si no se encuentra ninguna coincidencia, se emite un error.

Nota: un campo de una clase .NET se trata como si fuera un método sin argumentos. Para llamar a una propiedad se usa la sintaxis `get_nombrePropiedad()`.

Ejemplos

Este ejemplo de código XSLT muestra una llamada a un método con un argumento (`System.Math.Sin(arg)`):

```
<xsl:value-of select="math:Sin(30)" xmlns:math="clitype:System.Math"/>
```

Este ejemplo de código XSLT muestra una llamada a un campo (que se trata como si fuera un método sin argumentos) (`System.Double.MaxValue()`):

 © 2018 Altova GmbH

```
<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>
```

10.2.2.3 Métodos de instancia y campos de instancia

Un método de instancia es un método al que se le pasa un objeto .NET como primer argumento de la llamada al método. Este objeto .NET se suele crear usando una función de extensión (por ejemplo, una llamada a un constructor) o un parámetro o una variable de una hoja de estilos. Un ejemplo de código XSLT para este tipo de método sería:

```
<xsl:stylesheet version="2.0"
   xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
   xmlns:xs="http://www.w3.org/2001/XMLSchema"
   xmlns:fn="http://www.w3.org/2005/xpath-functions">
   <xsl:output method="xml" omit-xml-declaration="yes"/>
   <xsl:template match="/">
      <xsl:variable name="releasedate"
         select="date:new(2008, 4, 29)"
         xmlns:date="clitype:System.DateTime"/>
      <doc>
         <date>
            <xsl:value-of select="date:ToString(date:new(2008, 4, 29))"
               xmlns:date="clitype:System.DateTime"/>
         </date>
         <date>
            <xsl:value-of select="date:ToString($releasedate)"
               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.

10.2.2.4 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

 © 2018 Altova GmbH

Los subtipos de los tipos de datos XML Schema de la tabla anterior (que se usan en XPath y XQuery) también se convierten en los tipos .NET correspondientes al tipo antecesor del subtipo.

En algunos casos quizás no sea posible seleccionar el método .NET correcto usando la información dada. Por ejemplo, imagine que:

- El argumento presentado es un valor `xs:untypedAtomic` de 10 y está destinado al método `mimétodo(float)`.
- Sin embargo, hay otro método en la clase que toma un argumento de otro tipo de datos: `mimétodo(double)`.
- Puesto que los métodos tienen el mismo nombre y el tipo suministrado (`xs:untypedAtomic`) se puede convertir correctamente tanto en `float` como en `double`, es posible que `xs:untypedAtomic` se convierta en `double` en lugar de en `float`.
- Por consiguiente, el método seleccionado no será el método necesario y puede que no produzca el resultado esperado. Una solución es crear un método definido por el usuario con un nombre diferente y usar ese método.

Los tipos que no aparecen en la lista anterior (p. ej. `xs:date`) no se convertirán y generarán un error.

10.2.2.5 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).

10.2.3 Funciones XBRL para XSLT

Puede llamar desde un contexto XSLT a todas las funciones definidas en el registro de funciones XBRL para transformar documentos de instancia XBRL. Estas funciones XBRL se definen en uno de estos dos espacios de nombres:

`http://www.xbrl.org/2008/function/instance` (que suele utilizarse con el prefijo `xfi:`)
`http://www.xbrl.org/2010/function/formula` (que suele utilizarse con el prefijo `xff:`)

Por ejemplo, la función XBRL `xfi:context` se amplía hasta `http://www.xbrl.org/2008/function/instance:context` (asumiendo que este espacio de nombres se enlazara al prefijo `xfi:`).

Para ver la lista de funciones visite el sitio http://www.xbrl.org/functionregistry/functionregistry.xml.

10.2.4 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.

 © 2018 Altova GmbH

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.

 © 2018 Altova GmbH

Altova RaptorXML+XBRL Server 2018

Altova LicenseServer

11 Altova LicenseServer

Altova LicenseServer 2.7 (en adelante ***LicenseServer***) ofrece una ubicación central donde se administran todas las licencias de los productos de Altova. Las aplicaciones de Altova que se ejecuten en la red pueden tener asignadas licencias desde el servidor LicenseServer. Esto permite al administrador gran flexibilidad a la hora de gestionar y supervisar licencias.

Versión actual: 2.7 *

*** LicenseServer 2.7 puede utilizarse para asignar licencias a (i) productos de software de Altova versión 2018r2 o anterior y a (ii) Altova MobileTogether Server versión 4.1 o anterior. Con esta versión no se pueden asignar licencias a productos de Altova con una versión más reciente a la indicada en esta nota. Para más información sobre la compatibilidad entre productos consulte el apartado Actualizar Altova LicenseServer.*

Proceso de asignación de licencias con Altova LicenseServer

Para asignar una licencia a un producto de Altova con LicenseServer:

1. Inicie LicenseServer
2. Abra la página de configuración de LicenseServer, es decir, la interfaz de administración de LicenseServer en Windows, Linux o macOS.
3. Cargue los archivos de licencia de Altova que recibió al repositorio de licencias de LicenseServer. Esto se hace en la pestaña **License Pool** de la página de configuración
4. Registre los productos de Altova con LicenseServer.
5. En la pestaña de gestión de clientes **Client Management** de la página de configuración asigne licencias a los productos de Altova.

A partir de ese momento el administrador puede supervisar y administrar las licencias desde LicenseServer. Para más información consulte la *Referencia de la página de configuración*.

Nota: la página de configuración de LicenseServer no es compatible con SSL.

▼ Versiones de LicenseServer y compatibilidad con los productos de Altova

Cuando actualice su producto de Altova a una versión más reciente, asegúrese de actualizar también su servidor LicenseServer con la versión más reciente. Las versiones anteriores de los productos de Altova funcionarán con las conversiones más recientes de LicenseServer.

Es decir, si instala una versión nueva de un producto de Altova y la versión actual de su servidor LicenseServer no es la más reciente, desinstale la versión antigua de LicenseServer e instale la versión más reciente (disponible en el sitio web de Altova). Toda la información de registro y licencias almacenada en la versión antigua se guardará en una base de datos su equipo durante la desinstalación y se importará automáticamente a la versión nueva. Cuando instale una versión más reciente de LicenseServer, la versión más antigua se desinstalará antes de comenzar la instalación de la versión más reciente.

El número de versión del servidor LicenseServer que está instalado aparece al final de la página de configuración de LicenseServer (en todas las pestañas).

 © 2018 Altova GmbH

Versión actual: 2.7

Notas sobre la documentación

Esta documentación se divide en tres secciones:

- Información general sobre requisitos de red, proceso de instalación en Windows, Linux y macOS y descripción de Altova ServiceController.
- Asignación de licencias: describe paso a paso el proceso de asignación de licencias en Altova LicenseServer.
- Referencia de la página de configuración: describe la interfaz de administración de LicenseServer y sus funciones.

Última actualización: 6/22/2018

11.1 Información de red

Todos los equipos clientes que ejecutan productos servidor de Altova que necesitan una licencia deben tener acceso a un equipo servidor en el que debe estar instalado Altova LicenseServer. Los servidores de seguridad o firewalls (tanto del servidor como del cliente) deben permitir el flujo del tráfico de red con dicho servidor LicenseServer.

En el equipo donde está instalado LicenseServer, el **puerto 35355** se utiliza para distribuir licencias y, por tanto, debe estar abierto al tráfico de red con los equipos clientes.

Estos son los parámetros de red predeterminados y los requisitos de LicenseServer:

- *Para distribución de licencias con LicenseServer:*
  Uno de estos dos:
  Conexión TCP IPv4 en el puerto `35355`
  Conexión TCP IPv6 en el puerto `35355`

Para trabajar en tareas administrativas el servidor LicenseServer ofrece una interfaz web que utiliza el puerto 8088. El usuario puede configurar el puerto utilizado para la interfaz web según sus requisitos.

Conexión con el servidor maestro de licencias altova.com

Altova LicenseServer necesita comunicarse con el servidor maestro de licencias `altova.com` para validar y verificar los datos de licencia y para asegurarse de que los contratos de licencia de Altova se cumplen en todo momento. Esta comunicación se establece por HTTPS a través del puerto 443. Si tras completar la verificación inicial con el servidor maestro `altova.com`, Altova LicenseServer no puede volver a conectarse con `altova.com` durante más de 5 días (120 horas), se bloqueará el uso de los productos de Altova que estén conectados con Altova LicenseServer.

Cuando se pierde la conexión con el servidor maestro `altova.com`, la incidencia se registra en la pestaña **Messages** de la página de configuración de Altova LicenseServer. Además el administrador puede configurar Altova LicenseServer para que le envíe correos de alerta cuando se pierda la conexión con `altova.com`. Las opciones de configuración de estos correos de alerta se pueden definir en la pestaña **Settings** de la página de configuración.

Cómo identifica LicenseServer los equipos clientes

Cuando a un producto de Altova se le asigna una licencia con LicenseServer, esa licencia queda almacenada en LicenseServer como asignada a un equipo cliente específico. LicenseServer identifica el equipo cliente por su nombre de host, que obtiene al realizar una búsqueda en el servidor DNS de la dirección IP del cliente. Si no puede obtener un nombre de host con esa búsqueda, usará la dirección IP del cliente como identificador. Por tanto, cada vez que se inicia un producto de Altova en el equipo del cliente, LicenseServer comprueba si la dirección IP del cliente que le llega coincide con el nombre de host o la dirección IP de algún cliente al que se le haya asignado una licencia. Si es así, los detalles de la licencia quedan verificados.

 © 2018 Altova GmbH

Conexiones VPN con direcciones IP dinámicas

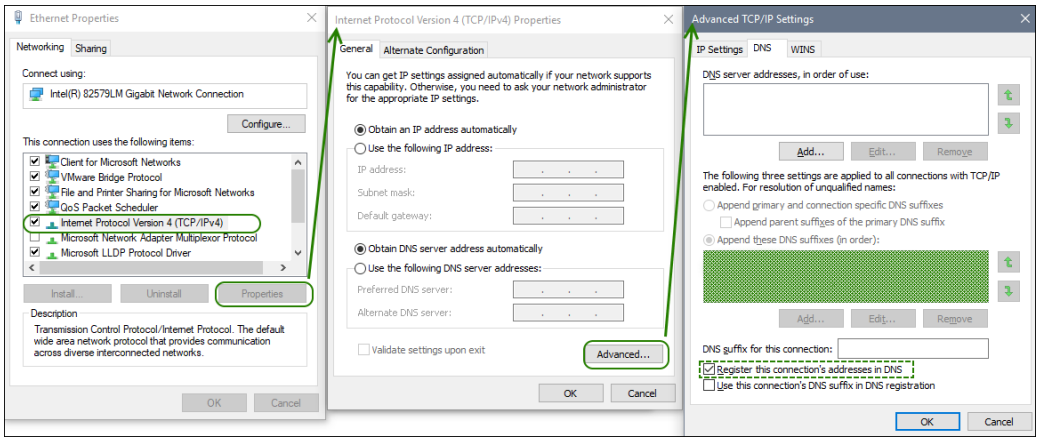
Si un equipo cliente se conecta a LicenseServer con una red privada virtual (VPN por sus siglas en inglés), la conexión a menudo se realiza con una dirección IP asignada de forma dinámica. En este caso, LicenseServer almacenará cada nueva conexión del cliente como proveniente de una dirección IP nueva, es decir, de un cliente antes deconocido (consulte el apartado anterior "Cómo identifica LicenseServer los equipos clientes").

Ello tiene como consecuencia que:

- Si **hay una licencia disponible** en LicenseServer en ese momento, se le asignará una nueva licencia al cliente (que puede que tenga ya una o más licencias asignadas de conexiones anteriores). Las licencias que se asignan a un cliente no se liberan, por lo que un solo cliente podría estar consumiendo múltiples licencias, lo que haría que escasearan las licencias en el servidor.
- Si **no hay licencias disponibles** en LicenseServer, no se le podrá asignar ninguna licencia a ese producto, incluso aunque se le hayan asignado licencias con anterioridad (para direcciones IP que el cliente ya no usa).

Hay dos maneras de resolver este problema:

- *Registrar a los clientes VPN con un DNS.* EL proceso para hacerlo varía en función del sistema operativo. En Windows 10, por ejemplo, esa configuración se encuentra en las propiedades de conexión de red del cliente (*imagen siguiente*).



- *Asignar una dirección IP estática al usuario de la VPN.* El proceso para hacerlo varía de un software de VPN a otro. Para saber más, consulte con su proveedor o administrador VPN. Por ejemplo, aquí se describe el proceso para OpenVPN.

11.2 Actualizar LicenseServer

Cada versión nueva de un producto servidor de Altova (p. ej. RaptorXML Server) viene acompañada de una nueva versión de Altova LicenseServer (la versión correspondiente). **Por tanto, si se pasa a una nueva versión de su producto servidor de Altova, deberá actualizar también su versión de LicenseServer** (es decir, deberá pasarse a la versión correspondiente de LicenseServer). Esto se deba a que la versión actualizada del producto servidor no podrá recibir licencias de una versión de LicenseServer anterior a la versión correspondiente. No obstante, debe tener en cuenta que las versiones de LicenseServer son compatibles con versiones anteriores (esto significa que LicenseServer puede utilizarse para asignar licencias a las versiones correspondientes de los productos servidor de Altova y a sus versiones más antiguas).

Para actualizar LicenseServer y pasarse a una versión más reciente:

- *En sistemas Windows*: haga doble clic en el programa de instalación de la nueva versión (en el archivo ejecutable) o llame al programa de instalación desde la línea de comandos. El programa de instalación desinstalará la versión actual de LicenseServer e instalará la versión nueva.
- *En sistemas Linux y macOS*: desinstale a mano la versión actual y después inicie el archivo de instalación de la versión nueva.

Además deberán cumplirse las mismas condiciones que cuando se instala LicenseServer por primera vez (véase apartado correspondiente para instalación en Windows, Linux o macOS).

 © 2018 Altova GmbH

11.3 Instalación en Windows

En los sistemas Windows puede instalar Altova LicenseServer de dos maneras:

- como producto servidor independiente.
- con el paquete de instalación de un producto servidor de Altova (Altova FlowForce Server, Altova MapForce Server, Altova StyleVision Server y Altova RaptorXML(+XBRL) y Altova MobileTogether Server).

Si LicenseServer no está instalado, la opción *Instalar LicenseServer* se activa por defecto durante el proceso de instalación del producto servidor de Altova. Si LicenseServer ya está instalado, la opción *Instalar LicenseServer* se desactiva por defecto. Puede activar o desactivar la opción si quiere.

Para aprender a asignar licencias con Altova LicenseServer consulte la sección Asignación de licencias.

Requisitos del sistema

- *Windows*

  Windows 7 SP1 con actualización de la plataforma, Windows 8, Windows 10

- *Windows Server*

  Windows Server 2008 R2 SP1 con actualización de la plataforma o superior

- Versiones de LicenseServer y compatibilidad con los productos de Altova

  Cuando actualice su producto de Altova a una versión más reciente, asegúrese de actualizar también su servidor LicenseServer con la versión más reciente. Las versiones anteriores de los productos de Altova funcionarán con las conversiones más recientes de LicenseServer.

  Es decir, si instala una versión nueva de un producto de Altova y la versión actual de su servidor LicenseServer no es la más reciente, desinstale la versión antigua de LicenseServer e instale la versión más reciente (disponible en el sitio web de Altova). Toda la información de registro y licencias almacenada en la versión antigua se guardará en una base de datos su equipo durante la desinstalación y se importará automáticamente a la versión nueva. Cuando instale una versión más reciente de LicenseServer, la versión más antigua se desinstalará antes de comenzar la instalación de la versión más reciente.

  El número de versión del servidor LicenseServer que está instalado aparece al final de la página de configuración de LicenseServer (en todas las pestañas).

  Versión actual: 2.7

El número de versión de LicenseServer correspondiente para la versión del producto servidor aparece durante el proceso de instalación del producto servidor. Puede instalar esta versión de

LicenseServer junto con el producto servidor o instalar una versión más reciente de LicenseServer por separado. En ambos casos el programa de instalación desinstalará la versión anterior (si todavía está instalada) e instalará la versión más reciente.

 © 2018 Altova GmbH

11.4 Instalación en Linux

Puede instalar LicenseServer en sistemas Linux (Debian, Ubuntu, CentOS y RedHat).

Requisitos del sistema

Linux

- CentOS 6 o superior
- RedHat 6 o superior
- Debian 7 o superior
- Ubuntu 12.04 o superior

Las bibliotecas que aparecen a continuación son un requisito previo para la instalación y ejecución de la aplicación. Si los paquetes que aparecen en esta tabla no están en su equipo Linux, ejecute el comando `yum` (o `apt-get` si procede) para instalarlos.

Requisito para	CentOS, RedHat	Debian	Ubuntu
LicenseServer	krb5-libs	libgssapi-krb5-2	libgssapi-krb5-2
RaptorXML+XBRL Server	qt4, krb5-libs, qt-x11	libqtcore4, libqtgui4, libgssapi-krb5-2	libqtcore4, libqtgui4, libgssapi-krb5-2

Desinstalar versiones anteriores de LicenseServer

En la interfaz de la línea de comandos de Linux, compruebe si LicenseServer está instalado ya. Para ello use este comando:

```
[Debian, Ubuntu]:       dpkg --list | grep Altova

[CentOS, RedHat]:       rpm -qa | grep server
```

Si LicenseServer no está instalado, instale el producto tal y como se explica más abajo. Si LicenseServer ya está instalado y desea instalar una versión más reciente, desinstale la versión antigua con este comando:

```
[Debian, Ubuntu]:           sudo dpkg --remove licenseserver

[CentOS, RedHat]:           sudo rpm -e licenseserver
```

Instalar Altova LicenseServer

En los sistemas Linux debe instalar LicenseServer por separado porque no viene incluido en los paquetes de instalación de los productos servidor de Altova. Descargue el instalador de Altova LicenseServer del sitio web de Altova y copie el paquete en cualquier directorio del sistema Linux.

Distribución	Extensión del instalador
Debian	.deb
Ubuntu	.deb
CentOS	.rpm
RedHat	.rpm

En una ventana de la Terminal, cambie al directorio donde copió el paquete de Linux. Por ejemplo, si lo copió en un directorio del usuario llamado MiAltova (en /home/User por ejemplo), cambie a ese directorio con esta línea de comandos:

```
cd /home/User/MiAltova
```

Instale LicenseServer como usuario root. Si no quiere iniciar sesión como root, puede usar el comando sudo para actuar temporalmente con privilegios de usuario root. Instale LicenseServer con este comando:

```
[Debian]:   sudo dpkg --install licenseserver-2.7-debian.deb

[Ubuntu]:   sudo dpkg --install licenseserver-2.7-ubuntu.deb

[CentOS, RedHat]:   sudo rpm -ivh licenseserver-2.7-1.x86_64.rpm
```

- El paquete de LicenseServer se instalará en la carpeta: **/opt/Altova/LicenseServer/bin**
- Se creará un usuario llamado **altovalicenseserver** con los derechos necesarios para ejecutar LicenseServer. Si empieza LicenseServer como demonio (o servicio), se iniciará automáticamente como este usuario. (Por supuesto, también puede (i) crear un usuario nuevo con privilegios que le permitan acceder a LicenseServer o (ii) iniciar LicenseServer como usuario root (que tiene el nivel máximo de privilegios), pero ninguno de estos pasos es necesario ni tampoco los recomendamos, ya que (i) ambos tienen sus desventajas y (ii) el usuario **altovalicenseserver** tiene tiene todos los derechos necesarios para ejecutar LicenseServer y se trata de un usuario independiente y específico.)

  Nota: también puede ejecutar LicenseServer como **altovalicenseserver** en modo interactivo. Por ejemplo, en CentOS 7 puede depurar con este comando: **sudo runuser -l altovalicenseserver -c '/opt/Altova/LicenseServer/bin/licenseserver debug'**. La desventaja del modo interactivo es que LicenseServer dejará de funcionar cuando se cierre la sesión de terminal.

Información contextual sobre la ejecución de LicenseServer

Tenga en cuenta que:

 © 2018 Altova GmbH

- recomendamos que ejecute LicenseServer como demonio (o servicio). Si lo ejecuta en modo interactivo dejará de funcionar cuando se cierre la sesión de terminal.
- Al ejecutar LicenseServer como demonio, lo mejor es gestionar LicenseServer con los comandos **`initctl`** o **`systemctl`**. Estos dos dos comandos solo se pueden ejecutar privilegios de usuario `root`, así que si está usando otro usuario debe usar el comando `sudo` para obtener temporalmente privilegios de usuario `root` (**`sudo initctl...`** y **`sudo systemctl...`**).
- Una vez haya iniciado LicenseServer como demonio, este se ejecutará automáticamente como usuario `altovalicenseserver`, que (i) creó el instalador de LicenseServer durante la instalación y (ii) tiene todos los derechos necesarios para ejecutar LicenseServer. Para más información, consulte el apartado Iniciar LicenseServer.

Se creará un usuario llamado **`altovalicenseserver`**. **Solamente podrá acceder a LicenseServer como este usuario**. Por ejemplo, en CentOS 7, con este comando:

```
sudo runuser -l altovalicenseserver -c '/opt/Altova/LicenseServer/bin/
licenseserver debug'
```

Para aprender a asignar licencias con Altova LicenseServer consulte la sección Asignación de licencias.

Versiones de LicenseServer y compatibilidad con los productos de Altova

Cuando actualice su producto de Altova a una versión más reciente, asegúrese de actualizar también su servidor LicenseServer con la versión más reciente. Las versiones anteriores de los productos de Altova funcionarán con las conversiones más recientes de LicenseServer.

Es decir, si instala una versión nueva de un producto de Altova y la versión actual de su servidor LicenseServer no es la más reciente, desinstale la versión antigua de LicenseServer e instale la versión más reciente (disponible en el sitio web de Altova). Toda la información de registro y licencias almacenada en la versión antigua se guardará en una base de datos su equipo durante la desinstalación y se importará automáticamente a la versión nueva. Cuando instale una versión más reciente de LicenseServer, la versión más antigua se desinstalará antes de comenzar la instalación de la versión más reciente.

El número de versión del servidor LicenseServer que está instalado aparece al final de la página de configuración de LicenseServer (en todas las pestañas).

Versión actual: 2.7

11.5 Instalación en macOS

Puede instalar Altova LicenseServer en sistemas (Mac) OS X, macOS (*ver requisitos del sistema más abajo*). En este apartado describimos primero el proceso de desinstalación, por si necesita desinstalar una versión previa de LicenseServer.

Requisitos del sistema

- (*Mac) OS X, macOS*

  OS X 10.10 o superior

Desinstalar versiones previas de LicenseServer

Antes de desinstalar LicenseServer debe detener el servicio con este comando:

```
sudo launchctl unload /Library/LaunchDaemons/com.altova.LicenseServer.plist
```

Para comprobar si el servicio se detuvo o no, abra la terminal del Monitor de actividad y confirme que LicenseServer no está en la lista.

En **Aplicaciones** haga clic con el botón derecho en el icono de LicenseServer y seleccione **Mover a la papelera**. La aplicación se envía a la papelera pero debe quitar la aplicación de la carpeta `usr`. Para ello puede utilizar este comando:

```
sudo rm -rf /usr/local/Altova/LicenseServer
```

Instalar Altova LicenseServer

Abra la página del centro de descargas de Altova (http://www.altova.com/es/download.html) y busque el programa de instalación de Altova LicenseServer para Mac en la sección ***Productos de software servidor***. Después de descargarlo, abra el archivo `.dmg`, que monta una unidad virtual nueva en el equipo. En la unidad virtual haga doble clic en el archivo `.pkg` y siga las instrucciones que aparecen en pantalla. Para poder continuar con la instalación deberá hacer clic en **Aceptar** en el contrato de licencia.

El paquete de LicenseServer se instalará en esta carpeta:

```
/usr/local/Altova/LicenseServer
```

Se creará un usuario llamado `altovalicenseserver`. **Solamente podrá acceder a LicenseServer como este usuario**.

Una vez finalizada la instalación puede expulsar la unidad virtual (clic con el botón derecho y después clic en **Expulsar**).

 © 2018 Altova GmbH

Tenga en cuenta que:

- el paquete de LicenseServer se instalará en esta carpeta: **/usr/local/Altova/LicenseServer**
- se creará un usuario llamado **altovalicenseserver**, que tendrá los derechos necesarios para ejecutar LicenseServer. Si empieza LicenseServer como demonio (o servicio), se iniciará automáticamente como este usuario.

Información contextual sobre la ejecución de LicenseServer

Tenga en cuenta que:

- recomendamos que ejecute LicenseServer como demonio (o servicio). Si lo ejecuta en modo interactivo dejará de funcionar cuando se cierre la sesión de terminal.
- al ejecutar LicenseServer como demonio, lo mejor es gestionar LicenseServer con los comandos **initctl** o **systemctl**. Estos dos dos comandos solo se pueden ejecutar privilegios de usuario `root`, así que si está usando otro usuario debe usar el comando `sudo` para obtener temporalmente privilegios de usuario `root` (**sudo initctl...** y **sudo systemctl...**).
- una vez haya iniciado LicenseServer como demonio, este se ejecutará automáticamente como usuario `altovalicenseserver`, que (i) creó el instalador de LicenseServer durante la instalación y (ii) tiene todos los derechos necesarios para ejecutar LicenseServer. Para más información, consulte el apartado Iniciar LicenseServer.

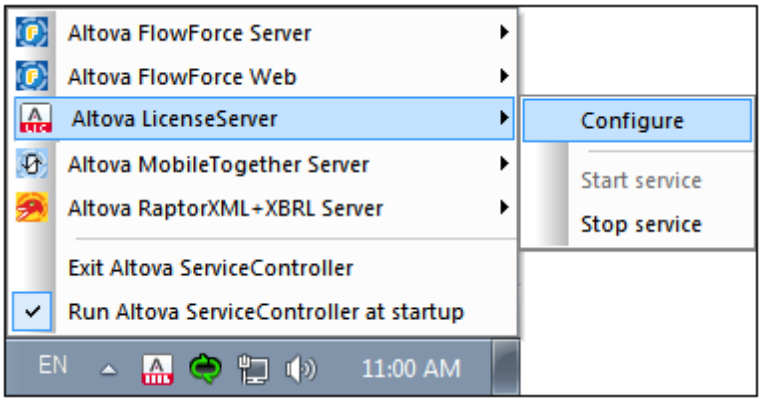
Para expulsar la unidad virtual después de la instalación, haga clic en ella con el botón derecho y seleccione **Expulsar**.

11.6 Altova ServiceController

Altova ServiceController (en adelante *ServiceController*) es una práctica aplicación que sirve para iniciar, detener y configurar los servicios de Altova **en sistemas Windows**.

ServiceController se instala con Altova LicenseServer y con *los productos servidor de Altova que se instalan como servicios* (FlowForce Server, RaptorXML(+XBRL) Server y Mobile Together Server). Se puede iniciar haciendo clic en **Inicio | Altova LicenseServer | Altova ServiceController**. (Este comando también está en las carpetas del menú **Inicio** de *los productos servidor de Altova que se instalan como servicios* (FlowForce Server, RaptorXML(+XBRL) Server y Mobile Together Server).) Una vez iniciado, podrá acceder a ServiceController desde la bandeja del sistema (*imagen siguiente*).

Si quiere que ServiceController se inicie automáticamente nada más iniciar sesión en el sistema, haga clic en el icono de ServiceController de la bandeja del sistema para abrir el menú de opciones de **ServiceController** (*imagen siguiente*) y active la opción **Run Altova ServiceController at Startup** (*Ejecutar Altova ServiceController al inicio*), que de todas maneras es la opción predeterminada. Para cerrar ServiceController haga clic en el icono de ServiceController de la bandeja del sistema y en el menú haga clic en la opción **Exit Altova ServiceController** (Salir de Altova ServiceController).



Iniciar y detener servicios de Altova

Todos los servicios de Altova instalados en el equipo aparecen en el menú contextual de ServiceController (*imagen anterior*). Los servicios de Altova se pueden iniciar o detener con los comandos del menú contextual de ServiceController. Este menú también incluye comandos de algunas tareas de administración. Por ejemplo, en la imagen anterior puede ver el submenú del servicio Altova LicenseServer, donde puede elegir la opción **Configure** para abrir la página de configuración de Altova LicenseServer.

 © 2018 Altova GmbH

11.7 Asignación de licencias

Siga estos pasos para asignar una licencia a un producto de Altova desde Altova LicenseServer:

1. Inicie LicenseServer
2. Abra la página de configuración de LicenseServer, es decir, la interfaz de administración de LicenseServer en Windows, Linux o macOS.
3. Cargue las licencias que Altova le envió. Esto se hace en la pestaña **License Pool** (Repertorio de licencias) de la página de configuración de LicenseServer.
4. Registre el producto de Altova con LicenseServer. El método de registro depende del tipo de producto: en el cuadro de diálogo "Activación del software" del producto, en una interfaz web o desde la línea de comandos. Consulte la documentación del producto de Altova para obtener más información.
5. En la pestaña de gestión de clientes **Client Management** de la página de configuración de LicenseServer, asigne la licencia al producto de Altova.

Nota sobre núcleos y licencias
La asignación de licencias a productos servidor de Altova depende de cuántos núcleos de procesador tiene el equipo donde se ejecuta el producto servidor de Altova. Por ejemplo, un procesador dual tiene dos núcleos, un procesador *quad* tiene cuatro núcleos, un procesador *hexa-core* tiene seis núcleos, y así sucesivamente. El número de núcleos de la licencia asignada a un producto debe ser mayor o igual al número de núcleos disponibles en dicho equipo servidor, ya sea un servidor físico o un equipo virtual.

Por ejemplo, si un servidor tiene ocho núcleos (un procesador *octa-core*), deberá comprar una licencia para ocho núcleos. También puede combinar varias licencias para alcanzar el número de núcleos necesario. Es decir, puede usar dos licencias para cuatro núcleos para un servidor *octa-core* en lugar de una licencia para ocho núcleos, por ejemplo.

Si usa un equipo servidor con gran cantidad de núcleos, pero tiene un bajo volumen de procesamiento, también puede crear un equipo virtual que tenga adjudicados menos núcleos y comprar una licencia para ese menor número de núcleos. No obstante, dicha implementación será menos rápida que si utilizara todos los núcleos disponibles en el servidor.

Nota: cada licencia de los productos servidor de Altova se puede usar de forma simultánea en un equipo como máximo (en el equipo donde está instalado el producto servidor de Altova), incluso si la capacidad de la licencia no está agotada. Por ejemplo, si utiliza una licencia para 10 núcleos para un equipo cliente que tiene 6 núcleos, los 4 núcleos restantes de la licencia no se pueden usar simultáneamente en otro equipo cliente.

Licencias de MobileTogether Server
Las licencias de MobileTogether Server funciona en base al número de núcleos CPU que tenga el equipo donde se ejecuta MobileTogether Server. Las licencias basadas en el número de núcleos permiten conectar un número ilimitado de dispositivos al servidor. Sin embargo, si marca la casilla *Limit to single thread execution*, entonces solo se podrá conectar a MobileTogether Server un máximo de un dispositivo. Esto puede ser útil en tareas de evaluación y de pruebas a pequeña escala. No obstante, si estando marcada esta casilla, se conecta otro dispositivo a

MobileTogether Server, este otro dispositivo se hará con la licencia. El primer dispositivo ya no se podrá conectar y recibirá un mensaje de error a tal efecto.

11.7.1 Iniciar LicenseServer

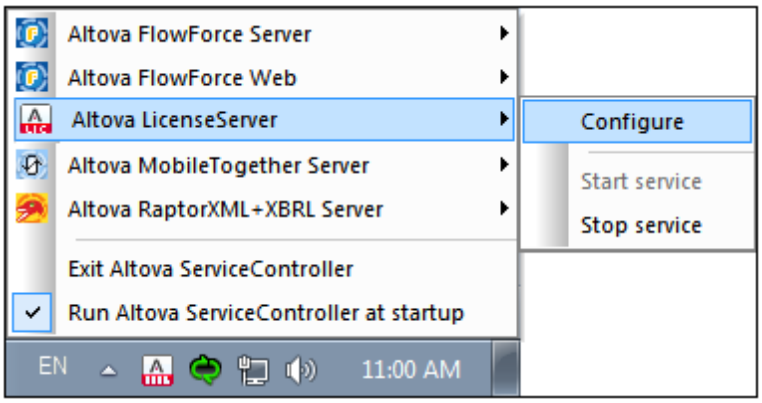
Este apartado explica:

- Cómo iniciar LicenseServer en sistemas Windows
- Cómo iniciar LicenseServer en sistemas Linux
- Cómo iniciar LicenseServer en sistemas macOS
- La conexión con el servidor maestro altova.com

Sistemas Windows

Puede iniciar LicenseServer con ayuda de ServiceController, disponible en la bandeja del sistema.

Primero haga clic en **Inicio | Todos los programas | Altova LicenseServer | Altova ServiceController** para iniciar Altova ServiceController, que a partir de ese momento estará disponible en la bandeja del sistema (*imagen siguiente*). Si selecciona la opción **Run Altova ServiceController at Startup**, Altova ServiceController se iniciará cada vez que se inicie sesión en el sistema y su icono aparecerá en la bandeja del sistema.



Para iniciar LicenseServer, haga clic en el icono de ServiceController de la bandeja del sistema. En el menú emergente haga clic en **Altova LicenseServer** (*imagen anterior*) y seleccione **Start Service** para iniciar el servicio. Si LicenseServer ya está en ejecución, la opción **Start Service** se deshabilita (*imagen anterior*).

Sistemas Linux

Para iniciar LicenseServer como servicio en sistemas Linux, ejecute este comando en una ventana de la Terminal.

```
[Debian 7]:       sudo /etc/init.d/licenseserver start
[Debian >=8]:     sudo systemctl start licenseserver
```

 © 2018 Altova GmbH

```
[Ubuntu <=14]:    sudo initctl start licenseserver
[Ubuntu >=15]:    sudo systemctl start licenseserver
[CentOS 6]:       sudo initctl start licenseserver
[CentOS >=7]:     sudo systemctl start licenseserver
[RedHat]:         sudo initctl start licenseserver
```

(Si necesita detener LicenseServer, reemplace `start` por `stop` en el comando anterior.)

Para más información, consulte Instalación en Linux.

Sistemas macOS

Para iniciar LicenseServer como servicio en sistemas macOS, ejecute este comando en una ventana de la Terminal:

```
sudo launchctl load /Library/LaunchDaemons/com.altova.LicenseServer.plist
```

Si necesita detener LicenseServer por cualquier motivo, use este comando:

```
sudo launchctl unload /Library/LaunchDaemons/com.altova.LicenseServer.plist
```

Para más información, consulte Instalación en macOS.

Conexión con el servidor maestro de licencias altova.com

Altova LicenseServer necesita comunicarse con el servidor maestro de licencias `altova.com` para validar y verificar los datos de licencia y para asegurarse de que los contratos de licencia de Altova se cumplen en todo momento. Esta comunicación se establece por HTTPS a través del puerto 443. Si tras completar la verificación inicial con el servidor maestro `altova.com`, Altova LicenseServer no puede volver a conectarse con `altova.com` durante más de 5 días (120 horas), se bloqueará el uso de los productos de Altova que estén conectados con Altova LicenseServer.

Cuando se pierde la conexión con el servidor maestro `altova.com`, la incidencia se registra en la pestaña **Messages** de la página de configuración de Altova LicenseServer. Además el administrador puede configurar Altova LicenseServer para que le envíe correos de alerta cuando se pierda la conexión con `altova.com`. Las opciones de configuración de estos correos de alerta se pueden definir en la pestaña **Settings** de la página de configuración.

11.7.2 Abrir la página de configuración de LicenseServer (Windows)

Este apartado explica:

- Cómo abrir la página de configuración cuando LicenseServer está ejecutándose en el mismo equipo
- Cómo abrir la página de configuración cuando LicenseServer está ejecutándose en otro equipo

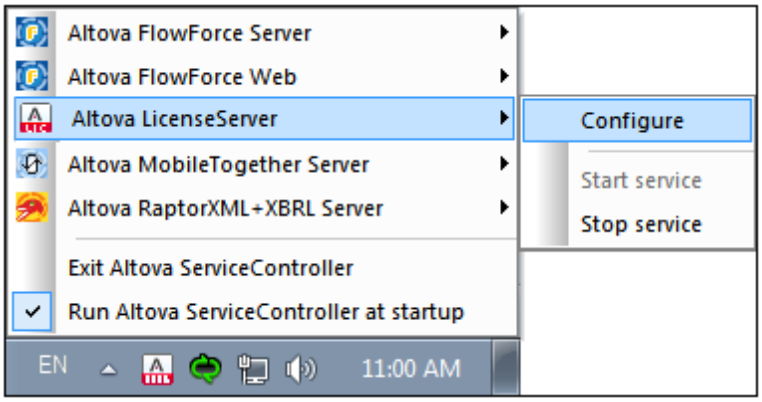
- Cómo iniciar sesión con la contraseña inicial
- Cómo establecer un puerto fijo para la página de configuración

Abrir la página de configuración cuando LicenseServer está ejecutándose en el mismo equipo

En los sistemas Windows, si el servidor LicenseServer está ejecutándose en el mismo equipo, puede abrir la página de configuración de LicenseServer de dos maneras:

- Haga clic en **Inicio | Todos los programas | Altova LicenseServer | Página de configuración de LicenseServer**. La página de configuración se abre en una pestaña nueva del explorador web.

- Haga clic en el icono de Altova ServiceController de la bandeja del sistema. En el menú emergente haga clic en **Altova LicenseServer** y seleccione **Configure**.



La página de configuración se abre en una ventana del explorador nueva por la página de inicio de sesión (*imagen siguiente*).

Abrir la página de configuración cuando LicenseServer está ejecutándose en otro equipo

Para abrir la página de configuración de LicenseServer desde otro equipo Windows de la red local (cuando LicenseServer no está instalado en el equipo local), introduzca la URL de la página de configuración de LicenseServer en la barra de dirección del explorador web y pulse **Entrar**. La URL predeterminada de la página de configuración es:

```
http://<DirecciónIPoNombreDelServidor>:8088/
```

La URL de la página de configuración está presente en el código HTML de la página de configuración propiamente dicha, que se llama `WebUI.html` y está en este directorio:

```
C:/ProgramData/Altova/LicenseServer/WebUI.html
```

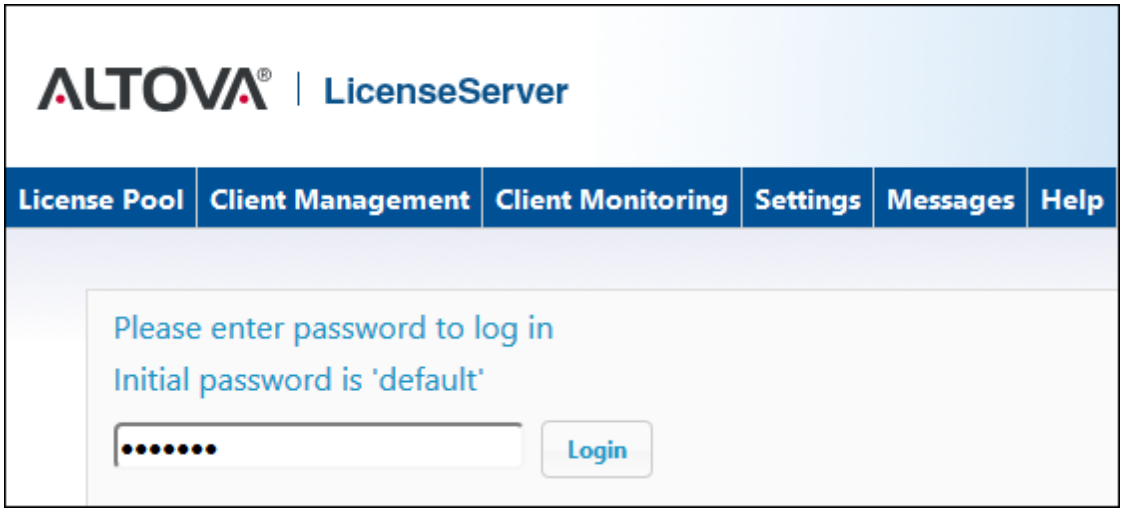
Si eligió que la URL de la página de configuración se genere de forma dinámica (en la pestaña **Configuración** de la página de configuración), la URL nueva se genera cada vez que se inicia LicenseServer. Para averiguar la URL actual de la página de configuración necesitará ver la

versión actual de **`WebUI.html`**.

El formato de la URL generada dinámicamente en **`WebUI.html`** será similar a `http://127.0.0.1:55541/con-una-cadena-adicional-si-quiere` y se encuentra dentro de la función `checkIfServiceRunning()` en un script situado cerca del final del elemento `<head>`. El número de puerto de la URL se asigna dinámicamente y la parte de la dirección IP identifica el servidor en el que está instalado LicenseServer. Si quiere acceder a la página de configuración de LicenseServer desde otro equipo, compruebe que la dirección IP de la URL es correcta o incluye el nombre del servidor en el que está instalado LicenseServer. Por ejemplo: `http://UnServidor:55541`.

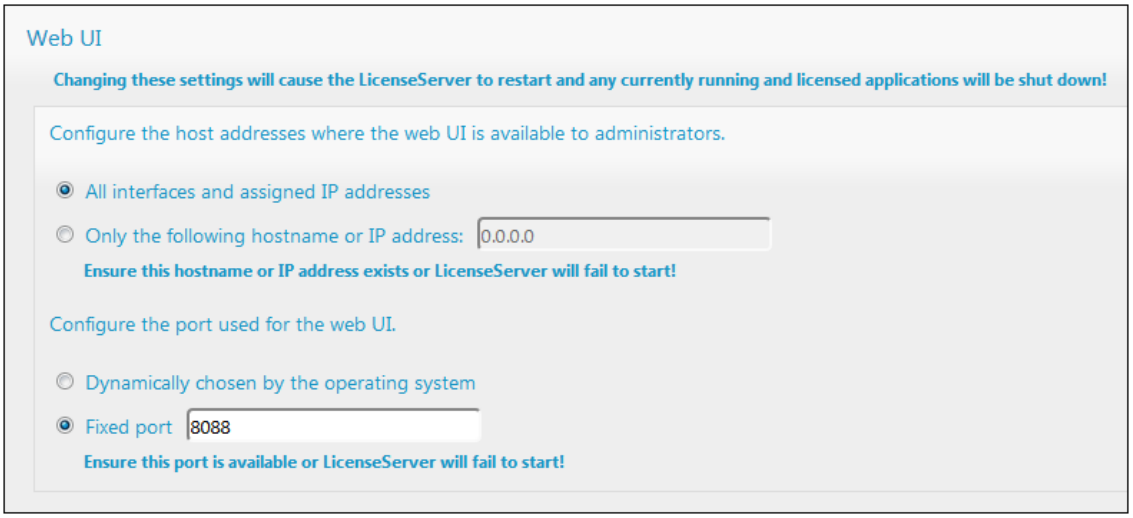
Iniciar sesión con la contraseña inicial

Después de seguir todas estas instrucciones se abre la página de configuración por la página de inicio de sesión (*imagen siguiente*). Puede iniciar sesión con la contraseña predeterminada `default`. Después de iniciar sesión puede cambiar la contraseña en la pestaña **Settings**.



Establecer un puerto fijo o dinámico para la página de configuración

El puerto de la página de configuración (y, por consiguiente, su dirección) se puede especificar en la pestaña **Settings**. El puerto predeterminado es `8088` pero puede definir cualquier puerto para la página de configuración de LicenseServer (*imagen siguiente*). También puede definir que el puerto se seleccione de forma dinámica cada vez que se inicie LicenseServer. En este caso, necesitará buscar la URL de la página de configuración en el archivo `WebUI.html` (consulte los apartados Abrir la página de configuración de LicenseServer (Windows), Abrir la página de configuración de LicenseServer (Linux) y Abrir la página de configuración de LicenseServer (macOS)).



La ventaja de usar un puerto fijo es que la URL de la página se conoce por adelantado y se puede abrir con más facilidad. Si el puerto se asigna de forma dinámica, la parte del puerto de la URL debe buscarse en el archivo `WebUI.html` cada vez que se inicia LicenseServer.

11.7.3 Abrir la página de configuración de LicenseServer (Linux)

Este apartado explica:

- Cómo abrir por primera vez la página de configuración usando la URL obtenida
- La URL de la página de configuración de LicenseServer
- Cómo iniciar sesión con la contraseña inicial
- Cómo establecer un puerto fijo para la página de configuración

Abrir por primera vez la página de configuración usando la URL obtenida

Cuando se registra el producto servidor de Altova con LicenseServer por la interfaz de la línea de comandos en sistemas Linux, se devuelve la URL de la página de configuración de LicenseServer. Al abrir esta URL en el explorador web, se le solicita que lea y acepte el contrato de licencia. Después de aceptarlo, aparece la página de inicio de sesión de la página de configuración (*imagen siguiente*).

Nota: los productos de Altova solamente son compatibles con Windows.

URL de la página de configuración de LicenseServer

Después, cuando quiera abrir la página de configuración, escriba la URL en la barra de dirección del explorador web y pulse **Entrar**. La URL predeterminada de la página de configuración es:

```
http://<DirecciónIPoNombreDelServidor>:8088/
```

La URL de la página de configuración está presente en el código HTML de la página de configuración propiamente dicha, que se llama **`WebUI.html`** y está en este directorio:

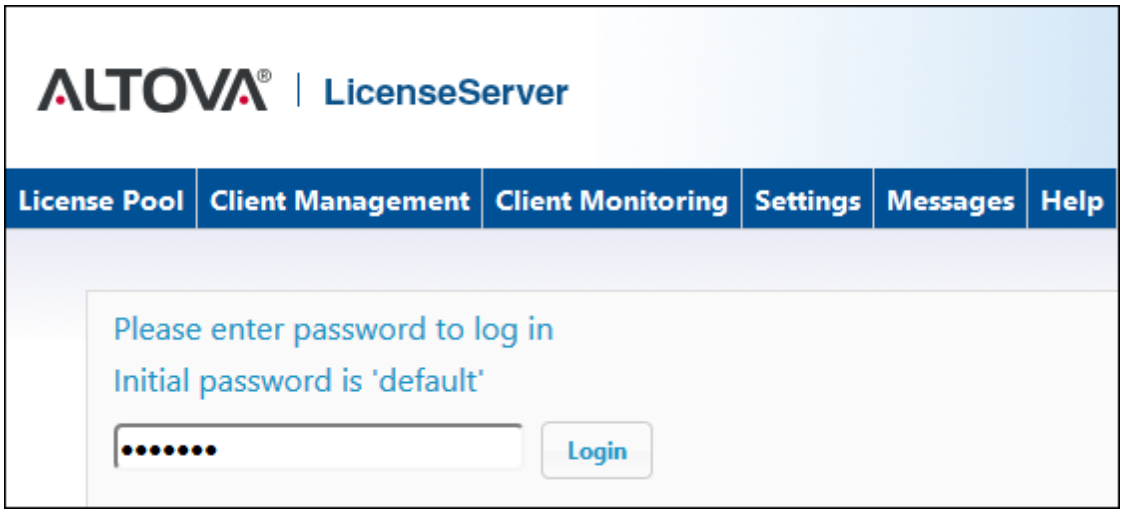
```
/var/opt/Altova/LicenseServer/webUI.html
```

Si eligió que la URL de la página de configuración se genere de forma dinámica (en la pestaña **Configuración** de la página de configuración), se genera una URL nueva cada vez que se inicia LicenseServer. Para averiguar la URL actual de la página de configuración necesitará mirar la versión actual de **`WebUI.html`**.

El formato de la URL generada dinámicamente en **`WebUI.html`** será similar a `http://127.0.0.1:55541` y se encuentra dentro de la función `checkIfServiceRunning()` en un script situado cerca del final del elemento `<head>`. El número de puerto de la URL se asigna dinámicamente y la parte de la dirección IP identifica el servidor en el que está instalado LicenseServer. Si quiere acceder a la página de configuración de LicenseServer desde otro equipo, compruebe que la dirección IP de la URL es correcta o incluye el nombre del servidor en el que está instalado LicenseServer. Por ejemplo: `http://MiServidor:55541`.

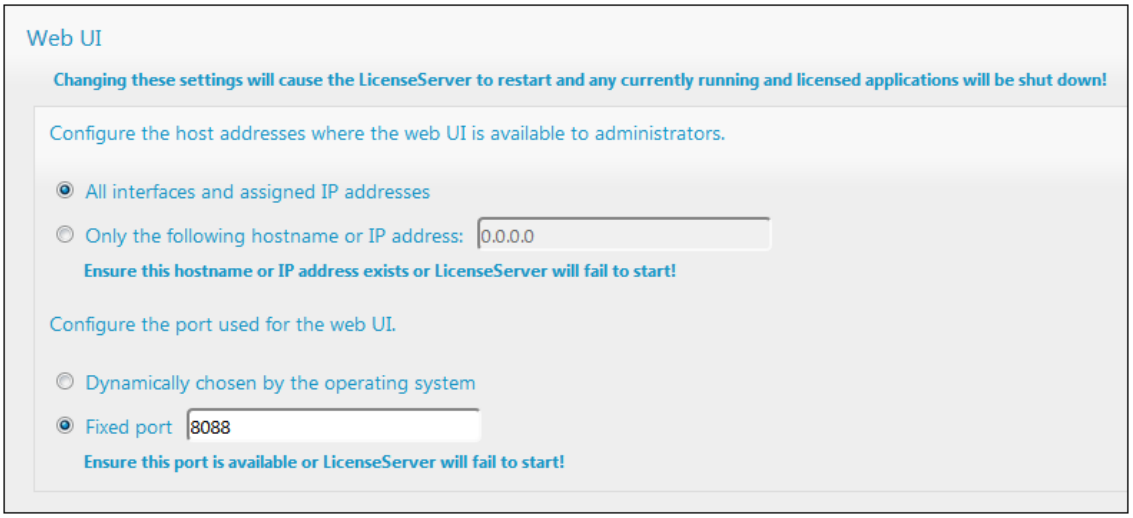
Iniciar sesión con la contraseña inicial

Después de seguir todas estas instrucciones se abre la página de configuración por la página de inicio de sesión (*imagen siguiente*). Puede iniciar sesión con la contraseña predeterminada `default`. Después de iniciar sesión puede cambiar la contraseña en la pestaña **Settings**.



Establecer un puerto fijo o dinámico para la página de configuración

El puerto de la página de configuración (y, por consiguiente, su dirección) se puede especificar en la pestaña **Settings**. El puerto predeterminado es `8088` pero puede definir cualquier puerto para la página de configuración de LicenseServer (*imagen siguiente*). También puede definir que el puerto se seleccione de forma dinámica cada vez que se inicie LicenseServer. En este caso, necesitará buscar la URL de la página de configuración en el archivo `WebUI.html` (consulte los apartados Abrir la página de configuración de LicenseServer (Windows), Abrir la página de configuración de LicenseServer (Linux) y Abrir la página de configuración de LicenseServer (macOS)).



La ventaja de usar un puerto fijo es que la URL de la página se conoce por adelantado y se puede abrir con más facilidad. Si el puerto se asigna de forma dinámica, la parte del puerto de la URL debe buscarse en el archivo `WebUI.html` cada vez que se inicia LicenseServer.

11.7.4 Abrir la página de configuración de LicenseServer (macOS)

Este apartado explica:

- Cómo abrir la página de configuración por primera vez usando la URL obtenida
- La URL de la página de configuración de LicenseServer
- Cómo iniciar sesión con la contraseña inicial
- Cómo establecer un puerto fijo para la página de configuración

Abrir por primera vez la página de configuración usando la URL obtenida

Cuando se registra el producto servidor de Altova con LicenseServer por la interfaz de la línea de comandos en sistemas macOS, se devuelve la URL de la página de configuración de

LicenseServer. Al abrir esta URL en el explorador web, se le solicita que lea y acepte el contrato de licencia. Después de aceptarlo, aparece la página de inicio de sesión de la página de configuración (*imagen siguiente*).

Nota: los productos de Altova solamente son compatibles con Windows.

URL de la página de configuración de LicenseServer

Después, cuando quiera abrir la página de configuración, escriba la URL en la barra de dirección del explorador web y pulse **Entrar**. La URL predeterminada de la página de configuración es:

```
http://<DirecciónIPoNombreDelServidor>:8088/
```

La URL está presente en el código HTML de la página de configuración propiamente dicha, que se llama **`WebUI.html`** y está en este directorio:

```
/var/Altova/LicenseServer/webUI.html
```

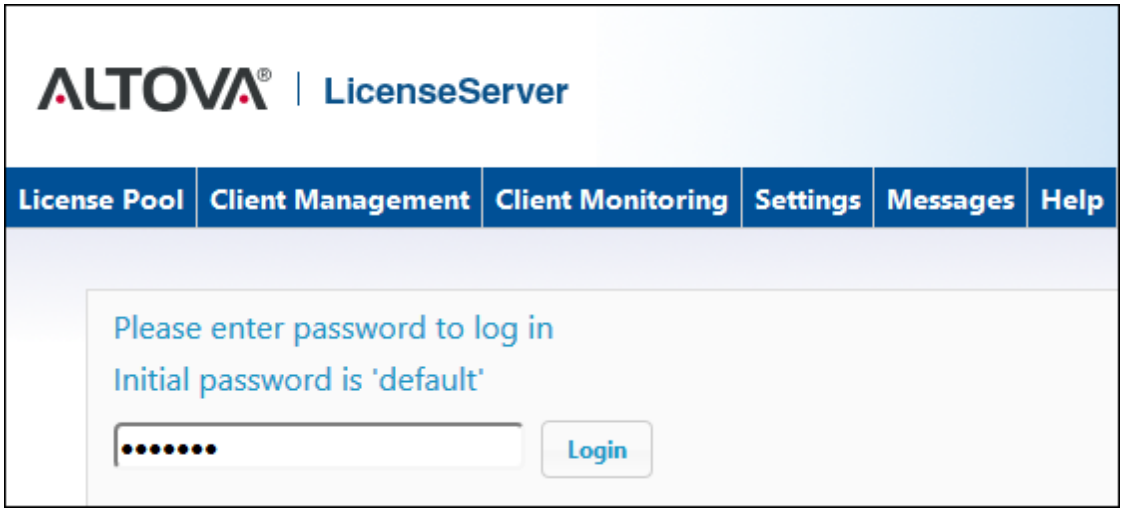
Si eligió que la URL de la página de configuración se genere de forma dinámica (en la pestaña **Configuración** de la página de configuración), se genera una URL nueva cada vez que se inicia LicenseServer. Para averiguar la URL actual de la página de configuración necesitará mirar la versión actual de **`WebUI.html`**.

El formato de la URL generada dinámicamente en **`WebUI.html`** será similar a `http://127.0.0.1:55541` y se encuentra dentro de la función `checkIfServiceRunning()` en un script situado cerca del final del elemento `<head>`. El número de puerto de la URL se asigna dinámicamente y la parte de la dirección IP identifica el servidor en el que está instalado LicenseServer. Si quiere acceder a la página de configuración de LicenseServer desde otro equipo, compruebe que la dirección IP de la URL es correcta o incluye el nombre del servidor en el que está instalado LicenseServer. Por ejemplo: `http://MiServidor:55541`.

Nota: la página de configuración también se puede abrir directamente desde el icono **Finder | Aplicaciones | Altova License Server**.

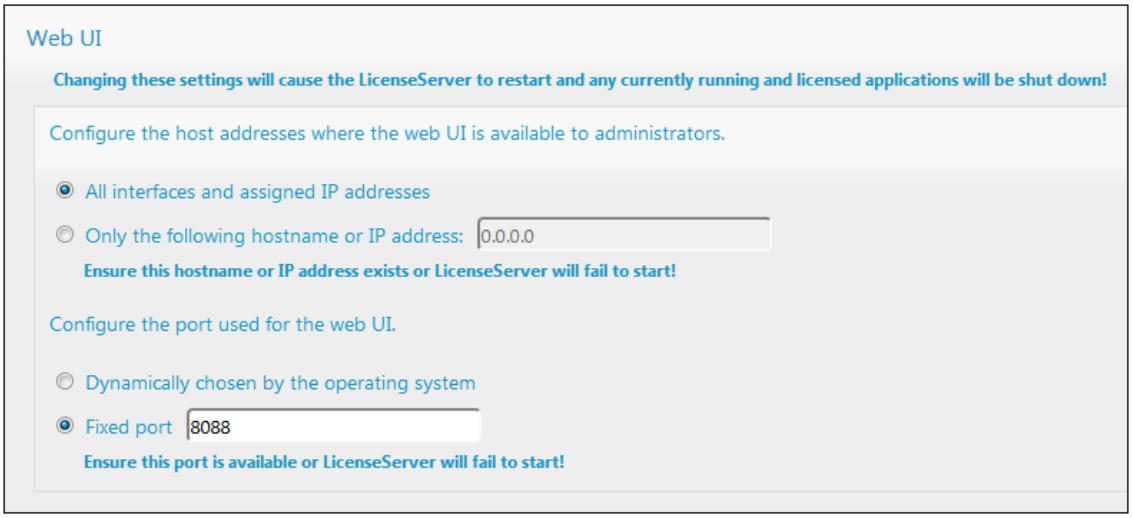
Iniciar sesión con la contraseña inicial

Después de seguir todas estas instrucciones se abre la página de configuración por la página de inicio de sesión (*imagen siguiente*). Puede iniciar sesión con la contraseña predeterminada `default`. Después de iniciar sesión puede cambiar la contraseña en la pestaña **Settings**.



Establecer un puerto fijo o dinámico para la página de configuración

El puerto de la página de configuración (y, por consiguiente, su dirección) se puede especificar en la pestaña **Settings**. El puerto predeterminado es `8088` pero puede definir cualquier puerto para la página de configuración de LicenseServer (*imagen siguiente*). También puede definir que el puerto se seleccione de forma dinámica cada vez que se inicie LicenseServer. En este caso, necesitará buscar la URL de la página de configuración en el archivo `WebUI.html` (consulte los apartados Abrir la página de configuración de LicenseServer (Windows), Abrir la página de configuración de LicenseServer (Linux) y Abrir la página de configuración de LicenseServer (macOS)).



La ventaja de usar un puerto fijo es que la URL de la página se conoce por adelantado y se puede abrir con más facilidad. Si el puerto se asigna de forma dinámica, la parte del puerto de la URL debe buscarse en el archivo `WebUI.html` cada vez que se inicia LicenseServer.

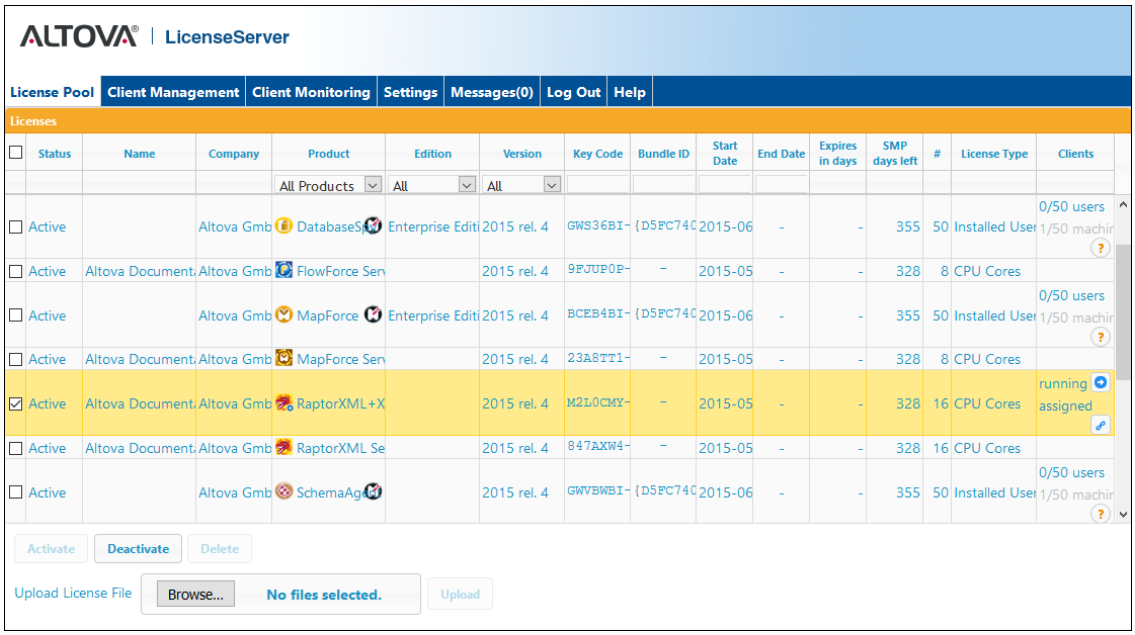
11.7.5 Cargar licencias en LicenseServer

Este apartado explica:

- Cómo cargar un archivo de licencias al repertorio de licencias de LicenseServer
- El estado de las licencias
- Cómo activar las licencias que desea usar
- Pasos siguientes

Cargar un archivo de licencias al repertorio de licencias de LicenseServer

Cuando reciba el archivo de licencias de Altova (un archivo con la extensión `.altova_licenses`), guárdelo en su equipo y cárguelo al repertorio de licencias de Altova LicenseServer. Un archivo de licencias puede contener varias licencias. Cuando cargue el archivo a LicenseServer (uno cuya extensión sea `.altova_licenses`), todas las licencias que contengan se cargarán al repertorio de licencias de LicenseServer. Después podrá asignarlas a los productos de Altova que estén registrados con el servidor LicenseServer. Las licencias se almacenan en LicenseServer en un repertorio de licencias. Para acceder al repertorio de licencias abra la pestaña **License Pool** de la página de configuración (*imagen siguiente*).



Los archivos de licencias se cargan a LicenseServer con la función *Upload License File* del repertorio de licencias (*imagen anterior*).

Haga clic en el botón Browse y seleccione el archivo de licencia que desea cargar. (El archivo que debe cargar es el que recibió como adjunto con el correo electrónico Licencias de Altova; este archivo tiene la extensión `.altova_licenses`.) Verá que el archivo de licencias aparece en el campo de texto *Upload License File* y que se habilita el botón **Upload**. Haga clic en este botón para cargar todas las licencias al repertorio. En la imagen anterior, por ejemplo, puede ver varias licencias que se cargaron desde archivos diferentes.

Estado de las licencias

Las licencias pueden tener varios estados:

- ***Activating*** *(En activación):* cuando se carga una licencia al repertorio de licencias de LicenseServer, el servidor transmite datos sobre la licencia al servidor maestro `altova.com` para validarla, verificarla y activarla. Este paso es necesario para comprobar que se cumple el contrato de licencia de Altova. Mientras se completa esta transacción (que dura entre 30 segundos y varios minutos, dependiendo de la velocidad y del tráfico de la conexión) el estado de la licencia es ***Activating...*** (*En activación...*).

- ***Failed verification*** *(No se puede verificar):* si no se puede establecer la conexión con el servidor maestro `altova.com`, el estado de la licencia es ***Failed verification*** *(No se puede verificar).* Si esto ocurre, revise su conexión a Internet y las reglas de su servidor de seguridad y compruebe que LicenseServer puede comunicarse con el servidor maestro `altova.com`.

- ***Active*** *(Activa):* la licencia se activa tras ser verificada.

- ***Inactive*** *(No activa):* si la licencia se verifica pero está presente en otro LicenseServer de la red, su estado es ***Inactive*** *(No activa)*. Esto también puede ocurrir cuando el administrador desactiva la licencia a mano en el repertorio de licencias.

- ***Pending*** *(Pendiente):* este es el estado de la licencia si su fecha de inicio es una fecha futura. El día de la fecha de inicio el estado pasará a ser ***Active*** a las 00.00 horas de la fecha de inicio. Puede asignar licencias con estado ***Pending*** a un producto y así asegurarse de que siga funcionando cuando la licencia actual expire. La transición a la nueva licencia está diseñado para que sea lo menos problemático posible y no interrumpe ningún proceso que pueda estar en ejecución en el cliente.

- ***Blocked*** *(Bloqueada):* la licencia tiene este estado cuando se produce un error al verificar la licencia y el servidor maestro `altova.com` no concede permiso a LicenseServer para usar la licencia. Esto puede deberse a una infracción del contrato de licencia o a un uso excesivo de una licencia, por ejemplo. Tras solucionar el problema, elimine la licencia, vuelva a cargarla y vuelva a activarla. Consulte la tabla que aparece a continuación para obtener más información.

A continuación resumimos la información anterior:

Estado	Significado
Activating *(En activación)*	Al cargar un archivo de licencias, los datos se envían a `altova.com` para verificar las licencias. Actualice el explorador para ver el estado de las licencias. El proceso de verificación y activación puede durar unos minutos.
Failed verification *(No se puede verificar)*	No se puede establecer la conexión con `altova.com`. Cuando consiga establecer la conexión, vuelve a iniciar LicenseServer o intente activar la licencia otra vez (con el botón **Activate**).
Active *(Activa)*	La licencia se verificó correctamente y la licencia está activa.
Inactive *(No activa)*	La licencia se verificó correctamente pero la licencia también está en otro servidor LicenseServer de la red. Las licencias se pueden desactivar con el botón **Deactivate**.
Pending *(Pendiente)*	Este es el estado de la licencia si su fecha de inicio es una fecha futura. En la fecha de inicio el estado pasa a ***Active***. Puede asignarse a un producto para garantizar la renovación automática de la licencia antes de expire la licencia actual.
Blocked *(Bloqueada)*	No se puede verificar la licencia. La licencia no es válida y se bloqueó. Una vez resuelto el problema que tenía bloqueada la licencia, elimine la licencia, vuelva a cargarla y vuelva a activarla. Cada vez que se carga una licencia, el servidor se pone en contacto con el servidor maestro de Altova, que iniciará

	una nueva verificación. Si no vuelve a cargar la licencia, la verificación deberá esperar hasta que se produzca la siguiente comunicación con el servidor maestro de Altova, que podría estar programada el día siguiente. Póngase en contacto con el equipo de soporte técnico de Altova si el problema continúa.

Nota: tras enviar los datos de licencia a `altova.com` es necesario actualizar la ventana del explorador para actualizar el estado. La verificación y activación puede tardar unos minutos.

Nota: si no se puede establecer la conexión con `altova.com`, el estado de la licencia es ***Failed verification*** *(No se puede verificar)*. Cuando consiga establecer la conexión, vuelve a iniciar LicenseServer o intente activar la licencia otra vez (con el botón **Activar**).

Nota: cuando la licencia tiene el estado ***Inactive*** *(No activa)* o ***Blocked*** *(Bloqueada)* aparece un mensaje explicando el motivo en la pestaña **Messages**.

A los productos solamente se les pueden asignar licencias activas o pendientes. Las licencias activas se pueden activar o eliminar del repertorio de licencias. Si una licencia se elimina del repertorio, se puede cargar otra vez cargando el archivo de licencias correspondiente. Cuando se actualiza un archivo de licencias, solamente se cargan al repertorio las licencias que todavía no están en él. Para activar, desactivar o eliminar una licencia, selecciónela y haga clic en el botón **Activate**, **Deactivate** o **Delete** respectivamente.

Activar las licencias que desea usar

Antes de asignar una licencia a un producto de Altova es necesario activarla. Si está inactiva, selecciónela y haga clic en el botón **Activate**.

¿Y ahora?

Tras cargar el archivo de licencias al repertorio de LicenseServer y comprobar que las licencias que quiere usar están activas:

1. Registre el producto servidor de Altova (FlowForce Server, MapForce Server, StyleVision Server) con LicenseServer. Si lo hizo antes de cargar las licencias, ya puede empezar a asignar las licencias.
2. Asigne una licencia al producto de Altova que está registrado con LicenseServer.

11.7.6 Registrar productos con LicenseServer

Para poder asignar una licencia a un producto de Altova antes es necesario registrar el producto con el servidor LicenseServer. El proceso de registro se lleva a cabo desde el producto de Altova y el proceso de registro dependerá del tipo de producto.

- *Productos de escritorio:* el registro se hace desde el cuadro de diálogo "Activación del software" del producto.

 © 2018 Altova GmbH

- *Productos servidor que cuentan con una interfaz web:* el registro de FlowForce Server y MobileTogether Server se hace desde la pestaña **Configuración** de la interfaz web o desde la línea de comandos.
- *Productos servidor **sin** interfaz web:* el registro de MapForce Server, RaptorXML(+XBRL) Server y StyleVision Server se hace desde la línea de comandos. Necesitará conocer el nombre de servidor o la dirección IP del equipo donde está instalado LicenseServer.

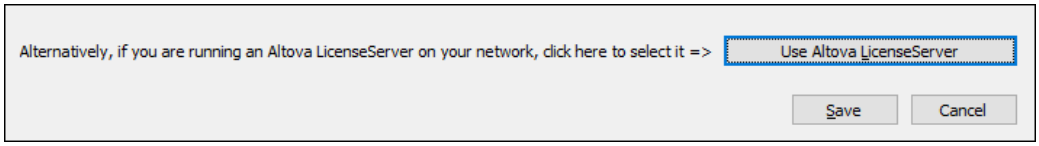
En esta sección explicamos cómo registrar los productos de Altova con LicenseServer:

- Registrar productos de escritorio de Altova
- Registrar FlowForce Server
- Registrar MapForce Server
- Registrar MobileTogether Server
- Registrar RaptorXML(+XBRL) Server
- Registrar StyleVision Server

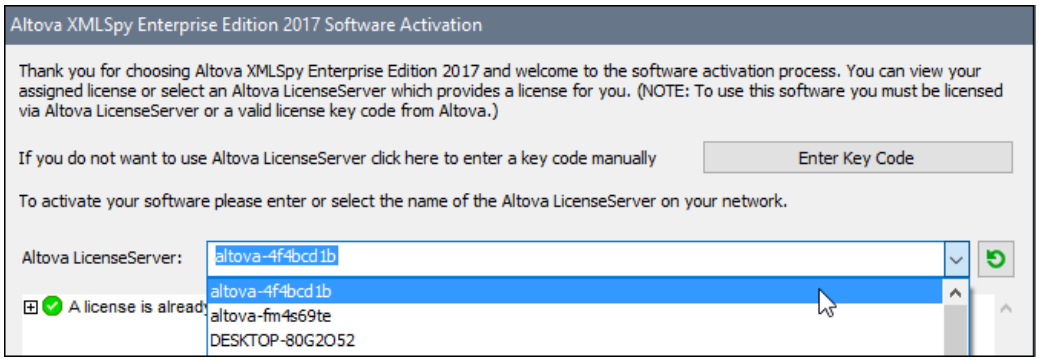
11.7.6.1 Registrar productos de escritorio de Altova

Siga estos pasos para registrar productos de escritorio de Altova con un servidor Altova LicenseServer:

1. Seleccione el comando de menú **Ayuda | Activación del software** para abrir el cuadro de diálogo del mismo nombre. El cuadro de diálogo ofrece dos opciones: (i) activar el software a través de Altova LicenseServer o (ii) activarlo con el código clave del producto. En esta documentación solamente se describe el proceso de activación a través de Altova LicenseServer.
2. Para activar el producto a través de LicenseServer haga clic en el botón **Usar Altova LicenseServer** (situado en la parte inferior del cuadro de diálogo).



3. El cuadro de diálogo cambia al modo de activación de LicenseServer (*imagen siguiente*). Ahora, en el cuadro combinado *Altova LicenseServer*, seleccione un servidor LicenseServer de la lista desplegable.



Cuando se establezca la conexión con el servidor LicenseServer seleccionado, el producto se registrará inmediatamente. Observe que en la pestaña de gestión de clientes **Client Management** de LicenseServer, el producto aparece en la lista de productos del equipo cliente.

Quitar productos del registro

Para quitar productos del registro de licencias abra la pestaña de gestión de clientes **Client Management** de LicenseServer y en el panel derecho (llamado *Product licensing*) haga clic en el botón **Unregister Product** del producto correspondiente.

11.7.6.2 Registrar FlowForce Server

Este apartado explica:

- Cómo registrar FlowForce Server con LicenseServer
- Cómo abrir la página de configuración de FlowForce Server (Windows)
- Cómo abrir la página de configuración de FlowForce Server (Linux)
- Cómo registrar FlowForce Server desde la página de configuración (Windows/Linux)
- Cómo registrar FlowForce Server desde la línea de comandos (Windows)
- Cómo registrar FlowForce Server desde la línea de comandos (Linux)
- Pasos siguientes

Cómo registrar FlowForce Server con LicenseServer

Puede registrar FlowForce Server con LicenseServer de dos maneras:

- Desde la página de configuración de FlowForce Server (Windows/Linux)
- Desde la interfaz de la línea de comandos de FlowForce Server (Windows)
- Desde la interfaz de la línea de comandos de FlowForce Server (Linux)

Abrir la página de configuración de FlowForce Server (Windows)

Puede abrir la página de configuración de FlowForce Server de dos maneras:

- Desde el menú **Inicio** | **Altova FlowForce Server 2018 | Página de configuración de FlowForce Server**
- Con Altova ServiceController: haga clic en el icono de ServiceController de la bandeja del sistema. En el menú que aparece haga clic en **Altova FlowForce Web | Setup**. Aparece la página de configuración de FlowForce Server (*imagen anterior*).

Abrir la página de configuración de FlowForce Server (Linux)

Tras instalar FlowForce Server en Linux (consulte el manual de FlowForce Server para obtener

más información), inicie FlowForce Web Server como servicio con este comando:

```
sudo /etc/init.d/flowforcewebserver start
```

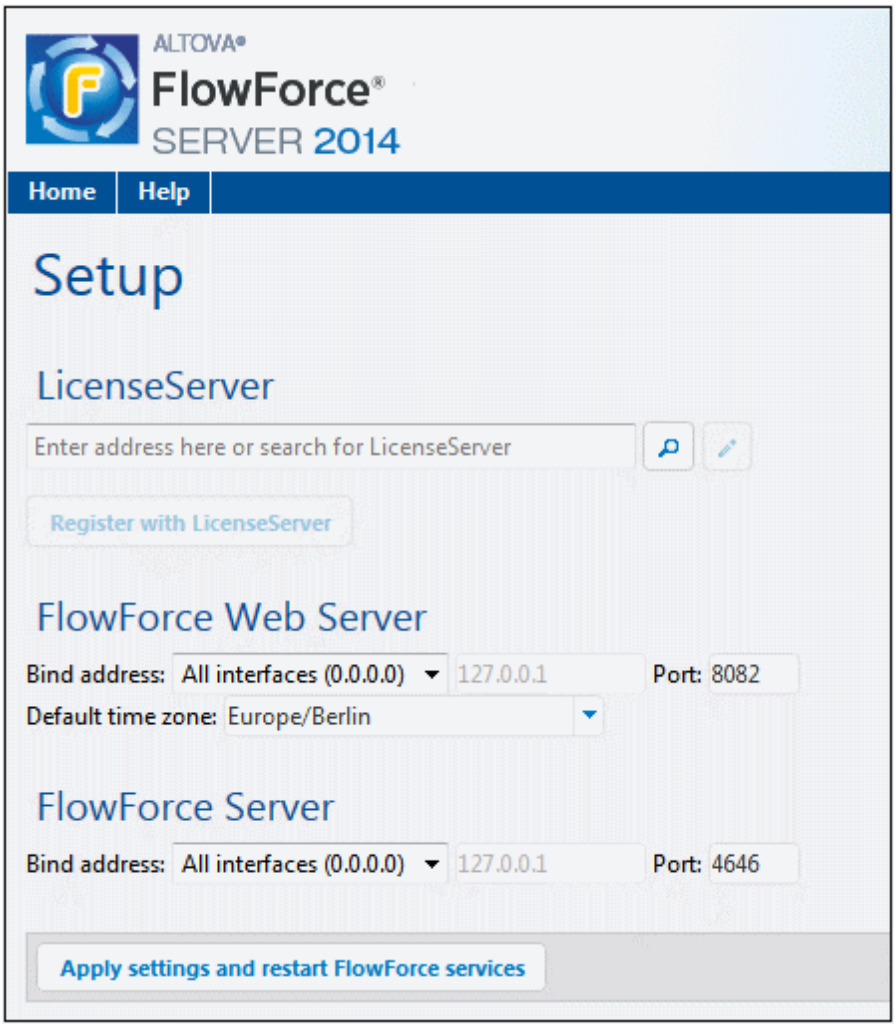
En la ventana de la Terminal aparece un mensaje con la URL en la que se está ejecutando FlowForceWeb:

FlowForceWeb running on `http://127.0.1.1:3459/setup?key=52239315203`

Copie la URL que aparece en el mensaje y péguela en la barra de dirección de su explorador web. Aparece la página de configuración de FlowForce Server (*imagen siguiente*).

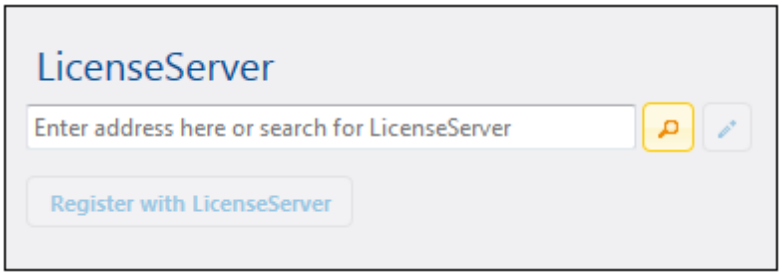
Registrar FlowForce Server desde la página de configuración

En la página de configuración (*imagen siguiente*) en el campo *LicenseServer* se debe indicar el servidor Altova LicenseServer con el que desea registrar el producto.

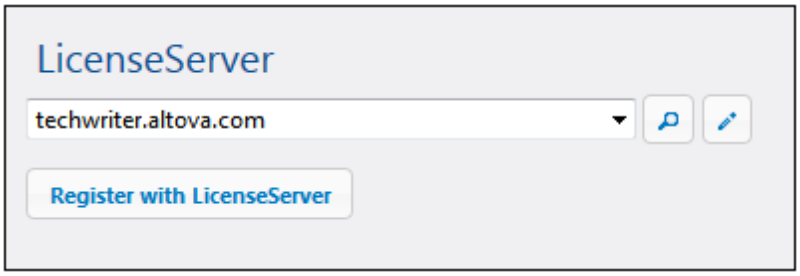


Puede rellenar este campo de dos maneras:

- Puede buscar servidores LicenseServer en la red (los que estén en ejecución). Para ello haga clic en el botón en forma de lupa (*resaltado en amarillo en la imagen siguiente*).



La búsqueda devuelve una lista con los servidores LicenseServer que están ejecutándose en la red. Seleccione el LicenseServer que almacena su licencia de FlowForce Server (*imagen siguiente*).

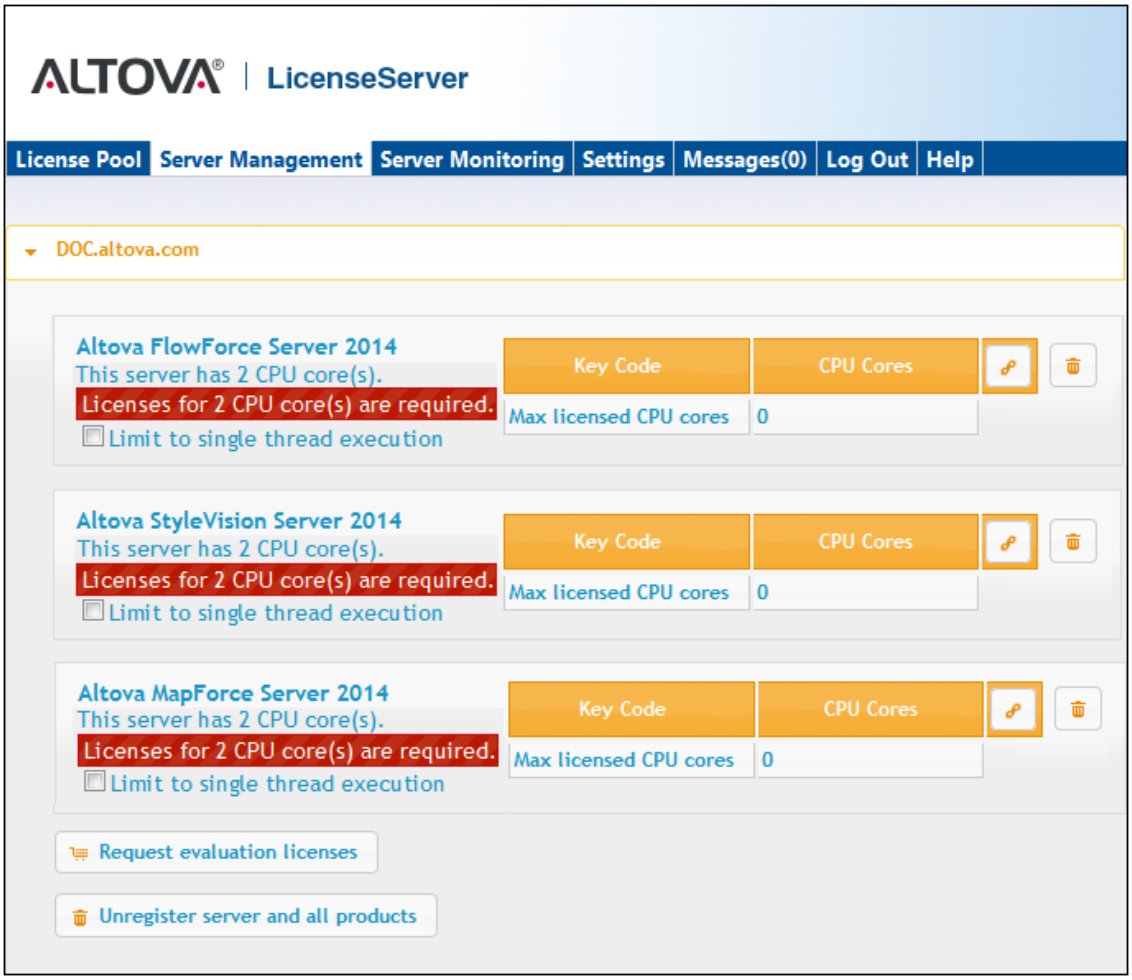


- O puede introducir la dirección del servidor LicenseServer directamente en el campo *LicenseServer*. Si está disponible una lista de servidores, haga clic en el botón en forma de lápiz para introducir la dirección en el campo *LicenseServer*.

Tras especificar el servidor LicenseServer, haga clic en el botón **Registrarse con LicenseServer**. FlowForce Server se registra con el LicenseServer indicado y se abre la página de configuración de LicenseServer por la pestaña de gestión de clientes **Client Management** (*imagen siguiente*).

Nota: quizá sea necesario permitir páginas emergentes en el explorador web.

 © 2018 Altova GmbH



En el ejemplo de la imagen anterior puede ver que se registraron tres productos con el servidor Altova LicenseServer `techwriter.altova.com`. El apartado Asignar licencias a productos registrados explica cómo se asignan las licencias.

Registrar FlowForce Server desde su interfaz de la línea de comandos (Windows)

En los equipos Windows también puede registrar FlowForce Server con un LicenseServer de la red desde la línea de comandos usando el comando `licenseserver`:

```
FlowForceServer licenseserver Servidor-O-Dirección-IP
```

Por ejemplo, si LicenseServer se está ejecutando en `http://localhost:8088`, entonces debe registrar FlowForce Server con:

```
FlowForceServer licenseserver localhost
```

Si el paquete de instalación de FlowForce Server incluía otros productos servidor de Altova, al registrar FlowForce Server se registran automáticamente los demás productos. Tras registrar FlowForce Server, vaya a LicenseServer y asígnele una licencia. Para más información consulte el apartado Asignar licencias a productos registrados.

Registrar FlowForce Server desde su interfaz de la línea de comandos (Linux)

En los equipos Windows también puede registrar FlowForce Server con un LicenseServer desde la línea de comandos usando el comando `licenseserver`. Recuerde que FlowForce Server debe iniciarse con privilegios de root:

```
sudo /opt/Altova/FlowForceServer2018/bin/flowforceserver licenseserver
localhost
```

En el comando anterior **`localhost`** es el nombre del servidor en el que está instalado LicenseServer. Observe que el ejecutable de FlowForce Server se encuentra en:

```
/opt/Altova/FlowForceServer2018/bin
```

Cuando termine de registrar FlowForce Server, vaya a LicenseServer y asígnele una licencia. Para más información consulte el apartado Asignar licencias a productos registrados.

¿Y ahora?

1. Si todavía no ha cargado su archivo de licencias en LicenseServer (ver el apartado Cargar licencias), cárguelo ahora y compruebe si la licencia que quiere usar está activa.
2. Asigne una licencia a su producto de Altova.

11.7.6.3 Registrar MapForce Server

Este apartado explica:

- Cómo registrar MapForce Server desde FlowForce Server (Windows)
- Cómo registrar MapForce Server (Windows)
- Cómo registrar MapForce Server (Linux)
- Pasos siguientes

MapForce Server se puede instalar como componente de FlowForce o como producto servidor independiente. En ambos casos es necesario registrarlo con Altova LicenseServer. Una vez registrado se le puede asignar una licencia desde LicenseServer. En sistemas Windows puede instalar MapForce Server como componente de FlowForce (desde el mismo paquete de instalación). Si MapForce Server está instalado como componente del paquete de FlowForce Server, se registrará automáticamente con LicenseServer cuando se registre FlowForce Server. En sistemas Linux, sin embargo, es necesario instalar MapForce Server por separado. Si

 © 2018 Altova GmbH

MapForce Server se instala después de FlowForce Server, entonces se registrará con LicenseServer cuando se registre FlowForce Server. Pero si se instala antes que FlowForce Server, deberá registrar los dos productos por separado.

Registrar MapForce Server desde FlowForce Server (Windows)

MapForce Server es un componente de FlowForce Server así que, al registrar FlowForce Server con un LicenseServer de la red, MapForce Server se registra con el mismo LicenseServer automáticamente. Para más información consulte Registrar FlowForce Server con LicenseServer.

Una vez registrado, asígnele una licencia desde LicenseServer. Para más información consulte Asignar licencias a productos registrados.

Registrar MapForce Server como producto independiente (Windows)

Si instaló MapForce Server como producto independiente, regístrelo con un Altova LicenseServer de su red y después asígnele una licencia. Para registrar MapForce Server desde la línea de comandos use el comando `licenseserver`:

```
MapForceServer licenseserver Servidor-o-Dirección-IP
```

Por ejemplo, si LicenseServer se está ejecutando en `http://localhost:8088`, entonces debe registrar MapForce Server con:

```
MapForceServer licenseserver localhost
```

Una vez registrado, asígnele una licencia desde LicenseServer. Para más información consulte Asignar licencias a productos registrados.

Registrar MapForce Server (Linux)

En equipos Linux utilice el comando `licenseserver` para registrar MapForce Server con LicenseServer desde la línea de comandos. Recuerde que debe iniciar MapForce Server con privilegios de root:

```
sudo /opt/Altova/MapForceServer2018/bin/mapforceserver licenseserver
localhost
```

En el comando anterior **`localhost`** es el nombre del servidor en el que está instalado LicenseServer. Observe que el ejecutable de MapForce Server se encuentra en:

```
/opt/Altova/MapForceServer2018/bin
```

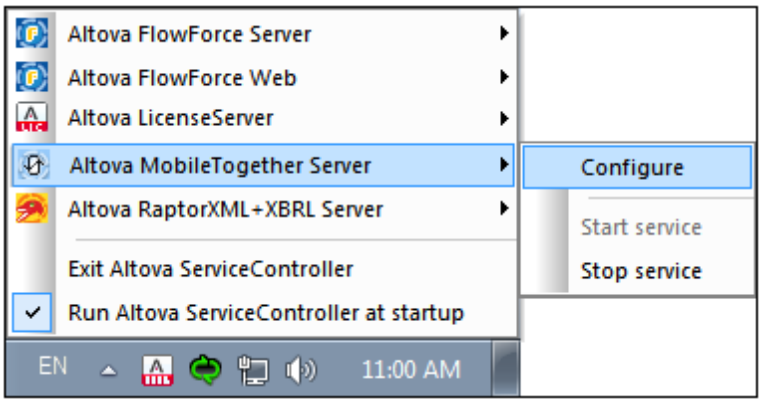
Una vez registrado, asígnele una licencia desde LicenseServer. Para más información consulte Asignar licencias a productos registrados.

¿Y ahora?

1. Si todavía no ha cargado su archivo de licencias en LicenseServer (ver el apartado Cargar licencias), cárguelo ahora y compruebe si la licencia que quiere usar está activa.
2. Asigne una licencia a su producto de Altova.

11.7.6.4 Registrar MobileTogether Server

Para iniciar MobileTogether Server, haga clic en el icono del controlador de servicios en la bandeja del sistema. En el menú que aparece pase el puntero por encima de **Altova MobileTogether Server** (*imagen siguiente*) y seleccione **Start service** para iniciar el servicio. Si MobileTogether Server ya está en ejecución, la opción **Start service** estará deshabilitada.



MobileTogether Server se puede registrar:

- desde la página **Configuración** de la interfaz web de MobileTogether Server: (i) inicie MobileTogether Server desde el controlador de servicios (punto anterior); (ii) Escriba la contraseña para acceder a la página de configuración; (iii) Seleccione el nombre o la dirección del servidor LicenseServer en el panel *LicenseServer* situado al final de la página y haga clic en el botón **Registrar con LicenseServer**.

- a través de la interfaz de la línea de comandos con ayuda del comando `licenseserver`:
  `MobileTogetherServer licenseserver [opciones] NombreServidor-O-DirecciónIP`

  Por ejemplo, si el nombre del servidor donde está instalado LicenseServer es `localhost`, entonces debe usar este comando:
  `MobileTogetherServer licenseserver localhost`

Tras registrar MobileTogether Server con el servidor LicenseServer, abra la pestaña **Client Management** de la interfaz web de LicenseServer para asignar una licencia a MobileTogether Server.

11.7.6.5 Registrar RaptorXML(+XBRL) Server

Este apartado explica:

- Cómo registrar RaptorXML(+XBRL) Server (Windows)
- Cómo registrar RaptorXML(+XBRL) Server (Linux)
- Pasos siguientes

Primero debe instalar RaptorXML(+XBRL) Server en el equipo servidor o en la misma red a la que está conectado LicenseServer y después debe iniciarlo como servicio. Después debe registrarlo con LicenseServer. Una vez registrado, podrá asignarle una licencia desde LicenseServer. Esta sección explica cómo registrar RaptorXML(+XBRL) Server con LicenseServer.

Registrar RaptorXML(+XBRL) Server (Windows)

Puede registrar RaptorXML(+XBRL) Server desde la interfaz de la línea de comandos con ayuda del comando `licenseserver`:

RaptorXML Server:	**`RaptorXML licenseserver`** ***`Servidor-O-Dirección-IP`***
RaptorXML+XBRL Server:	**`RaptorXMLXBRL licenseserver`** ***`Servidor-O-Dirección-IP`***

Por ejemplo, si LicenseServer se está ejecutando en `http://localhost:8088`, entonces debe registrar RaptorXML(+XBRL) Server con:

RaptorXML Server:	**`RaptorXML licenseserver hostlocal`**
RaptorXML+XBRL Server:	**`RaptorXMLXBRL licenseserver hostlocal`**

Tras registrar RaptorXML(+XBRL) Server, abra la interfaz de administración de LicenseServer y asigne una licencia a RaptorXML(+XBRL) Server. Esto se describe en el apartado Asignar licencias a productos registrados.

Registrar RaptorXML(+XBRL) Server (Linux)

En los equipos Linux puede registrar RaptorXML(+XBRL) Server con LicenseServer con ayuda del comando `licenseserver` de la interfaz de la línea de comandos de RaptorXML(+XBRL) Server. Recuerde que para ello es necesario iniciar RaptorXML(+XBRL) Server con privilegios root.

```
sudo /opt/Altova/RaptorXMLServer2018/bin/raptorxmlserver licenseserver
localhost
sudo /opt/Altova/RaptorXMLXBRLServer2018/bin/raptorxmlxbrlserver licenseserver
localhost
```

En el comando anterior **`localhost`** es el nombre del servidor en el que está instalado LicenseServer. Además observe que la ubicación del ejecutable de RaptorXML(+XBRL) Server es:

```
/opt/Altova/RaptorXMLServer2018/bin
/opt/Altova/RaptorXMLXBRLServer2018/bin
```

Tras registrar RaptorXML(+XBRL) Server, abra la interfaz de administración de LicenseServer y asigne una licencia a RaptorXML(+XBRL) Server. Esto se describe en el apartado Asignar

licencias a productos registrados.

¿Y ahora?

1. Si todavía no ha cargado su archivo de licencias en LicenseServer (ver el apartado Cargar licencias), cárguelo ahora y compruebe si la licencia que quiere usar está activa.
2. Asigne una licencia a su producto de Altova.

11.7.6.6 Registrar StyleVision Server

Este apartado explica:

- Cómo registrar StyleVision Server desde FlowForce Server (Windows)
- Cómo registrar StyleVision Server (Windows)
- Cómo registrar StyleVision Server (Linux)
- Pasos siguientes

StyleVision Server se puede instalar como componente de FlowForce o como producto servidor independiente. En ambos casos es necesario registrarlo con Altova LicenseServer. Una vez registrado se le puede asignar una licencia desde LicenseServer. Si instala StyleVision Server como componente de FlowForce (desde el mismo paquete de instalación) en un sistema Windows, StyleVision Server se registra automáticamente al registrar FlowForce Server. En sistemas Linux, sin embargo, StyleVision Server se registra automáticamente al registrar FlowForce Server solamente si instaló StyleVision Server después de instalar FlowForce Server.

Registrar StyleVision Server desde FlowForce Server (Windows)

StyleVision Server es un componente de FlowForce Server así que, al registrar FlowForce Server con un LicenseServer de la red, StyleVision Server se registra automáticamente con el mismo LicenseServer. Para más información consulte Registrar FlowForce Server con LicenseServer.

Una vez registrado, asígnele una licencia desde LicenseServer. Para más información consulte Asignar licencias a productos registrados.

Registrar StyleVision Server como producto independiente (Windows)

Si instaló StyleVision Server como producto independiente, regístrelo con un Altova LicenseServer de su red y después asígnele una licencia. Para registrar StyleVision Server desde la línea de comandos use el comando `licenseserver`:

```
StyleVisionServer licenseserver Servidor-o-Dirección-IP
```

Por ejemplo, si LicenseServer se está ejecutando en `http://localhost:8088`, entonces debe registrar StyleVision Server con:

 © 2018 Altova GmbH

```
StyleVisionServer licenseserver localhost
```

Una vez registrado, asígnele una licencia desde LicenseServer. Para más información consulte Asignar licencias a productos registrados.

Registrar StyleVision Server (Linux)

En equipos Linux utilice el comando `licenseserver` para registrar StyleVision Server con LicenseServer desde la línea de comandos. Recuerde que debe iniciar StyleVision Server con privilegios de root:

```
sudo /opt/Altova/StyleVisionServer2018/bin/stylevisionserver licenseserver
localhost
```

En el comando anterior **`localhost`** es el nombre del servidor en el que está instalado LicenseServer. Observe que ejecutable de StyleVision Server se encuentra en:

```
/opt/Altova/StyleVisionServer2018/bin
```

Una vez registrado, asígnele una licencia desde LicenseServer. Para más información consulte Asignar licencias a productos registrados.

¿Y ahora?

1. Si todavía no ha cargado su archivo de licencias en LicenseServer (ver el apartado Cargar licencias), cárguelo ahora y compruebe si la licencia que quiere usar está activa.
2. Asigne una licencia a su producto de Altova.

11.7.7 Asignar licencias a productos registrados

Este apartado explica:

- Los pasos previos a la asignación de licencias
- La pestaña **Client Management**
- Los iconos de la pestaña **Client Management**
- Núcleos y licencias
- Cómo asignar una licencia
- Cómo quitar productos del registro de LicenseServer

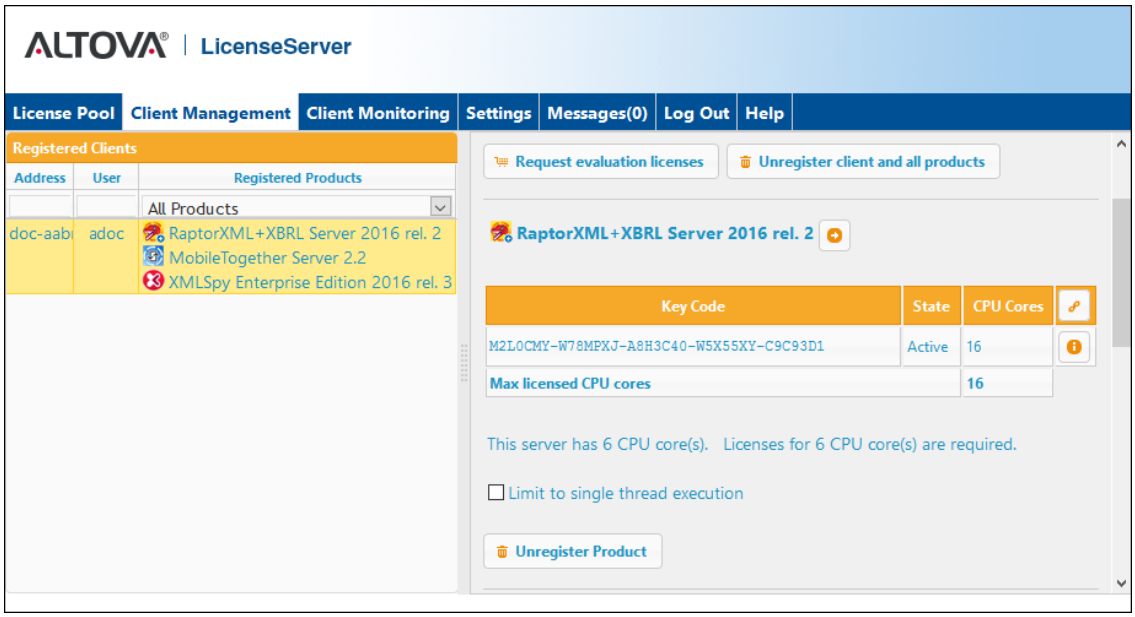
Pasos previos a la asignación de licencias

Antes de asignar una licencia a su producto de Altova, compruebe que:

- La licencia pertinente está cargada en el repertorio de licencias de LicenseServer y que la licencia está activa.
- Su producto de Altova está registrado con LicenseServer.

Pestaña Client Management

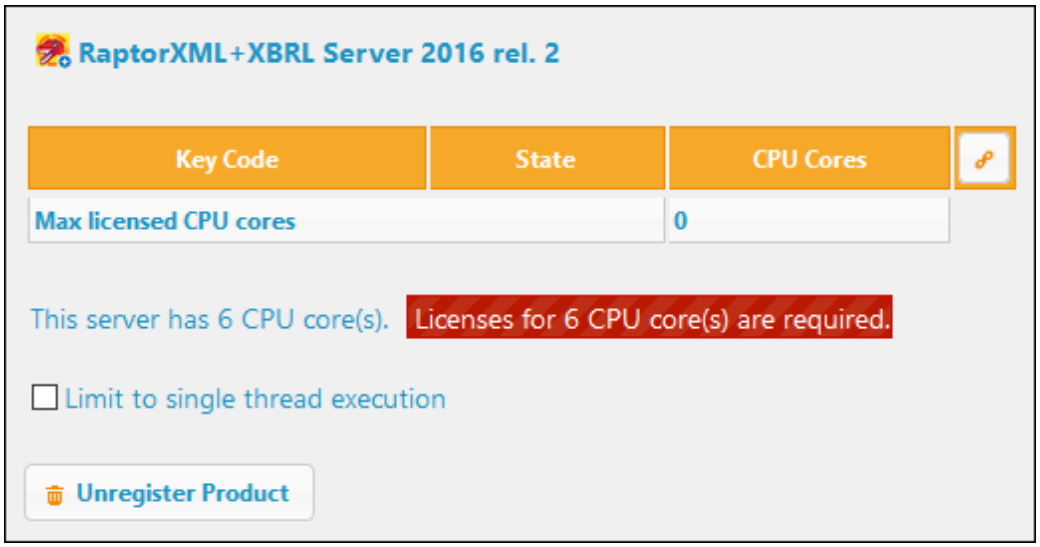
Las licencias se asignan en la pestaña de gestión de clientes **Client Management** de la página de configuración de LicenseServer (*imagen siguiente*). En el ejemplo de la imagen puede observarse (en el panel izquierdo) que hay un equipo cliente con tres productos de Altova registrados con LicenseServer.



Es necesario mencionar estos aspectos de la pestaña **Client Management**:

- En el panel izquierdo se enumeran todos los equipos cliente y sus productos. En la imagen anterior, por ejemplo, vemos que la lista incluye un solo equipo cliente. Este equipo cliente tiene tres productos de Altova registrados con LicenseServer.
- Cuando seleccione un equipo cliente en el panel izquierdo, la información sobre las licencias de los productos registrados aparecen en el panel derecho de la pestaña. En este panel podrá editar las asignaciones de licencia de cada producto.
- Todos los productos de Altova registrados con el servidor de licencias tienen una entrada llamada *Key Code*, donde aparece el código clave de la licencia que tiene asignada. Para asignar una licencia a un producto registrado basta con hacer clic en el icono **Editar licencias asignadas** (ver lista de iconos más abajo) y seleccionar la licencia correspondiente en el repertorio de licencias.
- Los productos servidor también tienen una entrada donde se especifica el número de núcleos CPU que deben contar una licencia para poder ejecutar el producto en ese cliente. Si el número de núcleos con licencia es inferior al número necesario, aparece un aviso en color rojo (*imagen siguiente*). Recuerde que el número de núcleos CPU que necesitan licencia es el número de núcleos CPU que hay en el cliente.

 © 2018 Altova GmbH



- Si se instalaron **varias versiones** del mismo producto en el mismo equipo (p. ej. StyleVision Server 2013 y StyleVision Server 2014) y todas ellas se registraron con el mismo servidor LicenseServer, todos los registros se reúnen en un solo registro en la pestaña **Client Management**, es decir, aparecen como un solo registro. Cuando se asigna una licencia a este registro, esta se asigna a todas las instalaciones reunidas en ese registro. Sin embargo, en el equipo cliente se pueden ejecutar simultáneamente varias instancias de una instalación solamente. Por ejemplo, puede ejecutar simultáneamente varias instancias de StyleVision Server 2013 o varias instancias de StyleVision Server 2014, pero puede ejecutar simultáneamente una instancia de StyleVision Server 2013 y otra de StyleVision Server 2014. Recuerde que si instala versiones nuevas, debe asignarles una licencia para poder ejecutarlas.
- Las versiones nuevas de los productos servidor solamente pueden recibir licencias desde la versión más reciente de LicenseServer. Las versiones anteriores de los productos servidor de Altova funcionarán con las conversiones más recientes de LicenseServer. Es decir, si instala una versión nueva de un producto servidor de Altova y la versión actual de su servidor LicenseServer no es la más reciente, desinstale la versión antigua de LicenseServer e instale la versión más reciente (disponible en el sitio web de Altova). Toda la información de registro y licencias almacenada en la versión antigua se guardará en una base de datos del equipo cliente durante la desinstalación y se importará automáticamente a la versión nueva. El número de versión de su servidor LicenseServer actual aparece en la parte inferior de la página de configuración. El número de versión de LicenseServer también aparece durante la instalación (tanto de LicenseServer como del producto servidor que contiene el paquete de instalación de LicenseServer). Recuerde que puede instalar LicenseServer junto con el producto servidor o por separado. Si quiere actualizar LicenseServer con una versión más reciente, recuerde que antes debe desinstalar la versión anterior.

Iconos de la pestaña Client Management

Editar licencias asignadas. Este icono aparece junto a cada uno de los productos y sirve para abrir el cuadro de diálogo Edit Assigned Licenses, donde puede asignar

nuevas licencias y editar licencias ya asignadas.

Mostrar licencias. Este icono aparece junto a cada una de las licencias y sirve para cambiar a la pestaña del repertorio de licencias (**License Pool**), donde se resalta la licencia seleccionada. Allí podrá leer toda la información disponible sobre la licencia.

Quitar producto del registro: elimina el producto seleccionado del registro de LicenseServer.

Nota sobre núcleos y licencias

La asignación de licencias a productos servidor de Altova depende de cuántos núcleos de procesador tiene el equipo donde se ejecuta el producto servidor de Altova. Por ejemplo, un procesador dual tiene dos núcleos, un procesador *quad* tiene cuatro núcleos, un procesador *hexa-core* tiene seis núcleos, y así sucesivamente. El número de núcleos de la licencia asignada a un producto debe ser mayor o igual al número de núcleos disponibles en dicho equipo servidor, ya sea un servidor físico o un equipo virtual.

Por ejemplo, si un servidor tiene ocho núcleos (un procesador *octa-core*), deberá comprar una licencia para ocho núcleos. También puede combinar varias licencias para alcanzar el número de núcleos necesario. Es decir, puede usar dos licencias para cuatro núcleos para un servidor *octa-core* en lugar de una licencia para ocho núcleos, por ejemplo.

Si usa un equipo servidor con gran cantidad de núcleos, pero tiene un bajo volumen de procesamiento, también puede crear un equipo virtual que tenga adjudicados menos núcleos y comprar una licencia para ese menor número de núcleos. No obstante, dicha implementación será menos rápida que si utilizara todos los núcleos disponibles en el servidor.

Nota: cada licencia de los productos servidor de Altova se puede usar de forma simultánea en un equipo como máximo (en el equipo donde está instalado el producto servidor de Altova), incluso si la capacidad de la licencia no está agotada. Por ejemplo, si utiliza una licencia para 10 núcleos para un equipo cliente que tiene 6 núcleos, los 4 núcleos restantes de la licencia no se pueden usar simultáneamente en otro equipo cliente.

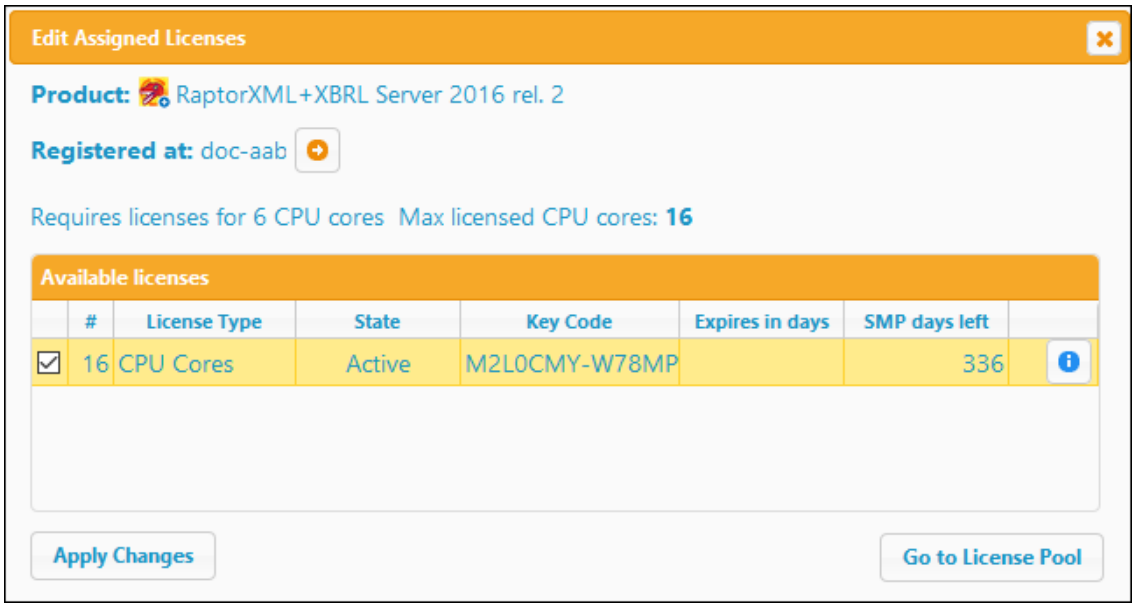
Licencias de MobileTogether Server

Las licencias de MobileTogether Server funciona en base al número de núcleos CPU que tenga el equipo donde se ejecuta MobileTogether Server. Las licencias basadas en el número de núcleos permiten conectar un número ilimitado de dispositivos al servidor. Sin embargo, si marca la casilla *Limit to single thread execution*, entonces solo se podrá conectar a MobileTogether Server un máximo de un dispositivo. Esto puede ser útil en tareas de evaluación y de pruebas a pequeña escala. No obstante, si estando marcada esta casilla, se conecta otro dispositivo a MobileTogether Server, este otro dispositivo se hará con la licencia. El primer dispositivo ya no se podrá conectar y recibirá un mensaje de error a tal efecto.

 © 2018 Altova GmbH

Asignación de licencias

Para asignar una licencia a un producto registrado, haga clic en el icono **Editar licencias asignadas** del producto. Aparece el cuadro de diálogo de edición de licencias (*imagen siguiente*).



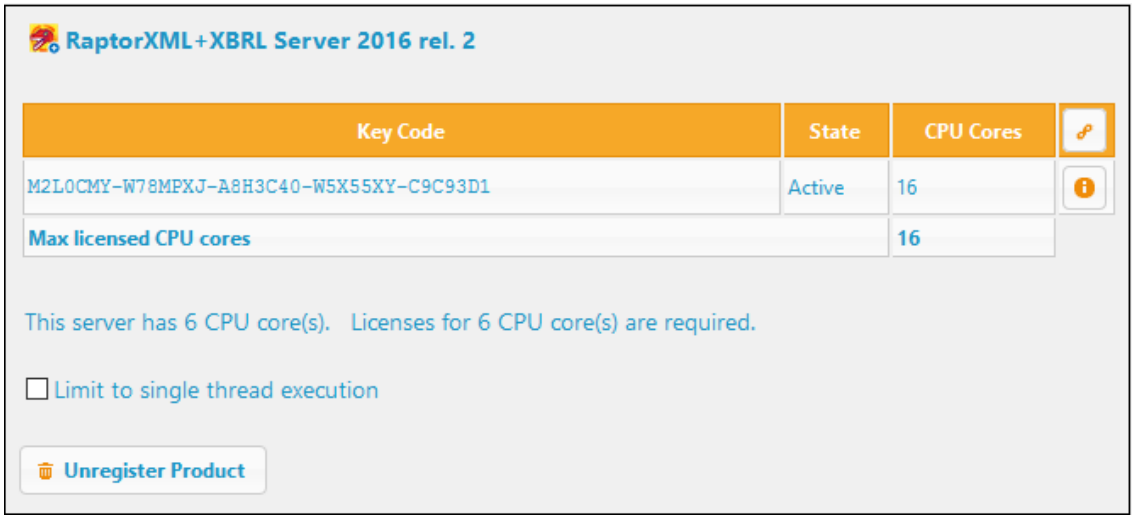
En este cuadro de diálogo:

- El producto al que se debe asignar la licencia aparece en negrita en la parte superior izquierda del cuadro de diálogo. Por ejemplo, en la imagen anterior se trata de Altova RaptorXML+XBRL Server.
- Después aparece el equipo donde está instalado el servidor (en este caso `doc-aab`).
- Las licencias que están activas en el repertorio para este producto aparecen enumeradas en la tabla "Available licenses". En nuestro ejemplo hay una licencia activa para RaptorXML+XBRL Server (LicenseServer detecta automáticamente para qué producto se emitió cada licencia del repertorio). El tipo de la licencia puede ser **`Cores`** (para todos los productos servidor de Altova, incluido MobileTogether Server) o **`Users`** (solo para MobileTogether Server). El tipo de licencia aparece en la columna *License Type*.
- La licencia de nuestro ejemplo es para 16 núcleos.
- Debe conocer el número de núcleos de procesador que hay en el servidor en el que está instalado el producto servidor de Altova. Si el equipo tiene un procesador dual, necesitará una licencia para dos núcleos (el número que aparece en la columna *CPU Cores*). Justo debajo del nombre del equipo aparece el número de núcleos que requiere el producto servidor registrado. La licencia que se asigne a este servidor debe ser válida para un número suficiente de núcleos. Además, recuerde que puede combinar licencias para alcanzar el número de núcleos necesario. Por ejemplo, si el procesador del equipo es de 8 núcleos puede combinar dos licencias para 4 núcleos.
- El cuadro de diálogo de edición de licencias solamente se enumeran las licencias que están activas para este producto. No aparecen licencias de otros productos.
- La casilla de las licencias ya asignadas previamente (por ejemplo en otra instalación del mismo producto en la red) aparecen marcadas. Es decir, solamente puede seleccionar las licencias cuya casilla no esté marcada.
- La columna *CPU Cores* (o # para los usuarios de MobileTogether Server) indica para

cuántos núcleos CPU (o clientes MobileTogether) se emitió la licencia.
- Si desea realizar cambios en el repertorio de licencias (para cargar, activar, desactivar o eliminar licencias) haga clic en el botón **Go to License Pool** para ir al repertorio de licencias.

Seleccione la licencia que desea asignar. Observe que la casilla de la licencia seleccionada ahora aparece marcada y que el número máximo de núcleos para esa licencia aparece en la parte superior izquierda del diálogo (*Max licensed CPU Cores)*. Puede seleccionar más licencias si quiere incrementar el número de núcleos para el producto y el cliente seleccionados. El número máximo de núcleos es la suma de los núcleos de todas las licencias seleccionadas.

Tras seleccionar las licencias haga clic en **Apply changes** para aplicar los cambios realizados en el diálogo. Las licencias se asignan al producto y aparecen en la tabla de la pestaña **Client Management** (*imagen siguiente*). En el ejemplo de la imagen se asignó una licencia de 16 núcleos a Altova RaptorXML+XBRL al equipo cliente.



Quitar productos del registro

Los productos de Altova registrados con LicenseServer se enumeran en el panel derecho de la pestaña **Client Management** bajo el nombre del equipo cliente donde están instalados y van acompañados del botón **Quitar producto del registro** . Haga clic en este botón para quitar el producto del registro de LicenseServer. Si el producto tiene asignada una licencia, la asignación expira nada más quitar el producto del registro. Para quitar todos los productos del registro, haga clic en el botón **Unregister Client and all products**, situado al principio del panel derecho de la pestaña.
Para volver a registrar un producto con LicenseServer:

- *Productos servidor:* abra la página **Opciones** en la interfaz web del producto servidor. Si el producto no cuenta con una interfaz web, abra una ventana del símbolo del sistema y use la interfaz de la línea de comandos para registrarlo. Para más información consulte Registrar FlowForce Server, Registrar MapForce Server, Registrar MobileTogether Server, Registrar StyleVision Server y Registrar RaptorXML(+XBRL) Server.

- *Productos de escritorio:* abra el cuadro de diálogo "Activación del software" (**Ayuda | Activación del software**), cambie al modo de activación a través de LicenseServer y en el campo *Altova LicenseServer* elija el servidor LicenseServer con el que desea registrar el producto. El producto se registrará y aparecerá en la pestaña **Client Management** del servidor LicenseServer, en la lista de productos registrados de dicho cliente.

11.8 Referencia de la página de configuración

La página de configuración de LicenseServer es la interfaz del administrador con el servidor de licencias. Permite gestionar el servidor y asignar licencias a los productos de Altova que se registraron con LicenseServer. La página de configuración de LicenseServer se abre con un explorador web. Las instrucciones para abrir la página de configuración están en los apartados Abrir la página de configuración de LicenseServer (Windows), Abrir la página de configuración de LicenseServer (Linux) y Abrir la página de configuración de LicenseServer (macOS).

Esta sección es una referencia del usuario de la página de configuración y sus apartados se corresponden con las pestañas en que está dividida la página:

- **Repertorio de licencias**: describe el uso de la pestaña **License Pool**.
- **Gestión de clientes**: describe el uso de la pestaña **Client Management**.
- **Supervisión de clientes**: describe el uso de la pestaña **Client Monitoring**.
- **Opciones de configuración**: describe el uso de la pestaña **Settings**.
- **Mensajes, Cerrar sesión**

Consulte el apartado Asignar licencias para obtener más información sobre el proceso de asignación de licencias.

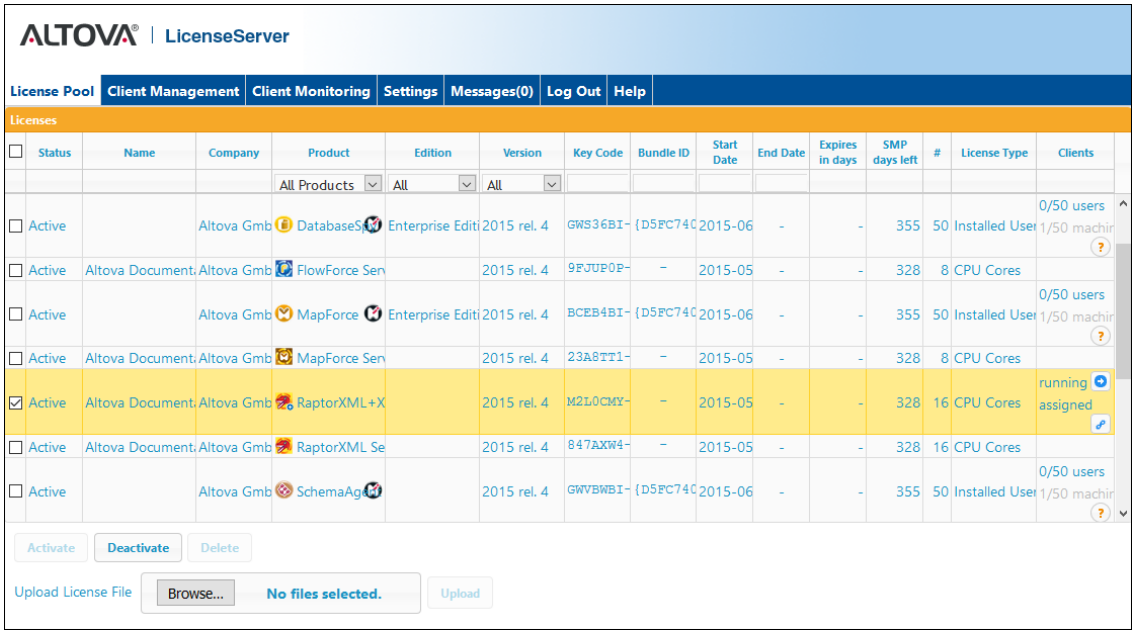
11.8.1 Repertorio de licencias

Temas de este apartado:

- Cargar licencias
- Estado de las licencias
- Activar, desactivar y eliminar licencias
- Iconos de la pestaña **License Pool**
- Información sobre las licencias
- Nota sobre licencias para productos de escritorio
- Nota sobre núcleos y licencias

El repertorio de licencias (pestaña **License Pool**) muestra información sobre las licencias que están disponibles en LicenseServer (*imagen siguiente*). Cuando cargue un archivo de licencias con el botón **Upload**, todas las licencias que contiene el archivo se colocan en el repertorio de licencias LicenseServer. En otras palabras, el repertorio de licencia ofrece un resumen de todas las licencias de productos Altova que están disponibles en LicenseServer e información sobre cada una de estas licencias. En el repertorio de licencias no solo puede cargar licencias, sino que además puede activarlas, desactivarlas y eliminarlas.

 © 2018 Altova GmbH



Cargar licencias

Siga estos pasos para cargar un archivo de licencias (el archivo con extensión `.altova_licenses` que Altova GmbH le envía para su producto de Altova):

1. Haga clic en el botón **Browse** para buscar el archivo de licencias.
2. Seleccione el archivo de licencias y haga clic en el botón **Upload** para cargar el archivo de licencias.
3. Las licencias del archivo se colocan en el repertorio de licencias y aparecen en la tabla de la pestaña (*imagen anterior*).

Estado de las licencias

Las licencias pueden tener varios estados:

- ***Activating*** *(En activación):* cuando se carga una licencia al repertorio de licencias de LicenseServer, el servidor transmite datos sobre la licencia al servidor maestro `altova.com` para validarla, verificarla y activarla. Este paso es necesario para comprobar que se cumple el contrato de licencia de Altova. Mientras se completa esta transacción (que dura entre 30 segundos y varios minutos, dependiendo de la velocidad y del tráfico de la conexión) el estado de la licencia es ***Activating...*** (*En activación...*).

- ***Failed verification*** *(No se puede verificar):* si no se puede establecer la conexión con el servidor maestro `altova.com`, el estado de la licencia es ***Failed verification*** *(No se*

puede verificar). Si esto ocurre, revise su conexión a Internet y las reglas de su servidor de seguridad y compruebe que LicenseServer puede comunicarse con el servidor maestro `altova.com`.

- ***Active*** *(Activa):* la licencia se activa tras ser verificada.
- ***Inactive*** *(No activa):* si la licencia se verifica pero está presente en otro LicenseServer de la red, su estado es ***Inactive*** *(No activa)*. Esto también puede ocurrir cuando el administrador desactiva la licencia a mano en el repertorio de licencias.
- ***Pending*** *(Pendiente):* este es el estado de la licencia si su fecha de inicio es una fecha futura. El día de la fecha de inicio el estado pasará a ser ***Active*** a las 00.00 horas de la fecha de inicio. Puede asignar licencias con estado ***Pending*** a un producto y así asegurarse de que siga funcionando cuando la licencia actual expire. La transición a la nueva licencia está diseñado para que sea lo menos problemático posible y no interrumpe ningún proceso que pueda estar en ejecución en el cliente.
- ***Blocked*** *(Bloqueada):* la licencia tiene este estado cuando se produce un error al verificar la licencia y el servidor maestro `altova.com` no concede permiso a LicenseServer para usar la licencia. Esto puede deberse a una infracción del contrato de licencia o a un uso excesivo de una licencia, por ejemplo. Tras solucionar el problema, elimine la licencia, vuelva a cargarla y vuelva a activarla. Consulte la tabla que aparece a continuación para obtener más información.

A continuación resumimos la información anterior:

Estado	Significado
Activating *(En activación)*	Al cargar un archivo de licencias, los datos se envían a `altova.com` para verificar las licencias. Actualice el explorador para ver el estado de las licencias. El proceso de verificación y activación puede durar unos minutos.
Failed verification *(No se puede verificar)*	No se puede establecer la conexión con `altova.com`. Cuando consiga establecer la conexión, vuelve a iniciar LicenseServer o intente activar la licencia otra vez (con el botón **Activate**).
Active *(Activa)*	La licencia se verificó correctamente y la licencia está activa.
Inactive *(No activa)*	La licencia se verificó correctamente pero la licencia también está en otro servidor LicenseServer de la red. Las licencias se pueden desactivar con el botón **Deactivate**.
Pending *(Pendiente)*	Este es el estado de la licencia si su fecha de inicio es una fecha futura. En la fecha de inicio el estado pasa a ***Active***. Puede asignarse a un producto para garantizar la renovación automática de la licencia antes de expire la licencia actual.
Blocked *(Bloqueada)*	No se puede verificar la licencia. La licencia no es válida y se bloqueó. Una vez resuelto el problema que tenía bloqueada la licencia, elimine la licencia, vuelva a cargarla y vuelva a activarla. Cada vez que se carga una licencia, el servidor se pone en contacto con el servidor maestro de Altova, que iniciará una nueva verificación. Si no vuelve a cargar la licencia, la verificación deberá esperar hasta que se produzca la siguiente comunicación con el servidor maestro de Altova, que podría estar programada el día siguiente. Póngase en

 © 2018 Altova GmbH

	contacto con el equipo de soporte técnico de Altova si el problema continúa.

Nota: tras enviar los datos de licencia a `altova.com` es necesario actualizar la ventana del explorador para actualizar el estado. La verificación y activación puede tardar unos minutos.

Nota: si no se puede establecer la conexión con `altova.com`, el estado de la licencia es ***Failed verification*** *(No se puede verificar)*. Cuando consiga establecer la conexión, vuelve a iniciar LicenseServer o intente activar la licencia otra vez (con el botón **Activar**).

Nota: cuando la licencia tiene el estado ***Inactive*** *(No activa)* o ***Blocked*** *(Bloqueada)* aparece un mensaje explicando el motivo en la pestaña **Messages**.

A los productos solamente se les pueden asignar licencias activas o pendientes. Las licencias activas se pueden activar o eliminar del repertorio de licencias. Si una licencia se elimina del repertorio, se puede cargar otra vez cargando el archivo de licencias correspondiente. Cuando se actualiza un archivo de licencias, solamente se cargan al repertorio las licencias que todavía no están en él. Para activar, desactivar o eliminar una licencia, selecciónela y haga clic en el botón **Activate**, **Deactivate** o **Delete** respectivamente.

Conexión con el servidor maestro de licencias altova.com

Altova LicenseServer necesita comunicarse con el servidor maestro de licencias `altova.com` para validar y verificar los datos de licencia y para asegurarse de que los contratos de licencia de Altova se cumplen en todo momento. Esta comunicación se establece por HTTPS a través del puerto 443. Si tras completar la verificación inicial con el servidor maestro `altova.com`, Altova LicenseServer no puede volver a conectarse con `altova.com` durante más de 5 días (120 horas), se bloqueará el uso de los productos de Altova que estén conectados con Altova LicenseServer.

Cuando se pierde la conexión con el servidor maestro `altova.com`, la incidencia se registra en la pestaña **Messages** de la página de configuración de Altova LicenseServer. Además el administrador puede configurar Altova LicenseServer para que le envíe correos de alerta cuando se pierda la conexión con `altova.com`. Las opciones de configuración de estos correos de alerta se pueden definir en la pestaña **Settings** de la página de configuración.

Activar, desactivar y eliminar licencias

Para desactivar una licencia activa, selecciónela y haga clic en el botón **Deactivate**. Para activar una licencia inactiva, selecciónela y haga clic en el botón **Activate**. Para eliminarla haga clic en el botón **Delete**. Cuando se elimina una licencia, se quita del repertorio. Las licencias eliminadas se pueden añadir otra vez al repertorio cargando el archivo de licencias pertinente. Si vuelve a cargar un archivo de licencias, solamente se añaden las licencias que todavía no están en el repertorio. Las licencias que ya están no se vuelve a añadir.

Iconos de la pestaña License Pool

Logotipo de Altova MissionKit logo. Este icono aparece junto a cada producto de escritorio cuya licencia forme parte de una licencia para el kit de aplicaciones Altova MissionKit. *Consulte la nota sobre licencias para productos de escritorio que aparece más abajo*.

Mostrar cliente asignado. Este icono aparece en la columna *Clients* de una licencia que está asignada y sirve para cambiar a la pestaña **Client Management**, donde se pueden gestionar las licencias de los productos registrados de un cliente.

Mostrar cliente en ejecución. Este icono aparece en la columna *Clients* de una licencia que está asignada a un producto que está en ejecución en ese momento. Sirve para cambiar a la pestaña **Client Management**, donde se resalta el equipo cliente donde se ejecuta dicho producto. Aquí podrá consultar información sobre el cliente seleccionad y sobre su producto registrado.

Mostrar información. Este icono aparece en la columna Clients de una licencia que no está asignada. Haciendo clic en este icono podrá consultar el recuento de usuarios y si la licencia forma parte de un lote de licencias.

Información sobre las licencias

La tabla donde se enumeran las licencias incluye estos datos:

- ***Status.*** Una licencia puede tener estos estados: *Activating* | *Failed Verification* | *Active* | *Inactive* | *Blocked*. (*ver más arriba*).
- ***Name, Company***. Nombre y compañía del titular de la licencia. Estos datos se aportaron en el momento de compra.
- ***Product, Edition, Version***. Versión y edición de los productos con licencia. En la primera fila de cada columna aparece un cuadro combinado que sirve para filtrar las licencias por categorías.
- ***Key Code, Bundle ID.*** La clave de licencia para desbloquear el producto. Todos los productos de un lote de licencias de Altova MissionKit tienen el mismo identificador (*Bundle ID*). Los productos que no forman parte del MissionKit no tienen identificador (*Bundle ID*).
- ***Start Date, End Date.*** Estos dos datos (fecha de inicio y fecha de finalización) permiten conocer el período de validez de las licencias. Las licencias permanentes no tienen fecha de finalización.
- ***Expires in days, SMP (days left).*** El número de días que faltan para que la licencia expire. Todas las licencias vienen con un paquete de soporte y mantenimiento (SMP), que es válido durante un período determinado de tiempo. La columna SMP indica cuántos días de validez le quedan al paquete de soporte y mantenimiento.
- ***#, License Type***. El número de usuarios o núcleos CPU permitidos para la licencia aparece en la columna #. En cambio la columna *License Type* indica si la licencia regula usuarios o núcleos. En el caso de MobileTogether Server, también es necesario asignar las licencias que correspondan al número de **dispositivos** cliente MobileTogether que se conecten a MobileTogether Server. En los demás productos servidor de Altova es necesario asignar las licencias que correspondan al número de **núcleos CPU** solamente (*ver nota más abajo*). En los productos de escritorio de Altova las licencias se asignan en función del número de **usuarios**. *Consulte la nota sobre licencias para productos de escritorio que aparece más abajo*.
- ***Clients.*** Esta columna incluye entradas para licencias de MobileTogether Server (MTS) y para licencias de productos servidor solamente. No tiene entradas para licencias de

productos servidor aparte de MobileTogether Server. La columna indica si se asignaron licencias a dispositivos MobileTogether. Y, en el caso de los productos de escritorio, indica el número de equipos y de usuarios tal y como se explica a continuación.

Productos de escritorio: recuento de equipos y usuarios

- El recuento de equipos indica cuántos equipos cuentan con licencia para ejecutar el software. Por ejemplo: `7/10 machines` significa que la licencia puede utilizarse para instancias del software en 10 equipos y actualmente se utiliza para instancias del software en 7 equipos solamente. Haga clic en el icono **Mostrar cliente asignado** para cambiar a la pestaña de gestión de clientes **Client Management** y consulte los detalles de la licencia.
- El recuento de usuarios indica cuántos usuarios están utilizando la licencia. Para este recuento se tienen en cuenta solamente las instalaciones del software que cuentan con licencia y que están en ejecución. Por ejemplo, `3/10 users` significa que la licencia está siendo utilizada por 3 usuarios de un total de 10 usuarios permitidos. Si se está ejecutando un software que cuenta con licencia, puede hacer clic en el icono **Mostrar cliente en ejecución** para abrir la pestaña de supervisión de clientes **Client Monitoring** y consultar la información sobre los productos de Altova que se están ejecutando en los equipos cliente de la red.
- Estos dos recuentos (de equipos y usuarios) permiten conocer la capacidad actual de las licencias y su utilización. Por ejemplo: si el recuento de equipos de una licencia instalada basada en el número de usuarios es `7/10` y el recuento de usuarios es `3/10`, entonces podemos deducir que: (i) el software puede recibir licencias en un máximo de 10 equipos, (ii) el software recibió licencias en 7 equipos y (iii) 3 de estas 7 instalaciones están en ejecución en ese momento.

Anular asignación de licencias

Para anular la asignación de una licencia a un producto instalado en un equipo, abra la pestaña de gestión de clientes **Client Management**. Después seleccione el equipo y el software para el que desea anular la asignación de licencia. Después haga clic en el icono **Editar licencias asignadas**, anule la asignación de licencia y haga clic en el botón **Apply Changes**.

Nota sobre licencias para productos de escritorio

Existen tres tipos de licencias para productos de escritorio:

- ***Instalado:*** se adquiere una licencia para el número exacto de equipos donde desea instalar el software. Por ejemplo, si adquiere una licencia de tipo instalado para 10 usuarios, podrá instalar y usar el software en un máximo de 10 equipos. En cada equipo podrá iniciar tantas instancias del software como quiera de forma simultánea. La licencia para cada *usuario instalado* permite usar el producto de software en ese equipo.
- ***Concurrente:*** este tipo de licencia permite instalar el software en un número máximo de equipos que es 10 veces el número permitido de usuarios concurrentes y todas las instalaciones deben estar en la misma red. El software solo puede ser utilizado por el número permitido de usuarios concurrentes. Por ejemplo, si adquiere una licencia para 20 usuarios concurrentes, podrá instalar el software en 200 equipos de la red como máximo y se podrá usar en 20 equipos a la vez como máximo. Si quiere usar licencias de usuarios concurrentes en redes diferentes, deberá adquirir una licencia para cada red. Es decir, no puede usar una licencia para usuarios concurrentes en redes distintas.
- ***Usuario designado:*** cada licencia de usuario designado permite instalar el software en

un máximo de 5 equipos, pero el único usuario que puede usar el software es el usuario designado en la licencia. Con esta licencia el *usuario designado* puede trabajar en distintos equipos siempre y cuando se ejecute **un máximo de una instancia** del software.

Nota sobre licencias para Altova MissionKit

Altova MissionKit es un conjunto de herramientas formado por productos de escritorio de Altova. La licencia para Altova MissionKit incluye licencias independientes para cada uno de los productos que conforman el conjunto de herramientas. Cada una de estas licencias independientes tiene un código clave único, pero comparten el mismo identificador Bundle ID. Cuando cargue una licencia de Altova MissionKit en el repertorio de licencias, la lista de licencias incluirá cada una de las licencias de los diferentes productos del MissionKit (y junto a ellas aparecerá el logotipo de Altova MissionKit). Si quiere asignar una de estas licencias a un usuario concreto, a ese usuario se le asignarán también las licencias de los demás productos del MissionKit. El resultado es que no podrá asignar ningún producto de Altova MissionKit a ningún otro usuario.

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. (El número exacto de días se especifica en la opción *Extracción de licencias* de la pestaña de configuración **Settings**.) Esto le permitirá trabajar sin conexión a Internet, lo cual puede ser útil si desea trabajar en un entorno que no dispone de acceso a su servidor Altova LicenseServer (p. ej. cuando el producto servidor de Altova está instalado en un equipo portátil y el usuario se encuentra de viaje). Mientras la licencia esté extraída, LicenseServer indicará que la licencia está en uso y no podrá ser utilizada por ningún otro equipo. La licencia vuelve automáticamente a su estado insertado cuando finaliza el período de extracción de la licencia. La licencia extraída también se puede insertar en el servidor en cualquier momento con el botón **Insertar** del cuadro de diálogo "Activación del software". Para extraer una licencia del repertorio y almacenarla en el equipo, haga clic en el menú **Ayuda** del producto de Altova y seleccione el comando **Activación del software**. Para más información consulte el manual del usuario de su producto de Altova.

Nota sobre núcleos y licencias

La asignación de licencias a productos servidor de Altova depende de cuántos núcleos de procesador tiene el equipo donde se ejecuta el producto servidor de Altova. Por ejemplo, un procesador dual tiene dos núcleos, un procesador *quad* tiene cuatro núcleos, un procesador *hexa-core* tiene seis núcleos, y así sucesivamente. El número de núcleos de la licencia asignada a un producto debe ser mayor o igual al número de núcleos disponibles en dicho equipo servidor, ya sea un servidor físico o un equipo virtual.

Por ejemplo, si un servidor tiene ocho núcleos (un procesador *octa-core*), deberá comprar una licencia para ocho núcleos. También puede combinar varias licencias para alcanzar el número de núcleos necesario. Es decir, puede usar dos licencias para cuatro núcleos para un servidor *octa-core* en lugar de una licencia para ocho núcleos, por ejemplo.

Si usa un equipo servidor con gran cantidad de núcleos, pero tiene un bajo volumen de procesamiento, también puede crear un equipo virtual que tenga adjudicados menos núcleos y

comprar una licencia para ese menor número de núcleos. No obstante, dicha implementación será menos rápida que si utilizara todos los núcleos disponibles en el servidor.

Nota: cada licencia de los productos servidor de Altova se puede usar de forma simultánea en un equipo como máximo (en el equipo donde está instalado el producto servidor de Altova), incluso si la capacidad de la licencia no está agotada. Por ejemplo, si utiliza una licencia para 10 núcleos para un equipo cliente que tiene 6 núcleos, los 4 núcleos restantes de la licencia no se pueden usar simultáneamente en otro equipo cliente.

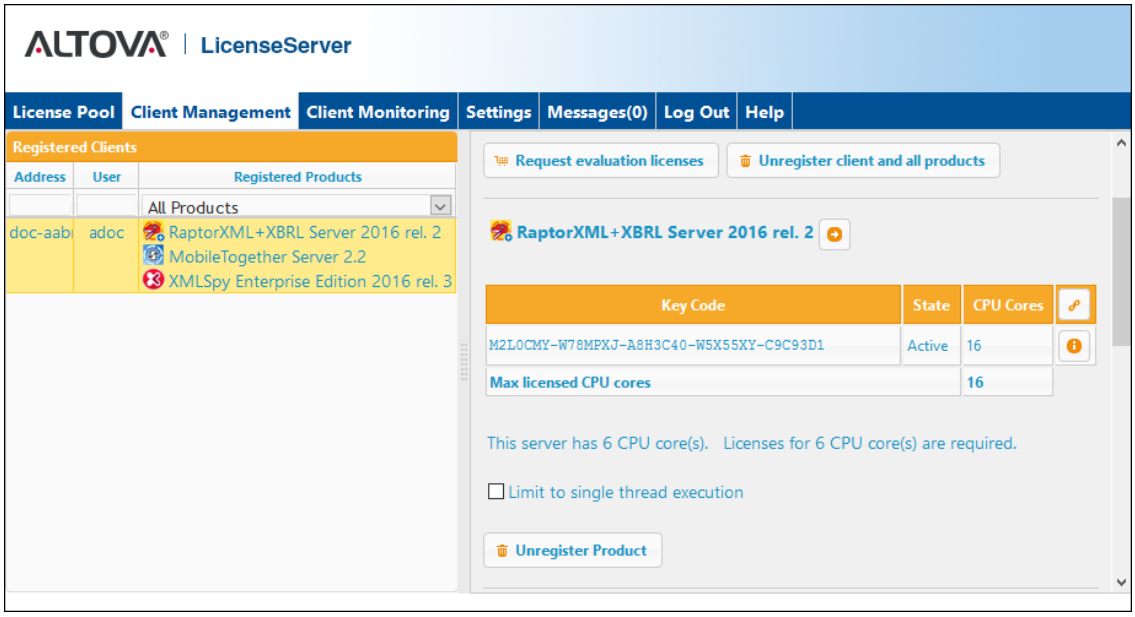
Licencias de MobileTogether Server
Las licencias de MobileTogether Server funciona en base al número de núcleos CPU que tenga el equipo donde se ejecuta MobileTogether Server. Las licencias basadas en el número de núcleos permiten conectar un número ilimitado de dispositivos al servidor. Sin embargo, si marca la casilla *Limit to single thread execution*, entonces solo se podrá conectar a MobileTogether Server un máximo de un dispositivo. Esto puede ser útil en tareas de evaluación y de pruebas a pequeña escala. No obstante, si estando marcada esta casilla, se conecta otro dispositivo a MobileTogether Server, este otro dispositivo se hará con la licencia. El primer dispositivo ya no se podrá conectar y recibirá un mensaje de error a tal efecto.

11.8.2 Gestión de clientes

Temas de este apartado:

- Iconos de la pestaña **Client Management**
- Gestión de licencias en el panel *Product Licensing*
- Asignación de licencias
- Ejecución por subprocesos simples
- Un equipo cliente con varios nombres diferentes
- Solicitar una licencia de evaluación
- Quitar productos del registro

La pestaña de gestión de clientes **Client Management** (*imagen siguiente*) se divide en dos grandes paneles:



- *Panel de clientes registrados:* el panel izquierdo se llama ***Registered clients*** y muestra una lista de equipos de red en los que hay instalados productos de Altova que están registrados con LicenseServer. Estos equipos se denominan *clientes registrados*. Cada cliente registrado aparece en el panel izquierdo junto a los productos que tiene registrados con LicenseServer. Las instrucciones para registrar productos con LicenseServer están en la sección Registrar productos con LicenseServer. Este panel también incluye una función de filtro en la primera fila de la tabla que le ayudará a navegar por la lista de clientes registrados.
- *Panel de licencias del producto:* cuando seleccione un cliente registrado en el panel izquierdo, el panel derecho mostrará la información de licencias de los productos registrados que hay en ese cliente. En este panel derecho podrá gestionar las licencias de cada uno de los productos registrados (ver instrucciones más abajo).

Iconos de la pestaña Client Management

Editar licencias asignadas. Este icono aparece junto a cada uno de los productos y sirve para abrir el cuadro de diálogo Edit Assigned Licenses, donde puede asignar nuevas licencias y editar licencias ya asignadas.

Mostrar licencias. Este icono aparece junto a cada una de las licencias y sirve para cambiar a la pestaña del repertorio de licencias (**License Pool**), donde se resalta la licencia seleccionada. Allí podrá leer toda la información disponible sobre la licencia.

Quitar producto del registro: elimina el producto seleccionado (del equipo cliente seleccionado) del registro de LicenseServer (más abajo encontrará más información). Para quitar un cliente y todos sus productos del registro, haga clic en el botón **Unregister client and all products** situado en la parte superior del panel derecho.

 © 2018 Altova GmbH

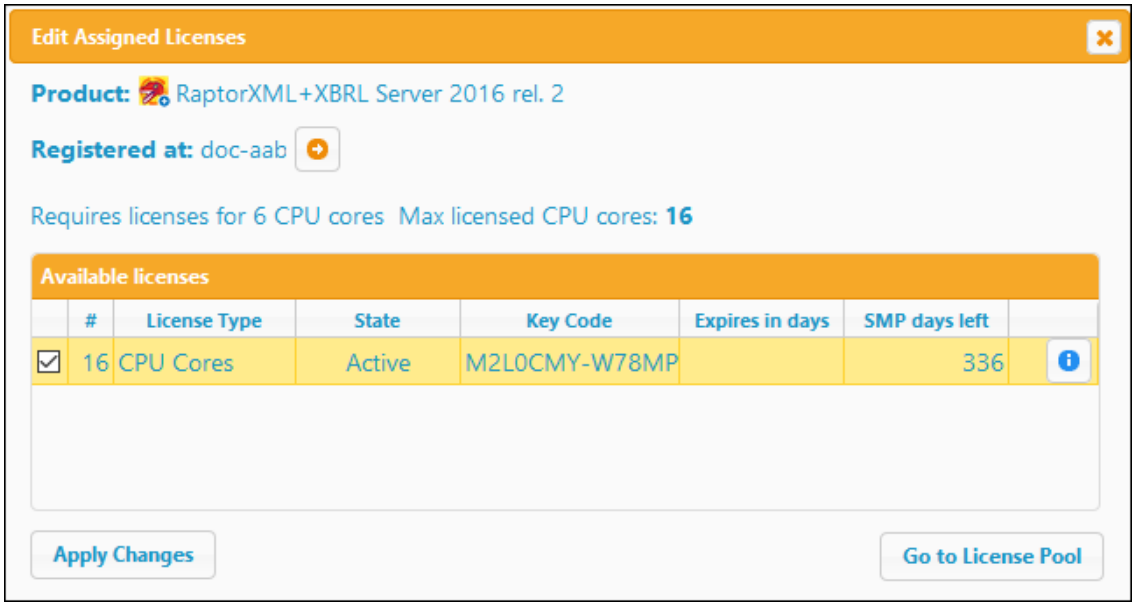
Gestión de licencias en el panel de licencias del producto

Estas son las acciones que se pueden llevar a cabo en el panel derecho:

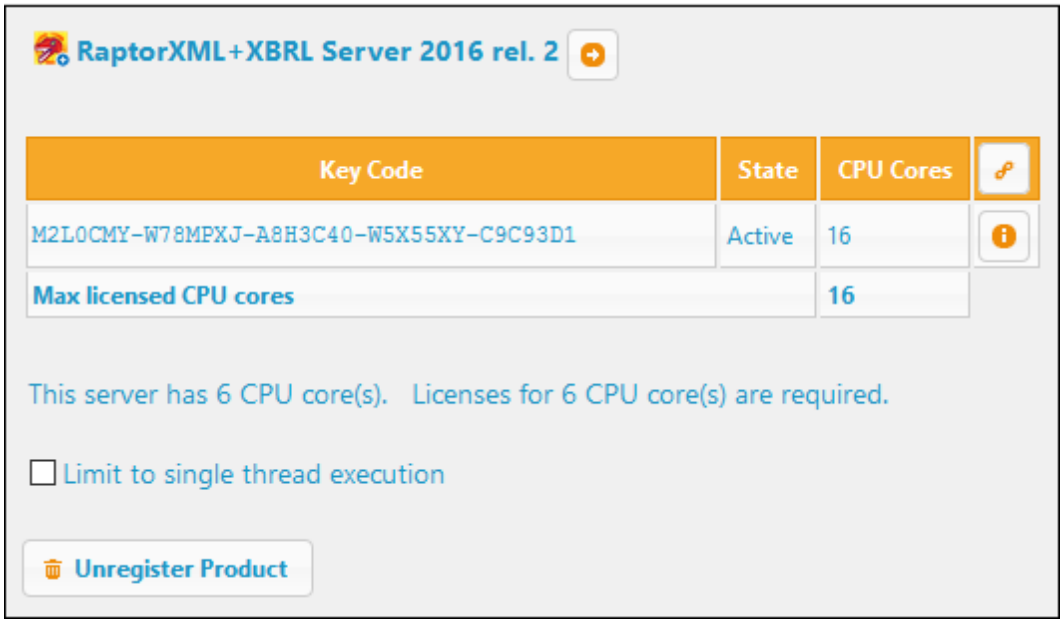
- *Asignar licencias, anular asignación de licencias y cambiar de licencia:* todas estas acciones se llevan a cabo con el icono **Editar licencias asignadas** que abre el cuadro de diálogo de edición de licencias (más abajo encontrará más información). Recuerde que los productos servidor incluyen una entrada dónde se especifica el número de núcleos CPU que deben contar con una licencia para poder ejecutar el producto servidor en ese cliente. Si el número de núcleos con licencia es inferior al número necesario, entonces aparece un aviso en color rojo. (El número de núcleos CPU que deben contar con una licencia es el número de núcleos CPU que tiene el cliente).
- *Configurar licencias de producto servidor para que usen un núcleo de procesador de un cliente:* consulte el apartado sobre ejecución por subprocesos simples que aparece más abajo.
- *Quitar un producto del registro de LicenseServer:* esto se hace con el botón **Quitar producto del registro** del producto. Consulte el apartado *Quitar productos del registro* que aparece más abajo.

Asignación de licencias

Para asignar una licencia a un producto registrado, haga clic en el botón **Editar licencias asignadas** del producto. Aparece el cuadro de diálogo de edición de licencias (*imagen siguiente*).



Seleccione qué licencias desea asignar y después haga clic en el botón **Apply Changes** para que los cambios surtan efecto. Las licencias se asignarán al producto seleccionado y aparecerán en el panel de licencias del producto (panel derecho) de la pestaña **Client Management** (*imagen siguiente*).



Ejecución por subprocesos simples
Si en el repertorio de licencias hay una licencia de producto servidor de Altova para un solo núcleo, puede asignársela a un equipo que tenga varios núcleos. En este caso, el equipo ejecutará el producto en un solo núcleo. El procesamiento será lógicamente más lento porque solo se usa un núcleo. Es decir, el producto se ejecutará en modo de subprocesamiento simple.

Para asignar una licencia de un solo núcleo a un equipo con varios núcleos basta con marcar la casilla *Limit to single thread execution* del producto.

En el caso concreto de **MobileTogether Server (MTS)**, si se seleccionó la ejecución por subprocesos simples para una licencia para un núcleo, entonces no se podrá conectar más de un dispositivo móvil a la vez al servidor MobileTogether Server. Es decir, cuando otro dispositivo se conecte a MobileTogether Server, este otro dispositivo tomará la licencia y el primer dispositivo ya no podrá conectarse al servidor (y recibirá un mensaje de error a tal efecto).

Un equipo cliente con varios nombres diferentes

Si registró varias veces el mismo equipo cliente con LicenseServer, puede que el equipo aparezca con varios nombres diferentes en la pestaña **Client Management**, es decir, puede que el equipo aparezca varias veces. Esto puede ocurrir si, por ejemplo, vuelve a registrar un equipo con un nombre de host distinto.

Cuando esto ocurra es importante evitar que se asignen licencias adicionales al mismo equipo o que se asigne la misma licencia varias veces al mismo equipo. Por ello lo mejor es quitar del registro los equipos cliente repetidos con el botón **Unregister client and all products** situado en la parte superior del panel derecho.

A continuación puede los nombres que puede recibir un equipo en la pestaña **Client Management**:

- *Nombre de host con nombre de dominio (es decir, el nombre de dominio completo), como por ejemplo:* **`win80-x64_1.mi.dominio.com`** *o* **`Doc3.mi.dominio.com`**. El equipo recibe este tipo de nombre cuando el nombre de host del equipo (con o sin la información de dominio) se pasa como argumento del comando `licenseserver` utilizado para registrar el producto servidor con LicenseServer. Por ejemplo: `<ProductoServidorAltova> licenseserver Doc3`. Este comando produce este nombre de dominio completo: `Doc3.mi.dominio.com`.

  También se produce un nombre de dominio completo cuando se introduce `localhost` como nombre de host en Windows 7 y Windows 10.

- *Nombre de host sin nombre de dominio, como por ejemplo:* **`win80-x64_1`** *o* **`Doc3`**. El equipo recibe este tipo de nombre cuando se introduce el nombre de host `localhost` en sistemas Windows 8.

- *localhost*. En algunos casos el equipo recibe el nombre **`localhost`**.

Nota: si durante la instalación de un producto servidor de Altova en un equipo Windows el equipo se registra automáticamente con LicenseServer, el programa de instalación usará **`localhost`** como nombre del equipo.

Conectarse a LicenseServer con una VPN

Si un equipo cliente se conecta a su red con una red privada virtual (VPN por sus siglas en inglés), es posible que se le asigne de forma dinámica una dirección IP el equipo cliente, lo que llevaría a que se identificara a ese equipo como uno distinto cada vez que se conectara. Para saber cómo solucionar este problema, consulte el tema Información de red.

Solicitar una licencia de evaluación

Puede obtener gratis una licencia de evaluación de 30 días para todos los productos de Altova que estén registrados con LicenseServer:

1. Haga clic en el botón **Request Evaluation Licenses** situado en la parte superior del panel derecho. Este botón abre un cuadro de diálogo donde aparecen todos los productos de Altova del equipo cliente que están registrados con LicenseServer.
2. Marque la casilla de los productos para los que necesita una licencia de evaluación.
3. Rellene el formulario de la parte inferior del cuadro de diálogo y envíe la solicitud.

Altova le enviará un correo electrónico con las licencias de evaluación solicitadas. Si solicitó licencias de evaluación para productos servidor, el número de núcleos compatible con la licencia será el número que núcleos que exigía el producto cuando se realizó la solicitud. Guarde las licencias en el disco y cárguelas al repertorio de licencias.

Quitar productos del registro de LicenseServer

Los productos de Altova registrados con LicenseServer de cada equipo cliente se enumeran en el panel derecho de la pestaña **Client Management**. Justo debajo del panel aparece el icono **Quitar producto del registro** . Haga clic en este icono para quitar el producto del registro. Si el producto tiene asignada una licencia, la asignación expira nada más quitar el producto del registro. Para quitar todos los productos del registro, haga clic en el botón **Unregister Client and all products**, situado al principio del panel derecho.

Para volver a registrar un producto con LicenseServer:

- *Productos servidor:* abra la página **Opciones** en la interfaz web del producto servidor. Si el producto no cuenta con una interfaz web, abra una ventana del símbolo del sistema y use la interfaz de la línea de comandos para registrarlo. Para más información consulte Registrar FlowForce Server, Registrar MapForce Server, Registrar MobileTogether Server, Registrar StyleVision Server y Registrar RaptorXML(+XBRL) Server.
- *Productos de escritorio:* abra el cuadro de diálogo "Activación del software" (**Ayuda | Activación del software**), cambie al modo de activación a través de LicenseServer y en el campo *Altova LicenseServer* elija el servidor LicenseServer con el que desea registrar el producto. El producto se registrará y aparecerá en la pestaña **Client Management** del servidor LicenseServer, en la lista de productos registrados de dicho cliente.

Consulte la sección Asignar licencias a productos registrados para obtener más información.

11.8.3 Supervisión de clientes

La pestaña de supervisión de clientes **Client Monitoring** sirve para consultar toda la información disponible sobre el equipo cliente seleccionado. Esta es la información que ofrece la pestaña de supervisión de clientes:

Checked-out clients (Clientes con licencias extraídas)
El usuario de productos de escritorio de Altova (no de productos servidor) como XMLSpy o MapForce puede extraer una licencia que esté registrada con LicenseServer. Por ejemplo, puede extraer la licencia del producto de escritorio si sabe que el equipo estará sin conexión a Internet durante un período de tiempo. La licencia se puede extraer de LicenseServer durante este plazo de tiempo para que el usuario pueda seguir usando el producto de escritorio de Altova sin necesidad de ponerse en contacto con LicenseServer. En este apartado de la pestaña puede ver qué licencias y usuarios están extraídos, así como el plazo de extracción de la licencia.

Nota: la extracción de licencias la realiza el usuario final desde el cuadro de diálogo **"Activación del software"** (**Ayuda | Activación del software**) del producto de escritorio de Altova.

Running clients (Clientes que están en ejecución)
Se trata de una lista con todos los productos de Altova que están en ejecución en dicho cliente. Si hay varias instancias del mismo producto en ejecución, la lista incluye todas estas instancias.

 © 2018 Altova GmbH

Running Clients								
Product	Edition	Version	User	Address	State	Failover	Last seen (seconds ago)	
RaptorXML+XBRL Serv		2016 rel. 2	DOBRA	doc-aab	Running		8	
XMLSpy	Enterprise Editic	2016 rel. 3	adoc	doc-aab	Running		11	

Nota: los servidores LicenseServer de conmutación por error funcionan solamente con aplicaciones cliente versión 2015rel3 o superior (en el caso de MobileTogether Server, con la versión 1.5 o superior). Si la versión del cliente es anterior a éstas, recibirá una notificación.

Nota: si una licencia de un producto de escritorio de Altova está extraída, aparecerá en una sección diferente llamada *Checked-out Clients* (clientes extraídos) de la pestaña de supervisión de clientes **Client Monitoring**. Se indicará que la licencia está en ejecución y no podrá ser utilizada por ningún otro equipo de la red.

Iconos de la pestaña Client Monitoring

Mostrar licencias. Este icono aparece junto a cada instancia del producto y sirve para cambiar a la pestaña del repertorio de licencias (**License Pool**), donde se resalta la licencia de la instancia del producto seleccionada.

Gestionar cliente. Este icono aparece junto a cada instancia del producto y sirve para cambiar a la pestaña de gestión de clientes (**Client Management**), donde se resalta el cliente de la instancia del producto seleccionada.

11.8.4 Opciones de configuración

Temas de este apartado:

- Configuración de servidores LicenseServer de conmutación por error
- Configuración de red
- Correos de alerta
- Información sobre cómo recibir e implementar licencias de evaluación

En la pestaña de configuración **Settings** (*imagen siguiente*) puede definir las opciones de configuración del servidor LicenseServer:

- Un tiempo de espera hasta que LicenseServer se cierre. El cierre suele implementarse para tareas de mantenimiento del servidor de licencias. El tiempo de espera hasta el cierre puede utilizarse para guardar el trabajo en los clientes en los que se ejecutan productos de escritorio de Altova. El tiempo seleccionado será el límite de tiempo máximo hasta el cierre. LicenseServer se cerrará antes si no está conectado a ningún

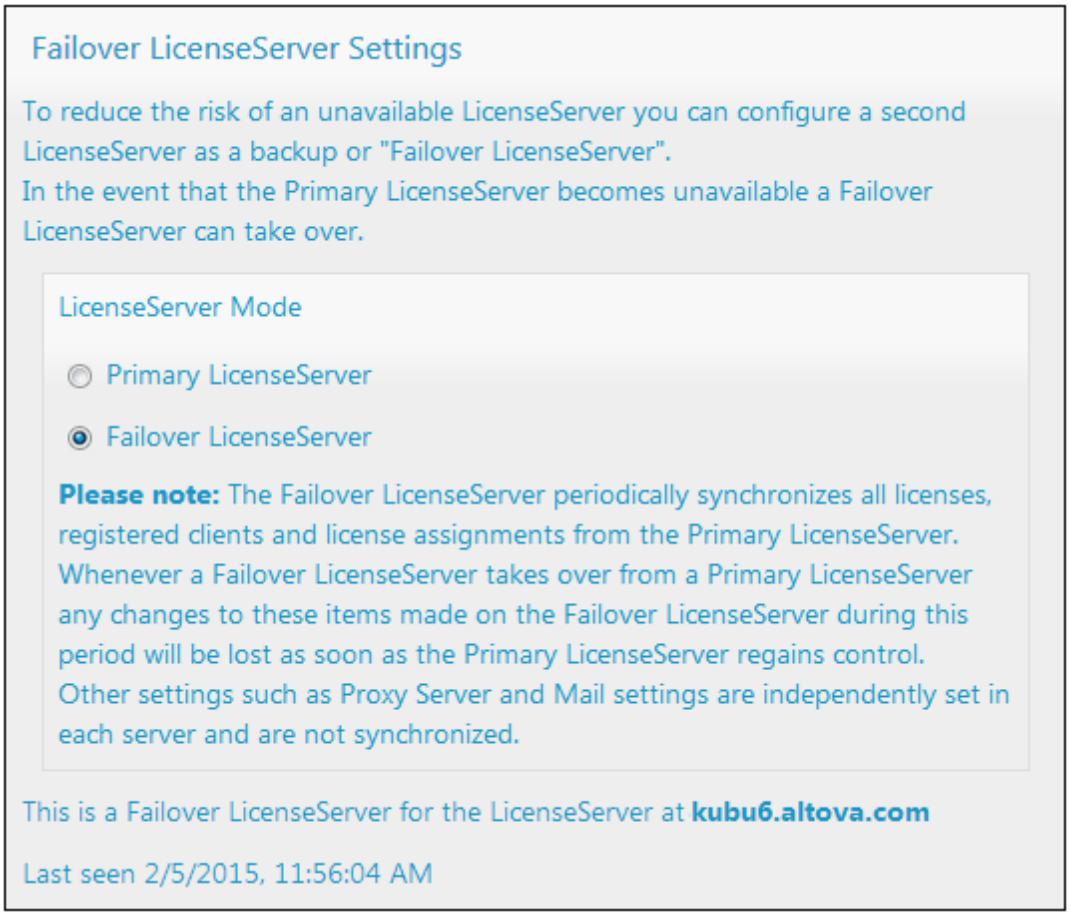
cliente en el que se ejecuten productos de escritorio. El tiempo límite hasta el cierre comienza cuando se hace clic en el botón **Shutdown**. Para cancelar el cierre haga clic en el botón **Abort Shutdown**. Si quiere que los clientes puedan ejecutarse mientras está cerrado LicenseServer, configure un **servidor LicenseServer de conmutación por error**.

- Puede configurar otro servidor LicenseServer para que releve al servidor principal si éste deja de estar disponible. Este otro servidor LicenseServer recibe el nombre de **servidor de conmutación por error (Failover LicenseServer)**. En este apartado encontrará información sobre configurar este tipo de servidor.
- La contraseña para iniciar sesión en LicenseServer. Escriba la contraseña y haga clic en el botón **Change Password** (*Cambiar contraseña*).
- Puede probar la conexión con Altova haciendo clic en el botón **Test Connection to Altova**. Recuerde que debe guardar las opciones de configuración nuevas (haciendo clic en el botón **Save** situado al final del panel) antes de probar la conexión. Mientras se prueba la conexión el botón **Test Connection to Altova** se deshabilita y se vuelve a habilitar una vez finalizada la prueba de conexión.
- Opciones de red para la página de configuración web (*Web UI*), para el servidor proxy utilizado para conectarse a Internet si lo hay (*Proxy Server*) y para LicenseServer (*License Service*). Estas opciones se describen más adelante.
- Opciones del servidor de correo y el destinatario de correos de alerta a quien avisar cuando ocurra algo importante en LicenseServer. Estas opciones se describen más adelante.
- Cuando termine de cambiar las opciones de configuración, haga clic en el botón **Save** situado al final del panel. Recuerde que los cambios no surten efecto hasta que se guardan con este botón.

Configuración de servidores LicenseServer de conmutación por error

Puede configurar otro servidor LicenseServer para que releve al servidor LicenseServer principal si éste deja de estar disponible. Este otro servidor recibe el nombre de servidor de conmutación por error (**Failover LicenseServer**).

 © 2018 Altova GmbH



Siga estos pasos para configurar un servidor LicenseServer como servidor de conmutación por error:

1. Instale LicenseServer tal y como se explica en la sección *Instalación*.
2. En la pestaña ***Settings*** cambie el modo de LicenseServer por *Failover LicenseServer* seleccionando el botón de opción correspondiente (*imagen anterior*).
3. Aparece el cuadro de diálogo "Find Primary LicenseServer" (imagen siguiente) donde puede buscar el servidor LicenseServer principal que quiere asegurar con este servidor de conmutación por error. Esto se puede hacer de dos maneras: (i) haciendo clic en **Search for LicenseServers** y seleccionando el servidor que quiere asegurar en la lista de servidores encontrados del cuadro combinado o (ii) haciendo clic en **Manually Enter Address** para introducir a mano la dirección del servidor LicenseServer que quiere asegurar. Cuando haya terminado de especificar el servidor LicenseServer principal, haga clic en **Connect to Primary LicenseServer** para establecer la conexión.

4. Ahora aparece un cuadro de diálogo de confirmación donde debe confirmar si quiere configurar el servidor LicenseServer actual como servidor de conmutación por error (del servidor principal que acaba de seleccionar). Haga clic en **Sí** para confirmar. Recuerde que al hacer clic en **Sí** se eliminarán las licencias instaladas y los clientes registrados en el servidor LicenseServer actual.

Cuando el servidor de conmutación por error esté configurado, tanto el servidor LicenseServer principal como el de conmutación por error incluirán notificaciones en la parte superior de la página indicando cuál es el modo actual del servidor. Por ejemplo, en las imágenes que aparecen a continuación puede ver un servidor de conmutación por error (Failover LicenseServer) y después el servidor principal (Primary LicenseServer).

Debe tener en cuenta estos aspectos:

- Una vez configurado, el servidor de conmutación por error sincroniza de forma periódica todas sus licencias, todos sus clientes registrados y todos sus acuerdos de licencia con los del servidor principal. Si el servidor principal deja de estar disponible, el servidor de conmutación por error desempeñará el papel de LicenseServer. Cuando vuelva a estar disponible, el servidor principal relevará al servidor de conmutación por error. Los cambios realizados en el servidor de conmutación por error en este plazo de tiempo se perderán cuando el servidor principal recupere el control.
- El servidor de conmutación por error solamente ofrecerá licencias para clientes que tengan la versión v2015r3 o superior (o en el caso de Altova MobileTogether Server, la versión 1.5 o superior). Los clientes que tengan versiones anteriores a éstas se marcarán en la pestaña de supervisión de clientes **Client Monitoring** del servidor principal (*imagen siguiente*). Por eso, si desea usar la función de servidor de conmutación por error, recomendamos que actualice sus aplicaciones con la versión v2015r3 o superior (o en el caso de MobileTogether Server, la versión 1.5 o superior).

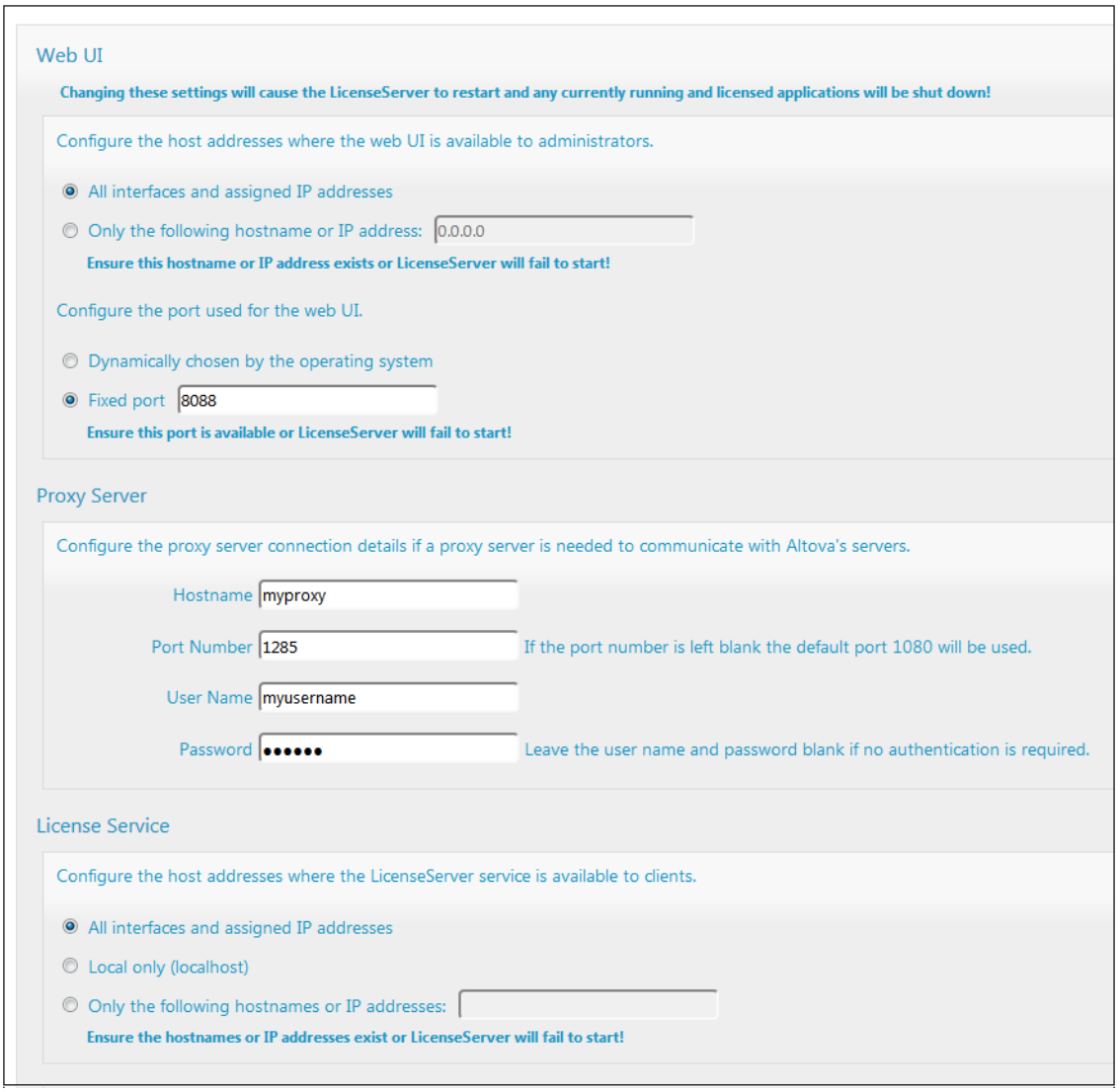
Extracción de licencias

Puede seleccionar si permite la extracción de licencias de productos de escritorios del repertorio y su almacenaje en el equipo en el que está instalado el producto de escritorio de Altova. Si permite la extracción de licencias, el cliente que haya adquirido una licencia del servidor LicenseServer podrá extraerla y funcionar sin supervisión durante el plazo de tiempo que se especifique aquí. El plazo de tiempo puede ser de 30 días como máximo. Durante ese plazo de tiempo el cliente podrá seguir usando el producto de escritorio aunque no tenga acceso a LicenseServer. El cliente puede extraer la licencia con el comando **Ayuda | Activación del software** de la aplicación de Altova.

Una vez extraída, la licencia pasará a la sección *Checked-out Clients* (clientes extraídos) de la pestaña de supervisión de clientes **Client Monitoring**. LicenseServer entenderá que la licencia está en uso y no podrá ser utilizada por ningún otro cliente de la red.

Configuración de red

El administrador puede especificar los puntos de acceso de red para la página de configuración web y para el servidor LicenseServer:



- *Web UI (interfaz web):* las direcciones IP permitidas pueden ser direcciones IP e interfaces del equipo o una dirección fija. Los puertos se pueden calcular de forma dinámica o fija. Esto permite un amplio abanico de opciones de configuración dirección IP:puerto. El puerto predeterminado es **8088**.

- *Proxy Server (disponible a partir de la versión 1.3):* si usa un servidor proxy para conectarse a Internet, los datos de este servidor proxy deben introducirse bajo este grupo de opciones (*imagen anterior*). Si no usa un servidor proxy, puede dejar en blanco estas opciones. Para configurar LicenseServer para el servidor proxy, escriba el nombre de host del servidor proxy y un número de puerto, si es necesario. Si el servidor proxy no requiere autenticación del usuario, puede dejar estos dos campos en blanco.

- *License Service (servicio de licencias):* el equipo donde está instalado LicenseServer se puede conectar a varias redes por medio de varias interfaces de red. En cada una de estas redes, el equipo de LicenseServer se conoce por su nombre de host y dirección IP.

 © 2018 Altova GmbH

En este grupo de opciones puede configurar en qué redes estará disponible el servicio de licencias. La opción localhost permite que el servicio esté disponible en el equipo local solamente. Si incluye varios nombres de host y varias direcciones IP, sepárelos con comas y sin espacios (por ejemplo: `nombrehost1,direcciónIP1,nombrehost2`). El número de puerto del servicio está fijado en **35355**.

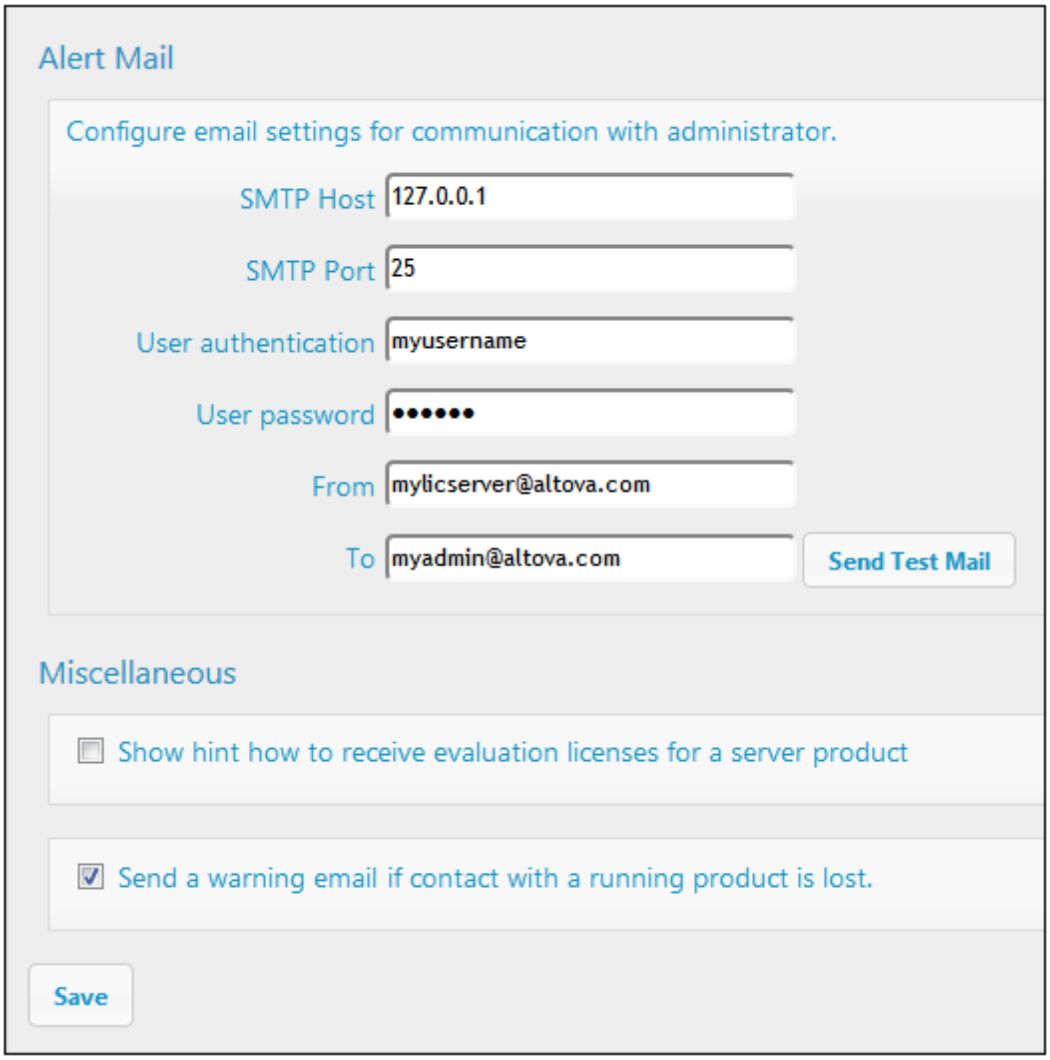
La configuración predeterminada permite un acceso no restringido a LicenseServer y a su página de configuración de forma predeterminada desde las redes a las que está conectado LicenseServer. Si quiere restringir el acceso a LicenseServer o a su página de configuración, defina las opciones correspondientes en esta pestaña y haga clic en **Save** para guardar.

Realice una prueba de conexión (*ver más arriba*) para comprobar si funcionan las opciones definidas.

Correos de alerta

Pueden enviarse correos de alerta a las direcciones de correo designadas cuando ocurren determinados eventos importantes en LicenseServer. Por ejemplo: imagine que Altova LicenseServer necesita estar conectado al servido maestro `altova.com`. Si la conexión se interrumpe más de 24 x 5 horas (es decir, 5 días), LicenseServer no permitirá licencias. Como resultado, puede que las sesiones de trabajo con productos Altova registrados con LicenseServer se interrumpan.

Si quiere puede configurar LicenseServer para que envíe un correo de alerta al administrador cuando la conexión se interrumpa. En el grupo de opciones *Alert Mail* (*imagen anterior*) puede configurar los correos de alerta y la dirección de correo del administrador.



Los campos *SMTP Host* y *SMTP Port* son los datos de acceso del servidor de correo electrónico desde el que se enviarán los correos de alerta. Los campos *User Authentication* y *User Password* son las credenciales de usuario para acceder al servidor de correo. El campo *From* toma la dirección de la cuenta de correo electrónico desde la que se enviará el correo. El campo *To* es la dirección de correo electrónico del destinatario.

Cuando termine haga clic en **Save** para guardar los cambios. A partir de ese momento se enviarán correos de alerta a las direcciones especificadas cuando se produzcan eventos relevantes, como cuando se pierde la conexión con `altova.com`, por ejemplo. No olvide que estos eventos también se registran en la pestaña **Messages**.

Opciones varias

Show hints for receiving and deploying evaluation licenses

Si marca esta casilla (*Mostrar información sobre cómo recibir e implementar licencias de*

evaluación), al principio de la página de configuración aparecerán instrucciones para recibir e implementar licencias de evaluación.

Send a warning email if contact with a running product is lost
Si marca esta casilla (*Enviar correo de alerta si se pierde el contacto con un producto en ejecución*), se enviará un correo de alerta desde la dirección From a la dirección To cuando se pierda la conexión con un producto que tiene asignada una licencia y que está en ejecución.

11.8.5 Mensajes, Cerrar sesión

La pestaña **Messages** muestra todos los mensajes relacionados con las licencias del repertorio de LicenseServer. Cada mensaje se puede eliminar haciendo clic en el correspondiente botón **Eliminar**.

La pestaña **Log out** es en realidad un botón de cierre de sesión. Al hacer clic en **Log out** se cierra inmediatamente la sesión y aparece la página de inicio de sesión.

11.9 Restaurar la contraseña

Si olvida su contraseña de LicenseServer, puede usar el comando **`passwordreset`** desde la línea de comandos para restaurar la contraseña predeterminada `default`.

1. Abra una ventana de la línea de comandos.
2. Cambie al directorio donde está instalado el ejecutable o el paquete de la aplicación LicenseServer.
3. Introduzca el comando: **`licenseserver passwordreset`**
   Esto restaura la contraseña de administrador predeterminada `default`.
4. Ahora puede iniciar sesión en la interfaz web del administrador con `default`.

 © 2018 Altova GmbH

Índice

.

.NET Framework API, 288

A

Actualizaciones, 504
Altova LicenseServer,
ver LicenseServer, 500
Altova ServiceController, 512
Anular registro de productos y clientes, 551
Archivo de configuración JSON,
para el módulo Python de RaptorXMLXBRLServer, 285
Asignación de licencias en Linux, 25
Asignación de licencias en macOS, 32
Asignación de licencias en Windows, 17
Asignar licencias, 537

B

Biblioteca Python,
de RaptorXML+XBRL Server, 285

C

Cargar licencias, 523, 544
Catálogos, 35
Catálogos XML, 35
Cerrar sesión, 565
Comando de ayuda de la ILC, 213
Comandos de licencias en la ILC, 214
Comandos XQuery, 145
Comandos XSLT, 130
Comprobar el formato, 71
Configuración, 557
Configuración del servidor, 247
Contraseña,
predeterminada al iniciar LicenseServer, 515
Contraseña predeterminada, 515
Correos de alerta, 557

D

Documentos XQuery,
validación, 161
Documentos XSLT,
validación, 138

E

Ejecución de XQuery, 145
Extensiones de Altova,
funciones para gráficos, 414

F

FlowForce Server,
registrar con LicenseServer, 528
Funciones de extensión .NET,
campos de instancia, 493
campos estáticos, 492
constructores, 491
conversiones de tipos de datos, 494, 495
métodos de instancia, 493
métodos estáticos, 492
para XSLT y XQuery, 489
resumen, 489
tipos de datos .NET en XPath/XQuery, 495
tipos de datos XPath/XQuery en .NET, 494
Funciones de extensión .NET para XSLT y XQuery,
ver Funciones de extensión .NET, 489
Funciones de extensión en scripts MSXSL,
msxsl:script, 496
Funciones de extensión Java,
archivos de clases definidos por el usuario, 481
archivos JAR definidos por el usuario, 484
campos de instancia, 486
campos estáticos, 485
constructores, 485
conversiones de tipos de datos, 487, 488

Funciones de extensión Java,
métodos de instancia, 486
métodos estáticos, 485
para XSLT y XQuery, 479
resumen, 479
tipos de datos Java en XPath/XQuery, 488
tipos de datos XPath/XQuery en Java, 487
Funciones de extensión Java para XSLT y XQuery,
ver Funciones de extensión Java, 479
Funciones de extensión para XSLT y XQuery, 479
Funciones para gráficos,
ejemplo, 473
estructura de datos de gráficos, 468
fragmento de código, 463

G

Gestión de clientes (página), 551

I

Información de red, 502
Instalación, 14
Linux, 22
macOS, 29
Windows, 15
Instalación en Linux, 22
Instalación en macOS, 29
Instalación en Windows, 15
Instalar el módulo Python de RaptorXMLXBRLServer, 285
Interfaces,
resumen, 5
Interfaz .NET, 5
Interfaz COM, 5
Interfaz del administrador, 544
Interfaz HTTP, 5, 242
configuración del servidor, 247
preparar el servidor, 244
problemas de seguridad, 43
solicitudes cliente, 257
Interfaz Java, 5
Interfaz Python, 5

L

Licencias,
asignar, 537, 551
cargar, 523, 544
gestionar, 551
Licencias de evaluación, 551
LicenseServer,
iniciar, 514
instalación en Linux, 507
instalación en macOS, 510
instalación en Windows, 505
instrucciones para asignar licencias, 513
interfaz con, 544
opciones de configuración, 557
página de configuración, 544
registrar FlowForce Server con, 528
registrar MapForce Server con, 532
registrar productos de escritorio, 527
registrar StyleVision Server con, 536
Línea de comandos,
opciones, 221
resumen de uso, 46
y XQuery, 145
Linux,
asignación de licencias en, 25
instalación en, 22
Localización, 218

M

macOS,
asignación de licencias en, 32
instalación en, 29
MapForce Server,
registrar con LicenseServer, 532
Mensajes, 565
Módulo Python,
de RaptorXML+XBRL Server, 285

O

Opciones de configuración de red, 557

P

Página de configuración, 544
abrir en Linux, 518
abrir en macOS, 520
abrir en Windows, 515
URL de, 515
URL de (Linux), 518
URL de (macOS), 520
Página de configuración de LicenseServer,
(ver Página de configuración), 518, 520
ver Página de configuración, 515
Pestaña Gestión de clientes, 537
pip (comando), 285
Problemas de seguridad, 43
Productos de escritorio,
registro con LicenseServer, 527
Python,
problemas de seguridad, 43
Python API, 282

R

RaptorXML,
características, 8
ediciones e interfaces, 5
especificaciones compatibles, 10
interfaces con COM, Java y .NET, 5
interfaz de la línea de comandos, 5
interfaz HTTP, 5
interfaz Python, 5
introducción, 3
requisitos del sistema, 7
Recursos globales, 41
Registrar FlowForce Server con LicenseServer, 528
Registrar MapForce Server con LicenseServer, 532
Registrar productos de escritorio con LicenseServer, 527
Registrar StyleVision Server con LicenseServer, 536
Repertorio de licencias, 523
Repertorio de licencias (página), 544
RootCatalog.xml, 285

S

Scripts en XSLT/XQuery,
ver Funciones de extensión, 479
ServiceController, 512
StyleVision Server,
registrar con LicenseServer, 536
Supervisión de clientes (página), 556

T

Transformación XSLT, 130

V

Validación,
de cualquier documento, 210
de documentos XQuery, 161
de documentos XSLT, 138
de DTD, 61
de instancias XBRL, 81
de instancias XBRL y taxonomías, 81
de instancias XML con DTD, 50
de instancias XML con XSD, 54
de taxonomías XBRL, 110
de XSD, 64
Validación XBRL,
ver Validación, 81

W

Windows,
asignación de licencias en, 17
instalación en, 15

X

XQuery,
funciones de extensión, 479
XSLT,
funciones de extensión, 479